AD-A214 460

DEPARTMENT OF MATERIALS SCIENCE AND MINERAL ENGINEERING

AIRBORNE ELECTROMAGNETIC SENSING OF SEA ICE THICKNESS

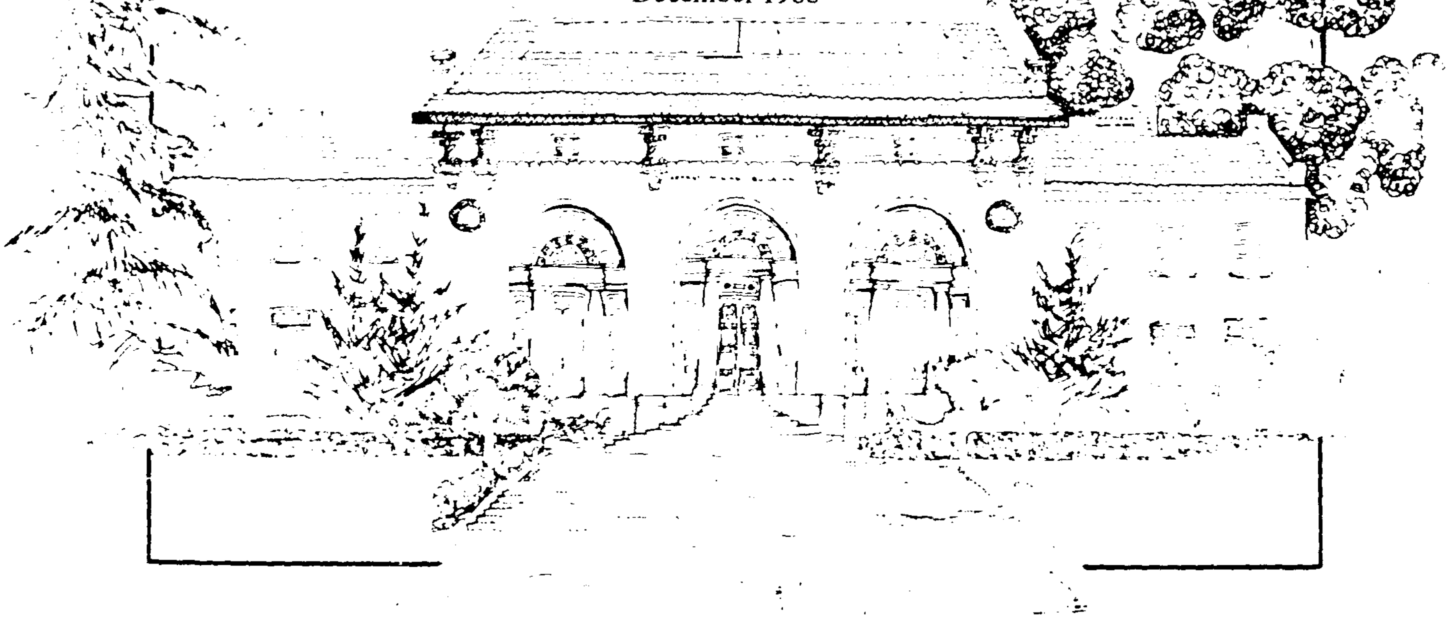
Final Report
prepared for NORDA
Under U.C.B. contract #N00014-87-K-6005
482456-25867 (GF-25867)

by

A. Becker and G. Liu

Engineering Geoscience

December 1988



Hearst Mining Building
University of California, Berkeley

89 11 16 114

REPORT DOCUMENTATION PAGE

Form Approved OMB No. 0704-0188

1. Agency Use Only *(Leave blank).*

2. Report Date.
Dec. 14, 1988

3. Report Type and Dates Covered.
Final

4. Title and Subtitle.
Airborne Electromagnetic Sensing of Sea Ice Thickness

5. Funding Numbers.
Program Element No DE63704
Project No R129905
Task No 300
Accession No DN257087

6. Author(s).
A. Becker and G. Lire

7. Performing Organization Name(s) and Address(es).
Department of Materials Science and Mineral Engineering
Hearst Mining Building
University of California, Berkeley

8. Performing Organization Report Number.
CR N00014-87-K-6005

9. Sponsoring/Monitoring Agency Name(s) and Address(es).
Naval Ocean Research and Development Activity
Code 311
SSC, MS 39529-5004

10. Sponsoring/Monitoring Agency Report Number.
CR N00014-87-K-6005

11. Supplementary Notes.

12a. Distribution/Availability Statement.
Approved for public release; distribution is unlimited.

12b. Distribution Code.

13. Abstract *(Maximum 200 words)*

14. Subject Terms.
electronic/electrical engineering, physical oceanography
snow ice and permafrost

15. Number of Pages.
121

16. Price Code.

17. Security Classification of Report.
Unclassified

18. Security Classification of This Page.
Unclassified

19. Security Classification of Abstract.
Unclassified

20. Limitation of Abstract.

Summary

A conventional frequency domain helicopter-borne electromagnetic (HEM) system can be used to map sea ice keels with a reasonable degree of accuracy. A preliminary interpretation of the acquired data can be made manually with the help of a nomogram or automated with the use of a table look-up routine on a small computer. Such data may also be more accurately interpreted with the use of an adaptation of Occam's inversion. This scheme allows for the practical non-uniqueness of the inverse solution but selects the smoothest keel shape that is consistent with the field data. The inversion method is much more computationally intensive than the table look-up technique. While the latter can be implemented on a small computer to form an interactive in-flight interpretation system, the inversion technique involves many forward computations and, for the present, is best reserved for past flight data analysis. It is possible that this difficulty can be resolved with the use of specialized computing equipment.

In the strict sense both the proposed interpretation techniques are only suitable for use on data acquired over two dimensional features whose strike length (measured in a direction perpendicular to the flight line) is much greater than the flight height. An examination of the anomalies for three-dimensional keels however, reveals that good data interpretation is possible whenever the keel strike length exceeds the system height by a factor of three.

Table of Contents

Table of Illustrations

Airborne Electromagnetic Mapping of Sea Ice Keels

Alex Becker and Guimin Liu

1.0 Introduction

Sea ice is a unique environment encountered in most Arctic work. This includes the transportation of vehicles through ice-covered waters, the construction of offshore drilling structures, and the safe operation of submarines. In such circumstances, knowledge of the thickness and properties of sea ice is important. This is true especially for ice keels, which constitute a local downward indentation of the ice-water interface.

Recently, Canpolar consultants (1985) reviewed the possible techniques for remotely measuring sea ice thickness, among these they included airborne impulse radar and airborne electromagnetics. The impulse radar technique has been very successful in some areas (Kovacs and Morey, 1985). It may however, occasionally yield erroneous data when saline moisture zones exist within the ice cover. Becker et al. (1983), on the other hand, examined the feasibility of using frequency-domain airborne electromagnetics to determine sea ice thickness. At low frequencies, sea ice is practically transparent to electromagnetic waves and the observed secondary magnetic field can be used to estimate the distance from the EM system boom to the ice-water interface. At the same time, a laser altimeter installed on the boom measures the distance to the surface of the ice, so that the difference of these two distances is the sea ice thickness. This technique was successfully used in mapping the average sea ice thickness in Prudhoe Bay, Alaska, but it failed in an area of a multi-year pressure keel because of the inappropriate one-dimensional (1-D) techniques used to interpret data (Kovacs, et al., 1986; Becker, et al., 1987). In order to recover e geometry of the keel, two- or three-dimensional interpretation techniques are required.

In this report we primarily concern ourselves with the development of techniques for interpreting AEM data for two-dimensional keels which are assumed to be infinitely long in a direction perpendicular to the flight line. The computational methods used to construct the necessary forward solutions are outlined in the Appendix. These serve either for the construction of interpretation nomograms or in the impletation of a numerical inversion scheme. In order to establish the validity of the two-dimensional interpretation scheme we also examine the effects of finite keel length on the observable anomalies.

In terms of computer time, the modeling technique we employ is particularly well suited for the interpretation of the AEM data collected over sea ice keels. Using the conventional finite element method, 30 minutes CRAY II CPU time is required to compute the AEM system response along a traverse line over an ice keel. However, for a similar problem, using an approach rooted in potential theory, the computation takes about 10 seconds on IBM 3090 (equivalent to about 3 seconds on CRAY) with a gain more than two orders of magnitude in speed. We will show that the assumption of a perfectly conducting sea water model is not a significant drawback since it can be used for any system operation frequency greater than 30 KHz.

2.0 AEM Anomalies for Ice Keels

Let us first briefly examine the airborne electromagnetic response to two-dimensional sea ice keels as a function of their size and shape (cf. Fig.1). For these calculations, the electromagnetic system has a coil separation of 6.5m and is "flown" 25m above the upper ice surface which is flat. With the exception of the zone containing the keel, the sea ice is 5m thick and is assumed to have negligible electrical conductivity. The data is calculated for both the vertical-coaxial coil configuration (HX system) and the horizontal-coplanar one (HZ system). The keel is assumed to have the shape of a Gaussian curve and its parameters and geometry are shown in Figure 2.

Figure 3 shows the HX system response for two different keels with parameters A = 12m, and W = 28m (solid line) or 14m (dashed line). The observed secondary magnetic field is expressed in parts per million (ppm) of the primary field at the receiver. There is a big anomaly in the system response due to either of these ice keels. For a keel width W = 28m, the maximum anomaly is about 35% of the background response and we define this to be the "percent anomaly". In order to characterize the shape of the observed anomaly, we define the "anomaly width", q, to be the width of the anomaly at one half of its maximum value. For a keel width W = 28m, the anomaly width q is 45m. When the keel width W is halved and other conditions are kept unchanged, the HX system anomaly is significantly reduced. In this case, the maximum anomaly (or percent anomaly) decreases by 37%, and the anomaly width decreases by 33%.

The situation for the HZ system is similar. Figure 4 shows the HZ system response due to the same two ice keels that were previously examined. For a keel defined by A = 12m and W =28m, the percent anomaly for the HZ system response is 32%. The anomaly width q is 66m in this case, which is much larger than that seen for the HX system. When the keel width is halved, the HZ system anomaly also decreases greatly. In this case, the anomaly amplitude (or percent anomaly) drops by 32% and the anomaly width decreases by 20%. From the comparison of Figures 3 and 4, we can see that the HX system response tracks the shape of the the keel closely, while the HZ system anomaly is much wider.

Next, the keel drawdown A is halved while the keel width is kept constant at W = 28m. The corresponding HX system response is shown in Figure 5 (dashed line), and HZ system response is shown in Figure 6 (dashed line). The anomaly amplitude contracts by 29% for the HX system and by 37% for the HZ system. However, the corresponding decrease in the anomaly width is small, 8% for the HX system, and 3% for the HZ system.

From the above model studies, we find that the amplitude of the anomaly (or accordingly the percent anomaly) is sensitive to both the thickness and the width of the keel. Thus, it is essentially a function of the area of the cross section of the keel. In contrast, the anomaly width is primarily related to the keel width. It is not sensitive to the keel thickness as long as the shape of the keel does not change. It is also noticed that the HX system response is more sensitive to keel shape than the HZ system response. The curve of HX system response resembles the keel shape and the change in the anomaly width due to the change of the keel width is much larger than that for the HZ system (33% compared to 20%).

It is worthwhile to compare the result calculated with the above technique to that obtained by the finite element method (Lee and Morrison, 1985) that allows a finite conductivity for both the sea ice and the sea water as well as a low operation frequency. This is done in Figure 7 for a triangular sea ice keel. The ice is uniformly 5m thick except for the keel area and it has a conductivity of 0.002 S/m. The conductivity of the sea water is 4 S/m and the HX system flies at 30m above the surface of the ice and operates at 2500 Hz. The dashed line in Figure 7 (upper part) represents the in-phase component of the response, and the solid line corresponds to the system response that would be observed if the sea water conductivity were infinite and sea ice conductivity were zero. Notice the similarity of these two curves and that the percent anomalies and anomaly widths are almost identical for these two curves. Thus it appears appropriate to use the perfect conductor model to interpret low frequency data observed in practice.

3.0 Charts for Data Interpretation

From the above analysis of numerical data, it is seen that the percent anomaly is a function of the cross area of a sea ice keel and the anomaly width is primarily a function of the keel width. To interpret field data, we need a strategy to relate the observed electromagnetic anomaly to the keel geometry. In terms of the smooth keel used in this analysis (see Figure 2), we need to estimate the two keel parameters A and W from the observed anomaly of the system response. In the following, we are going to concentrate on the HX system configuration which appears superior in terms of sensitivity of the keel geometry. Parallel analysis can be carried out for the HZ system. .

From the computation and analysis of the HX system response for model keels with systematic values of A and W, one can construct an interpretation chart as shown in Figure 8. The vertical axis of the chart is the percent anomaly, while the horizontal axis represents the ratio of the anomaly width q to the flight height h above the flat part of the ice-water interface. Here, the two sets of parametric curves intersect each other clearly and are well separated. The solid lines are for constant values of the drawdown A, and the dashed lines are for constant keel width W. The charted values of a and w are the keel drawdown "A" and width "W" normalized by h, the system height above the flat seawater surface. As shown, the percent anomaly decreases with the decrease of the keel drawdown and keel width. In fact, the line a = 0.8 tends to zero fast and intersects the lines a = 0.4, a = 0.2, etc.. For purpose of clarity, this is not shown in Figure 8. Nonetheless, this does not pose a serious problem in practice since such narrow and sharp keels are highly improbable. Although the interpretation chart is constructed for h = 30m, it can be used for the range h = 25m - 50m with an error less than ± 4%. .

As pointed out at the end of last section, the anomaly shape for low frequency data is almost identical to that obtained in case of the perfect conductor model. This makes the interpretation chart shown in Figure 8 applicable for a wide range of frequencies. We expect that the interpretation chart is useful for the frequency range between 1 - 100 kHz. A similar interpretation chart can be constructed for the HZ system response. It is shown in Figure 9.

Now consider the 2500 Hz theoretical data shown in Figure 7 (dashed line) for an assymmetric triangular keel. From the in-phase component of the AEM response, the percent anomaly and anomaly width are calculated to be 7% and 27m respectively. Thus q/h = 27/35 = 0.72. We draw this point in the interpretation chart (point C in Figure 8) and find the corresponding values of a and w to be 0.1 and 0.35 respectively. Hence A = a x h = 3.5m, and W = w x h = 12.3m. The estimated keel is drawn (symmetrically about the maximum anomaly of the data) in the lower portion of the illustration (dashed line in Figure 7). We can see that the size of the interpreted keel is close to that of the model keel. But since we have assumed that the keel is symmetric in the interpretation, the position of the maximum of the interpreted keel is misplaced 3.5m to the right.

The field data collected over an ice keel in the Prudhoe Bay by Geotech Ltd. for CRREL (Kovacs et al., 1986) are interpreted next. The AEM system used consists of two pairs of vertical coaxial coils (HX system) and two pairs of horizontal coplanar coils (HZ system). The former operate at 930 Hz and 4158 Hz respectively, and the latter operate at 530 Hz and 16290 Hz respectively. The transmitter-receiver separation of each coil pair is about 6.5m.

A part of the 930 Hz and 16290 Hz data for line F6L3 is shown in Figure 10. The altitude is the distance from the system boom to the ice surface measured by a laser altimeter. Note that the quadrature component of the 16290 Hz data is of bad quality. As we can see in the figure, the data are highly correlated with the altitude as expected. First, we interpreted the data using a 1-D technique (Becker et al, 1987). The result shows an average of 3m thick ice (see Figure 11) but no ice keel is apparent in the interpretation. However, we notice that there is an anomaly in the system response from fiducial numbers 2668 - 2675, which can not be related to the small altitude variation in that area. Moreover, the 1-D interpreted results also show thicker ice in that region. For the 930 Hz data, the anomaly width q is found to be 31.4m and the percent anomaly is 6.5%. Since h is about 38.5m (altitude + average ice thickness), $q/h = 0.83$. The corresponding point is found in the interpretation chart (point D in Figure 8), which gives $a = 0.08$ and $w = 0.42$. Hence $A = a \times h = 3.08$m, $W = w \times h = 16.2$m. The interpreted keel is plotted symmetrically about the maximum anomaly in Figure 11 (dashed line). The solid line in the figure is the average of the drill hole measurements along three parallel lines 11.5 meters apart. As we can see, the interpreted keel is an good approximation of the real feature. Note, however, that the effect of the variation of the altitude of the system boom has been neglected in the interpretation.

The interpretation of 4158 Hz data yields similar result to the above. But difficulty was encountered in interpreting the HZ system data which were acquired at 530 and 16290 Hz. As shown in Figure 10, there is also an anomaly in the 16290 Hz data from fiducial numbers 2668 - 2675. If we calculate the percent anomaly and anomaly width in this case, the corresponding point will fall off the interpretation chart shown in Figure 9 and can not be interpreted.

Generally speaking, the chart interpretation method gives an overall estimate of the size and shape of an ice keel. In order to recover the detail shape of an ice keel, we consider in the next section an iterative technique for data inversion. The chart interpretation can be used to obtain an initial model for the iterative technique where the variation of the system altitude can be easily accounted for.

4.0 Inversion of AEM Data

Now consider the determination of sea ice keel shape by data inversion. In order to overcome the problem associated with 1-D data interpretation, we present a two-dimensional (2-D) AEM data inversion scheme where ice thickness may vary along the flight direction of the AEM system. In the perpendicular direction, however, the geometry of the ice/water interface is assumed to be invariant. Although the actual ice bottom topography is three dimensional (3-D), most ice keels do exhibit a dominant extension in one direction which is called the strike because they are formed by the interaction of two ice plates (Canpolar, 1985). Indeed, as will be shown in the next section, if the strike length of an ice keel of Gaussian shape is greater than three times the AEM system flight height 2-D interpretation may be quite appropriate.

The interpretation scheme that we have developed is based on the principle of the Occam's inversion presented by Constable et al. (1987). Due to practical non-uniqueness of the inverse problem, there may be many sea ice keels that may fit the observed data within a specified error. This scheme yields the smoothest among these. The smooth keel shows the basic features of the actual keel although small bumps on its surface may not appear in the inversion results. Because the process of the data acquisition however constitutes a low-pass filter, it is also possible that the information needed to define the keel in detail is absent from the acquired data.

We first examine the theory of constrained smooth inversion and then use it to interpret synthetic field data which are generated with our numerical modeling program (cf. Appendix). Both noise free and artificially contaminated data are used to test the scheme. Finally we propose a procedure to apply this inversion scheme to low frequency data. In addition to this field data sets (collected in Prudhoe Bay by Geotech,Ltd. in 1985) are also interpreted using this procedure.

4.1 Theory

Let x denote the traverse direction of the AEM system, y the strike direction of the ice keel and z the downward pointing vertical. The interface of the ice and the water is described by $z = h(x)$. Our purpose here is to reconstruct this interface from the airborne electromagnetic data $d(x)$ while the upper surface of the ice is mapped by a laser device. The difference in elevation between the bottom and the top surface of the ice is the required ice thickness. The data $d(x)$ can be either the horizontal or/and the vertical secondary magnetic field recorded during the flight.

Now consider h(x) as the only unknown. This is true when the two assumptions used in our previous modeling algorithm (see the Appendix) are valid. These are: 1) the sea ice is transparent to the electromagnetic wave, and 2) the sea water can be treated as a perfect electric conductor. With these considerations, the AEM data can be written as

$$d(x) = F[h(x)] \qquad \text{... (1)}$$

where F[ ] is the functional for forward modeling. Equation (1) immediately reveals that the inversion problem is actually one-dimensional (1-D) with the direction of variation along x instead of along z. Fortunately there is a large amount of geophysical literature dealing with 1-D inversion. One particular scheme that well suits our needs is Occam's inversion presented by Constable et al. (1987).

The basis for Occam's inversion is the search for the smoothest solution that fits the observed data within a specified tolerance. This principle applies particularly well to our work because electromagnetic wave propagation in sea water is a diffusion process, so that the resolution of sharp edges on the ice/water interface can not be expected from the data. Furthermore, any information on the sharp edges is contained in the high wave-number range of the data, which is contaminated by noise. Because inversion implies downward continuation (Parker, 1977), the attempts to reconstruct the fine structures of the interface will amplify any noise and result in unstable solutions. Hence a stable solution is necessarily smooth.

In order to find the smoothest solution, let us first define the roughness of the ice/water interface. Physically the rougher the interface is, the larger the magnitude of its derivatives. Thus we define

$$R = \int_{p_1}^{p_2} \left|\frac{dh(x)}{dx}\right|^2 dx$$

to be the roughness of the interface $h(x)$; (p_1, p_2) define the lateral keel extent outside of which the interface is assumed to be flat. In the discrete sense,

$$R = \sum_{i=2}^{N} (h_i - h_{i-1})^2 = ||\partial \mathbf{h}||^2 \qquad \text{... (2)}$$

where $h_i = h(x_i)$ and $|| \; ||$ denotes the l_2 norm , and

$$\partial = \begin{vmatrix} 0 & 0 & 0 & \dots & 0 \\ -1 & 1 & 0 & \dots & 0 \\ 0 & -1 & 1 & \dots & 0 \\ \vdots & \vdots & \dots & \dots & \vdots \\ 0 & 0 & \dots & -1 & 1 \end{vmatrix}$$

$$\mathbf{h} = (h_1, h_2, \dots, h_N)^T$$

Note that $p_1 = x_1$, $p_2 = x_N$ and that the points x_1 , x_2 , ..., x_N are usually equally spaced.

Suppose that there are M data points recorded over an ice keel, $d_j = d(X_j)$, j=1, 2, ... M. The corresponding computed predictions from the discrete model $\mathbf{h}$ are $F_j(\mathbf{h})$. The goodness of fit of the predictions to the actual data can be evaluated using the least-squares criterion

$$E = \sum_{j=1}^{M} [d_j - F_j(\mathbf{h})]^2 = ||\mathbf{d} - \mathbf{F}(\mathbf{h})||^2 \qquad \text{... (3)}$$

where

$$\mathbf{d} = (d_1, d_2, \dots, d_M)^T$$

$$\mathbf{F}(\mathbf{h}) = (F_1(\mathbf{h}), F_2(\mathbf{h}), \dots, F_M(\mathbf{h}))^T$$

The ice/water interface $\mathbf{h}$ can only vary within some physical bounds. If we assume that the z = o plane is chosen such that it coincides with the flat part of that interface then a reasonable lower bound is:

$$h_i \geq 0, \quad i = 1, 2, \dots, N \qquad \text{... (4)}$$

since the ice keel protrudes downward. An upper bound can also be set for each individual case

$$h_i \le T_i, \quad i = 1, 2, \ldots, N \qquad \ldots (5)$$

In the Arctic, small ice keels may protrude several meters into the water, whereas large keels can protrude tens of meters (Lowry and Wadhams, 1979). The values of T_i should be estimated for each specific ice keel encountered. Note that $h_1 = h_N = 0$ should be included in the constraints in order for the solution to be smooth at the two end points. This condition can be met by simply letting $T_1 = T_N = 0$.

Now the mathematical problem to be solved can be stated as follows: find a solution **h** which minimizes the roughness R and brings the misfit E within an acceptable tolerance, while the bound constraints (4) and (5) are satisfied. Without the bound conditions (4) and (5), this problem is exactly identical to the one solved by Constable et al. (1987).

The condition for the data misfit is

$$\|\mathbf{d} - \mathbf{F}(\mathbf{h})\|^2 \le E_* \qquad \ldots (6)$$

where E_* is the tolerance. If we treat this inequality as an equality and apply the method of Lagrange multipliers, the above problem can be reduced to the minimization of

$$U = \|\partial \mathbf{h}\|^2 + \mu^{-1}\left(\|\mathbf{d} - \mathbf{F}(\mathbf{h})\|^2 - E_* \right) \qquad \ldots (7)$$

with constraints (4) and (5). Here μ^{-1} is the Lagrange multiplier. As interpreted by Constable et al, μ is a smoothing parameter. The larger the μ is, the less the solution is affected by the misfit. On the contrary, if μ is small, the data misfit is minimized with little influence from the roughness term.

The original problem can now be solved with the following procedures: Solve the above minimization problem for a series of μ

values to obtain a set of solutions of the ice/water interface. Among these solutions, choose the one which satisfies the tolerance condition (6). If more than one solution satisfies (6), choose the one with the largest μ value for this corresponds to the smoothest solution desired.

Such solutions can not however be easily obtained by the direct minimization of the objective function U (equation (7)) since the minimization is non-linear. It is first necessary to transform the non-linear problem into a problem of quadratic programming, for which existing mathematical tools can be used.

Let us linearize the $\mathbf{F}(\mathbf{h})$ about an initial model $\mathbf{h}^0$

$$\mathbf{F}(\mathbf{h}) = \mathbf{F}(\mathbf{h}^0) + \mathbf{J}\,\Delta \qquad \ldots (8)$$

Here $\mathbf{h} = \mathbf{h}^0 + \Delta$, and the Jacobian matrix is

$$\mathbf{J} = \begin{vmatrix} \frac{\partial F_1}{\partial h_1} & \frac{\partial F_1}{\partial h_2} & \cdots & \frac{\partial F_1}{\partial h_N} \\ \frac{\partial F_2}{\partial h_1} & \frac{\partial F_2}{\partial h_2} & \cdots & \frac{\partial F_2}{\partial h_N} \\ \cdots & \cdots & \cdots & \cdots \\ \frac{\partial F_M}{\partial h_1} & \frac{\partial F_M}{\partial h_2} & \cdots & \frac{\partial F_M}{\partial h_N} \end{vmatrix} \qquad \ldots (9)$$

Substituting (8) into (7) and dropping the constant term $\mu^{-1}E_*$, we obtain

$$U = ||\partial \mathbf{h}||^2 + \mu^{-1}\, ||\tilde{\mathbf{d}} - \mathbf{J}\,\mathbf{h}\,||^2$$

where $\tilde{\mathbf{d}} = \mathbf{d} - \mathbf{F}(\mathbf{h}^0) + \mathbf{J}\,\mathbf{h}^0$ is the modified data. Rearrangement of the above equation gives

$$V = \frac{1}{2}\mu U - \tilde{\mathbf{d}}^T\tilde{\mathbf{d}} = \frac{1}{2}\mathbf{h}^T\mathbf{H}\,\mathbf{h} - \mathbf{C}^T\mathbf{h} \qquad \ldots (10)$$

Here V is the new objective function and

$$H = \mu \partial^T \partial + J^T J$$

$$C = J^T \tilde{d}$$

Note that **H** is a symmetric positive-definite matrix. The minimization of the new objective function V is equivalent to the minimization of U for a fixed value of μ .

Now that the new objective function is in quadratic form the problem of optimization with bound constraints (4) and (5) can be solved using quadratic programming (Gill et al., 1981). There are subroutines available in existing mathematical software libraries which can be used for this purpose. These are E04AF in the NAG Fortran Library and VE04A in the Harwell Fortran Library. We selected VE04A because of its simplicity.

The smoothest solution can now be actually obtained in the following way. Starting from an initial model $\mathbf{h}^0$, solve the minimization problem for different μ values. From these solutions choose the one that minimizes the actual misfit E instead of V. (Minimizing V may result in divergence of the solution (Constable et al., 1987)). Use this solution for the next initial model and iterate until the solution for the ice/water interface that brings the misfit below a specified tolerance is found.

The initial model $\mathbf{h}^0$ can be chosen arbitrarily since it does not affect the convergence of the inversion to the smoothest solution. This is one of the beauties of the smooth inversion scheme which sets out to seek a unique solution. In our problem we set the initial model to be a flat ice cake, i.e., $\mathbf{h}^0 = (0, 0, ..., 0)^T$.

4.2 Inversion of synthetic data

In this section we will apply the above scheme to invert some synthetic data which are generated with the fast modeling algorithm (see Appendix). To test the stability of the inversion scheme white noise will also be added to the numerical data. Most inversion schemes require the knowledge of solutions to the forward problem in the computation of the Jacobian matrix. Here these are obtained with the fast forward modeling algorithm. The partial derivatives in the Jacobian are computed using the forward finite-difference of two numerical solutions. Thus one iteration of the inversion process requires N+1 forward calculations for M data points.

Before considering the inversion results let us first describe the geometry of the problem for all the synthetic models. Unless otherwise specified, the transmitter and the receiver are both small circular loops separated by 6.5 meters, and both are maintained at 25 meters above the upper ice surface. For convenience the vertical co-axial coil system is referred to as the HX system because the axes of the coils are in the x direction while the horizontal co-planar coil system is referred to as the HZ system. The receiver measures the secondary magnetic field which is expressed in parts per million of the received primary field. The inversion is performed for the HX and the HZ system data independently although the joint inversion can be easily accomplished. Except in the keel area the ice thickness is assumed to be five meters. Note that for the synthetic data generation we assume the induction limit to hold so that the system frequency is not involved. Applications to low frequency data, however, will be shown in the next section.

The position of the end points p_1 and p_2 of the ice keel must now be chosen. These can be arbitrary as long as they are located outside the range of the keel. They should not be too far apart however, because the computation of the Jacobian matrix can be quite expensive and convergence of the inversion may be slow. From our experience, locating these two points at the one quarter of the peak HX system anomaly points, and at the two points at half of the peak HZ system anomaly yields good results . Note that this condition may need to be relaxed for field data because they may be acquired at fluctuating flight heights, which will distort the shape of the observed anomaly.

The sampling interval of $h(x)$, Δx, is taken to be 3 meters for all the examples shown in this report. The data $d(x)$ is sampled at an interval of 3.5 meters unless otherwise specified, which at a helicopter speed of 68 knots corresponds to a recording rate of 10 samples/second.

Figure 12(a) shows the inversion results of the HX system data for a triangular model keel (Model 1), which is symmetric and is located between 60 and 90 meters along the profile. The keel drawdown at the peak is 5 meters and its two sides slope at 18 degrees from the horizontal. The solid line in the upper graph is the synthetic data corresponding to the ice keel shown in solid line in the lower graph, while the dashed line is the system response from the inverted keel shown in dashed line in the lower graph. Similar results for the HZ system data are shown in Figure 12(b). In the inversion the two end points p_1 and p_2 are taken at 45 and 105 meters respectively. The tolerance criterion for convergence is set at 1 ppm for the HX system data and 2 ppm for the HZ system data. Three iterations in the inversion yield the convergent solutions shown in Figure 12 from an initial guess of a flat ice cake that is 5 meters thick. As we can see the interpreted keels have smooth vertices as can be expected for the constrained smooth inversion. The interpreted keels are about 1.5 meters too shallow, and the independent results for the HX and the HZ system data are almost identical.

In the next example we invert the synthetic data for an irregular trapezoidal ice keel (Model 2). The steep side of the keel is 34 degrees from the horizontal while the slope of the other side is 16 degrees (Figure 13). The keel protrudes 6 meters down into the water with a flat bottom of 12 meters. Again both the HX and the HZ system data are inverted independently and the results shown were obtained after three iterations. Misfit for the HX system data is again held below 1ppm, and that for the HZ system data is below 2ppm. For the HX system the inverted maximum keel thickness is correct although the two vertices on the steep side are smoothed out. The other side however is imaged correctly. For the HZ system, the results are similar except that the maximum keel thickness is a half meter too small.

Now white noise of 5% to 10% of the peak anomaly is added to the Model 1 and Model 2 theoretical data. Here system noise may be small but "geological noise" is estimated to be about 5 percent of the anomaly amplitude. The tolerance criterion for the inversion is set at the RMS noise level and the convergence is usually achieved within 4 -5 iterations. The inversion results are shown in Figures 14 - 16. As demonstrated in these figures the inversion still yields good results except at the 10% noise level where the keel interpreted from the HZ system data is flat at the bottom (Figure 15). Our experience with noisy data shows that the inversion results obtained from the HX system data are usually better than those obtained from the HZ system data. In practice 5% noise is probably excessive.

We next examine the case where the height of the system varies during a flight, as is usually the case in practice. The model keel is a symmetric trapezoid in shape with a maximum drawdown of 3 meters (Model 3). The flat bottom is 12 meters wide and the two sides of the keel slope at 12 degrees from the horizontal (Figure 17). The system height over the flat part of the ice/water interface, shown in Figure 17(a), varies between 37 and 41 meters. This height variation was taken from a test flight in Prudhoe Bay. Here it is assumed that the transmitter and the receiver are always at the same level although in practice the instrument pod may tilt in space. The inversion results for the HX and the HZ system response are shown in Figures 17(b) and (c). As can be seen the inverted keels agree very well with the model keel in this case. The final data misfit is 0.3ppm for the HX system and 0.8ppm for the HZ system.

4.3 Application to low frequency data

The inversion procedure can also be extended to the interpretation of low frequency field data. As mentioned earlier, the synthetic data used above were generated in the induction limit where the secondary field only exhibits an in-phase component. In practice however, this can not be realized because the sea water is not a superconductor and the operating frequency of the system must be low enough for the EM wave to penetrate the ice freely. Hence the measured secondary magnetic field has both an in-phase and a quadrature component. Ideally the in-phase and quadrature components can be inverted simultaneously using the above scheme if a fast modeling algorithm for a general conductivity distribution exists. The finite-element and finite-difference methods have been very successful in solving such problems (Lee and Morrison, 1985; Stoyer and Greenfield, 1983). The computational costs associated with these methods however, are prohibitive so that they can not be used in solving this problem as too many forward computations must to be carried out even for one iteration.

Although the fast modeling algorithm that we developed generates theoretical data at the induction limit it can be used to invert low-frequency data. For data collected at 30 kHz the arithmetic sum of the in-phase and quadrature components gives a good approximation of the data that would be obtained at the induction limit (Becker et al, 1983). This sum can be directly interpreted using the previous scheme, but it must first be multiplied by a scale factor prior to interpretation. The same scheme can be used since the anomaly shape changes but little with frequency (Becker and Liu, 1987). The scale factor can be computed as the ratio of the response at the induction limit to the sum of the in-phase and quadrature components at that frequency for a 1-D model. Note that this factor varies with the system altitude. If the altitude does not change much in a flight over the ice keel this factor can be taken as a constant.

Let us now consider an example of synthetic data of the HX system at the frequency of 2500 Hz which was generated by the finite-element method (Lee and Morrison, 1985). The conductivities of the ice and water are 0.002 S/m and 4 S/m respectively. The ice is uniformly 5 meters thick except at the keel area where it protrudes 5 meters into the sea water. The shape of the keel is step-triangular as shown in Figure 18 and the HX system is flown at 30 meters above the ice surface. The scaled sum of the in-phase and quadrature components of the theoretical data and the inversion results are also shown in Figure 18. There are ten unevenly spaced data points and the scale factor is 1.044 in this case. As we can see the inverted ice keel is close to the model although it is somewhat shallower. We consider this an encouraging success of the application of the above proposed procedure.

We have not yet had the opportunity to experiment with large quantities of low frequency data. Thus it is not clear in which frequency range the above procedure can be used to yield reasonable results. Furthermore the role of the water conductivity has not yet been investigated although we suspect that its main effect is to decrease the resolution of the inversion results expected from the superconductor assumption. It is worthwhile however to attempt an inversion of the data collected in the Prudhoe Bay by Geotech Ltd. in 1985 (Kovacs, et al., 1987). The data is for line F6L3 collected with the HZ system at the frequency of 16290 Hz. Because the quadrature component is of very poor quality only the in-phase component was scaled by a factor of 1.104 and interpreted. The system altitude, the scaled in-phase component and the inversion results are all shown in Figure 19. The inversion was performed with the two end points of the ice keel fixed at p_1 = 54 meters and p_2 =102 meters. The background ice thickness was fixed at 3.3 meters which is the average from the 1-D interpretation.

The interpreted ice thickness is 1.5 meters too shallow as compared to the drill-hole measurements (solid line in the bottom of Figure 19) in the keel area. However the keel is clearly visible in the inversion result, which is encouraging since it is much better than the 1-D inversion that shows but little of the keel (Becker et al., 1987). The inversion was stopped at an RMS data misfit of 15 ppm since more iterations could not reduce the misfit. There may be several causes for such a large misfit:

1) The effects of the pitch and roll of the system during the flight was not accounted for in this inversion.

2) The time constant of the instrument may be too large for such a small keel to be fully represented in the data. The inversion treats the data as being recorded with a zero time constant, which ignores the integral effect due to the instrument. This effect has been carefully studied by Becker and Cheng (1987).

3) The top surface of the ice was treated as a flat plane and the effect of the associated ridge was not considered.

4) *The assumed 2-D structure of the ice may be incorrect.*

5) For this frequency channel the operational system noise was large. The quadrature component was quite erratic and was not used in the inversion.

Errors due to 1) can be diminished by improving the computer code. Errors due to 2) and 5) can be reduced to a large extent by improving the instrument. But errors due to 4) can not be reduced in the present 2-D inversion scheme. To do this a 3-D inversion algorithm needs to be developed which can be an extension of the present scheme. The process will be computer intensive and furthermore, the data acquisition will necessarily cover an area instead of a line over the keel. This will be very costly and may not be worthwhile.

Errors due to 3) need to be further investigated. Treating the ice top as a flat plane has an effect of pressing the ridge down into the sea water. The ice thickness probably may not be largely affected. But it remains to to be seen whether a topographic correction is necessary before a 2-D inversion is attempted.

Applying the constrained smooth inversion to the scaled sum of the in-phase and quadrature components of the real data appears to be an attractive simple scheme. If the phase of the response does not ch ge, this is identical to scaling the in-phase component only. The performance, mainly the resolution, of this scheme will deteriorate with lowering the operation frequency. The range of the frequency, in which the above procedure can be applied, needs to be defined by further investigation. Effects of the conductivities of the sea ice and water will also reduce the resolution of the keel geometry expected from the assumption of transparent ice and superconducting sea water, which need be studied in the future.

All the computations in the above inversion have been done on a IBM RT PC computer, which has a comparable speed as the Microvax. An typical case of inversion of M = N = 20 takes about 4 hours of the CPU time. At the present the computer code is not optimized and it can be shown that its optimization may sharply reduce the CPU time by a factor of 5. It appears that in flight data interpretation will require some highly specialized computing equipment.

In all our examples the ice keel position was assumed known. In practice the position of the ice keel may be obtained from the videocamera record of the associated ridge. If this is not possible the AEM data itself can be used to estimate the position of the ice keel prior to the inversion. To do this the influence of the altitude should first be removed from the data so that the remaining anomaly will indicate the position of the keel.

4.4 Discussion

The 2-D inversion scheme can successfully recover the ice keel information, which is usually lost in 1-D inversion. Because the electromagnetic induction is a diffusion process, sharp edges can not be resolved from the data and we set out to seek a smooth ice keel, which retains the major features of the true ice keel. Although the scheme is designed to work for data collected in the induction limit with transparent ice and superconducting sea water, it can be applied to low frequency data, where the sum of the in-phase and quadrature components is scaled to approximate the required data. At the present the range of the frequency, in which this procedure can be applied, remains unclear and needs further investigation. With refinements and optimization of the computer program however, there is no doubt that this algorithm can be used routinely to process field data.

5.0 The Three-Dimensional Keel

Under usual circumstances the ice-water interface in arctic seas exhibits the same irregular topography that is normally associated with land forms. In areas covered by young ice however, the topographic features are minimal and a one-dimensional data interpretation technique for the average thickness of the ice sheet from airborne electromagnetic data (Kovacs, et. al. , 1987) can be used. This technique fails in areas of even moderate ice-water interface topography (Kovacs, et al, 1987). In cases where the axial length of the keel is very large (compared to the AEM system altitude) it is possible to remedy this situation with the use of interpretation charts that allow for a two-dimensional description of the keel. If necessary, a 2-D data inversion outlined above may be carried out to delineate ice keels in more details. Because the two-dimensional data interpretation method appears to produce adequately accurate results it appears worthwhile to find the minimal keel axis length that is necessary for its proper application. This is done below where theoretical data for three-dimensional (3-D) keels is presented. In all 14 different keel models are considered.

5.1 The keel model

Cartesian coordinates are used and the XY plane is chosen to coincide with the flat part of the ice-water interface so that the Z axis points vertically down into the sea water. The ice is assumed to be transparent to electromagnetic waves and sea water is assumed to be a perfect electric conductor. Both the co-axial (HX) and co-planar (HZ) coil systems are considered.

The 3-D Gaussian keel model is an extension of the 2-D model described above. Its shape is given by

$$t(x,y) = A\, e^{-\frac{y^2}{0.361\, s^2} - \frac{x^2}{0.361\, w^2}} \tag{1}$$

where

$t(x,y)$ is the protruding depth of the ice keel into the sea water,
A is the keel drawdown or maximum keel thickness,
w is the keel width at $t = A/2$ on the cross section $y = 0$,
s is the keel width at $t = A/2$ on the cross section $x = 0$.

Here we define the y direction as the strike direction of the ice keel and hence designate s as the strike length of the keel. An infinite value of s corresponds to a 2-D ice keel. When $s = w$, the keel is a body of revolution. The numerical calculations were carried out as outlined in the Appendix.

5.2 Airborne electromagnetic profiles

In this section, we examine AEM profiles over three sets of model ice keels. Except in the keel area, the sea ice is assumed to have a uniform thickness of 5 meters. The keel parameters A, w, and s are varied to demonstrate the corresponding changes in the AEM system response. For each model set, the keel drawdown A and keel width w are fixed, while the keel strike length is varied. Model Set 1 represents a shallow keel, where $A = 3$m, $w = 24$m, and keel strike lengths of 12, 24, 48, 96, and inf. meters respectively (Figure 20). The corresponding theoretical AEM response for the HX and HZ system over the center of these features along a line perpendicular to the keel strike is shown in Figure 21, where curves 1, 2, 3, 4, and 5 correspond to s = 12, 24, 48, 96, and inf. meters respectively. Both the transmitter and the receiver, which are separated by 6.5m, are "flown" 25m above the upper ice surface. The system response is plotted at a point located mid-way between the transmitter and receiver, and is expressed in parts per million of the primary field at the receiver. Figure 21 (c) shows the the cross section of the ice directly below the flight line.

It is evident in Figure 21(a) that as the strike length s decreases the HX system anomaly becomes smaller but the general shape of the anomaly curve changes little. If these system anomalies for 3-D ice keels are interpreted using the 2-D chart given in our previous report, the interpreted cross section of the keel will be smaller than that in the 3-D model. The significant effect here is a reduction of the interpreted keel drawdown. The keel width however is less affected since the anomaly width does not contract significantly with the keel strike.

As shown in Figure 21(a), the HX system response for a keel strike length s of 96m (curve 4) closely resembles the response for an infinitely long keel (curve 5). In fact the maximum difference in the relative amplitude or percent anomaly (c.f. Becker and Liu) is of the order of 1.5% and would result in an under-estimation of the keel depth of about 10%. In fact the relative error in the percent anomaly size is also of this order and suggests that a 10% value for this quantity be used as a threshhold value for the definition of a two-dimensional keel. Thus, If the maximum relative difference in the relative anomaly is less than 10%, the finite strike keel can be effectively considered as two-dimensional, and hence 2-D interpretation techniques can be applied.

Using this criterion, any ice keel whose strike length exceeds 96m can be effectively considered as two-dimensional for the purpose of data interpretation. In general, the ratio of the strike length to the system height over the ice-water interface must be larger than 96/30. This value is much greater than the value expected from the footprint for the HX system (see Becker, at al., 1987), which is 1.4. At the first sight, this may seem puzzling. A careful look at the current distribution on the water surface, however, will yield the explanation.

The current pattern for a horizontal-axis transmitter is shown in Figure 22 (from Becker, et al., 1987). It consists of two circular current rings while most of the current flows in a direction parallel to the strike of the ice keel. For a keel of finite-strike, this major current flow is distorted. The footprint however was computed on the basis of the unaltered current flow so that it is not surprising to find that the strike/height ratio needed for 2-D interpretation is much larger than the footprint of the HX system.

Now let us look at the HZ system response and see how it changes as the keel strike length is varied. Curves 1, 2, 3, 4, and 5 in Figure 21(b) correspond to s = 12, 24, 48, 96, and inf. meters respectively. As expected, the HZ system response also decreases when the strike length of the keel is reduced. The major change is the decrease of the anomaly amplitude. The anomaly width, in contrast, stays almost constant. Nonetheless, as the keel strike is further decreased, the anomaly shape also changes and the lower part of the anomaly flattens out.

The HZ system response for s = 96m is also close to that for the 2-D keel. The maximum relative difference in the relative anomaly in this case is about 12% . Thus taking 10% as the threshhold, the keel strike should be somewhat longer than 96m in order to use the 2-D interpretation. Roughly speaking, we may again take the strike/height ratio to be approximately 96/30 = 3.2 for the 2-D interpretation to be valid. This is the same as that for the HX system and is very close to the value obtained from the footprint of HZ system which is 3.7 . As a matter of fact, the current pattern on a flat water surface for a vertical-axis transmitter consists of concentric circles so that a 3-D ice keel will distort this pattern as much as a 2-D ice keel does.

Now let us increase the keel drawdown A to 6m and keep the keel width at w = 24m. Figure 23 shows the deeper ice keels with different strike lengths (Model Set 2). The corresponding HX and HZ system responses are shown in Figure 24 (a) and (b) respectively. Now both the HX and HZ system responses are much larger than those for Model Set 1 due to the larger cross section of these keels. But the characteristics of these anomalies are similar to those discussed for Model Set 1. Therefore, the analysis for Model Set 1 still holds in this case.

Figure 25 shows the narrower keels that make up Model Set 3, where A = 3m and w = 12m. The keel strike, again, is 12, 24, 48, 96, and inf. meters respectively. Figure 26 (a) and (b) displays the HX and HZ system response and curves 1, 2, 3, 4, and 5 correspond to s = 12, 24, 48, 96, and inf. meters. Figure 26 (c) shows the cross section of the keel below the flight line.

Now the HX system response is much smaller than those shown in Figure 21 (a) because of the smaller size of the ice keels. But the relative characteristics remain unaltered and the analysis for Model Set 1 still applies here.

The HZ system anomaly also reduces with the decrease of the strike length of the keel. Furthermore, it is observed that a minor feature begins to appear at the center of the anomaly when the keel strike is small. This feature, however, may not be real as it may be caused by numerical errors in the calculation. At present, we are unable to make this point clear due to computational limitations.

5.3 Discussion

1) The strike length of an ice keel must be at least three times the flight height of the AEM system in order to successfully use the 2-D techniques to interpret both HX and HZ systems response.

2) For the HX system, the anomaly amplitude decreases as the keel strike shortens, but the anomaly width changes little.

3) For the HZ system, the anomaly amplitude also decreases as the keel strike contracts. At the same time, the lower part of the anomaly tends to become flat.

References

Becker, A., Morrison, H.F., and Smits, K., 1983, Analysis of airborne electromagnetic systems for mapping thickness of sea ice: NORDA technical note 261.

Becker and Cheng, 1987, Detection of repetative electromagnetic signals, *Electromagnetic Methods in Apllied Geophysics*, Vol. 1, Edited by M.N. Nabighian, pp.443-465.

Becker, A., and Liu, G., 1987, Airborne electromagnetic mapping of sea ice keels, First Interim Report to NORDA.

Becker, A., Liu, G., and Morrison, H.F., 1987, Airborne electromagnetic sensing of sea ice thickness: Report prepared for CRREL under UCB contract #DACA89-85-K-0008.

Bergeron, C.J., Juliette, W.I., and Gus, A.M., 1987, Lateral resolution of the modified image method for sea-ice thickness, Extended Abstracts of the SEG 57th International Meeting, New Orleans.

CANPOLAR Consultants ltd., 1985, Review of floating ice thickness measurement: Report prepared for the Department of Fisheries and Oceans, Canada under contract 01SE.FP921-3-0624.

Gill, P.E., Murray, W., and Wright, M.H., 1981, *Practical Optimization*, Academic Press, Inc..

Graham,I.G., 1980, Some application areas for Fredholm integral equations of the second kind: The Application and Numerical Solution of Integral Equations, Anderssen et al. ed., 75-101.

Kovacs,A., and R.Morey, 1985, Electromagnetic measurements of multi-year sea ice using impulse radar: CRREL Report 85-13.

Kovacs., A., Valleau, N., and Holladay, J.S., 1987, Airborne electromagnetic sounding of sea ice thickness and sub-ice bathymetry, CRREL Report 87-23, pp.1-40.

Kovacs, A., Morey, R.M., Gordon, F.N.C., and Valleau, N.C., 1986, Modeling the electromagnetic property trends in sea ice and example impulse radar and frequency-domain electromagnetic ice thickness sounding results: CRREL report in preparation.

Lee, K.H., and Morrison, H.F., 1985, A numerical solution for the electromagnetic scattering by a two-dimensional inhomogeneity: Geophysics, 50, 466-472.

Liu, G., and Becker, A., 1988, Effect of three-dimensional sea ice keels on airborne electromagnetic system response, Second Quarterly Report to NORDA.

Liu, G., and Becker, A., 1987, Airborne electromagnetic sensing of sea ice thickness, presented at the 57th SEG Annual International Meeting.

Lowry, R.T., and Wadhams, P., 1979, On the statistical distribution of pressure ridges in sea ice, Journal of Geophysical Research, Vol.84, No. C5, pp.2487-2494.

Mikhlin,S.G., 1964, Integral equations: Pergamon Press, 7-15.

Parker, R.L., 1977, Understanding inverse theory, Ann. Rew. Earth Sci., vol.5, pp35-64.

Parry, J.R., and Ward, S.H., 1971, Electromagnetic scattering from cylinders of arbitrary cross-section in a conductive half-space Geophysics, 36, 67-100.

Rikitake,T.K., and K.Whitham, 1964, Interpretation of the Alert anomaly in geomagnetic variations: Canadian Journal of Earth Sciences, vol.I, 35-62.

Rossiter, J.R., and Lalumiere, L.A., 1988, Evaluation of sea ice thickness sensors, Canpolar Inc., Toronto, Project 1027.

Stoyer, C.H., and Greenfield, R.J., 1976, Numerical solutions of the response of a two-dimensional earth to an oscillating magnetic dipole source, Geophysics, Vol.41, No.3, pp.519-530.

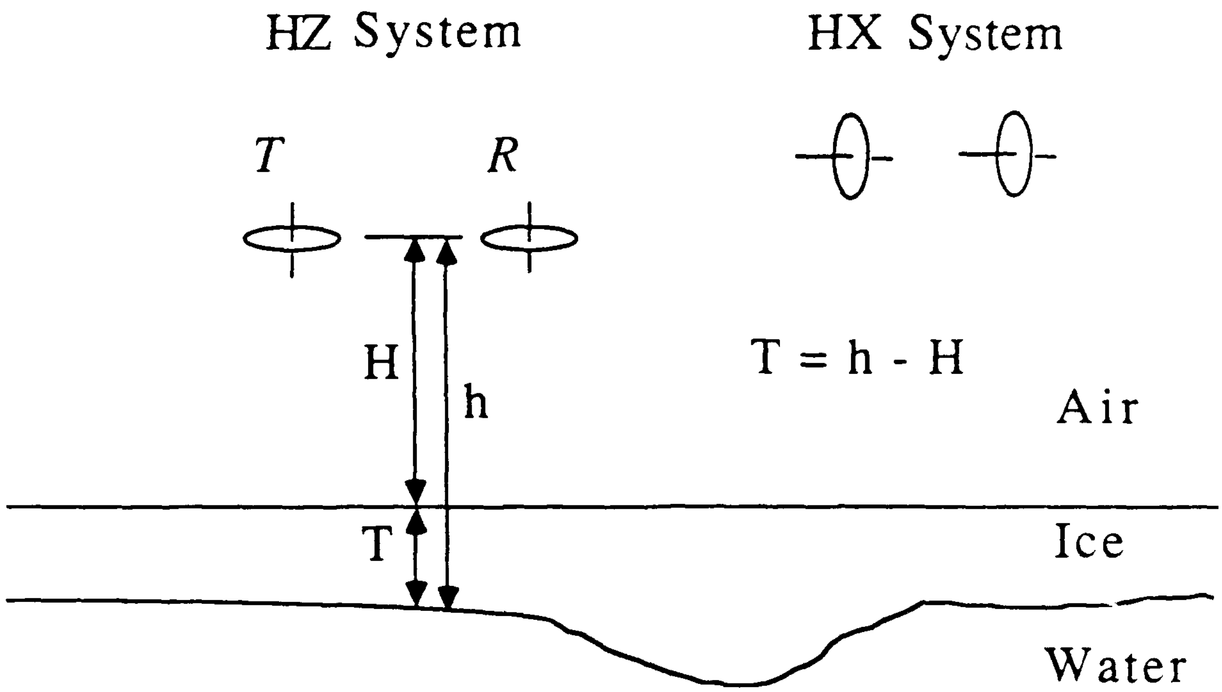


Fig.1 HX and HZ systems over sea ice

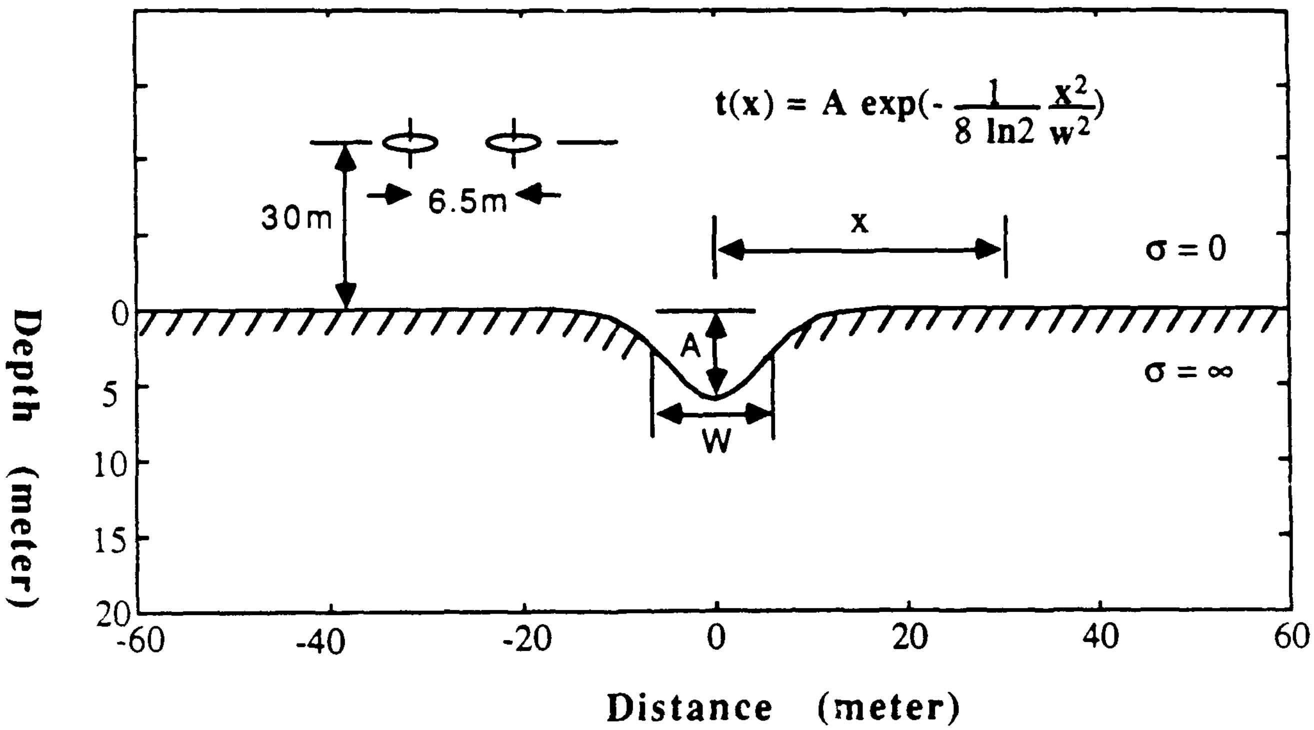


Fig.2 Cross section of a smooth keel

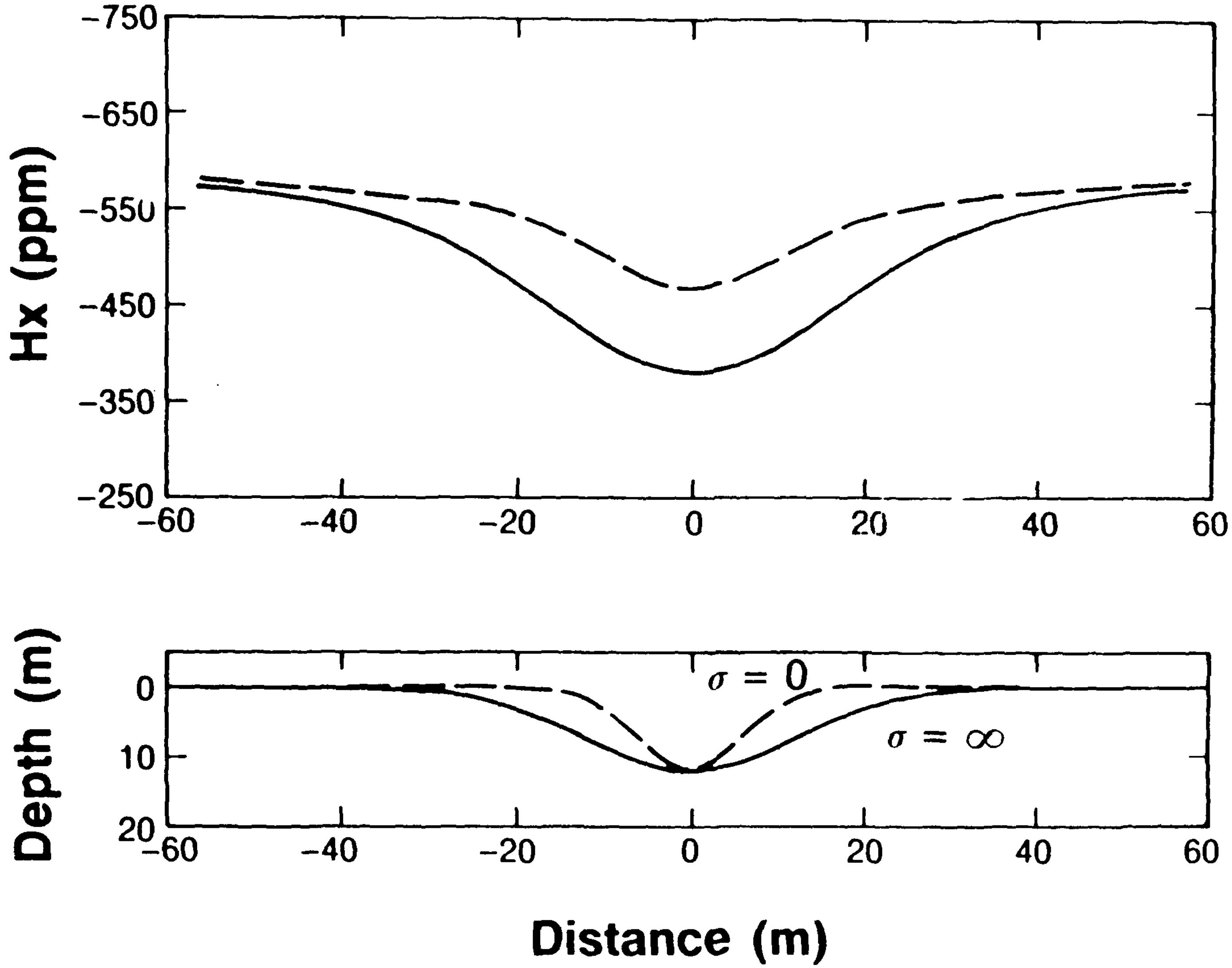


Fig.3 Effect of Keel Width on HX System Reponse

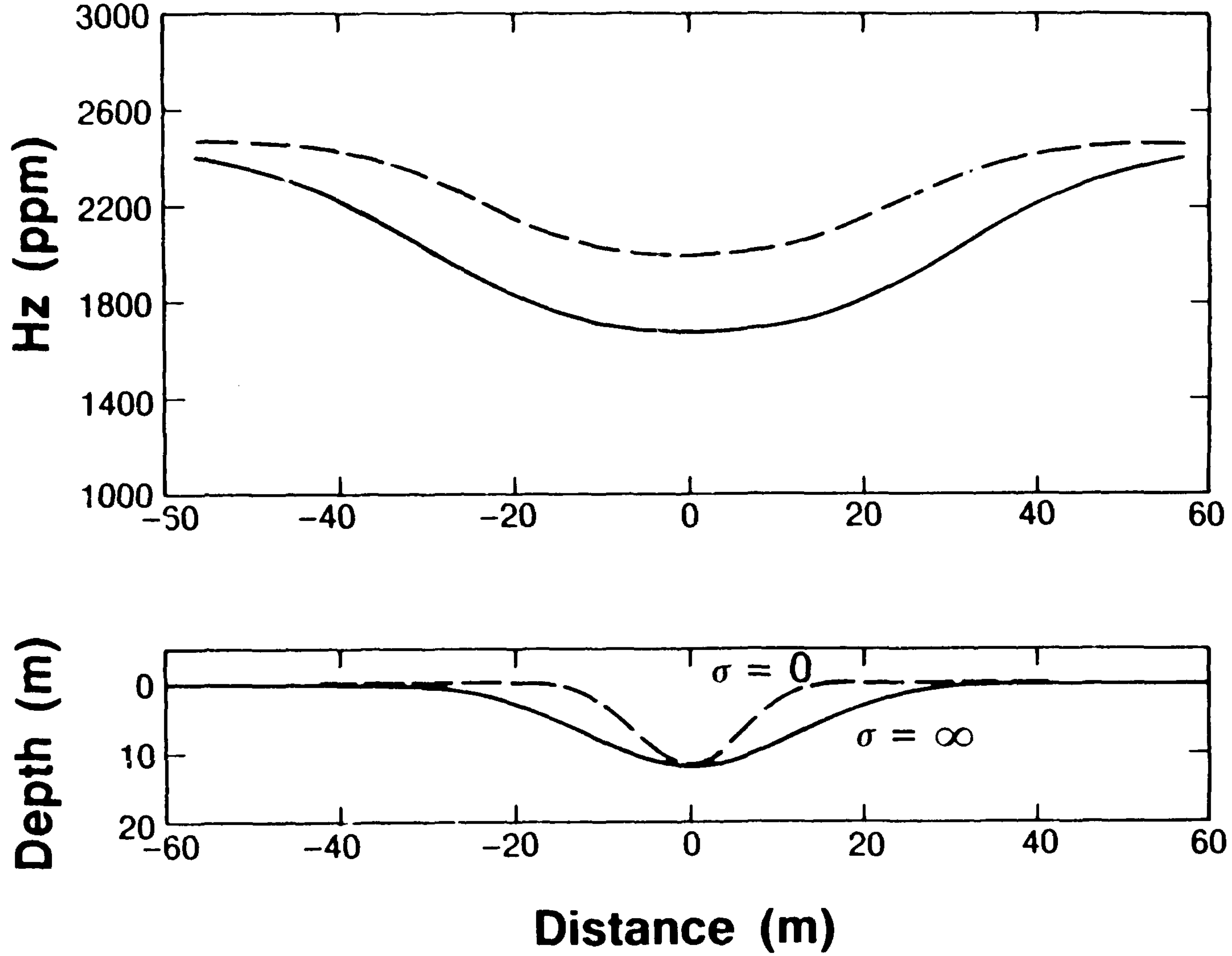


Fig.4 Effect of Keel Width on HZ System Response

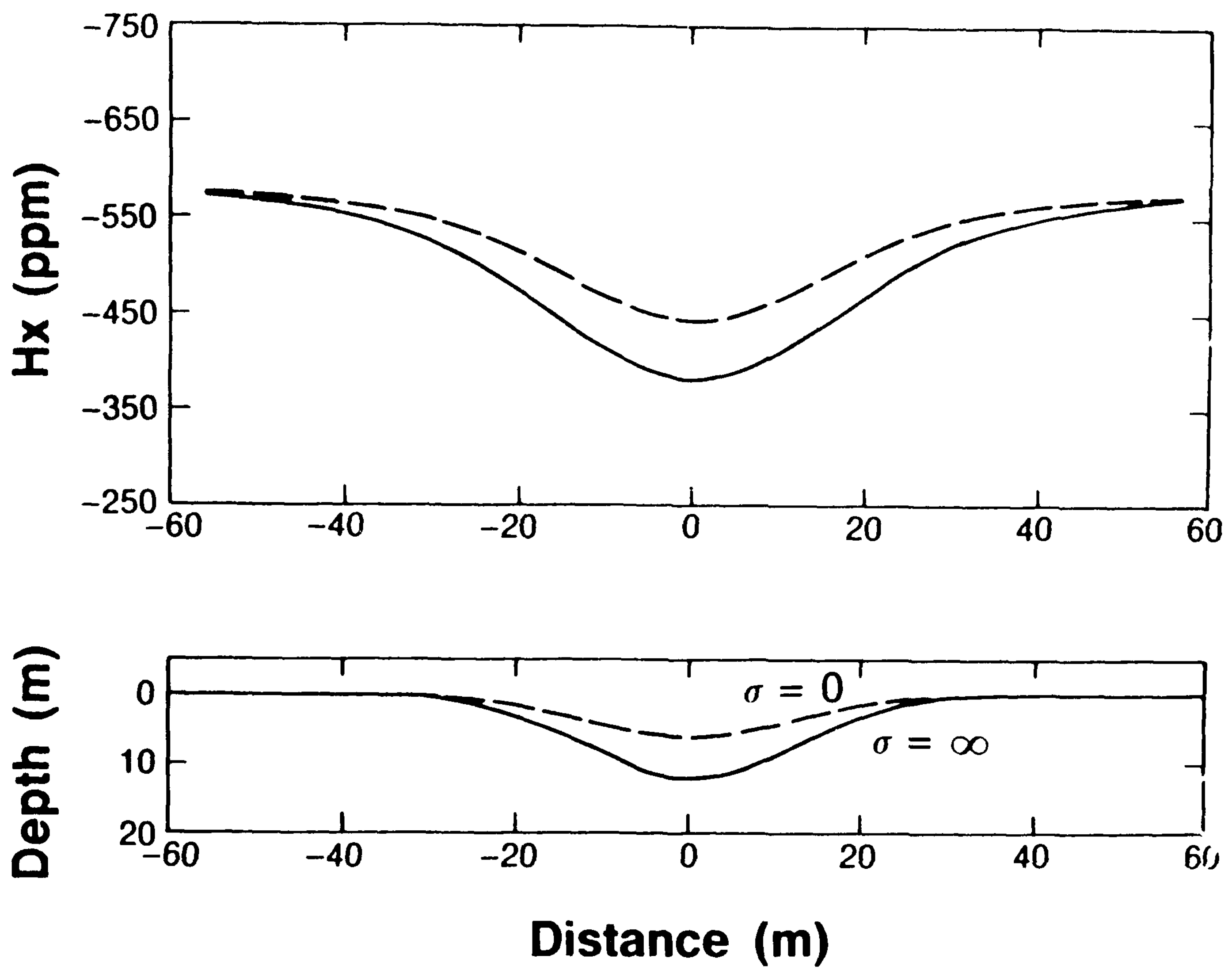


Fig.5 Effect of Keel Drawdown on HX System Response

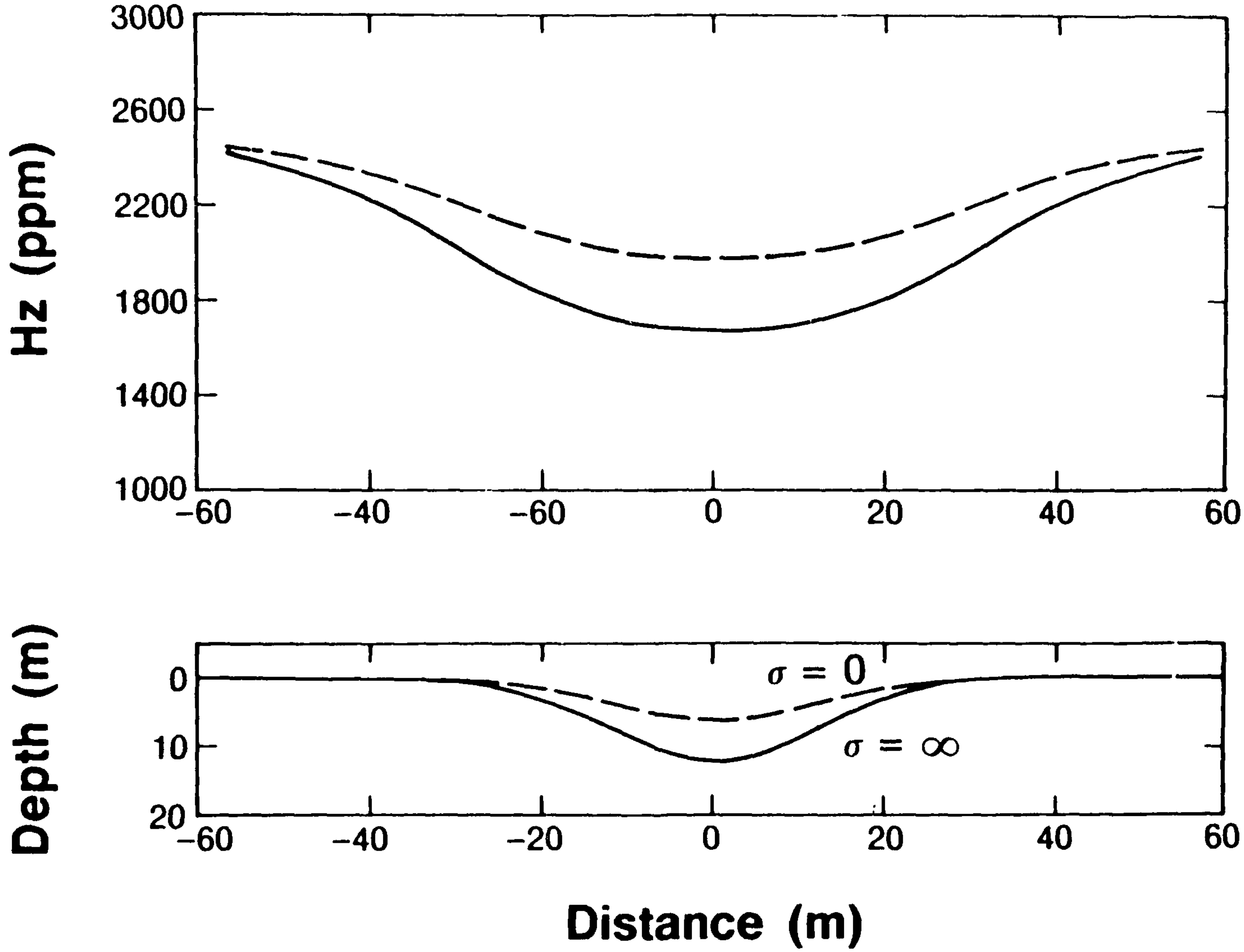


Fig.6 Effect of Keel Drawdown on HZ System Response

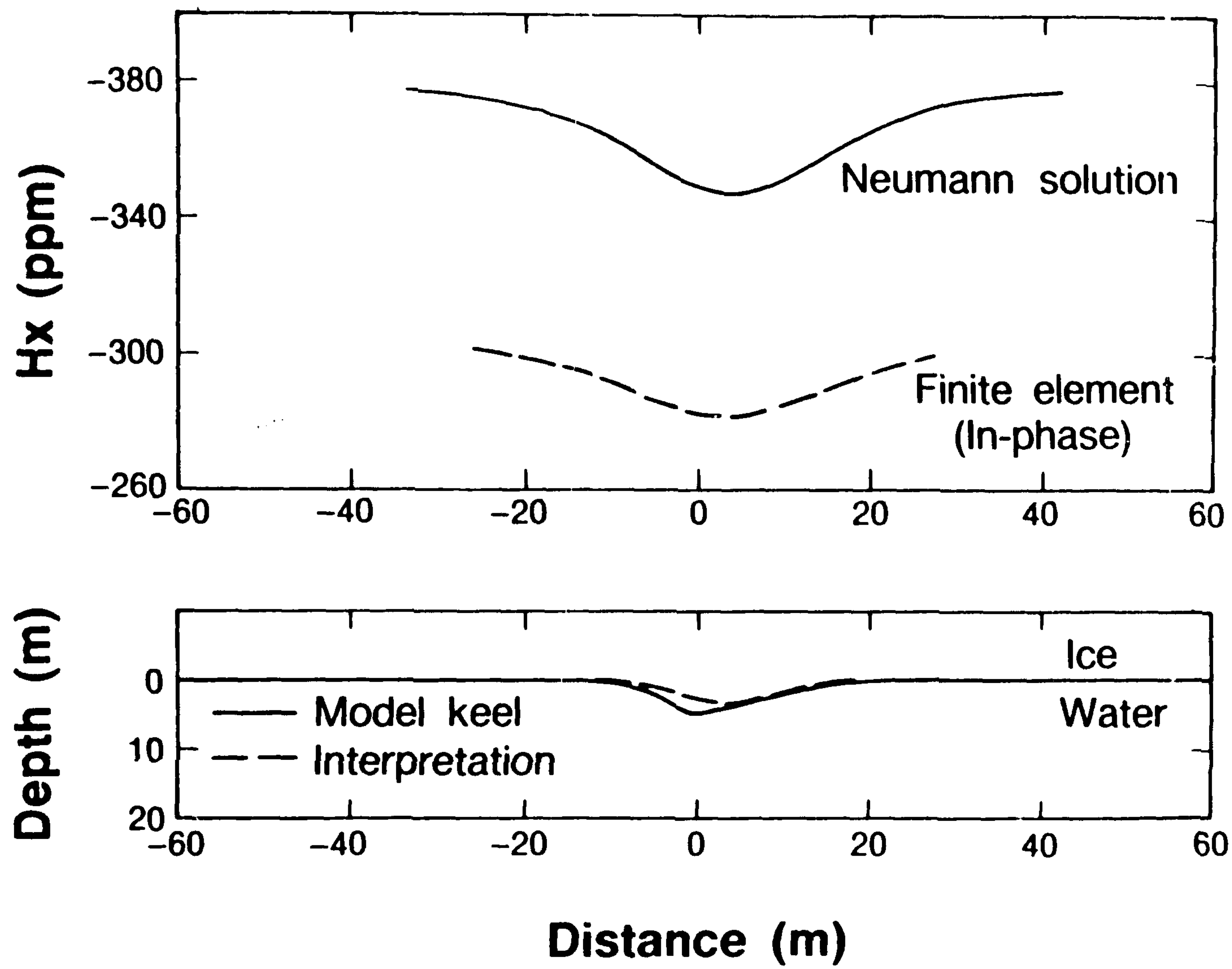


Fig.7 Model Response

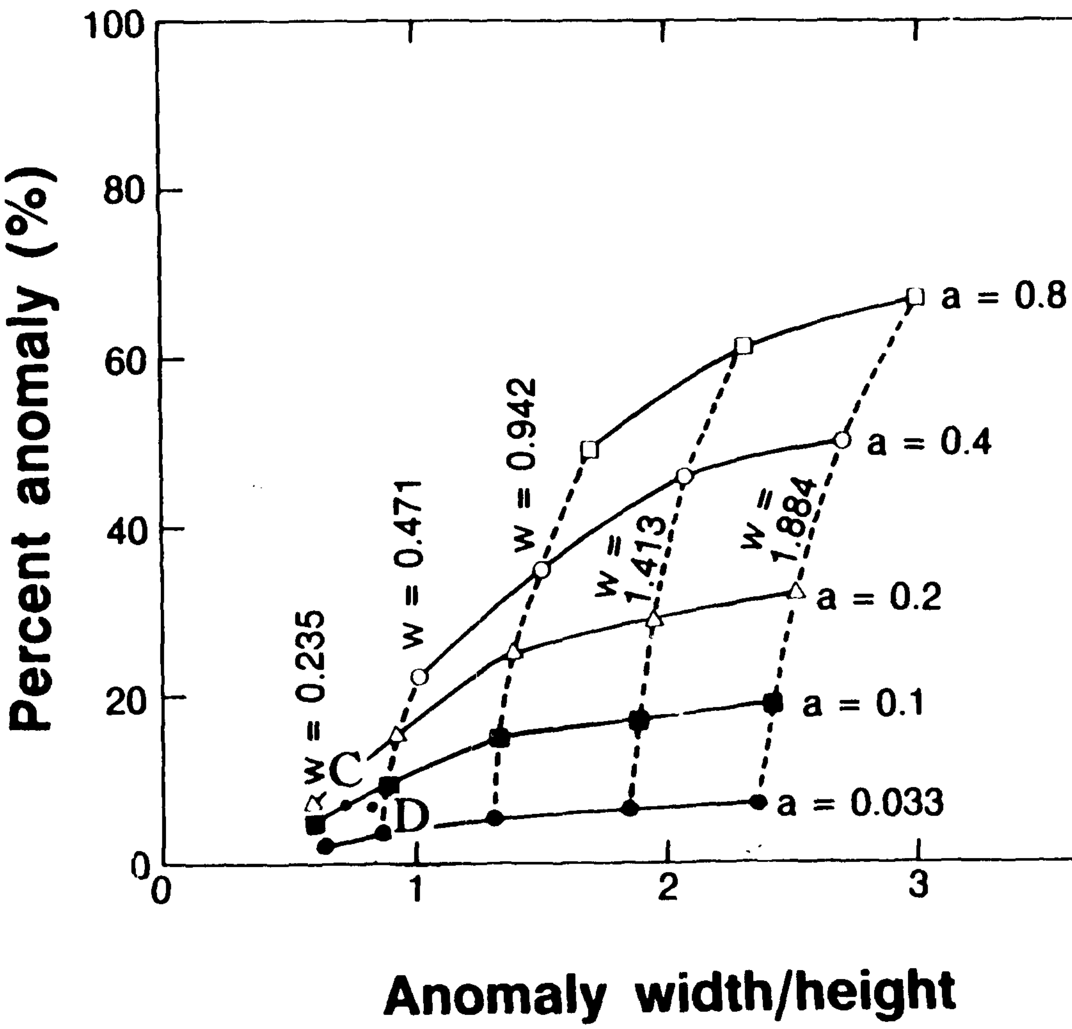


Fig.8 Interpretation Chart for HX System

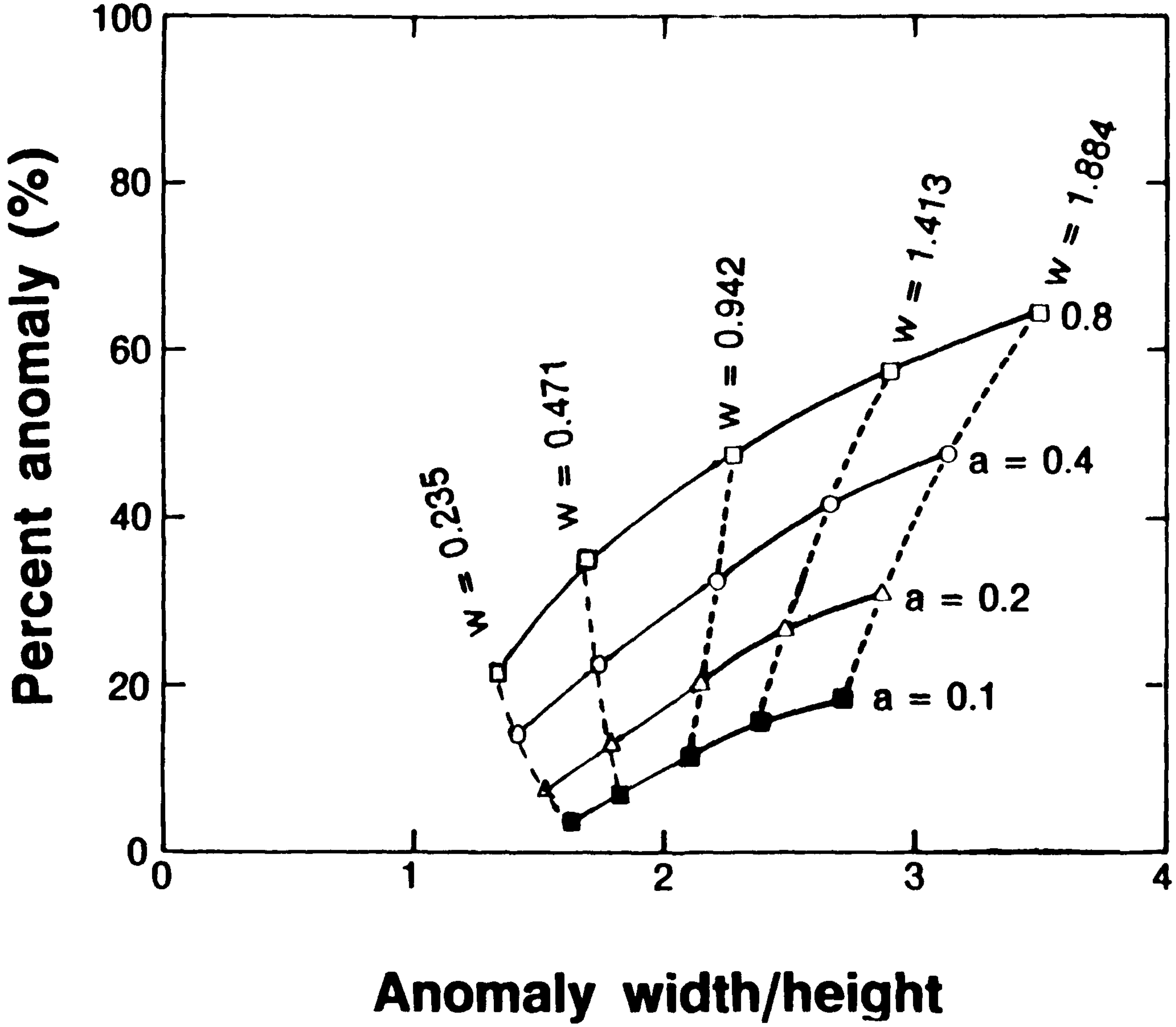


Fig.9 Interpretation Chart for HZ system

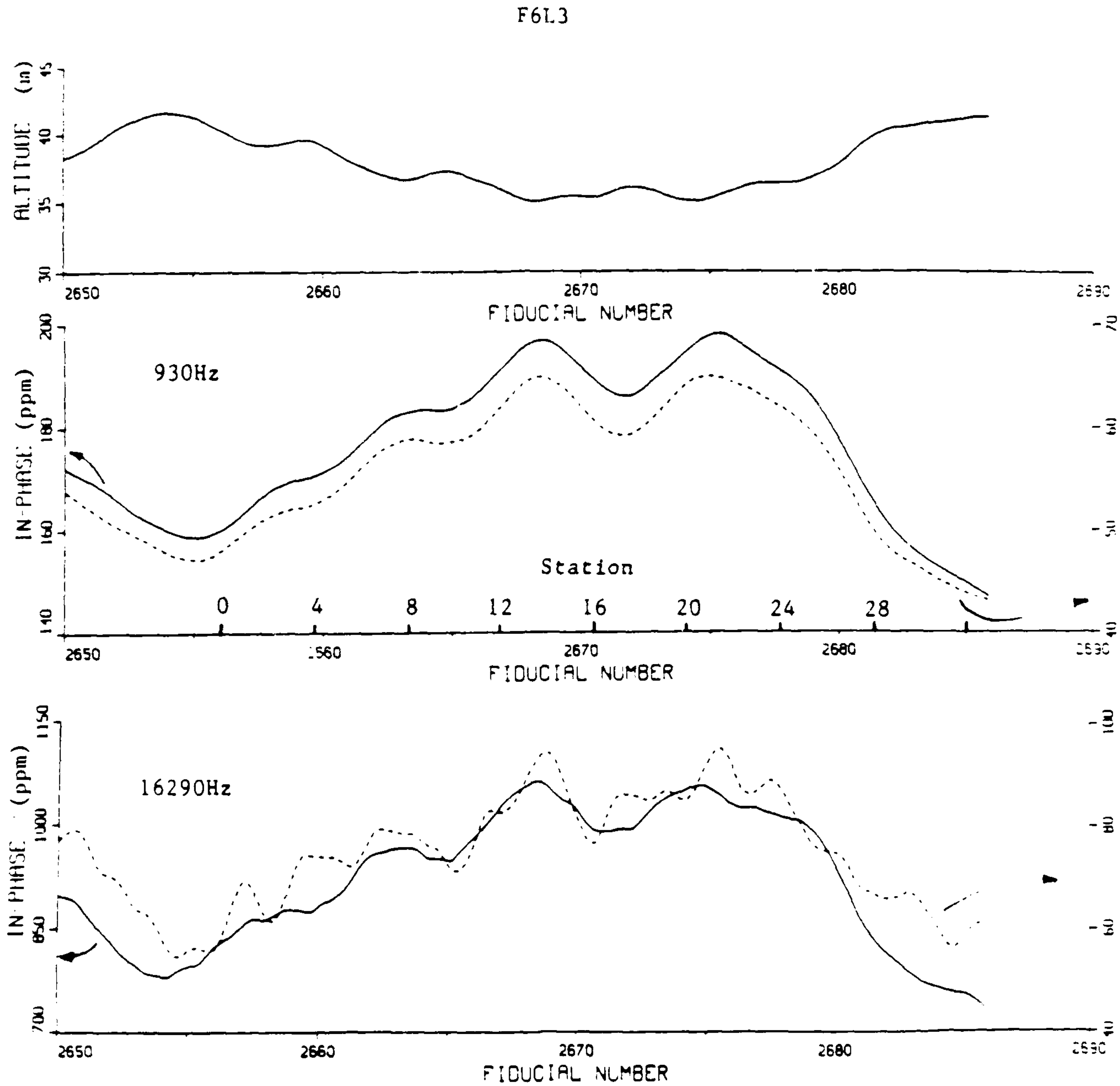


Fig.10 Field data. Line F6L3.

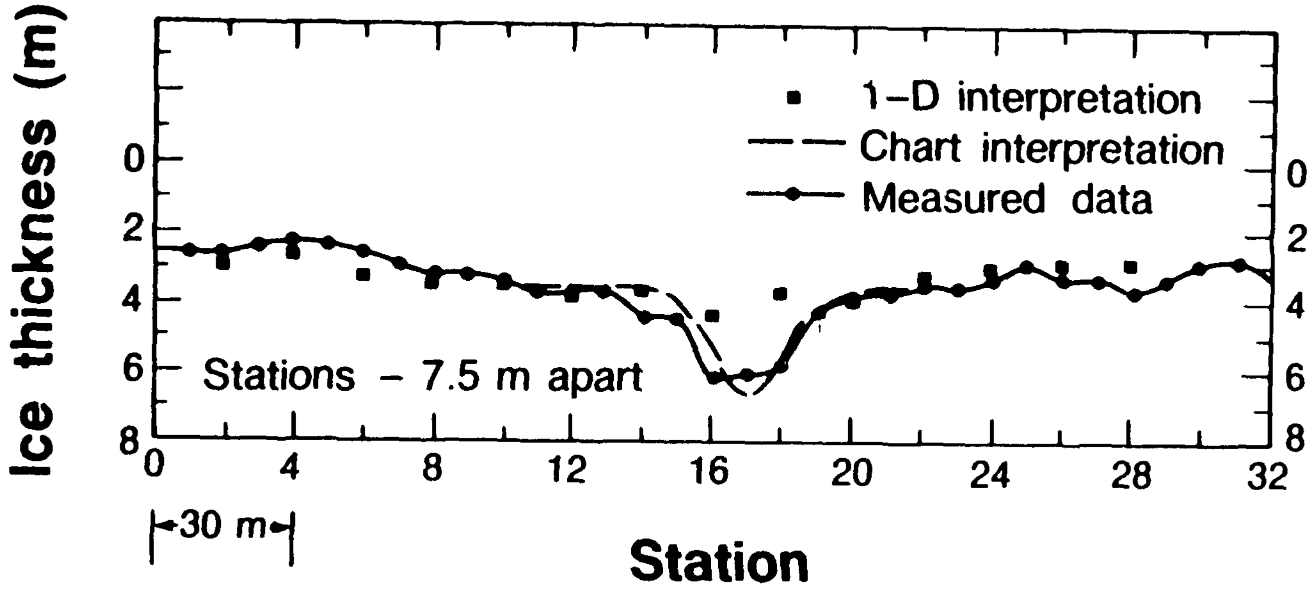


Fig.11 Interpretation of Field Data

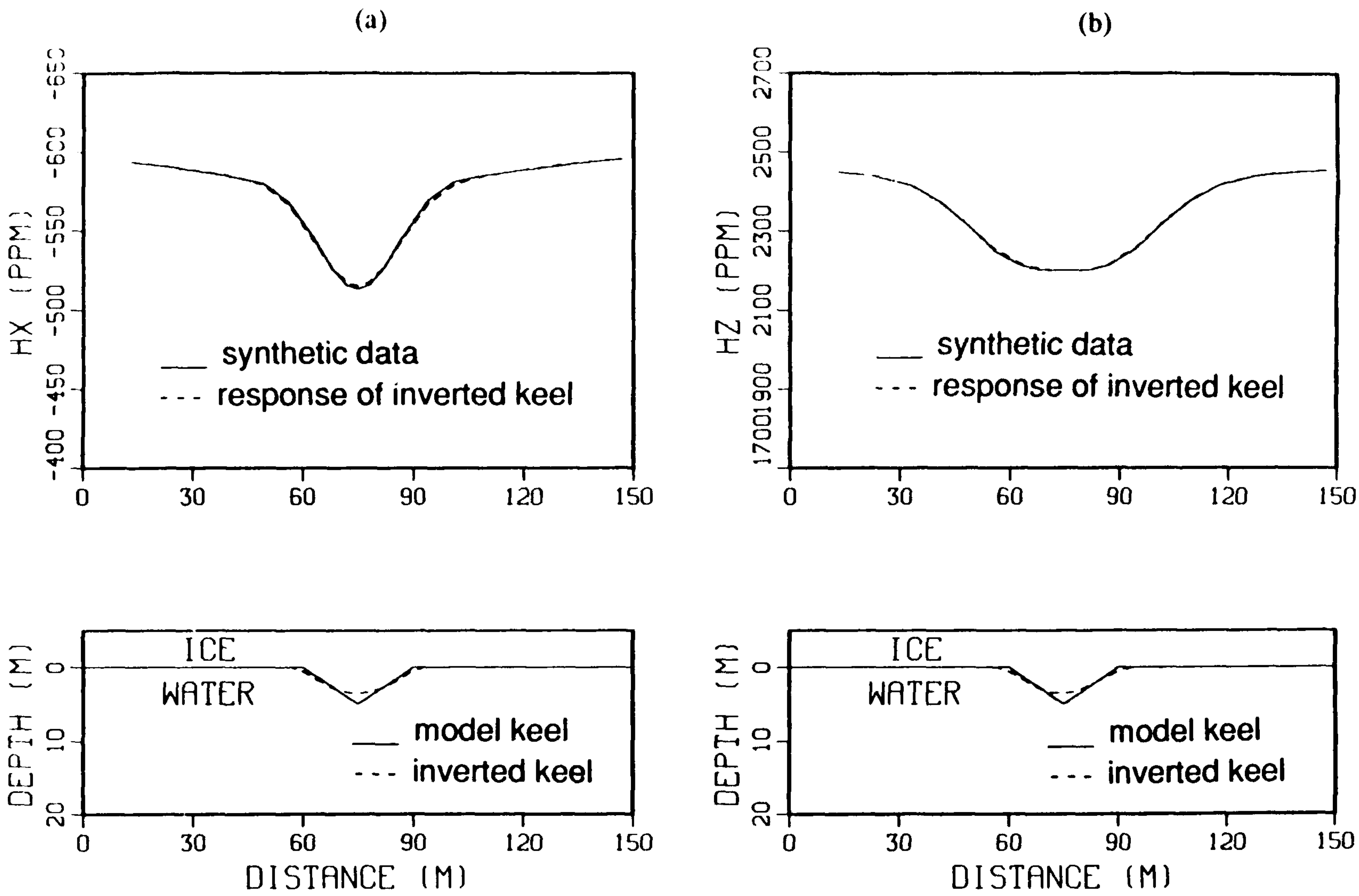


Figure 12 (a) Synthetic data for the HX system and inversion results, Model 1.
(b) Synthetic data for the HZ system and inversion results, Model 1.

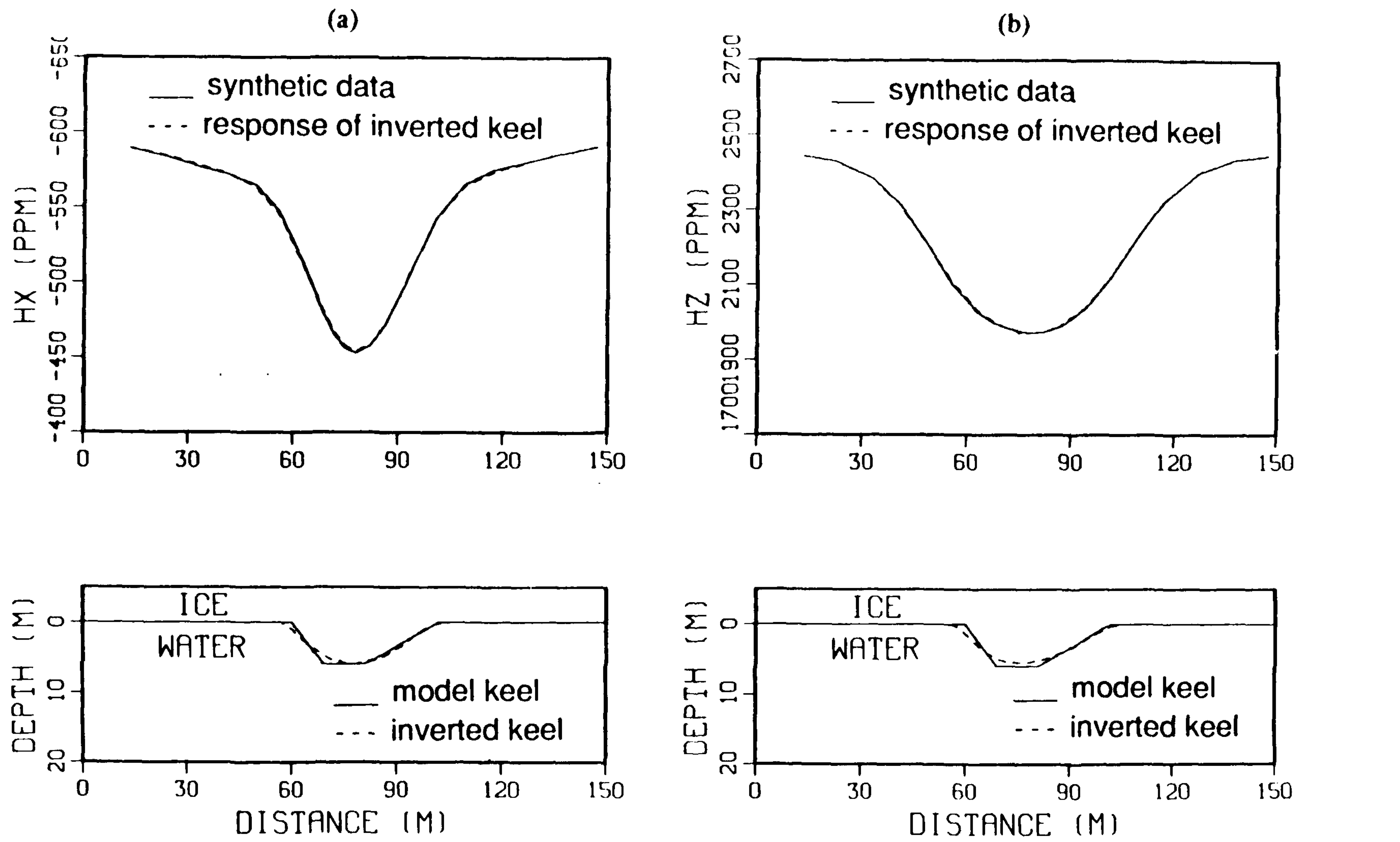


Figure 13 (a) Synthetic data for the HX system and inversion results, Model 2.
(b) Synthetic data for the HZ system and inversion results, Model 2.

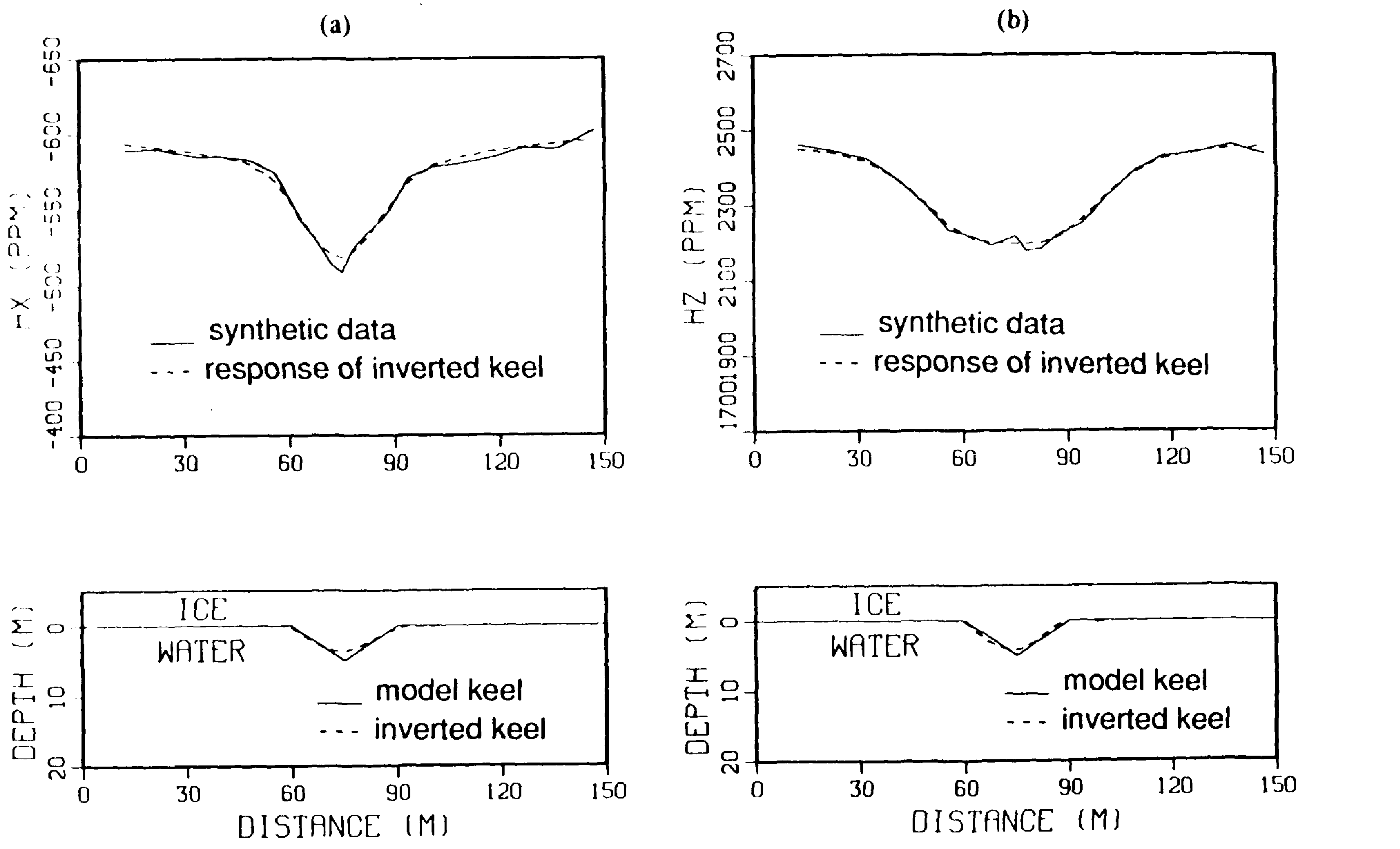


Figure 14 (a) Synthetic data with 5% added noise and inversion results for the HX system, Model 1. (b) Synthetic data with 5% added noise and inversion results for the HZ system, Model 1.

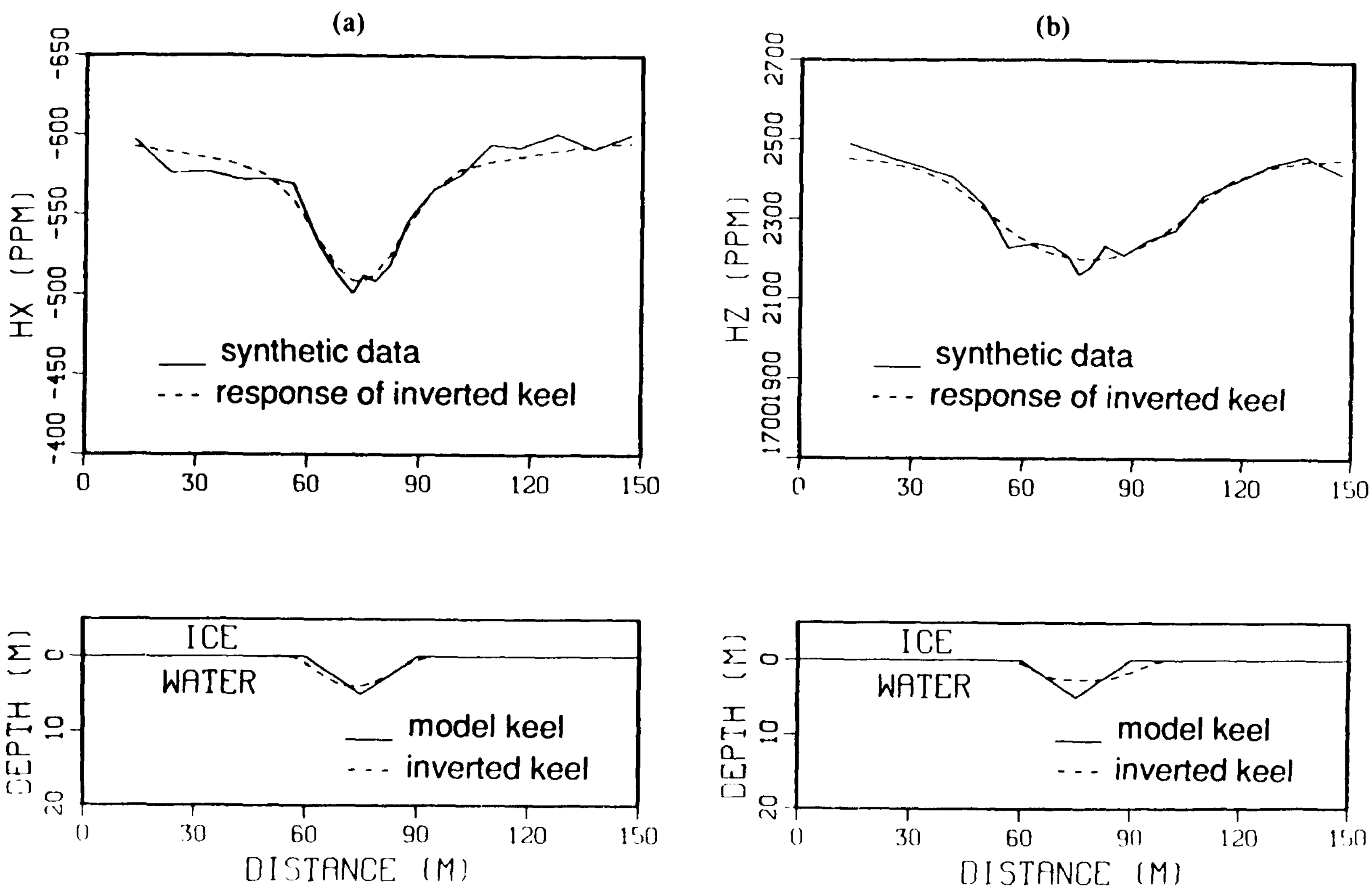


Figure 15 (a) Synthetic data with 10% added noise and inversion results for the HX system, Model 1. (b) Synthetic data with 10% added noise and inversion results for the HZ system, Model 1.

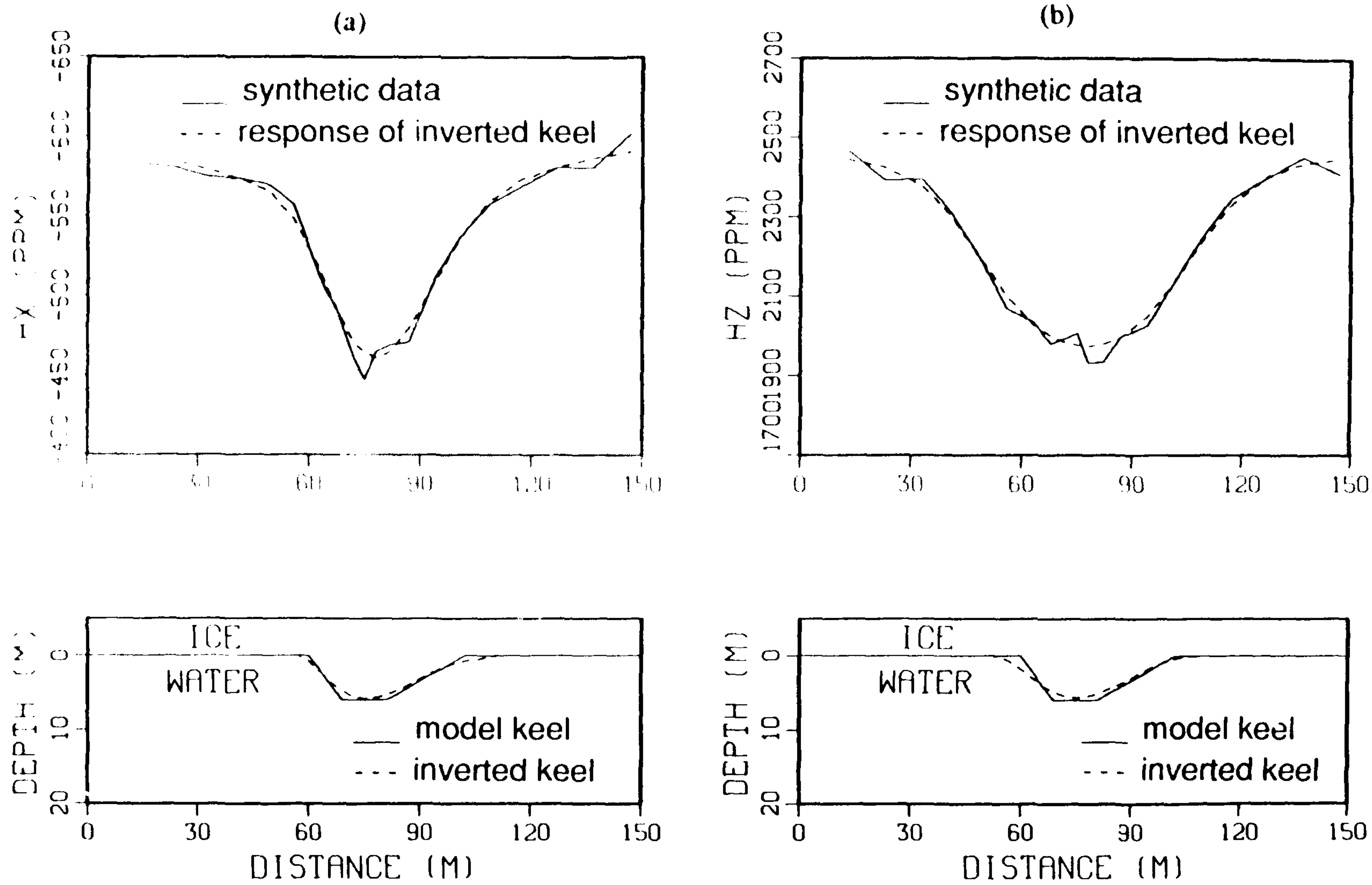


Figure 16 (a) Synthetic data with 5% added noise and inversion results for the HX system, Model 2. (b) Synthetic data with 5% added noise and inversion results for the HZ system, Model 2.

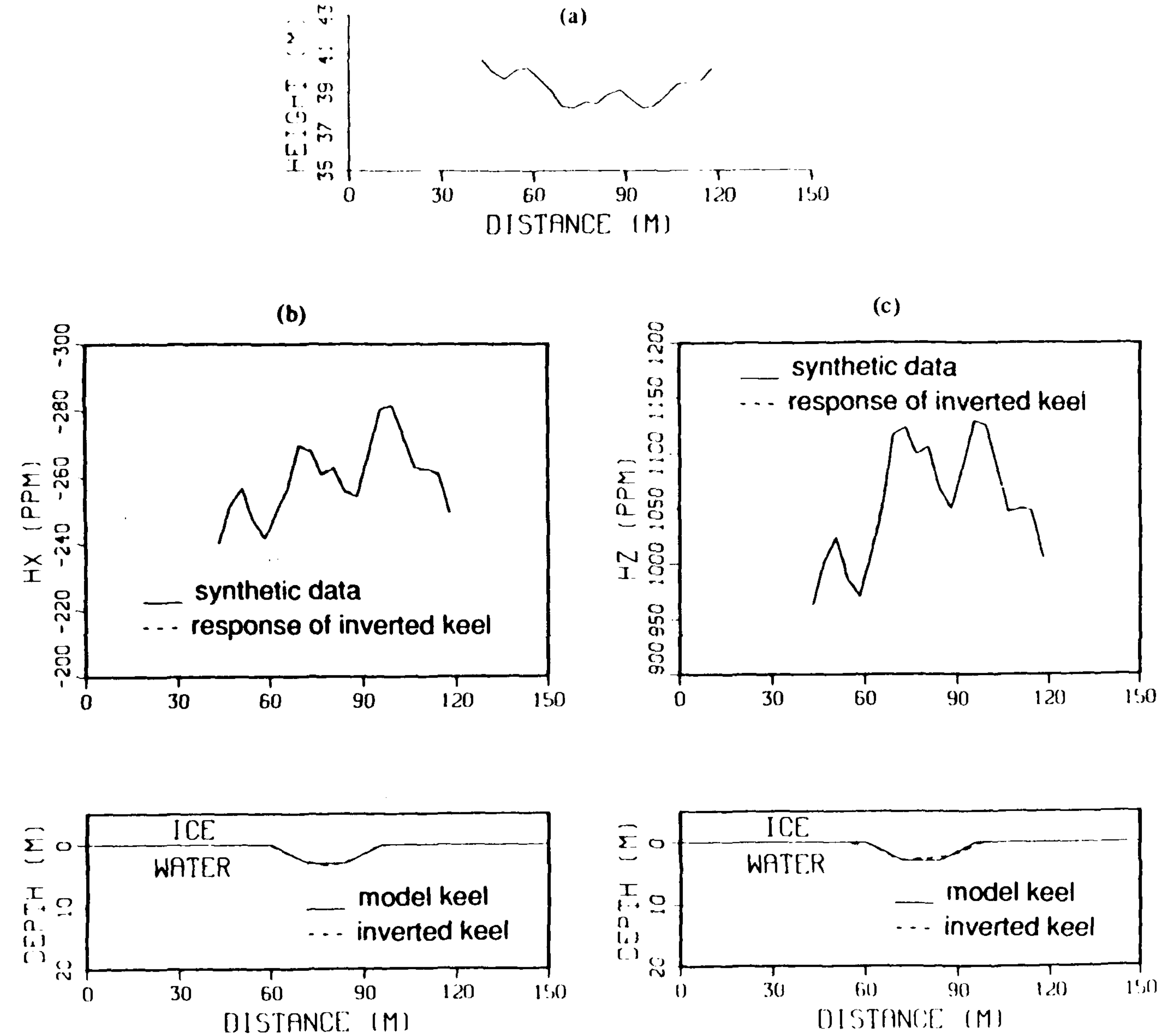


Figure 17 (a) Height of the system over the ice/water interface during the flight. (b) Synthetic data and inversion results for the HX system, Model 3. (c) Synthetic data and inversion results for the HZ system, Model 3.

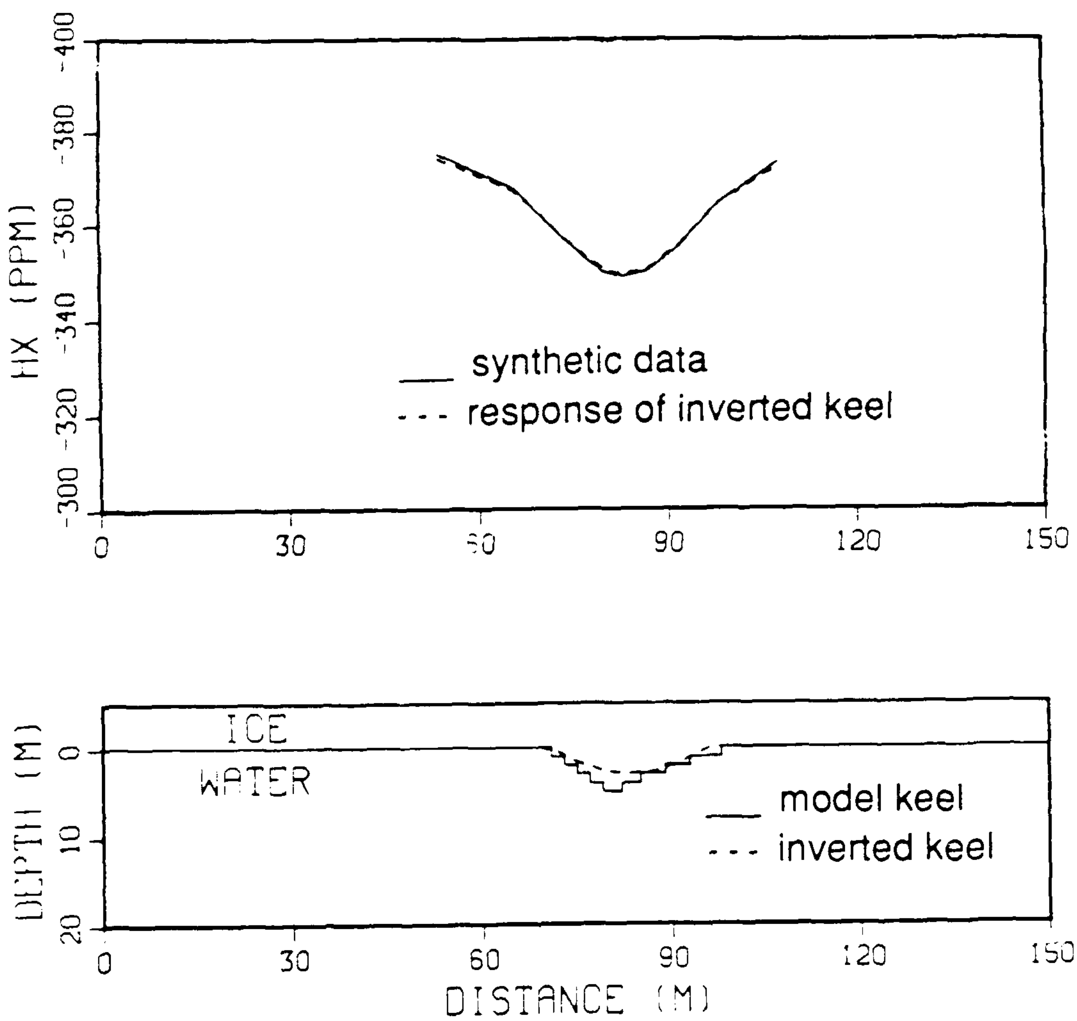


Figure 18 Synthetic data at 2500Hz and inversion results for the HX system, Model 4. The data shown are the scaled sum of the in-phase and quadrature components of the secondary magnetic field.

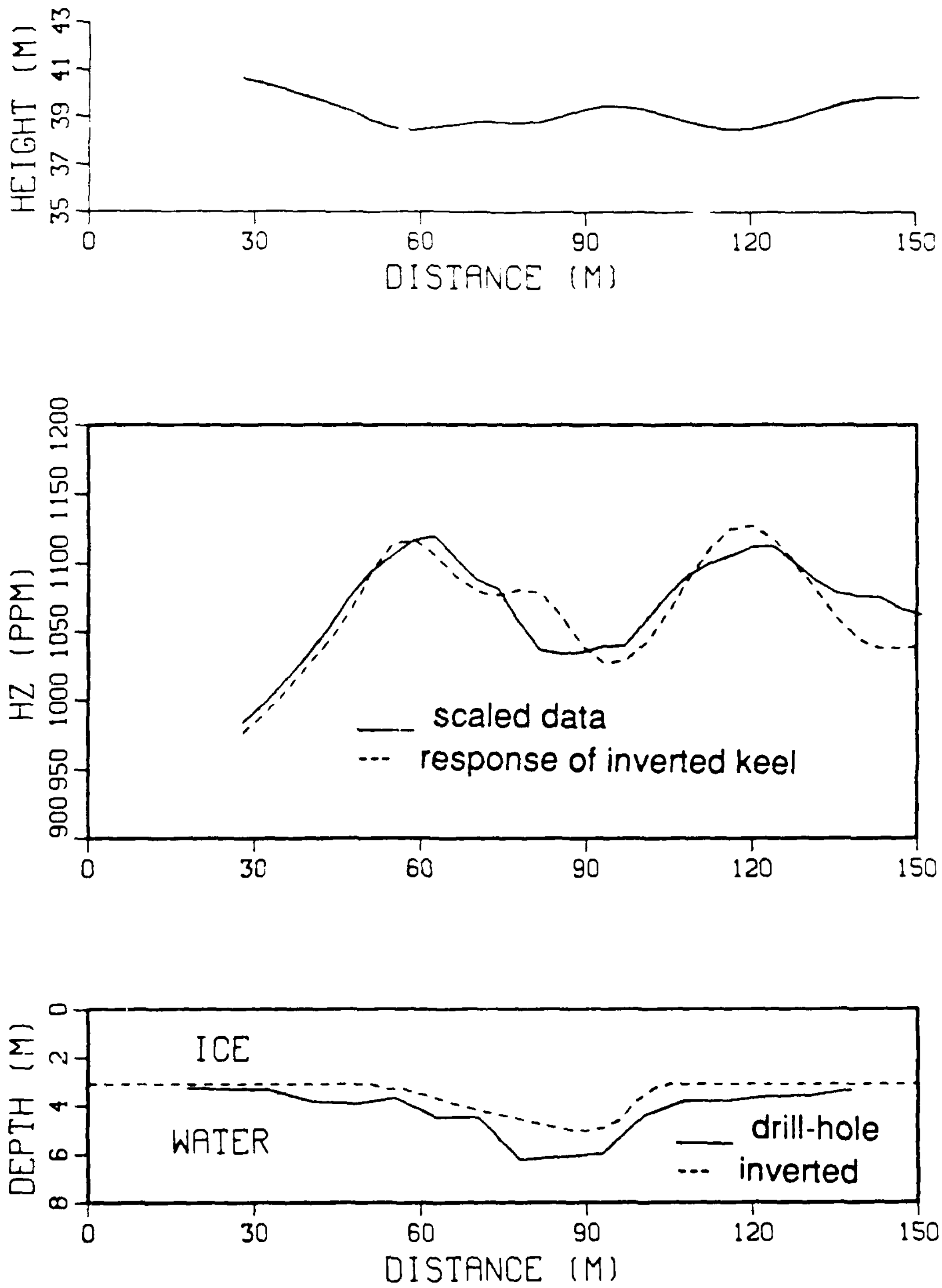
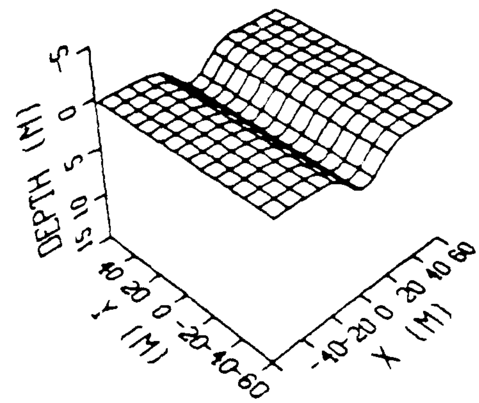
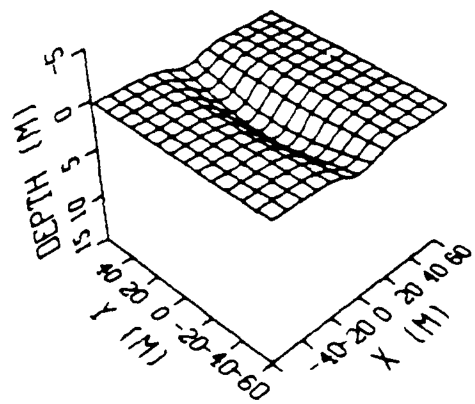


Figure 19 Scaled in-phase data at 16,290Hz and the inversion results for the HZ system. The data were collected at the Prudhoe Bay by Geotech Ltd. in 1985 on line F6L3.

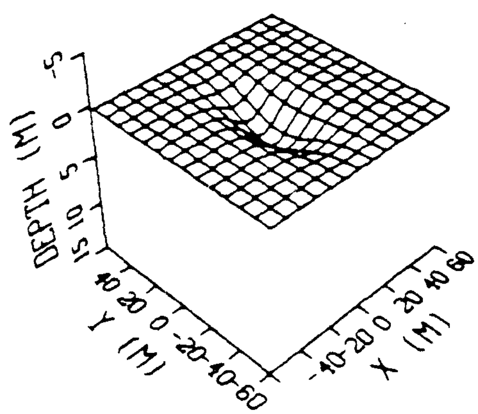
(a) s = inf.



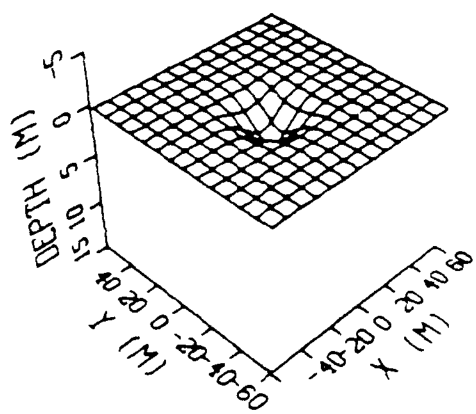
(b) s = 96m



(c) s = 48m



(d) s = 24m



(e) s = 12m

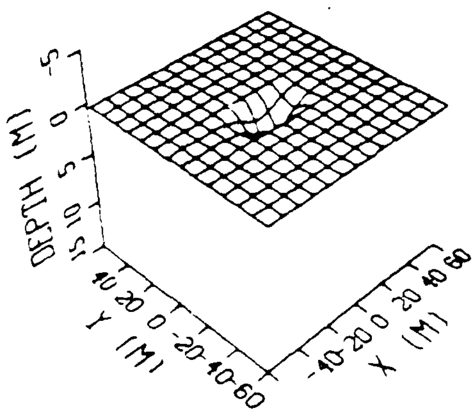


Figure 20 Sea ice keels of Model Set 1.
A = 3m, w = 24m.

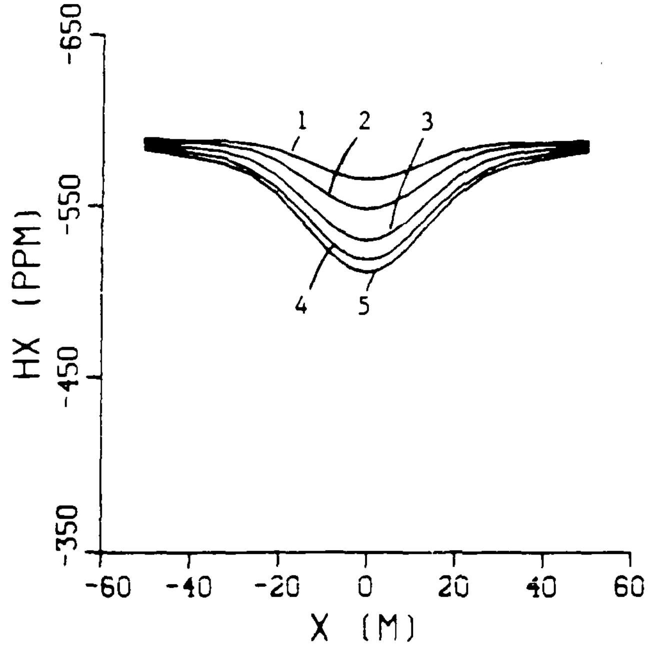


Figure 21(a) HX system response. Curves 1, 2, 3, 4, and 5 correspond to s = 12, 24, 48, 96, and inf meters respectively.

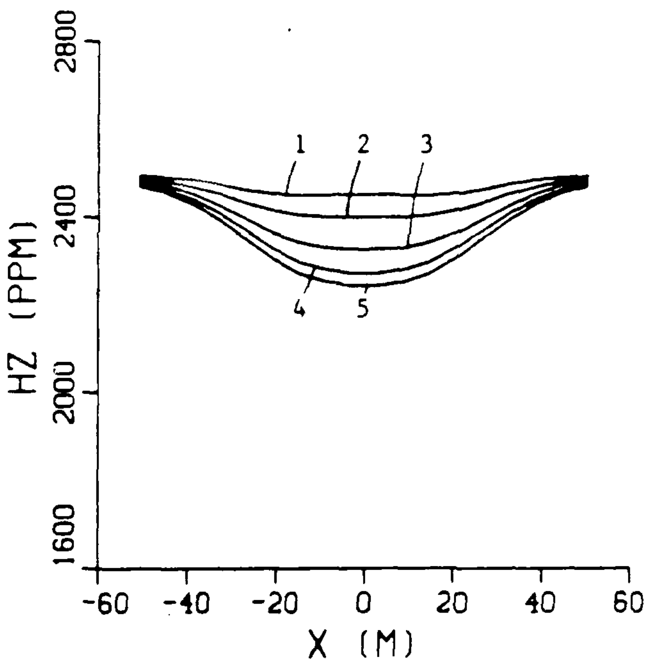


Figure 21(b) HZ system response. Curves 1, 2, 3, 4, and 5 correspond to s = 12, 24, 48, 96, and in: meters respectively.

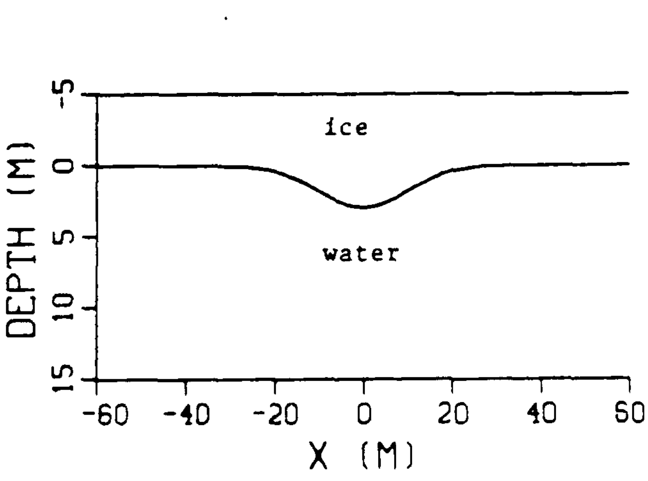


Figure 21(c) Cross section of the ice below the flight line.

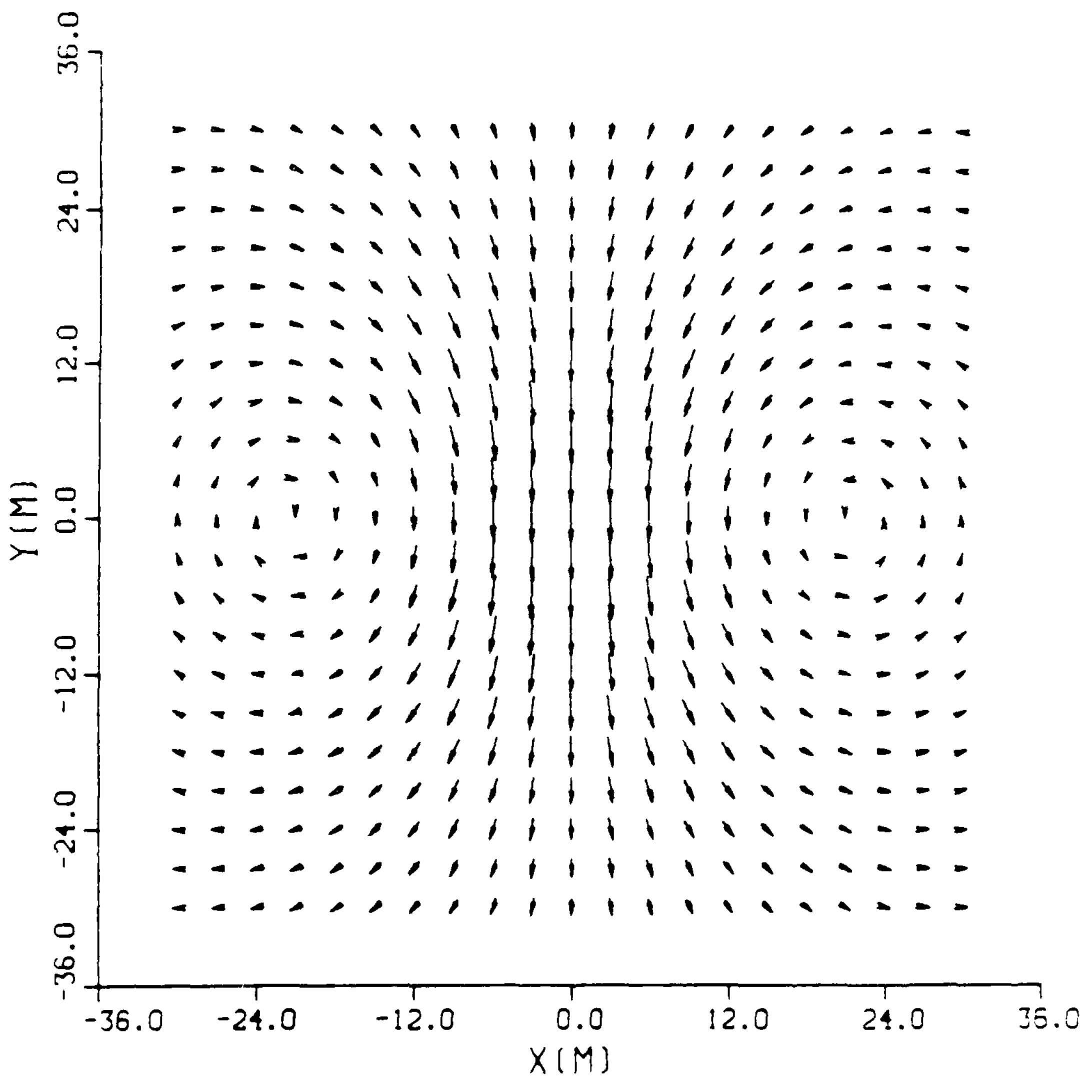


Figure 22 Surficial currents for a horizontal-axis transmitter, which is 30 meters above the center (0, 0).

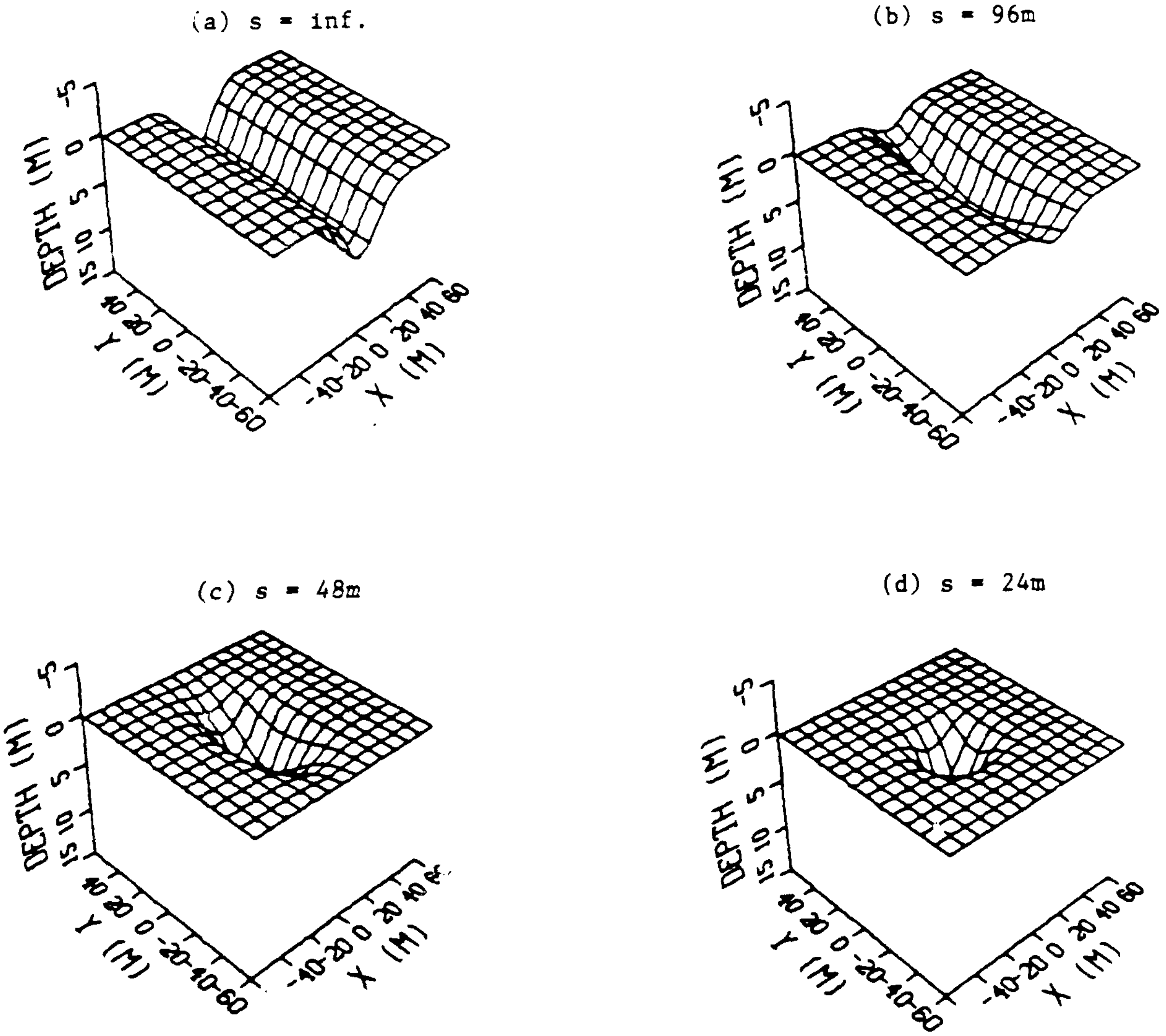


Figure 23 Sea ice keels of Model Set 2.
A = 6m, w = 24m.

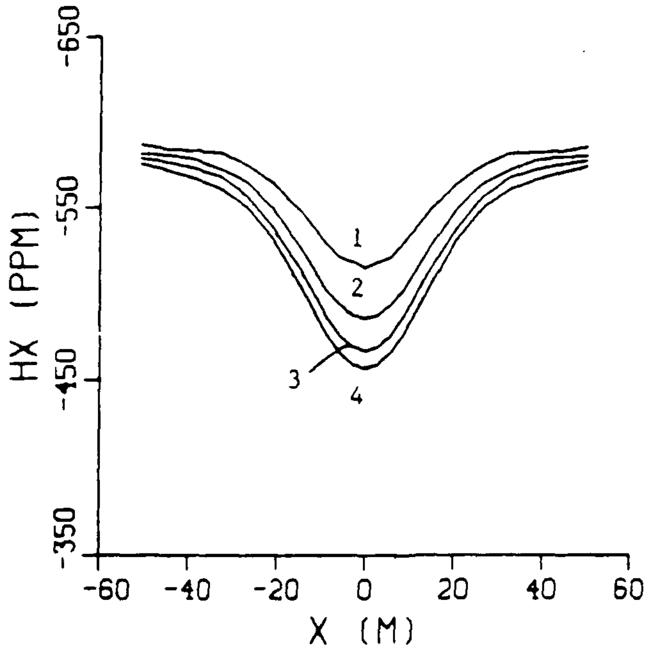


Figure 24(a) HX system response Curves 1, 2, 3, 4 correspond to s = 24, 48, 96, and inf. meters respectively.

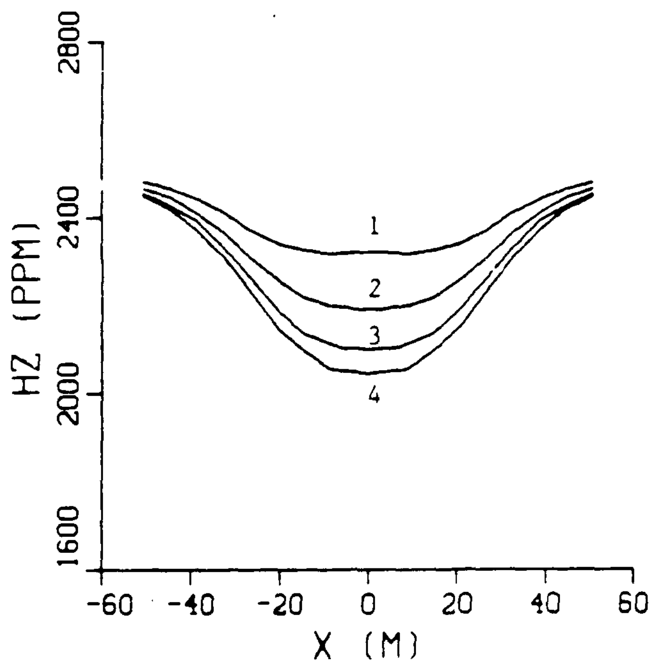


Figure 24(b) HZ system respons Curves 1, 2, 3, 4 correspond to s = 24, 48, 96, and inf. meters respectively.

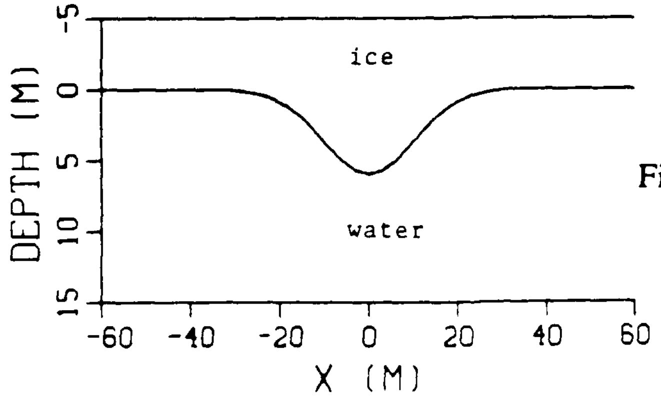
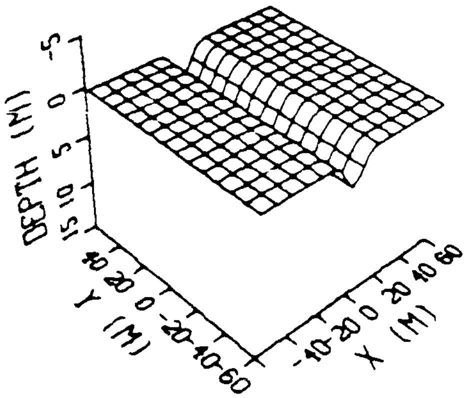
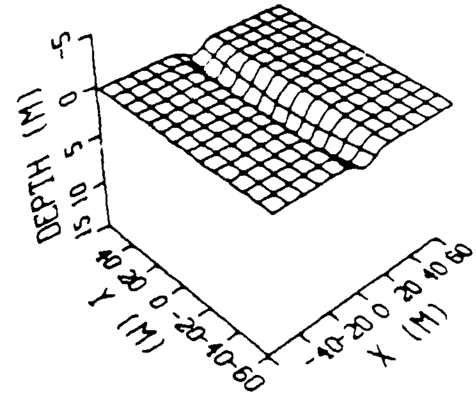


Figure 24(c) Cross section of the ic below the flight line.

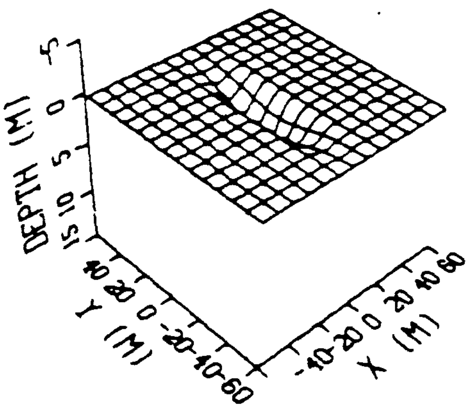
(a) s = inf.



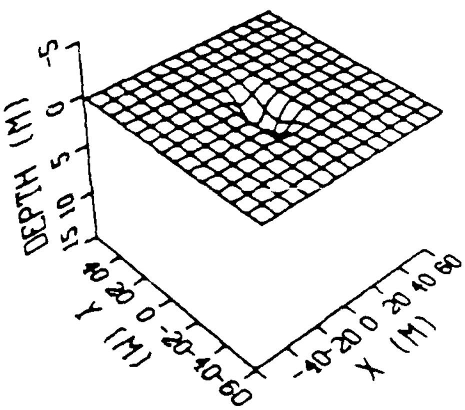
(b) s = 96m



(c) s = 48m



(d) s = 24m



(e) s = 12m

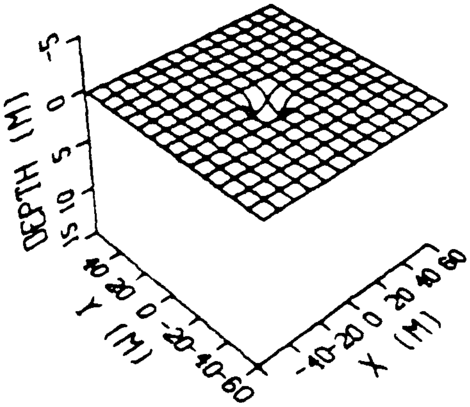


Figure 25 Sea ice keels of Model Set 3. A = 3m, w = 12m.

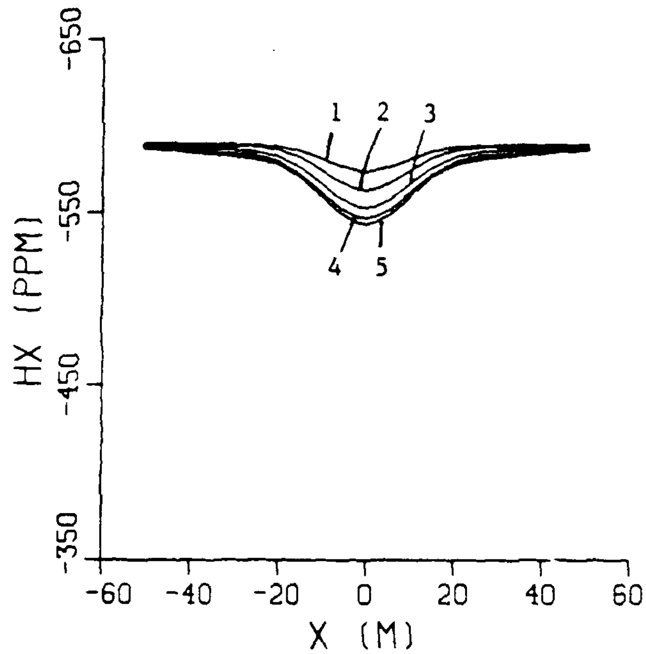


Figure 26(a) HX system respons
Curves 1, 2, 3, 4 and 5 corres-
pond to s = 12, 24, 48, 96, and i
meters respectively.

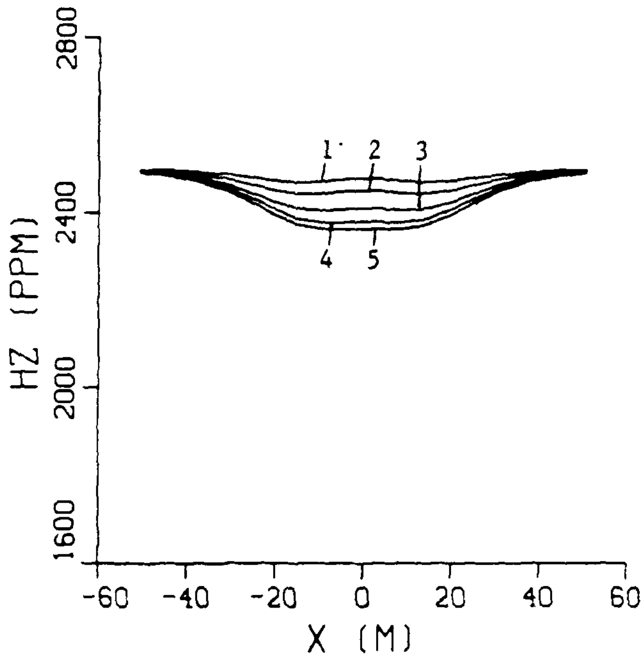


Figure 26(b) HZ system response
Curves 1, 2, 3, 4 and 5 corres-
pond to s = 12, 24, 48, 96, and in
meters respectively.

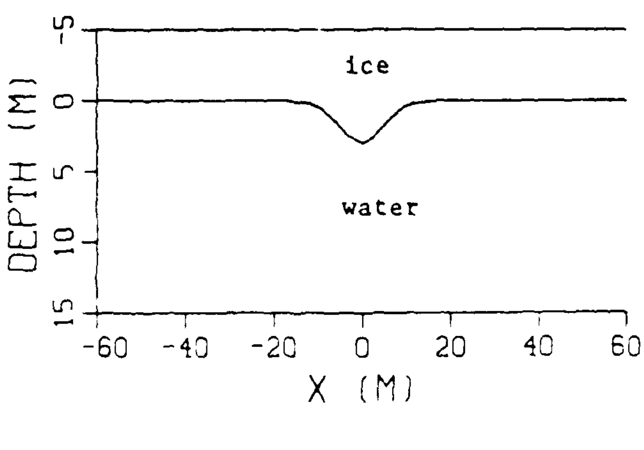


Figure 26(c) Cross section of the ic
below the flight line.

Appendix A

Computational Methods

A.1 The General Case

Consider an alternating magnetic dipole (current loop) source T, located in free space as shown in Figure A1. Its orientation is arbitrary and it is positioned over a homogeneous, perfectly conductive medium with three-dimensional surface relief. Due to this source, electric currents are induced on the surface of the medium and they give rise to the secondary magnetic field $\mathbf{H}_S$ in free space. It is our objective to calculate this quantity at any point above the surface S.

Here the height of the source above the medium is assumed to be small compared to the wavelength. The observation point is also assumed to be close to the source so that the electromagnetic field is quasi-static. In free space, $\nabla \times \mathbf{H}_S = 0$, and we may relate $\mathbf{H}_S$ to a scalar magnetic potential ϕ

$$\mathbf{H}_s = -\nabla x \phi \tag{1}$$

We also have $\nabla \bullet \mathbf{H}_S = 0$ and correspondingly

$$\nabla^2 \phi = 0 \tag{2}$$

Since the lower medium is assumed to be infinitely conductive, the normal component of the total magnetic field must vanish on the surface S, i.e.

$$H_{pn} + H_{sn} = 0 \qquad \text{on S} \tag{3}$$

Here, H_{pn} and H_{sn} are the normal components of the secondary and primary magnetic fields respectively. Hence,

$$\frac{\partial \phi}{\partial n}\Big|_S = - H_{sn}|_S = H_{pn}|_S \tag{4}$$

The Laplace equation (2) and the boundary condition (4) constitute the Neumann boundary value problem. That is, given the normal derivative of the potential on a surface S, we wish to calculate the potential itself in the free space. Once ϕ is found, $\mathbf{H}_S$ may be calculated from equation (1).

The solution of the Neumann problem outside a closed surface can be expressed as the potential of a surface charge layer (Graham, 1980)

$$\phi(o) = \int_S \frac{\xi(p)}{r_{PO}} ds \tag{5}$$

where $\xi(P)$ is a fictitious charge density function, r_{OP} is the distance between points P and O (see Figure 1). The charge density satisfies a Fredholm integral equation of the second kind

$$\xi(M) = - \frac{1}{2\pi} \frac{\partial \phi}{\partial n}\Big|_S - \frac{1}{2\pi} \int_S \xi(p) \frac{\cos(\mathbf{r}_{PM}, \mathbf{N})}{r_{PM}^2} ds \qquad \text{on } S \tag{6}$$

where ($\mathbf{r}_{PM}$, $\mathbf{N}$) is the angle between $\mathbf{r}_{PM}$ (the vector connecting P to M) and $\mathbf{N}$ (the unit normal vector at M). In our problem, the surface S extends to infinity and equations (5) and (6) are still valid.

To solve the integral equation (6), we use the successive approximation method (Mikhlin, 1964). The initial solution (first iteration) is assumed to be the first term on the right hand side of equation (6), i.e.

$$\xi(M) = - \frac{1}{2\pi} \frac{\partial \phi}{\partial n}\Big|_S = - \frac{1}{2\pi} H_{pn}$$

We then use this value in the integral in the equation to compute an improved value of $\xi(M)$ and so on. Once the charge density is known, the secondary magnetic field can be calculated from the following equation, which is obtained by combining equations (1) and (5)

$$\mathbf{H}_S = \int_S \frac{\xi(p)}{r_{PO}^3} \mathbf{r}_{PO}\, ds \tag{7}$$

Here $\mathbf{r}_{PO}$ is the vector connecting P to O.

In the above context, the electrodynamic problem is reduced to a potential problem under the quasi-static field assumption. The charge distribution on the surface of the conductive medium is computed first. The secondary magnetic field in the free space is then found by summing up the contributions from the individual charges. This is analogous to the integral equation approach for solving electromagnetic scattering problem (Parry and Ward, 1970), where the equivalent electric and magnetic currents are first sought; the electromagnetic field are then obtained by integrating the contributions from the current distribution.

In the special case where the surface S forms a plane, the solution given by equation (7) for an oscillating magnetic dipole source is identical to that obtained by the method of images (Jackson, 1975). To illustrate our computational method, we now show this to be true for a vertical magnetic dipole source. The coordinate system is chosen such that the z axis is pointing vertically downward. The XOY plane is on the surface of the conductor which occupies the half space $z > 0$. The dipole source is h meters above the plane on the z axis and points in the positive z direction.

Since $\mathbf{r}_{PM}$ is perpendicular to $\mathbf{N}$ (cf. Figure 1), equation (6) reduces to

$$\xi(M) = -\frac{1}{2\pi}\frac{\partial\phi}{\partial n}\Big|_S = -\frac{1}{2\pi} H_{pn} \tag{8}$$

On the conductor surface, the normal component of the primary magnetic field is

$$H_{pn} = -\frac{1}{4\pi}\frac{2h^2-x^2-y^2}{(x^2+y^2+h^2)^{5/2}} \tag{9}$$

Substituting equations (8) and (9) into (5), we obtain

$$\phi(x, y, z) = \int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \frac{\xi(x', y', 0)}{[(x-x')^2+(y-y')^2+z^2]^{1/2}}\, dx'dy' \qquad (z<0)$$

$$\phi(0, 0, z) = \frac{1}{8\pi^2}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} \frac{2h^2-x'^2-y'^2}{(x'^2+y'^2+h^2)^{5/2}}\frac{dx'dy'}{(x'^2+y'^2+z^2)^{1/2}}$$

Let $x' = R\cos\theta$, $y' = R\sin\theta$, $dx'dy' = Rd\theta dR$, then

$$\phi(0, 0, z) = \frac{1}{8\pi^2}\int_{0}^{+\infty}\int_{0}^{2\pi} \frac{2h^2-R^2}{(R^2+h^2)^{5/2}}\frac{Rd\theta dR}{(R^2+z^2)^{1/2}}$$

Furthermore, let $r^2 = R^2 + h^2$, $RdR = rdr$, then

$$\phi(0, 0, z) = \frac{1}{4\pi}\int_{h}^{\infty} \frac{3h^2-r^2}{r^4(r^2-h^2+z^2)^{1/2}}\, dr$$

The above integration can be carried out and the result is as follows

$$\phi(0, 0, z) = \frac{1}{4\pi}\frac{1}{(h-z)^2} \qquad (z<0)$$

The vertical component of the secondary magnetic field on z axis is

$$H_{s\,z}(0, 0, z) = -\frac{\partial\phi}{\partial z} = \frac{1}{2\pi}\frac{1}{(h-z)^3} \tag{10}$$

and as required is identical to the image field.

A.2 The 2-D keel

In the 2-D case, the geometry of the ice-water interface does not change in the strike direction (y-direction here). The relief of the interface is only a function of x , i.e. $h(x,y) = h(x)$. In this case, the potential of the scattered magnetic field is given by,

$$\phi(x, y, z) = \iint_{-\infty}^{+\infty} \frac{\xi(x',y')\sqrt{1+[dh(x')/dx']^2}}{\sqrt{(x-x')^2+(y-y')^2+(z-h(x'))^2}}\,dx'dy' \tag{11}$$

and the surface charge density $\xi(x, y)$ satisfies

$$\xi(x, y) = -\frac{1}{2\pi}H_{pn}(x,y) - \frac{1}{2\pi}\iint_{-\infty}^{+\infty}\xi(x',y')\,\frac{(x-x')dh(x')/dx' - (h(x)-h(x'))}{[(x-x')^2+(y-y')^2+(h(x)-h(x'))^2]^{\frac{3}{2}}}$$

$$\cdot\left[\frac{1+(dh(x')/dx')^2}{1+(dh(x)/dx)^2}\right]^{\frac{1}{2}}dx'dy' \tag{12}$$

Here $H_{pn}(x,y)$ is the normal component of the primary magnetic field at the ice-water interface.

Notice that in equations 11 and 12 the integral with regard to y' is a convolution. Taking the Fourier transform of both sides we get

$$\phi(x, k_y, z) = \int_{-\infty}^{+\infty}\phi(x, y, z)\,e^{-ik_y y}\,dy$$

$$= 2\int_{-\infty}^{+\infty}\xi(x',k_y)\sqrt{1+[dh(x')/dx']^2}\,K_0(\rho|k_y|)\,dx' \tag{13}$$

and

$$\xi(x, k_y) = -\frac{1}{2\pi}H_{pn}(x,k_y) - \frac{1}{2\pi}\int_{-\infty}^{+\infty}\xi(x',k_y)\,f(x,x',k_y)\,dx' \tag{14}$$

where the kernel of the integration is

$$f(x, x', k_y) = 2\left[\frac{1+(dh(x')/dx')^2}{1+(dh(x)/dx)^2}\right]^{\frac{1}{2}} \frac{(x-x')dh(x')/dx' - (h(x)-h(x'))}{[(x-x')^2 + (h(x)-h(x'))^2]^{\frac{1}{2}}}$$

$$\cdot |k_y| \, K_1(\rho' |k_y|) \tag{15}$$

In the above equations

k_y = angular wave number in the y direction

$$\rho = \sqrt{(x-x')^2 + (z - h(x'))^2}$$

$$\rho' = \sqrt{(x-x')^2 + (h(x) - h(x'))^2}$$

$K_0(\ )$ = modified Bessel function of the zeroth order. Second kind.

$K_1(\ )$ = modified Bessel function of the first order. Second kind.

We have thus simplified the problem by decomposing a two dimensional integral equation into a number of integral equations in one dimension.

For the case where the source is a horizontal or vertical magnetic dipole, the normal component of the primary magnetic field can be analytically transformed into the wave-number domain (i.e. $H_{pn}(x, k_y)$) as follows.

The normal component of the primary magnetic field at the ice-water interface is

$$H_{pn}(x,y) = \mathbf{H}_p(x, y)\cdot\mathbf{n}(x, y) \tag{16}$$

where $\mathbf{n}(x, y)$ is the outward normal at point (x, y) on the interface. It is given by

$$\mathbf{n} = \left(\frac{h'(x)}{\sqrt{1 + [h'(x)]^2}}, 0, -\frac{1}{\sqrt{1 + [h'(x)]^2}} \right) \tag{17}$$

where $h'(x) = dh(x)/dx$. Thus

$$H_{pn}(x,y) = \frac{h'(x)}{\sqrt{1+[h'(x)]^2}} H_{px} - \frac{1}{\sqrt{1+[h'(x)]^2}} H_{pz} \tag{18}$$

(a) Horizontal magnetic dipole source located at (x_S, y_S=0, z_S)

$$H_{px}(x,y) = \frac{2(x-x_s)^2 - y^2 - (h(x)-z_s)^2}{4\pi r^5}$$

$$H_{pz}(x,y) = \frac{3(x-x_s)(h(x)-z_s)}{4\pi r^5}$$

Here

$$r = \sqrt{(x-x_s)^2 + y^2 + (h(x)-z_s)^2}$$

Substituting the above equations into (18) and taking the Fourier transform give

$$H_{pn}(x,k_y) = \frac{1}{2\pi\rho^2\sqrt{1+[h'(x)]^2}} \Big\{ [\, h'(x)(x-x_s)^2 - (x-x_s)(h(x)-z_s)] |k_y|^2 K_0(\rho|k_y|)$$
$$- \{[(h(x)-z_s)^2 - (x-x_s)^2]\, h'(x) + 2(x-x_s)(h(x)-z_s)\} \frac{|k_y|}{\rho} K_1(\rho|k_y|) \Big\} \tag{19}$$

where

$$\rho = \sqrt{(x-x_s)^2 + (h(x)-z_s)^2}$$

As $k_y \rightarrow 0$, the asymptotic forms of the modified Bessel functions are

$$K_0(\rho|k_y|) \rightarrow -\ln \rho|k_y|$$

$$K_1(\rho|k_y|) \rightarrow \frac{1}{\rho|k_y|}$$

Therefore

$$H_{pn}(x,k_y=0) = \frac{-1}{2\pi\rho^4\sqrt{1+[h'(x)]^2}} \{[(h(x)-z_s)^2 - (x-x_s)^2]\, h'(x) + 2(x-x_s)(h(x)-z_s)\} \tag{20}$$

(b) Vertical magnetic dipole source located at (x_S, y_S=0, z_S)

$$H_{px}(x, y) = \frac{3(x - x_s)(h(x) - z_s)}{4\pi r^5}$$

$$H_{pz}(x, y) = \frac{2(h(x) - z_s)^2 - y^2 - (x - x_s)^2}{4\pi r^5}$$

$$H_{pn}(x,k_y) = \frac{1}{2\pi\rho^2\sqrt{1+[h'(x)]^2}} \Big\{ [-(h(x)-z_s)^2 + h'(x)(x-x_s)(h(x)-z_s)] |k_y|^2 K_0(\rho|k_y|)$$
$$- [(h(x)-z_s)^2 - (x-x_s)^2 - 2h'(x)(x-x_s)(h(x)-z_s)] \frac{|k_y|}{\rho} K_1(\rho|k_y|) \Big\} \tag{21}$$

$$H_{pn}(x,k_y=0) = \frac{-1}{2\pi\rho^4\sqrt{1+[h'(x)]^2}} \{(h(x)-z_s)^2 - (x-x_s)^2 - 2h'(x)(x-x_s)(h(x)-z_s)\} \tag{22}$$

Note here that the kernel f(x, x', k_y) is independent of the source and the receiver positions. Therefore it may be calculated once for each k_y value and stored for the computation of a complete profile of AEM system response. This is quite economical but impossible in the 3-D case where the matrix is too large to store. At x = x', the kernel has a singularity. But this presents no difficulty for the numerical computation because it is integrable in the sense of Cauchy principal value.

The integral equation (14) can be solved for ξ(x, k_y) using successive approximation method identical to the one suggested above for the general case. This needs to be done at a number of positive k_y harmonics (including k_y =0). The values of ξ(x, k_y) at the negative k_y harmonics may be easily obtained by its property of symmetry. As usual, the sampling in the k_y space is done on a logarithmic scale.

Once the charge density is known, the x- and z- components of the scattered magnetic field may be directly computed from the following equations

$$H_{sx}(x, k_y, z) = -\frac{\partial \phi(x, k_y, z)}{\partial x}$$

$$= 2\int_{-\infty}^{+\infty} (x - x')\, \xi(x', k_y) \sqrt{1 + [dh(x')/dx']^2}\, \frac{|k_y|}{\rho} K_1(\rho |k_y|)\, dx' \tag{23}$$

and

$$H_{sz}(x, k_y, z) = -\frac{\partial \phi(x, k_y, z)}{\partial z}$$

$$= 2\int_{-\infty}^{+\infty} (z - h(x'))\, \xi(x', k_y) \sqrt{1 + [dh(x')/dx']^2}\, \frac{|k_y|}{\rho} K_1(\rho |k_y|)\, dx' \tag{24}$$

Now that the scattered magnetic field is obtained in the wave-number domain its inverse Fourier transform will yield the desired result in the space domain. But prior to performing the inverse Fourier transform, the field values at the logarithmically-spaced points in the k_y space need to be interpolated for uniformly spaced values. This is accomplished using cubic spline interpolation.

The above algorithm has been successfully implemented and its speed is more than 20 times faster than that of the general 3-D algorithm. The computation of a 20 point profile of the AEM system response takes about only 10 seconds CPU time on the IBM 3090.

A.3 Computational Check

A scale model experiment was also performed to check the numerical solution. Aluminum was used to simulate the infinitely conductive medium at a scale of 1:250. A model airborne electromagnetic system was built at the same scale and was " flown " at a field height of 10m. The system consisted of a coplanar, vertical axis transmitter and receiver, which were operated at 6 kHz. Details of the model are shown in Figure A2 which also exhibits the traversed feature.

The cross section of the indented surface is a Gaussian curve that simulates a smooth ice keel. Its relief is given by

$$t(x) = A \exp\left(-\frac{x^2}{2\tau^2}\right) \qquad (25)$$

and

$$\tau = 0.425W$$

Here

x = distance from the keel center line

t = keel thickness

A = drawdown or maximum keel thickness

W = keel width at half drawdown

The shape of the keel does not change along its strike direction. For this scale model, A and W were taken to be 3.4m and 21m respectively. We chose this type of surface because its simulation on the computer is simple as there are only two input parameters. Furthermore, it is easy to adjust these two parameters to simulate any real sea ice keel. The measurements and numerical calculation results (two iterations) for this model are displayed in Figure A3 , which shows an excellent agreement between the numerical and experimental data.

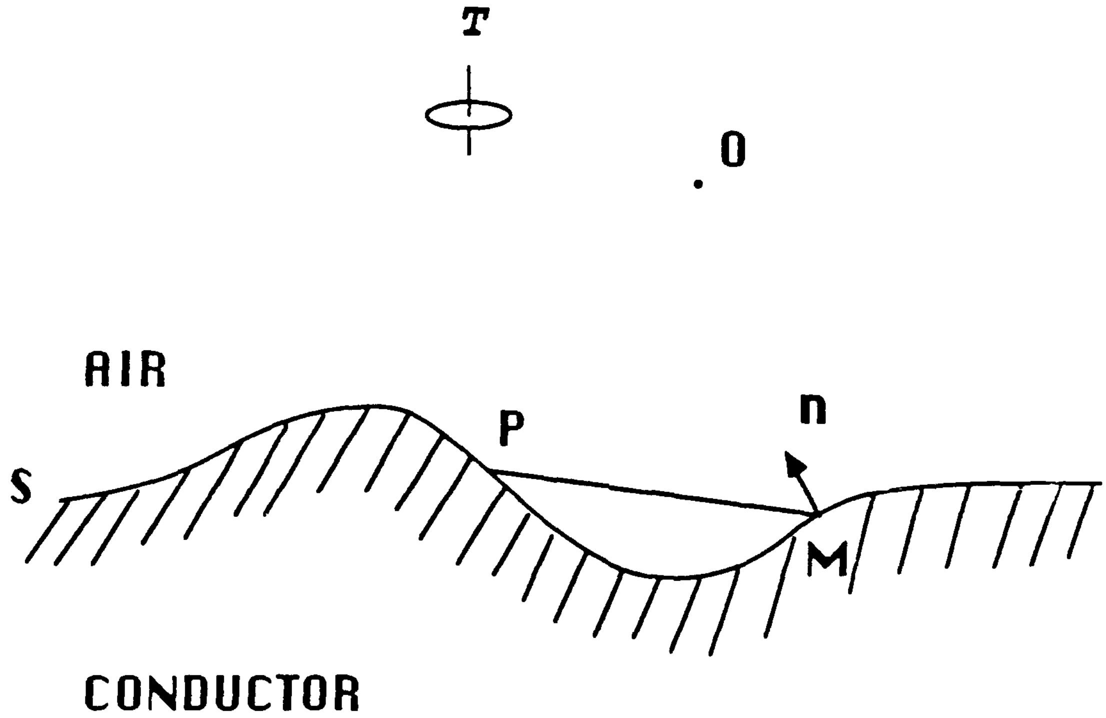


Fig.A1 **The Neumann Problem**

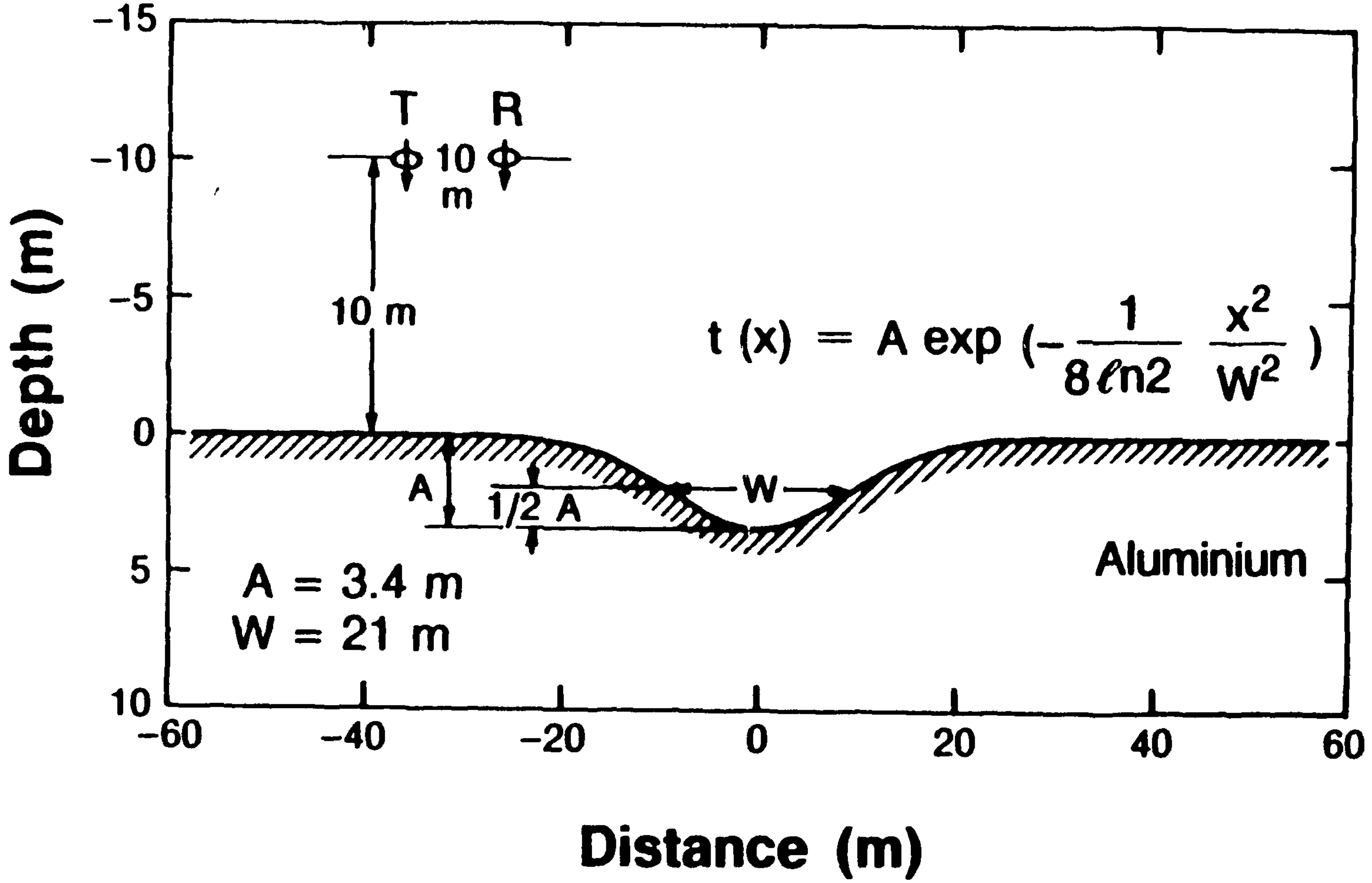


Fig.A2 Scale Model Geometry

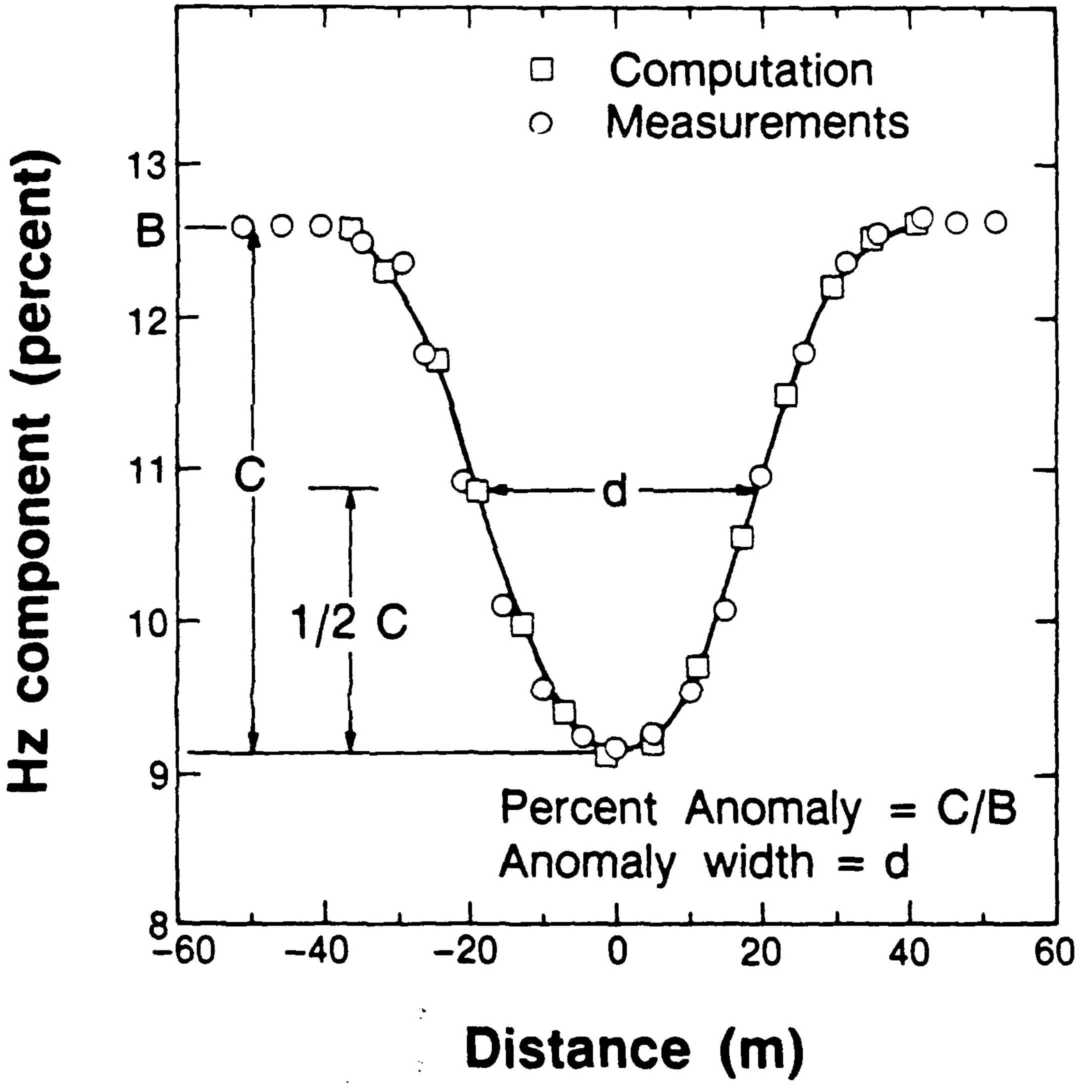


Fig.A3 Comparison of Measurements and Computation

Appendix B

This Appendix includes a listing of three Fortran programs: SURF3D, ICE2D1, and INVSM.

SURF3D is a program for computing the AEM system response for a general three-dimensional ice keel. It may also be used to calculate the surface current distribution on the ice/water interface.

ICE2D1 is a fast modeling program for the computation of AEM system response over a two-dimensional (2-D) ice keel. It is generally more than 20 times faster than using SURF3D for an identical problem.

INVSM is an inversion program for obtaining a 2-D smooth ice keel from the AEM data. It uses ICE2D1 for modeling AEM system response. Here it is acknowledged that part of this program is adapted from program OCCAM1 by Constable.

```
SSSSSSSSSS    0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000    SSSSSSSSSS
SSSSSSSSSS    000000000000000000000000  Digital Equipment Corporation - VAX/VMS Version V4.7  000000000000000000000000    SSSSSSSSSS
SSSSSSSSSS    0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000    SSSSSSSSSS
```

```
                 GGGG  U   U   III   M   M   III   N   N
                G      U   U    I    MM MM    I    N   N
                G      U   U    I    M M M    I    NN  N
                G      U   U    I    M   M    I    N N N
                G GGG  U   U    I    M   M    I    N  NN
                G   G  U   U    I    M   M    I    N   N
                 GGG   UUUUU   III   M   M   III   N   N

  SSSSSSSS  UU      UU  RRRRRRRR    FFFFFFFFFF    333333    DDDDDDDD
  SSSSSSSS  UU      UU  RRRRRRRR    FFFFFFFFFF    333333    DDDDDDDD
SS          UU      UU  RR      RR  FF          33      33  DD      DD
SS          UU      UU  RR      RR  FF          33      33  DD      DD
SS          UU      UU  RR      RR  FF                  33  DD      DD
SS          UU      UU  RR      RR  FF                  33  DD      DD
  SSSSSS    UU      UU  RRRRRRRR    FFFFFFFF          33    DD      DD
  SSSSSS    UU      UU  RRRRRRRR    FFFFFFFF          33    DD      DD
        SS  UU      UU  RR  RR      FF                  33  DD      DD
        SS  UU      UU  RR  RR      FF                  33  DD      DD
        SS  UU      UU  RR    RR    FF          33      33  DD      DD
        SS  UU      UU  RR    RR    FF          33      33  DD      DD
SSSSSSSS    UUUUUUUUUU  RR      RR  FF            333333    DDDDDDDD
SSSSSSSS    UUUUUUUUUU  RR      RR  FF            333333    DDDDDDDD

                        FFFFFFFFFF      ;;;;        11
                        FFFFFFFFFF      ;;;;        11
                        FF              ;;;;      1111
                        FF              ;;;;      1111
                        FF                          11
                        FF                          11
                        FFFFFFFF        ;;;;        11
                        FFFFFFFF        ;;;;        11
                        FF              ;;;;        11
                        FF              ;;;;        11
                  ....  FF                ;;        11
                  ....  FF                ;;        11
                  ....  FF              ;;        111111
                  ....  FF              ;;        111111
```

File 1DUS121:[GUIMIN]SURF3D.F;1 (1612,12,0), last revised on 8-DEC-1988 11:03, is a 75 block sequential file owned by UIC [GUIMIN]. The records are variable length with implied (CR) carriage control. The longest record is 80 bytes.

Job SURF3D (1197) queued to SYS$PRINT on 8-DEC-1988 11:18 by user GUIMIN, UIC [GUIMIN], under account 438901 at priority 100, started on printer LCA0 on 8-DEC-1988 11:18 from queue CSA2$LCA0.


```
SSSSSSSSSS    0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000    SSSSSSSSSS
SSSSSSSSSS    000000000000000000000000  Digital Equipment Corporation - VAX/VMS Version V4.7  000000000000000000000000    SSSSSSSSSS
SSSSSSSSSS    0000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000000    SSSSSSSSSS
```

```fortran
      PROGRAM SURF3D
      implicit real*8 (a-h, o-z)
c    ****************************************************************
c    *  Purpose...
c    *    to compute the secondary electromagnetic response due to a mag-
c    *    netic dipole which sits above a perfectly conductive medium.
c    *    The secondary field is expressed in parts per million of the
c    *    received primary field. The orientation of the dipole is arbi-
c    *    trary and the surface of the conductor is arbitrary. When ipattn
c    *    = 1, the program outputs the total surface current density.
c    *    Program not optimized and can be further improved.
c    *    Program not documented.
c    *    Gaussian quadrature is used for the integration of the kernel.
c    *    For fast speed, ignore the Gaussian quadrature by setting 0 in
c    *    the IF staement.
c    *  Author ... Guimin Liu, Engineering Geoscience, U.C.Berkeley.
c    *            (415)642-3809      .    1988
c    ****************************************************************
      DIMENSION ZH(121,121),XDER(121,121),YDER(121,121),HN(121,121),
     + W2(121,121)
      DIMENSION AMPDER(121,121)
      COMMON /AA/ZH/BB/XDER/CC/YDER/DD/HN/EE/W/AMP/AMPDER
      COMMON /TR/XS,YS,ZS,XR,YR,ZR
      COMMON  NX,NY,NX2,NY2,XDELTA,YDELTA
      open (unit=3,file='surf3d.dat')
      open (unit=4, file ='surf3d.out')
C
C    WHEN NPLOT=1 , THE OUTPUT IS ARRANGED FOR PLOTTING PURPOSE
      READ(3,*) NPLOT,ipattn
C    ANGV = VERTICAL ANGLE. ANGH = HORIZONTAL ANGLE FROM X DIRECTION
C    INORM = NORMALIZATION COMPONENT. 1 = HX, 2 = HY, 3 = HZ
      READ(3,*) ANGV, ANGH, INORM
C    READ IN NUMBER OF SOURCES
      READ(3,*)NSOR
C    READ IN GRID NUMBER , MAXIMUN NUMBER OF ITERATIONS
      READ(3,*)NX,NY,NLOOP
C    READ IN SURFACE PARAMETERS
C    GAUSSIAN DISTRIBUTION SHAPE
      READ(3,*)HEIGHT,XWIDTH,YWIDTH
C    READ IN GRID SIZE
      READ(3,*)XDELTA,YDELTA
      NS=0
      ANGV=ANGV/180*3.14159
      ANGH=ANGH/180*3.14159
      IF(INORM.EQ.1) WRITE(4,41)
      IF(INORM.EQ.3) WRITE(4,42)
 41   FORMAT(4X,'HORIZONTAL MAGNETIC DIPOLE SOURCE')
 42   FORMAT(4X,'VERTICAL MAGNETIC DIPOLE SOURCE')
      IF(NPLOT.EQ.1) GOTO 501
      WRITE(4,112) ANGV,ANGH,INORM
      WRITE(4,111)HEIGHT,XWIDTH,YWIDTH
      WRITE(4,200)NX,NY,XDELTA,YDELTA,NLOOP
 112  FORMAT(7X,'ANGLE OF THE DIPOLE FROM VERTICAL =', F6.2,/
     +  ,7X,'ANGLE OF THE DIPOLE FROM X DIRECTION =', F6.2,/,
     +  7X,'NORMALIZATION COMPONENT 1 = HX    2 = HY    3 = HZ ',I5,/)
 200  FORMAT(7X,'GRID             NX = ',I5,3X,' NY = ',I5 ,
     +    /,7X,'ELEMENT SIZE   DX = ', F10.5,'  DY =',F10.5,
     +    /,7X,'MAXIMUM ITERATION NUMBER', I8  ,//)
      GOTO 666
 501  CONTINUE
```

```
C     WRITE(4,113) ANGV,ANGH,INORM
      WRITE(4,101)HEIGHT,XWIDTH,YWIDTH
      WRITE(4,201)NX,NY,XDELTA,YDELTA
 101  FORMAT(5X,'HE =',F6.2,4X,'XW =',F6.2,4X,'YW =',F6.2)
 113  FORMAT(5X,2F18.2,I8)
 201  FORMAT(5X,2I8,5X,2F18.2,//)
 666  CONTINUE
C   READ IN NUMBER OF RECEIVERS
      READ(3,*)NREC
C   READ IN SOURCE POSITION
      READ(3,*)XS,YS,ZS
      NR=0
 111  FORMAT(7X,'SURFACE PARAETER    HEIGHT = ',F7.1 ,1X,' XWIDTH =',1X,
     +  F7.1 ,'YWIDTH =',F7.1 )
C
C   GENERATING THE SURFACE OF THE WATER
C
      CALL SURF(HEIGHT,XWIDTH,YWIDTH)


C
C
C   CALCULATE THE PRIMARY FIELD AT THE SURFACE. GENERATE THE BOUNDARY CO
C   FOR NEUMANN PROBLEM.
C
      CALL HNPRIM ( ANGV,ANGH )
C
C   CALCULATE THE SURFACE CHARGE DENSITY BY SOLVING THE FREDHOLM INTEGRA
C   EQUATION OF THE SECOND KIND.
C
      CALL SIGMA(NPLOT,NLOOP)
c
c    compute the surface current pattern and then stop
c
      if(ipattn.eq.1) call currnt(angv,anvh)
C
C   CALCULATE THE MAGNETIC FIELD AT A RECEIVER POINT.
C
 555  CONTINUE
C  READ IN RECEIVER POSITION
      READ(3,*)XR ,YR ,ZR
      CALL FIELD(NPLOT,ANGV,ANGH,INORM)
      NR=NR+1
      IF(NR.LT.NREC) GOTO 555
      NS=N +1
      IF(NS.LT.NSOR) GOTO 666
      STOP
      END
C
C
      SUBROUTINE SURF(Z0,XW,YW)
      implicit real*8 (a-h, o-z)
      DIMENSION ZH(121,121),XDER(121,121),YDER(121,121),AMPDER(121,121)
      COMMON /AA/ZH/BB/XDER/CC/YDER/AMP/AMPDER
      COMMON /TR/XS,YS,ZS,XR,YR,ZR
      COMMON  NX,NY,NX2,NY2,XDELTA,YDELTA
C
      NX2=(NX+1)/2
      NY2=(NY+1)/2
      TAUX=0.42466*XW
```

```
      TAUY=0.42466*YW
      XDEV=TAUX*TAUX*2.0
      YDEV=TAUY*TAUY*2.0
      DO 10 I=1,NX
      DO 20 J=1,NY
       X=(I-NX2)*XDELTA+XS
       Y=(J-NY2)*YDELTA+YS
       EX=-(X*X/XDEV+Y*Y/YDEV)
       IF(EX.LT.(-10)) GOTO 333
       RR=EXP(EX)
       GOTO 444
 333   RR=0.0
 444   ZH(I,J)=Z0*RR
       XDER(I,J)=-ZH(I,J)*X    /XDEV*2.0
       YDER(I,J)=-ZH(I,J)*Y    /YDEV*2.0
       AMPDER(I,J)=SQRT(1.0+XDER(I,J)*XDER(I,J)+YDER(I,J)*YDER(I,J))
 20   CONTINUE
 10   CONTINUE
C     WRITE(4,100)Z0,WIDTH,XDELTA,YDELTA,RR
C     WRITE(4,100)((ZH(I,J),J=1,NY),I=1,NX)
C     WRITE(4,100)((XDER(I,J),J=1,NY),I=1,NX)
C     WRITE(4,100)((YDER(I,J),J=1,NY),I=1,NX)
 100   FORMAT(3X,//,(5E12.5))
      RETURN
      END
C
C
      SUBROUTINE HNPRIM (ANGV,ANGH)
      implicit real*8 (a-h, o-z)
      DIMENSION HN(121,121),ZH(121,121),XDER(121,121),YDER(121,121)
      DIMENSION AMPDER(121,121)
      COMMON /AA/ZH/BB/XDER/CC/YDER/DD/HN/AMP/AMPDER
      COMMON  NX,NY,NX2,NY2,XDELTA,YDELTA
      COMMON /TR/XS,YS,ZS,XR,YR,ZR
C
       COSV=COS(ANGV)
       SINVCS=SIN(ANGV)*COS(ANGH)
       SINVSN=SIN(ANGV)*SIN(ANGH)
      DO 10 I=1,NX
      DO 20 J=1,NY
       XD=(I-NX2)*XDELTA
       YD=(J-NY2)*YDELTA
       ZD=ZH(I,J)-ZS
       XD2=XD*XD
       YD2=YD*YD
       ZD2=ZD*ZD
       XYD=XD*YD
       YZD=YD*ZD
       ZXD=ZD*XD
       R5=(XD2+YD2+ZD2)**2.5
       HX1=3.0*ZXD
       HY1=3.0*YZD
       HZ1=(2*ZD2  -XD2  -YD2  )
       HX2=(2*XD2 - YD2 - ZD2 )
       HY2=3.0*XYD
       HZ2=3.0*ZXD
       HX3=3.0*XYD
       HY3=(2*YD2 - XD2 - ZD2 )
       HZ3=3.0*YZD
       HX=(HX1*COSV+HX2*SINVCS+HX3*SINVSN)/R5
```

```
      HY=(HY1*COSV+HY2*SINVCS+HY3*SINVSN)/R5                            SUR01640
      HZ=(HZ1*COSV+HZ2*SINVCS+HZ3*SINVSN)/R5                            SUR01650
      HN(I,J)=(HX*XDER(I,J)+HY*YDER(I,J)-HZ)/AMPDER(I,J)                SUR01660
 20   CONTINUE                                                          SUR01670
 10   CONTINUE                                                          SUR01680
C     WRITE(4,100)((HN(I,J),J=1,NY),I=1,NX)                             SUR01690
 100   FORMAT(3X,///,(5E12.5))                                          SUR01700
      RETURN                                                            SUR01710
      END                                                               SUR01720
C                                                                       SUR01730
C                                                                       SUR01740
                                                                        SUR01750
      SUBROUTINE SIGMA (NPLOT,NLOOP)                                    SUR01760
      implicit real*8 (a-h, o-z)
      DIMENSION   W1(121,121),ZH(121,121),XDER(121,121),YDER(121,121)   SUR01770
      DIMENSION   W2(121,121),AMPDER(121,121),HN(121,121)               SUR01780
      DIMENSION   XD(200),YD(200),XD2(200),YD2(200)                     SUR01790
      COMMON /AA/ZH/BB/XDER/CC/YDER/DD/HN/EE/W2/AMP/AMPDER              SUR01800
      COMMON /TR/XS,YS,ZS,XR,YR,ZR                                      SUR01810
      COMMON  NX,NY,NX2,NY2,XDELTA,YDELTA                               SUR01820
      common /in/a,b,c,xm,ym,y1,y2,hdx,hdy
C                                                                       SUR01830
      DXY=XDELTA*YDELTA                                                 SUR01840
      FAC=1.0/(2*3.1415926)                                             SUR01850
      DO 10 I=1,NX                                                      SUR01860
      DO 10 J=1,NY                                                      SUR01870
       W2(I,J)=0.0                                                      SUR01880
 10    W1(I,J)=-HN(I,J)*FAC                                             SUR01890
      NUM=0                                                             SUR01910
      N2X=NX*2                                                          SUR01920
      N2Y=NY*2                                                          SUR01930
      DO 13 I=1,N2X                                                     SUR01940
      XD(I)=(I-NX)*XDELTA                                               SUR01950
 13   XD2(I)=XD(I)*XD(I)                                                SUR01960
      DO 14 I=1,N2Y                                                     SUR01970
      YD(I)=(I-NY)*YDELTA                                               SUR01980
 14   YD2(I)=YD(I)*YD(I)                                                SUR01990
 999   CONTINUE                                                         SUR02000
      DO 15 I=1,NX                                                      SUR02010
      DO 20 J=1,NY                                                      SUR02020
        WS=0.0D0                                                        SUR02030
      DO 40 I1=1,NX                                                     SUR02040
      DO 30 J1=1,NY                                                     SUR02050
       K1= I1-I+NX                                                      SUR02060
       K2= J1-J+NY                                                      SUR02070
       ZD=ZH(I1,J1)-ZH(I,J)                                             SUR02080
       IF(I.EQ.I1 .AND. J.EQ.J1) GOTO 30                                SUR02100
       IF(ABS(I1-I).LE.4.AND. ABS(J1-J).LE.4) THEN
         XM=(I-NX2)*XDELTA
         YM=(J-NY2)*YDELTA
         X1=(I1-NX2-0.5)*XDELTA
         X2=(I1-NX2+0.5)*XDELTA
         Y1=(J1-NY2-0.5)*YDELTA
         Y2=(J1-NY2+0.5)*YDELTA
         HDX=XDER(I,J)
         HDY=YDER(I,J)
          a=-XDER(I1,J1)
          b=-YDER(I1,J1)
          C=-ZD-a*(I1-NX2)*XDELTA-b*(J1-NY2)*YDELTA
          CALL QGAUS(X1,X2,SS4)
```

```
          WS   =WS+SS4*AMPDER(I1,J1)/AMPDER(I,J)*W1(I1,J1)
      else
         RPM2=XD2(K1)+YD2(K2)+ZD*ZD                                          SUR02090
         BT=(ZD-XDER(I,J)*XD(K1)-YDER(I,J)*YD(K2))*dxy                       SUR02110
         WS=WS+W1(I1,J1)*BT*AMPDER(I1,J1)/(RPM2*SQRT(RPM2)*AMPDER(I,J))      SUR02120
      endif
 30   CONTINUE                                                               SUR02130
 40   CONTINUE                                                               SUR02140
      W2(I,J)=FAC*(-WS-HN(I,J))                                              SUR02150
 20   CONTINUE                                                               SUR02160
 15   CONTINUE                                                               SUR02170
      SUM=0.0                                                                SUR02180
      IF(NPLOT.EQ.1) GOTO 501                                                SUR02190
      DO 50 I=1,NX                                                           SUR02200
      DO 50 J=1,NY                                                           SUR02210
       WW=W2(I,J)-W1(I,J)                                                    SUR02220
 50   SUM=SUM+WW*WW                                                          SUR02230
      AVER=SQRT(SUM)/(NX*NY)                                                 SUR02240
      WRITE(4,100) AVER                                                      SUR02250
 100  FORMAT(20X, 'AVERAGE DIFFERENCE BETWEEN TWO SUCCESIVE ITERATIONS'      SUR02260
     + ,/,E10.5)                                                             SUR02270
      NUM=NUM+1                                                              SUR02280
      TT=ABS(AVER/W2(NX2,NY2))                                               SUR02290
      WRITE(4,122) NUM                                                       SUR02300
 122  FORMAT(///,20X,'NUM= ',I5)                                             SUR02310
      IF(    TT          .LT.0.1E-4  .OR.NUM.GT.NLOOP) GOTO 1000             SUR02340
      GOTO 502                                                               SUR02350
 501  NUM=NUM+1                                                              SUR02360
      IF ( NUM.GT.NLOOP) GOTO 1000                                           SUR02370
 502  CONTINUE                                                               SUR02380
      DO 44 I=1,NX                                                           SUR02390
      DO 44 J=1,NY                                                           SUR02400
       W1(I,J)=W2(I,J)                                                       SUR02410
 44    W2(I,J)=0.0                                                           SUR02420
       GOTO 999                                                              SUR02430
 1000 CONTINUE                                                               SUR02440
C     DO 111 I=1,NX                                                          SUR02320
C111  WRITE(4,110)(W2(I,J),J=1,NY )                                          SUR02330
C110  FORMAT(1X, //,(5E12.5))                                                SUR02450
      RETURN                                                                 SUR02460
      END                                                                    SUR02470
C                                                                            SUR02700
C                                                                            SUR02710
      SUBROUTINE FIELD(NPLOT,ANGV,ANGH,INORM)                                SUR02390
      implicit real*8 (a-h, o-z)
      DIMENSION W(121,121),ZH(121,121),XDER(121,121),YDER(121,121)           SUR02500
      DIMENSION AMPDER(121,121)                                              SUR02510
      COMMON /AA/ZH/BB/XDER/CC/YDER/AMP/AMPDER                               SUR02520
     + /EE/W                                                                 SUR02530
      COMMON  NX,NY,NX2,NY2,XDELTA,YDELTA                                    SUR02540
      COMMON /TR/XS,YS,ZS,XR,YR,ZR                                           SUR02760
C                                                                            SUR02770
      COSV=COS(ANGV)                                                         SUR02780
      SINVCS=SIN(ANGV)*COS(ANGH)                                             SUR02790
      SINVSN=SIN(ANGV)*SIN(ANGH)                                             SUR02800
      XD=XR-XS                                                               SUR02810
      YD=YR-YS                                                               SUR02820
      ZD=ZR-ZS                                                               SUR02830
       XD2=XD*XD                                                             SUR02840
       YD2=YD*YD                                                             SUR02850
```

```
      ZD2=ZD*ZD                                                         SUR02860
      XYD=XD*YD                                                         SUR02870
      YZD=YD*ZD                                                         SUR02880
      ZXD=ZD*XD                                                         SUR02890
      R5=(XD2+YD2+ZD2)**2.5                                             SUR02900
      HX1=3.0*ZXD                                                       SUR02910
      HY1=3.0*YZD                                                       SUR02920
      HZ1=(2*ZD2 - XD2 - YD2 )                                          SUR02930
      HX2=(2*XD2 - YD2 - ZD2 )                                          SUR02940
      HY2=3.0*XYD                                                       SUR02950
      HZ2=3.0*ZXD                                                       SUR02960
      HX3=3.0*XYD                                                       SUR02970
      HY3=(2*YD2 - XD2 - ZD2 )                                          SUR02980
      HZ3=3.0*YZD                                                       SUR02990
      HXP=(HX1*COSV+HX2*SINVCS+HX3*SINVSN)/R5                           SUR03000
      HYP=(HY1*COSV+HY2*SINVCS+HY3*SINVSN)/R5                           SUR03010
      HZP=(HZ1*COSV+HZ2*SINVCS+HZ3*SINVSN)/R5                           SUR03020
     HXS=0.0                                                            SUR02700
     HYS=0.0                                                            SUR02710
     HZS=0.0                                                            SUR02720
     DO 10 I=1,NX                                                       SUR02730
     DO 20 J=1,NY                                                       SUR02740
      XD=(I-NX2)*XDELTA+XS-XR                                           SUR02750
      YD=(J-NY2)*YDELTA+YS-YR                                           SUR02760
      ZD=ZH(I,J)    - ZR                                                SUR02770
      RPM2=XD*XD+YD*YD+ZD*ZD                                            SUR02780
      RPM=SQRT(RPM2)                                                    SUR02790
      HXS  =HXS  +W(I,J)*XD*AMPDER(I,J)/(RPM*RPM2)                      SUR02800
      HYS  =HYS  +W(I,J)*YD*AMPDER(I,J)/(RPM*RPM2)                      SUR02810
      HZS  =HZS  +W(I,J)*ZD*AMPDER(I,J)/(RPM*RPM2)                      SUR02820
20   CONTINUE                                                           SUR02830
10   CONTINUE                                                           SUR02840
     HXS  =-HXS *XDELTA*YDELTA                                          SUR02850
     HYS  =-HYS *XDELTA*YDELTA                                          SUR02860
     HZS  =-HZS *XDELTA*YDELTA                                          SUR02870
     IF(INORM.EQ.1) GOTO 606                                            SUR03060
     IF(INORM.EQ.2) GOTO 607                                            SUR03070
     HX=HXS/HZP*1000000                                                 SUR03080
     HY=HYS/HZP*1000000                                                 SUR03090
     HZ=HZS/HZP*1000000                                                 SUR03100
     GOTO 608                                                           SUR03110
606  HX=HXS/HXP*1000000                                                 SUR03120
     HY=HYS/HXP*1000000                                                 SUR03130
     HZ=HZS/HXP*1000000                                                 SUR03140
     GOTO 608                                                           SUR03150
607  HX=HXS/H  *1000000                                                 SUR03160
     HY=HYS/HYP*1000000                                                 SUR03170
     HZ=HZS/HYP*1000000                                                 SUR03180
608  CONTINUE                                                           SUR03190
     IF(NPLOT.EQ.1) GOTO 501                                            SUR03200
     WRITE(4,200)    XS,YS,ZS                                           SUR03210
200  FORMAT(                                                            SUR03220
    +   /, 7X,'SOURCE POSITION    ( ',F10.5,1X,F10.5,1X,F10.5,')')      SUR03230
     WRITE(4,100)XR,YR,ZR,HX,HY,HZ                                      SUR03240
100  FORMAT( 7X,'RECEIVER POSITION ( ',F10.5,1X,F10.5,1X,F10.5, ')',/,  SUR03250
    +  7X,'SECONDARY MAGNETIC FIELD IN PPM',/,7X,'STATIC FIELD          SUR03260
    +  APPROXIMATION',/, 5X,                                            SUR03270
    +     'HX = ', 5.8, ' HY = ',E15.8, ' HZ = ',E15.8)                 SUR03280
     GOTO 502                                                           SUR03290
501  WRITE(4,101)    XS,YS,ZS                                           SUR03300
```

```
      WRITE(4,201)XR,YR,ZR,HX,HY,HZ                                      SUR03310
 101  FORMAT(10X,3F18.2)                                                 SUR03320
 201  FORMAT(10X,3F18.2,/,10X,3E18.5,//)                                 SUR03330
 502  RETURN                                                             SUR03340
      END                                                                SUR03350
c
c
c   the following routine compute the surface currents.
      SUBROUTINE currnt(angv,angh)
      implicit real*8 (a-h, c-z)
      DIMENSION   ZH(121,121),XDER(121,121),YDER(121,121)                SUR01770
      DIMENSION   W2(121,121),AMPDER(121,121)                            SUR01780
      dimension   curamp(121,121),curdir(121,121)
      DIMENSION   XD(200),YD(200),XD2(200),YD2(200)                      SUR01790
      COMMON /AA/ZH/BB/XDER/CC/YDER/EE/W2/AMP/AMPDER                     SUR01800
      COMMON /TR/XS,YS,ZS,XR,YR,ZR                                       SUR01810
      COMMON  NX,NY,NX2,NY2,XDELTA,YDELTA                                SUR01820
      common /in/a,b,c,xm,ym,y1,y2,hdx,hdy
C
      N2X=NX*2                                                           SUR01830
      N2Y=NY*2                                                           SUR01920
      DXY=XDELTA*YDELTA                                                  SUR01930
      DO 13 I=1,N2X                                                      SUR01940
      XD(I)=(I-NX)*XDELTA                                                SUR01950
 13   XD2(I)=XD(I)*XD(I)                                                 SUR01960
      DO 14 I=1,N2Y                                                      SUR01970
      YD(I)=(I-NY)*YDELTA                                                SUR01980
 14   YD2(I)=YD(I)*YD(I)                                                 SUR01990
      DO 15 I=1,NX                                                       SUR02010
      DO 20 J=1,NY                                                       SUR02020
        curx=0.0
        cury=0.0
        curz=0.0
      DO 40 I1=1,NX                                                      SUR02040
      DO 30 J1=1,NY                                                      SUR02050
       K1= I1-I+NX                                                       SUR02060
       K2= J1-J+NY                                                       SUR02070
       ZD=ZH(I1,J1)-ZH(I,J)                                              SUR02080
       IF(I.EQ.I1 .AND. J.EQ.J1) GOTO 30                                 SUR02100
       IF(ABS(I1-I).LE.4.AND. ABS(J1-J).LE.4) THEN
         XM=(I-NX2)*XDELTA
         YM=(J-NY2)*YDELTA
         X1=(I1-NX2-0.5)*XDELTA
         X2=(I1-NX2+0.5)*XDELTA
         Y1=(J1-NY2-0.5)*YDELTA
         Y2=(J1-NY2+0.5)*YDELTA
         HDX=XDER(I,J)
         HDY=YDER(I,J)
          a=-XDER(I1,J1)
          b=-YDER(I1,J1)
          C=-ZD-A*(I1-NX2)*XDEL    B*(J1-NY2)*YDEL
          CALL QGAUS2(X1,X2,SS1,SS2,SS3)
          WS  =W2(I1,J1)*AMPDER(I1,j1)/ampder(i,j)
          CURX=WS*SS1+CURX
          CURY=WS*SS2+CURY
          CURZ=WS*SS3+CURZ
      GOTO 30
      ENDIF
       RPM2=XD2(K1)+YD2(K2)+ZD*ZD                                        SUR02090
       IF(RPM2.LT.0.1 ) GOTO 30                                          SUR02100
```

```
      BTX=-ZD*YDER(I,J)-YD(K2)                                          SUR02110
      BTY= XD(K1)+XDER(I,J)*ZD                                          SUR02110
      BTZ=-XDER(I,J)*YD(K2)+XD(K1)*yder(i,j)                            SUR02110
      WS=W2(I1,J1)*AMPDER(I1,J1)/(RPM2*SQRT(RPM2)*AMPDER(I,J))*DXY
      curx=curx+WS*BTX
      cury=cury+WS*BTY
      curz=curz+WS*BTZ
 30   CONTINUE                                                          SUR02130
 40   CONTINUE                                                          SUR02140
c
c   primary field calculation
      COSV=COS(ANGV)                                                    SUR01390
      SINVCS=SIN(ANGV)*COS(ANGH)                                        SUR01400
      SINVSN=SIN(ANGV)*SIN(ANGH)                                        SUR01410
      X1=(I-NX2)*XDELTA                                                 SUR01440
      Y1=(J-NY2)*YDE  A                                                 SUR01450
      Z1=ZH(I,J)-ZS                                                     SUR01460
      X2=X1*X1                                                          SUR01470
      Y2=Y1*Y1                                                          SUR01480
      Z2=Z1*Z1                                                          SUR01490
      XYD=X1*Y1                                                         SUR01500
      YZD=Y1*Z1                                                         SUR01510
      ZXD=Z1*X1                                                         SUR01520
      R5=(X2+Y2+Z2)**2.5                                                SUR01530
      HX1=3.0*ZXD                                                       SUR01540
      HY1=3.0*YZD                                                       SUR01550
      HZ1=(2*Z2-X2-Y2)                                                  SUR01560
      HX2=(2*X2 - Y2 - Z2 )                                             SUR01570
      HY2=3.0*XYD                                                       SUR01580
      HZ2=3.0*ZXD                                                       SUR01590
      HX3=3.0*XYD                                                       SUR01600
      HY3=(2*Y2 - X2 - Z2 )                                             SUR01610
      HZ3=3.0*YZD                                                       SUR01620
      HX=(HX1*COSV+HX2*SINVCS+HX3*SINVSN)/R5                            SUR01630
      HY=(HY1*COSV+HY2*SINVCS+HY3*SINVSN)/R5                            SUR01640
      HZ=(HZ1*COSV+HZ2*SINVCS+HZ3*SINVSN)/R5                            SUR01650
      CURX=CURX+(HZ*YDER(I,J)+HY)/AMPDER(I,J)
      CURY=CURY+(-HX-HZ*XDER(I,J))/AMPDER(I,J)
      CURZ=CURZ+(HY*XDER(I,J)-HX*YDER(I,J))/AMPDER(I,J)
      CURAMP(I,J)=SQRT(CURX**2+CURY**2+CURZ**2)
      IF(ABS(CURX) LT.1.0E-30 .AND. CURY.GE.0.0) THEN
          curdir(i,j)=90.
      else
         if(abs(curx).lt.1.0e-30 .and. cury.lt.0.0) then
           curdir(i,j)=-90.
         else
           curdir(i,j)=atan2(cury,curx)*180./3.1415926
         endif
      endif
 20   CONTINUE                                                          SUR02160
 15   CONTINUE                                                          SUR02170
      DO 100 I=1,NX
        write(4,101)i,(curamp(i,j),j=1,ny)
 100  continue
      DO 110 I=1,NX
        write(4,102)i,(curdir(i,j),j=1,ny)
 110  continue
 101  FORMAT(4X,'AMPLITUDE OF SURFACE CURRENT i =',i5 ,/,(5(1PE12.3)))  )))
 102  FORMAT(4X,'DIRECTION OF SURFACE CURRENT i =',i5 ,/,(5(1PE12.3)))  )))
      stop
```

```
      END
C
C
      subroutine qgaus2(a,b,ss1,ss2,ss3)
      implicit real*8 (a-h, o-z)
c  nine -point Gauss' quadrature
      dimension x(5),w(5)
      data x/0.0,0.324253,0.613371,0.836031,0.968160/
      data w/0.330239,0.312347,0.260611,0.180648,0.081274/
      xm=0.5*(b+a)
      xr=0.5*(b-a)
      ss1=0.
      ss2=0.
      ss3=0.
      do 11 j=2,5
       dx=xr*x(j)
       ss1=ss1+w(j)*fun1(xm+dx,fun2,fun3)
       ss2=ss2+w(j)*fun2
       ss3=ss3+w(j)*fun3
       ss1=ss1+w(j)*fun1(xm-dx,fun2,fun3)
       ss2=ss2+w(j)*fun2
       ss3=ss3+w(j)*fun3
 11   continue
       ss1=xr*(ss1+w(1)*fun1(xm,fun2,fun3))
       ss2=xr*(ss2+w(1)*fun2)
       ss3=xr*(ss3+w(1)*fun3)
      return
      END
C
C
      function fun1(xp,fun2,fun3)
c  to compute the kernel for the Gaussian quadrature
      implicit real*8 (a-h, o-z)
      common /in/a,b,c,xm,ym,y1,y2,hdx,hdy
      d=a*xp+c
      e=(xm-xp)**2+d*d+ym*ym
      f=2.*(d*b-ym)
      g=1.+b*b
      del=4*e*g-f*f
      IF(ABS(DEL) .LT.1.E-20'    EN
        SG=1./(G*SQRT(G))
        FG=F/(G+G)
        YG1=1./(Y1+FG)
        YG2=1./(Y2+FG)
        AIX1=abs(0.5*SG*(YG2*YG2-YG1*YG1))
        AIX2=SG*(YG2*(-1.+0.5*FG*YG2)-YG1*(-1.+0.5*FG*YG1))
        IF(YG1.LT.0.0) AIX2=    -AIX2
      ELSE
        r2=1./(del*sqrt(e+(f+g*y2)*y2))
        r1=1./(del*sqrt(e+(f+g*y1)*y1))
        aix1=2.*((2.*g*y2+f)*r2-(2.*g*y1+f)*r1)
        aix2=-2.*((2.*e+f*y2)*r2-(2.*e+f*y1)*r1)
      ENDIF
      IF(AIX1.LT.0.0  ) THEN
        WRITE(*,*) 'XM,YM,Y1,XP,AIX1,AIX2',XM,YM,Y1,XP,AIX1,AIX2
        stop
      endif
      fun1=(d*hdy+ym)*aix1+(b*hdy-1.)*aix2
      fun2=(-(xm-xp)-d*hdx)*aix1-b*hdx*aix2
      fun3=(ym*hdx-(xm-xp)*hdy)*aix1-hdx*aix2
```

```
      return
      end
c
c
      subroutine qgaus(a,b,ss4)
      implicit real*8 (a-h, o-z)
c  nine -point Gauss' quadrature
      dimension x(5),w(5)
      data x/0.0,0.324253,0.613371,0.836031,0.968160/
      data w/0.330239,0.312347,0.260611,0.180648,0.081274/
      xm=0.5*(b+a)
      xr=0.5*(b-a)
      ss4=0.
      do 11 j=2,5
       dx=xr*x(j)
       ss4=ss4+w(j)*(fun4(xm+dx)+fun4(xm-dx))
 11   continue
       ss4=xr*(ss4+w(1)*fun4(xm))
      return
      END
c
c
      function fun4(xp)
      implicit real*8 (a-h, o-z)
      common /in/a,b,c,xm,ym,y1,y2,hdx,hdy
      d=a*xp+c
      e=(xm-xp)**2+d*d+ym*ym
      f=2.*(d*b-ym)
      g=1.+b*b
      del=4*e*g-f*f
      if(abs(del) .lt.1.e-20 ) then
        sg=1./(g*sqrt(g))
        fg=f/(g+g)
        yg1=1./(y1+fg)
        yg2=1./(y2+fg)
        aix1=abs(0.5*sg*(yg2*yg2-yg1*yg1))
        aix2=sg*(yg2*(-1.+0.5*fg*yg2)-yg1*(-1.+0.5*fg*yg1))
        if(yg1.lt.0) aix2=-aix2
      else
        r2=1./(del*sqrt(e+(f+g*y2)*y2))
        r1=1./(de.*sqrt(e+(f+g*y1)*y1))
        aix1=2.*((2.*g*y2+f)*r2-(2.*g*y1+f)*r1)
        aix2=-2.*((2.*e+f*y2)*r2-(2.*e+f*y1)*r1)
      endif
      fun4=((xm-xp)*hdx+ym*hdy-d)*aix1-(hdy+b)*aix2
      return
      end
```

```
      PROGRAM ICE2D1                                                    ICE00010
      implicit real*8 (a-h,o-z)
C    ...PURPOSE...                                                      ICE00020
C      COMPUTE THE AEM SYSTEM RESPONSE OVER A 2-D ICE-WATER PROFILE     ICE00030
      DIMENSION ZSURF(200),XDER(200),AMPDER(200),HPN(120,20)            ICE00040
      DIMENSION AKERN(120,120,20),CHAR(120,20),XSYS(50),ZSYS(50)        ICE00050
      COMMON /KY/AKY(20)/KER/AKERN/CH/CHAR/HN/HPN                       ICE00060
      COMMON KYE,NSYS,NST,NPT,NXT,NTRUNK,XDEL,TRDIS                     ICE00070
      open (unit=3, file='ice2d1.dat')
      open (unit=4, file='ice2d1.out')
      READ(3,*)                                                         ICE00090
      READ(3,*)NSYS,NST,NPT,NXT,NRESP,NTRUNK,NLOOP                      ICE00100
      READ(3,*)                                                         ICE00110
      READ(3,*)XDEL,TRDIS                                               ICE00120
      READ(3,*)                                                         ICE00130
      READ(3,*)(ZSURF(I),I=1,NPT)                                       ICE00140
      READ(3,*)                                                         ICE00150
      READ(3,*)(XSYS(I),ZSYS(I),I=1,NRESP)                              ICE00160
      IF(NSYS.EQ.1) WRITE(4,101)                                        ICE00170
      IF(NSYS.EQ.2) WRITE(4,102)                                        ICE00180
      IF(NSYS.NE.1 .AND. NSYS.NE.2) PAUSE 'BAD INPUT NSYS'              ICE00190
      WRITE(4,104)TRDIS,XDEL                                            ICE00200
      WRITE(4,103)NST,NPT,NXT,NRESP,NTRUNK,NLOOP                        ICE00210
      WRITE(4,105)(ZSURF(I),I=1,NPT)                                    ICE00220
C     WRITE(4,106)(XSYS(I),I=1,NRESP)                                   ICE00230
C     WRITE(4,107)(ZSYS(I),I=1,NRESP)                                   ICE00240
      CALL SURF(NST,NPT,NXT,XDEL,ZSURF,XDER,AMPDER)                     ICE00250
      K=1                                                               ICE00260
C ..................................................................... ICE00270
C     ASSIGN  KY HOMONICS. THE FOLLOWING IS OK FOR ZR=10 - 100 METERS.  ICE00280
      KYE= 15                                                           ICE00290
      NM=7                                                              ICE00300
      AKYMIN=1.0/(2.0*1000)                                             ICE00310
      AKDEL=0.0006                                                      ICE00320
      AKLOG=0.17                                                        ICE00330
      AKYEX=LOG10(AKYMIN)                                               ICE00340
      AKY(1)=0.0                                                        ICE00350
      DO 20 I=1,KYE-1                                                   ICE00360
        EX=AKYEX+(I-1)*AKLOG                                            ICE00370
        AKY(I+1)=10**EX                                                 ICE00380
 20   CONTINUE                                                          ICE00390
      WRITE(4,201)(AKY(I),I=1,KYE)                                      ICE00400
C ..................................................................... ICE00410
      WRITE(4,202)                                                      ICE00420
 1001 CONTINUE                                                          ICE00430
      KS=(XSYS(K)-TRDIS/2.)/XDEL+2-NTRUNK/2                             ICE00440
      IF(KS.LE.0 .OR. KS.GE.NXT-NTRUNK/2) PAUSE'BAD INPUT XSYS'         ICE00450
      CALL HNPRIM(K,KS,ZSURF,XDER,AMPDER,XSYS,ZSYS)                     ICE00460
      CALL KERNEL(K,KS-1,KS1,ZSURF,XDER,AMPDER)                         ICE00470
      CALL CHARGE(NTRUNK,KYE,XDEL,NLOOP)                                ICE00480
      CALL FIELD(K,KS,NM,AKDEL,HXS,HZS,XSYS,ZSYS,ZSURF,AMPDER)          ICE00490
      WRITE(4,203) XSYS(K),ZSYS(K),HXS,HZS                              ICE00500
      K=K+1                                                             ICE00510
      KS1=KS                                                            ICE00520
      IF(K.LE.NRESP) GOTO 1001                                          ICE00530
 101  FORMAT(4X,'HORIZONTAL AXIS DIPOLE SYSTEM',/)                      ICE00540
 102  FORMAT(4X,'VERTICAL   AXIS DIPOLE SYSTEM',/)                      ICE00550
 103  FORMAT(4X,'NST =',I5,2X,'NPT =',I5,2X,'NXT =',I5,/,               ICE00560
     + 4X,'NRESP =',I5,2X,'NTRUNK =',I5,2X,'NLOOP =',I5)                ICE00570
 104  FORMAT(4X, 'T-R SEPARATION =',F7.2,' METERS',/,4X,'XDEL =',F7.2,  ICE00580
```

```
RRRRRRRRRR   9999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999   RRRRRRRRRR
RRRRRRRRRR   99999999999999999999999  Digital Equipment Corporation - VAX/VMS Version V4.7  99999999999999999999999   RRRRRRRRRR
RRRRRRRRRR   9999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999   RRRRRRRRRR

                         GGGG  U   U  III  M   M  III  N   N
                        G      U   U   I   MM MM   I   N   N
                        G      U   U   I   M M M   I   NN  N
                        G      U   U   I   M   M   I   N N N
                        G GGG  U   U   I   M   M   I   N  NN
                        G   G  U   U   I   M   M   I   N   N
                         GGG   UUUUU  III  M   M  III  N   N

       IIIIII     CCCCCCCC  EEEEEEEEEE    222222    DDDDDDDD       11
       IIIIII     CCCCCCCC  EEEEEEEEEE    222222    DDDDDDDD       11
         II     CC          EE          22      22  DD      DD   1111
         II     CC          EE          22      22  DD      DD   1111
         II     CC          EE                  22  DD      DD     11
         II     CC          EE                  22  DD      DD     11
         II     CC          EEEEEEEE          22    DD      DD     11
         II     CC          EEEEEEEE          22    DD      DD     11
         II     CC          EE              22      DD      DD     11
         II     CC          EE              22      DD      DD     11
         II     CC          EE            22        DD      DD     11
         II     CC          EE            22        DD      DD     11
       IIIIII     CCCCCCCC  EEEEEEEEEE  2222222222  DDDDDDDD     111111
       IIIIII     CCCCCCCC  EEEEEEEEEE  2222222222  DDDDDDDD     111111

                            FFFFFFFFFF      ;;;;        11
                            FFFFFFFFFF      ;;;;        11
                            FF              ;;;;      1111
                            FF              ;;;;      1111
                            FF                          11
                            FF                          11
                            FFFFFFFF        ;;;;        11
                            FFFFFFFF        ;;;;        11
                            FF              ;;;;        11
                            FF              ;;;;        11
                      ....  FF                ;;        11
                      ....  FF                ;;        11
                      ....  FF              ;;        111111
                      ....  FF              ;;        111111
```

File 1DUS121:[GUIMIN]ICE2D1.F;1 (1606,10,0), last revised on 8-DEC-1988 11:02, is a 84 block sequential file owned by UIC [GUIMIN]. The records are variable length with implied (CR) carriage control. The longest record is 80 bytes.

Job ICE2D1 (1176) queued to SYS$PRINT on 8-DEC-1988 11:03 by user GUIMIN, UIC [GUIMIN], under account 438901 at priority 100, started on printer LCA0 on 8-DEC-1988 11:03 from queue CSA2$LCA0.


```
RRRRRRRRRR   9999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999   RRRRRRRRRR
RRRRRRRRRR   99999999999999999999999  Digital Equipment Corporation - VAX/VMS Version V4.7  99999999999999999999999   RRRRRRRRRR
RRRRRRRRRR   9999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999999   RRRRRRRRRR
```

```
      IF(NSYS.EQ.1) THEN                                                ICE01170
        GEOM1=( -XDER(J)*XD+ZD)*XD/D                                    ICE01180
        GEOM2=(-(XD*XD-ZD*ZD)*XDER(J)+2.0*(XD*ZD))/D                    ICE01190
      ELSE                                                              ICE01200
        GEOM1=( -XDER(J)*XD+ZD)*ZD/D                                    ICE01210
        GEOM2=((ZD*ZD-XD*XD)-XDER(J)*2.0* XD*ZD )/D                     ICE01220
      ENDIF                                                             ICE01230
    DO 20 I =1,KYE                                                      ICE01240
        WK=AKY(I)*PI2                                                   ICE01250
        RKY=RHO*WK                                                      ICE01260
             IF(I.GT.1)THEN                                             ICE01270
               HPN(M,I)=(GEOM1*WK *BESSK0(RKY)+GEOM2*BESSK1(RKY)        ICE01280
     *                   /RHO)*WK                                       ICE01290
             ELSE                                                       ICE01300
               HPN(M,I)=GEOM2/(RHO*RHO)                                 ICE01310
             ENDIF                                                      ICE01320
 20        CONTINUE                                                     ICE01330
 10   CONTINUE                                                          ICE01340
C      WRITE(4,30)(HPN(J,1),J=1,NTR)                                    ICE01350
 30   FORMAT(4X,'HPN',/,(5(1PE12.4)))                                   ICE01360
      RETURN                                                            ICE01370
      END                                                               ICE01380
C                                                                       ICE01390
C                                                                       ICE01400
      SUBROUTINE KERNEL(K,KS,KS1,ZSURF,XDER,AMPDER)                     ICE01410
      implicit real*8 (a-h,o-z)
C     ..................................................                ICE01420
C    COMPUTE THE KERNEL IN THE INTEGRAL EQUATION                        ICE01430
C    AND SAVE IT                                                        ICE01440
C     ..................................................                ICE01450
      PARAMETER (PI=3.1415926)                                          ICE01460
      DIMENSION ZSURF(1),XDER(1),AMPDER(1)                              ICE01470
      DIMENSION AKERN(120,120,20)                                       ICE01480
      COMMON /KER/AKERN/KY/AKY(20)                                      ICE01490
      COMMON KYE,NSYS,NST,NPT,NXT,NTR,XDEL,TRDIS                        ICE01500
      PI2=PI*2.                                                         ICE01510
      DO 1 J=1,NTR                                                      ICE01520
        DO 1 I=1,KYE                                                    ICE01530
           AKERN(J,J,I)=0.0                                             ICE01540
  1   CONTINUE                                                          ICE01550
      IF(K.GT.1) GOTO 30                                                ICE01560
C  COMPUTE THE AKERNEL                                                  ICE01570
 1001 CONTINUE                                                          ICE01580
      DO 20 J=1,NTR                                                     ICE01590
         DO 20 L=1,NTR                                                  ICE01600
             IF(J.EQ.L) GOTO 20                                         ICE01610
             XD=(J-L)*XDEL                                              ICE01620
             ZD=ZSURF(J+KS)-ZSURF(L+KS)                                 ICE01630
             RHO=SQRT(XD*XD+ZD*ZD)                                      ICE01640
             GEOM=2.*(AMPDER(L+KS)/AMPDER(J+KS))*( XD*XDER(J+KS)-ZD)/RHOICE01650
             DO 10 I=1,KYE                                              ICE01660
             WK=AKY(I)*PI2                                              ICE01670
             IF(I.EQ.1) THEN                                            ICE01680
               AKERN(J,L,I)=   GEOM/RHO                                 ICE01690
             ELSE                                                       ICE01700
               AKERN(J,L,I)=    GEOM*WK*BESSK1(RHO*WK)                  ICE01710
             ENDIF                                                      ICE01720
 10        CONTINUE                                                     ICE01730
 20       CONTINUE                                                      ICE01740
      RETURN                                                            ICE01750
```

```fortran
     +   ' METERS')
 105  FORMAT(4X,'RELIEF OF ICE-WATER INTERFACE',/,    (10F7.2))
 106  FORMAT(4X,'X POSITION OF THE SYSTEM',/,(10F7.2))
 107  FORMAT(4X,'Z POSITION OF THE SYSTEM',/,(10F7.2))
 201  FORMAT(4X,'KY HOMONICS =',/,    (5E14.4))
 202  FORMAT(/,4X,'  XSYS (M)     ZSYS (M)      HXS(PPM)    HZS(PPM)')
 203  FORMAT(2X,2F11.2,3X,2(1PE12.4))
      STOP
      END
C
C
C
      SUBROUTINE SURF(NST,NPT,NXT,XDEL,ZSURF,XDER,AMPDER)
      implicit real*8 (a-h,o-z)
C ............................................................
C  GENERATE THE INFORMATION OF THE ICE-WATER INTERFACE, I.E.,
C  ZSURF, XDER, AMPDER
C ............................................................
      DIMENSION ZSURF(NXT),XDER(NXT),AMPDER(NXT)
      NE =NST+NPT
      CALL CUBDER(NPT+2,XDEL,ZSURF,XDER)
      DO 10 I=NXT,1,-1
        IF(I.GE.NE .OR. I.LT.NST) THEN
          ZSURF(I)=0.0
          XDER(I)=0.
          AMPDER(I)=1.
          GOTO 10
        ENDIF
       ZSURF(I)=ZSURF(I-NST+1)
       XDER(I)=XDER(I-NST+1)
       AMPDER(I)=SQRT(1.0+XDER(I)*XDER(I))
 10   CONTINUE
C     WRITE(4,50)(ZSURF(I),I=1,NXT-1)
C     WRITE(4,60)( XDER(I),I=1,NXT-1)
C     WRITE(4,70)(AMPDER(I),I=1,NXT-1)
 50   FORMAT(4X,'ZSURF',/,4X,(10F7.3))
 60   FORMAT(4X,' XDER',/,4X,(10F7.3))
 70   FORMAT(4X,'AMPDER',/,4X,(10F7.3))
      RETURN
      END
C
C
      SUBROUTINE HNPRIM(K,KS,ZSURF,XDER,AMPDER,XSYS,ZSYS)
C   ..................................................
C   COMPUTE THE NORMAL COMPONENT OF THE MAGNETIC FIELD
C   ..................................................
      implicit real*8 (a-h,o-z)
      PARAMETER (PI=3.1415926)
      DIMENSION ZSURF(1),XDER(1),AMPDER(1),XSYS(1),ZSYS(1)
      DIMENSION HPN(120,20)
      COMMON /KY/AKY(20)/HN/HPN
      COMMON KYE,NSYS,NST,NPT,NXT,NTR,    XDEL,TRDIS
      PI2=PI*2.
      DO 10 J=KS,NTR+KS-1
          M=J-KS+1
          XD=(J-1)*XDEL-XSYS(K)+TRDIS/2.
          ZD=ZSURF(J)-ZSYS(K)
          RHO=SQRT(XD*XD+ZD*ZD)
          D=RHO*RHO*AMPDER(J)*PI2
          IF(D.EQ.0.) PAUSE'BAD HPN'
```

```fortran
      I=1
      KK=0
 1001 CONTINUE
      DO 10 J=1,NTR
 10    W1(J) =HPN(J,I)*FAC
 999  CONTINUE
      DO 20 J=1,NTR
         SUM=0.0
         DU 30 L=1,NTR
          SUM=SUM+W1(L) *AKERN(J,L,I)
 30      CONTINUE
         W2(J)=FAC*(-SUM*XDEL+HPN(J,I))
 20   CONTINUE
      DO 40 J=1,NTR
 40     W1(J)=W2(J)
      KK=KK+1
      IF(KK.LT.NLOOP) GOTO 999
      DO 50 J=1,NTR
 50      CHAR(J,I)=W1(J)
      I=I+1
      IF(I.LE.KYE) GOTO 1001
C     WRITE(4,22)(CHAR(J,1),J=1,NTR)
 22   FORMAT(4X,'CHARGE',/,(5E14.4))
      RETURN
      END
C
C
C
      SUBROUTINE FIELD(K,KS,NM,AKDEL,HXS,HZS,XSYS,ZSYS,ZSURF,AMPDER)
      implicit real*8 (a-h,o-z)
C   ...................................................
C   .   COMPUTE THE SECONDARY MAGNETIC FIELD, BOTH
C   .   X AND Z COMPONENTS, EXPRESSED IN PPM OF THE
C   .   RECEIVED PRIMARY FIELD.
C   ...................................................
C
      PARAMETER (PI=3.1415926)
      DIMENSION AMPDER(1),CHAR(120,20),ZSURF(1),XSYS(1),ZSYS(1),AA(513)
      DIMENSION TEMP1(20),TEMP2(20)
      COMPLEX*16 CF(1024)
      INTEGER IWK(20)
      COMMON /CH/CHAR/KY/AKY(20)
      COMMON KYE,NSYS,NST,NPT,NXT,NTR,XDEL,TRDIS
      R =TRDIS
      XF=R/2.
       IF(NSYS.EQ.1) THEN
          HNORM=1./(2.*PI*R**3)
       ELSE
          HNORM=-1./(4.*PI*R**3)
       ENDIF
      DO 10 I=1,KYE
        WK=AKY(I)*PI*2.
        AZ=0.0
        AX=0.0
        DO 20 J=KS,NTR+KS-2
          M=J-KS+1
          RHO=SQRT((XSYS(K)-(J-1)*XDEL+XF)**2+(ZSYS(K)-ZSURF(J))**2)
          IF(I.GT.1) THEN
             AXZ=2.0*CHAR(M,I)*AMPDER(J)*WK*BESSK1(RHO*WK)/RHO
          ELSE
```

```
C
C
C   COMPUTE THE UNKNOWN PART OF THE  KERNEL
 30   CONTINUE
      M=KS-KS1+1
      KE2=NTR-M
      IF(M.LE.0) PAUSE 'BAD INPUT XSYS'
      IF(M.GE.NTR) GOTO 1001
      DO 40 J=1,KE2
          DO 40 L=1,KE2
           DO 40 I=1,KYE
              AKERN(J,L,I)=AKERN(J+M,L+M,I)
 40    CONTINUE
      DO 60 J=1,KE2
        DO 60 L=KE2+1,NTR
             XD=(J-L)*XDEL
             ZD=ZSURF(J+KS)-ZSURF(L+KS)
             RHO=SQRT(XD*XD+ZD*ZD)
             GEOM=2.*(AMPDER(L+KS)/AMPDER(J+KS))*( XD*XDER(J+KS)-ZD)/RHO
             DO 50 I=1,KYE
             WK=AKY(I)*PI2
             IF(I.EQ.1) THEN
               AKERN(J,L,I)=    GEOM/RHO
             ELSE
               AKERN(J,L,I)=     GEOM*WK*BESSK1(RHO*WK)
             ENDIF
 50        CONTINUE
 60   CONTINUE
         DO 80 J=KE2+1,NTR
           DO 80 L=1,NTR
             IF(J.EQ.L) GOTO 80
             XD=(J-L)*XDEL
             ZD=ZSURF(J+KS)-ZSURF(L+KS)
             RHO=SQRT(XD*XD+ZD*ZD)
             GEOM=2.*(AMPDER(L+KS)/AMPDER(J+KS))*( XD*XDER(J+KS)-ZD)/RHO
             DO 70 I=1,KYE
             WK=AKY(I)*PI2
             IF(I.EQ.1) THEN
               AKERN(J,L,I)=    GEOM/RHO
             ELSE
               AKERN(J,L,I)=     GEOM*WK*BESSK1(RHO*WK)
             ENDIF
 70        CONTINUE
 80   CONTINUE
C     WRITE(4,90)( AKERN(40,10,I)          ,I=1,KYE)
 90   FORMAT(4X,'KERNEL',/,(5(1PE12.4)))
      RETURN
      END
C
C
      SUBROUTINE CHARGE(NTR,KYE,XDEL,NLOOP)
      implicit real*8 (a-h,o-z)
C ..................................................................
C    COMPUTE THE CHARGE DISTRIBUTION ON THE SURFACE OF
C    THE SEA WATER
C ..................................................................
      DIMENSION CHAR(120,20),HPN(120,20),AKERN(120,120,20),W1(120)
      DIMENSION W2(120)
      COMMON /KER/AKERN/HN/HPN/CH/CHAR
      FAC=1.0/(2*3.1415926)
```

```
C    ...............................................
C    .  CUBIC SPLINE INTERPOLATION BEFORE TAKING THE
C    .  INVERSE FOURIER TRANSFORM.
C    ...............................................
C
      DIMENSION AKY(KYE), O(NK1),YA(KYE),Y2(20)
      CALL SPLINE(AKY,YA   ,KYE,1.E30,1.E30,Y2)
      O(1)=YA(1)
      DO 10 I=2,NK1
        XX=(I-1)*AKDEL
        KLO=1
        KHI=KYE
 1        IF(KHI-KLO.GT.1) THEN
            K=(KHI+KLO)/2
            IF(AKY(K).GT.XX) THEN
              KHI=K
            ELSE
              KLO=K
            ENDIF
          GOTO 1
          ENDIF
          H=AKY(KHI)-AKY(KLO)
          A=(AKY(KHI)-XX)/H
          B=(XX-AKY(KLO))/H
          O(I)=A*YA(KLO)+B*YA(KHI)+(A*(A*A-1.)*Y2(KLO)+
     *          B*(B*B-1.)*Y2(KHI))*(H*H)/6.
 10   CONTINUE
      RETURN
      END
C
      SUBROUTINE SPLINE(X,Y,N,YP1,YPN,Y2)
      implicit real*8 (a-h,o-z)
C    ..............................................................
C    . GIVEN ARRAYS X AND Y OF LENGHT N CONTAINING A TABULATED FUNCTION,
C    . I.E., YJ=F(XJ), WITH X1 Y2<X3...<XN, THIS ROUTINE RETURNS AN ARRAY
C    . Y2 OF LENGTH N WHICH CONTAINS THE SECOND DERIVATIVES OF THE INTER-
C    . POLATING FUNCTION AT THE TABULATED POINTS XJ. NATURAL SPLINE FROM
C    . NUMERICAL RECIPES, PRESS AT AL.
C    ..............................................................
C
      PARAMETER (NMAX=1024)
      DIMENSION X(N),Y(N),Y2(N),U(NMAX)
      IF (YP1.GT..99E30) THEN
        Y2(1)=0.
        U(1)=0.
      ELSE
        Y2(1)=-0.5
        U(1)=(3./(X(2)-X(1)))*((Y(2)-Y(1))/(X(2) X(1))-YP1)
      ENDIF
      DO 11 I=2,N-1
        SIG=(X(I)-X(I-1))/(X(I+1)-X(I-1))
        P=SIG*Y2(I-1)+2.
        Y2(I)=(SIG-1.)/P
        U(I)=(6.*((Y(I+1)-Y(I))/(X(I+1)-X(I))-(Y(I)-Y(I-1))
     *        /(X(I)-X(I-1)))/(X(I+1)-X(I-1))-SIG*U(I-1))/P
 11   CONTINUE
      IF (YPN.GT..99E30) THEN
        QN=0.
        UN=0.
      ELSE
```

```
           AXZ=2.0*CHAR(M,I)*AMPDER(J)/(RHO*RHO)                      ICE02940
         ENDIF                                                        ICE02950
          AZ=(ZSYS(K)-ZSURF(J))*AXZ+AZ                                ICE02960
          AX=(XSYS(K)-(J-1)*XDEL+XF)*AXZ+AX                           ICE02970
 20     CONTINUE                                                      ICE02980
        TEMP1(I)=AZ*XDEL                                              ICE02990
        TEMP2(I)=AX*XDEL                                              ICE03000
 10   CONTINUE                                                        ICE03010
      NK =2**NM                                                       ICE03020
      IF(NK*AKDEL .GT. AKY(KYE)) PAUSE 'BAD AKDEL OR AKY'             ICE03030
      CALL CUBSPL(KYE,NK,AKDEL,AKY,TEMP1,AA)                          ICE03040
      SUM=0.0                                                         ICE03050
      DO 11 I=2,NK                                                    ICE03060
 11     SUM=SUM+AA(I)                                                 ICE03070
      HZS=(SUM+AA(1)/2.)*AKDEL*2.                                     ICE03080
      CALL CUBSPL(KYE,NK,AKDEL,AKY,TEMP2,AA)                          ICE03090
      SUM=0.0                                                         ICE03100
      DO 12 I=2,NK                                                    ICE03110
 12     SUM=SUM+AA(I)                                                 ICE03120
      HXS=(SUM+AA(1)/2.)*AKDEL*2.                                     ICE03130
      HXS=HXS/HNORM*1000000                                           ICE03140
      HZS=HZS/HNORM*1000000                                           ICE03150
      RETURN                                                          ICE03160
      END                                                             ICE03170
C                                                                     ICE03180
C                                                                     ICE03190
      SUBROUTINE CUBDER(N,XDEL,YA,YD)                                 ICE03200
      implicit real*8 (a-h,o-z)
C  ..............................................                     ICE03210
C  .  COMPUTE THE DERIVATIVES AT EACH NODE USING                      ICE03220
C  .  CUBIC SPLINE.                                                   ICE03230
C  .  NOTE: N=NPT+2 . ONE POINT ADDED TO EACH END.                    ICE03240
C  ..............................................                     ICE03250
C                                                                     ICE03260
      DIMENSION XA(50),YA(N),Y2(50),YD(N)                             ICE03270
      DO 11 I=1,N                                                     ICE03280
 11    XA(I)=(I-1)*XDEL                                               ICE03290
      DO 12 I=N-1,2,-1                                                ICE03300
 12    YA(I)=YA(I-1)                                                  ICE03310
      YA(1)=0.                                                        ICE03320
      YA(N)=0.                                                        ICE03330
      CALL SPLINE(XA,YA,N,0.,0.,Y2)                                   ICE03340
      DO 10 I=2,N-1                                                   ICE03350
        XX=XA(I)                                                      ICE03360
        KLO=I                                                         ICE03370
        KHI=I+1                                                       ICE03380
          H=XA(KHI)-XA(KLO)                                           ICE03390
          A=(XA(KHI)-XX)/H                                            ICE03400
          B=(XX-XA(KLO))/H                                            ICE03410
          XD=XA(KHI)-XA(KLO)                                          ICE03420
          YD(I-1)=(YA(KHI)-YA(KLO))/XD-XD*((3.*A*A-1.)*Y2(KLO)        ICE03430
     *            -(3.*B*B-1.)*Y2(KHI))/6.                            ICE03440
 10   CONTINUE                                                        ICE03450
      DO 14 I=1,N-2                                                   ICE03460
 14    YA(I)=YA(I+1)                                                  ICE03470
      RETURN                                                          ICE03480
      END                                                             ICE03490
C                                                                     ICE03500
      SUBROUTINE CUBSPL(KYE,NK1,AKDEL,AKY,YA,D)                       ICE03510
      implicit real*8 (a-h,o-z)
```

```fortran
c
c
      Function BESSI1(x)
      implicit real*8 (a-h,o-z)
c ...............................................................
c ... Returns the modified Bessel function I1(x) for any real x.
c ...............................................................
      real*8 y,p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,q8,q9
      data p1,p2,p3,p4,p5,p6,p7/0.5d0,0.87890594d0,0.51498869d0,
     + 0.15084934d0,0.2658733d-1,0.301532d-2,0.32411d-3/
      data q1,q2,q3,q4,q5,q6,q7,q8,q9/0.39894228d0,-0.3988024d-1,
     +  -0.362018d-2,0.163801d-2,-0.1031555d-1,0.2282967d-1,
     +  -0.289531d-1,0.1787654d-1,-0.420059d-2/
      if(abs(x).lt.3.75) then
        y=(x/3.75)**2
        BESSI1=x*(p1+y*(p2+y*(p3+y*(p4+y*(p5+y*(p6+y*p7))))))
      else
        ax=abs(x)
        y=3.75/ax
        BESSI1=(exp(ax)/sqrt(ax))*(q1+y*(q2+y*(q3+y*(q4+y*(q5+
     +     y*(q6+y*(q7+y*(q8+y*q9))))))))
      endif
      return
      end
c
c
      Function BESSK1(x)
      implicit real*8 (a-h,o-z)
c ...............................................................
c ... Returns the modified Bessel function K1(x) for positive x.
c ...............................................................
      real*8 y,p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7
      data p1,p2,p3,p4,p5,p6,p7/1.0d0,0.15443144d0,-0.67278579d0,
     +      -0.18156897d0,-0.1919402d-1,-0.110404d-2,-0.4686d-4/
      data q1,q2,q3,q4,q5,q6,q7/1.25331414d0,0.23498619d0,-0.3655620d-1,
     +      0.1504268d-1,-0.780353d-2,0.325614d-2,-0.68245d-3/
      if(x.le.2.0) then
        y=x*x/4.0
        BESSK1= (log(x/2.0)*BESSI1(x))+(1.0/x)*(p1+y*(p2+y*(p3+
     +             y*(p4+y*(p5+y*(p6+y*p7))))))
      else
        if(x.gt.40) then
          BESSK1=0.0
        else
          y=(2.0/x)
          BESSK1=(exp(-x)/sqrt(x))*(q1+y*(q2+y*(q3+y*(q4+y*(q5+
     +             y*(q6+y*q7))))))
        endif
      endif
      return
      end
```

```
      QN=0.5
      UN=(3./(X(N)-X(N-1)))*(YPN-(Y(N)-Y(N-1))/(X(N)-X(N-1)))       ICE04120
    ENDIF                                                           ICE04130
    Y2(N)=(UN-QN*U(N-1))/(QN*Y2(N-1)+1.)                            ICE04140
    DO 12 K=N-1,1,-1                                                ICE04150
      Y2(K)=Y2(K)*Y2(K+1)+U(K)                                      ICE04160
12  CONTINUE                                                        ICE04170
    RETURN                                                          ICE04180
    END                                                             ICE04190
C                                                                   ICE04200
C                                                                   ICE04210
    Function BESSI0(x)                                              ICE04220
    implicit real*8 (a-h,o-z)
C .........................................................          ICE04230
C .  RETURNS THE MODIFIED BESSEL FUNCTION I0(X) FOR ANY REAL X.      ICE04240
C ..........................................................         ICE04250
    real*8 y,p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,q8,q9        ICE04260
    data p1,p2,p3,p4,p5,p6,p7/1.0d0,3.5156229d0,3.0899424d0,        ICE04270
   +  1.2067492d0,0.2659732d0,0.360768d-1,0.45813d-2/               ICE04280
    data q1,q2,q3,q4,q5,q6,q7,q8,q9/0.39894228d0,0.1328592d-1,      ICE04290
   +  0.225319d-2,-0.157565d-2,0.916281d-2,-0.2057706d-1,           ICE04300
   +  0.2635537d-1,-0.1647633d-1,0.392377d-2/                       ICE04310
    if(abs(x).lt.3.75) then                                         ICE04320
      y=(x/3.75)**2                                                 ICE04330
      BESSI0=p1+y*(p2+y*(p3+y*(p4+y*(p5+y*(p6+y*p7)))))             ICE04340
    else                                                            ICE04350
      ax=abs(x)                                                     ICE04360
      y=3.75/ax                                                     ICE04370
      BESSI0=(exp(ax)/sqrt(ax))*(q1+y*(q2+y*(q3+y*(q4+y*(q5+        ICE04380
   +    y*(q6+y*(q7+y*(q8+y*q9))))))))                              ICE04390
    endif                                                           ICE04400
    return                                                          ICE04410
    end                                                             ICE04420
c                                                                   ICE04430
c                                                                   ICE04440
    Function BESSK0(x)                                              ICE04450
    implicit real*8 (a-h,o-z)
C ..............................................                     ICE04460
c ...  Returns the modified Bessel function K0(x) for positive x.   ICE04470
C ..............................................                     ICE04480
    real*8 y,p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7              ICE04490
    data p1,p2,p3,p4,p5,p6,p7/-0.57721566d0,0.42278420d0,0.23069756d0,ICE04500
   +     0.3488590d-1,0.262698d-2,0.10750d-3,0.74d-5/               ICE04510
    data q1,q2,q3,q4,q5,q6,q7/1.25331414d0,-0.7832358d-1,0.2189568d-1,ICE04520
   +     -0.1062446d-1,0.587872d-2,-0.251540d-2,0.53208d-3/         ICE04530
    if(x.le.2.0) then                                               ICE04540
      y=x*x/4.0                                                     ICE04550
      BESSK0=(-log(x/2.0)*BESSI0(x))+(p1+y*(p2+y*(p3+              ICE04560
   +            y*(p4+y*(p5+y*(p6+y*p7))))))                        ICE04570
    else                                                            ICE04580
      if(x.gt.40.) then                                             ICE04590
        BESSK0=0.0                                                  ICE04600
      else                                                          ICE04610
        y=(2.0/x)                                                   ICE04620
        BESSK0=(exp(-x)/sqrt(x))*(q1+y*(q2+y*(q3+y*(q4+y*(q5+      ICE04630
   +            y*(q6+y*q7))))))                                    ICE04640
      endif                                                         ICE04650
    endif                                                           ICE04660
    return                                                          ICE04670
    end                                                             ICE04680
```

```
0000000000   44444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444   0000000000
0000000000   4444444444444444444444444  Digital Equipment Corporation - VAX/VMS Version V4.7  4444444444444444444444444   0000000000
0000000000   44444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444   0000000000
```

```
                 GGGG  U   U  III  M   M  III  N   N
                G      U   U   I   MM MM   I   N   N
                G      U   U   I   M M M   I   NN  N
                G      U   U   I   M   M   I   N N N
                G GGG  U   U   I   M   M   I   N  NN
                G   G  U   U   I   M   M   I   N   N
                 GGG   UUUUU  III  M   M  III  N   N

        IIIIII    NN        NN  VV        VV  HH        HH  ZZZZZZZZZZ
        IIIIII    NN        NN  VV        VV  HH        HH  ZZZZZZZZZZ
          II      NN        NN  VV        VV  HH        HH          ZZ
          II      NN        NN  VV        VV  HH        HH          ZZ
          II      NNNN      NN  VV        VV  HH        HH        ZZ
          II      NNNN      NN  VV        VV  HH        HH        ZZ
          II      NN  NN    NN  VV        VV  HHHHHHHHHH      ZZ
          II      NN  NN    NN  VV        VV  HHHHHHHHHH      ZZ
          II      NN    NNNN    VV        VV  HH        HH    ZZ
          II      NN    NNNN    VV        VV  HH        HH    ZZ
          II      NN        NN    VV    VV    HH        HH  ZZ
          II      NN        NN    VV    VV    HH        HH  ZZ
        IIIIII    NN        NN      VV        HH        HH  ZZZZZZZZZZ
        IIIIII    NN        NN      VV        HH        HH  ZZZZZZZZZZ

                          FFFFFFFFFF      ;;;;        11
                          FFFFFFFFFF      ;;;;        11
                          FF              ;;;;      1111
                          FF              ;;;;      1111
                          FF                          11
                          FF                          11
                          FFFFFFFF        ;;;;        11
                          FFFFFFFF        ;;;;        11
                          FF              ;;;;        11
                          FF              ;;;;        11
                  ....    FF                ;;        11
                  ....    FF                ;;        11
                  ....    FF              ;;        111111
                  ....    FF              ;;        111111
```

File 1DUS121:[GUIMIN]INVHZ.F;1 (1607,8,0), last revised on 8-DEC-1988 11:02, is a 243 block sequential file owned by UIC [GUIMIN]. The records are variable length with implied (CR) carriage control. The longest record is 80 bytes.

Job INVHZ (1178) queued to SYS$PRINT on 8-DEC-1988 11:03 by user GUIMIN, UIC [GUIMIN], under account 418901 at priority 100, started on printer LCA0 on 8-DEC-1988 11:03 from queue CSA3$LCA0.

```
0000000000   44444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444   0000000000
0000000000   4444444444444444444444444  Digital Equipment Corporation - VAX/VMS Version V4.7  4444444444444444444444444   0000000000
0000000000   44444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444444   0000000000
```

```fortran
         IF(KK.GT.NITER) THEN
         WRITE(4,17)
         STOP
         ENDIF
      ENDIF
 11   FORMAT(4X,'COMPUTED ICE-WATER INTERFACE',/,(5(1PE14.2)))
 12   FORMAT(4X,'COMPUTED DATA',/,(5(1PE14.4)))
 16   FORMAT(4X,'FIRST CONVERGENCE CRITERION SATISFIED')
 17   FORMAT(4X,'SECOND CONVERGENCE CRITERION SATISFIED')
 101  FORMAT(4X,'HORIZONTAL AXIS DIPOLE TRANSMITTER',/)
 102  FORMAT(4X,'VERTICAL   AXIS DIPOLE TRANSMITTER',/)
 103  FORMAT(4X,'NST =',I5,2X,'NPT =',I5,2X,'NXT =',I5,/,
     + 4X,'NRESP =',I5,2X,'NTRUNK =',I5,2X,'NLOOP =',I5,/,4X,'IREC =',
     + I5,2X,'NITER =',I5)
 104  FORMAT(4X, 'T-R SEPARATION =',F7.2,' METERS',/,4X,'XDEL =',F7.2,
     +  ' METERS',/,4X,'FIRST CONVERGENCE CRITERIA RMS ERR =',1PE12.2)
 105  FORMAT(4X,'INITIAL ICE-WATER INTERFACE',/,    (10F .2))
 106  FORMAT(11X,'XSYS (M)        ZSYS (M)    DATA IN PPM',/,(3F17.2))
      GOTO 1001
      END
C
      SUBROUTINE HARMOC(KYE,NM,AKDEL,AKY)
C ..................................................
C     ASSIGN  KY HOMONICS. THE FOLLOWING IS OK FOR ZR=10 - 100 METERS.
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION AKY(KYE)
      KYE=15
      NM=7
      AKYMIN=1.0/(2.0*1000)
      AKDEL=0.0006
      AKLOG=0.17
      AKYEX=DLOG10(AKYMIN)
      AKY(1)=0.0
      DO 20 I=1,KYE-1
        EX=AKYEX+(I-1)*AKLOG
        AKY(I+1)=10**EX
 20   CONTINUE
C     WRITE(4,201)(AKY(I),I=1,KYE)
 201  FORMAT(4X,'KY HOMONICS =',/,   (5E14.4))
C ..................................................
      RETURN
      END
C
C
      SUBROUTINE FORDIV(NPT,NRESP,ZH1,CALC,WJ)
C
C    COMPUTE THE APPROXIMATE JACOBIAN MATRIX
C
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      PARAMETER (NDD=40,NPP=30,NXT=200)
      PARAMETER (PI2=3.1415926*2.,PI=3.1415926,KM=256)
      DIMENSION ZH0(NXT),XSYS(NDD),ZSYS(NDD),WJ(NDD,NPP),ZH1(1)
      DIMENSION AMPDER(NXT),TEMP(20),AA(KM),ZH2(NPP),CALC2(NDD),CALC(1)
      COMMON /FCM/ZH0,XSYS,ZSYS,AMPDER,NM,IREC,AKDEL
      COMMON /KY/AKY(20)
      COMMON KYE,NSYS,NST,XDEL,TRDIS
      CALL FORMOD(NPT,NRESP,ZH1,CALC)
      K=1
 1001 CONTINUE
      DO 10 I=1, NPT
```

```
      PROGRAM INVSM
C
C   INVERSION PROGRAM FOR IMAGING THE ICE-WATER INTERFACE FROM THE
C   AIRBORNE ELECTROMAGNETIC DATA.
C   QUADTRATIC PROGRAMING APPROACH.
C   SMOOTH INVERSION
C
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      PARAMETER (NXT=200,NTRUNK=100,NLOOP=8,NDD=40,NPP=30)
      DIMENSION FKERN(NDD,NPP),DELZH(NDD),AMPDER(NXT),ZH0(NXT)
      DIMENSION XSYS(NDD),ZSYS(NDD),CALC(NDD)
      COMMON /DATA/NRESP,DAT(NDD),SD(NDD)
      COMMON /MODEL/NPT,ZH1(NPP),DM(NDD),LJMP(NPP)
      COMMON /KY/AKY(20)
      COMMON /FCM/ZH0,XSYS,ZSYS,AMPDER,NM,IREC,AKDEL
      COMMON KYE,NSYS,NST,XDEL,TRDIS
      open (unit=3, file='invhz.dat')
      open (unit=4, file='invhz.out')
      READ(3,*)
      READ(3,*)NSYS,IREC,NST,NPT,NRESP,NITER
      READ(3,*)
      READ(3,*)XDEL,TRDIS,ERR,PMU
      READ(3,*)
      READ(3,*)(ZH1(I),I=1,NPT)
      READ(3,*)
      READ(3,*)(XSYS(I),ZSYS(I),DAT(I),CALC(I),I=1,NRESP)
      READ(3,*)
      READ(3,*)(SD(I),I=1,NRESP)
      READ(3,*)
      READ(3,*)(LJMP(I),I=1,NPT)
      IF(IREC.EQ.2) THEN
        DO 3 I=1,NRESP
 3        DAT(I)=CALC(I)
      ENDIF
C
C     OUTPUT THE INPUT DATA
C
      IF(NSYS.EQ.1) WRITE(4,101)
      IF(NSYS.EQ.2) WRITE(4,102)
      IF(NSYS.NE.1 .AND. NSYS.NE.2) PAUSE 'BAD INPUT NSYS'
      IF(IREC.NE.1 .AND. IREC.NE.2) PAUSE 'BAD INPUT IREC'
      WRITE(4,104)TRDIS,XDEL,ERR
      WRITE(4,103)NST,NPT,NXT,NRESP,NTRUNK,NLOOP,IREC,NITER
      WRITE(4,105)(  ZH1(I),I=1,NPT)
      WRITE(4,108)(XSYS(I),ZSYS(I),DAT(I),I=1,NRESP)
C
C  ASSIGN THE KY HARMONICS
C
      CALL HARMOC(KYE,NM,AKDEL,AKY)
      KK-1
 1001 CONTINUE
      KK=KK+1
C  CALL THE SMOOTH INVERSION ROUTINE
      CALL OCCAM1(1,ERR,TOBT,0,NIT,STEPSZ,KONV,4,PMU)
      WRITE(4,11 )(ZH1(I),I=1,NPT)
      WRITE(4,12)(DM(I),I=1,NRESP)
      IF(TOBT.LT.ERR) THEN
         WRITE(4,16)
         STOP
      ELSE
```

```
C  TOBT = RMS MISFIT OF NEW MODEL RESPONSE.                              INV01800
C  NIT = NUMBER OF PREVIOUS CALLS TO OCCAM1 DURING THIS INVERSION        INV01810
C  STEPSZ = SUM OF SQUARES OF THE CHANGES IN THE MODEL PARAMETERS        INV01820
C  KONV = STATUS FLAG:                                                   INV01830
C    0 = NORMAL EXIT FOR A SUCCESSFUL ITERATION,                         INV01840
C    1 = A PERFECTLY SMOOTH MODEL HAS BEEN FOUND FOR THE REQUIRED TOLERAINV01850
C    2 = CONVERGENCE PROBLEMS.  UNABLE TO FIND A MODEL WITH AN R.M.S. MIINV01860
C          LESS THAN OR EQUAL TO THAT OF THE INITIAL MODEL.              INV01870
C  UM = THE MU VALUE USED AT THE MINIMUM OF THE MISFIT IN THIS ITERATIONINV01880
C                                                                        INV01890
C SUBROUTINES REQUIRED AND SUPPLIED:                                     INV01900
C  MAKDEL, A SUBROUTINE WHICH CREATES THE ROUGHENING MATRIX              INV01910
C  TOFMU(AMU), A FUNCTION WHICH RETURNS THE RMS MISFIT OF THE RESPONSE   INV01920
C     OF THE MODEL CONSTRUCTED USING THE LAGRANGE MULTIPLIER AMU.        INV01930
C     CALLS CHOLIN,CHOLSL,FORMOD,ANORM.                                  INV01940
C  INITM, TRMULT, MULT AND ANORM, SUBROUTINES FOR SIMPLE MATRIX OPERATIOINV01950
C  FMIN, A SUBROUTINE WHICH MINIMISES A UNIVARIATE FUNCTION              INV01960
C  FROOT, A SUBROUTINE WHICH FINDS THE ROOT OF A UNIVARIATE FUNCTION     INV01970
C  MINBRK, A SUBROUTINE WHICH BRACKETS A UNIVARIATE MINIMUM              INV01980
C                                                                        INV01990
C SUBROUTINES WHICH MUST BE SUPPLIED BY THE USER:                        INV02000
C  FORMOD(NP,ND,PM,DP,DM), COMPUTES THE FORWARD FUNCTION FOR MODEL PM()  INV02010
C    THE DATA PARAMETERS DP() AND RETURNS IT IN DM().                    INV02020
C  FORDIV(NP,ND,PM,DP,DM,WJ), COMPUTES THE FORWARD FUNCTION AS DOES FORMINV02030
C    BUT ALSO RETURNS THE MATRIX OF PARTIALS, WJ(ND,NP).                 INV02040
C                                                                        INV02050
C  PARAMETERS NDD AND NPP SHOULD BE SET TO THE MAXIMUM DIMENSIONS OF THEINV02060
C    DATA AND PARAMETER VECTORS, RESPSECTFULLY.  DON'T FORGET TO ADJUST  INV02070
C    THEM IN TOFMU, MAKDEL, BLOCKDATA AND ANY I/O SUBROUTINES.           INV02080
C                                                                        INV02090
C NOTE THAT DOUBLE PRECISION IS USED THROUGHOUT.  SOME COMPILERS WILL REINV02100
C  EXTERNAL TOFMU TO BE DECLARED IN OCCAM1'S CALLING PROGRAM.            INV02110
C                                                                        INV02120
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)                               INV02130
      PARAMETER (NDD = 40, NPP = 30)                                     INV02140
      COMMON /DATA/ ND,D(NDD),SD(NDD)                                    INV02150
      COMMON /MODEL/ NP,PM(NPP),DM(NDD),LJMP(NPP)                        INV02160
C /RESULT/ IS USED TO CARRY INFO IN AND OUT OF TOFMU AND STORE LAST      INV02170
C    VALUE OF MU USED.                                                   INV02180
      COMMON /RESULT/ PTP(NPP,NPP),WJTWJ(NPP,NPP),WJTWD(NPP),PM2(NPP),   INV02190
     * PMU,NFOR,NITER,FRAC,RLAST,IFFTOL                                  INV02200
      DIMENSION WJ(NDD,NPP),DD(NDD),DHAT(NDD),WK(NPP)                    INV02210
      EXTERNAL TOFMU                                                     INV02220
C                                                                        INV02230
C ON FIRST ENTRY CREATE TRANS(P).P, WHERE P IS THE ROUGHENING MATRIX OR  INV02240
C  ROUGHENING MATRIX SQUARED, AND SET MU TO AN ARBITRARY (LARGE) VALUE.  INV02250
C                                                                        INV02260
      IF (NITER .EQ. 0) THEN                                             INV02270
        CALL MAKDEL(IRUF)                                                INV02280
C       PMU = 10000.                                                     INV02290
        PMU=UM                                                           INV02300
        RLAST = 10000.                                                   INV02310
        KONV = 0                                                         INV02320
      END IF                                                             INV02330
C FRAC CONTROLS THE STEP SIZE; NORMALLY WILL REMAIN AT 1.0 UNLESS WE HAVINV02340
C CONVERGENCE PROBLEMS                                                   INV02350
      FRAC = 1.0                                                         INV02360
C CALCULATE THE MATRIX OF PARTIALS AND MODEL RESPONSE                    INV02370
      CALL FORDIV(NP,ND,PM,DM,WJ)                                        INV02380
      WRITE(IOUT,11)(DM(I),I=1,ND)                                       INV02390
```

```
10       ZH2(I) = ZH1(I)                                                INV01200
         ZH2(K) = ZH1(K)+0.2                                            INV01210
      CALL FORMOD(NPT,NRESP,ZH2,CALC2)                                  INV01220
      DO 30 I=1, NRESP                                                  INV01230
         WJ(I,K) = (CALC2(I)-CALC(I))/0.2                               INV01240
30    CONTINUE                                                          INV01250
      K=K+1                                                             INV01260
      IF(K.LE.NPT) GOTO 1001                                            INV01270
C     DO 61 I=1,NRESP                                                   INV01280
C     WRITE(4,60)(WJ(I,J),J=1,NPT)                                      INV01290
C61   CONTINUE                                                          INV01300
60    FORMAT(4X,'INVERSION KERNEL',/,(6(1PE14.4)))                      INV01310
      RETURN                                                            INV01320
      END                                                               INV01330
C***********************************************************************INV01340
      SUBROUTINE OCCAM1(IRUF,TOLREQ,TOBT,IBUG,NIT,STEPSZ,KONV,IOUT,UM)  INV01350
C                                                                       INV01360
C MODIFIED VERSION OF OCCAM1                                            INV01370
C OCCAM1 EXECUTES ONE ITERATION OF A ONE DIMENSIONAL SMOOTH MODEL FINDERINV01380
C REFERENCE: CONSTABLE, PARKER & CONSTABLE, 1987: GEOPHYSICS 52, 289-300INV01390
C                                                                       INV01400
C S. CONSTABLE, MARCH 1986, S.I.O., LA JOLLA, CA 92093, U.S.A. (VERSION INV01410
C                OCTOBER 1987,                                 (VERSION INV01420
C                                                                       INV01430
C IF YOU OBTAIN THIS CODE FROM A THIRD PERSON, PLEASE SEND YOUR NAME ANDINV01440
C TO S. CONSTABLE.  YOU WILL THEN RECEIVE UPDATES, NEWS ON BUGS, ETC.   INV01450
C                                                                       INV01460
C                                                                       INV01470
C ON INPUT:                                                             INV01480
C  /DATA/ CONTAINS                                                      INV01490
C     ND = NUMBER OF DATA                                               INV01500
C     D(NDD) = VECTOR OF DATA VALUES                                    INV01510
C     SD(NDD) = VECTOR OF ABSOLUTE STANDARD ERRORS IN THE DATA          INV01520
C     DP(NDD,4) = MATRIX CONTAINING UP TO FOUR PARAMETERS ASSOCIATED WITINV01530
C        EACH DATUM (RANGE, FREQUENCY, DATA TYPE ETC)                   INV01540
C     NPD = NUMBER OF PARAMETERS ASSOCIATED WITH EACH DATUM (E.G. FOR FRINV01550
C              NPD = 1, FOR FREQ. AND RANGE NPD = 2)                    INV01560
C                                                                       INV01570
C  /MODEL/ CONTAINS                                                     INV01580
C     NP = NUMBER OF MODEL PARAMETERS, USUALLY THE NUMBER OF LAYERS     INV01590
C     PM(NPP) = VECTOR OF INITIAL MODEL PARAMETERS, USUALLY LOG10(LAYER INV01600
C     LJMP(NPP) = A VECTOR DESCRIBING PLACES WITHIN THE MODEL TO RELAX TINV01610
C          SMOOTHNESS CONSTRAINT, IN MONOTONIC ORDER FOLLOWED BY A ZERO.INV01620
C          THUS LJMP = 4,10,0                                           INV01630
C          WOULD ALLOW A DISCONTINUITY BETWEEN LAYER 3 AND 4 AND LAYERS INV01640
C                                                                       INV01650
C  IRUF = 1 FOR FIRST DERIVATIVE MINIMIZATION                           INV01660
C       = 2 FOR SECOND DERIVATIVE MINIMIZATION                          INV01670
C  TOLREQ = REQUIRED RMS MISFIT                                         INV01680
C  IBUG = 1 IF A TABLE OF RMS MISFIT VERSUS LAGRANGE MULTIPLIER IS REQUIINV01690
C    (USEFUL FOR TRACKING DOWN CONVERGENCE PROBLEMS)                    INV01700
C         0 OTHERWISE (LOTS FASTER)                                     INV01710
C  IOUT = FORTRAN UNIT NUMBER FOR OUTPUT, CAN EITHER BE TERMINAL OR A FIINV01720
C  UM = THE MU VALUE USED AT THE MINIMUM OF THE MISFIT IN THE LAST CALL INV01730
C       TO OCCAM1                                                       INV01740
C                                                                       INV01750
C ON OUTPUT:                                                            INV01760
C  /MODEL/ CONTAINS                                                     INV01770
C     PM(NPP) = VECTOR OF UPDATED MODEL PARAMETERS                      INV01780
C     DM(NDD) = VECTOR CONTAINING RESPONSE OF NEW MODEL                 INV01790
```

```
165        WK(I) = PM2(:) - PM2(I-1)
      ELSE
        DO 170 I = 2,NP-1
170        WK(I) = PM2(:+1) - 2.*PM2(I) + PM2(I-1)
        WK(NP) = 0.0
      END IF
C REMOVE ENTRIES IN ROUGHNESS VECTOR WHERE JUMPS ARE ALLOWED
      I = 1
175   IF (LJMP(I) .NE. 0) THEN
        WK(LJMP(I)) = 0.0
        I = I+1
        GOTO 175
      END IF
      RUF = ANORM(NP,WK)
C IF WE HAVE ATTAINED THE INTERCEPT BUT THE MODEL IS GETTING ROUGHER
C WE HAVE PROBLEMS WITH CONERGENCE.
C SAVE NEW MODEL AND COMPUTE STEP SIZE
      DO 180 I = 1,NP
        WK(I) = PM2(I) - PM(I)
180     PM(I) = PM2(I)
      STEPSZ = ANORM(NP,WK)
      STEPSZ = SQRT(STEPSZ/NP)
      WRITE(IOUT,*) 'STEPSIZE IS = ',STEPSZ
      WRITE(IOUT,*) 'ROUGHNESS IS = ',RUF
      NITER = NITER + 1
      RETURN
      END
C
C***********************************************************************
      SUBROUTINE MAKDEL(IRUF)
C
C MAKDEL CONSTRUCTS TRANS(DEL).DEL FOR 1ST OR 2ND DERIVATIVE ROUGHNESS
C
C  ON INPUT:
C   IRUF = 1 FOR 1ST DERIVATIVE ROUGHNESS
C          2 FOR 2ND DERIVATIVE ROUGHNESS
C
C /MODEL/ CONTAINS
C   LJMP = ARRAY OF LAYER LOCATIONS WHERE A JUMP IN RESISTIVITY IS ALLOW
C          TERMINATED BY A ZERO
C
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      PARAMETER (NDD = 40, NPP = 30)
      COMMON /MODEL/ NP,PM(NPP),DM(NDD),LJMP(NPP)
      COMMON /RESULT/ PTP(NPP,NPP),WJTWJ(NPP,NPP),WJTWD(NPP),PM2(NPP),
     * PMU,NFOR,NITER,FRAC,RLAST,IFFTOL
      DIMENSION DEL(NPP,NPP)
C
C MAKE PARTIALS MATRIX
      CALL INITM(NPP,NP,NP,DEL)
      DO 10 I = 2,NP
        DEL(I,I-1) = -1.
10      DEL(I,I) = 1.
C REMOVE ANY ENTRIES WHERE MODEL IS ALLOWED TO JUMP
      I = 0
20    I = I+1
      N = LJMP(I)
C CANNOT REMOVE AIR/EARTH INTERFACE (N=1):
      IF (N .EQ. 1) GOTO 20
      IF (I.NE.NP+1 .AND. N.NE.0) THEN
```

```
 11   FORMAT(4X,'COMPUTED DATA',/,(5(1PE14.4)))
C CALC MISFIT VECTOR AND MISFIT
      DO 60 I = 1,ND
60      DD(I) = (D(I) - DM(I))/SD(I)
      CHI0 = ANORM(ND,DD)
      TOL0 = SQRT(CHI0/ND)
      IF (NITER .EQ. 0) THEN
        WRITE(IOUT,*) 'STARTING R.M.S. = ',TOL0
      END IF
C WEIGHT THE PARTIALS MATRIX
      DO 70 I = 1,ND
        DO 70 J = 1,NP
70        WJ(I,J) = WJ(I,J)/SD(I)
C FORM W.J.TRANS(W.J)
      CALL TRMULT(NDD,ND,NPP,NP,NP,WJ,WJ,WJTWJ)
C FORM THE WEIGHTED, TRANSLATED DATA AND PREMULTIPLY BY TRANS(W.J)
      CALL MULT(NDD,ND,NPP,NP,1,WJ,PM,DHAT)
      DO 80 I = 1,ND
80      DHAT(I) = DD(I) + DHAT(I)
      CALL TRMULT(NDD,ND,NPP,NP,1,WJ,DHAT,WJTWD)
C PRODUCE THE MISFIT FUNCTION IF REQUIRED
      IF (IBUG .EQ. 1) THEN
        WRITE(IOUT,*) 'MISFIT AS A FUNCTION OF MU'
        DO 100 K = 18,1,-1
          AMU = 10.**(FLOAT(K)/2. - 1.)
          T = TOFMU(AMU)
          IF (T .GE. 1.0E+09) GOTO 110
100       WRITE(IOUT,*) AMU,T
      END IF
110   CONTINUE
      WRITE(IOUT,*) ' '
      WRITE(IOUT,*) '** ITERATION ',NITER+1,' **'
C THE NEXT BLOCK OF CODE CONTROLS THE SELECTION OF THE LAGRANGE MULTIPLI
C PMU
120   TOLM = 0.2
      AMU1 = 0.9*PMU
      AMU2 = PMU
      NFOR = 0
      IFFTOL = 0
C FIND MINIMUM
      CALL MINBRK(AMU1,AMU2,AMU3,TAMU1,TAMU2,TAMU3,TOFMU)
      WRITE(IOUT,*)'AMU1',AMU1,'MISFIT',TAMU1
      WRITE(IOUT,*)'AMU2',AMU2,'MISFIT',TAMU2
      WRITE(IOUT,*)'AMU3',AMU3,'MISFIT',TAMU3
C     TOBT = FMIN(AMU1,AMU2,AMU3,TAMU2,TOFMU,TOLM,PMU)
      PMU = AMU2
      TOBT = TAMU2
C  COMPUTE THE MODEL PARAMETERS (AGAIN) AT WHICH MINIMUM MISFIT WAS
C  OBTAINED. TOBDUM IS DUMMY
      TOBDUM=TOFMU(PMU)
      WRITE(IOUT,*) 'MINIMUM TOL IS AT MU =',PMU
      WRITE(IOUT,*) 'AND IS =',TOBT
      WRITE(IOUT,*) 'USING ',NFOR,' EVALUATIONS OF FUNCTION'
C IF THE NEW MINIMUM TOLERANCE IS GREATER THAN THE TOLERANCE FROM THE
C PREVIOUS MODEL, WE ARE HAVING CONVERGENCE PROBLEMS.
C***END LAGRANGE MULTIPLIER SELECTION
C COMPUTE ROUGHNESS
      WK(1) = 0.0
      IF (IRUF .EQ. 1) THEN
        DO 165 I = 2,NP
```

```
      PD (I) = PM(I)
      CVEC(I) = WJTWD(I)
 10   CONTINUE
      CALL QUAD(PD ,NPP,  NPP, NP,HESS,CVEC)
      DO 15 I=1,NP
 15    PM2(I) = PD(I)
C CUT STEP SIZE IF NECESSARY
      IF (FRAC .LT. 0.999) THEN
        DO 30 I = 1,NP
30        PM2(I) = PM(I) + FRAC*(PM2(I)-PM(I))
      END IF
C CALC. MODEL RESPONSE AND MISFIT
      CALL FORMOD(NP,ND,PM2,DM)
      DO 40 I = 1,ND
40      WK(I) = (D(I) - DM(I))/SD(I)
      CHI = ANORM(ND,WK)
      TOFMU = SQRT(CHI/ND)
      RETURN
      END
C
      SUBROUTINE QUAD(PD ,NROWH,NCOLH,N,HESS,CVEC)
      IMPLICIT REAL*8 (A-H,O-Z)
      PARAMETER (ITMAX=50, MSGLVL=1, NROWA=1,
     1  BIGBND=1.E+10,LIWORK=60,LWORK=1960, NCLIN=0, NPP=30)
      REAL*8     A(1,NPP), BL(NPP), BU(NPP), CVEC(N), FEATOL(NPP),
     1    HESS(NROWH,NCOLH), CLAMDA(NPP), WORK(LWORK), IWORK(LIWORK),
     2    ISTATE(NPP), PD(N)
      LOGICAL COLD, LP, ORTHOG
      EXTERNAL QPHESS
      DATA  NOUT/7/
C     CALL X04ABF(1,NOUT)
      COLD=.TRUE.
      LP = .FALSE.
      ORTHOG = .FALSE.
      NCTOTL=N+NCLIN
      DO 10 I=1,N
         BL(I) =0.
         BU(I) = 20.
         FEATOL(I) = 0.02
  10  CONTINUE
      BU(1)=0.
      BU(N)=0.
      IFAIL=0
C  QUADRATIC PROGRAMMING
C     CALL E04NAF(ITMAX,MSGLVL,N,NCLIN,NCTOTL,NROWA,NROWH,NCOLH,
C    1     BIGBND,A,BL,BU,CVEC,FEATOL,HESS,QPHESS,
C    2     COLD,LP,ORTHOG,PD ,ISTATE,ITER,OBJ,
C    .     CLAMDA,IWORK,LIWORK,WORK,LWORK,IFAIL)
      CALL VE04A(N,HESS,NROWH,CVEC,BL,BU,PD,Q,IWORK,K,WORK)
      IF (IFAIL.GT.0) THEN
         WRITE(4,20) IFAIL
         STOP
      ENDIF
 20   FORMAT('E04NAF TERMINATED WITH IFAIL = ', I3)
      RETURN
      END
       SUBROUTINE QPHESS(N, NROWH, NCOLH, JTHCOL, HESS, X, HX)
      IMPLICIT REAL*8 (A-H,O-Z)
      INTEGER JTHCOL, N, NCOLH, NROWH
      REAL*8    HESS(NROWH,NCOLH), HX(N), X(N)
```

```
        DEL(N,N-1) = 0.                                                 INV03600
        DEL(N,N) = 0.                                                   INV03610
        GOTO 20                                                         INV03620
      END IF                                                            INV03630
C SQUARE MATRIX AND COPY BACK INTO DEL IF SECOND DERIV. SMOOTHING REQUIRINV03640
      IF (IRUF .EQ. 2) THEN                                             INV03650
        CALL MULT(NPP,NP,NPP,NP,NP,DEL,DEL,PTP)                         INV03660
        DO 30 I = 1,NP                                                  INV03670
          DO 30 J = 1,NP                                                INV03680
30          DEL(J,I) = PTP(J,I)                                         INV03690
      END IF                                                            INV03700
C FORM TRANS(DEL).DEL                                                   INV03710
      CALL TRMULT(NPP,NP,NPP,NP,NP,DEL,DEL,PTP)                         INV03720
      RETURN                                                            INV03730
      END                                                               INV03740
C                                                                       INV03750
      BLOCKDATA BLOCK                                                   INV03760
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)                              INV03770
      PARAMETER (NDD = 40, NPP = 30)                                    INV03780
      COMMON /RESULT/ PTP(NPP,NPP),WJTWJ(NPP,NPP),WJTWD(NPP),PM2(NPP),  INV03790
     * PMU,NFOR,NITER,FRAC,RLAST,IFFTOL                                 INV03800
      DATA NITER /0/                                                    INV03810
      END                                                               INV03820
C                                                                       INV03830
C                                                                       INV03840
C***********************************************************************INV03850
      FUNCTION TOFMU(AMU)                                               INV03860
C                                                                       INV03870
C FUNCTION TOFMU RETURND THE RMS MISFIT OF THE RESPONSE OF A MODEL CONSTINV03880
C USING THE GIVEN VALUE OF LAGRANGE MULTIPLIER                          INV03890
C S. CONSTABLE, MARCH 1986, S.I.O., LA JOLLA, CA 92093, U.S.A.          INV03900
C                                                                       INV03910
C ON INPUT:                                                             INV03920
C  AMU = LAGRANGE MULTIPLIER                                            INV03930
C THE MODEL IS ALSO A FUNCTION OF THE ARRAYS PASSED IN COMMON BLOCK /RESINV03940
C                                                                       INV03950
C ON OUTPUT                                                             INV03960
C   TOFMU = R.M.S. MISFIT OR 1.0E+10 IF CHOLESKY DECOMPOSITION FAILED   INV03970
C   /MODEL/ CONTAINS THE MODEL WHICH PRODUCES THE MISFIT TOFMU, AS WELL INV03980
C         AS ITS RESPONSE                                               INV03990
C  SUBROUTINES USED:                                                    INV04000
C    CHOLIN, CHOLSL ARE R.L. PARKER'S SUBROUTINES TO PERFORM CHOLESKY DEINV04010
C       AND THEN SOLVE A LINEAR SYSTEM BY BACK SUBSTITUTION.            INV04020
C    ANORM, FORMOD ARE EXPLAINED IN OCCAM1                              INV04030
C                                                                       INV04040
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)                              INV04050
      PARAMETER (NDD = 40, NPP = 30)                                    INV04060
      COMMON /DATA/ ND,D(NDD),SD(NDD)                                   INV04070
      COMMON /MODEL/ NP,PM(NPP),DM(NDD),LJMP(NPP)                       INV04080
      COMMON /RESULT/ PTP(NPP,NPP),WJTWJ(NPP,NPP),WJTWD(NPP),PM2(NPP),  INV04090
     * PMU,NFOR,NITER,FRAC,RLAST,IFFTOL                                 INV04100
      DIMENSION WK(NDD)                                                 INV04110
      REAL*8 PD(NPP),HESS(NPP,NPP),CVEC(NPP)                            INV04120
C                                                                       INV04130
      NFOR = NFOR + 1                                                   INV04140
C ADD AMU.TRANS(PARTIAL).PARTIAL TO TRANS(W.J).W.J                      INV04150
      DO 20 I = 1,NP                                                    INV04160
        DO 20 J = 1,NP                                                  INV04170
20        HESS(I,J) = AMU*PTP(I,J) + WJTWJ(I,J)                         INV04180
      DO 10 I = 1,NP                                                    INV04190
```

```
12    CONTINUE
      RETURN
      END
C
C
C***********************************************************************
      SUBROUTINE TRMULT(MAD,MA,NAD,NA,NB,A,B,C)
C MULTIPLIES THE TRANSPOSE OF A 2D MATRIX BY ANOTHER MATRIX
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION A(MAD,NA),B(MAD,NB),C(NAD,NB)
      DO 12 I = 1,NA
        DO 12 J = 1,NB
          CIJ = 0.0
          DO 10 K = 1,MA
10          CIJ = A(K,I)*B(K,J) + CIJ
12        C(I,J) = CIJ
      RETURN
      END
C
C
C
C***********************************************************************
      SUBROUTINE MINBRK(AX,BX,CX,FA,FB,FC,FUNC)
C
C MINBRK BRACKETS A UNIVARIATE MINIMUM OF A FUNCTION. TO BE USED PRIOR T
C A UNIVARIATE MINIMISATION ROUTINE.
C
C ON INPUT:
C   AX, BX = TWO DISTINCT ESTIMATES OF THE MINIMUM'S WHEREABOUTS
C   FUNC = THE FUNCTION IN QUESTION
C ON OUTPUT:
C   AX,BX,CX = THREE NEW POINTS WHICH BRACKET THE MINIMUM
C   FA,FB,FC = FUNC(AX), FUNC(BX) ETC.
C
C
      IMPLICIT DOUBLE PRECISION (A - H, O - Z)
      PARAMETER (GOLD=1.618034, GLIMIT=100., TINY=1.E-21)
      FA = FUNC(AX)
      FB = FUNC(BX)
      IF(FB.GT.FA)THEN
        DUM = AX
        AX = BX
        BX = DUM
        DUM = FB
        FB = FA
        FA = DUM
      ENDIF
      CX = BX + GOLD*(BX - AX)
      FC = FUNC(CX)
1     IF(FB.GE.FC)THEN
        R = (BX - AX)*(FB - FC)
        Q = (BX - CX)*(FB - FA)
        U = BX-((BX-CX)*Q-(BX-AX)*R)/(2.*SIGN(MAX(ABS(Q-R),TINY),Q-R))
        ULIM = BX + GLIMIT*(CX - BX)
        IF((BX - U)*(U - CX).GT.0.)THEN
          FU = FUNC(U)
          IF(FU.LT.FC)THEN
            AX = BX
            FA = FB
            BX = U
```

```
      INTEGER I,J
      DO 60 I=1,N
 60   HX(I) =0.0
      DO 100 J=1,N
         DO 80 I=1,N
 80     HX(I)=HX(I)+HESS(I,J)*X(J)
 100  CONTINUE
      END
C
C
C
C************************************************************************
      FUNCTION ANORM(N,D)
C RETURNS THE SQUARE OF THE EUCLIDEAN NORM OF A VECTOR
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION D(N)
      ANORM = 0.0
      DO 10 I = 1,N
        ANORM = ANORM + D(I)*D(I)
10    CONTINUE
      RETURN
      END
C
C
C************************************************************************
      SUBROUTINE INITM(MD,M,N,A)
C ZEROS A 2D MATRIX
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION A(MD,N)
      DO 10 I = 1,M
        DO 10 J = 1,N
10        A(I,J) = 0.0
      RETURN
      END
C
C
C************************************************************************
      SUBROUTINE MULT(MAD,MA,NAD,NA,NB,A,B,C)
C MULTIPLIES TWO 2D MATRICES
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION A(MAD,NA),B(NAD,NB),C(MAD,NB)
      DO 12 I = 1,MA
        DO 12 J = 1,NB
          CIJ = 0.0
          DO 10 K = 1,NA
10          CIJ = A(I,K)*B(K,J) + CIJ
12        C(I,J) = CIJ
      RETURN
      END
C
C
C************************************************************************
      SUBROUTINE DIMULT(MAD,MA,NA,DIAG,A,B)
C MULTIPLIES A 2D MATRIX BY A DIAGONAL MATRIX
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION DIAG(MA),A(MAD,NA),B(MAD,NA)
      DO 12 J = 1,NA
        DO 11 I = 1,MA
          B(I,J) = DIAG(I)*A(I,J)
11      CONTINUE
```

```
      V = BX                                                            INV06590
      W = V                                                             INV06600
      X = V                                                             INV06610
      E = 0.                                                            INV06620
      FX = FBX                                                          INV06630
      FV = FX                                                           INV06640
      FW = FX                                                           INV06650
      DO 11 ITER = 1,ITMAX                                              INV06660
        XM = 0.5*(A + B)                                                INV06670
        TOL1 = TOL*ABS(X) + ZEPS                                        INV06680
        TOL2 = 2.*TOL1                                                  INV06690
        IF(ABS(X - XM).LE.(TOL2 - .5*(B - A))) GOTO 3                   INV06700
        IF(ABS(E).GT.TOL1) THEN                                         INV06710
          R = (X - W)*(FX - FV)                                         INV06720
          Q = (X - V)*(FX - FW)                                         INV06730
          P = (X - V)*Q - (X - W)*R                                     INV06740
          Q = 2.*(Q - R)                                                INV06750
          IF(Q.GT.0.) P =  - P                                          INV06760
          Q = ABS(Q)                                                    INV06770
          ETEMP = E                                                     INV06780
          E = D                                                         INV06790
          IF(ABS(P).GE.ABS(.5*Q*ETEMP).OR.P.LE.Q*(A - X).OR.            INV06800
     *        P.GE.Q*(B - X)) GOTO 1                                    INV06810
          D = P/Q                                                       INV06820
          U = X + D                                                     INV06830
          IF(U - A.LT.TOL2 .OR. B - U.LT.TOL2) D = SIGN(TOL1,XM - X)    INV06840
          GOTO 2                                                        INV06850
        ENDIF                                                           INV06860
1       IF(X.GE.XM) THEN                                                INV06870
          E = A - X                                                     INV06880
        ELSE                                                            INV06890
          E = B - X                                                     INV06900
        ENDIF                                                           INV06910
        D = CGOLD*E                                                     INV06920
2       IF(ABS(D).GE.TOL1) THEN                                         INV06930
          U = X + D                                                     INV06940
        ELSE                                                            INV06950
          U = X + SIGN(TOL1,D)                                          INV06960
        ENDIF                                                           INV06970
        FU = F(U)                                                       INV06980
        IF(FU.LE.FX) THEN                                               INV06990
          IF(U.GE.X) THEN                                               INV07000
            A = X                                                       INV07010
          ELSE                                                          INV07020
            B = X                                                       INV07030
          ENDIF                                                         INV07040
          V = W                                                         INV07050
          FV = FW                                                       INV07060
          W = X                                                         INV07070
          FW = FX                                                       INV07080
          X = U                                                         INV07090
          FX = FU                                                       INV07100
        ELSE                                                            INV07110
          IF(U.LT.X) THEN                                               INV07120
            A = U                                                       INV07130
          ELSE                                                          INV07140
            B = U                                                       INV07150
          ENDIF                                                         INV07160
          IF(FU.LE.FW .OR. W.EQ.X) THEN                                 INV07170
            V = W                                                       INV07180
```

```
          FB = FU
          GO TO 1
        ELSE IF(FU.GT.FB)THEN
          CX = U
          FC = FU
          GO TO 1
        ENDIF
        U = CX + GOLD*(CX - BX)
        FU = FUNC(U)
      ELSE IF((CX - U)*(U - ULIM).GT.0.)THEN
        FU = FUNC(U)
        IF(FU.LT.FC)THEN
          BX = CX
          CX = U
          U = CX + GOLD*(CX - BX)
          FB = FC
          FC = FU
          FU = FUNC(U)
        ENDIF
      ELSE IF((U - ULIM)*(ULIM - CX).GE.0.)THEN
        U = ULIM
        FU = FUNC(U)
      ELSE
        U = CX + GOLD*(CX - BX)
        FU = FUNC(U)
      ENDIF
      AX = BX
      BX = CX
      CX = U
      FA = FB
      FB = FC
      FC = FU
      GO TO 1
    ENDIF
    IF(AX.LT.0.) AX=0.
    IF(CX.LT.0.) CX=0.
    RETURN
    END
C
C
C
C*****************************************************************************
      FUNCTION FMIN(AX,BX,CX,FBX,F,TOL,XMIN)
C FMIN RETURNS THE MINIMUM VALUE OF A FUNCTION
C
C ON INPUT:
C  AX,BX,CX = INDEPENDENT VARIABLE WHICH BRACKET THE MINIMUM
C  FBX = F(BX) (USUALLY AVAILABLE FROM THE BRACKETING PROCEDURE)
C  F = FUNCTION IN QUESTION
C  TOL = FRACTIONAL TOLERANCE REQUIRED IN THE INDEPENDENT VARIABLE
C ON OUTPUT:
C  XMIN = ABSCISSA OF MINIMUM
C  FMIN = F(XMIN)
C
C
C
      IMPLICIT DOUBLE PRECISION (A - H, O - Z)
      PARAMETER (ITMAX=100,CGOLD=.3819660,ZEPS=1.0E - 10)
      A = MIN(AX,CX)
      B = MAX(AX,CX)
```

```
      CALL CUBDER(NPT+2,XDEL,ZSURF,XDER)
      DO 10 I=NXT,1,-1
        IF(I.GE.NE .OR. I.LT.NST) THEN
          ZSURF(I)=0.0
          XDER(I)=0.
          AMPDER(I)=1.
          GOTO 10
        ENDIF
       ZSURF(I)=ZSURF(I-NST+1)
       XDER(I)=XDER(I-NST+1)
       AMPDER(I)=SQRT(1.0+XDER(I)*XDER(I))
 10   CONTINUE
C     WRITE(4,50)(ZSURF(I),I=1,NXT)
C     WRITE(4,60)( XDER(I),I=1,NXT)
C     WRITE(4,70)(AMPDER(I),I=1,NXT)
 50   FORMAT(4X,'ZSURF',/,4X,(10F7.3))
 60   FORMAT(4X,' XDER',/,4X,(10F7.3))
 70   FORMAT(4X,'AMPDER',/,4X,(10F7.3))
      RETURN
      END
C
C
      SUBROUTINE HNPRIM(K,KS,ZSURF,XDER,AMPDER,XSYS,ZSYS,NTR)
C ..................................................
C   COMPUTE THE NORMAL COMPONENT OF THE MAGNETIC FIELD
C ..................................................
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      PARAMETER (PI=3.1415926)
      DIMENSION ZSURF(1),XDER(1),AMPDER(1),XSYS(1),ZSYS(1)
      DIMENSION HPN(120,20)
      COMMON /KY/AKY(20)/HN/HPN
      COMMON KYE,NSYS,NST,XDEL,TRDIS
      PI2=PI*2.
      DO 10 J=KS,NTR+KS-1
          M=J-KS+1
          XD=(J-1)*XDEL-XSYS(K)+TRDIS/2.
          ZD=ZSURF(J)-ZSYS(K)
          RHO=SQRT(XD*XD+ZD*ZD)
          D=RHO*RHO*AMPDER(J)*PI2
          IF(D.EQ.0.) PAUSE'BAD HPN'
          IF(NSYS.EQ.1) THEN
           GEOM1=( -XDER(J)*XD+ZD)*XD/D
           GEOM2=(-(XD*XD-ZD*ZD)*XDER(J)+2.0*(XD*ZD))/D
          ELSE
           GEOM1=( -XDER(J)*XD+ZD)*ZD/D
           GEOM2=((ZD*ZD-XD*XD)-XDER(J)*2.0* XD*ZD )/D
          ENDIF
        DO 20 I =1,KYE
           WK=AKY(I)*PI2
           RKY=RHO*WK
                IF(I.GT.1)THEN
                  HPN(M,I)=(GEOM1*WK *BESSK0(RKY)+GEOM2*BESSK1(RKY)
     +                      /RHO)*WK
                ELSE
                  HPN(M,I)=GEOM2/(RHO*RHO)
                ENDIF
 20      CONTINUE
 10   CONTINUE
C      WRITE(4,30)(HPN(J,1),J=1,NTR)
 30   FORMAT(4X,'HPN',/,(5(1PE12.4)))
```

```
          FV = FW                                                       INV07190
          W = U                                                         INV07200
          FW = FU                                                       INV07210
        ELSE IF(FU.LE.FV .OR. V.EQ.X .OR. V.EQ.W) THEN                  INV07220
          V = U                                                         INV07230
          FV = FU                                                       INV07240
        ENDIF                                                           INV07250
      ENDIF                                                             INV07260
11    CONTINUE                                                          INV07270
      WRITE(*,*) 'MAXIMUM ITERATIONS EXCEEDED IN FMIN'                  INV07280
3     XMIN = X                                                          INV07290
      FMIN = FX                                                         INV07300
      RETURN                                                            INV07310
      END                                                               INV07320
C                                                                       INV07330
C                                                                       INV07340
C                                                                       INV07350
C***********************************************************************INV07360
      SUBROUTINE FORMOD(NPT,NRESP,ZH1,CALC)                             INV07370
C                                                                       INV07380
C   ...PURPOSE...                                                       INV07390
C    COMPUTE THE AEM SYSTEM RESPONSE OVER A 2-D ICE-WATER PROFILE       INV07400
C                                                                       INV07410
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)                              INV07420
      PARAMETER (NDD=40,NPP=30,NXT=200,NTRUNK=100,NLOOP=8)              INV07430
      DIMENSION ZSURF(NXT),XDER(NXT),AMPDER(NXT),HPN(120,20)            INV07440
      DIMENSION AKERN(120,120,20),CHAR(120,20)                          INV07450
      DIMENSION XSYS(NDD),ZSYS(NDD),ZH1(NPP),CALC(NDD)                  INV07460
      COMMON /KY/AKY(20)/KER/AKERN/CH/CHAR/HN/HPN                       INV07470
      COMMON /FCM/ZSURF,XSYS,ZSYS,AMPDER,NM,IREC,AKDEL                  INV07480
      COMMON KYE,NSYS,NST,XDEL,TRDIS                                    INV07490
      DO 1 I=1,NPT                                                      INV07500
 1    ZSURF(I)=ZH1(I)                                                   INV07510
      CALL SURF(NST,NPT,NXT,XDEL,ZSURF,XDER,AMPDER)                     INV07520
      K=1                                                               INV07530
 1001 CONTINUE                                                          INV07540
      KS=(XSYS(K)-TRDIS/2.)/XDEL+2-NTRUNK/2                             INV07550
      IF(KS.LE.0 .OR. KS.GE.NXT-NTRUNK/2) PAUSE'BAD INPUT XSYS'         INV07560
      CALL HNPRIM(K,KS,ZSURF,XDER,AMPDER,XSYS,ZSYS,NTRUNK)              INV07570
      CALL KERNEL(K,KS-1,KS1,ZSURF,XDER,AMPDER,NTRUNK)                  INV07580
      CALL CHARGE(NTRUNK,KYE,XDEL,NLOOP)                                INV07590
      CALL FIELD(K,KS,NM,AKDEL,HXS,HZS,XSYS,ZSYS,ZSURF,AMPDER,NTRUNK)   INV07600
      IF(IREC.EQ.1) CALC(K)=HXS                                         INV07610
      IF(IREC.EQ.2) CALC(K)=HZS                                         INV07620
      K=K+1                                                             INV07630
      KS1=KS                                                            INV07640
      IF(K.LE.NRESP) GOTO 1001                                          INV07650
      RETURN                                                            INV07660
      END                                                               INV07670
C                                                                       INV07680
C                                                                       INV07690
C                                                                       INV07700
      SUBROUTINE SURF(NST,NPT,NXT,XDEL,ZSURF,XDER,AMPDER)               INV07710
C ...........................................................           INV07720
C  GENERATE THE INFORMATION OF THE ICE-WATER INTERFACE, I.E.,           INV07730
C  ZSURF, XDER, AMPDER                                                  INV07740
C ...........................................................           INV07750
       IMPLICIT DOUBLE PRECISION (A-H, O-Z)                             INV07760
       DIMENSION ZSURF(NXT),XDER(NXT),AMPDER(NXT)                       INV07770
      NE =NST+NPT                                                       INV07780
```

```
          WK=AKY(I)*PI2                                                INV08990
          IF(I.EQ.1) THEN                                              INV09000
            AKERN(J,L,I)=   GEOM/RHO                                   INV09010
          ELSE                                                         INV09020
            AKERN(J,L,I)=    GEOM*WK*BESSK1(RHO*WK)                    INV09030
          ENDIF                                                        INV09040
   50       CONTINUE                                                   INV09050
   60   CONTINUE                                                       INV09060
           DO 80 J=KE2+1,NTR                                           INV09070
             DO 80 L=1,NTR                                             INV09080
               IF(J.EQ.L) GOTO 80                                      INV09090
               XD=(J-L)*XDEL                                           INV09100
               ZD=ZSURF(J+KS)-ZSURF(L+KS)                              INV09110
               RHO=SQRT(XD*XD+ZD*ZD)                                   INV09120
               GEOM=2.*(AMPDER(L+KS)/AMPDER(J+KS))*( XD*XDER(J+KS)-ZD)/RHOINV09130
               DO 70 I=1,KYE                                           INV09140
               WK=AKY(I)*PI2                                           INV09150
               IF(I.EQ.1) THEN                                         INV09160
                 AKERN(J,L,I)=   GEOM/RHO                              INV09170
               ELSE                                                    INV09180
                 AKERN(J,L,I)=    GEOM*WK*BESSK1(RHO*WK)               INV09190
               ENDIF                                                   INV09200
   70        CONTINUE                                                  INV09210
   80   CONTINUE                                                       INV09220
C       WRITE(4,90)( AKERN(40,10,I)         ,I=1,KYE)                  INV09230
 90     FORMAT(4X,'KERNEL',/,(5(1PE12.4)))                             INV09240
        RETURN                                                         INV09250
        END                                                            INV09260
C                                                                      INV09270
C                                                                      INV09280
        SUBROUTINE CHARGE(NTR,KYE,XDEL,NLOOP)                          INV09290
C     .......................................................          INV09300
C      COMPUTE THE CHARGE DISTRIBUTION ON THE SURFACE OF               INV09310
C      THE SEA WATER                                                   INV09320
C     .......................................................          INV09330
        IMPLICIT DOUBLE PRECISION (A-H, O-Z)                           INV09340
        DIMENSION CHAR(120,20),HPN(120,20),AKERN(120,120,20),W1(120)   INV09350
        DIMENSION W2(120)                                              INV09360
        COMMON /KER/AKERN/HN/HPN/CH/CHAR                               INV09370
        FAC=1.0/(2*3.1415926)                                          INV09380
        I=1                                                            INV09390
        KK=0                                                           INV09400
 1001   CONTINUE                                                       INV09410
        DO 10 J=1,NTR                                                  INV09420
   10     W1(J) =HPN(J,I)*FAC                                          INV09430
  999   CONTINUE                                                       INV09440
        DO 20 J=1,NTR                                                  INV09450
           SUM=0.0                                                     INV09460
           DO 30 L=1,NTR                                               INV09470
             SUM=SUM+W1(L) *AKERN(J,L,I)                               INV09480
   30      CONTINUE                                                    INV09490
           W2(J)=FAC*(-SUM*XDEL+HPN(J,I))                              INV09500
   20   CONTINUE                                                       INV09510
        DO 40 J=1,NTR                                                  INV09520
   40     W1(J)=W2(J)                                                  INV09530
        KK=KK+1                                                        INV09540
        IF(KK.LT.NLOOP) GOTO 999                                       INV09550
        DO 50 J=1,NTR                                                  INV09560
   50      CHAR(J,I)=W1(J)                                             INV09570
        I=I+1                                                          INV09580
```

```
      RETURN                                                            INV08390
      END                                                               INV08400
C                                                                       INV08410
C                                                                       INV08420
      SUBROUTINE KERNEL(K,KS,KS1,ZSURF,XDER,AMPDER,NTR)                 INV08430
C     ..............................................................    INV08440
C    COMPUTE THE KERNEL IN THE INTEGRAL EQUATION                        INV08450
C    AND SAVE IT                                                        INV08460
C     ..............................................................    INV08470
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)                              INV08480
      PARAMETER (PI=3.1415926)                                          INV08490
      DIMENSION ZSURF(1),XDER(1),AMPDER(1)                              INV08500
      DIMENSION AKERN(120,120,20)                                       INV08510
      COMMON /KER/AKERN/KY/AKY(20)                                      INV08520
      COMMON KYE,NSYS,NST,XDEL,TRDIS                                    INV08530
      PI2=PI*2.                                                         INV08540
      DO 1 J=1,NTR                                                      INV08550
        DO 1 I=1,KYE                                                    INV08560
           AKERN(J,J,I)=0.0                                             INV08570
  1   CONTINUE                                                          INV08580
      IF(K.GT.1) GOTO 30                                                INV08590
C  COMPUTE THE AKERNEL                                                  INV08600
 1001 CONTINUE                                                          INV08610
      DO 20 J=1,NTR                                                     INV08620
         DO 20 L=1,NTR                                                  INV08630
             IF(J.EQ.L) GOTO 20                                         INV08640
             XD=(J-L)*XDEL                                              INV08650
             ZD=ZSURF(J+KS)-ZSURF(L+KS)                                 INV08660
             RHO=SQRT(XD*XD+ZD*ZD)                                      INV08670
             GEOM=2.*(AMPDER(L+KS)/AMPDER(J+KS))*( XD*XDER(J+KS)-ZD)/RHOINV08680
             DO 10 I=1,KYE                                              INV08690
             WK=AKY(I)*PI2                                              INV08700
             IF(I.EQ.1) THEN                                            INV08710
               AKERN(J,L,I)=   GEOM/RHO                                 INV08720
             ELSE                                                       INV08730
               AKERN(J,L,I)=    GEOM*WK*BESSK1(RHO*WK)                  INV08740
             ENDIF                                                      INV08750
 10     CONTINUE                                                        INV08760
 20    CONTINUE                                                         INV08770
      RETURN                                                            INV08780
C                                                                       INV08790
C                                                                       INV08800
C   COMPUTE THE UNKNOWN PART OF THE  KERNEL                             INV08810
 30   CONTINUE                                                          INV08820
      M=KS-KS1+1                                                        INV08830
      KE2=NTR-M                                                         INV08840
      IF(M.LE.0) PAUSE 'BAD INPUT XSYS'                                 INV08850
      IF(M.GE.NTR) GOTO 1001                                            INV08860
      DO 40 J=1,KE2                                                     INV08870
          DO 40 L=1,KE2                                                 INV08880
           DO 40 I=1,KYE                                                INV08890
              AKERN(J,L,I)=AKERN(J+M,L+M,I)                             INV08900
 40    CONTINUE                                                         INV08910
      DO 60 J=1,KE2                                                     INV08920
        DO 60 L=KE2+1,NTR                                               INV08930
             XD=(J-L)*XDEL                                              INV08940
             ZD=ZSURF(J+KS)-ZSURF(L+KS)                                 INV08950
             RHO=SQRT(XD*XD+ZD*ZD)                                      INV08960
             GEOM=2.*(AMPDER(L+KS)/AMPDER(J+KS))*( XD*XDER(J+KS)-ZD)/RHOINV08970
             DO 60 I=1,KYE                                              INV08980
```

```
      RETURN
      END
C
C
      SUBROUTINE CUBDER(N,XDEL,YA,YD)
C    ..................................................
C    .  COMPUTE THE DERIVATIVES AT EACH NODE USING
C    .  CUBIC SPLINE.
C    .  NOTE: N=NPT+2 . ONE POINT ADDED TO EACH END.
C    ..................................................
C
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION XA(50),YA(N),Y2(50),YD(N)
      DO 11 I=1,N
 11    XA(I)=(I-1)*XDEL
      DO 12 I=N-1,2,-1
 12    YA(I)=YA(I-1)
      YA(1)=0.
      YA(N)=0.
      CALL SPLINE(XA,YA,N,0.,0.,Y2)
      DO 10 I=2,N-1
        XX=XA(I)
        KLO=I
        KHI=I+1
          H=XA(KHI)-XA(KLO)
          A=(XA(KHI)-XX)/H
          B=(XX-XA(KLO))/H
          XD=XA(KHI)-XA(KLO)
          YD(I-1)=(YA(KHI)-YA(KLO))/XD-XD*((3.*A*A-1.)*Y2(KLO)
     +           -(3.*B*B-1.)*Y2(KHI))/6.
 10   CONTINUE
      DO 14 I=1,N-2
 14    YA(I)=YA(I+1)
      RETURN
      END
C
      SUBROUTINE CUBSPL(KYE,NK1,AKDEL,AKY,YA,O)
C    ..................................................
C    .  CUBIC SPLINE INTERPOLATION BEFORE TAKING THE
C    .  INVERSE FOURIER TRANSFORM.
C    ..................................................
C
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      DIMENSION AKY(KYE), O(NK1),YA(KYE),Y2(20)
      CALL SPLINE(AKY,YA  ,KYE,1.E30,1.E30,Y2)
      O(1)=YA(1)
      DO 10 I=2,NK1
        XX=(I-1)*AKDEL
        KLO=1
        KHI=KYE
  1       IF(KHI-KLO.GT.1) THEN
            K=(KHI+KLO)/2
            IF(AKY(K).GT.XX) THEN
              KHI=K
            ELSE
              KLO=K
            ENDIF
          GOTO 1
          ENDIF
          H=AKY(KHI)-AKY(KLO)
```

```
      IF(I.LE.KYE) GOTO 1001                                               INV09590
C     WRITE(4,22)(CHAR(J,1),J=1,NTR)                                       INV09600
 22   FORMAT(4X,'CHARGE',/,(5E14.4))                                       INV09610
      RETURN                                                               INV09620
      END                                                                  INV09630
C                                                                          INV09640
C                                                                          INV09650
C                                                                          INV09660
      SUBROUTINE FIELD(K,KS,NM,AKDEL,HXS,HZS,XSYS,ZSYS,ZSURF,AMPDER,NTR)INV09670
C .................................................                        INV09680
C .     COMPUTE THE SECONDARY MAGNETIC FIELD, BOTH                         INV09690
C .     X AND Z COMPONENTS, EXPRESSED IN PPM OF THE                        INV09700
C .     RECEIVED PRIMARY FIELD.                                            INV09710
C .................................................                        INV09720
C                                                                          INV09730
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)                                 INV09740
      PARAMETER (PI=3.1415926)                                             INV09750
      DIMENSION AMPDER(1),CHAR(120,20),ZSURF(1),XSYS(1),ZSYS(1),AA(513) INV09760
      DIMENSION TEMP1(20),TEMP2(20)                                        INV09770
      COMMON /CH/CHAR/KY/AKY(20)                                           INV09780
      COMMON KYE,NSYS,NST,XDEL,TRDIS                                       INV09790
      R =TRDIS                                                             INV09800
      XF=R/2.                                                              INV09810
       IF(NSYS.EQ.1) THEN                                                  INV09820
          HNORM=1./(2.*PI*R**3)                                            INV09830
       ELSE                                                                INV09840
          HNORM=-1./(4.*PI*R**3)                                           INV09850
       ENDIF                                                               INV09860
      DO 10 I=1,KYE                                                        INV09870
        WK=AKY(I)*PI*2.                                                    INV09880
        AZ=0.0                                                             INV09890
        AX=0.0                                                             INV09900
        DO 20 J=KS,NTR+KS-2                                                INV09910
          M=J-KS+1                                                         INV09920
          RHO=SQRT((XSYS(K)-(J-1)*XDEL+XF)**2+(ZSYS(K)-ZSURF(J))**2)       INV09930
          IF(I.GT.1) THEN                                                  INV09940
             AXZ=2.0*CHAR(M,I)*AMPDER(J)*WK*BESSK1(RHO*WK)/RHO             INV09950
          ELSE                                                             INV09960
             AXZ=2.0*CHAR(M,I)*AMPDER(J)/(RHO*RHO)                         INV09970
          ENDIF                                                            INV09980
           AZ=(ZSYS(K)-ZSURF(J))*AXZ+AZ                                    INV09990
           AX=(XSYS(K)-(J-1)*XDEL+XF)*AXZ+AX                               INV10000
 20     CONTINUE                                                           INV10010
        TEMP1(I)=AZ*XDEL                                                   INV10020
        TEMP2(I)=AX*XDEL                                                   INV10030
 10   CONTINUE                                                             INV10040
      NK =2**N'I                                                           INV10050
      IF(NK*AK EL .GT. AKY(KYE)) PAUSE 'BAD AKDEL OR AKY'                  INV10060
      CALL CUB.PL(KYE,NK,AKDEL,AKY,TEMP1,AA)                               INV10070
      SUM=0.0                                                              INV10080
      DO 11 I=2,NK                                                         INV10090
 11     SUM=SUM+AA(I)                                                      INV10100
      HZS=(SUM AA(1)/2.)*AKDEL*2.                                          INV10110
      CALL CUB PL(KYE,NK,AKDEL,AKY,TEMP2,AA)                               INV10120
      SUM=0.0                                                              INV10130
      DO 12 I=2,NK                                                         INV10140
 12     SUM=SUM+AA(I)                                                      INV10150
      HXS=(SUM+AA(1)/2.)*AKDEL*2.                                          INV10160
      HXS=HXS/HNORM*1000000                                                INV10170
      HZS=HZS/HNORM*1000000                                                INV10180
```

```fortran
      if(abs(x).lt.3.75) then
        y=(x/3.75)**2
        BESSI0=p1+y*(p2+y*(p3+y*(p4+y*(p5+y*(p6+y*p7)))))
      else
        ax=abs(x)
        y=3.75/ax
        BESSI0=(exp(ax)/sqrt(ax))*(q1+y*(q2+y*(q3+y*(q4+y*(q5+
     +    y*(q6+y*(q7+y*(q8+y*q9))))))))
      endif
      return
      end
c
c
      Function BESSK0(x)
C     ......................................................
c ...  Returns the modified Bessel function K0(x) for positive x.
C     ......................................................
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      real*8 y,p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7
      data p1,p2,p3,p4,p5,p6,p7/-0.57721566d0,0.42278420d0,0.23069756d0,
     +     0.3488590d-1,0.262698d-2,0.10750d-3,0.74d-5/
      data q1,q2,q3,q4,q5,q6,q7/1.25331414d0,-0.7832358d-1,0.2189568d-1,
     +     -0.1062446d-1,0.587872d-2,-0.251540d-2,0.53208d-3/
      if(x.le.2.0) then
        y=x*x/4.0
        BESSK0=(-log(x/2.0)*BESSI0(x))+(p1+y*(p2+y*(p3+
     +            y*(p4+y*(p5+y*(p6+y*p7))))))
      else
        if(x.gt.40.) then
           BESSK0=0.0
        else
           y=(2.0/x)
           BESSK0=(exp(-x)/sqrt(x))*(q1+y*(q2+y*(q3+y*(q4+y*(q5+
     +            y*(q6+y*q7))))))
        endif
      endif
      return
      end
c
c
      Function BESSI1(x)
C     ................................................................
c ...  Returns the modified Bessel function I1(x) for any real x.
C     ................................................................
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      real*8 y,p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,q8,q9
      data p1,p2,p3,p4,p5,p6,p7/0.5d0,0.87890594d0,0.51498869d0,
     +  0.15084934d0,0.2658733d-1,0.301532d-2,0.32411d-3/
      data q1,q2,q3,q4,q5,q6,q7,q8,q9/0.39894228d0,-0.3988024d-1,
     +  -0.362018d-2,0.163801d-2,-0.1031555d-1,0.2282967d-1,
     +  -0.289531d-1,0.1787654d-1,-0.420059d-2/
      if(abs(x).lt.3.75) then
        y=(x/3.75)**2
        BESSI1=x*(p1+y*(p2+y*(p3+y*(p4+y*(p5+y*(p6+y*p7))))))
      else
        ax=abs(x)
        y=3.75/ax
        BESSI1=(exp(ax)/sqrt(ax))*(q1+y*(q2+y*(q3+y*(q4+y*(q5+
     +    y*(q6+y*(q7+y*(q8+y*q9))))))))
      endif
```

```fortran
         A=(AKY(KHI)-XX)/H
         B=(XX-AKY(KLO))/H
         O(I)=A*YA(KLO)+B*YA(KHI)+(A*(A*A-1.)*Y2(KLO)+
     +        B*(B*B-1.)*Y2(KHI))*(H*H)/6.
 10   CONTINUE
      RETURN
      END
C
      SUBROUTINE SPLINE(X,Y,N,YP1,YPN,Y2)
C  ..................................................................
C  . GIVEN ARRAYS X AND Y OF LENGHT N CONTAINING A TABULATED FUNCTION,
C  . I.E., YJ=F(XJ), WITH X1<X2<X3...<XN, THIS ROUTINE RETURNS AN ARRAY
C  . Y2 OF LENGTH N WHICH CONTAINS THE SECOND DERIVATIVES OF THE INTER-
C  . POLATING FUNCTION AT THE TABULATED POINTS XJ. NATURAL SPLINE FROM
C  . NUMERICAL RECIPES, PRESS AT AL.
C  ..................................................................
C
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      PARAMETER (NMAX=1024)
      DIMENSION X(N),Y(N),Y2(N),U(NMAX)
      IF (YP1.GT..99E30) THEN
        Y2(1)=0.
        U(1)=0.
      ELSE
        Y2(1)=-0.5
        U(1)=(3./(X(2)-X(1)))*((Y(2)-Y(1))/(X(2)-X(1))-YP1)
      ENDIF
      DO 11 I=2,N-1
        SIG=(X(I)-X(I-1))/(X(I+1)-X(I-1))
        P=SIG*Y2(I-1)+2.
        Y2(I)=(SIG-1.)/P
        U(I)=(6.*((Y(I+1)-Y(I))/(X(I+1)-X(I))-(Y(I)-Y(I-1))
     *      /(X(I)-X(I-1)))/(X(I+1)-X(I-1))-SIG*U(I-1))/P
 11   CONTINUE
      IF (YPN.GT..99E30) THEN
        QN=0.
        UN=0.
      ELSE
        QN=0.5
        UN=(3./(X(N)-X(N-1)))*(YPN-(Y(N)-Y(N-1))/(X(N)-X(N-1)))
      ENDIF
      Y2(N)=(UN-QN*U(N-1))/(QN*Y2(N-1)+1.)
      DO 12 K=N-1,1,-1
        Y2(K)=Y2(K)*Y2(K+1)+U(K)
 12   CONTINUE
      RETURN
      END
C
C
      Function BESSI0(x)
C  ..............................................................
C  .   RETURNS THE MODIFIED BESSEL FUNCTION I0(X) FOR ANY REAL X.
C  ..............................................................
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      real*8 y,p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7,q8,q9
      data p1,p2,p3,p4,p5,p6,p7/1.0d0,3.5156229d0,3.0899424d0,
     + 1.2067492d0,0.2659732d0,0.360768d-1,0.45813d-2/
      data q1,q2,q3,q4,q5,q6,q7,q8,q9/0.39894228d0,0.1328592d-1,
     + 0.225319d-2,-0.157565d-2,0.916281d-2,-0.2057706d-1,
     + 0.2635537d-1,-0.1647633d-1,0.392377d-2/
```

```
      IOUT=0                                                            INV12590
      DEL=0.                                                            INV12600
      DO 21 I=K1,N                                                      INV12610
      LI=LT(I)                                                          INV12620
      IF(X(LI).EQ.BL(LI).AND.G(I).GE.0.)GOTO21                          INV12630
      IF(X(LI).EQ.BU(LI).AND.G(I).LE.0.)GOTO21                          INV12640
      IF(G(I).LT.0.)GOTO22                                              INV12650
      Z=X(LI)-BL(LI)                                                    INV12660
      J=1                                                               INV12670
      GOTO23                                                            INV12680
   22 CONTINUE                                                          INV12690
      Z=BU(LI)-X(LI)                                                    INV12700
      J=0                                                               INV12710
   23 CONTINUE                                                          INV12720
      IF(G(ICAC+I).LE.0.)GOTO24                                         INV12730
      BETA=ABS(G(I))/G(ICAC+I)                                          INV12740
      IF(BETA.GE.Z)GOTO24                                               INV12750
      Z=BETA                                                            INV12760
      D=.5*Z*ABS(G(I))                                                  INV12770
      J=-1                                                              INV12780
      GOTO26                                                            INV12790
   24 CONTINUE                                                          INV12800
      D=Z*(ABS(G(I))-.5*Z*G(ICAC+I))                                    INV12810
   26 CONTINUE                                                          INV12820
      IF(D.LT.DEL)GOTO21                                                INV12830
      DEL=D                                                             INV12840
      ALPHA=Z                                                           INV12850
      IOUT=I                                                            INV12860
      IIN=I                                                             INV12870
      IF(J.LT.0)IIN=0                                                   INV12880
      LB=J                                                              INV12890
   21 CONTINUE                                                          INV12900
      IF(IOUT.NE.0)GOTO29                                               INV12910
   27 CONTINUE                                                          INV12920
      Q=0.                                                              INV12930
      DO 28 I=1,N                                                       INV12940
      LI=LT(I)                                                          INV12950
   28 Q=Q+X(LI)*(G(I)-B(LI))                                            INV12960
      Q=.5*Q                                                            INV12970
      RETURN                                                            INV12980
   29 CONTINUE                                                          INV12990
      SIG=1.                                                            INV13000
      IF(G(IOUT).GT.0.)SIG=-1.                                          INV13010
      LIOUT=LT(IOUT)                                                    INV13020
      LIIN=LIOUT                                                        INV13030
   25 CONTINUE                                                          INV13040
      SAS=G(ICAC+IOUT)                                                  INV13050
      IF(K.EQ.0)GOTO31                                                  INV13060
      DO 30 I=1,K                                                       INV13070
   30 G(IS+I)=G(ID+I)*A(IOUT,I)                                         INV13080
   31 CONTINUE                                                          INV13090
      DO 37 I=K1,N                                                      INV13100
      LI=LT(I)                                                          INV13110
      IF(LI-LIOUT)32,37,33                                              INV13120
   32 Z=A(LI,LIOUT)                                                     INV13130
      GOTO34                                                            INV13140
   33 Z=A(LIOUT,LI)                                                     INV13150
   34 CONTINUE                                                          INV13160
      IF(K.EQ.0)GOTO36                                                  INV13170
      DO 35 J=1,K                                                       INV13180
```

```
      return
      end
c
c
      Function BESSK1(x)
C .....................................................................
c ...  Returns the modified Bessel function K1(x) for positive x.
C .....................................................................
      IMPLICIT DOUBLE PRECISION (A-H, O-Z)
      real*8 y,p1,p2,p3,p4,p5,p6,p7,q1,q2,q3,q4,q5,q6,q7
      data p1,p2,p3,p4,p5,p6,p7/1.0d0,0.15443144d0,-0.67278579d0,
     +     -0.18156897d0,-0.1919402d-1,-0.110404d-2,-0.4686d-4/
      data q1,q2,q3,q4,q5,q6,q7/1.25331414d0,0.23498619d0,-0.3655620d-1,
     +     0.1504268d-1,-0.780353d-2,0.325614d-2,-0.68245d-3/
      if(x.le.2.0) then
        y=x*x/4.0
        BESSK1= (log(x/2.0)*BESSI1(x))+(1.0/x)*(p1+y*(p2+y*(p3+
     +              y*(p4+y*(p5+y*(p6+y*p7))))))
      else
        if(x.gt.40) then
          BESSK1=0.0
        else
          y=(2.0/x)
          BESSK1=(exp(-x)/sqrt(x))*(q1+y*(q2+y*(q3+y*(q4+y*(q5+
     +              y*(q6+y*q7))))))
        endif
      endif
      return
      end
      SUBROUTINE VE04A(N,A,IA,B,BL,BU,X,Q,LT,K,G)
      IMPLICIT REAL*8 (A-H,O-Z)
C
C     QUADRATIC PROGRAMMING FROM HARWELL LIBRARY.
C     SIMPLE BOUND CONSTRAINTS.
C
      DIMENSION A(IA,1),B(1),BL(1),BU(1),X(1),LT(1),G(1)
      IS=N
      IAS=N
      IV=N
      ICAC=N+N
      ID=ICAC
      DO 9 I=1,N
    9 G(I)=-B(I)
      DO 10 I=1,N
      X(I)=0.
      LT(I)=I
      G(ICAC+I)=A(I,I)
      IF(0..GE.BL(I).AND.0..LE.BU(I))GOTO10
      IF(0..LT.BL(I))X(I)=BL(I)
      IF(0..GT.BU(I))X(I)=BU(I)
      DO 12 J=1,I
   12 G(J)=G(J)+A(J,I)*X(I)
      IF(I.EQ.N)GOTO10
      II=I+1
      DO 11 J=II,N
   11 G(J)=G(J)+A(I,J)*X(I)
   10 CONTINUE
      K=0
      K1=1
   20 CONTINUE
```

```
      VD=V/G(ID+I1)
      S1=SØ+V*VD
      R=S1/SØ
      G(ID+I)=G(ID+I1)*R
      BETA=VD/S1
      IF(R.GT.4.)GOTO841
      DO 81 J=I2,N
   81 G(IV+J)=G(IV+J)-V*A(J,I1)
      IF(I1.GT.K2)GOTO83
      DO 82 J=I1,K2
      J1=J+1
   82 A(J,I)=A(J1,I1)+BETA*G(IV+J1)
   83 CONTINUE
      A(K,I)=BETA
      DO 84 J=K1,N
   84 A(J,I)=A(J,I1)+BETA*G(IV+J)
      GOTO849
  841 CONTINUE
      IF(I1.GT.K2)GOTO843
      DO 842 J=I1,K2
      J1=J+1
  842 A(J,I)=BETA*G(IV+J1)+A(J1,I1)/R
  843 CONTINUE
      A(K,I)=BETA
      DO 844 J=K1,N
  844 A(J,I)=BETA*G(IV+J)+A(J,I1)/R
      DO 845 J=I2,N
  845 G(IV+J)=G(IV+J)-V*A(J,I1)
  849 CONTINUE
      LT(I)=LT(I1)
      SØ=S1
      I1=I2
   85 I2=I2+1
      SG=1./S1
      LT(K)=LIIN
      G(ID+K)=SG
      IF(IIN.EQ.1)GOTO851
      II=IIN-1
      DO 852 I=1,II
      Z=A(IIN,I)
      DO 853 J=IIN,K2
  853 A(J,I)=A(J+1,I)
  852 A(K,I)=Z
  851 CONTINUE
   86 CONTINUE
      DO 87 I=K1,N
   87 G(ICAC+I)=G(ICAC+I)+SG*G(IV+I)**2
      K1=K
      K=K2
      IIN=Ø
      ALPHA=1E75
      SAS=G(ICAC+IOUT)
      IF(SAS.GT.Ø.)ALPHA=ABS(G(IOUT))/SAS
      IF(G(IOUT).LT.Ø.)GOTO898
      J=1
      Z=X(LIOUT)-BL(LIOUT)
      GOTO899
  898 CONTINUE
      J=Ø
      Z=BU(LIOUT)-X(LIOUT)
```

```
36 Z=Z-A(I,J)*G(IS+J)                                    INV13190
36 G(IS+I)=Z                                             INV13200
37 CONTINUE                                              INV13210
   G(IS+IOUT)=SAS                                        INV13220
   IF(K.EQ.0)GOTO42                                      INV13230
   G(IS+K)=-A(IOUT,K)                                    INV13240
   IF(K.EQ.1)GOTO42                                      INV13250
   I=K                                                   INV13260
   DO 41 II=2,K                                          INV13270
   I=I-1                                                 INV13280
   Z=-A(IOUT,I)                                          INV13290
   I1=I+1                                                INV13300
   DO 40 J=I1,K                                          INV13310
40 Z=Z-G(IS+J)*A(J,I)                                    INV13320
41 G(IS+I)=Z                                             INV13330
42 CONTINUE                                              INV13340
   IF(SIG.EQ.1.)GOTO61                                   INV13350
   DO 60 I=1,N                                           INV13360
60 G(IS+I)=-G(IS+I)                                      INV13370
61 CONTINUE                                              INV13380
   IF(K.EQ.0)GOTO82                                      INV13390
   DO 81 I=1,K                                           INV13400
   IF(G(IS+I).EQ.0.)GOTO81                               INV13410
   LI=LT(I)                                              INV13420
   J=1                                                   INV13430
   Z=BL(LI)-X(LI)                                        INV13440
   IF(G(IS+I).LT.0.)GOTO80                               INV13450
   J=0                                                   INV13460
   Z=BU(LI)-X(LI)                                        INV13470
30 CONTINUE                                              INV13480
   Z=Z/G(IS+I)                                           INV13490
   IF(Z.GE.ALPHA)GOTO81                                  INV13500
   ALPHA=Z                                               INV13510
   LB=J                                                  INV13520
   IIN=I                                                 INV13530
   LIIN=LI                                               INV13540
81 CONTINUE                                              INV13550
82 CONTINUE                                              INV13560
   X(LIOUT)=X(LIOUT)+SIG*ALPHA                           INV13570
   IF(K.EQ.0)GOTO71                                      INV13580
   DO 70 I=1,K                                           INV13590
   LI=LT(I)                                              INV13600
70 X(LI)=X(LI)+ALPHA*G(IS+I)                             INV13610
71 CONTINUE                                              INV13620
   DO 72 I=K1,N                                          INV13630
72 G(I)=G(I)+ALPHA*G(IAS+I)                              INV13640
   IF(IIN.EQ.0)GOTO90                                    INV13650
   X(LIIN)=BL(LIIN)                                      INV13660
   IF(LB.EQ.0)X(LIIN)=BU(LIIN)                           INV13670
   IF(IIN.EQ.IOUT)GOTO20                                 INV13680
   K2=K-1                                                INV13690
   SG=G(ID+IIN)                                          INV13700
   I1=IIN+1                                              INV13710
   DO 80 I=I1,N                                          INV13720
80 G(IV+I)=A(I,IIN)                                      INV13730
   IF(IIN.EQ.K)GOTO86                                    INV13740
   I2=IIN+2                                              INV13750
   S0=1./SG                                              INV13760
   DO 86 I=IIN,K2                                        INV13770
   V=G(IV+I1)                                            INV13780
```

```
      DC 100 L=I1,J1                                                    INV14990
  100 Z=Z+A(J,L)*A(L,I)                                                 INV15000
  101 A(J,I)=-Z                                                         INV15010
  102 CONTINUE                                                          INV15020
      AA=1./G(ID+I1)                                                    INV15030
      G(N+I1)=AA                                                        INV15040
      DO 111 J=1,I                                                      INV15050
      Z=A(I1,J)*AA                                                      INV15060
      G(N+J)=G(N+J)+Z*A(I1,J)                                           INV15070
      IF(I.EQ.1)GOTO111                                                 INV15080
      J1=J+1                                                            INV15090
      DO 110 L=J1,I                                                     INV15100
  110 A(L,J)=A(L,J)+A(I1,L)*Z                                           INV15110
  111 A(I1,J)=Z                                                         INV15120
      RETURN                                                            INV15130
      END                                                               INV15140
```

```
  899 CONTINUE                                                          INV14390
      IF(Z.GE.ALPHA)GOTO26                                              INV14400
      ALPHA=Z                                                           INV14410
      LB=J                                                              INV14420
      IIN=IOUT                                                          INV14430
      LIIN=LIOUT                                                        INV14440
      GOTO26                                                            INV14450
   90 CONTINUE                                                          INV14460
      K2=K1+1                                                           INV14470
      IF(SIG.EQ.1.)GOTO91                                               INV14480
      DO 901 I=K1,N                                                     INV14490
  901 G(IAS+I)=-G(IAS+I)                                                INV14500
   91 CONTINUE                                                          INV14510
      IF(IOUT.EQ.K1)GOTO97                                              INV14520
      LT(IOUT)=LT(K1)                                                   INV14530
      LT(K1)=LIOUT                                                      INV14540
      G(IAS+IOUT)=G(IAS+K1)                                             INV14550
      G(ICAC+IOUT)=G(ICAC+K1)                                           INV14560
      G(ICAC+K1)=SAS                                                    INV14570
      G(IOUT)=G(K1)                                                     INV14580
      IF(K.EQ.0)GOTO97                                                  INV14590
      DO 92 I=1,K                                                       INV14600
      Z=A(K1,I)                                                         INV14610
      A(K1,I)=A(IOUT,I)                                                 INV14620
   92 A(IOUT,I)=Z                                                       INV14630
   93 CONTINUE                                                          INV14640
      IF(K2.EQ.IOUT)GOTO95                                              INV14650
      I1=IOUT-1                                                         INV14660
      DO 94 I=K2,I1                                                     INV14670
   94 A(IOUT,I)=A(I,K1)                                                 INV14680
   95 CONTINUE                                                          INV14690
      IF(IOUT.EQ.N)GOTO97                                               INV14700
      I1=IOUT+1                                                         INV14710
      DO 96 I=I1,N                                                      INV14720
   96 A(I,IOUT)=A(I,K1)                                                 INV14730
   97 CONTINUE                                                          INV14740
      G(K1)=0.                                                          INV14750
      K=K1                                                              INV14760
      IF(K.EQ.N)GOTO27                                                  INV14770
      DO 98 I=K2,N                                                      INV14780
      Z=G(IAS+I)/SAS                                                    INV14790
      A(I,K1)=Z                                                         INV14800
   98 G(ICAC+I)=G(ICAC+I)-Z*G(IAS+I)                                    INV14810
      K1=K2                                                             INV14820
      GOTO20                                                            INV14830
C                                                                       INV14840
      ENTRY VE04B(N,A,IA,G,K)                                           INV14850
      IF(K.EQ.0)RETURN                                                  INV14860
      ID=N+N                                                            INV14870
      G(N+1)=1./G(ID+1)                                                 INV14880
      IF(K.EQ.1)RETURN                                                  INV14890
      N1=K-1                                                            INV14900
      DO 111 I=1,N1                                                     INV14910
      I1=I+1                                                            INV14920
      A(I1,I)=-A(I1,I)                                                  INV14930
      IF(I.EQ.N1)GOTO102                                                INV14940
      II=I+2                                                            INV14950
      DO 101 J=II,K                                                     INV14960
      Z=A(J,I)                                                          INV14970
      J1=J-1                                                            INV14980
```

Introduction

The utilization of downhole current sources in resistivity mapping increases the resolution for detecting and delineating subsurface features. The effects of near surface inhomogeneities are immensely reduced as shown by Asch and Morrison (1988). Being sensitive to changes in resistivities, the surveys with downhole sources are well suited for monitoring surface processes such as injection or leakage of contaminants from a waste site, steam flooding for enhanced oil recovery, or production of geothermal reservoirs.

In most of these applications, the holes are steel-cased and the casing distorts the current flow in the medium. Holladay and West (1984) has shown the surface resistivity surveys are strongly affected by casings. Also, Kauahikaua et al (1980) showed that if the casing itself is used as an electrode, the results are unpredictable because the current leaves the pipe irregularly due to the variability of the contact resistance between the pipe and the formation.

To study the casing effects in more detail, we have recently formulated the problem for a point source of current, either inside or outside the pipe, on the axis of a finite length metal pipe in a conductive half space. The first part of this study (Schenkel and Morrison, 1987) showed that only the region very near the pipe exerts any substantial influence on the potential fields for a point source 100 casing diameters beyond the end of the pipe (Figure 1). For a 5% or less field distortion, the surface measurements must not be closer than 1/2 the pipe's length. In cross-hole surveys, the affected area is greatly reduced; near the pipe's end, measurements as close as 1/6 the length of the pipe for a 5% or less distortion. If the pipe-source separation is sufficiently large, then the resistivity survey can be corrected for the casing effects (assuming that the target are not too close to the pipe).

This study showed that cross-hole and hole-surface studies may be conducted with very little effects; even when the pipe exerts some influence, e.g., when the current source is close to the end, time monitoring experiments could be carried out with very little reduction in the signal strength. Further, this study revealed the intriguing possibility that segments of pipe, separated by insulated couplings, could be used as electrodes.

Proposed Work

The above study was only developed for a very simple case, i.e., a pipe in a homogeneous half-space. A more complex model is required to simulate field situations. Several aims are proposed to create a realistic model and to evaluate field measurements for downhole sources in steel cased wells. These objectives are:

1) To determine the effects of contact resistance between the pipe and the host medium. Contact resistance is used to describe the resistance of the pipe-medium contact. If there is a large contact resistance, the results will completely change. The currents in the pipe will only leak out of the pipe in areas where the pipe has made a good contact with the host medium, thus completely changing the potential fields around the pipe. The contact resistance may be found by assuming a very thin layer between the host and pipe for each segment (Figure 2). An equivalent layer can be used to represent the effects of the contact resistance for each segement. The calculated potential field is obtained from the integral equation solution of the pipe variables. With the additional layer included to the model, the effects of pipe coating, corrosion, and cement on the potential fields can be evaluated.

2) To use insulated pipe segements as downhole sources and receivers. Downhole current electrodes can be created by energizing isolated segements. Likewise, insulated segements could be used as potential electrodes. By insulating several segments in the well (Figure 3), multiple

downhole source locations could be used to image a target in a hole-surface survey. A pole-pole configuration can be acheived by attaching to different segments current and potential electrodes. The isolated segments in the well can be used for an AC vertical electric dipole. By attaching to adjacent segments a positive and negative AC source, a grounded electric dipole can be produced to study EM properties of the medium. If an additional well is drilled with multiple isolated segments, then cross-hole DC and AC measurements can be acquired. Thus, cross-hole DC tomography (Daily and Morely, 1988 and Shima and Saito, 1988) could be used to reconstruct an image of a target between the two wells.

3) To determine the interaction between the source, pipe, and the anomalous body (Figure 4). To monitor the changes in the body, the effects of the metal pipe-body interaction must be investigated. The extent of this interaction will greatly depend on the source separation from the pipe, the distance between the pipe and the body, and the conductivity of the body. The determination of the limiting values of these parameters in which the body and pipe has very little coupling will be the main objective. For this situation, the body can be modeled alone saving computational time.

4) To invert for the geometric parameters for a plume-like body. An integral equation solution of an ellipsoidal model will be used to represent the plume. The parameters of the three axial lengths will be obtained by a non-linear least squares inversion which will make use of the integral equation solution. Sensitivity analysis and minimum spatial coverage will be evaluated for various acquisition array configurations.

Computer models will be required to investigate the above proposed tasks. The current algorithm is flexible so that variable source locations, segment lengths, and segment conductivities can be used. The development of an algorithm which includes an anomalous body and an outer layer is needed. The current integral equation solution can be extented to include an additional layer and a circular disk. The circular disk is a first order approximation to a plume and will give an estimate of the pipe-body interaction.

Various lengths and separations of the insulated segments will be investigated to determine when a point approximation of the segments can be used. The outer layer will be used to evaluate the role of contact resistance on the field distortion and may be included to the pipe-body coupling phenomenon. The spatial separation of the source, pipe, and body to decouple the pipe and the body will be studied.

Field test and model verification are also required. The test would be conducted at U.C. Berkeley, Richmond Field Station where there exist several plastic-cased holes. Four additional 100 foot holes are also needed. Two of the wells will be composed of alternating steel and fiberglass segments. Another would be cased with steel with the last 20 feet perforated. The last well needs to be plastic cased and perforated at the bottom. A surface grid and/or radial arrays of potential electrodes would be installed over the area. Cross-hole and hole-surface measurements would be compared to calculated fields of various pipe models. An injection of the salt water in both the steel-cased and plastic-cased wells would be measured for hole-surface, cross-hole, and pole-pole configurations. These data would be forward modeled and inverted to determine the geometry of the plume. Lastly, other configurations for which models have been published can be field tested with this field setup.

REFERENCES

Asch, T., and Morrison, H.F., 1988, Mapping and monitoring electrical resistivity with surface and subsurface electrode arrays: (Submitted to *Geophysics*).

Daily, W. and Yorkey, T.J., 1988, Evaluation of cross-borehole resistivity tomography: Presented at the *58th Annual International SEG* meeting in Anaheim.

Holladay, J.S. and West, G.F., 1984, Effect of well casings on surface electrical surveys: *Geophysics*, **49** , p. 177-188.

Kauahikaua, J., Mattice, M., and Jackson, D., 1980, Mise-a-la-masse mapping of the HGP-A geothermal reservoir, Hawaii: *Geothermal Resource Council, Transactions*, **4** , p. 65-68.

Schenkel, C.J. and Morrison, H.F., 1988, Effects of the well casing on potential field measurements: (Submitted to *Geoexploration*).

Shima, H. and Saito, H., 1988, Application of resistivity tomography for detection of faults and evaluation of their hydraulic continuity: some numerical experiments: Presented at the *58th Annual International SEG* meeting in Anaheim.