Surv Geophys (2012) 33:5–27
DOI 10.1007/s10712-011-9122-6

Three-Dimensional Magnetotelluric Inversion: An Introductory Guide for Developers and Users

Weerachai Siripunvaraporn

Received: 23 November 2010 / Accepted: 30 March 2011/ / Published online: 28 May 2011

Abstract In the last few decades, the demand for three-dimensional (3-D) inversions for magnetotelluric data has significantly driven the progress of 3-D codes. There are currently a lot of new 3-D inversion and forward modeling codes. Some, such as the WSINV3DMT code of the author, are available to the academic community. The goal of this paper is to summarize all the important issues involving 3-D inversions. It aims to show how inversion works and how to use it properly. In this paper, I start by describing several good reasons for doing 3-D inversion instead of 2-D inversion. The main algorithms for 3-D inversion are reviewed along with some comparisons of their advantages and disadvantages. These algorithms are the classical Occam's inversion, the data space Occam's inversion, the Gauss–Newton method, the Gauss–Newton with the conjugate gradient method, the non-linear conjugate gradient method, and the quasi-Newton method. Other variants are based on these main algorithms. Forward modeling, sensitivity calculations, model covariance and its parallel implementation are all necessary components of inversions and are reviewed here. Rules of thumb for performing 3-D inversion are proposed for the benefit of the 3-D inversion novice. Problems regarding 3-D inversions are discussed along with suggested topics for future research for the developers of the next decades.

Keywords 3-D Inversion algorithms · Magnetotelluric · Electromagnetic induction

1 Introduction

Many three-dimensional (3-D) magnetotelluric (MT) inversion algorithms have been developed in the past few decades (e.g. Smith and Booker 1991; Mackie and Madden

W. Siripunvaraporn (✉)
Department of Physics, Faculty of Science, Mahidol University,
Rama 6 Rd., Rachatawee, Bangkok 10400, Thailand
e-mail: wsiripun@gmail.com

W. Siripunvaraporn
ThEP, Commission of Higher Education, 328 Si Ayuthaya Road, Bangkok 10400, Thailand

1993; Newman and Alumbaugh 2000; Zhdanov et al. 2000; Sasaki 2001, 2004; Zhdanov 2002; Siripunvaraporn et al. 2004, 2005a; Sasaki and Meju 2006; Gribenko and Zhdanov 2007; Mackie and Watts 2007; Han et al. 2008; Lin et al. 2008, 2009; Farquharson and Craven 2008; Avdeev and Avdeeva 2009; Siripunvaraporn and Egbert 2009; Gribenko et al. 2010; Siripunvaraporn and Sarakorn 2011). The major factors driving progress in the development of 3-D algorithms are the ambiguity of the two-dimensional (2-D) interpretation and the increasing number of 3-D MT data acquisitions.

3-D MT inversions have been used in various studies such as ore deposit exploration (e.g., Tuncer et al. 2006; Farquharson and Craven 2008; Türkoğlu et al. 2009; Xiao et al. 2010; Goldak and Kosteniuk 2010), waste characterization (e.g., Newman et al. 2003), tectonic studies (e.g., Uyeshima 2007; Patro and Egbert 2008), volcano studies (e.g., Spichak et al. 2007; Heise et al. 2008, 2010; Jones et al. 2008; Hill et al. 2009; Ingham et al. 2009), hydrocarbon exploration (e.g., Mackie and Watts 2007; Hautot and Tarits 2009; Zhanxiang et al. 2010) and geothermal studies (e.g., Han et al. 2008; Newman et al. 2008; Árnason et al. 2010; Cumming and Mackie 2010). Improvements in 3-D inversion are likely to continue as long as MT data are still being acquired. New technologies, theories, methods, and 'new' people with a support of 'old' people are all major driving forces for the bright future of 3-D inversion.

In this paper, I start with a discussion of the importance of 3-D inversions. This section is aimed at both users and developers. I then review the many different 3-D inversion algorithms. Other necessary components for 3-D inversions are given next. Novice or inexperienced developers can use this section as a guide or recipe for developing their own codes. At the same time, it is a good idea for users to learn how the inversion and other components work. Often, many users make simple mistakes when running 3-D inversions which could lead to a major fault in interpretation. The next section is therefore designed to introduce the strategy for carrying out 3-D inversion. Problems and future suggested research will be discussed in the final section to motivate and encourage new developments. Previous 3-D electromagnetic modeling and inversion reviews can be found in Avdeev (2005), Abubakar et al. (2009) and Borner (2010).

2 When is 3-D Inversion Useful? (Advantages of 3-D Inversion Over 2-D Inversion)

Because the Earth is 3-D, a 2-D Earth model cannot be used to explain or represent the 3-D Earth. This is a simple and obvious reason why one needs 3-D inversion. Other reasons are described in more detail in this section.

For several decades, MT acquisitions have usually been conducted along a profile or several profiles parallel to each other (e.g., Unsworth et al. 2000; Tuncer et al. 2006; Newman et al. 2008). Prior to 2-D inversion, dimensionality analyses are carried out to determine which data are consistent with a 2-D interpretation. These analyses include skew analysis (Swift 1967; Vozoff 1972), phase-sensitive skew analysis (Bahr 1991), the Groom and Bailey decomposition (Groom and Bailey 1989), tensor decomposition (Chave and Smith 1994), rotational invariant analysis (Weaver et al. 2000; Marti et al. 2009), strike decomposition (McNeice and Jones 2001), the phase tensor (Caldwell et al. 2004), and the Mohr circle (Lilley and Weaver 2010). A good review of 2-D techniques including modeling, dimensional analysis and interpretation is summarized in Ogawa (2002).

Although MT profile data often demonstrate the influence of 3-D effects or show deviations from being purely 2-D, 2-D assumptions are usually allowed (e.g., Becken et al. 2008b; Ingham et al. 2009; Hill et al. 2009; among many others). 2-D inversions are then

performed to yield 2-D cross-sectional models for profile interpretation. The dangers of 2-D inversions being influenced by 3-D structures are demonstrated with synthetic data in Siripunvaraporn et al. (2005b) and Ledo (2006) and with real data in Simpson and Bahr (2005) and Newman et al. (2003). All studies indicate that if the data contains 3-D structures, 2-D inversion can mislead an interpretation. Ledo's analysis suggests that it is still possible to conduct 2-D interpretation on some modes (either TM or TE). This depends significantly on the position of the 3-D structure with respect to the regional 2-D strike direction. A separation of data into TM and TE modes by inclusion of the vertical magnetic transfer function to allow 2-D inversions has been demonstrated with synthetic data (Becken et al. 2008a). However, in practice, techniques similar to Ledo's analysis and mode separation are not easy to justify unless the 3-D structures are already known.

The ambiguity of the data can be overcome by using 3-D inversion programs such as the WSINV3DMT code (Siripunvaraporn et al. 2005a). With 3-D inversion it is not necessary to make an assumption about the strike direction. 3-D inversion can immediately be applied to the data after data processing and noise removal. This is a clear advantage of 3-D inversion over 2-D inversion. Although dimensionality analysis is not strictly necessary, it is still recommended in order to determine the nature of the observed data.

In the past, most data acquisitions were conducted using one or more 2-D profiles. Instead of using 2-D inversion, the 2-D profile data can be inverted with 3-D inversion code. This is demonstrated in Siripunvaraporn et al. (2005b) for synthetic data and in Xiao et al. (2010) for real field data. The advantage of using 3-D inversion on 2-D profile data is that it can recover anomalies correctly as demonstrated in Newman et al. (2003) with real data. With 3-D inversion, where the number of degrees of freedom is high, the data can force the inversion to put the 3-D effects from outside the profiles to where they belong. These 3-D effects are in the form of structures, although their size, shape, depth and location cannot be determined exactly. This is in contrast to 2-D inversion where the number of degrees of freedom is significantly smaller. To accommodate the 3-D effects required by the data, 2-D inversion can only insert the structure beneath the profiles which could lead to misinterpretation (Siripunvaraporn et al. 2005b).

Due to many limitations such as inaccessibility by road, presence of large water reservoirs or rough topography, conducting MT experiments along a profile may not be possible. MT stations can only be sited within a broad corridor about the profile line, and must be projected onto the profile (e.g., Ingham et al. 2009; Árnason et al. 2010; Heise et al. 2010). 2-D inversion is possible only if the data are projected on the same profile (e.g., Hill et al. 2009; Ingham et al. 2009). With 3-D inversion, data from the scattered stations can be easily inverted without data projection. This could help avoid errors in estimation of size, shape and location of the structures. 3-D inversion is helping to open a new era in data acquisition where sites do not have to lie in a straight line.

As with 2-D inversion, 3-D inversion is usually used to construct the model that fits the inductive responses produced by the regional conductivity structure. Another beauty of 3-D inversion is that it can be used to fit the inductive effects due to localized near-surface structures both directly beneath and outside the station. It can also be used to obtain the galvanic distortion or the static shift (see Newman et al. 2003; Sasaki 2004; Sasaki and Meju 2006). This can be accomplished by finely discretizing the model near the surface.

Recently, phases of over 90° have been mentioned in many publications (e.g., Ichihara et al. 2010; Lilley and Weaver 2010). Phases over 90° result from the L-shape structure where the current is higher at the corners (Ichihara and Mogi 2009). 2-D inversion cannot be used to fit phases over 90°. Other explanations must therefore be sought out. One of them may be anisotropy (Chen and Weckmann 2010). Ichihara et al. (2010) demonstrated

that 3-D inversion can recover the L-shape structure that produces phases of over 90°. This is another advantage of 3-D inversion over 2-D inversion.

The drawback of 3-D inversion is the large computational cost in terms of both memory and CPU time. This could discourage many from using 3-D inversion and in favor of 2-D interpretation. With new technology, new theories and methods, these large requirements will be reduced. Other current problems could be the inaccessibility of many codes. Again, these problems are likely to be overcome in the not too distant future.

3 Mathematical Review of Magnetotelluric Inversion Algorithms

In this section, I explain how the main algorithms for 3-D inversions work. These algorithms are the classical Occam's inversion, the data space Occam's inversion, the Gauss–Newton (GN) method, the Gauss–Newton with the conjugate gradient (GN-CG) method, the quasi-Newton (QN) method, and the non-linear conjugate gradient (NLCG) method. Many other algorithms are based on these algorithms with some modifications or a combination of these techniques and will not be described in detail here.

All inversion algorithms share the same goal of finding the 'best' model that is geologically interpretable and fitting the data to an acceptable level. Two main functionals for magnetotelluric data are normally minimized. The first functional is the unconstrained functional,

$$U(\mathbf{m},\lambda)=\lambda^{-1}\left\{(\mathbf{d}-\mathbf{F}[\mathbf{m}])^{T}\mathbf{C}_{\mathbf{d}}^{-1}(\mathbf{d}-\mathbf{F}[\mathbf{m}])-\chi^{2}\right\}+(\mathbf{m}-\mathbf{m}_{0})^{T}\mathbf{C}_{\mathbf{m}}^{-1}(\mathbf{m}-\mathbf{m}_{0}), \quad (1)$$

where $\mathbf{m}$ is the resistivity model of dimension M, $\mathbf{m_0}$ is the prior model, $\mathbf{C_m}$ is the model covariance matrix, $\mathbf{d}$ is the observed data with dimension N, $\mathbf{F}[\mathbf{m}]$ is the forward model response, $\mathbf{C_d}$ is the data covariance matrix, χ^2 is a desired level of misfit, and λ^{-1} is a Lagrange multiplier. This functional is used mostly in Occam's inversions (Constable et al. 1987; Degroot-Hedlin and Constable 1990; Siripunvaraporn and Egbert 2000, 2009; Siripunvaraporn et al. 2004, 2005a). In Occam's algorithm, λ is not fixed but varies from iteration to iteration.

Another unconstrained functional is $W(\mathbf{m})$ which is sometimes called the objective functional or the penalty functional depending on where it is used. This functional has the form

$$W(\mathbf{m})=\lambda^{-1}(\mathbf{d}-\mathbf{F}[\mathbf{m}])^{T}\mathbf{C}_{\mathbf{d}}^{-1}(\mathbf{d}-\mathbf{F}[\mathbf{m}])+(\mathbf{m}-\mathbf{m}_{0})^{T}\mathbf{C}_{\mathbf{m}}^{-1}(\mathbf{m}-\mathbf{m}_{0}), \quad (2)$$

where λ is a trade-off parameter controlling whether to heavily minimize the data misfit or the model norm. For large λ, the data misfit is less important and therefore the model norm is minimized to produce a smoother model. For small λ, the inversion tends to fit the data better. This is closer to a least-squares inverse problem and often produces a rougher model. The second functional is used in most algorithms, for example, in NLCG (e.g., Newman and Alumbaugh 2000; Rodi and Mackie 2001; Mackie and Watts 2007; Commer and Newman 2009; Zhanxiang et al. 2010), in GN (e.g., Haber et al. 2000, Sasaki 2001, 2004, Gunther et al. 2006), in QN (e.g., Haber 2005; Avdeev and Avdeeva 2009), and in GN-CG (e.g., Mackie and Madden 1993; Newman and Alumbaugh 1997; Gunther et al. 2006; Siripunvaraporn and Egbert 2007). In these algorithms, λ must be pre-selected and its value remains fixed. Recently, many algorithms practically do not fix λ but start the inversion with large λ before decreasing to a smaller value to decrease the computational

time (e.g., Newman and Hoversten 2000; Haber et al. 2000; Kelbert et al. 2008; Avdeev and Avdeeva 2009; and Siripunvaraporn and Sarakorn 2011).

To achieve the minimum structure of $U(\mathrm{m},\lambda)$ with $W(\mathrm{m})$, all of these algorithms (GN, GN-CG, QN and NLCG) must be separately run with different values of pre-selected λ. Each run should be terminated when the misfit reaches the desired level. The model norm is then computed and compared to other norms from other runs. In this case, the minimum model norm is then equivalent to the output of (1) if using the same definition of the norm. This was demonstrated in Siripunvaraporn and Sarakorn (2011) and is shown again in Fig. 1. However, if the objective of the inversion is just to minimize the objective function (2), there is no need to terminate the run when the misfit reaches the desired level.

Other 3-D inversions that should be mentioned here, but are not elaborated upon further, are the Bayesian Statistics method (Spichak 1999), the neural network method (Spichak and Popova 2000), the genetic algorithm inversion (Toh and Honma 2008), and also the imaging technique (Szarka et al. 2006).

3.1 Occam's Inversion (OCCAM)

Occam's inversion searches for the stationary point of (1) through the penalty functional (2). This is because, for a fixed λ, $\partial U/\partial \mathbf{m} = \partial W/\partial \mathbf{m}$. To obtain the stationary point of (1), several λ must be used in $\partial W/\partial \mathbf{m}$. The OCCAM process can be divided into two phases. Phase I is to bring the RMS misfit down to the level of χ^2 by varying λ. This phase makes the first term on the right hand side of (1) equal to zero. Phase II is to minimize the model norm, the second term on the right hand side of (1). This is accomplished by again varying λ to find the model with the smallest norm while keeping the RMS misfit at the χ^2 level. The algorithm used to search for the best λ can be a simple scheme like the bisection

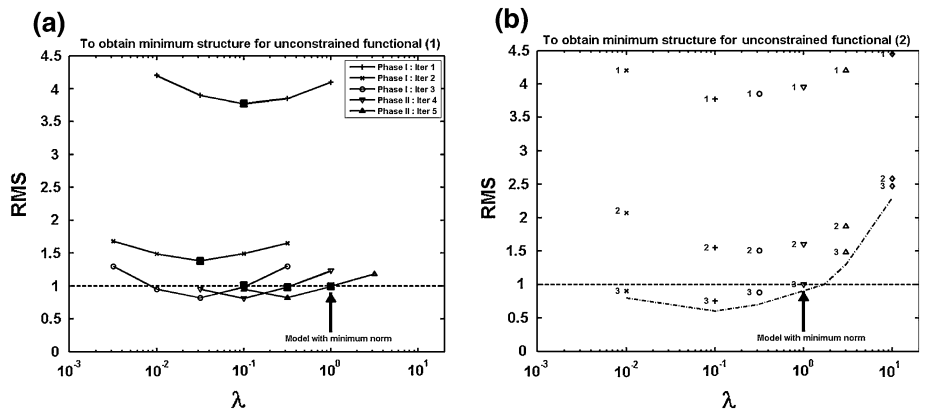

Fig. 1 Plots to display the equivalent of (*1*) via (*2*) (after Siripunvaraporn and Sarakorn 2011). **a** In Occam's inversion where (*1*) is used, λ is automatically searched and varied from iteration to iteration. In Phase I, the inversion seeks the model that produces the minimum RMS misfit. On achieving the target misfit, Phase II of the inversion seeks the model with minimum norm. *Solid squares mark* the values of λ used to calculate the sensitivity matrix for the next iterations. **b** In other inversion schemes where (*2*) is used, λ is pre-selected. To achieve the smoothest structure goal, each λ is fixed and run separately. The model that has the minimum norm with the target misfit is selected as an equivalent model to (*1*). Integers next to symbols indicate the number of iterations

search (see Press et al. 1992; Siripunvaraporn and Egbert 2000). An example plot of RMS versus λ for both phases is shown in Fig. 1a.

Phase II is necessary to get rid of the spurious structures due to over-fitting the data. Usually, χ^2 is set to 1 RMS but frequently in real applications this can not be attained (e.g., Tuncer et al. 2006). By either decreasing the error bars or increasing the χ^2 level, we would be able to obtain the minimum norm model. The model norm is usually controlled by the model covariance $\mathbf{C_m}$ in Siripunvaraporn et al. (2005a) or the roughness operator in many others (e.g., Constable et al. 1987; Rodi and Mackie 2001). Both will be discussed in the next section.

Currently, there are two variants of Occam's inversion; the original or model space Occam's inversion and the data space Occam's inversion. If both schemes use the same parameters (same λ and $\mathbf{C}_m$), the results will be identical.

3.1.1 The Model Space Occam's Inversion

The OCCAM algorithm starts with the linearization of the model response $\mathbf{F}[\mathbf{m}]$ using the first order Taylor's series expansion, i.e., $\mathbf{F}[\mathbf{m}_{k+1}] = \mathbf{F}[\mathbf{m}_k] + \mathbf{J}_k(\mathbf{m}_{k+1} - \mathbf{m}_k)$, where $\mathbf{J}_k = \partial\mathbf{F}/\partial\mathbf{m}_k$ is the $N \times M$ sensitivity matrix calculated with respect to the model at the k^{th} iteration, $\mathbf{m}_k$. By differentiating the linearized W with respect to the model $\mathbf{m}$ and setting it to zero, i.e., $\partial W/\partial\mathbf{m} = 0$, the following iterative sequence of linear equations is obtained:

$$\mathbf{m}_{k+1} - \mathbf{m}_0 = \left[\lambda\mathbf{C}_\mathbf{m}^{-1} + \mathbf{J}_k^T\mathbf{C}_d^{-1}\mathbf{J}_k\right]^{-1}\mathbf{J}_k^T\mathbf{C}_d^{-1}\tilde{\mathbf{d}}_k, \tag{3}$$

where $\tilde{\mathbf{d}}_k = \mathbf{d} - \mathbf{F}[\mathbf{m}_k] + \mathbf{J}_k(\mathbf{m}_k - \mathbf{m_0})$. In each iteration, (3) is solved with various values of λ to obtain several updated models. Then, the model with the smallest misfit is chosen in Phase I and the model with the smallest norm and misfit at the target level is chosen in Phase II. This new model is used to compute its sensitivity for the next iteration. The algorithm for the model space Occam's inversion is given in Fig. 2.

The advantage of the code is its ability to converge in a small number of iterations (see examples in Constable et al. 1987; Siripunvaraporn and Egbert 2000; and Siripunvaraporn et al. 2005a). In addition, because it searches for the minimum structure model, the model can be treated as a "lower bound" for interpretation indicating that the structures are required by the data (Constable et al. 1987). The disadvantage of this scheme is the requirement of large computational resources. Equation (3) requires inverting a $M \times M$ matrix $[\lambda\ \mathbf{C}_\mathbf{m}^{-1} + \mathbf{J}_k^T\mathbf{C}_\mathbf{d}^{-1}\mathbf{J}_k]$ and also computing its $N \times M$ sensitivity matrix $\mathbf{J}_k$. Both processes require a substantial amount of CPU time for each iteration and also a large amount of memory. For example, to invert the EXTECH data set (Siripunvaraporn and Egbert 2009) which has 131 stations, 16 frequencies and 12 responses, $N = 25{,}152$, and the model parameter $M = 56 \times 56 \times 33 = 103{,}488$. Inversion of this data set with the model space Occam's inversion would require a minimum of 84 Gigabytes to store its inverted matrix and 20 Gigabytes to store its sensitivity matrix. This type of 3-D inversion is therefore impractical on most current computers.

3.1.2 The Data Space Occam's Inversion

To maintain the advantages of Occam's algorithm, Siripunvaraporn and Egbert (2000) transformed the computational space from model space to data space. Equation (3) can be rearranged as

(1) read initial model $\mathbf{m}_k$ and prior model $\mathbf{m_0}$ and compute RMS misfit X_k from model $\mathbf{m}_k$

(2) start outer loop iteration k:

(2.1) form sensitivity matrix $\mathbf{J}_k$ from model $\mathbf{m}_k$

(2.2) compute $\mathbf{đ}_k = \mathbf{d} - \mathbf{F}[\mathbf{m}_k] + \mathbf{J}_k(\mathbf{m}_k - \mathbf{m_0})$

(2.3) if mOCCAM: compute $\mathbf{\Gamma}_k^m = \mathbf{J}_k^T\mathbf{C_d}^{-1}\mathbf{J}_k$

if dOCCAM: compute $\mathbf{\Gamma}_k^d = \mathbf{C_d}^{-½}\mathbf{J}_k\mathbf{C_m}\mathbf{J}_k^T\mathbf{C_d}^{-½}$

(2.4) for various values of λs

(2.4.1) if mOCCAM : form and factorize $\mathbf{R}_k^m = [\lambda\mathbf{C_m}^{-1} + \mathbf{\Gamma}_k^m]$

if dOCCAM : form and factorize $\mathbf{R}_k^d = [\lambda\,\mathbf{I} + \mathbf{\Gamma}_k^d]$

(2.4.2) if mOCCAM : update model $\mathbf{m}_{k+1} - \mathbf{m_0} = [\mathbf{R}_k^m]^{-1}\,\mathbf{J}_k^T\mathbf{C}_d^{-1}\mathbf{đ}_k$,

if dOCCAM : update model $\mathbf{m}_{k+1} - \mathbf{m_0} = \mathbf{C_m}\mathbf{J}_k^T\,\mathbf{C_d}^{-½}\,[\mathbf{R}_k^d]^{-1}\,\mathbf{C_d}^{-½}\mathbf{đ}_k$

(2.4.3) compute RMS misfit $X_{k+1}(\lambda)$ from model $\mathbf{m}_{k+1}(\lambda)$ of each λ

(2.4.4) if Phase I: select $\mathbf{m}_{k+1}(\lambda)$ that has minimum misfit

if $X_{k+1}(\lambda) < \chi^2$, search for $\mathbf{m}_{k+1}$ that yield χ^2

if Phase II: select $\mathbf{m}_{k+1}(\lambda)$ that yield χ^2 with min imum model norm

(2.5) end

(2.6) exit when misfit $< \chi^2$ with minimum model norm ; else go to (2.1)

(3) end outer loop iteration

Fig. 2 Algorithms for model space (mOCCAM) and data space (dOCCAM) Occam's inversion (after Siripunvaraporn and Egbert 2007)

$$\mathbf{m} - \mathbf{m}_0 = \mathbf{C_m}\mathbf{J}^T\boldsymbol{\beta}, \tag{4}$$

where $\boldsymbol{\beta}$ is an $N \times 1$ unknown coefficient vector. The derivation of (4) from (3) can be seen in Siripunvaraporn et al. (2005a) and Parker (1994). Equation (4) and the linearized $\mathbf{F}[\mathbf{m}]$ are inserted into (2) to obtain the penalty functional in data space. Differentiating the penalty functional in data space with respect to $\boldsymbol{\beta}$ and rearranging, the following iterative sequence of approximate solutions can be obtained:

$$\mathbf{m}_{k+1} - \mathbf{m}_0 = \mathbf{C_m}\mathbf{J}_k^T\mathbf{C_d}^{-1/2}\left[\lambda\mathbf{I} + \mathbf{C_d}^{-1/2}\mathbf{J}_k\mathbf{C_m}\mathbf{J}_k^T\mathbf{C_d}^{-1/2}\right]^{-1}\mathbf{C_d}^{-1/2}\mathbf{đ}_k, \tag{5}$$

where $\mathbf{I}$ is the identity matrix. The algorithm for the data space Occam's inversion is also given in Fig. 2.

Transforming the model space calculation to the data space calculation helps to counter the disadvantage of the model space Occam's inversion. Equation (5) requires only inverting a $N \times N$ matrix $[\lambda\,\mathbf{I} + \mathbf{C_d}^{-½}\mathbf{J}_k\mathbf{C_m}\mathbf{J}_k^T\,\mathbf{C_d}^{-½}]$, rather than an $M \times M$ matrix as in the model space, but still requires computing its $N \times M$ sensitivity matrix $\mathbf{J}_k$. By decreasing the matrix dimension, CPU time is significantly decreased (Siripunvaraporn and Egbert 2000; Siripunvaraporn et al. 2005a). But the most significant advantage is the reduction of memory usage. Using EXTECH data as an example, in model space, we need 84 Gigabytes to store the inverted matrix of (3). In data space, we only need about 5 Gigabytes to store the inverted matrix of (5). However, both methods require about 20 Gigabytes to store the sensitivity matrix. This data space method is used in the WSINV3DMT program for 3-D

MT data (Siripunvaraporn et al. 2005a; Siripunvaraporn and Egbert 2009) and DASOCC for 2-D MT data (Siripunvaraporn and Egbert 2000). It is also used in network-MT inversion (Siripunvaraporn et al. 2004) and 2-D direct current resistivity inversion (Boonchaisuk et al. 2008).

3.2 The Gauss–Newton (GN) Method

Occam's inversion is actually a variant of a classical Gauss–Newton (GN) method which is a modification of Newton's method. In Newton's method, the objective functional is first approximated by a Taylor series expansion. The quadratic approximation of the objective functional is then minimized to produce a series of the updated model (see Eq. 8). In the GN method, the first derivative in the Hessian matrix of Newton's method is kept but the second-order derivative is discarded. This leads to an iterative sequence of approximate solutions,

$$\mathbf{m}_{k+1} - \mathbf{m}_k = \left[\lambda \mathbf{C}_{\mathbf{m}}^{-1} + \mathbf{J}_k^T \mathbf{C}_d^{-1} \mathbf{J}_k + \in_k \mathrm{I}\right]^{-1} \left[\mathbf{J}_k^T \mathbf{C}_d^{-1} (\mathbf{d}_k - -\mathbf{F}[\mathbf{m}_k]) - \lambda \mathbf{C}_{\mathbf{m}}^{-1} (\mathbf{m}_k - \mathbf{m}_0)\right], \tag{6}$$

where $\in_k$ is a damping parameter for numerical stability (Marquardt 1963). A scaling factor can be added to the right-hand side of (6) which could help to speed up the convergence rate. As with the model space Occam's inversion (6) requires storing the $M \times M$ inverted matrix and the $N \times M$ sensitivity matrix $\mathbf{J}_k$ and so a lot of memory is required. The GN method is therefore mostly used for 2-D cases (e.g., Rodi and Mackie 2001). Direct implementation on 3-D problems is impossible. Recently, Li et al. (2009) applied the adaptive cross approximation (ACA) technique to decompose the sensitivity matrix into two smaller matrices to make the large scale 3-D GN inversion method possible.

To get away with this large memory requirement problem and to make it possible for 3-D, the modeling mesh and inversion mesh can be discretized differently. Sasaki (2001) and Sasaki (2004) selected a subset of the modeling mesh as the inversion mesh to reduce the total number of model parameters. The inversion mesh is therefore larger than the modeling mesh. However, this selection process can lead to new problems with interpretation because it is difficult to know which subset is best suited for the inversion.

Although (6) computes $\mathbf{m}_{k+1} - \mathbf{m}_k$ whereas (3) computes $\mathbf{m}_{k+1}$, both (3) and (6) are actually equivalent. A major difference is that the step length control in Occam's inversion is obtained from varying λ to compute the data misfits from a series of trial solutions $\mathbf{m}(\lambda)$ in (3) in each iteration. Thus in the OCCAM inversion, λ is determined from the search process, while in GN, λ must be pre-chosen. This is an advantage of Occam's inversion scheme over the GN method. The GN algorithm is also given in Fig. 3.

3.3 The Gauss–Newton with Conjugate Gradient (GN-CG) Approach

To avoid storing both large matrices of the GN method, a conjugate gradient (CG) can be used to solve (6) and also (3) and (5). If CG is applied to (3) or (6), the algorithm is referred to as the model space conjugate gradient method. This algorithm is used in Mackie and Madden (1993), Rodi and Mackie (2001), Haber et al. (2004, 2007), and Lin et al. (2008). If CG is used to solve (5), the algorithm is the data space conjugate gradient method. It is used in Siripunvaraporn and Egbert (2007) and Siripunvaraporn and Sarakorn (2011). The GN-CG algorithms are summarized in Fig. 4.

(1) read initial model $\mathbf{m}_k$ and prior model $\mathbf{m_0}$ and compute RMS misfit X_k from model $\mathbf{m}_k$

(2) select λ

(3) start main GN iteration k:

(3.1) form sensitivity matrix $\mathbf{J}_k$ from model $\mathbf{m}_k$

(3.2) compute $\underline{\mathbf{d}}_k = \mathbf{J}_k^T\mathbf{C}_d^{-1}(\mathbf{d}_k - \mathbf{F}[\mathbf{m}_k]) - \lambda\,\mathbf{C}_m^{-1}(\mathbf{m}_k - \mathbf{m}_0)$

(3.3) compute $\mathbf{\Gamma}_k^m = \mathbf{J}_k^T\mathbf{C}_d^{-1}\mathbf{J}_k$

(3.4) form and factorize $\mathbf{R}_k = [\lambda\mathbf{C}_m^{-1} + \mathbf{\Gamma}_k^m + \epsilon_k\mathbf{I}]$; in case of instability, add ϵ_k.

(3.5) update model $\mathbf{m}_{k+1} - \mathbf{m}_k = [\mathbf{R}_k]^{-1}\underline{\mathbf{d}}_k$,

(3.6) compute RMS misfit X_{k+1} from model $\mathbf{m}_{k+1}$

(3.7) exit when misfit $< \chi^2$

(4) end main iteration

Fig. 3 Algorithm for Gauss–Newton (*GN*) inversion

(1) read initial model $\mathbf{m}_k$ and prior model $\mathbf{m_0}$ and compute RMS misfit X_k from model $\mathbf{m}_k$

(2) select λ

(3) start main GN-CG iteration k:

(3.1) compute $\mathbf{đ}_k = \mathbf{d} - \mathbf{F}[\mathbf{m}_k] + \mathbf{J}_k(\mathbf{m}_k - \mathbf{m_0})$ for model and data space inversion,

or compute $\underline{\mathbf{d}}_k = \mathbf{J}_k^T\mathbf{C}_d^{-1}(\mathbf{d}_k - \mathbf{F}[\mathbf{m}_k]) - \lambda\,\mathbf{C}_m^{-1}(\mathbf{m}_k - \mathbf{m}_0)$ for GN inversion

(3.2) start CG iteration

(3.2.1) solve $\mathbf{Rx} = \mathbf{b}$ with CG method

where $\mathbf{R} = [\lambda\mathbf{C}_m^{-1} + \mathbf{J}_k^T\mathbf{C}_d^{-1}\mathbf{J}_k]$, $\mathbf{b} = \mathbf{J}_k^T\mathbf{C}_d^{-1}\mathbf{đ}_k$, and $\mathbf{m}_{k+1} - \mathbf{m_0} = \mathbf{x}$ for model space method,

$\mathbf{R} = [\lambda\,\mathbf{I} + \mathbf{C}_d^{-1/2}\mathbf{J}_k\mathbf{C}_m\mathbf{J}_k^T\mathbf{C}_d^{-1/2}]$, $\mathbf{b} = \mathbf{C}_d^{-1/2}\mathbf{đ}_k$, and $\mathbf{m}_{k+1} - \mathbf{m_0} = \mathbf{C}_m\mathbf{J}_k^T\mathbf{C}_d^{-1/2}\mathbf{x}$

for data space method,

and $\mathbf{R} = [\lambda\mathbf{C}_m^{-1} + \mathbf{J}_k^T\mathbf{C}_d^{-1}\mathbf{J}_k + \epsilon_k\mathbf{I}]$, $\mathbf{b} = \underline{\mathbf{d}}_k$, and $\mathbf{m}_{k+1} - \mathbf{m}_k = \mathbf{x}$ for GN method.

(3.2.2) stop CG when $r < r_{tol}$ and exit, where $r = \|\mathbf{Rx} - \mathbf{b}\|/\|\mathbf{b}\|$

and r_{tol} is desired tolerance residual

(3.3) end CG iteration

(3.4) compute RMS misfit X_{k+1} from model $\mathbf{m}_{k+1}$

(3.5) exit when misfit $< \chi^2$

(4) end main GN-CG iteration

Fig. 4 Algorithm for Gauss–Newton conjugate gradient (*GN-CG*) inversion

With the application of CG, a large data set (like EXTECH) can be inverted even on a desktop PC because tens of Gigabytes of memory is not need. This can be accomplished because matrix **J** and the inverse matrix are never explicitly formed and stored in memory. Only a product of **J** or $\mathbf{J}^T$ with an arbitrary vector is required. $\mathbf{Jp}$ and $\mathbf{J}^T\mathbf{q}$ for each period, where **p** and **q** are arbitrary vectors, can be computed from solving one forward problem (see Mackie and Madden 1993; Newman and Alumbaugh 2000; Rodi and Mackie 2001; Siripunvaraporn and Egbert 2007; Lin et al. 2008). A description of how $\mathbf{Jp}$ and $\mathbf{J}^T\mathbf{q}$ are computed is given in the next section.

With the CG method, the inversion is divided into two main loops, the outer inversion loop and the CG inner loop. Most computations in the outer loop are the same including the model update for GN/OCCAM and GN-CG. The major difference in the outer loop is that for the GN-CG method there is no construction of $\mathbf{J}$ or $\mathbf{J}^{\mathrm{T}}\mathbf{C}_{\mathbf{d}}^{-1}\mathbf{J}$. Solving (3), (5) or (6) inside the GN/OCCAM method with a direct method (e.g., LU-factorization or Cholesky-decomposition) is replaced with a CG solver in the inner loop. Computational efficiency of the CG method is then controlled by the number of CG iterations (N_{cg}) which can be relatively large (Avdeev 2005; Siripunvaraporn and Egbert 2007; Siripunvaraporn and Sarakorn 2011). A large N_{cg} leads to a longer runtime. A pre-conditioner can be included to speed up the CG solver (Haber et al. 2004).

Siripunvaraporn and Egbert (2007) and Siripunvaraporn and Sarakorn (2011) use the data space conjugate gradient method for 2-D and 3-D MT, respectively, to show that N_{cg} is a function of λ. Larger λ would require smaller N_{cg}, while smaller λ demands larger N_{cg}. In addition, they also show that to minimize N_{cg}, the CG solver can be terminated earlier, for example, when the relative residual is about 10^{-2}, instead of 10^{-8} or less for more accurate solutions. The approximate solution obtained from CG can be used to update the model. The response computed from this model usually differs from the true value by less than 1–2%. However, if the CG solver is terminated too early (in order to have a very small N_{cg}), the inversion may fail to converge (Siripunvaraporn and Egbert 2007; Siripunvaraporn and Sarakorn 2011)

Although CG can significantly reduce memory usage, CG could fail when used to solve (3), (5) or (6) for small λ. This is because the orthogonality of the system matrix is broken (Siripunvaraporn and Sarakorn 2011). This is a significant disadvantage of the CG method. In addition, the CG method has been used to minimize the Tikhonov parametric functional (see Zhdanov 2002; Gribenko et al. 2010).

3.4 The Nonlinear Conjugate Gradient Inversion (NLCG)

In the GN-CG section, we discussed using the CG method to minimize the quadratic form of (3), (5) or (6). For non-quadratic cases, the non-linear conjugate gradient (NLCG) method can be directly applied to minimize the objective functional (2). The model is updated through

$$\mathbf{m}_{\mathrm{k}+1} = \mathbf{m}_{\mathrm{k}} + \alpha_{\mathrm{k}}\mathbf{u}_{\mathrm{k}}, \tag{7}$$

by finding α such that $W(\mathbf{m}_{\mathrm{k}} + \alpha_{\mathrm{k}}\mathbf{u}_{\mathrm{k}})$ is minimized. Because the problem is non-linear, the line search process is introduced instead of finding the step length parameter as in the CG approach. This line search process helps to avoid computing a large Hessian matrix. The new conjugate gradient direction $\mathbf{u}_{\mathrm{k}}$ is updated using either the Fletcher-Reeves (Fletcher and Reeves 1964) or the Polak-Ribiere (Polyak and Ribiere 1969) methods.

As with GN–CG, there is no need to construct any large matrices; only a product of $\mathbf{J}$ or $\mathbf{J}^{\mathrm{T}}$ with any vector is required. Because of its minimal memory requirement, NLCG for MT inversion has been gaining in popularity in the past decade as shown in many publications (Rodi and Mackie 2001; Newman and Alumbaugh 2000; Newman and Boggs 2004; Commer and Newman 2008; Kelbert et al. 2008; Commer and Newman 2009). Most of these algorithms use the Polak-Ribiere formula. The NLCG algorithm with Polak-Ribiere method is shown in Fig. 5.

As with the CG method, computational efficiency is controlled by the numbers of NLCG iterations and line search steps. To minimize the number of forward modeling calls,

Fig. 5 Algorithm for non-linear conjugate gradient (*NLCG*) inversion (after Newman and Alumbaugh 2000)

(1) read initial model $\mathbf{m}_k$ and compute RMS misfit X_k from model $\mathbf{m}_k$

(2) select λ

(3) start main NLCG iteration k:

(3.1) compute $\mathbf{r}_k = -\nabla W(\mathbf{m}_k)$

(3.2) compute $\mathbf{u}_k = \mathbf{M}_k^{-1}\,\mathbf{r}_k$ where $\mathbf{M}_k$ is a system preconditioner

(3.3) search for α_k that minimizes $W(\mathbf{m}_k + \alpha_k \mathbf{u}_k)$

(3.4) update model $\mathbf{m}_{k+1} = \mathbf{m}_k + \alpha_k \mathbf{u}_k$ and $\mathbf{r}_{k+1} = -\nabla W(\mathbf{m}_k)$

(3.5) compute RMS misfit X_{k+1} from model $\mathbf{m}_{k+1}$

(3.6) if $\|\mathbf{r}_{k+1}\|$ is small or when misfit $< \chi^2$, exit

else $\beta_{k+1} = (\mathbf{r}_{k+1}^{\mathrm{T}}\,\mathbf{M}_{k+1}^{-1}\mathbf{r}_{k+1} - \mathbf{r}_{k+1}^{\mathrm{T}}\mathbf{M}_k^{-1}\mathbf{r}_k)/(\mathbf{r}_k^{\mathrm{T}}\mathbf{M}_k^{-1}\mathbf{r}_k)$

(3.7) compute $\mathbf{u}_{k+1} = \mathbf{M}_{k+1}^{-1}\,\mathbf{r}_{k+1} + \beta_{k+1}\mathbf{u}_k$

(4) end main NLCG iteration

the method of Rodi and Mackie (2001) requires only 3 calls per NLCG iteration for the 2-D MT cases. As shown in Rodi and Mackie (2001), to converge to the desired level without a pre-conditioner, NLCG is not as efficient as the GN method (see Fig. 1, 3, 5 and 6 of Rodi and Mackie 2001). To improve the efficiency, a pre-conditioner can be constructed from the approximated Hessian related to the Laplacian operator (Rodi and Mackie 2001) or through the quasi-Newton rank two update formula (Newman and Alumbaugh 2000; Newman and Boggs 2004). With a pre-conditioner, the number of NLCG iterations needed to converge to the desired level is greatly reduced. However, the CPU time is comparable to that when using the model space GN-CG method (see Fig. 1 of Rodi and Mackie 2001). A comparison with the data space Occam's inversion or data space GN-CG has not been done.

3.5 The Quasi-Newton (QN) Method

Instead of applying CG as in the NLCG method, here a Newton method is directly applied to the objective functional W. The model is then updated through

$$\mathbf{m}_{k+1} = \mathbf{m}_k - \alpha_k \mathbf{H}_k^{-1}\mathbf{g}_k, \tag{8}$$

by finding α such that $W(\mathbf{m}_k - \alpha_k\,\mathbf{H}_k^{-1}\mathbf{g}_k)$ is minimized, where $\mathbf{H}_k = \partial^2 W/\partial \mathbf{m}^2$ and $\mathbf{g}_k = \partial W/\partial \mathbf{m}$ determined at $\mathbf{m}_k$ are the Hessian and the gradient, respectively.

To avoid constructing and inverting a large matrix as in the Occam's inversion, the quasi-Newton (QN) method approximates the inverse matrix $\mathbf{H}^{-1}$ through a recursive update process. There are two ranks of update matrix. Rank one updates (Broyden 1967) are simple but not robust, while rank two updates are more robust in minimizing general nonlinear functions, and are the most widely used schemes in many practical applications. The popular rank two updates are the Davidon-Fletcher-Powell (DFP) formula (Fletcher and Powell 1963) and the Broyden-Fletcher-Goldfarb-Shanno (BFGS) formula (Shanno 1970). The main computations for each QN iteration therefore consist of just computing the gradient $\mathbf{g}$ and performing the line search as in NLCG, and only a product of $\mathbf{J}$ or $\mathbf{J}^{\mathrm{T}}$ with any vector is computed. The memory requirement is therefore insignificant compared to that of the direct GN method or Occam's inversion.

Table 1 Comparisons of inversion algorithms

Inversion algorithm	Advantages	Disadvantages
Model space Occam's inversion	Small number of iterations to converge; Robust; λ is obtained automatically; obtain minimum model norm	Large CPU time and memory usage; often impractical to apply
Data space Occam's inversion	Same as the model space Occam's inversion but faster and uses significantly less memory; Can even be run on a PC	May require large memory for large problems
GN method	Small number of iterations to converge	Same as the model space Occam's inversion; convergence depends on λ; must use several values of λ
GN-CG method	Same as GN method but small amount of memory usage	Convergence depends on λ; Must use several values of λ; Can fail to converge for some λ; Can be slower than GN method
NLCG method	Small amount of memory usage	Convergence rate depends on λ, but comparable to GN. Must run with several values of λ; Can fail to converge for some λ
QN method	Small amount of memory usage	Convergence rate is slower than others

As with NLCG, the low computational cost has motivated the use of QN. However, a straightforward application of QN may not be successful in the case of electromagnetic inverse problems. Some successful attempts at using the QN method for large-scale electromagnetic problems can be found in Haber (2005) and Avdeev and Avdeeva (2009). Because of its Hessian approximation, Haber (2005) demonstrated that a direct application of QN method is not efficient. It converges slowly or could even fail to converge. To improve its efficiency, some modifications are necessary such as approximating only the Hessian related to the data misfit instead of the full Hessian (Haber 2005) or introducing an additional regularization (Avdeev and Avdeeva 2009). Often the approximate inverse Hessian of the QN method is used as a pre-conditioner in the CG solver in the GN method (Haber et al. 2007) or in the non-linear conjugate gradient (NLCG) method described in the previous section (Newman and Boggs 2004). Because of its sophistication, I do not include the basic QN algorithm in this paper.

3.6 Comparisons of Algorithms

All algorithms have both advantages and disadvantages (Table 1). All have limitations. The convergence rate (CPU time) and memory usage depend on the algorithms used. Parameters used in the model covariance and roughness operator control the smoothness of the inverted model. Different parameters should be tested before proceeding to the interpretation.

4 Other Necessary Components for 3-D MT Inversion

To start developing an inversion code, developers should pay particular attention to these necessary components, as well as the inversion algorithms, in order to obtain a reliable and efficient inversion.

4.1 Magnetotelluric Forward Modeling

The 3-D MT forward modeling (FWD) routine is the most important element for efficient 3-D inversion. It is used to compute the model responses in most inversion schemes. It is heavily called to construct the sensitivity matrix **J** in the OCCAM or GN methods, or to compute its products with any vectors in the CG or QN or NLCG methods. An inaccurate forward routine can mislead the interpretation, while an inefficient routine can lead to an inefficient 3-D inversion. A fast and accurate routine would be an ideal for 3-D inversion which as of now is still unreachable, particularly when the model is complicated.

3-D MT responses at the observed sites can be obtained by solving the second order Maxwell's equation in either the magnetic field (Mackie et al. 1994) or the electric field (Smith 1996; Siripunvaraporn et al. 2002;) from a known electrical conductivity (σ) or resistivity (ρ) model. Starting with an $e^{-i\omega t}$ time-dependence, the electric field **E** can be solved from

$$\nabla \times \nabla \times \mathbf{E} = i\omega\mu\sigma\mathbf{E}, \tag{9}$$

where ω is an angular frequency and μ is the magnetic permeability. For more complicated models, staggered grid finite difference (FD) or finite element (FE) are the most two commonly used methods for solving (9). The advantage of FD is that it is easy to apply and is fast. However, topography inclusion can be difficult. In contrast, FE is difficult to apply and slow to converge but topography inclusion is more natural (Vachiratienchai et al. 2010). In addition to FD or FE, another technique is the integral equation (IE) formulation (Weidelt 1975; Hohmann 1975; Wannamaker 1991; Zhdanov 2002, 2009).

After grid discretization, the application of either FD or FE to (9) yields a system of equations for a given period or frequency,

$$\mathbf{Se} = \mathbf{b}, \tag{10}$$

where **e** represents the unknown internal electric fields, and **b** is a vector containing the terms associated with the boundary electric fields. The coefficient matrix **S** is large, sparse, symmetric, and complex. The difference between FE and FD is that **S** for FE has about 3 times more elements than that of FD. This larger **S** therefore requires significantly more computational time. Comparisons of accuracy and computational time of magnetotelluric forward modeling can be found in Han et al. (2009).

For 3-D problems, **S** is relative large, and almost impossible to solve with any direct methods (Streich 2009). It is therefore commonly solved with iterative solvers, for example, the bi-conjugate gradient (BiCG) method (Smith 1996), the quasi minimum residual (QMR) method (Siripunvaraporn et al. 2002), and the minimum residual method (MRM; Mackie et al. 1994). A divergence correction (see Mackie et al. 1994; Smith 1996) is intermittently imposed inside the iterative solver, particularly when solving for long period responses. After obtaining the interior electric fields, the surface impedance responses can be obtained from a linear combination of a vector **a** associated at a measurement site and the computed electric fields using

$$\mathbf{F}[\mathbf{m}] = \mathbf{a}^{\mathrm{T}}\mathbf{e} = \mathbf{a}^{\mathrm{T}}\mathbf{S}^{-1}\mathbf{b}. \tag{11}$$

Vector **a** is derived from the combinations of the first order Maxwell's equations with some approximation.

4.2 Sensitivity Calculation

To obtain the sensitivity $\mathbf{J} = \partial\mathbf{F}/\partial\mathbf{m}$ of all sites at a given period, we start with the differentiation of (11) with respect to the model $\mathbf{m}$. We obtain

$$\mathbf{J} = \partial\mathbf{F}/\partial\mathbf{m} = \partial\left(\mathbf{a}^{\mathrm{T}}\mathbf{e}\right)/\partial\mathbf{m} = \mathbf{a}^{\mathrm{T}}\mathbf{S}^{-1}\Theta + \Psi, \tag{12}$$

where $\Theta = \partial\mathbf{b}/\partial\mathbf{m} - (\partial\mathbf{S}/\partial\mathbf{m})\mathbf{e}$ and $\Psi = (\partial\mathbf{a}^{\mathrm{T}}/\partial\mathrm{m})\mathbf{e}$.

Rodi (1976) demonstrated that there are two algorithms to compute (12) and these are summarized in Fig. 6. The first method is a straightforward application of (12) to generate each column of $\mathbf{J}$. This requires looping over M to obtain the entire matrix $\mathbf{J}$. In the outer loop from 1 to M, we start by constructing $\boldsymbol{\Theta}$ and evaluating $\mathbf{S}^{-1}\boldsymbol{\Theta}$. This therefore requires solving (10) M times per period and per polarization. The result of $\mathbf{S}^{-1}\boldsymbol{\Theta}$ is then used in the inner loop from 1 to N_s, where N_s is the number of station, by multiplying it by $\mathbf{a}^{\mathrm{T}}$ and adding the result to $\boldsymbol{\Psi}$. Because this method requires calling the forward code M times, it can be very computationally costly for 3-D case and is therefore not a popular technique in recent 3-D codes.

Another method is to construct each row of $\mathbf{J}$. This requires looping over N_s to obtain the full matrix $\mathbf{J}$. In the outer loop from 1 to N_s, we only evaluate $(\mathbf{a}^{\mathrm{T}}\mathbf{S}^{-1})^{\mathrm{T}}$ which therefore requires solving (10) only N_s times per period per polarization. In the inner loop from 1 to M, $(\mathbf{a}^{\mathrm{T}}\mathbf{S}^{-1})^{\mathrm{T}}$ is multiplied by $\boldsymbol{\Theta}^{\mathrm{T}}$ and the result is added to $\boldsymbol{\Psi}^{\mathrm{T}}$. The second method is equivalent to the reciprocity property of the electromagnetic fields (see Rodi 1976; Mackie and Madden 1993; Siripunvaraporn and Egbert 2000). The reciprocity theorem helps to significantly decrease the computational time of the program.

To speed up the inversion, many developers use approximate sensitivities by using the electric fields from the previous iteration (Han et al. 2008) or from the half-space model (Farquharson and Oldenburg 1996). Sasaki (2001) calculated the sensitivity at every other iteration to update the model. Siripunvaraporn et al. (2005a) used a higher tolerance level to terminate solving (10) to obtain the incomplete or approximate sensitivities. All of these techniques can significantly reduce computational time but one must be cautious. If the approximate sensitivity differs greatly from the actual sensitivity, it could result in failure of the inversion code to reach the target misfit.

```
for  i = 1 to M
   solve   Sx = Θ
   for  j = 1, Ns
      J(j,i) = a^T x + Ψ
   end
end
```

a) a direct method to compute the sensitivity matrix $\mathbf{J}$.

```
for  j = 1 to Ns
   solve   Sx = a
   for  i = 1 to M
      J(j,i) = Θ^T x + Ψ
   end
end
```

b) reciprocity technique to compute the sensitivity matrix $\mathbf{J}$.

Fig. 6 Two algorithms used to form the sensitivity matrix $\mathbf{J}$. **a** a direct method. **b** reciprocity technique

4.3 Multiplication of Vectors by J or J^T

CG, QN or NLCG do not require explicitly forming **J**, but only a product of it with any vector. By not explicitly forming **J**, memory usage is significantly reduced. To compute the product of **J** with a given vector **p**, (12) becomes

$$\mathbf{Jp} = \mathbf{a}^T\mathbf{S}^{-1}\mathbf{\Theta p} + \mathbf{\Psi p}. \tag{13}$$

The process to obtain (13) is similar to the first method of Rodi (1976) except there is no looping over M (Fig. 6a). $\mathbf{\Theta p}$ is computed first followed by $\mathbf{S}^{-1}\mathbf{p}$. The result is then used in the loop over N_s by multiplying it with $\mathbf{a}^T$ and the result is added to $\mathbf{\Psi p}$.

To compute the product of $\mathbf{J}^T$ with a given vector **q**, Eq. (12) becomes

$$\mathbf{J}^T\mathbf{q} = \mathbf{\Theta}^T\left[\mathbf{S}^T\right]^{-I}\mathbf{aq} + \mathbf{\Psi}^T\mathbf{q}. \tag{14}$$

The process to obtain (14) is the same as the second method in Rodi (1976) except there is no looping over N_s (Fig. 6b). **aq** is computed first. Because $\mathbf{S} = \mathbf{S}^T$, we can then proceed to evaluate $\mathbf{S}^{-1}\mathbf{aq}$. Looping over M, $\mathbf{S}^{-1}\mathbf{aq}$ is multiplied by $\mathbf{\Theta}^T$ before the result is added to $\mathbf{\Psi}^T\mathbf{q}$.

Both processes show that computing the product of **J** or $\mathbf{J}^T$ requires solving (10) only once per period and per polarization. This is the main reason why inversion algorithms relying on the product of **J** with a vector are gaining in popularity. However, this must be done with care. Through detailed investigation, Siripunvaraporn and Sarakorn (2011) have shown that one forward modeling call to construct one row of **J** uses less CPU time than a call to compute **Jp** or $\mathbf{J}^T\mathbf{q}$. Although they solve the same system of equations (10), the right-hand sides are different. Forming one row of **J** has **a** as its right hand side, while computing **Jp** and $\mathbf{J}^T\mathbf{q}$ have $\mathbf{\Theta p}$ and **aq**, as their right-hand sides, respectively. All vectors (**a**, $\mathbf{\Theta p}$ and **aq**) are sparse, but $\mathbf{\Theta p}$ and **aq** involve more non-zero terms than **a**. Consequently, solving (10) with $\mathbf{\Theta p}$ and **aq** as the right-hand sides will require a larger number of QMR iterations than with just **a** as the right-hand side. Thus, one forward call to compute **Jp** or $\mathbf{J}^T\mathbf{q}$ requires more CPU time than just one call to construct one row of **J**. Comparing the efficiency of the inversion algorithms by counting the number of forward modeling calls can be misleading (e.g., Siripunvaraporn and Egbert 2007).

4.4 Model Covariance

The model covariance $\mathbf{C_m}$ is a matrix that describes the expected magnitude of the resistivity variations relative to the prior model $\mathbf{m}_0$. It can also be used as a smoothness operator. Because $\mathbf{C_m}$ itself is a full $M \times M$ matrix, it is very computationally expensive to explicitly form and store the whole matrix. Siripunvaraporn and Egbert (2000) and Siripunvaraporn et al. (2005a) compute its product with each column of $\mathbf{J}^T$ or $\mathbf{C_m}\mathbf{J}^T$ by solving a series of 1-D diffusion equations. The smoothness is controlled by the diffusion parameter γ. A large γ produces smoother structures, while a small γ yields localized structures. In addition, γ can be defined differently in each direction. For example, it can be defined as a function of depth which is in agreement with the loss of resolution of MT data at greater depth (Siripunvaraporn and Egbert 2000). Kelbert et al. (2008) also use the same concept of $\mathbf{C_m}$ but defined the smoothness differently to be consistent with their global EM induction problem.

In the model space calculation, (3) and (6) require $\mathbf{C}_\mathbf{m}^{-1}$, not $\mathbf{C}_\mathbf{m}$. The direct benefit of having $\mathbf{C}_\mathbf{m}$ in the data space is its ability to directly include the prior information such as faults or oceans. These features can be easily incorporated by setting different parameters across these features (Siripunvaraporn and Egbert 2000). Finding $\mathbf{C}_\mathbf{m}^{-1}$ is not practical due to the size and singularness of the matrix. Many therefore use roughness, the inverse of the smoothness, to define the model norm (Constable et al. 1987; deGroot-Hedlin and Constable 1990). The roughness is usually sparse and easy to apply. It can be defined in many different ways. However, many define it as the finite difference approximation to the Laplacian (∇^2) operator (e.g., Constable et al. 1987; Rodi and Mackie 2001; Newman and Alumbaugh 2000).

4.5 Parallel Implementation for PC Clusters

To speed up the inversion, many developers implement their codes on a parallel system which can be run directly through the software provided by the parallel system manufacturer or manually. One of the most popular parallel machines is the PC cluster which is easy to build and operate, and is also cheaper than a massive parallel machine. Implementation of the inversion code for the PC cluster is straightforward. Solving (10) for each frequency is independent of the results from other frequencies. Therefore, the calculation for each frequency (to solve for its response or its sensitivity or the product of sensitivity with any vectors) can be computed separately on different nodes. This parallelizing over frequencies is done via MPI (Message Passing Interface) libraries. This type of parallelization is used in many algorithms (e.g., Siripunvaraporn et al. 2004; Han et al. 2008; Lin et al. 2009; Siripunvaraporn and Egbert 2009; Siripunvaraporn and Sarakorn 2011). For multi-core desktops, OpenMP is another choice for parallelizing the 3-D inversion codes (Virginie and Wanamaker 2010).

In the OCCAM/GN algorithms, all nodes must send their sensitivity calculation to the master node to compute the cross-product matrix in (3), (5) or (6). This step can be performed efficiently using a cyclic data distribution and was demonstrated in Siripunvaraporn and Egbert (2009). To solve (3), (5) or (6), one can solve the system of equation on just one node, usually the master node, by using a direct method (Siripunvaraporn et al. 2004). Another option is to distribute the system to many nodes and use an iterative solver which is easy to implement on a parallel system (Siripunvaraporn and Egbert 2009).

5 Dos and Don'ts When Performing 3-D Inversion

As seen in previous sections, inversion is just a mathematical tool to convert the observed or collected data to a more meaningful model structure. Because of the non-uniqueness of the inversion, the same data set with the same error bars can generate different models that fit the data at the same level. Here is a list of "dos" and "don'ts" to help prevent simple mistakes which could lead to major misinterpretations when using any 3-D inversion schemes.

- Do check your observed data and try to remove biased or bad data. Often, inversion of a large number of frequencies is almost impossible. A subset of the data is therefore used in the inversion. Select the frequencies that have less biased or bad data. Bad data can be excluded from a frequency band by allowing very large error bars. Bad data and very small error bars can affect the convergence of the inversion which can produce a large misfit and can generate artifacts within the inverted models. Bad data should be removed before any inversions are carried out.

- Do look at the actual data misfit between the observed data and the calculated data after the inversion. Relying on a statistical RMS value can be misleading.
- Do test the accuracy of your grid discretization. Grid discretization can generate errors in the responses. The user should know the level of the error arising from his/her grid discretization first. The error level must be used to account for the fitting to the data by either increasing the size of the error bars or setting a higher desired misfit. To find the error level, a simple strategy is to test the grid discretization with a half-space model but with different resistivities covering all resistivities expected in the area. In addition, the test should be conducted at all frequencies. This strategy is good for testing the vertical grid discretization which is very important. In half-space cases, the apparent resistivities should equal the true resistivity, and phases should be 45°. If the error is high, repeat the process. After obtaining a good vertical grid discretization, a simple way to test the horizontal discretization is to add some conductors or resistors at the depth of interest in the models. An additional mesh is generated by doubling the number of grid points in the horizontal direction. Both meshes on the same model are tested with the forward modeling routine at various frequencies. If both grids produce a large discrepancy, use a finer grid (assuming that a finer mesh is better) and repeat the horizontal grid discretization process again until the discrepancy is acceptable.
- Boundary conditions used inside the forward modeling routines are usually derived from simple 1-D or 2-D cases. It is necessary to construct the initial models that have the boundaries far away from all of the sites so that the boundaries will not affect the area of interest.
- Do check the units of the observed data and the time-dependence. Data processing and inversion program can use different units and time-dependence. The difference in units comes from using different definitions. The MT transfer function measures **E** over **B** and has a unit of mV/km/nT, while the impedance is defined as **E** over **H**, and has units of ohms. Notice that $1\Omega = 796$ mV/km/nT. The time-dependence is either $e^{-i\omega t}$ or $e^{+i\omega t}$. If the data processing and the inversion use different time-dependences, then to make it consistent, replace the impedance responses by their complex conjugates.
- Do know capability of your inversion program and how the inversion works and do know your hardware. Some code has memory limitations. The user needs to determine how large the data set and model discretization can be before inverting the data.
- Do not proceed to the interpretation right away after just one inversion. Do run with different parameters, different initial models with different grid discretizations, and, if possible, with different algorithms. This is necessary because of the non-uniqueness of the inversion. Features which appear on all runs are likely to be required by the data.
- Do feasibility studies with the inversion codes both before and after obtaining the inverted model to test for artifacts or suspicious features generated from the inversion. This process could be conducted by building a synthetic model from the main features obtained from the inversion. The calculated response of the synthetic model is then used for the inversion. Comparison of the inverted models from the synthetic model and the real data can be used to distinguish the main features from any artifacts.

6 Outstanding Problems and the Future of 3-D Inversion

3-D MT inversion has made significant progress over the past decade. The biggest problem is the speed of the inversion which is the result of the slow convergence of the forward

modeling routines. The forward modeling code is at the heart of all inversion schemes. In order for 3-D inversion to progress significantly, substantial improvements in the speed and accuracy of the 3-D forward modeling code are needed. This problem is not easy, but can be overcome in many different ways.

Future technology will significantly increase the speed of computers. One approach would be to hang on to the current forward modeling routines and wait for improvements in computers. Another is to adapt the current routines to fit with the new technologies. Currently, a PC Cluster is cheap and easy to build and is as efficient as a massively parallel system. Many adapt the forward routines to run each frequency on a different processor via MPI or OpenMP as described earlier. However, this technique does not speed up the forward routine, it just distributes the workload. Using graphics processing units (GPU) could be a new trend for future research to significantly increase the speed of MT forward modeling through technology (Schultz et al. 2010).

There are so many different ways to speed up the forward modeling routines without using new technology. Preconditioners for iterative solvers could be another topic for future research. Most of the preconditioners in forward routines are still based on the incomplete factorization of the diagonal blocks of the coefficient matrix **S** (Siripunvaraporn et al. 2002; Smith 1996; Mackie et al. 1994). In addition, a divergence correction which significantly helps speed up the code (Smith 1996) could retard the code as well if improperly applied. A better preconditioner and a more efficient divergence correction process could help greatly speed up the forward routine. In addition, transforming a large system into smaller system such as in the domain decomposition technique (Rung-Arunwan and Siripunvaraporn 2010) can further decrease the CPU time and significantly lower the memory requirement. In addition, a direct solver instead of the iterative solvers can be used to solve (10).

In addition to speeding up the forward routine, there are also several ways to improve the efficiency of the inversion. This can be divided into improving or combining the existing algorithms as reviewed above or inventing a new scheme. Improvements in the existing algorithms could include a new preconditioner for the CG/NLCG algorithms and a reduction in the number of equations in the OCCAM/GN methods. A hybrid of different schemes would be achieved by combining the advantages of each method and discarding the disadvantages (Siripunvaraporn and Sarakorn 2011; Egbert 2010). Approximate sensitivities are another issue but should be treated with caution. There are currently many new variants of the mathematical algorithms. These new algorithms should be tested with MT problems, but significant improvements may not occur unless the other parts mentioned earlier are efficient. In addition, new algorithms for large scale electromagnetic problems are currently being developed, particularly the marine controlled-source electromagnetic surveys (e.g., Abubakar et al. 2009; Li et al. 2009).

Another major problem for MT inversion is to efficiently incorporate topography/bathmetry into the inversion. Currently, using finite elements (FE) is the most efficient method to incorporate the topography/bathmetry. However, the FE technique leads to a much larger system of equations than with FD. Consequently, the computational time needed for the FE method at each frequency is still a lot more than for FD. This would lead to a very inefficient inversion. Using FD to model the topography/bathmetry is not efficient because the grid must be finely discretized in order to account for the geometry of the topography/bathmetry sufficiently well to obtain an accurate response. This would again lead to a larger system of equations and require much more CPU time to obtain the same accuracy as with FE. An efficient hybrid FE–FD method has been successfully used to incorporate the topography for 2-D direct current (DC) resistivity codes (Vachiratienchai

et al. 2010). It has been shown to be as accurate as the FE method but requires less CPU time than the FD method for the same grid discretization size. The hybrid technique could again be applied to MT problems.

7 Final remarks

Inversion is just a mathematical tool to convert observed data to a resistivity model. In a field survey, it can be used as a quick and easy way to look at the anomalies which could then be used as the basis for adding more sites. However, in a laboratory, users must invert the data with different parameters with a lot of feasibility studies to distinguish the artifacts generated by the inversion from the "true" structure recovered from the inversion. To significantly improve the 3-D MT inversion, I confirm the conclusion in Avdeev (2005) which recommends that developers make their codes available for others to test and use. However, the users should also obey the copyright law and should not treat other people's code as their own. Doing so discourages the developers which consequently would slow the progress of 3-D inversion. The future is bright for 3-D inversion.

Acknowledgments This research has been supported by the Thai Center of Excellence in Physics (ThEP) and by the Thailand Research Fund (TRF: RMU5380018).

References

Abubakar A, Habashy TM, Li M, Liu J (2009) Inversion algorithms for large-scale geophysical electromagnetic measurements. Inv Prob 25:1–30

Árnason K, Eysteinsson H, Hersir GP (2010) Joint 1D inversion of TEM and MT data and 3D inversion of MT data in the Hengill area, SW Iceland. Geothermics 39:13–34

Avdeev D (2005) Three-dimensional electromagnetic modeling and inversion from theory to application. Surv Geophys 26:767–799

Avdeev D, Avdeeva A (2009) 3D Magnetotelluric inversion using a limited-memory quasi-Newton optimization. Geophysics 74:F45–F57

Bahr K (1991) Geological noise in Magnetotelluric data: a classification of distortion types. Phys Earth Plan Int 66:24–38

Becken M, Ritter O, Burkhardt H (2008a) Mode separation of Magnetotelluric responses in three-dimensional environments. Geophys J Int 172:67–86

Becken M, Ritter O, Park SK, Bedrosian PA, Weckmann U, Weber M (2008b) A deep crustal fluid channel into the San Andreas Fault system near Parkfield, California. Geophys J Int 173:718–732

Boonchaisuk S, Vachiratienchai C, Siripunvaraporn W (2008) Two-dimensional direct current (DC) resistivity inversion: data space Occam's approach. Phys Earth Plan Int 168:204–211

Borner RU (2010) Numerical modeling in geo-electromagnetics: advances and challenges. Surv Geophys 31:225–245

Broyden CG (1967) Quasi-Newton methods and their application to function minimization. Math Comput 21:368

Caldwell TG, Bibby H, Brown C (2004) The magnetotelluric phase tensor. Geophys J Int 158. doi: 10.1111/j.1365-246X.2004.02281.x

Chave AD, Smith JT (1994) On electric and magnetic galvanic distortion tensor decompositions. J Geophys Res 99:4669–4682

Chen X, Weckmann U (2010) From forward modeling of MT phases over 90° towards 2D anisotropic inversion, IAGA WG 1.2 on electromagnetic induction in the earth, 20th workshop abstract, Giza, Egypt, September 18–24

Commer M, Newman GA (2008) New advances in three-dimensional controlled-source electromagnetic inversion. Geophys J Int 172:513–535

Commer M, Newman GA (2009) Three-dimensional controlled-source electromagnetic and Magnetotelluric joint inversion. Geophys J Int. doi:10.1111/j.1365-246X.2009.04216.x

Constable CS, Parker RL, Constable CG (1987) Occam's inversion: a practical algorithm for generating smooth models from electromagnetic sounding data. Geophysics 52:289–300
Cumming W, Mackie R (2010) Resistivity imaging of geothermal resources using 1D, 2D and 3D MT inversion and TDEM static shift correction illustrated by a Glass Mountain case history. In: Proceedings world geothermal congress 2010, Bali, Indonesia, 25–29 April
Degroot-Hedlin C, Constable S (1990) Occam's inversion to generate smooth, two-dimensional models from magnetotelluric data. Geophysics 55(12):1613–1624
Egbert G (2010) Efficient inversion of multi-frequency and multi-transmitter EM data, IAGA WG 1.2 on electromagnetic induction in the earth, 20th workshop abstract, Giza, Egypt, September 18–24
Farquharson CG, Craven JA (2008) Three-dimensional inversion of Magnetotelluric data for mineral exploration: an example from the McArthur River uranium deposit, Saskatchewan, Canada. J Appl Geophys 68:450–458
Farquharson CG, Oldenburg DW (1996) Approximate sensitivities for the electromagnetic inverse problem. Geophy J Int 126:235–252
Fletcher R, Powell MJD (1963) A rapidly convergent descent method for minimization. Comput J 6:163–168
Fletcher R, Reeves CM (1964) Function minimization by conjugate gradients. Comput J 7:149–154
Goldak D, Kosteniuk P (2010) 3D inversion of transient magnetotelluric data: an example from Pasfield Lake, Saskatchewan, EGM 2010 International Workshop, 11–14 April, 2010. Capri, Italy
Gribenko A, Zhdanov M (2007) Rigorous 3D inversion of marine CSEM data based on the integral equation method. Geophysics 72:WA73–WA84
Gribenko A, Green M, Cuma M, Zhdanov MS (2010) Efficient 3D inversion of MT data using integral equations method and the receiver footprint approach: application to the large-scale inversion of the EarthScope MT data: expanded Abstracts of the SEG meeting, Denver, Colorado, pp 644–649
Groom RW, Bailey R (1989) Decomposition of Magnetotelluric impedance tensors in the presence of local three-dimensional galvanic distortion. J Geophys Res 94:1913–1925
Gunther T, Rucker C, Spitzer K (2006) Three-dimensional modeling and inversion of dc resistivity data incorporating topography–II. Inv Geophys J Int 166:506–517
Haber E (2005) Quasi-Newton methods for large scale electromagnetic inverse problem. Inverse Problem 21:305–317
Haber E, Asher U, Oldenburg D (2000) On optimization techniques for solving nonlinear inverse problems. Inv Prob 16:1263–1280
Haber E, Ascher U, Oldenburg D (2004) Inversion of 3D electromagnetic data in frequency and time domain using an inexact all-at-once approach. Geophysics 69:1216–1228 (n5)
Haber E, Oldenburg DW, Shekhtman R (2007) Inversion of time domain three-dimensional electromagnetic data. Geophys J Int 171:550–564
Han N, Nam MJ, Kim HJ, Lee TJ, Song Y, Suh JH (2008) Efficient three-dimensional inversion of Magnetotelluric data using approximate sensitivities. Geophys J Inter 175:477–485
Han N, Nam MJ, Kim HJ, Song Y, Suh JH (2009) A comparison of accuracy and computation time of three-dimensional Magnetotelluric modeling algorithms. J Geophys Eng 6:136. doi:10.1088/1742-2132/6/2/005
Hautot S, Tarits P (2009) A new coarse-to-fine 3-D Magnetotelluric inversion method—application to field data for hydrocarbon exploration, Society of Petroleum Engineers—71st European association of geoscientists and engineers conference and exhibition, 1, pp 663–667
Heise W, Caldwell TG, Bibby HM, Bannister SC (2008) Three-dimensional modelling of magnetotelluric data from the Rotokawa geothermal field, Taupo Volcanic Zone, New Zealand. Geophys J Inter 173:740–750
Heise W, Caldwell TG, Bibby HM, Bennie SL (2010) Three-dimensional electrical resistivity image of magma beneath an active continental rift, Taupo Volcanic Zone, New Zealand. Geophys Res Lett 37 (10):art. No. L10301
Hill GJ, Caldwell TG, Heise W, Chertkoff DG, Bibby HM, Burgess MK, Cull JP, Cas RAF (2009) Distribution of melt beneath Mount St Helens and Mount Adams inferred from magnetotelluric data. Nat Geosci 2:785–789. doi:10.1038/NGE0661
Hohmann GW (1975) Three dimensional induced polarization and EM modeling. Geophysics 40:309–324
Ichihara H, Mogi T (2009) A realistic 3-D resistivity model explaining anomalous large magnetotelluric phases: the L-shaped conductor model. Geophys J Int 179:14–17
Ichihara H, Mogi T, Uyeshima M, Sakanaka S (2010) Three dimensional conductor models explaining out of quadrant magnetotelluric phases, IAGA WG 1.2 on Electromagnetic Induction in the earth, 20th workshop abstract, Giza, Egypt, September 18–24

Ingham MR, Bibby HM, Heise W, Jones KA, Cairns P, Dravitzki S, Bennie SL, Caldwell TG, Ogawa Y (2009) A Magnetotelluric study of Mount Ruapehu volcano, New Zealand. Geophys J Inter 179:887–904

Jones KA, Ingham MR, Bibby HM (2008) The hydrothermal vent system of Mount Ruapehu, New Zealand–a high frequency MT survey of the summit plateau. J Volcanol Geotherm Res 176:591–600

Kelbert A, Egbert GD, Schultz A (2008) Non-linear conjugate gradient inversion for global EM induction: resolution studies. Geophys J Int 173:365–381

Ledo J (2006) 2-D versus 3-D Magnetotelluric data interpretation. Surv Geophys 27:111–148

Li M, Abubakar A, Habashy TM (2009) Regularized Gauss–Newton method using compressed Jacobian matrix for controlled source electromagnetic data inversion: expanded abstracts of the SEG meeting, Houston, Texas, pp 704–708

Lilley FEM, Weaver JT (2010) Phases greater than 90° in MT data: analysis using dimensionality tools. J Appl Geophys 70:9–16

Lin C, Tan H, Tong T (2008) Three-dimensional conjugate gradient inversion of Magnetotelluric sounding data. Appl Geophys 5:314–321

Lin C, Tan H, Tong T (2009) Parallel rapid relaxation inversion of 3D Magnetotelluric data. Appl Geophys 6:77–83

Mackie RL, Madden TR (1993) Three-dimensional magnetotelluric inversion using conjugate gradients. Geophys J Int 115:215–229

Mackie R, Watts MD (2007) Joint 3D inversion of marine CSEM and MT data, SEG, San Antonio 2007 annual meeting, pp 574–578

Mackie RL, Smith JT, Madden TR (1994) Three-dimensional electromagnetic modeling using finite difference equations: the Magnetotelluric example. Radio Sci 29:923–935

Marquardt DW (1963) An algorithm for least-squares estimation of nonlinear parameters. J Soc Indust Appl Math 11:431–441

Marti A, Queralt P, Ledo J (2009) WALDIM: A code for the dimensionality analysis of Magnetotelluric data using the rotational invariants of the Magnetotelluric tensor. Comput Geosci 35:2295–2303

McNeice G, Jones AG (2001) Multisite, multifrequency tensor decomposition of Magnetotelluric data. Geophysics 66:158–173

Newman GA, Alumbaugh DL (1997) Three-dimensional massively parallel electromagnetic inversion—I. Theory Geophys J Int 128:345–354

Newman GA, Alumbaugh DL (2000) Three-dimensional magnetotelluric inversion using non-linear conjugate gradients. Geophys J Int 140:410–424

Newman GA, Boggs PT (2004) Solution accelerators for large-scale three-dimensional electromagnetic inverse problems. Inv Prob 20:S151–S170

Newman GA, Hoversten GM (2000) Solution strategies for two- and three-dimensional electromagnetic inverse problems. Inv Prob 16:1357–1375

Newman GA, Recher S, Tezkan B, Neubauer FM (2003) Case History 3D inversion of a scalar radio Magnetotelluric field data set. Geophysics 68:791–802

Newman GA, Gasperikova E, Hoversten GM, Wannamaker PE (2008) Three-dimensional magnetotelluric characterization of the Coso geothermal field. Geothermics. doi:10.1016/j.geothermics.2008.02.006

Ogawa Y (2002) On two-dimensional modeling of magnetotelluric field data. Surv Geophys 23:251–272

Parker RL (1994) Geophysical inverse theory. Princeton University Press, Princeton

Patro PK, Egbert GD (2008) Regional conductivity structure of Cascadia: preliminary results from 3D inversion of USArray transportable array Magnetotelluric data. Geophys Res Lett 35:art. no. L20311

Polyak E, Ribiere G (1969) Note sur la convergence des methods conjugees. Rev Fr Inr Rech Oper 16:35–43

Press WH, Teukolsky SA, Vetterling WT, Flannery BP (1992) Numerical Recipes in FORTRAN: the art of scientific computing, 2nd edn. Cambridge University Press, Cambridge

Rodi WL (1976) A technique for improving the accuracy of finite element solutions for Magnetotelluric data. Geophys J Roy Astr Soc 44:483–506

Rodi W, Mackie RL (2001) Nonlinear conjugate gradients algorithm for 2-D magnetotelluric inversion. Geophysics 66:174–187

Rung-Arunwan T, Siripunvaraporn W (2010) An efficient modified hierarchical domain decomposition for two-dimensional Magnetotelluric forward modeling. Geophys J Int 183:634–644

Sasaki Y (2001) Full 3D inversion of electromagnetic data on PC. J Appl Geophys 46:45–54

Sasaki Y (2004) Three-dimensional inversion of static-shifted Magnetotelluric data. Earth Planets Space 56:239–248

Sasaki Y, Meju MA (2006) Three-dimensional joint inversion for Magnetotelluric resistivity and static shift distributions in complex media. J Geophys Res B Solid Earth 111:art. no. B05101

Schultz A, Weiss C, Urquhart S (2010) Progress toward massively parallel frequency domain 3D EM forward/inverse solutions through domain decomposition on general purpose graphics processors, IAGA WG 1.2 on electromagnetic induction in the earth, 20th workshop abstract, Giza, Egypt, September 18–24

Shanno DF (1970) Conditioning of quasi-Newton methods for function minimization. Math Comput 24:647–656

Simpson F, Bahr K (2005) Practical magnetotellurics. Cambridge.

Siripunvaraporn W, Egbert G (2000) An effcient data-subspace inversion method for 2D magnetotelluric data. Geophysics 65(3):791–803

Siripunvaraporn W, Egbert G (2007) Data space conjugate gradient inversion for 2-D Magnetotelluric data. Geophys J Int 170:986–994

Siripunvaraporn W, Egbert G (2009) WSINV3DMT: vertical magnetic field transfer function inversion and parallel implementation. Phys Earth Planet Int 173:317–329

Siripunvaraporn W, Sarakorn W (2011) An efficient data space conjugate gradient Occam's method for three-dimensional Magnetotelluric inversion. Geophys J Int. doi:10.1111/j.1365-246x.2011.05079.x

Siripunvaraporn W, Egbert G, Lenbury Y (2002) Numerical accuracy of magnetotelluric modeling: a comparison of finite difference approximations. Earth Planets Space 54(6):721–725

Siripunvaraporn W, Uyeshima M, Egbert G (2004) Three-dimensional inversion for Network-Magnetotelluric data. Earth Planets Space 56:893–902

Siripunvaraporn W, Egbert G, Lenbury Y, Uyeshima M (2005a) Three-dimensional Magnetotelluric inversion: data-space method. Phys Earth Plan Int 150:3–14

Siripunvaraporn W, Egbert G, Uyeshima M (2005b) Interpretation of two-dimensional Magnetotelluric profile data with three-dimensional inversion: synthetic examples. Geophys J Int 160:804–814

Smith JT (1996) Conservative modeling of 3-D electromagnetic fields, Part II: biconjugate gradient solution and an accelerator. Geophysics 61:1319–1324

Smith JT, Booker JR (1991) Rapid inversion of two- and three-dimensional magnetotelluric data. J Geophys Res 96:3905–3922

Spichak V (1999) Three-dimensional inversion of MT fields using Bayesian statistics. In: Oristaglio M, Spies B (eds) Three-dimensional electromagnetics. SEG, Tulsa, USA, pp 406–417

Spichak V, Popova I (2000) Artificial neural network inversion of Magnetotelluric data in terms of three-dimensional earth macroparameters. Geophys J Int 142:15–26

Spichak VV, Borisova VP, Fainberg EB, Khalezov AA, Goidina AG (2007) Electromagnetic 3D tomography of the Elbrus volcanic center according to Magnetotelluric and satellite data. J Volcanol Seismol 1:53–66

Streich R (2009) 3D finite-difference frequency-domain modeling of controlled-source electromagnetic data: direct solution and optimization for high accuracy. Geophysics 74(5):F95–F105

Swift CM (1967) A magnetotelluric investigation of electrical conductivity anomaly in the southwestern United States. PhD thesis, MIT, Cambridge, MA

Szarka L, Novak A, Szalai S, Adam A (2006) Imaging experiences in Magnetotellurics and in geoelectrics, 17th international geophys congress & exhibition, November 14–17

Toh H, Honma S (2008) Mantle upwelling revealed by genetic algorithm inversion of the magnetovariational anomaly around Kyushu Island, Japan. J Geophys Res 113:B10103. doi:10.1029/2006JB004891

Tuncer V, Unsworth MJ, Siripunvaraporn W, Craven JA (2006) Exploration for unconformity-type uranium deposits with audiomagnetotelluric data: a case study from the McArthur River mine, Saskatchewan, Canada. Geophysics 71:B201–B209

Türkoğlu E, Unsworth M, Pana D (2009) Deep electrical structure of northern Alberta (Canada): implications for diamond exploration. Can J Earth Sci 46:139–154

Unsworth M, Bedrosian P, Eisel M, Egbert G, Siripunvaraporn W (2000) Along strike variations in the electrical structure of the San Andreas Fault at Parkfield, California. Geophys Res Lett 27:3021–3024

Uyeshima M (2007) EM monitoring of crustal processes including the use of the Network-MT observations. Surv Geophys 28:199–237

Vachiratienchai C, Boonchaisuk S, Siripunvaraporn W (2010) A hybrid finite difference-finite element method to incorporate topography for 2D direct current (DC) resistivity modeling. Phys Earth Plan Int 183(3–4), 426–434

Virginie M, Wanamaker P (2010) Parallelizing a 3D finite difference MT inversion algorithm on a multicore PC using OpenMP. Comput Geosci 36:1384–1387

Vozoff K (1972) The magnetotelluric method in the exploration of sedimentary basins. Geophysics 37: 98–141

Wannamaker PE (1991) Advances in three dimensional magnetotelluric modeling using integral equations. Geophysics 56:1716–1728

Weaver JT, Agarwal AK, Lilley FEM (2000) Characterization of the Magnetotelluric impedance tensor. Geophys J Inter 129:133–142
Weidelt P (1975) EM induction in three dimensional structures. Geophysics 41:85–109
Xiao Q, Cai X, Xu X, Liang G, Zhang B (2010) Application of the 3D Magnetotelluric inversion code in a geologically complex area. Geophys Prospect. doi:10.1111/j.1365-2478.2010.00896.x
Zhanxiang H, Hu Z, Luo W (2010) Mapping reservoirs based on resistivity and induced polarization derived from continuous 3D Magnetotelluric profiling: case study from Qaidam basin, China. Geophysics 75: B25–B33
Zhdanov MS (2002) Geophysical inverse theory and regularization problems. Elsevier, Amsterdam, p 609
Zhdanov MS (2009) Geophysical electromagnetic theory and methods. Elsevier, Amsterdam, p 848
Zhdanov MS, Fang S, Hursan G (2000) Electromagnetic inversion using quasi-linear approximation. Geophysics 65:1501–1513