Chapter 2

COMPRESSION IN 3-D INTEGRAL EQUATION MODELING

Michael S. Zhdanov, Oleg Portniaguine and Gabor Hursan

University of Utah, Salt Lake City, UT 84112, USA

Abstract: The integral equations (IE) method is a powerful tool for forward electromagnetic (EM) modeling. However, due to a dense matrix arising from the IE formulation, practical application of the IE method is limited to modeling of relatively small bodies. The use of a compression technique can overcome this limitation. The compression transformation is formulated as a multiplication by a compression matrix. Using this matrix as a preconditioner to an integral equation, we convert the originally dense matrix of the problem to a sparse matrix, which reduces its size and speeds up computations. Thus, compression helps to overcome practical limitations imposed on the numerical size of the anomalous domain in IE modeling. With the compression, the flexibility of the IE method approaches that of finite-difference (FD) or finite-element (FE) methods, allowing modeling of large-scale conductivity variations.

1. INTRODUCTION

The integral equations (IE) method is a powerful tool for forward electromagnetic (EM) modeling. The basic principles of constructing integral equations in 3-D cases were outlined by Hohmann (1975) and Weidelt (1975). A comprehensive implementation of the IE methods was realized by Xiong (1992) in the SYSEM code. The main advantage of the IE method in comparison with the FD and FE methods is the fast and accurate simulation of compact 3-D bodies in a layered background (Hohmann, 1975; Weidelt, 1975; Wannamaker, 1991; Xiong, 1992). At the same time, the area of the FD and FE methods is modeling of EM fields in complex structures with large-scale conductivity variations. In principle, the IE method can handle these models; however, the tremendous demand on computer resources places practical limits on its use. This happens due to the large dense matrix arising from IE formulation.

Another advantage that the IE method has over the FD and FE methods is its greater suitability for inversion. IE formulation readily contains a sensitivity matrix, which can be recomputed at each inversion iteration at little expense. With FD, in contrast, this matrix has to be established anew on each iteration at a cost equal to the cost of the full forward simulation.

However, for inversion purposes, and for greater flexibility in forward simulations, the IE method has to overcome severe practical limitations imposed on the numerical size of the anomalous domain.

Several approximate methods have been developed recently in this direction. These are: localized nonlinear approximation (Habashi et al., 1993), quasi-linear approximation (Zhdanov and Fang, 1996a,b), quasi-linear series (Zhdanov and Fang, 1997), and quasi-analytical approximation (Zhdanov et al., 2000). At the same time, most of these methods produce approximate solutions, therefore we still seek to develop a rigorous IE forward modeling technique.

In this paper we develop a technique that makes the IE method a more flexible tool for simulations in complex geological structures. This technique is based on applying compression to the solution of IE. Compression is routinely applied in the telecommunications industry for data transmission (Klarke, 1995; Natravali, 1995; Ramstad et al., 1995; Henson et al., 1996; Losano and Laget, 1996; Bhaskaran and Konstantinides, 1997). To decrease the traffic through an information channel, the data are compressed at the transmitting end, passed through the channel in compact form, and decompressed to their original form at the receiving end.

A similar approach can be applied to numerical simulations. That is, original equations are transformed into a compressed domain and the solution is obtained there in a compact form (Portniaguine, 1999).

We introduce a compression matrix and explain the principles of compression with an *interpolation pyramid* in multiple dimensions. A compression matrix is then used as a preconditioner to the discretized IE. This approach is similar to the use of preconditioners for FD solutions to express multi-resolution and multigrid methods in numerical simulations (Bank and Xu, 1994; Manteuffel et al., 1994; McCormick, 1994; Ainsworth et al., 1997).

Using compression, we convert the original dense matrix of the forward problem to a sparse matrix. This reduces the memory required for storage and speeds up computations.

2. FORWARD MODELING WITH 3-D INTEGRAL EQUATIONS

Let us represent a 3-D distribution of conductivity $\tilde{\sigma}$ as a sum of background (normal) complex conductivity $\tilde{\sigma}_b$ and anomalous conductivity $\tilde{\sigma}$, which is non-zero only within local domain D.

We assume that magnetic permeability μ is constant everywhere and is equal to that of a free-space $\mu = 4\pi \times 10^{-7}$ H/m. This model is excited by a harmonic source of radian frequency ω. The complex conductivity includes the effect of displacement currents: $\tilde{\sigma} = \sigma - i\omega\varepsilon$, where σ and ε are electrical conductivity and dielectric permittivity.

The vectors of total electric $\mathbf{E}$ and magnetic $\mathbf{H}$ fields in this model can be presented as a sum of background (normal) and anomalous (scattered) fields:

$$\mathbf{E} = \mathbf{E}^b + \mathbf{E}^a, \qquad \mathbf{H} = \mathbf{H}^b + \mathbf{H}^a, \tag{2.1}$$

The background field $\mathbf{E}^b$ is a field generated by the given sources in the background model $\tilde{\sigma}_b$, and the anomalous $\mathbf{E}^a$ field is cased by the presence of anomalous conductivity $\Delta\tilde{\sigma}$.

The anomalous field is presented as an integral over the excess currents in the

inhomogeneous domain D:

$$\mathbf{E}^{\mathrm{a}}\left(\mathbf{r}_j\right)=\iiint_D \widehat{\mathbf{G}}_E\left(\mathbf{r}_j \mid \mathbf{r}\right)\cdot\mathbf{j}^{\mathrm{a}}(\mathbf{r})\mathrm{d}v, \tag{2.2}$$

$$\mathbf{H}^{\mathrm{a}}\left(\mathbf{r}_j\right)=\iiint_D \widehat{\mathbf{G}}_H\left(\mathbf{r}_j \mid \mathbf{r}\right)\cdot\mathbf{j}^{\mathrm{a}}(\mathbf{r})\mathrm{d}v, \tag{2.3}$$

where $\widehat{\mathbf{G}}_E\left(\mathbf{r}_j \mid \mathbf{r}\right)$ and $\widehat{\mathbf{G}}_H\left(\mathbf{r}_j \mid \mathbf{r}\right)$ are the electric and magnetic Green's tensors defined for an unbounded conductive medium with the background conductivity $\tilde{\sigma}_{\mathrm{b}}$. Excess current $\mathbf{j}^{\mathrm{a}}(\mathbf{r})$ at the point $\mathbf{r}$ is determined by the equation

$$\mathbf{j}^{\mathrm{a}}(\mathbf{r})=\Delta\tilde{\sigma}(r)\left(\mathbf{E}^{\mathrm{b}}(r)+\mathbf{E}^{\mathrm{a}}(r)\right). \tag{2.4}$$

Expression (2.2) becomes an integral equation with respect to anomalous electric field $\mathbf{E}^{\mathrm{a}}(\mathbf{r})$, if point $\mathbf{r}_j$ is inside D.

Inserting Equation (2.4) into (2.2) produces:

$$\mathbf{E}^{\mathrm{a}}\left(\mathbf{r}_j\right)=\iiint_D \widehat{\mathbf{G}}_E\left(\mathbf{r}_j \mid \mathbf{r}\right)\cdot\Delta\tilde{\sigma}(r)\left(\mathbf{E}^{\mathrm{b}}(r)+\mathbf{E}^{\mathrm{a}}(r)\right)\mathrm{d}v. \tag{2.5}$$

Because the background field is known, it is convenient to rewrite (2.5) as:

$$\mathbf{E}^{\mathrm{a}}\left(\mathbf{r}_j\right)=\iiint_D \widehat{\mathbf{G}}_E\left(\mathbf{r}_j \mid \mathbf{r}\right)\cdot\Delta\tilde{\sigma}(r)\mathbf{E}^{\mathrm{a}}(r)\mathrm{d}v+\mathbf{E}^{\mathrm{B}}(r_j), \tag{2.6}$$

where $\mathbf{E}^{\mathrm{B}}(r_j)$ is the Born approximation at point $\mathbf{r_j}$:

$$\mathbf{E}^{\mathrm{B}}\left(\mathbf{r}_j\right)=\iiint_D \widehat{\mathbf{G}}_E\left(\mathbf{r}_j \mid \mathbf{r}\right)\cdot\Delta\tilde{\sigma}(r)\mathbf{E}^{\mathrm{b}}(r)\mathrm{d}v. \tag{2.7}$$

We express vector Equation (2.6) via individual scalar components. Let us denote:

$$\mathbf{E}^{\mathrm{a}}=\left(E_x^{\mathrm{a}}\;\;E_y^{\mathrm{a}}\;\;E_z^{\mathrm{a}}\right)=\left(E_1^{\mathrm{a}}\;\;E_2^{\mathrm{a}}\;\;E_3^{\mathrm{a}}\right), \tag{2.8}$$

where lower index 1 denotes the x-component, 2 denotes the y component, and 3 denotes the z component.

Then, Expression (2.6) breaks into three equations:

$$E_n^{\mathrm{a}}\left(\mathbf{r}_j\right)=\sum_{m=1}^{3}\left(\iiint_D G_{E_{nm}}\left(\mathbf{r}_j \mid \mathbf{r}\right)\cdot\Delta\tilde{\sigma}(r)E_m^{\mathrm{a}}(r)\mathrm{d}v\right)+E_n^{\mathrm{B}}(r_j), \tag{2.9}$$

where $G_{E_{nm}}$ ($n=1,2,3$, and $m=1,2,3$) are the components of the Green's tensor.

For the purpose of computer simulations we must discretize Equation (2.9). To accomplish this, we divide domain D into rectangular cells. Individual cells are denoted D_k, and the total number of cells is N_{c}. We assume that anomalous conductivity is constant within each individual cell, and the cell size is small enough to consider the anomalous electric field to be constant inside the cell (Xiong, 1992).

Under these assumptions the triple integral over the entire domain D (in Expression (2.9)) transforms into the sum of contributions from the individual cells:

$$E_n^{\mathrm{a}}(\mathbf{r_j}) = \sum_{m=1}^{3}\sum_{k=1}^{N_c}\left(\iiint_{D_k} G_{E_{nm}}(\mathbf{r}_j \mid \mathbf{r})\,\mathrm{d}v \cdot \Delta\tilde{\sigma}(\mathbf{r}_k)E_m^{\mathrm{a}}(\mathbf{r}_k)\right) + E_n^{\mathrm{B}}(\mathbf{r}_j), \tag{2.10}$$

where $n = 1,2,3$, $\mathbf{j} = 1,\ldots,N_c$; $\mathbf{r_j}$ is the central point of the cell with the index j, and $\mathbf{r_k}$ is the central point of the cell with the index k.

Equation (2.10) can be written using matrix notations as follows:

$$\mathbf{e} = \widehat{\mathbf{G}}\widehat{\mathbf{S}}\mathbf{e} + \mathbf{b}, \tag{2.11}$$

where $\widehat{\mathbf{G}}$ is the matrix of a size $3N_c \times 3N_c$, of the known Green's tensor integrated over elementary cell D_k, with the scalar components $G_{(l,\,i)}$:

$$G_{(j+(n-1)N_c,\,k+(m-1)N_c)} = \iiint_{D_k} G_{E_{nm}}(\mathbf{r}_j \mid \mathbf{r})\,\mathrm{d}v, \tag{2.12}$$

$\widehat{\mathbf{S}}$ is a sparse diagonal matrix, of a size $3N_c \times 3N_c$, with the diagonal elements $S_{(l,\,l)}$ equal to the known conductivities within each cell:

$$S_{(k+(m-1)N_c,\,k+(m-1)N_c)} = \Delta\tilde{\sigma}(r_k), \tag{2.13}$$

$\mathbf{b}$ is a vector of a length $3N_c$, containing three components of the Born approximation, with the scalar components $b_{(l)}$:

$$b_{(j+(n-1)N_c)} = E_n^{\mathrm{B}}(r_j). \tag{2.14}$$

and $\mathbf{e}$ is a vector of a length $3N_c$, containing three components of the unknown anomalous field E_n^{a} in every cell, with the scalar components $e_{(l)}$:

$$e_{(j+(n-1)N_c)} = E_n^{\mathrm{a}}(\mathbf{r_j}). \tag{2.15}$$

Thus, a forward electromagnetic modeling problem is reduced to numerically solving Equation (2.11) to find the unknown vector $\mathbf{e}$ representing a discretized anomalous electric field inside a domain D. One way of solving (2.11) is by using the block-relaxation method (Xiong, 1992). Here we discuss an alternative method, that is, solving (2.11) using compression.

The main problem with the integral equations method is that in general 3-D cases matrix $\widehat{\mathbf{G}}$ in Equation (2.11) may be very large. Assume that anomalous domain D is a rectangular prism. Each side of the prism is divided by N to produce rectangular prismatic cells. Then the number of cells is $N_c = N^3$. The number of scalar components in matrix $\widehat{\mathbf{G}}$ is $(3 * N_c^3)^2 = 9 * N^6$. We can see that this number grows as the sixth power of N. This growth is the main limiting factor of the integral equation method. If $N = 5$ the problem is small and readily solvable. Yet, for $N = 10$ the size of the problem becomes very large.

By using the compression technique, we may reduce the size of the problem.

3. COMPRESSION MATRIX

First, let us consider the compression with an interpolation pyramid for a 1-D function.

An interpolation pyramid consists of levels (grids), with each level twice as coarse as the previous one. The first and finest grid is that of an original discretized curve.

On each grid we distinguish odd nodes, or *reference nodes*, and even nodes, or *intermediate nodes*. Values at reference nodes can be used to predict (by interpolation) the values at the intermediate nodes. If the curve is smooth, then the difference between predicted and original values is small. That is the key point which enables compression. Take a curve uniformly discretized at $(2N+1)$ points, which we denote as a vector $\mathbf{v}_1$:

$$\mathbf{v}_1 = \{d_1, d_2, d_3, \ldots, d_{2N}, d_{2N+1}\}. \tag{2.16}$$

Consider the transformation of $\mathbf{v}_1$ to $\mathbf{v}_2$, where values at odd points $(1,3,\ldots,2N+1)$ are retained as they were, and values at even points $(2,4,\ldots,2N)$ are transformed into residuals between the predicted (by interpolation from the odd grid) and the original values. For example, points 1 and 3 are retained as they were, and point 2 is transformed as the half-sum of the values at points 1 and 3, minus the original value at point 2. Such transformation can be described by the matrix, which we denote as $\mathbf{W}_1$:

$$\mathbf{v}_2 = \widehat{\mathbf{W}}_1 \mathbf{v}_1. \tag{2.17}$$

Matrix $\widehat{\mathbf{W}}_1$ has the following structure:

$$\begin{bmatrix} 1 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 \\ 1/2 & -1 & 1/2 & \ldots & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & \ldots & 0 & 0 & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & \ldots & 0 & 1/2 & -1 & 1/2 \\ 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.18}$$

Note that matrix $\widehat{\mathbf{W}}_1$ is inverse to itself

$$\widehat{\mathbf{W}}_1 \widehat{\mathbf{W}}_1 = \widehat{\mathbf{I}}$$

which means it is positive definite.

The next level uses a grid that is twice as coarse. Now, the intermediate values have indices $(3,7,11,\ldots,2N-1)$. Again, they are predicted using reference values with indices $(1,5,9,13,\ldots,2N+1)$, and the originals are subtracted from the predictions. This transformation is described by the matrix $\widehat{\mathbf{W}}_2$:

$$\mathbf{v}_3 = \widehat{\mathbf{W}}_2 \mathbf{v}_2. \tag{2.19}$$

where matrix $\widehat{\mathbf{W}}_2$ has the following structure

$$\begin{bmatrix} 1 & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 & 0 \\ 1/2 & 0 & -1 & 0 & 1/2 & \ldots & 0 & 0 & 0 & 0 & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & 0 & 0 & 0 & 0 & \ldots & 1/2 & 0 & -1 & 0 & 1/2 \\ 0 & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & \ldots & 0 & 0 & 0 & 0 & 1 \end{bmatrix}. \tag{2.20}$$

We repeat this process for L pyramid levels:

$$\mathbf{v}_L = \widehat{\mathbf{W}}_{L-1} \ldots \widehat{\mathbf{W}}_2 \widehat{\mathbf{W}}_1 \mathbf{v}_1,$$

and

$$\hat{\mathbf{v}}_L = \widehat{\mathbf{W}}_c \mathbf{v}_1, \tag{2.21}$$

where $\widehat{\mathbf{W}}_c$ is a *compression matrix*:

$$\widehat{\mathbf{W}}_c = \widehat{\mathbf{W}}_{L-1} \ldots \widehat{\mathbf{W}}_2 \widehat{\mathbf{W}}_1. \tag{2.22}$$

The resulting vector $\mathbf{v}_L$ contains only few meaningful values. The rest of the values are close to zero. Thus, vector $\mathbf{v}_L$ can be approximated by a sparse vector. The final stage of compression involves thresholding. We find all values in the vector $\mathbf{v}_L$ which are close to zero (say, less than 1 percent of vector maximum) and set them equal to zero:

$$\mathbf{v}'_L = \mathit{threshold}(\widehat{\mathbf{W}}_c \mathbf{v}_1, 0.01), \tag{2.23}$$

where vector $\mathbf{v}'_L$ is the resulting sparse vector. Operator *threshold* denotes threshold transformation.

4. COMPRESSION IN THREE DIMENSIONS

In multiple dimensions, compression with an interpolation pyramid is a series of subsequent 1-D linear interpolations on coarsening grids. Compressing a 3-D function, for example, we would first reduce values at even nodes in the x-direction, then reduce the values at even nodes at the y-direction, then in the z-direction. After that we would consider a coarse grid, and start the process from the x-direction, etc.

To illustrate how compression is done, let us consider the following example. We take a thin conductive plate, as shown in Figure 1. We divide this plate into 81 blocks $N_x = 9$, $N_y = 9$ and $N_z = 1$. Components of electrical Green's tensor for the cell in the middle of the domain (the influence of the middle cell on all other cells) are shown in Figure 2.

Consider the compression of one component of Green's tensor on this uniform 9×9 grid (shown in Figure 3). In three dimensions, compression with an interpolation pyramid is a series of 1-D linear interpolations on coarsening grids. Assume that the original function (one component of Green's tensor given on 3-D mesh $N_x \times N_y \times N_z$) is denoted as a vector $\mathbf{d}$. First, values at nodes with even x indices $i_x = 2,4,6,8$ (denoted by stars in Figure 3, case a) are predicted by a 1-D linear interpolation in the x direction from values at odd points $i_x = [1,3,5,7,9]$. Prediction is subtracted from true values at even points; odd point values are retained as is. This transformation can be denoted as a linear operation:

$$\mathbf{d}_{x1} = \widehat{\mathbf{W}}_{x1} \mathbf{d} \tag{2.24}$$

where $\widehat{\mathbf{W}}_{x1}$ is the first elementary compression matrix in the x direction. Note that it is a sparse matrix. Portniaguine (1999) considers the structure of a 1-D elementary compression matrix in more detail.

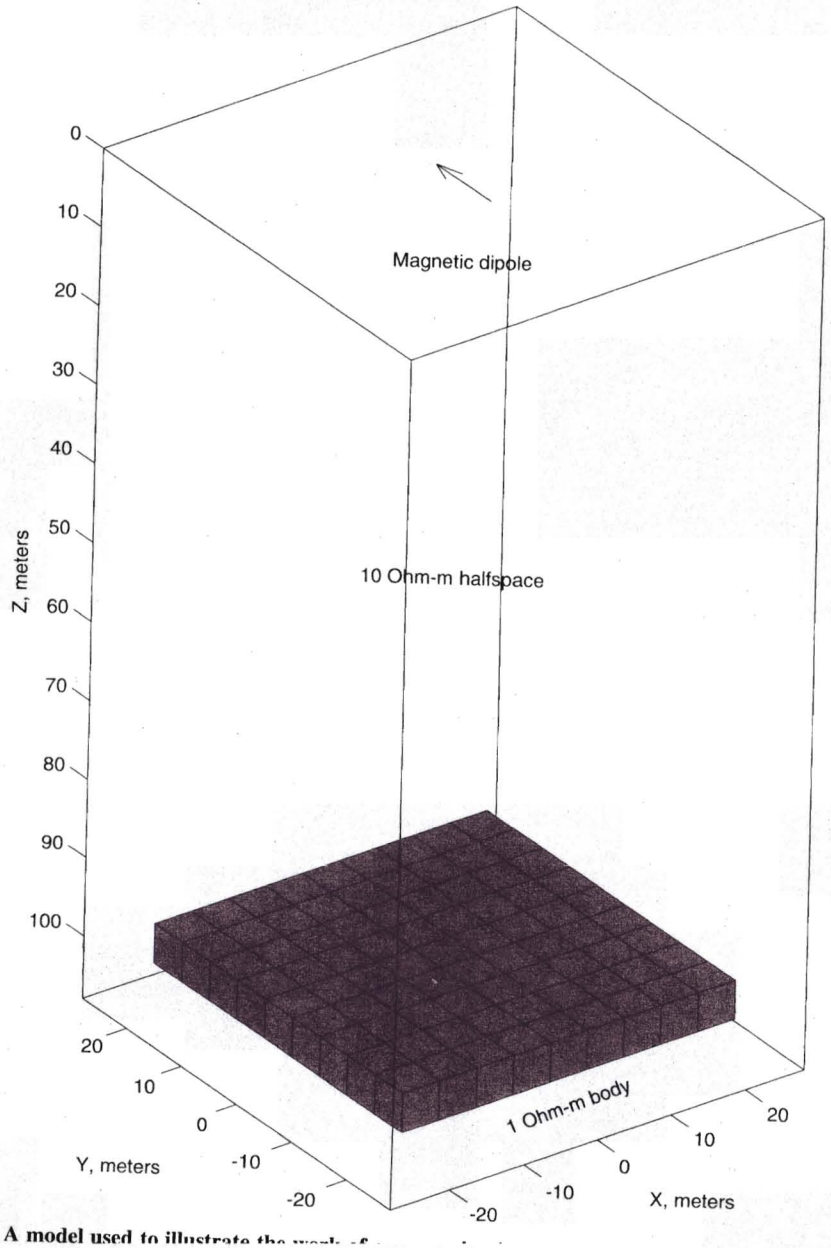


Figure 1. A model used to illustrate the work of

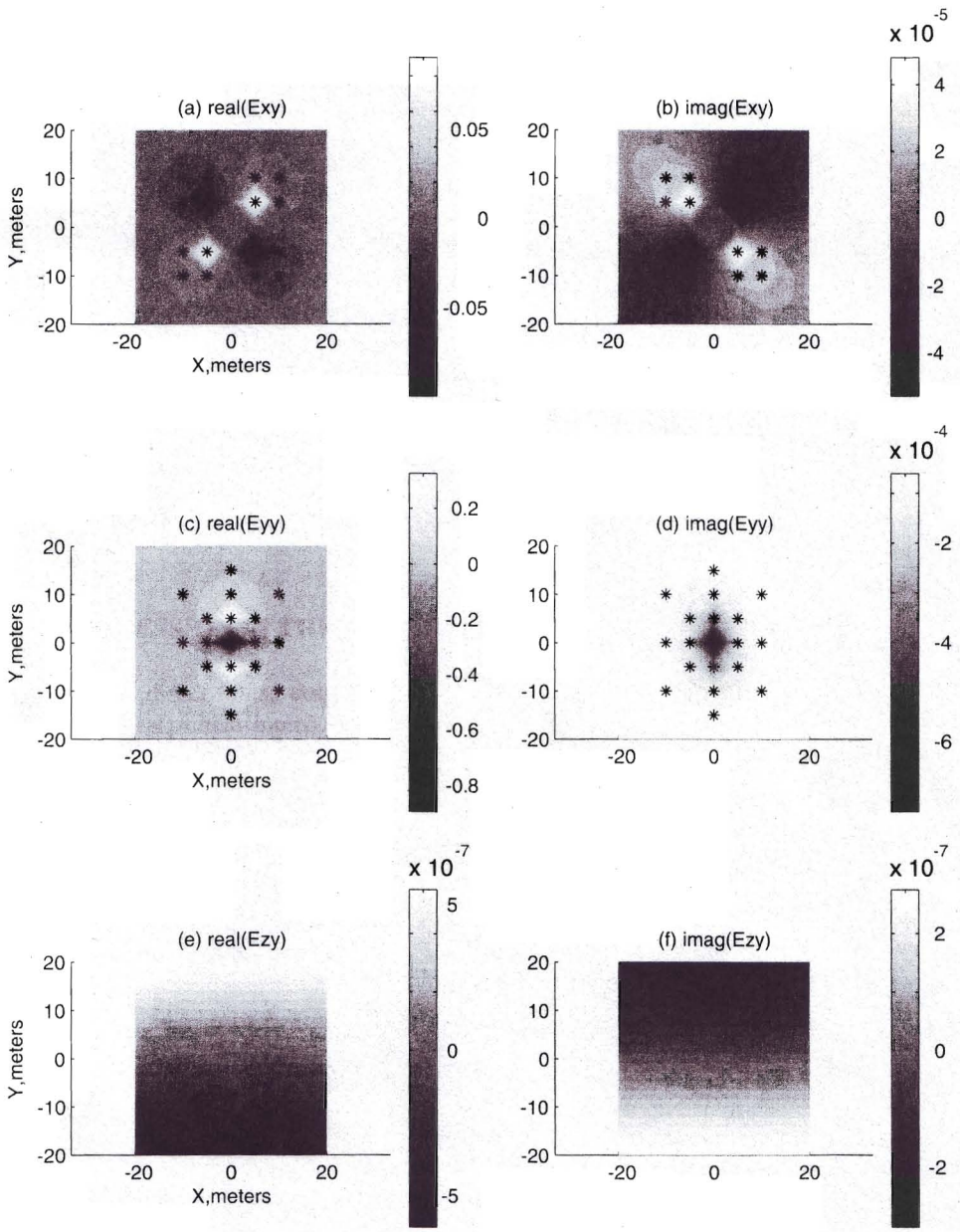
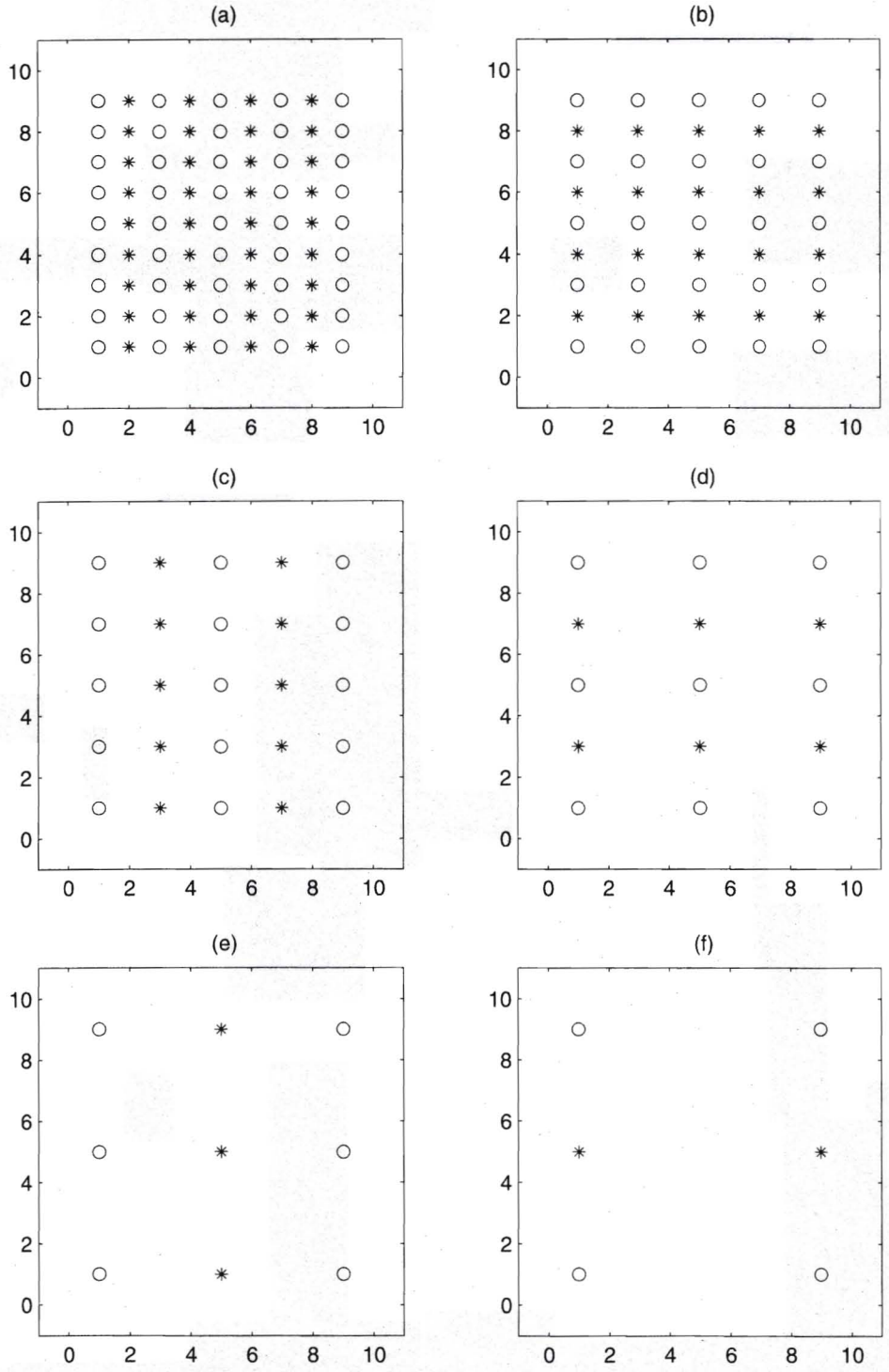


Figure 2. Components (E_{xy}, E_{yy} and E_{zy}) of electric Green's tensor for the geometry shown in Figure 1. Stars denote nodes where the values were retained after compression. The values at all other nodes were thresholded to zero; hence the compressed Green tensor matrix has become sparse.

Figure 3. Compression scheme of a function on a 9 x 9 grid. At every step, the values at starred nodes are predicted by linear interpolation from the values at circled nodes. In panels (a), (c) and (e) 1-D interpolation is done in direction x. In panels (b), (d) and (f) interpolation is done in direction y.


(a)
(b)
(c)
(d)
(e)
(f)
0
2
4
6
8
10

Next, the same transformation is done in direction y. Values at nodes with even y indices $i_y = 2,4,6,8$ (denoted by stars in Figure 3, case b) are predicted by a 1-D linear interpolation in the direction y from the values at odd points $i_y = [1,3,5,7,9]$. Prediction is subtracted from true values at even points; odd point values are retained as is. This transformation, again, can be described as a linear operation:

$$\mathbf{d}_{y1} = \widehat{\mathbf{W}}_{y1}\mathbf{d}_{x1} \tag{2.25}$$

In general cases, the same should be repeated in the direction z after step (2.25):

$$\mathbf{d}_{z1} = \widehat{\mathbf{W}}_{z1}\mathbf{d}_{y1} \tag{2.26}$$

In our specific example the direction z is a singleton, that is $N_z = 1$. For generality, we will retain this direction in our formulas. For the singleton dimension, the compression matrix is simply a unit matrix.

Note that in the case of multiple dimensions, compression on one pyramid level consists of a series of successive 1-D interpolations in all dimensions.

On the next pyramid level we return to the x direction and repeat the same transformation starting on a coarser grid (Figure 3, case c):

$$\mathbf{d}_{x2} = \widehat{\mathbf{W}}_{x2}\mathbf{d}_{z1}. \tag{2.27}$$

The same along the direction y (Figure 3, case d):

$$\mathbf{d}_{y2} = \widehat{\mathbf{W}}_{y2}\mathbf{d}_{x2}, \tag{2.28}$$

and the same along the direction z:

$$\mathbf{d}_{z2} = \widehat{\mathbf{W}}_{z2}\mathbf{d}_{y2}. \tag{2.29}$$

This process should be repeated for as many pyramid levels as necessary. The number of levels N_l in interpolation pyramid is:

$$N_l = 1 + \text{int}(\log_2(\max([N_x, N_y, N_z]))) \tag{2.30}$$

In our example, the third level is the last one. Again, we start from the direction x (mesh is shown in Figure 3, case e):

$$\mathbf{d}_{x3} = \widehat{\mathbf{W}}_{x3}\mathbf{d}_{z2}. \tag{2.31}$$

The same in the y direction (mesh is shown in Figure 3, case f):

$$\mathbf{d}_{y3} = \widehat{\mathbf{W}}_{y3}d_{x3}, \tag{2.32}$$

and in the z direction:

$$\mathbf{d}_{z3} = \widehat{\mathbf{W}}_{z3}\mathbf{d}_{y3}. \tag{2.33}$$

Let us put the whole process together:

$$\mathbf{d}_{z3} = \widehat{\mathbf{W}}_{z3}\widehat{\mathbf{W}}_{y3}\widehat{\mathbf{W}}_{x3}\widehat{\mathbf{W}}_{z2}\widehat{\mathbf{W}}_{y2}\widehat{\mathbf{W}}_{x2}\widehat{\mathbf{W}}_{z1}\widehat{\mathbf{W}}_{y1}\widehat{\mathbf{W}}_{x1}\mathbf{d}. \tag{2.34}$$

In general cases, where the number of compression levels equals N_l, we have:

$$\mathbf{d}_{zN_l} = \prod_{k=1}^{N_l}(\widehat{\mathbf{W}}_{zk}\widehat{\mathbf{W}}_{yk}\widehat{\mathbf{W}}_{xk})\mathbf{d}, \tag{2.35}$$

or,

$$\mathbf{d}_{zN_1} = \widehat{\mathbf{W}}_c \mathbf{d}, \tag{2.36}$$

where $\widehat{\mathbf{W}}_c$ is a sparse compression matrix:

$$\widehat{\mathbf{W}}_c = \prod_{k=1}^{N_1} (\widehat{\mathbf{W}}_{zk} \widehat{\mathbf{W}}_{yk} \widehat{\mathbf{W}}_{xk}). \tag{2.37}$$

If information in the original vector $\mathbf{d}$ is redundant, then interpolation predicts intermediate values accurately. In this case the result of transformation (2.37), vector $\mathbf{d}_{zN_1}$, contains many small values and therefore can be approximated by a sparse vector $\mathbf{d}_c$:

$$\mathbf{d}_c = threshold(\widehat{\mathbf{W}}_c \mathbf{d}, \varepsilon) \tag{2.38}$$

where $threshold(,\varepsilon)$ denotes threshold transformation with the threshold level ε.

5. COMPRESSION AS PRECONDITIONER TO THE INTEGRAL EQUATION

The notion of a compression matrix gives crucial advantage for analytical work and simplifies coding. The advantage of this approach is significant in its application to forward problem solution.

Consider again a discrete 3-D electromagnetic integral Equation (2.11). Full matrix $\widehat{\mathbf{G}}$ in integral Equation (2.11) consists of nine blocks. Each of the blocks is a smaller matrix containing corresponding components of Green's tensor:

$$\widehat{\mathbf{G}} = \begin{bmatrix} \widehat{\mathbf{G}}_{xx} & \widehat{\mathbf{G}}_{xy} & \widehat{\mathbf{G}}_{xz} \\ \widehat{\mathbf{G}}_{yx} & \widehat{\mathbf{G}}_{yy} & \widehat{\mathbf{G}}_{yz} \\ \widehat{\mathbf{G}}_{zx} & \widehat{\mathbf{G}}_{zy} & \widehat{\mathbf{G}}_{zz} \end{bmatrix}. \tag{2.39}$$

Our compression matrix $\widehat{\mathbf{W}}_c$ (2.37) applies to one component of $\widehat{\mathbf{G}}$. Using $\widehat{\mathbf{W}}_c$, we may establish the sparse global compression matrix $\widehat{\mathbf{W}}_g$ to be used as preconditioner to Equation (2.11):

$$\widehat{\mathbf{W}}_g = \begin{bmatrix} \widehat{\mathbf{W}}_c & 0 & 0 \\ 0 & \widehat{\mathbf{W}}_c & 0 \\ 0 & 0 & \widehat{\mathbf{W}}_c \end{bmatrix}. \tag{2.40}$$

Using $\widehat{\mathbf{W}}_g$ as a preconditioner to Equation (2.11), we have

$$\widehat{\mathbf{W}}_g \mathbf{e} = \widehat{\mathbf{W}}_g \widehat{\mathbf{G}} \widehat{\mathbf{S}} \mathbf{e} + \widehat{\mathbf{W}}_g \mathbf{b}. \tag{2.41}$$

Rearranging (2.41), we arrive at

$$(\widehat{\mathbf{W}}_g - \widehat{\mathbf{W}}_g \widehat{\mathbf{G}} \widehat{\mathbf{S}}) \mathbf{e} = \widehat{\mathbf{W}}_g \mathbf{b}. \tag{2.42}$$

Now, we can approximate (2.42) by applying a threshold to $\widehat{\mathbf{W}}_g \widehat{\mathbf{G}}$ and $\widehat{\mathbf{W}}_g \mathbf{b}$:

$$(\widehat{\mathbf{W}}_g - threshold(\widehat{\mathbf{W}}_g \widehat{\mathbf{G}}, \varepsilon) \widehat{\mathbf{S}}) \hat{\mathbf{e}} \approx threshold(\widehat{\mathbf{W}}_g \mathbf{b}, \varepsilon). \tag{2.43}$$

Replacing

$$threshold(\widehat{\mathbf{W}}_g \widehat{\mathbf{G}}, \varepsilon) = \widehat{\mathbf{G}}_c \tag{2.44}$$

and

$$threshold(\widehat{\mathbf{W}}_g\mathbf{b}, \varepsilon) = \mathbf{b}_c,$$

we obtain

$$(\widehat{\mathbf{W}}_g - \widehat{\mathbf{G}}_c\widehat{\mathbf{S}})\mathbf{e} \approx \mathbf{b}_c. \tag{2.45}$$

In the compressed Equation (2.45) all matrices are sparse. Thus, we have managed to approximate the originally full system of equations (2.11) by a sparse system (2.45). This reduces memory requirements and speeds up computations. The downside is the possible loss of accuracy, which is connected to the threshold level.

6. THE ILU PRECONDITIONED CONJUGATE GRADIENT METHOD

Assume we are solving a linear problem:

$$\widehat{\mathbf{A}}\mathbf{m} = \mathbf{d}. \tag{2.46}$$

The objective is to find the parameters $\mathbf{m}$ given the matrix $\widehat{\mathbf{A}}$ and the right-hand side vector $\mathbf{d}$. In our notations, $\widehat{\mathbf{A}}$ stands for $\widehat{\mathbf{W}}_g - \widehat{\mathbf{G}}_c\widehat{\mathbf{S}}$ from Equation (2.45). The vector of parameters $\mathbf{m}$ stands for the electric field $\mathbf{e}$, and the vector of data $\mathbf{d}$ stands for the right-hand side vector $\mathbf{b}_c$ from Equation (2.45).

For 3-D cases, matrix $\widehat{\mathbf{A}}$ is large, so iterative methods have to be employed to solve Equation (2.46).

One of the methods is the conjugate gradient (CG) method (Fletcher, 1981; Gill et al., 1981; Press et al., 1996).

In the framework of the CG method, the solution of (2.46) is found iteratively, according to the following formulas:

$$\begin{aligned}
&\mathbf{l}_i = \widehat{\mathbf{A}}^T\mathbf{r}_{i-1} && \text{(a)}\\
&s_i = \mathbf{l}_i^T\mathbf{l}_i && \text{(b)}\\
&\mathbf{h}_i = \mathbf{l}_i + \mathbf{h}_{i-1}\frac{s_i}{s_{i-1}} && \text{(c)}\\
&\mathbf{f}_i = \widehat{\mathbf{A}}\mathbf{h}_i && \text{(d)}\\
&k_i = \frac{\mathbf{f}_i^T\mathbf{r}_i}{\mathbf{f}_i^T\mathbf{f}_i} && \text{(e)}\\
&\mathbf{m}_i = \mathbf{m}_{i-1} - k_i\mathbf{h}_i && \text{(f)}\\
&\mathbf{r}_i = \mathbf{r}_{i-1} - k_i\mathbf{f}_i && \text{(g),}
\end{aligned} \tag{2.47}$$

where i is the iteration number, $\mathbf{r}$ is the residual vector, $\mathbf{l}$ is the gradient vector, s is its length, $\mathbf{h}$ is the conjugate direction vector in the solution space, $\mathbf{f}$ is its projection to the right-hand side, and k is the step length, a scalar. The starting values (for $i = 0$) are

$$\begin{aligned}
&\mathbf{x}_0 = 0 && \text{(a)}\\
&\mathbf{r}_0 = \widehat{\mathbf{A}}\mathbf{m}_0 - \mathbf{d} = -\mathbf{d} && \text{(b)}\\
&\mathbf{s}_0 = 1 && \text{(c).}
\end{aligned} \tag{2.48}$$

Concerning the performance of (2.47), two principal cases should be noted. The first case is when matrix $\widehat{\mathbf{A}}$ arises from the inverse problem, where $\widehat{\mathbf{A}}$ can be rectangular, and most often, even singular. In this case the CG method converges to the least-squares or minimum norm solution. It requires as many iterations as there are non-zero singular values of $\widehat{\mathbf{A}}$. The case with matrix $\widehat{\mathbf{A}}$ having all rows equal (and therefore only one non-zero singular value) illustrates this property, where CG converges to a solution in just one iteration.

The forward problem represents exactly the opposite case, where the matrix $\widehat{\mathbf{A}}$ is of full rank. This case corresponds to the finite-difference or finite-element solution of a PDE, or, in our case, the compressed IE. That means the problem is well-posed, with a unique solution, and matrix $\widehat{\mathbf{A}}$ is $N \times N$ square matrix.

For this case the standard method (2.47) works very slowly. To fully converge, it requires N iterations, which in 3-D cases is a large number. An illustration to this property is a case when matrix $\widehat{\mathbf{A}}$ is square and strictly diagonal, with all non-zero values on the main diagonal (and therefore non-singular). For this simple case CG requires N iterations to converge.

For the case, when matrix $\widehat{\mathbf{A}}$ arises from the forward problem, we apply the CG method, preconditioned with incomplete LU decomposition of $\widehat{\mathbf{A}}$ (ILU) (Freund, 1992; Van der Vorst, 1992; Chan et al., 1994; Bank and Xu, 1994).

The basic idea of this method can be described as follows. Assume we are given an approximate inverse matrix to $\widehat{\mathbf{A}}$ in the form of LU decomposition. This approximate inverse matrix is called incomplete LU, or ILU decomposition. Later we will consider various forms of ILU decomposition. Now, we concentrate on the advantages of ILU decomposition in the iterative algorithm.

Consider the non-singular matrices $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{U}}$ with the property:

$$\widehat{\mathbf{A}} \approx \widehat{\mathbf{L}}\widehat{\mathbf{U}}, \tag{2.49}$$

It follows from (2.49) that

$$\mathbf{L}^{-1}\widehat{\mathbf{A}}\widehat{\mathbf{U}}^{-1} \approx \widehat{\mathbf{I}}. \tag{2.50}$$

Since the problem (2.46) is well-posed, preconditioning and post-conditioning with any non-singular matrix does not change the solution.

Thus, we replace (2.46) with

$$\widehat{\mathbf{L}}^{-1}\widehat{\mathbf{A}}\widehat{\mathbf{U}}^{-1}\widehat{\mathbf{U}}\mathbf{m} = \widehat{\mathbf{L}}^{-1}\mathbf{d}, \tag{2.51}$$

and introduce new variables

$$\mathbf{x} = \widehat{\mathbf{U}}\mathbf{m}, \tag{2.52}$$

$$\mathbf{b} = \widehat{\mathbf{L}}^{-1}\mathbf{d}, \tag{2.53}$$

arriving at

$$\widehat{\mathbf{L}}^{-1}\widehat{\mathbf{A}}\widehat{\mathbf{U}}^{-1}\mathbf{x} = \mathbf{b}. \tag{2.54}$$

In Equation (2.54) the matrix of the problem is stored as a factorization of three matrices. The standard CG algorithm is applied to (2.54). In this case, the matrix of

the problem is close to the unit matrix $\widehat{\mathbf{I}}$, according to (2.50). Since it is equal to $\widehat{\mathbf{I}}$ only approximately, the CG still requires N iterations to fully converge. But the rate of convergence is much faster than for the original algorithm without preconditioning.

Note also, that the application of the inverse triangular matrix to a vector can be readily implemented via Gaussian backsubstitution. Let us denote Gaussian backsubstitution as backslash operation '\':

$$\widehat{\mathbf{L}}^{-1}\mathbf{d} = \widehat{\mathbf{L}}\backslash\mathbf{d} \tag{2.55}$$

We now cast the ILU preconditioned CG algorithm as follows:

$$\begin{aligned}
&\mathbf{l}_i = \widehat{\mathbf{U}}^{\mathrm{T}}\backslash(\widehat{\mathbf{A}}^{\mathrm{T}}(\mathbf{L}^{\mathrm{T}}\backslash\mathbf{r}_{i-1})) && \text{(a)}\\
&s_i = \mathbf{l}_i^{\mathrm{T}}\mathbf{l}_i && \text{(b)}\\
&\mathbf{h}_i = \mathbf{l}_i + \mathbf{h}_{i-1}\frac{s_i}{s_{i-1}} && \text{(c)}\\
&\mathbf{f}_i = \mathbf{L}\backslash(\widehat{\mathbf{A}}(\widehat{\mathbf{U}}\backslash\mathbf{h}_i)) && \text{(d)}\\
&k_i = \frac{\mathbf{f}_i^{\mathrm{T}}\mathbf{r}_i}{\mathbf{f}_i^{\mathrm{T}}\mathbf{f}_i} && \text{(e)}\\
&\mathbf{x}_i = \mathbf{x}_{i-1} - k_i\mathbf{h}_i && \text{(f)}\\
&\mathbf{r}_i = \mathbf{r}_{i-1} - k_i\mathbf{f}_i && \text{(g),}
\end{aligned} \tag{2.56}$$

where initial values are:

$$\begin{aligned}
&\mathbf{x}_0 = 0 && \text{(a)}\\
&\mathbf{r}_0 = -\widehat{\mathbf{L}}\backslash\mathbf{d} && \text{(b)}\\
&\mathbf{s}_0 = 1 && \text{(c).}
\end{aligned} \tag{2.57}$$

After the method converges to the solution $\mathbf{x}_n$, we have to return to original space of $\mathbf{m}$ using

$$\mathbf{m} = \widehat{\mathbf{U}}\backslash\mathbf{x}_n \tag{2.58}$$

If we use a complete LU decomposition in the algorithm, it will converge in one step. The downside of this technique, however, is the need to compute full LU, which may be very costly for large matrices. In addition, for a sparse matrix $\widehat{\mathbf{A}}$, the complete decomposition produces full matrices $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{U}}$.

It is possible, to some extent, to avoid these difficulties using incomplete LU decomposition with the partial fill-up. The complete decomposition of $\widehat{\mathbf{A}}$ often produces many small terms in $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{U}}$. The ILU decomposition with the partial fill-up is based on thresholding these matrices, producing sparse $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{U}}$.

This method performs faster than the complete LU decomposition, because the construction of incomplete LU is faster. Still, precomputing and storing matrices $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{U}}$ along with the matrix $\widehat{\mathbf{A}}$ introduces significant overhead into the algorithm.

Another form of ILU decomposition, which we call 'compositional ILU', does not have these disadvantages. Below, we discuss it in detail.

We consider separate parts of matrix $\widehat{\mathbf{A}}$, namely the lower part $\widehat{\mathbf{L}}_{\mathbf{a}}$ and the upper part $\widehat{\mathbf{U}}_{\mathbf{a}}$ (without the diagonal), and the diagonal $\widehat{\mathbf{D}}$:

$$\widehat{\mathbf{A}} = \widehat{\mathbf{L}}_{\mathbf{a}} + \widehat{\mathbf{D}} + \widehat{\mathbf{U}}_{\mathbf{a}}. \tag{2.59}$$

If matrix $\widehat{\mathbf{A}}$ is sparse and diagonally dominant, then the ILU property holds:

$$\widehat{\mathbf{A}} \approx \widehat{\mathbf{L}}\widehat{\mathbf{U}}, \tag{2.60}$$

where, if we define

$$\widehat{\mathbf{L}} = \widehat{\mathbf{L}}_{\mathbf{a}} + \widehat{\mathbf{D}}, \tag{2.61}$$

then

$$\widehat{\mathbf{U}} = \widehat{\mathbf{D}}^{-1}\widehat{\mathbf{U}}_{\mathbf{a}} + \widehat{\mathbf{I}}. \tag{2.62}$$

If $\widehat{\mathbf{A}}$ is diagonally dominant, from (2.61) and (2.62) it is clear that matrices $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{U}}$ are diagonally dominant as well. Since they are triangular, they are non-singular as well.

Equation (2.60) is an incomplete factorization of the matrix $\widehat{\mathbf{A}}$. An important advantage of this equation over other types of ILU decompositions is that it is computationally cheap, since it requires only separation of upper and lower parts of $\widehat{\mathbf{A}}$ and inversion of the diagonal matrix $\widehat{\mathbf{D}}$. Moreover, compositional ILU does not require extra storage space for saving $\widehat{\mathbf{A}}$ and separately $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{U}}$, since we can represent matrix $\widehat{\mathbf{A}}$ via matrices $\widehat{\mathbf{L}}$ and $\widehat{\mathbf{U}}$ as

$$\widehat{\mathbf{A}} = \widehat{\mathbf{D}}\widehat{\mathbf{U}} + \widehat{\mathbf{L}} - \widehat{\mathbf{D}}. \tag{2.63}$$

The CG algorithm using (2.63) becomes

$$\begin{aligned}
&\mathbf{t}_i = \mathbf{r}^{\mathrm{T}}\widehat{\mathbf{L}}^{-1}\widehat{\mathbf{D}} && \text{(a)}\\
&\mathbf{l}_i = (\mathbf{t}_i + (\mathbf{r}^{\mathrm{T}} - \mathbf{t}_i)\widehat{\mathbf{U}}^{-1})^{\mathrm{T}} && \text{(b)}\\
&s_i = \mathbf{l}_i^T \mathbf{l}_i && \text{(c)}\\
&\mathbf{h}_i = \mathbf{l}_i + \mathbf{h}_{i-1}\frac{s_i}{s_{i-1}} && \text{(d)}\\
&\mathbf{p}_i = \widehat{\mathbf{U}}^{-1}\mathbf{h}_i && \text{(e)}\\
&\mathbf{f}_i = \mathbf{p}_i + \widehat{\mathbf{L}}^{-1}\widehat{\mathbf{D}}(\mathbf{h} - \mathbf{p}_i) && \text{(f)}\\
&k_i = \frac{\mathbf{f}_i^{\mathrm{T}}\mathbf{r}_i}{\mathbf{f}_i^{\mathrm{T}}\mathbf{f}_i} && \text{(g)}\\
&\mathbf{x}_i = \mathbf{x}_{i-1} - k_i\mathbf{h}_i && \text{(h)}\\
&\mathbf{r}_i = \mathbf{r}_{i-1} - k_i\mathbf{f}_i && \text{(i),}
\end{aligned} \tag{2.64}$$

Note that algorithm (2.64) requires two backsubstitutions for matrix $\widehat{\mathbf{U}}^{-1}$ and two for $\widehat{\mathbf{L}}^{-1}$ (items (a,b,e,f) in algorithm (2.64)). This is equivalent of two matrix multiplications with $\widehat{\mathbf{A}}$.

The performance of the CG method, preconditioned with the compositional ILU, is better, if the degree of sparsity is higher, or diagonal dominance of $\widehat{\mathbf{A}}$ is stronger, since these properties improve the accuracy of incomplete factorization (2.60). This is exactly the case with finite-difference and finite-element problems, and also with the compressed integral equation method.

It should be noted that if $\widehat{\mathbf{A}}$ is not diagonally dominant, the method still works because Equation (2.54) is exact. It converges slower, however.

7. MODELING EXAMPLES

We have performed a comparison of compressed and non-compressed problem solution for a model shown in Figure 4. This is Model 3D-1 of COMMEMI project, excited by a rectangular loop operating at 10 Hz.

The experiments were performed with the body divided to 256, 512 and 1024 cells. For each case, three experiments with different degrees of compression were performed. The non-compressed case corresponds to 0 threshold level. Compressed cases include 0.0001 and 0.001 threshold levels.

The experiments were performed on ULTRA SPARC workstation with 160 MHz processor frequency and 128 Mbytes of memory. Table 1 summarizes the result of comparison, in terms of memory usage, CPU time and accuracy of the solution. The accuracy was measured as the L_2 norm of the difference between the compressed and uncompressed solutions (anomalous field inside the body) for the same number of cells.

We can see that compression speeds up the solution up to two orders of magnitude and decreases memory requirements one order of magnitude. The drawback, however, is loss of accuracy. Higher threshold level leads to less accurate solution. The worst case

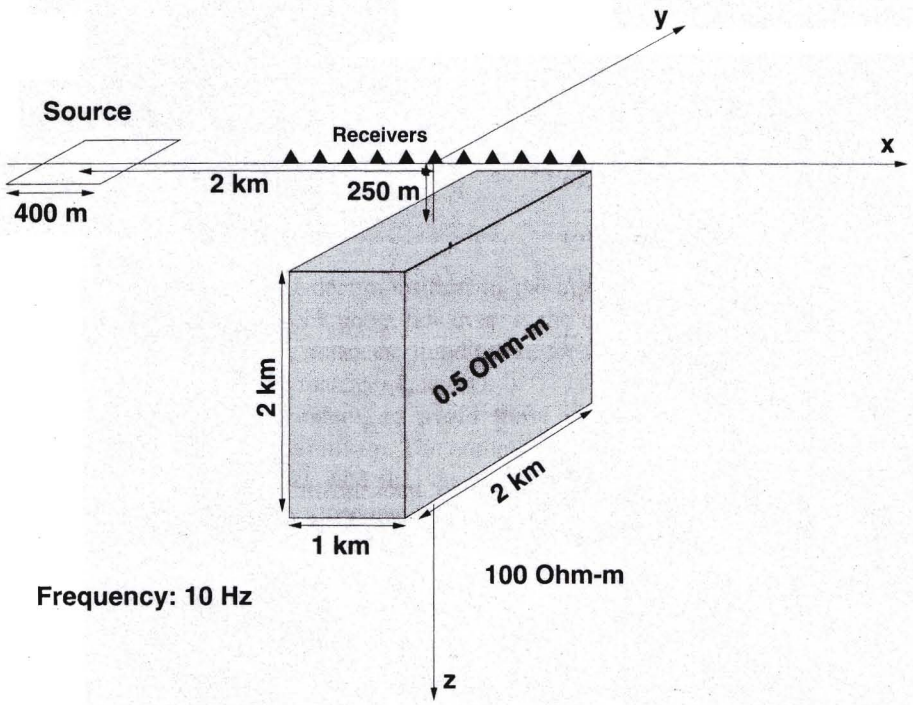


Figure 4. A 3-D geoelectrical model of a conductive rectangular body embedded in a homogeneous half-space. The model is excited by a horizontal rectangular loop source. The magnetic field is measured at the receivers marked by triangles along the x axis.

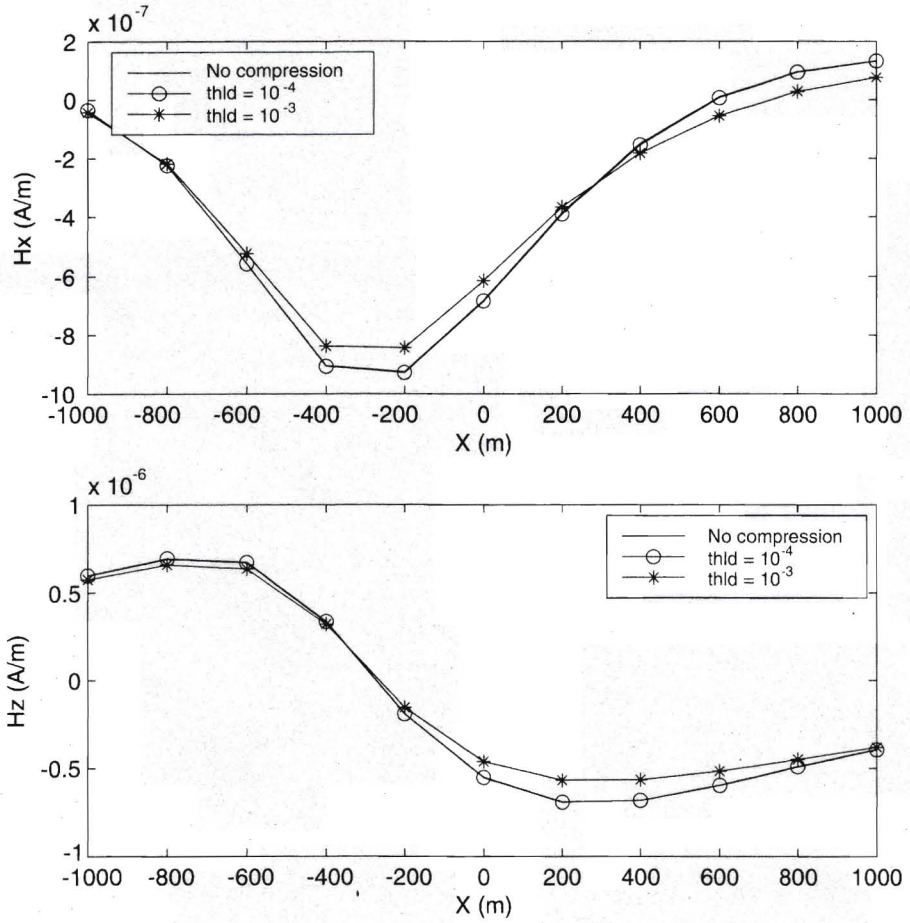


Figure 5. Behavior of the anomalous magnetic field obtained by IE solution without and with compression using different threshold levels.

here is 3% accuracy loss. The difference between the non-compressed solution and the solution compressed with 0.001 threshold level is shown in Figure 5.

Another model, consisting of two conductive bodies, is shown in Figure 6. The observation array is a cross-borehole frequency-domain system. It consists of twenty receivers located in one well. Twenty transmitters located in another well operate at 33 kHz frequency. Transmitters are shown as circles, and receivers are shown as stars.

To build such a model, we first establish a large grid that covers the entire domain between boreholes (Figure 7). The discretized integral equation for this domain is given by Formula (2.11). After compression, the size of the problem matrix was reduced in about five times. The solution of this forward EM problem (an axial magnetic field component) for a cross-borehole method is shown in Figure 8. Note that existing versions of the IE codes without compression cannot compute this model.

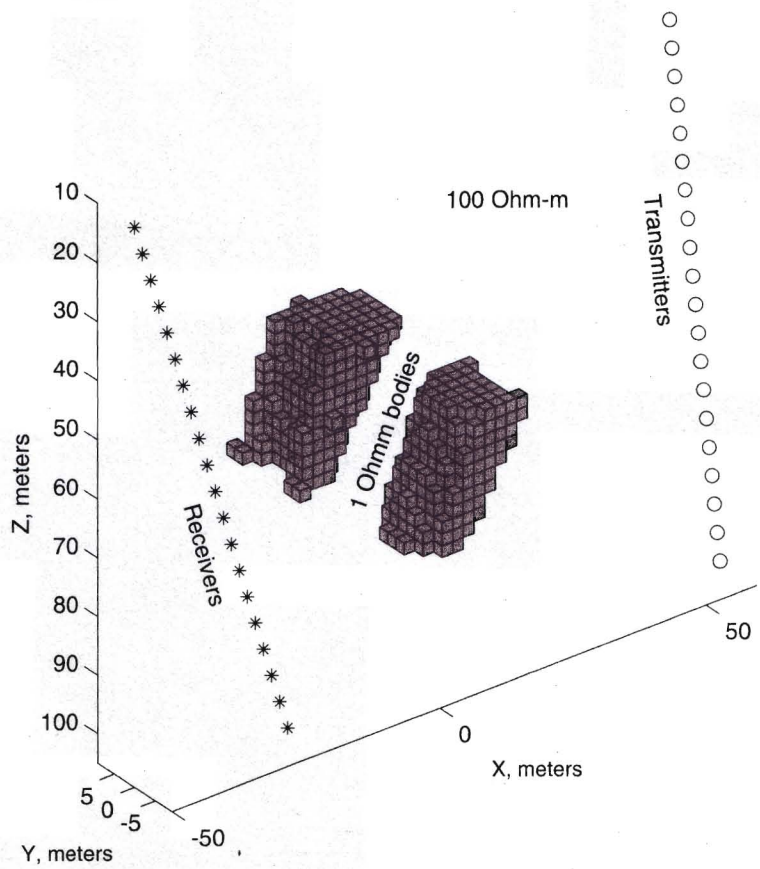


Figure 6. A model with two local conductive bodies and a cross-borehole observation system. Stars show receivers, circles show transmitters.

Table 1. Results of comparison

Threshold level	256 cells	512 cells	1024 cells
Memory usage, megabytes			
0	9.44	37.75	151
0.0001	5.47	16.61	46.81
0.001	1.52	4.81	11.22
CPU time, seconds			
0	38	538	16047
0.0001	21	188	1952
0.001	6.58	53	203
Relative error			
0	0	0	0
0.0001	0.0009	0.0033	0.0038
0.001	0.013	0.028	0.032

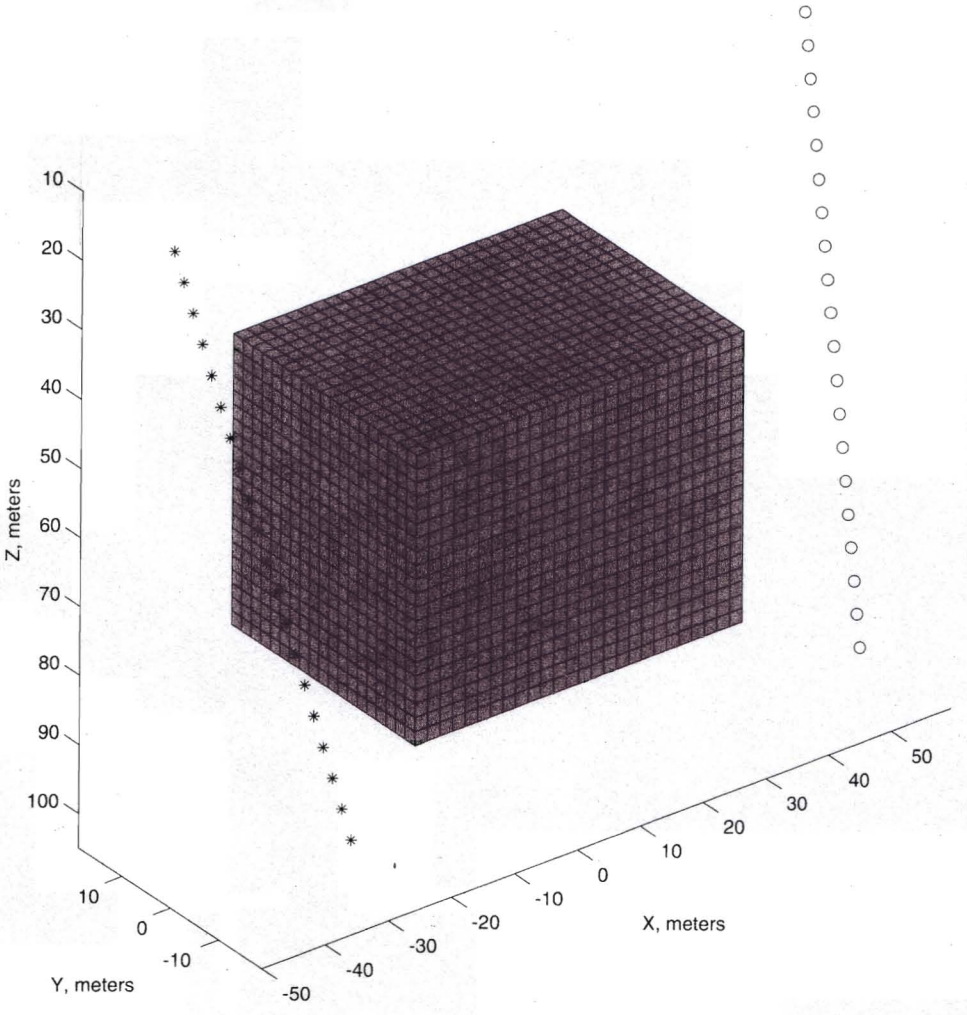


Figure 7. A domain grid for two local bodies and a cross-borehole observation system for the model with two conductive bodies shown in Figure 6. Most of the anomalous conductivity values for the cells in this grid are equal to zero. The whole grid, which is shown in this figure, is used to set up the model. This allows greater flexibility of the model's geometry. However, the anomalous field values are found only inside the bodies shown in Figure 6.

8. CONCLUSIONS

We have demonstrated the application of compression to the solution of a 3-D EM forward problem with IE. Numerical study indicates that the method can be applied to speed up computations and enable solution of larger problems.

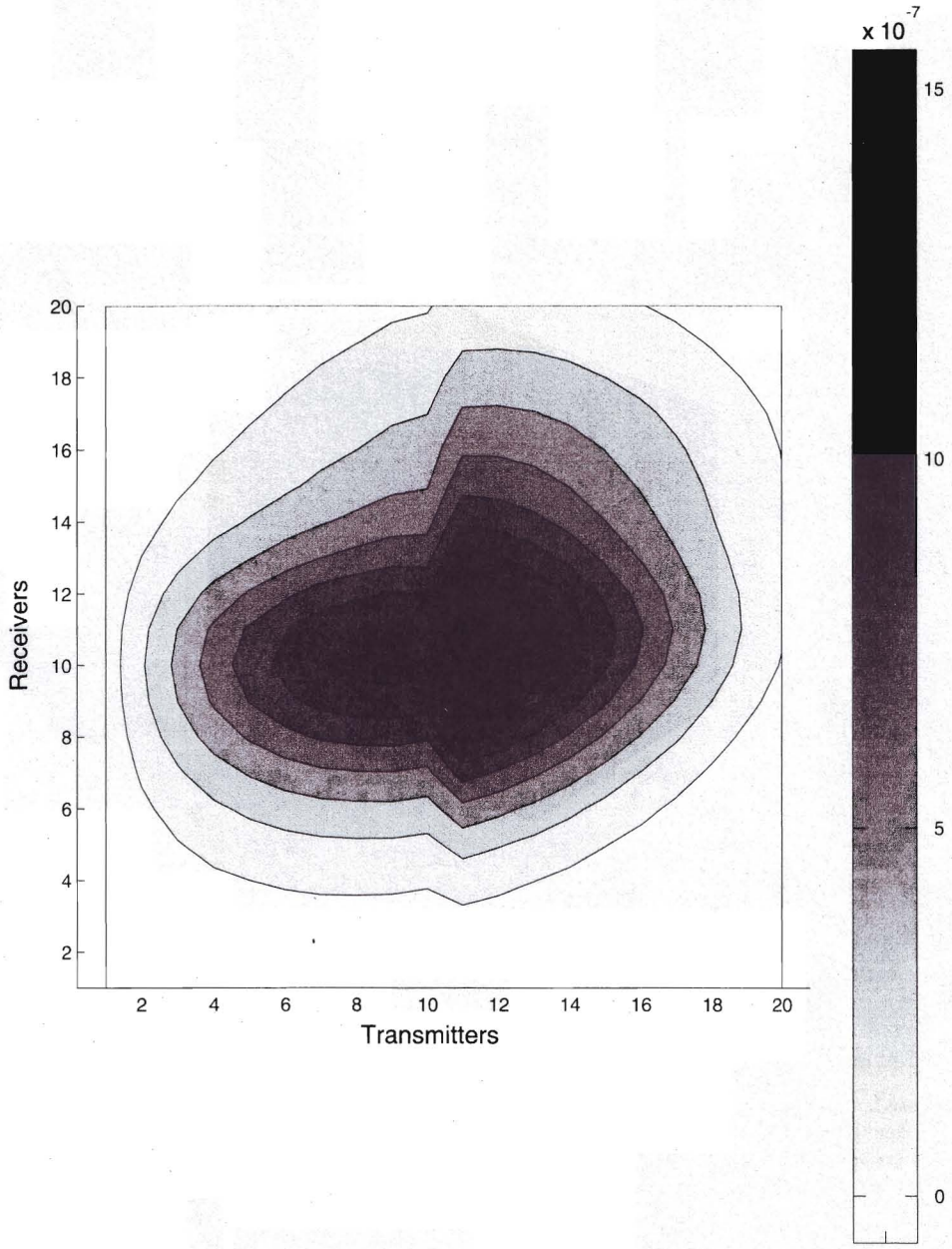


Figure 8. Axial magnetic field, the result of cross-borehole simulation for a model with two anomalous bodies shown in Figure 6. Gray scale shows magnetic field magnitude in A/m.

As a result, we have developed a new generation of IE methods that possess flexibility in forward simulations compatible with the flexibility of FD modeling, but preserves the advantages of IE techniques. There are several promising directions for further research.

It seems possible to develop fast exact solutions based on iterative series consisting of models compressed with different levels of accuracy. It is also possible to apply compression to build fast 3-D inverse solutions.

ACKNOWLEDGEMENTS

The financial support for this work was provided by the National Science Foundation under the grant No. ECS-9987779. The authors also acknowledge the support of the University of Utah Consortium for Electromagnetic Modeling and Inversion (CEMI), which includes Advanced Power Technologies Inc., Baker Atlas Logging Services, BHP Minerals, ExxonMobil Upstream Research Company, INCO Exploration, Japan National Oil Corporation, MINDECO, Naval Research Laboratory, Newmont Gold Company, Rio Tinto, Shell International Exploration and Production, Schlumberger–Doll Research, Unocal Geothermal Corporation, and Zonge Engineering.

REFERENCES

Ainsworth, M., Levesley, J., Ligth, W.A. and Marletta, M., 1997. Wavelets, Multilevel Methods and Elliptic PDEs. Oxford Science Publications, 302 pp.

Bank, R.E. and Xu, J., 1994. The hierarchical basis multigrid method and incomplete LU decomposition. In: D. Keyes and J. Xu (Eds.), Seventh International Symposium on Domain Decomposition Methods for Partial Differential Equations. AMS, Providence, RI, pp. 163–173.

Bhaskaran, V. and Konstantinides, K., 1997. Image and Video Compression Standards: Algorithms and Architectures. Kluwer Academic Publishers, Hingham, MA, 454 pp.

Chan, T.F., Gallopoulos, E. and Simoncini, V., 1994. A quasi-minimum residual variant of the Bi-CGSTAB algorithm for non-symmetric systems. SIAM J. Sci. Statist. Comput., 15, 338–347.

Fletcher, R., 1981. Practical Methods of Optimization. Wiley, New York, NY, 680 pp.

Freund, R., 1992. Conjugate gradient type methods for linear systems with complex symmetric coefficient matrices. SIAM J. Sci. Statist. Comput., 13, 425–448.

Gill, P., Murray, W. and Wright, M., 1981. Practical Optimization. Academic Press, London, 561 pp.

Henson, V., Limber, M., McCormick, S. and Robinson, B., 1996. Multilevel image reconstruction with natural pixels. SIAM J. Sci. Statist. Comput., 17, 193–216.

Habashi, T.M., Groom, R.W. and Spies, B.R., 1993. Beyond the Born and Rytov approximations: a non-linear approach to electromagnetic scattering. J. Geophys. Res., 98, 1759–1775.

Hohmann, G.W., 1975. Three-dimensional induced polarization and EM modeling. Geophysics, 40, 309–324.

Klarke, R.J., 1995. Digital Compression of Still Images and Video. Academic Press, London, 453 pp.

Losano, V. and Laget, B., 1996. Fractional pyramids for color image segmentation. Proceedings of the IEEE Southwest Symposium on Image Analysis and Interpretation, April 8–9, San Antonio, TX, pp. 13–17.

McCormick, S., 1994. Multilevel adaptive methods for elliptic eigenproblems: two-level convergence theory. SIAM J. Numer. Anal., 31, 1731–1745.

Manteuffel, T., McCormick, S., Morel, J., Oliviera, S. and Yang, G., 1994. A fast multigrid algorithm for isotropic transport problems, part I. Pure scattering. SIAM J. Sci. Statist. Comput., 15, 474–493.

Natravali, A.N., 1995. Digital Pictures: Representation, Compression and Standards. AT&T Bell Laboratories, 686 pp.

Portniaguine, O., 1999. Image Focusing and Data Compression in the Solution of Geophysical Inverse Problems. PhD dissertation, University of Utah, 116 pp.

Press, W.H., Teukolsky, S.A., Vetterling, W.T. and Flannery, B.P., 1996. Numerical Recipes in C, The Art of Scientific Computing. Cambridge University Press, 785 pp.

Ramstad, T.A., Aase, S.O. and Husoy, J.H., 1995. Subband Compression of Images: Principles and Examples. Elsevier, Amsterdam, 379 pp.

Van der Vorst, H.A., 1992. Bi-CGSTAB: a fast and smoothly convergent variant of Bi-CG for the solution of non-symmetric linear systems. SIAM J. Sci. Statist. Comput., 13, 631–644.

Wannamaker, P.E., 1991, Advances in three-dimensional magnetotelluric modeling using integral equations. Geophysics, 56, 1716–1728.

Weidelt, P., 1975. EM induction in three-dimensional structures. Geophysics, 41, 85–109.

Xiong, Z., 1992. EM modeling of three-dimensional structures by the method of system iteration using integral equations. Geophysics, 57, 1556–1561.

Zhdanov, M.S. and Fang, S., 1996a. Quasi-linear approximation in 3-D EM modeling. Geophysics, 61, 646–665.

Zhdanov, M.S. and Fang, S., 1996b. 3-D quasi-linear electromagnetic inversion: Radio Sci., 31, 741–754.

Zhdanov, M.S. and Fang, S., 1997. Quasi-linear series in 3-D EM modeling. Radio Sci., 32, 2167–2188.

Zhdanov, M.S., Dmitriev, V.I., Fang, S. and Hursan, G., 2000. Quasi-analytical approximations and series in electromagnetic modeling. Geophysics, 65, 1746–1757.