GENERALIZED LINEAR INVERSION OF REFLECTION SEISMIC DATA

by

Dennis A. Cooke

T-2459

A thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirement for the degree of Master of Science in Geophysics.

Golden, Colorado

Date 12/4/81

Signed: Dennis A. Cooke

Dennis A. Cooke

Approved: William A. Schneider

Dr. William A. Schneider

Thesis Advisor

Golden, Colorado

Date 4 Dec '81

Frank A. Hadsell

Dr. Frank A. Hadell

Associate Head of Geophysics Dept.

ABSTRACT

Generalized linear inversion, sometimes known as model perturbation, nonlinear regression or inverse modeling, is applied to synthetic and real seismic data sets with the objective of obtaining an impedance profile as a function of time. The impedances solved for are parameterized in a manner that describes the unknown earth using fewer variables than previous seismic generalized linear inversion techinques. In this application only single traces of CDP processed data will be inverted. The method of generalized linear inversion (G.L.I.) presented is designed to improve on the shortcomings of recursive inversion with respect to relative and absolute scale of the impedance results, resolution of impedance boundaries, and distortion from residual wavelet effects. In obtaining these goals other advantageous aspects of G.L.I. were discovered. For example, it is insensitive to noise in many cases, and it will allow an interpreter to fix the impedance of any number of known lithologies in an interval being inverted. This last property is extremely useful when evaluating a prospect on an otherwise well understood seismic line. The G.L.I. method is illustrated on a number of synthetic examples and one field data set from the Powder River Basin of Wyoming.

CONTENTS

LIST OF FIGURES

ACKNOWLEDGMENTS

This work has been carried out at the Colorado School of Mines with the financial support of the Integrated Geophysics Project. The field data and impedance log used were supplied by Amoco Production Company.

INTRODUCTION

Seismic inversion is the calculation of the earth's structure and physical parameters from some set of observed seismic data. In this paper the data to be inverted are single common depth point stacked seismic reflection traces. The inversion results are single traces of impedance versus depth (measured in units of two way travel time).

The technique used to obtain these results is generalized linear inversion (G.L.I.). Parker (1977) has formalized this technique, and Backus and Gilbert (1968, 1970) have discussed its resolving power and uniqueness. This technique has been used in a similar setting by Gjoystdal and Ursin (1981) to simultaneously invert and migrate 3-D seismic data. Wiggins (1972, 1976) used G.L.I. to invert earthquake data and to solve seismic statics problems.

The G.L.I. technique should be conceptually familiar to most seismic interpreters. It is an automation of the interpretational technique of finding a hypothetical earth cross-section whose response accounts for (is identical to) the data being analyzed. In such a case it is assumed that the data was collected from an earth whose structure matches the hypothetical cross-section. This "automation" is an iterative technique that refines a user-supplied impedance guess until the response of the guess matches the data being inverted.

These refinements are the solution to a system of linear equations generated from a truncated (linearized) Taylor Series expansion of the forward model. This forward model is also known as the forward problem or the seismic response. Many different algorithms can be used for the forward model, and the best model would exactly mimic the seismic response of the earth. The finite difference technique would be the best forward model in terms of accuracy and completeness, but unfortunately it is extremely expensive in terms of computer time and storage. The forward model used here sacrifices accuracy for speed. It was developed by Wuenschel (1960) and Goupillaud (1961) and later improved by Larner (1977) and Robinson (1967). This model is exact for a plane layer earth and normally incident plane wave source and is computationally very fast. Because the forward model is one-dimensional only single traces of C.D.P. data from relatvely flat geologic areas can be inverted.

Vandell (1979) has used generalized inversion with Larner's foward model to invert synthetic seismic data. Although Vandell's technique works quite well on small, synthetic data sets, it is expected to be extremely expensive and in many cases unstable when applied to real seismic data. This is because Vandell solved for an impedance unknown corresponding to every data point on the trace being inverted. If one wishes to invert a seismic trace of 1,000 points with this technique, a 1,000 x 1,000 matrix equation must be solved, a task that is extremely time consuming and often impossible in the presence of noise. Another drawback of Vandell's technique is that it requires that the user supply the sourve wavelet, which is rarely known for real seismic data sets.

The goal of this work is to build a generalized linear inversion technique that is stable and cost effective when applied to large sets of real seismic data. To obtain this goal it was realized that the following problems must be solved: 1) The source wavelet on the observed seismic data is poorly known and must be solved for. The same is true for the scale factor of the observed data. Both of the above are of little use to the interpreter, but they can drastically alter the appearance of the modelled seismic response and are thus very important. 2) The number of parameters that describe the earth's impedance profile must be reduced so that the addition of the parameters describing the source wavelet and scale factor does not make the problem underdetermined or unstable. This reduction of impedance parameters also makes the problem computationally faster and more stable in the presence of noise.

These objectives were met and the technique was applied to real seismic data with very encouraging results.

THEORY OF GENERALIZED LINEAR INVERSION

The inversion of some data by any method starts with the selection of a function that models the generation of those data. This function can be an empirical relationship or a mathematical model of the physical processes that generated those data. This function will be referred to as the "forward model" which is not to be confused with a geologic model of the earth. In this paper the forward model generates a one-dimensional, plane wave synthetic seismogram from an impedance profile of the earth. If the forward model is simple enough it can be rewritten to express the impedance profile in terms of the observed seismic trace. This approach is known as direct or analytical inversion and unfortunately it does not work well with many forward models. A good seismic model is a non-linear function of many variables that cannot be analytically inverted. If such a model is to be inverted it must be done with a numerical technique; the numerical technique used here is generalized inversion.

The generalized linear inversion technique is based on a Taylor series expansion of the forward model. The Taylor series expansion of the forward model is of the form:

$$F(\bar{I}) = F(\bar{IG}) + \frac{\partial F(\bar{IG})}{\partial(\bar{IG})} (\bar{I} - \bar{IG}) + \frac{\partial^2 F(\bar{IG})(\bar{I} - \bar{IG})^2}{\partial IG^2 2!} + \ldots \qquad 1)$$

In this expansion:

$\bar{I}$ = the impedance profile to be solved for.

$\overline{IG}$ = A guess of what the impedance profile is. (*)

$(\bar{I} - \overline{IG})$ = Error in the above guess.

F = Forward modelling function.

$F(\bar{I})$ = Observed seismic trace.

$F(\overline{IG})$ = Synthetic seismic trace computed using $\overline{IG}$ in the forward modelling algorithm.

$\frac{\partial F(\overline{IG})}{\partial(\overline{IG})}$ = A partial derivative matrix.

It is important to note that the above expansion can be computed for any forward model. If the forward model is complicated it may be advantageous to calculate the derivative terms with finite differences as was done in this paper. In equation 1 what we wish to solve for is $(\bar{I}-\overline{IG})$ which tells how to correct $\overline{IG}$ to make it $\bar{I}$. Unfortunately the above infinite series cannot be inverted for $(\bar{I} - \overline{IG})$, but a truncated (linearized) version of it can be. This linearized version of equation (1) is:

$$F(\bar{I}) = F(\overline{IG}) + \frac{\partial F(\overline{IG})}{\partial\ \overline{IG}} (\bar{I} - \overline{IG}) \qquad 2)$$

or

$$F(\bar{I}) - F(\overline{IG}) = \frac{\partial F(\overline{IG})}{\partial\ \overline{IG}} (\bar{I} - \overline{IG}) \qquad 3)$$

*footnote. See the section on generation of initial guess for a more complete description of this vector.

The term $(F(\bar{I}) - F(\bar{IG}))$ is a vector generated by sub tracting the synthetic seismic trace from the observed seismic trace. It will be referred to as the difference vector. The term $(\partial F(\bar{IG})/\partial \bar{IG})$ is a partial derivative or sensitivity matrix. Each column of this matrix is the partial derivative of the synthetic seismic trace with respect to one of the unknown impedance values. The term to be solved for, $(\bar{I} - \bar{IG})$, will be called the correction vector. Equation 3 in matrix form is:

$$\begin{vmatrix} F(I_1)-F(IG_1) \\ F(I_2)-F(IG_2) \\ \cdot \\ \cdot \\ F(I_M)-F(IG_M) \end{vmatrix} = \begin{vmatrix} \frac{\partial F(I_1)}{\partial IG_1} & \frac{\partial F(I_1)}{\partial IG_2} & \cdot & \cdot & \frac{\partial F(I_1)}{\partial IG_N} \\ \frac{\partial F(I_2)}{\partial IG_1} & \frac{\partial F(I_2)}{\partial IG_2} & \cdot & \cdot & \frac{\partial F(I_2)}{\partial IG_N} \\ \cdot & \cdot & \cdot & \cdot & \cdot \\ \cdot & \cdot & & & \cdot \\ \frac{\partial (I_M)}{\partial IG_1} & \frac{\partial F(I_M)}{\partial IG_2} & \cdot & \cdot & \frac{\partial F(I_M)}{\partial IG_N} \end{vmatrix} \begin{vmatrix} I_1-IG_1 \\ I_2-IG_2 \\ \cdot \\ \cdot \\ I_M-IG_M \end{vmatrix} \qquad 4)$$

Where M=number of observations (points in the observed seismic trace), and N=number of parameters (impedances to be solved for).

This set of simultaneous linear equations is solved for the correction vector using a modified least-squared-error matrix inversion technique. Once the correction vector is known it is a simple matter to solve for $(\bar{I})$:

$$\bar{I} \simeq \bar{IG} + (\bar{I} - \bar{IG}) \qquad 5)$$

Equation 5 is an approximation because in going from equation 1 to equation 2 the non-linear terms in the Taylor series expansion were truncated. This causes the solution for the correction vector ($\bar{I}-\bar{I}\bar{G}$) from equation 3 to be an approximation and the solution for $\bar{I}$ in equation 5 must also be an approximation. The error represented by this approximation can be reduced by using the corrected initial guess from equation 5 as a new initial guess in equation 3 and iterating through the problem again. This iterative procedure is outlined in Figure 1. Figure 2 is an example of successive iterations converging to an acceptable answer.

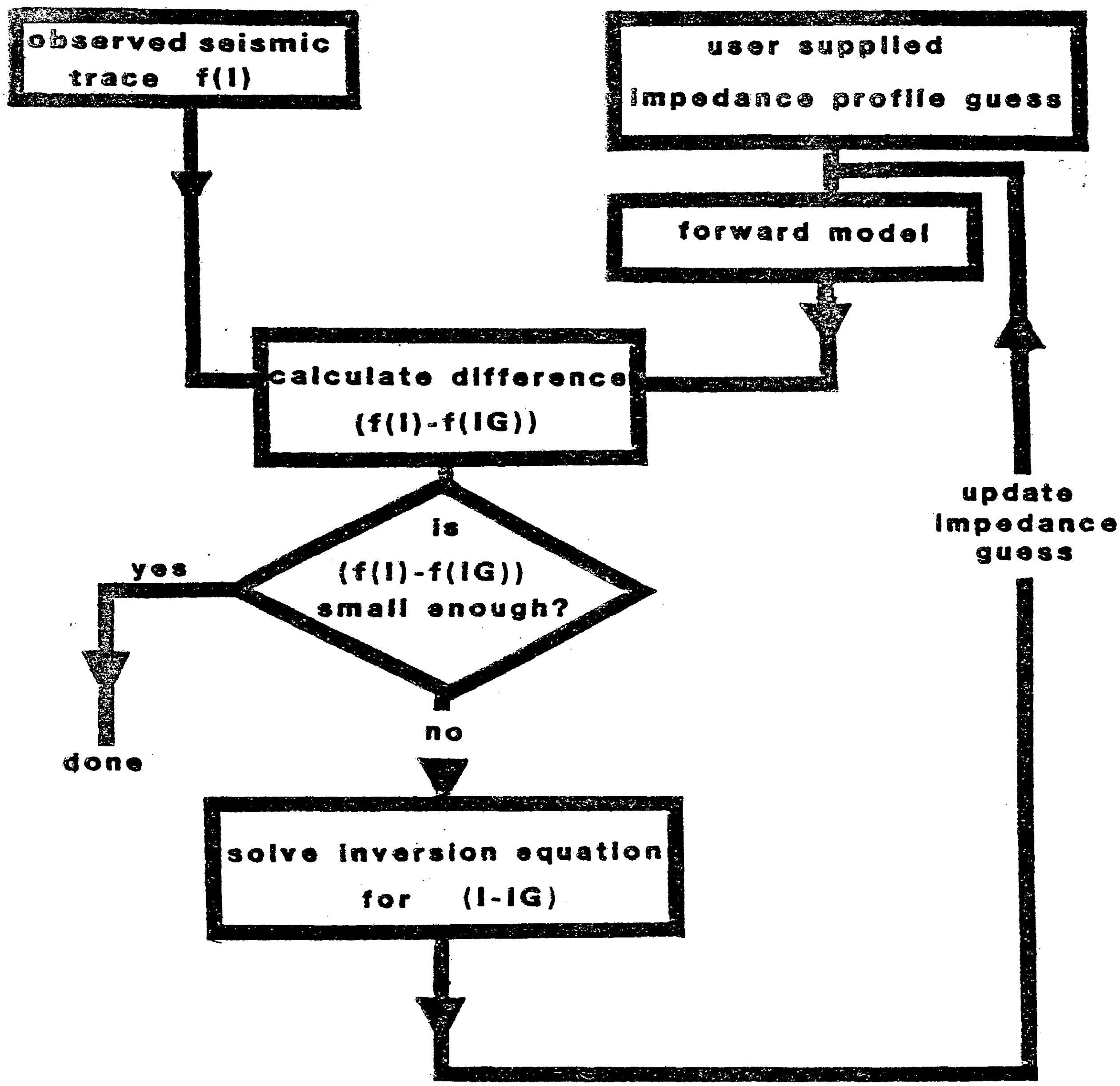


Figure 1. Flow Chart for generalized linear inversion.

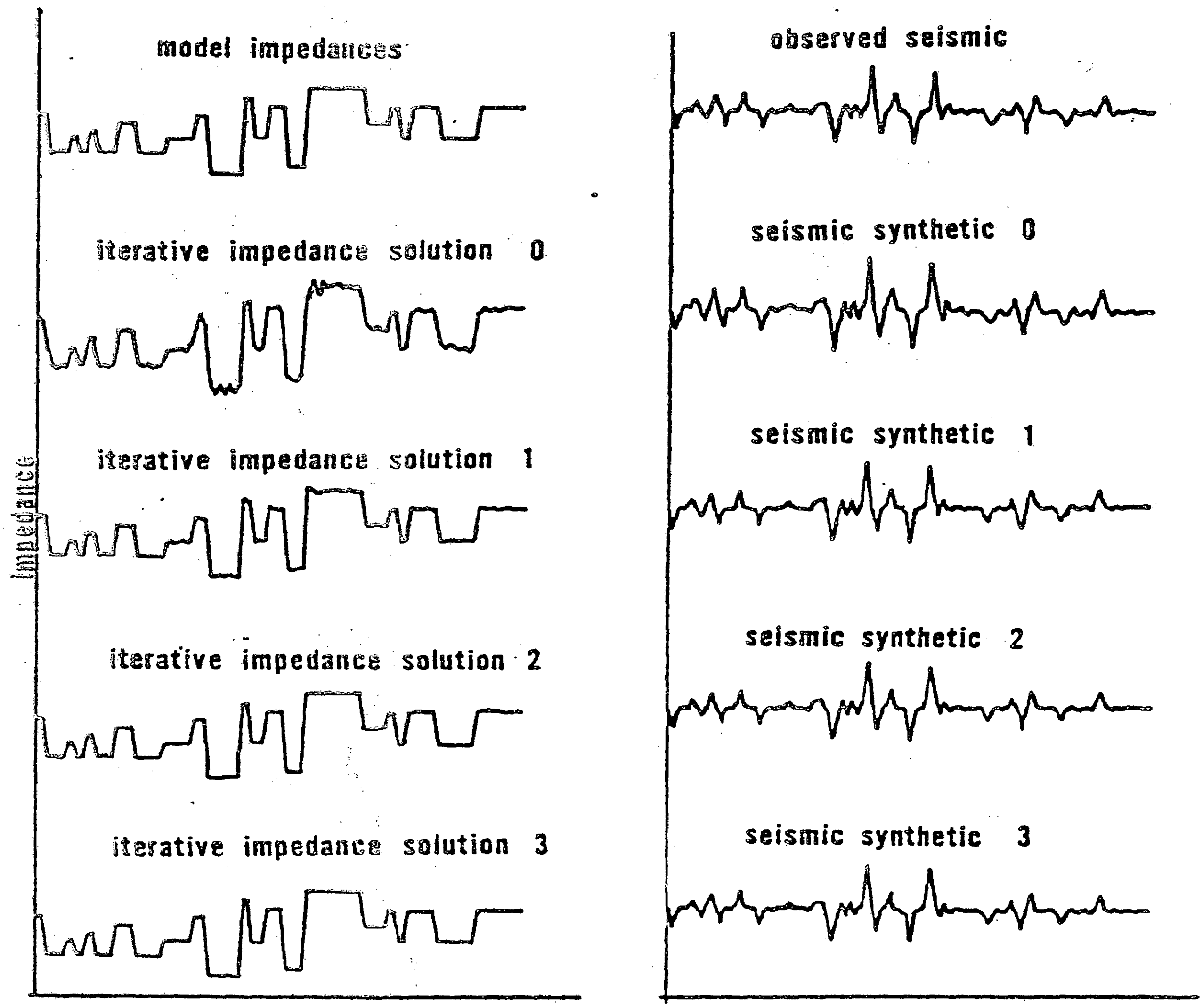


Figure 2. Example of generalized linear inversion.

LEAST-SQUARED-ERROR MATRIX EQUATION SOLUTION

Equation 4 is a set of simultaneous equations that often cannot be solved in such a way that each individual equation is satisfied. This situation can occur when there is noise present in the observations and/or when there are more equations than unknowns. When it is impossible to find a solution which exactly satisfies all the equations one looks for the best answer defined by a least-squared-error criteria.

For the purpose of notational simplicity we rewrite equation 4 by letting:

$$(F(\bar{I}) - F(\bar{IG}) = \bar{d} = \text{difference vector.}$$

$$\frac{\partial F(\bar{IG})}{\partial IG} = S = \text{sensitivity matrix.}$$

$$(\bar{I} - \bar{IG}) = \Delta\bar{p} = \text{correction vector.}$$

Using the above, equation 4 becomes

$$\bar{d} = S\Delta\bar{p} \tag{6}$$

where we wish to solve for $\Delta\bar{p}$. The error in the solution for $\Delta\bar{p}$ is defined as:

$$\text{error} = \bar{e} = (\bar{d} - S\ \Delta\bar{p}) \tag{7}$$

We wish to minimize the magnitude of the error vector, which is defined as:

$$
\begin{aligned}
|e| &= \bar{e}^2 = \bar{e}^T\bar{e} \\
&= (\bar{d} - S\Delta\bar{p})^T (\bar{d} - S\Delta\bar{p}) \\
&= \bar{d}^T\bar{d} - \bar{d}^T S\Delta\bar{p} - \Delta\bar{p}^T S^T\bar{d} + \Delta\bar{p}^T S^T S\Delta\bar{p} \\
&= \bar{d}^T \bar{d} - 2\Delta\bar{p}^T S^T\bar{d} + \Delta\bar{p}^T S^T S\Delta\bar{p} \qquad (8)
\end{aligned}
$$

To minimize the above error set

$$d|e|/d\Delta\bar{p} = 0 = -2S^T\bar{d} + 2S^T S\Delta\bar{p}$$

or

$$|\Delta\bar{p}| = (S^T S)^{-1} (S^T\bar{d}) \qquad (9)$$

This is the optimum solution for $\Delta\bar{p}$ using the least-squared-error criteron.

MODIFICATIONS TO THE LEAST-SQUARED-ERROR SOLUTION

Equation 9 is a solution to equation 4 based on minimizing the magnitude of the error vector. There is an additional condition that can be imposed in obtaining the solution for $\Delta\bar{p}$. This condition is to constrain the magnitude of the solution vector $\Delta\bar{p}$ as discussed by Marquardt (1963). The weighted magnitude of $\Delta\bar{p}$ is:

$$\Delta p_w = k^2 \ (\Delta\bar{p}^T \Delta\bar{p}) \qquad 10)$$

where k^2 is some weighting factor to be determined later. The magnitude of the error vector can be redefined by modifying equation 8 to include the above term. This gives:

$$e + k^2 \Delta p = \bar{d}^T \bar{d} - 2\, \Delta\bar{p}^T S^T \bar{d} + \Delta\bar{p}^T S^T S\, \Delta\bar{p} + k^2\, \Delta\bar{p}^T \Delta\bar{p}$$

When this error is minimized, the result is:

$$d\, e / d\Delta\bar{p} = 0 = -2S^T \bar{d} + 2S^T S\Delta\bar{p} + 2k^2 \Delta\bar{p}$$

or

$$(S^T S + k^2 I)\, \Delta\bar{p} = S^T d$$

where I is the identity matrix. This can be rewritten as

$$\Delta\bar{p} = (S^T S + k^2 I)^{-1} S^T \bar{d} \tag{11}$$

In this expansion, k^2 is usually referred to as the damping factor.

The question arises as to what purpose this damping factor can serve. It turns out that often a little bit of noise in $\bar{d}$ or the S matrix (which is due to truncation of the Taylor series expansion) can lead to a large error in $\Delta\bar{p}$, especially when S is nearly singular. If we constrain or damp the magnitude of $\Delta\bar{p}$, it is possible to obtain a solution for $\Delta\bar{p}$ that favors the signal and discriminates against the noise. This is analogous to the prewhitening (or adding white noise to diagonal of the matrix) often done in deconvolution using the Wiener-Levinson algorithm.

An explanation of how the damping factor works is given by considering all possible states of the error vector as a multidimensional surface. The term $d|e|/d\Delta\bar{p}$ is the gradient of the surface at the point $\overline{IG}$ ($\overline{IG}$ in this problem is the initial or the current updated guess of the earth's impedance profile). Setting $d|e|/d\Delta\bar{p} = 0$ and solving for $\Delta\bar{p}$ will give an exact answer only if the error surface is linear between the point $\overline{IG}$ and the point of minimum error, but the error surface is non-linear due to noise and the way in which the problem is defined. The choice of k^2 should reflect the linearity of the error surface between the point where the gradient was calculated and the point of minimum error (see Figure 3).

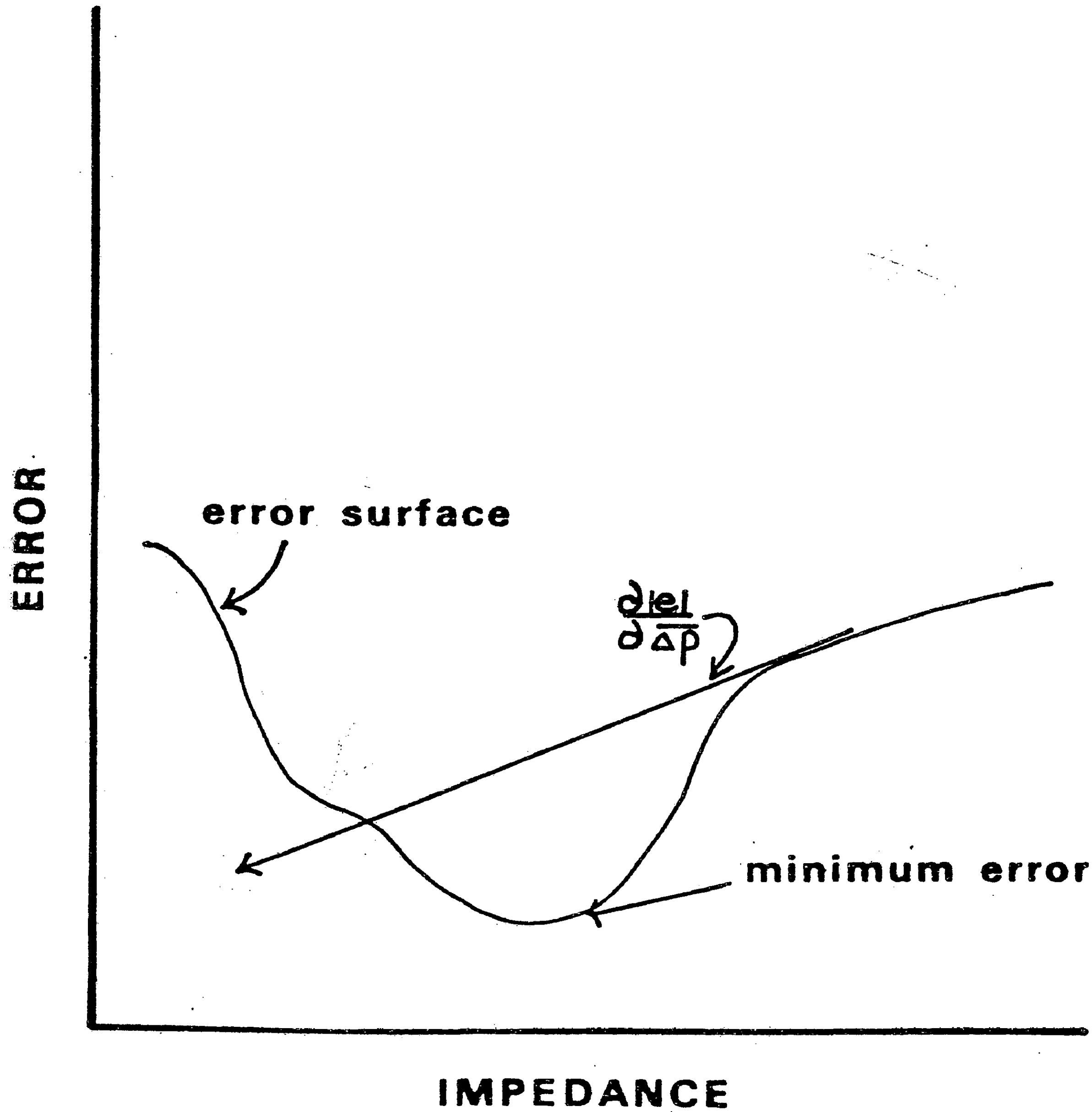


Figure 3 Two dimensional slice of the multidimensional error surface. Note non-linearity of surface.

CHOOSING THE DAMPING FACTOR

Theoretically, k^2 should be chosen to reflect the degree of linearity of the error surface. Since the error surface is not known, a more empirical method is used.*

A short outline of how the damping factor is chosen follows:

1) Compute $\Delta\bar{p}$ using a number of damping factors; that is, solve equation (11) for a number of different k^2.

2) For each of the new $\Delta\bar{p}$ vectors calculate a new impedance profile using equation 5 (remember $\Delta\bar{p} = (\bar{I}-\bar{I}\bar{G})$).

3) For each of the new impedance profiles compute a new synthetic seismic trace.

4) Calculate an rms error between the observed seismic trace and each of these new synthetic seismic traces.

5) Interpolate between the known values to get the damping factor that gives the smallest rms error.

The neighborhood about which the guesses for k^2 in step 1 are chosen is determined by experience.

*In his paper Marquardt gives another method of choosing the damping factor. The method discussed here comes from Stoyer (personal communication).

Uniqueness and Resolution

Once the correction vector $\Delta\bar{p}$ (or $\bar{I}-\bar{I}\bar{G}$) has been computed from equation 11 it is possible to compute the resolution matrix as defined by Backus and Gilbert (1968):

$$R = [S]^{-1}S.$$

The matrix R is a measure of the uniqueness of the solution. When R is the identity matrix, the solution is unique. When R is not the identity matrix it indicates just which parameters are not well resolved. If the nth element of the nth row (that is the element that lies along the diagonal of R) is one, then the nth impedance in the solution is unique. When this element is not unity, the adjacent elements indicate just how well resolved the nth parameter is compared to its neighbors. For example, if the nth row of R is all zeros except for the n-1, n, n+1 elements which are all 1/3, then the nth impedance in the solution is ill-resolved with respect to the n-1 and n+1 impedance.

THE FORWARD MODEL

The term "forward model" in the context of this paper refers to the function $F(\overline{IG})$ found in equation 1. The forward model used here generates a one dimensional synthetic seismogram complete with all multiples and transmission losses. Although only one forward model will be discussed and used here it is important to realize that many forward models can be used with generalized inversion. The forward model used here was first formulated by Wuenschel (1960) and later numerically simplified by Robinson (1967) and Larner (1977).

When doing generalized inversion, it is extremely important that the forward model accurately describe the generation of the data that are to be inverted. Wuenschel's forward model assumes a plane-wave source and a laterally homogeneous earth. These assumptions are clearly violated in most if not all seismic data acquisition, and thus Wuenschel's solution appears unsuitable. This dilemma is solved by inverting only common-depth-point stacked data. C.D.P. data mimics the response of a laterally homogeneous earth to a plane wave source in flat geologic areas and thus can be inverted using Wuenschel's forward model.

To develop Wuenschel's algorithm we need first define the reflection coeficient C_j:

$$c_j = \frac{(P_{j+1}\ V_{j+1}) - P_j\ V_j}{P_{j+1}\ V_j + P_j\ V_j}$$

where P_j = density of the jth layer and V_j = velocity of the jth layer.

In Wuenschel's method, the earth is discretized into arbitrary intervals of equal two way travel time (delta t). As long as this basic interval is small compared to the geology of interest this discrete approximation to the real geology is acceptable. The thicker lithologic units correspond to integer multiples of delta t with artificial interior boundaries having reflection coefficients set to zero. The output of Wuenschel's model is the reflectivity function, which when convolved with a source wavelet equals the synthetic seismic trace. The advantage of modelling using this discrete earth is that the reflectivity function becomes a discrete time series which allows the use of Z transforms.

If one is to consider a one-layer earth (see Figure 4) the reflectivity function consists of 2 primaries and an infinite string of multiples. This reflectivity function is denoted by $R_o(z)$ and in the Z domain is equal to:

$$R_o(z)=c_o + c_1(1-c_o^2)Z - c_oc_1^2(1-c_o^2)z^2 + c_o^2c_1^3(1-c_o^2)Z^3 + \ldots \qquad 13)$$

where c_j = is the reflection coefficient as defined above

Z = the unit delay operator.

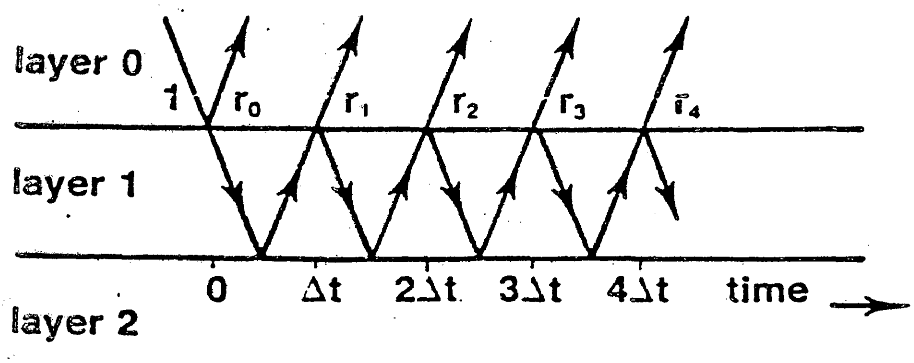


Figure 4. Primaries plus multiples from the single layer model. (After Larner et al., 1977)

Equation 12 is an infinite series, but it can be written in closed form as the quotient:

$$R_o(Z) = \frac{c_o + (1-c_o^2)c_1 Z}{1+c_o\, c_1\, Z} = \frac{c_o + c_1 Z}{1+c_o\, c_1\, Z} \qquad 14)$$

Equation 14 gives the response of a single layer earth; what is wanted here is the response of a multi-layer earth as in Figure 5. Equation 14 (for the single layer case) can be used with the multilayer earth if the multilayer earth has an equivalent single layer representation as in Figure 6. Using the single layer representation we see that c_1 in equation 14 becomes R_1 and

$$R_o(Z) = \frac{c_o + R_1\, Z}{1+c_o\, R_1\, Z} \qquad 15)$$

Now to calculate $R_o(Z)$, the term $R_1(Z)_1$ is needed which would be calculated from $R_2(Z)$, but $R_2(Z)$ depends on $R_3(Z)$ and so on down to the last layer $R_{n-1}(Z)$. In general:

$$R_j(Z) = \frac{c_j + R_{j+1}\, Z}{1+c_j\, R_{j+1}\, Z} \qquad j=n-1,n-2,\ldots 2,1,0 \qquad 16)$$

and for the bottom (nth) interface under consideration:

$$R_n(Z) = c_n. \qquad 17)$$

This is the starting place for recursive application of equation 16. $R_n(Z)$ is given from equation 17; then one can solve

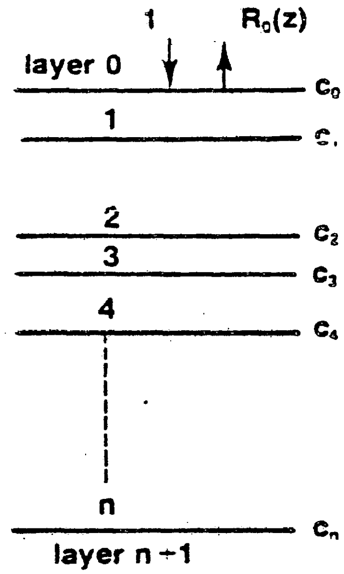


Figure 5 Layered model for the normal incidence synthethic seismogram. The medium is excited at the bottom of the upper half-space by a unit spike. Layer thickness may be nonuniform but correspond to equal time units. (After Larner et al., 1977)

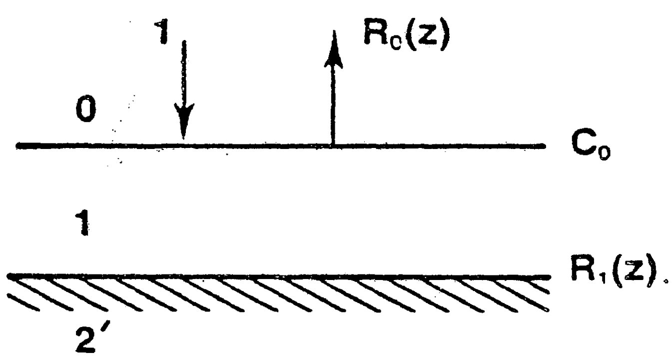


Figure 6 Equivalent single layer representation of the multilayer model in Figure 5. $R_1(Z)$ is a generalization of C_1. Layer 2 may be considered as the interior of a black box. (After Larner et al., 1977)

for R_{n-1}, $R_{n-2} \ldots R_1, R_o$. This method works well, but is computationally inefficient due to the polynomial division required with each step. This can be avoided if in equation 16 the numerator is renamed B_j and the denominator is termed A_j:

$$R_j(Z) = \frac{c_j + R_{j+1} Z}{1 + c_j R_{j+1} Z} = \frac{B_j}{A_j} \qquad 18)$$

This implies that

$$R_j = B_j / A_j, \qquad 19)$$
$$R_{j+1} = B_{j+1}/A_{j+1}$$

If 19 is substituted in equation 16 the result is:

$$R_j = \frac{c_j + (B_{j+1} / A_{j+1}) Z}{1 + c_j (B_{j+1} / A_{j+1}) Z} = \frac{c_j A_{j+1} + B_{j+1} Z}{A_{j+1} + c_j B_{j+1} Z} \qquad 21)$$

or

$$A_j = A_{j+1} + c_j B_{j+1} Z \quad \text{for } j=n-1, n-2 \ldots 2,1,0 \qquad 21)$$

and $B_j = c_j A_{j+1} + B_{j+1} Z$

Equation 21 can be applied recursively starting with the bottom layer where $B_n = c_n$ and $A_n = 1$. When A_o and B_o are found then $R_o(z) = B_o/A_o$. This gives the same answer as a recursive application of equation 16, but since this method requires only one polynomial division, it is much quicker.

In summary, note that equation 21 gives a computationally fast method of computing a normally incident synthetic seismogram with all intrabed multiples and transmission losses. The above discussion has assumed a normally incident unit impulse as a source, but any source wavelet can be convolved with the results.

PARAMETERS OTHER THAN IMPEDANCE

The output of the forward modelling algorithm discussed in the previous section is a reflectivity function that does not adequately resemble what it is supposed to - a real seismic trace. As mentioned in the beginning of the last section it is imperative that these two be as similar as possible when the input impedance profile to the modelling algorithm is identical to the earth responsible for real seismic trace. The difference between this reflectivity function and the real seismic data is that the real seismic trace has been scaled and has a source wavelet convolved with it. The reflectivity function can be made to resemble the seismic trace by convolving it with a source wavelet and then scaling the result. Unfortunately, neither the source wavelet nor this scale factor are known. This problem can be circumvented by solving for the source wavelet and the scale factor at the same time that one solves for the unknown impedances.

The Source Wavelet

Figure 7 illustrates why the source wavelet needs to be known. This figure shows inversion of some data where the source wavelet is assumed to be known but is incorrect. In

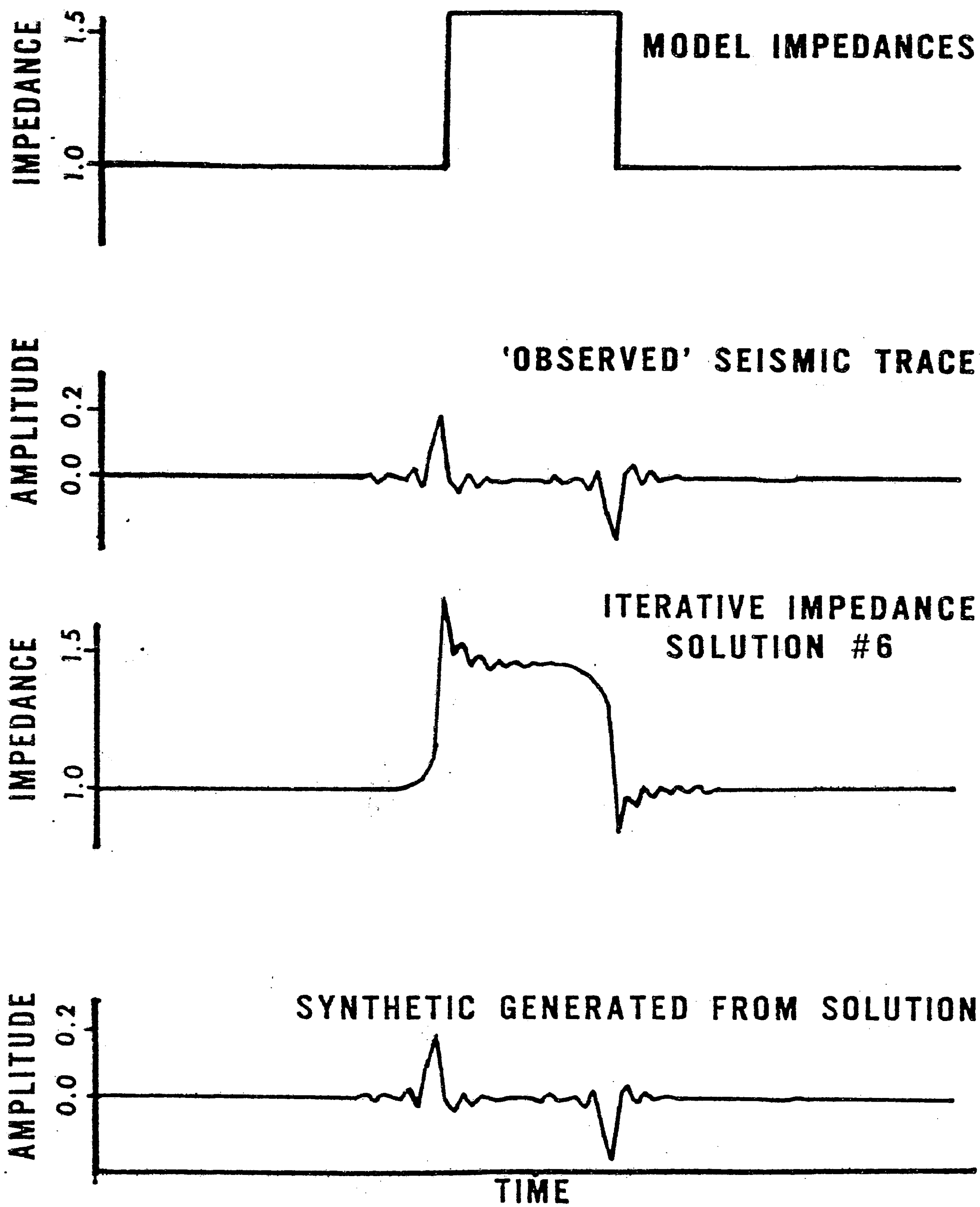


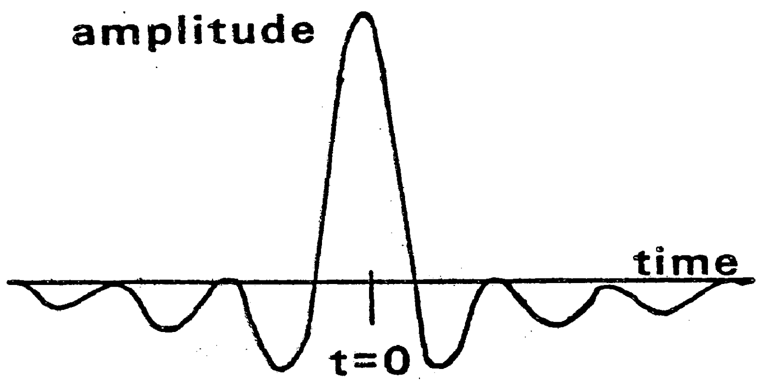
Figure 7 Inversion with incorrect source wavelet.

Figure 7 we have generated an "observed" seismic trace from the shown model impedances. The source wavelet has a 5-10-115-120 band pass and a 22.5 degree linear phase shift. When inversion is done, a 5-10-115-120 zero phase wavelet is assumed (incorrectly) to be the source wavelet. The solution impedance profile is erroneous due to this incorrect assumption.* The deterioration of the results shows just how important it is to have an exact description of the source wavelet. The alternative to this is to solve for the source wavelet.

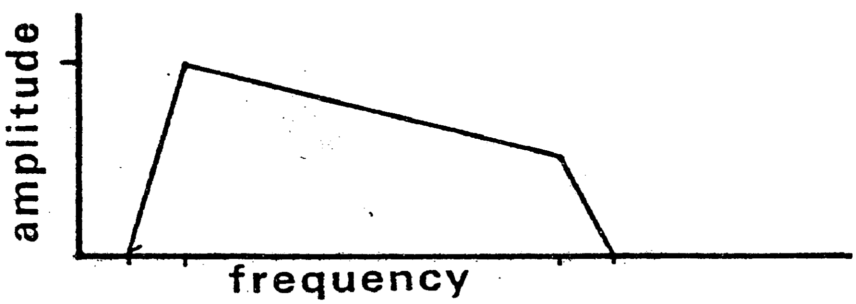
To solve for the wavelet, the wavelet must first be parameterized - i.e., a set of variables must be found that can describe any source wavelet. These variables could be the time series that is the sampled wavelet, but it is more efficient to parameterize the wavelet in the frequency domain. In the frequency domain the wavelet's amplitude spectrum is defined by five variables that generate a trapezoidal band pass, and the phase spectrum is described by a single number that gives a linear phase shift (see Figure 8). This type of phase characteristic is chosen to invert zero phase processed seismic data. The linear phase shift is included to accommodate

*Figure 7 was computed using the inversion program written by Vandell. The inversion program listed in the back of this thesis parameterizes the earth in a more efficient manner than Vandell's and gives better results than those shown in Figure 7 as shown in Figure 15. Please see the section on parameterization for a more complete explanation of this.

A



B



C

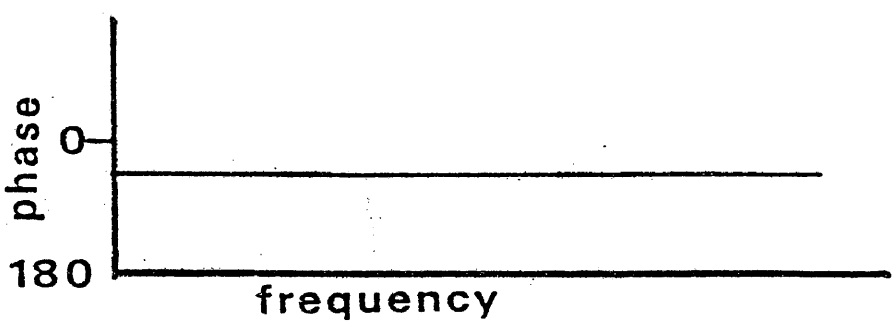


Figure 8 Parameterization of the source wavelet. Part A is the time domain representation of the wavelet. Parts B and C show the frequency domain paramterization of A.

a type of phase error described by Fausset (1979). Fausset showed that a minimum phase seismic trace with noise will have non-zero phase characteristics after spiking deconvolution. These non-zero phase attributes can be characterized as a linear phase spectrum of some intercept and slope. The slope is irrecoverable - it does not change the shape of the wavelet, it only delays or advances it in time. The intercept can radically alter the shape of the wavelet and must be accounted for (as in Figure 8 part c).

The disadvantage of this type of wavelet parameterization is that the wavelet is constrained to having a trapezoidal band pass and zero or linear phase characteristics. Complicated amplitude spectra and minimum or mixed phase spectra cannot be described with this wavelet parameterization. However, any wavelet that can be parameterized with other methods, can be solved for if that parameterization is used.

To solve for the wavelet using this parameterization, one need only make an initial guess for the wavelet, modify equation 4, and then solve for the corrections just as for the impedances. The final band pass filter that the trace to inverted has been filtered with is used at the initial guess. The modifications to equation 4 consist of 1) augmenting the sensitivity matrix with the six vectors that are the partial derivatives of the synthetic trace with respect to the six wavelet parameters and 2) lengthening the correction vector by six. When the correction vector is solved for it will tell how to correct the initial guess for the wavelet to make it more like the actual source wavelet.

The Scale Factor

Even if the impedance profile and source wavelet used to generate the above synthetic seismic trace are identical to those that generate a real seismic trace, the real and synthetic seismic traces can still differ by a large scale factor. If the source wavelet is normalized to a maximum amplitude of one (as was done here) then the maximum possible amplitude on the synthetic seismic trace will be one. (This is because the maximum possible reflection coefficient is one.) Actual seismic traces are scaled and have peak amplitudes much larger than this. This scale factor is the result of gain encountered in recording, stacking and processing the data. This scale factor must be solved for since it is unknown.

This scale factor does not change the relationship between the impedances and the reflection coefficient series. This can be seen by scaling the impedances in equation 12 by some constant scale factor k:

$$C_j \text{ scaled} = \frac{k(P_{j+1}\ V_{j+1})-k(P_jV_j)}{k(P_{j+1}\ V_{j+1})+k(P_jV_j)} = \frac{k[(P_{j+1}V_j)-(P_jV_j)]}{k[(P_{j+1}V_j)+(P_jV_j)]}$$

$$= \frac{(P_{j+1}V_j)-(P_jV_j)}{(P_{j+1}V_j)+(P_jV_j)} = C_j \text{ unscaled}$$

Where k = constant scale factor.

The above shows that when the impedance profile is scaled by a constant, the reflection coefficient series is unaffected. This means that any impedance solution generated from the reflection coefficients is non-unique with respect to a scale factor. To make the impedance solution unique, one needs only to fix at least one of the impedances. When one impedance is fixed, all other impedances can be related to the known value and the constant scale uniqueness problem is avoided. In recursive inversion techniques it is the first impedance value that must be fixed to avoid this scale problem. In generalized inversion any impedance value can be fixed in this manner (please see section on constraints for an explanation of how this is done).

The above paragraph discusses the scale of the impedances when computing the reflection coeffecients. We must also consider the scale of the reflection coeffecient series when inverting for the impedances. The easiest way to do this is to use recursive (Seislog*) inversion which is exact when the trace being inverted is the reflection coeffecient series. Recursive inversion is just equation 12 rewritten to express

*Seislog is a registered trademark of the Technica Co.

the impedances in terms of the reflection coeffecient series or:

$$(pv)_{j+1} = pv_j \frac{(1+RC_j)}{(1-RC_j)} \qquad 22)$$

Now if the reflection coeffecient series is scaled by a factor x, the above equation becomes:

$$(pv)_{j+1} = (pv_j) \frac{(1+XRC_j)}{(1-XRC_j)} \qquad 23)$$

When inverting a reflection coeffecient series one hopes to get a solution based on equation 22, but if the data are scaled by an unknown amount then the recursive result is given by equation 23. Table 1 illustrates the magnitude of this type of error. In Table 1 $(pv)_{j+1}$ is computed using equation 23 and various scale factors x. The term $(pv)_j$ is set at 16,000 $(\frac{gm^3}{cm}\frac{ft}{sec})$ and RC_j, is 0.2. The correct value for $(pv)_{j+1}$ is given when x=1.

Table 1

x	$(pv)_{j+1}$
0.5	19,555
1	24,000
1.5	29,714
2	37,333
5	$\pm\infty$
10	-48,000
100	-17,684

Figure 9 illustrates how this scale error will effect recursive inversion. In Figure 9 the response of a three layer

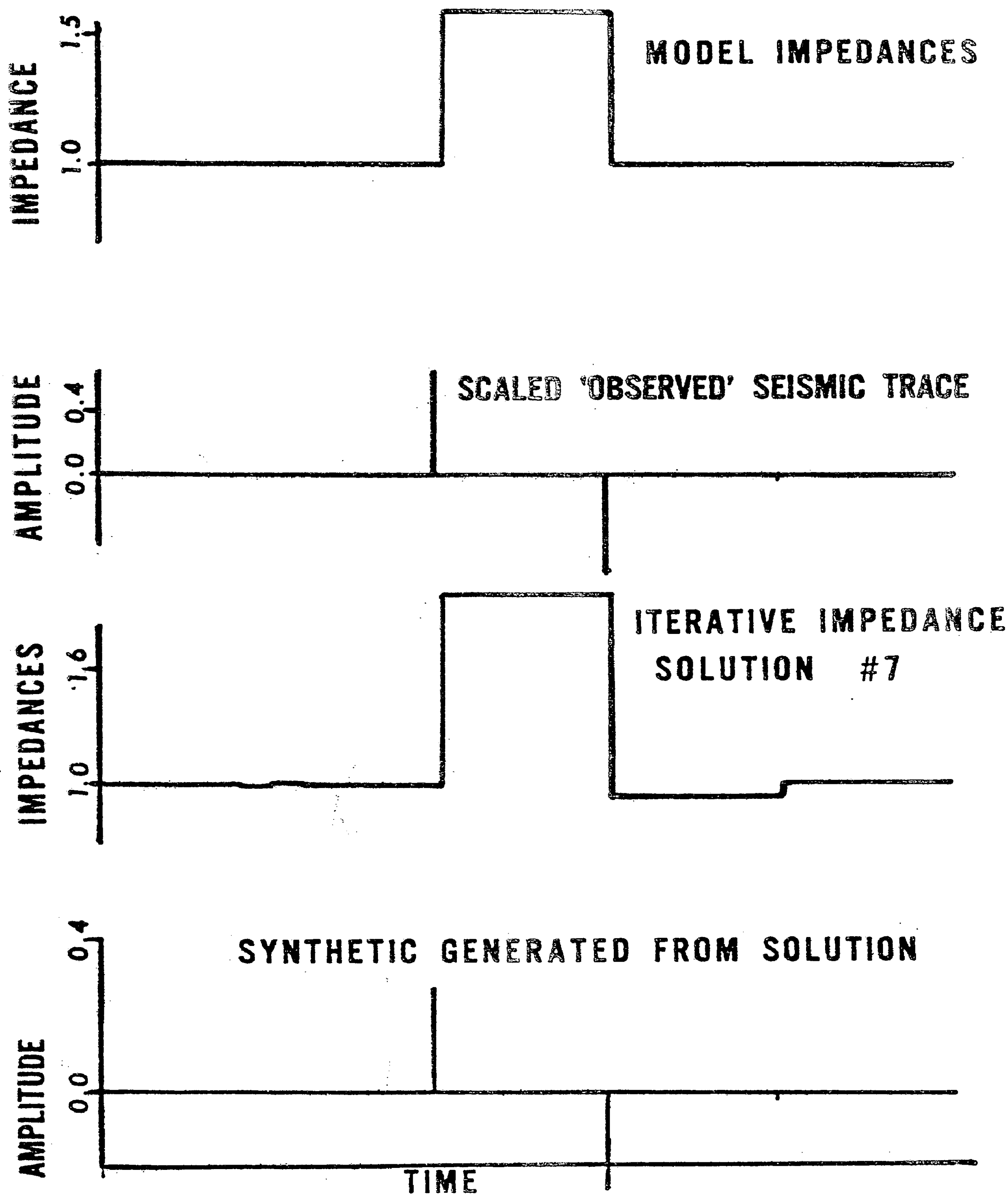


Figure 9 Inversion with incorrect scale assumption.

earth is computed and then scaled by a factor of 2.0. When inversion is done it is assumed that there is no scale factor. The inversion results are in error due to this incorrect assumption.

This type of error is due to a non-linear relationship that exists between the amplitudes in the reflectivity function. Appendix A has a description of how this non-linear relationship comes about and how the inversion errors in Figure 9 are caused. The existence of this non-linear amplitude relationship allows one to solve for the scale factor if multiples are present. Since there can be only one time-constant scale factor that will give the amplitude relationship between primary and multiple arrivals as described in Appendix A, this scale factor is unique and can be solved for with generalized inversion. Practice has shown, though, that the multiples must be fairly large -- they must be larger than the noise and larger than 1/10 of the primary event amplitude. It is also necessary that one build the initial guess with no impedance contrasts corresponding to the multiple arrivals. If these conditions are met the scale factor can be solved for in the same manner described above for the source wavelet.

PARAMETERIZATION OF THE IMPEDANCE PROFILE

The easiest way to parameterize an impedance profile is to use the same method that was used to parameterize to observed seismic trace - that is to list an impedance value at discrete and fixed intervals of 1, 2 or 4 milliseconds. This "equal sample digitization" of the impedance profile is the method that Vandell (1979) used, and with good reason; the fastest and most efficient forward modelling algorithms require an impedance input in just this form. Since the forward model must be computed many times in each inversion iteration, Vandell's method of parameterization is well chosen.

The problem with fixed interval digitization is that when inversion is done with it the number of unknowns exactly equals the number of observations. This happens because the above mentioned type of forward modelling algorithm requires an input of N impedance values (the corrections to these impedances are the unknowns) at intervals of delta t to create a synthetic seismic trace of the same dimensions. Since this synthetic seismic trace must be of the same dimensions as the observed seismic trace to be inverted, the number of points (observations) in the trace to be inverted equals the number of unknowns.

Equation (5) can theoretically be solved when the number of unknowns eauals number of equations, but there are three problems associated with this situation. They are;

1) Stability. The inversion of a NxN matrix equation is no problem when all the equations are independent and there is no noise. When these conditions are violated, the inversion can become singular and/or unstable.

2) Cost. If one inverts a 2 sec. seismic trace recorded at a 2 millisecond sample interval the matrix equation to be inverted will be 1000x1000. This will be prohibitively expensive in computer time and space.

3) Other unknowns. It is important to solve for other parameters than impedance. If the number of unknowns equals number of equations these extra parameters cannot be added without the problem becoming under-constrained (more unknowns than observations).

The above listed problems can be avoided if the earth is parameterized using fewer variables (unknowns). Of course this parameterization must include as much information as the previous method if it is to work. The parameterization used in this paper describes the earth in terms of separate blocks or lithologies and is termed "lithology dependent parameterization." Each lithology has assigned to it:

1) a variable impedance value at the start of the "block" (the first impedance value),

2) a variable linear rate of change of the impedance within the block (the impedance gradient), and

3) a variable time thickness.

The starting time of a given interval is determined by the total time thickness of all intervals above it. Figure 10 compares this "lithology dependent" method to the continuous earth.

Since three variables are assigned to each "lithologic" interval in this scheme, the number or parameters are not lessened unless the average thickness of all intervals is greater than three time units (where one time unit equals the sample rate of the observed trace). In practice this limit is not even approached because most lithologies need not be parameterized. In many applications of the algorithm presented here, the value of known lithologies are fixed as in examples (25-26). These lithologies need not be parameterized and solved for. It is also possible to not parameterize many unknown lithologies. These lithologies are the ones that have no impact on the zone of interest. The only lithologies that must be parameterized and solved for are those that are of interest and those that change the appearance of the zone of interest - i.e., intervals that cause multiples to fall on the zone of interest.

Using lithology dependent parameterization instead of equal sample digitization can make the number of unknowns less than the number of observations which leads to increases

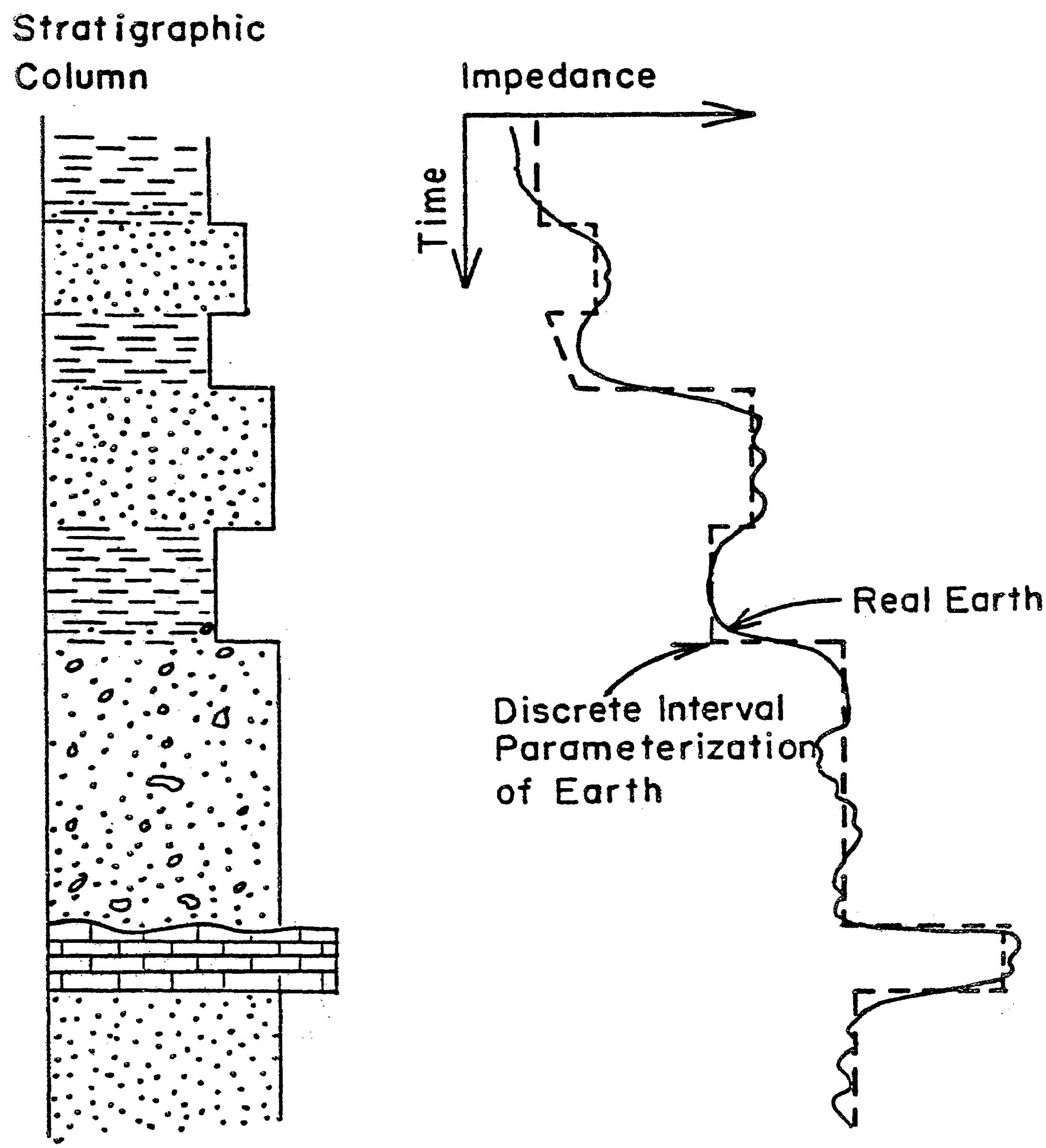


Figure 10. Continuous vs. discrete interval parameterization.

in speed and stability while enabling the inversion process to accept other parameters. It should be noted that Wuenshel's (1960) efficient forward modeling algorithm can still be used with lithology dependent parameterization. Only in calculation of the S (sensitivity) matrix is it important to lessen the number of unknowns describing the earth. After this is done one can transform an impedance profile from lithology dependent parameterization to equal sample parameterization and then be able to use Wuenschel's solution.

Two unexpected benefits of the lithology dependent parameterization are 1) the way it avoids inverting intra-boundary noise (see Figure 15) and 2) the way it lessens the need for an exact description of the source wavelet (see Figure 15).

It was found that when using lithology dependent parameterization the boundary locations must be solved for independently of the impedances. The reason for this is apparent when one views the generalized linear inversion process as a procedure that minimizes the error between the observed seismic trace and the synthetic seismic response of a corrected initial guess. If a given boundary has the correct impedance contrast but the wrong location for that contrast there are two ways to lessen the error: 1) Move the location of the boundary - this is what is wanted. 2) Remove the impedance contrast by setting the impedances on both sides of the boundary equal. This will lessen the error only if the current boundary location is incorrect - even if the impedances are correct. To avoid this the impedances are solved for only after all the boundary locations are known.

PARTIAL DERIVATIVES

To compute the sensitivity matrix in equation 4 it is necessary to take partial derivatives of the synthetic seismic trace with respect to each boundary location, each first impedance value and the linear impedance trend within each "lithologic interval." Also needed are the partial derivatives of the synthetic with respect to the wavelet (amplitude and phase) and the scale factor. Computation and storage of these derivatives is the most time and computer memory consuming operation encountered in generalized linear inversion. Fortunately, generalized linear inversion is a very robust process and will allow one to use approximations to these partial derivatives that may not be exact, but are computationally more efficient than the exact derivatives. In fact, the error introduced by these approximations is probably less than the error due to truncating the Taylor Series expansion (see Page 14). A number of different approximation techniques are used here to generate the desired derivatives. Depending on the derivative needed an analytical, numerical (finite difference) or approximation to an analytical derivative is used.

The partial derivative of the synthetic seismic with respect to boundary locations is calculated by convolution with a derivative operator. A derivation of this approximation follows from considering a seismic trace that contains only primary arrivals and is represented by:

$$s(t)=w(t)*\sum_{i=1}^{N}(a_i\ (t-\ {}_i)$$

where $s(t)$ = seismic trace

$w(t)$ = source wavelet

${}_i$ = location (in time) of a boundary.

a_i = magnitude of primary arrival.

Taking the partial derivative of the above with respect to the boundary location ${}_i$:

$$\frac{\partial s(t)}{\partial\ {}_i} = aw(t)* -\ \ '(t-\ {}_i)$$

This is an approximation to the exact derivative $\partial(s(t)/\partial_i)$ because the multiple arrrivals from the boundaries are not included in the above analysis. This technique was compared with a finite difference derivative that did take the multiple arrivals into account. The finite difference technique gave the same results for simple test cases but was slower to compute. The doublet ['(t)] used is a numerical approxima tion from Butkov (1968) page 226: (See Figure (11))

$$'(t)= \frac{-2n^3}{\pi}\ te^{-n^2t^2}$$

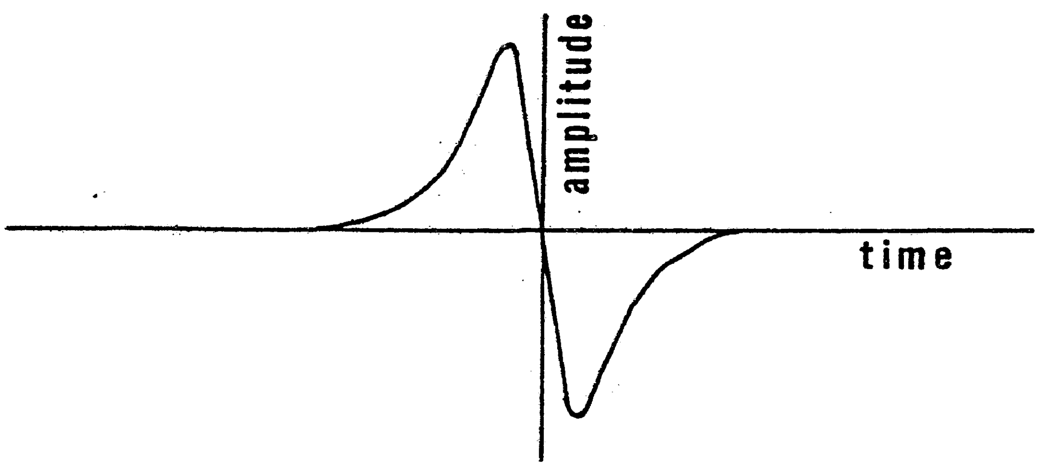


Figure 11 The doublet, a derivative operator.

where n is a constant.

A numerical technique was used to calculate the partial derivative of the synthetic with respect to the impedance of a given interval - both the first impedance and the impedance gradient. The numerical technique used is a left-finite difference:

$$\frac{d(s(t))}{dI_i} = \frac{(s(I+\Delta I_i) - s(I_i)}{\Delta I_i}$$

Vandell (1979) compared analytical partial derivatives with the above finite difference method for a very similar calculation and found that the finite difference method gave a very good approximation to the analytical derivative when ΔI_i was varied from $0.1I_i$ to $.001\ I_i$. The increment used here was $0.1I_i$.

The partial derivatives with respect to the wavelet parameters were also calculated using the above finite difference technique. The results were compared to an exact partial derivative with excellent results. The exact derivative did not take much longer to calculate, but it did require substantially more computer memory. For this reason the finite difference technique was used for all of the wavelet parameters.

The partial derivative of the synthetic seismic trace with respect to a scale factor is calculated using the exact analytical derivative. For any seismic trace:

$$s(t)=K \sum_{i=1}^{N} a_i \quad (t-T_i)$$

Where K is a constant scale factor

The partial derivative with respect to the scale factor is:

$$\frac{\partial s(t)}{\partial K} = \Sigma a_i \delta(t-T_i) = s(t)/K$$

This partial is extremely simple to calculate - it is just the scaled synthetic divided by the current scale factor.

GENERATION OF THE INITIAL GUESS

In order to generate equation 4 one must first have an initial guess of the impedance profile ($\bar{IG}$). Vandell (1979) generated his initial guess by doing a Seislog* inversion of the observed data and ignoring the effects of the source wavelet and mulitple arrivals. This will not work with the technique used in this paper due to the difference in impedance parameterization. If one was to "block" this type of initial guess it would be compatible with the parameterization used here. This was done with excellent results. Another method used here to generate an initial guess is to assign an increase in impedance to each positive polarity arrival in the observed data and a decrease in impedance to each negative polarity arrival. This method has the limitation that it will only invert the arrivals that one includes in the guess, but it will also allow for the interpreter to include lithologies in the guess that the blocked Seislog* method did not discern. When sequentially inverting a number of traces the guess for one trace would be the impedance solution from the adjacent inverted trace.

*Seislog is a registered trademark of the Technica Co.

Some guidelines for how exact a guess must be for the guess to converge on the correct solution were determined by experience. For the location of a given boundary, the guess must lie within a time x of the boundary's true location. The time x is equal to 1/2 of the central lobe's width for the source wavelet. The magnitude of the impedance change at a boundary is of little importance - one need only have the correct polarity change. The initial guess of the impedance gradient within boundaries was always set to zero. The initial guess for the source wavelet is usually the band pass filter on the data to be inverted.

CONSTRAINTS AND USE OF EXTERNAL INFORMATION

Equation 4 is a system of linear equations that is solved for a set of parameter corrections. In deriving equation 4 it is assumed that there is no a priori information concerning the parameters to be corrected. It often happens that a seismic interpreter will have quite a bit of useful information concerning these parameters. This information can be an interpreter's knowledge of some lithology or it might reflect physical constraints within which the parameters must lie. There are two ways that one can use this external information. The first method involves how to use the parameter corrections solved for and the other is actually a modification to equation 4.

The parameter corrections supplied by equation 4 often can lead to nonsensical results. For example it is theoretically possible for some solution to give an interval a negative thickness or impedance. This sort of result does not necessarily mean that the inversion process is unstable. Such unrealistic results are somtimes encountered in an iterative procedure when the corrections in a give iteration overshoot the answer. The corrections from the following iteration would tend back towards the correct solution and thus remove the nonsensical results (that is unless the forward model blows up when supplied with nonsensical parameters). It is possible, though, to avoid the nonsensical results and lessen the number of iterations needed by constraining the corrected parameters to lie within realistic bounds. In this manner all lithologies are constrained (when the corrections are added) to have a thickness greater than zero. Each individual impedance correction is constrained to lie between plus or minus 5,000 gm/cm^3 ft/sec to avoid large oscillations. Note that this type of constraint is applied exterior to equation 4 and thus is not used in calculating a given correction vector.

As mentioned above one can modify equation 4 to reflect knowledge about any given parameter. One way of doing this assumes that a parameter(s) in the initial guess is exact and should not be changed. Furthermore, the solution for other parameters should be based on this exact parameter. To do this the partial derivative with respect to this parameter can be set to zero or it can be removed entirely from the sensitivity matrix. If the partial derivative is removed then a correction for that parameter will not be included in the correction vector. If a partial derivative is set to zero the parameter correction will automatically be zero. The latter method is computationally slower but it requires less bookkeeping in any computer scheme. Figures 26-27 use this latter method of fixing a given parameter.

RESULTS

Figures 12A and 12B show an example of generalized linear inversion. This is the simplest inversion problem possible and is shown here for illustrative purposes. In this example the initial guess for the boundary location was on the edge of the region of convergence. The initial guess of the scale factor and source wavelet are correct and they are not solved for in this example. The solution took 9 iterations to converge, but only because the boundary corrections were constrained to moving one time unit (2 milliseconds) per iteration.

In this and most of the following examples six traces are displayed. The first three traces (as in Figure 12A) are impedance profiles. The earth's impedance profile is used to compute the trace to be inverted. The initial guess of the earth is the user's interpretation of the impedance profile that corresponds to the observed data. The iterative impedance solution is the generalized linear inversion refinement of the initial guess. If the inversion is exact the solution will be identical to the earth's impedance. The second three traces (as in Figure 12B) are synthetic seismograms computed from the impedance profiles.

Figures 13 (A,B,&C) are basically the same as the previous example except that an incorrect initial guess for the source

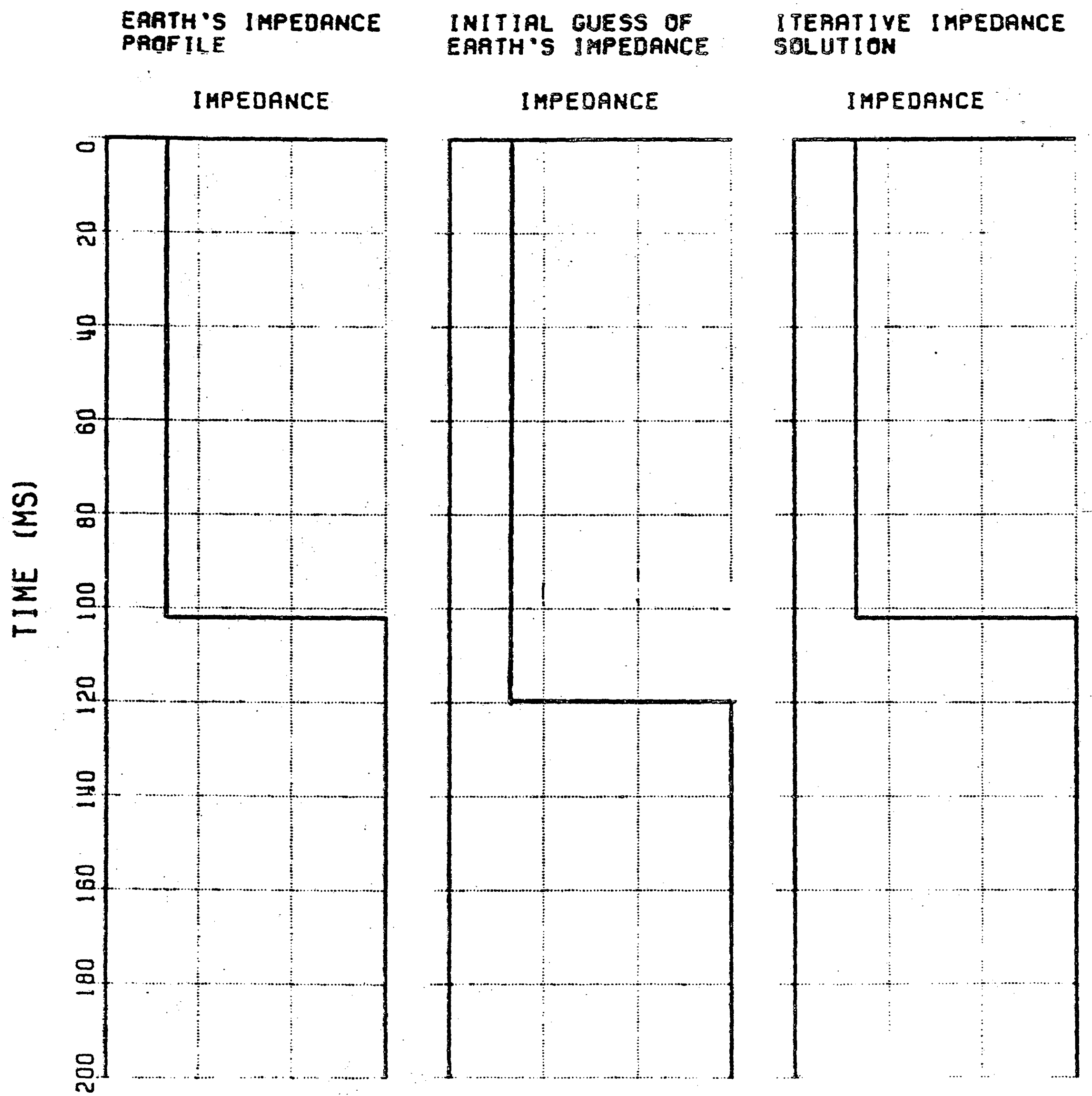


Figure 12A Inversion of a single reflector.

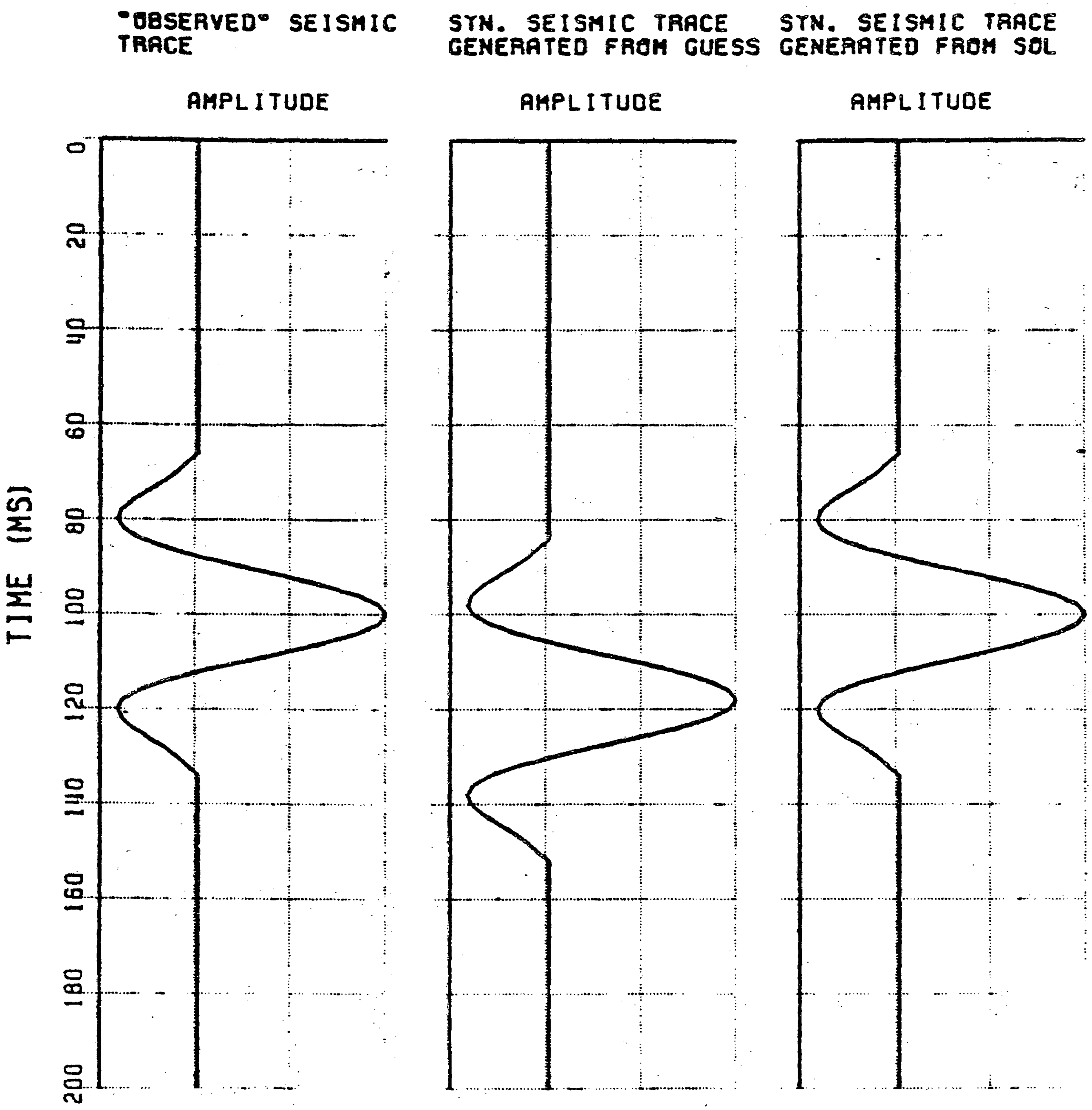


Figure 12B Inversion of a single reflector.

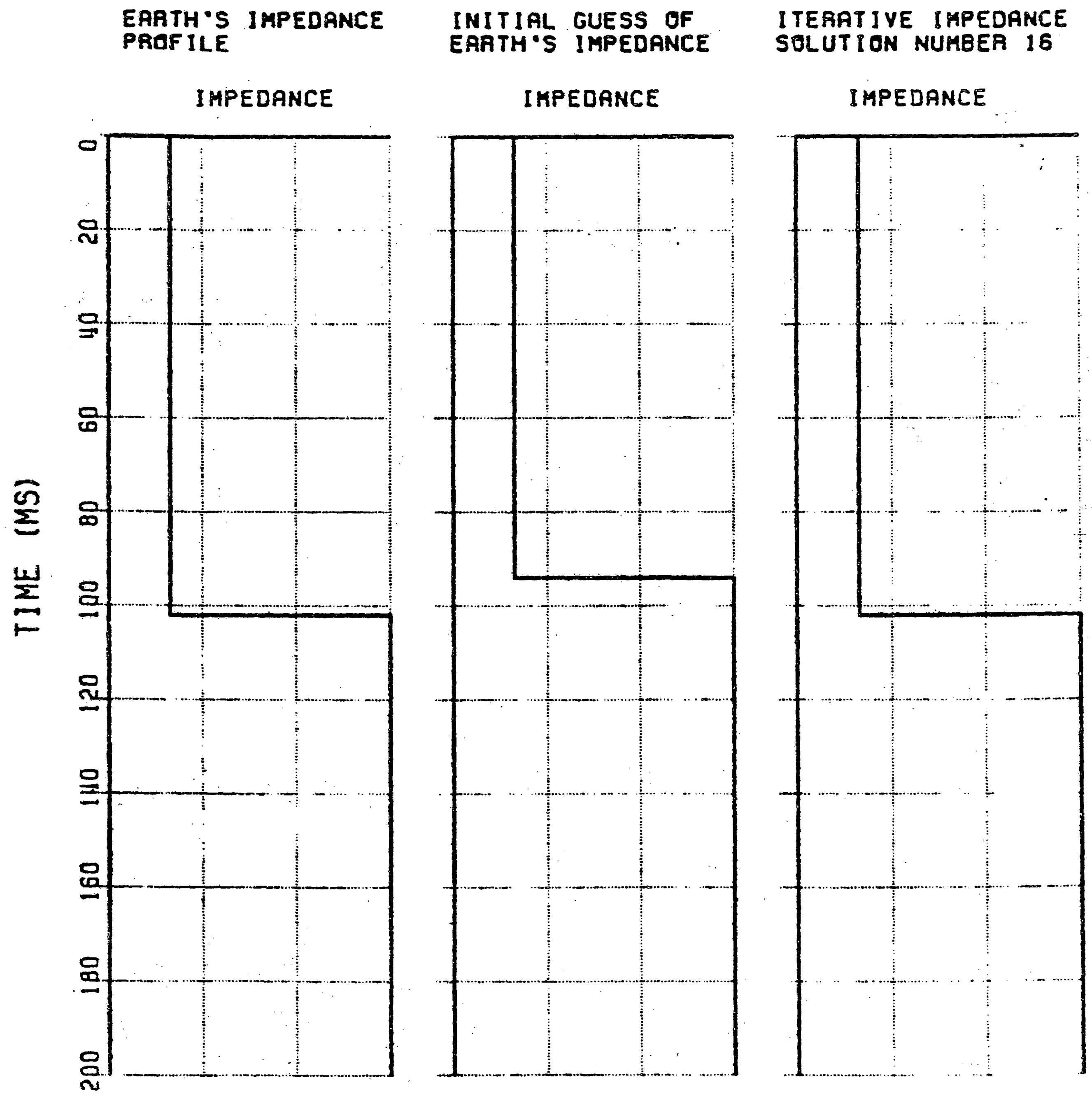


Figure 13A Inversion for the source wavelet and impedance profile.

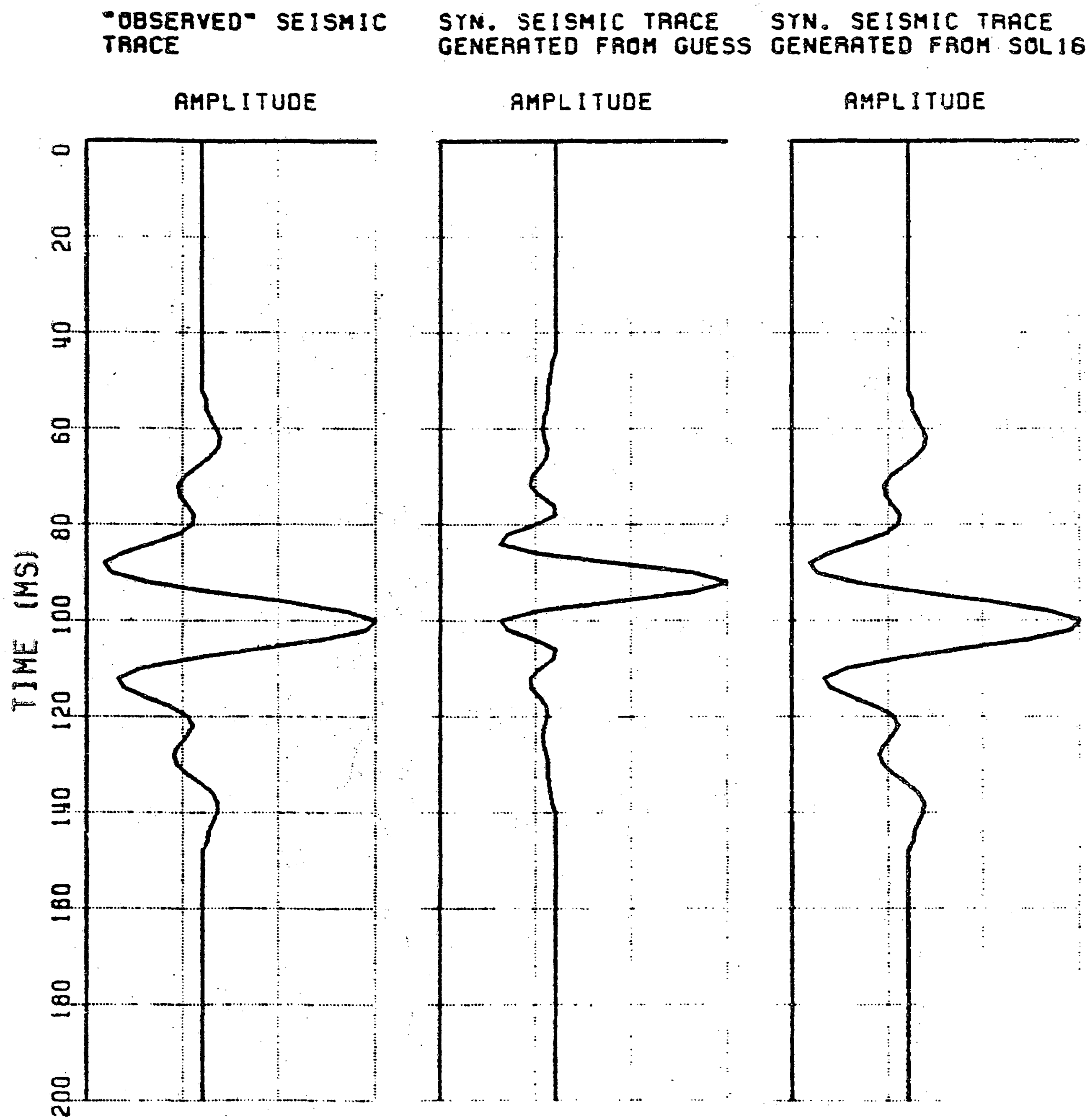


Figure 13B Inversion for the source wavelet and impedance profile.

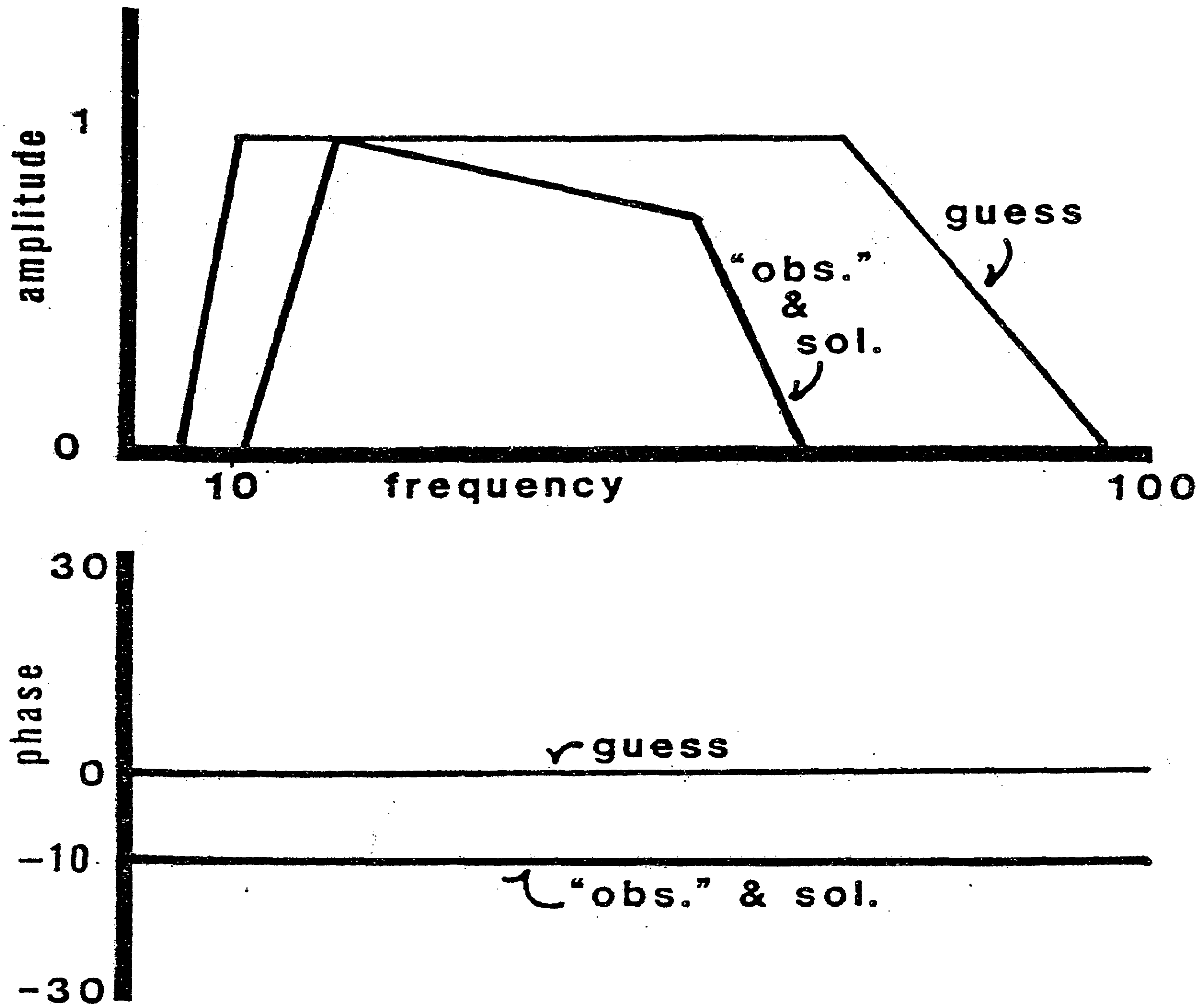


Figure 13C. Frequency domain representation of the guess and solution for the source wavelet in Figure 13B.

wavelet is used and the program solves for source wavelet. The solution wavelet exactly equals the "observed" wavelet as illustrated in Figure 13-C.

Figures 14A and 14B show how the lithology dependent parameterization has lessened the need for an exact description of the source wavelet. This is a repeat of the inversion done in Figure 7 in which discrete interval digitization was used. In Figure 14 the observed wavelet was a 5-10-115-120 band pass filter with a 22.5 degree phase shift. The guess wavelet had the same band pass but was zero phase. In the inversion the source wavelet was not solved for. Note that the inversion results are not perfect, but they are far superior to those in Figure 7 (Page 26).

In Figures 15A and 15B the boundary location, impedances and wavelet are solved for, but in this case the observed data has simulated noise added to it. The noise is white and the signal to noise ratio is 2.0 (this is rms s/n as calculated over the entire 200 millisecond window). The presence of the noise has a slight effect on the results. The solution source wavelet is slightly incorrect and the impedance of the lower interval is a bit larger than it should be.

Figures 16A and 16B show an example where there is a linear trend in the earth's impedance. The algorithm solved exactly for the earth's impedance profile. In this example there was no noise and the initial guess was generated by assigning an impedance contrast to each of each of the four

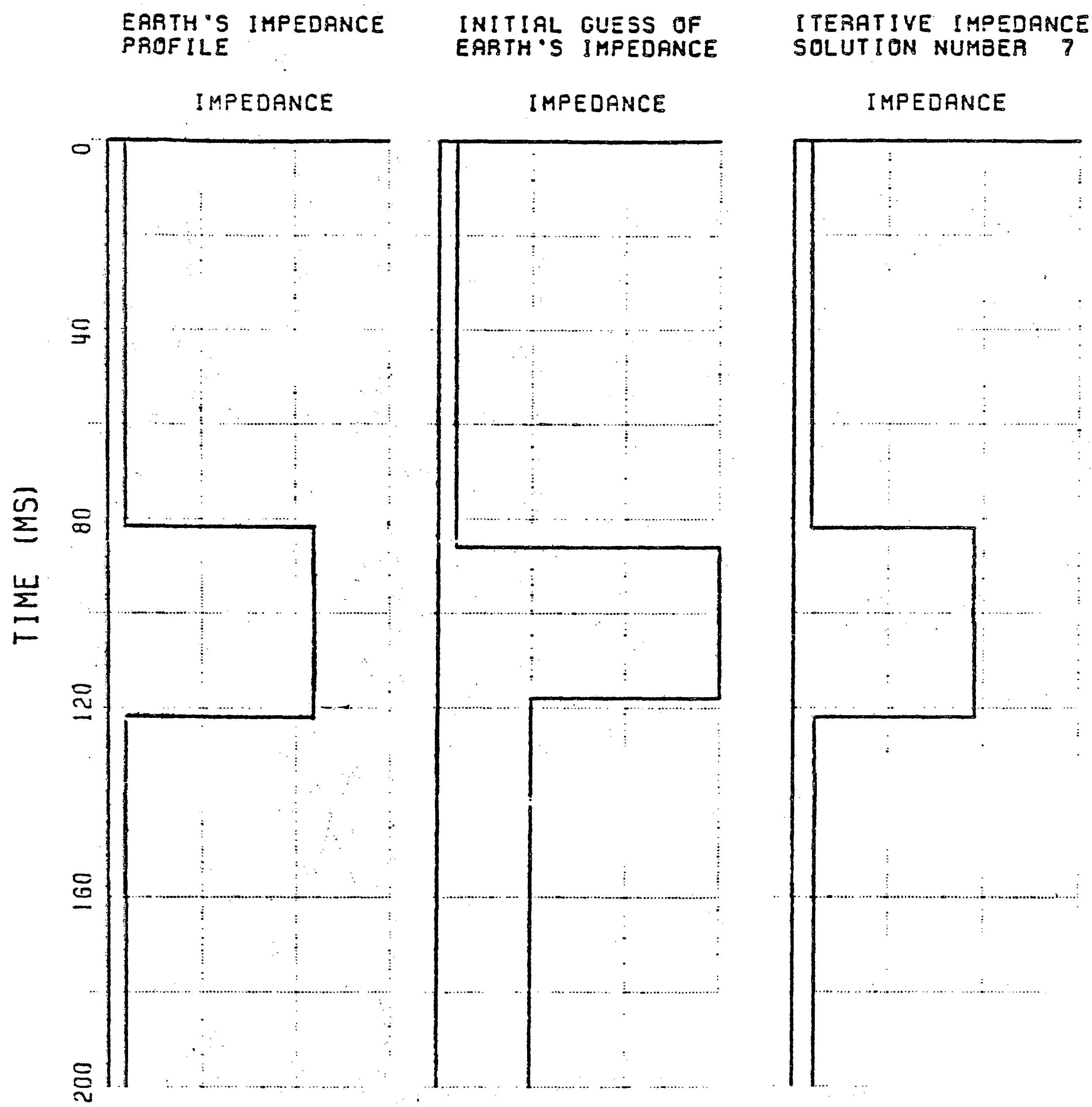


Figure 14A Inversion where source wavelet is not solved for.

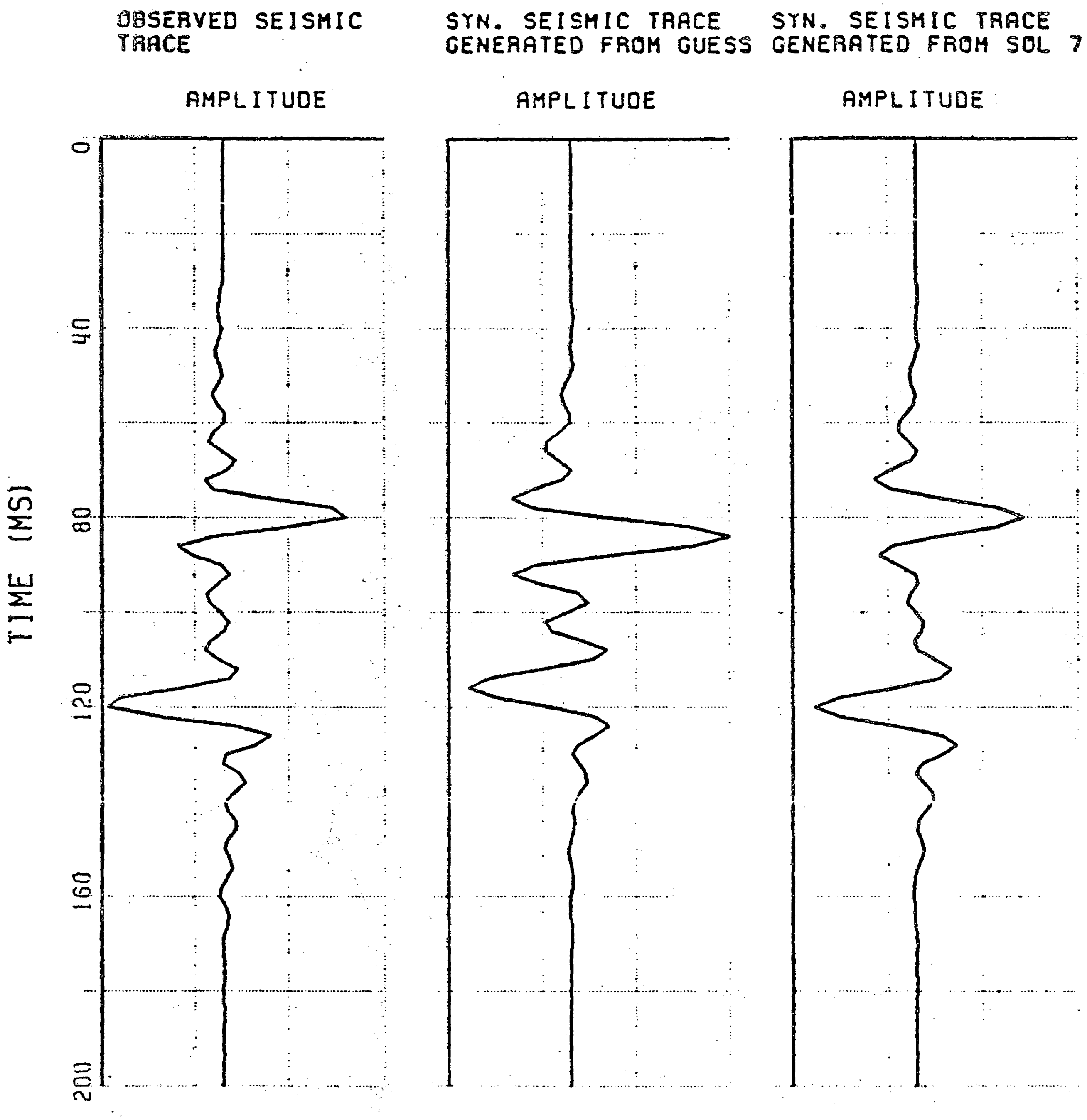


Figure 14B Inversion where source wavelet is not solved for.

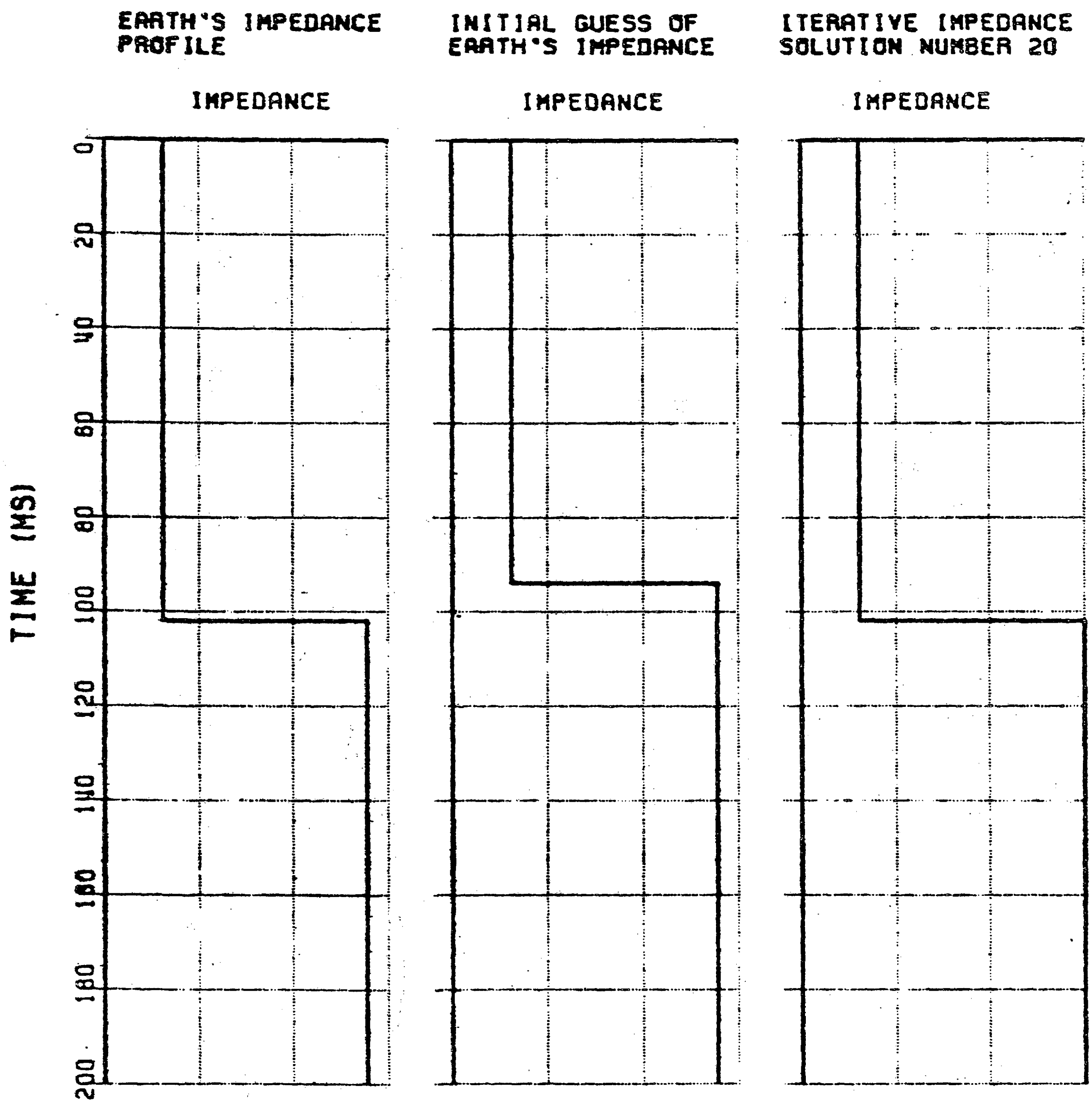


Figure 15A Inversion for source wavelet and impedance profile in presence of noise.

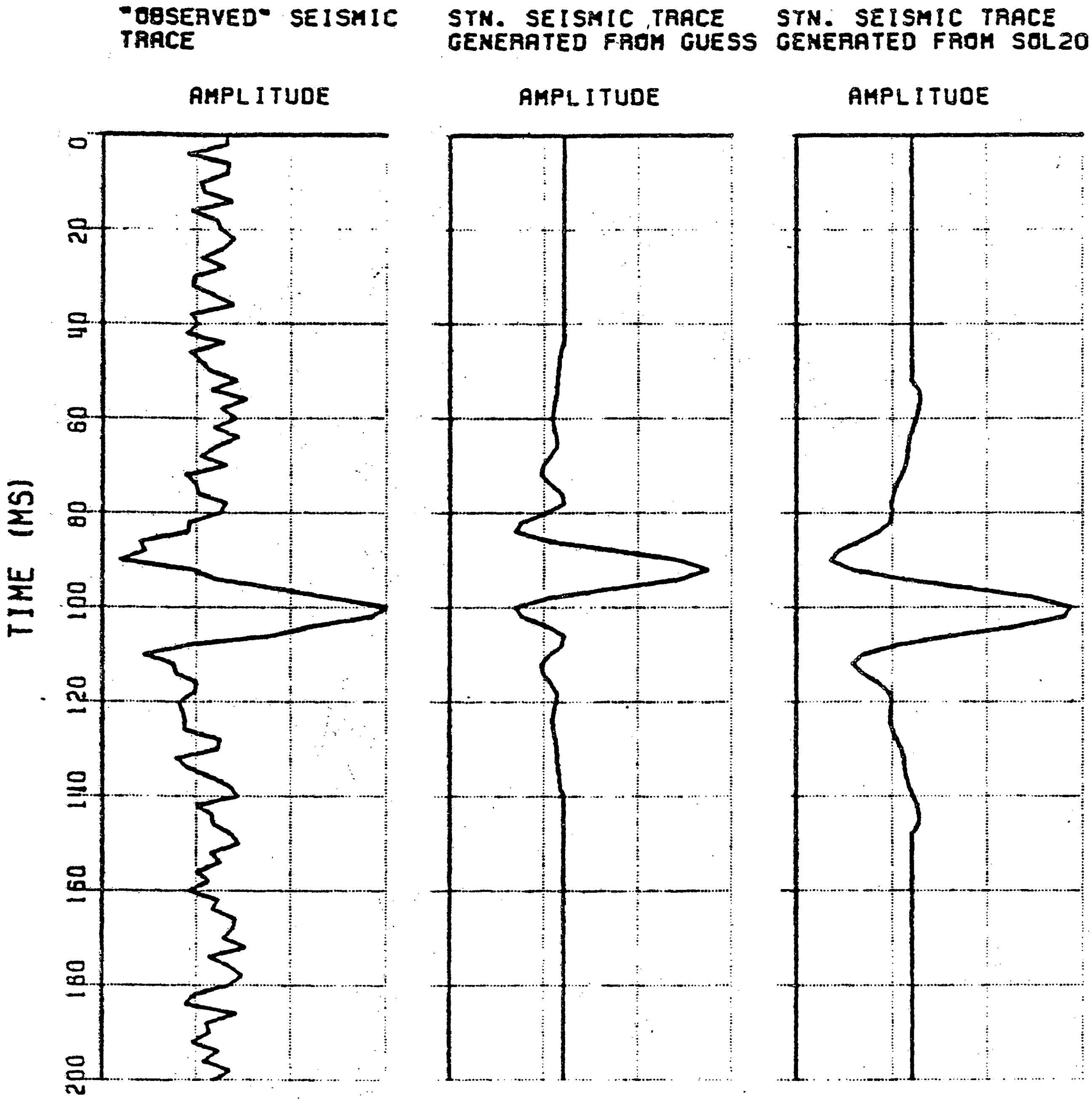


Figure 15B Inversion for source wavelet and impedance profile in presence of noise.

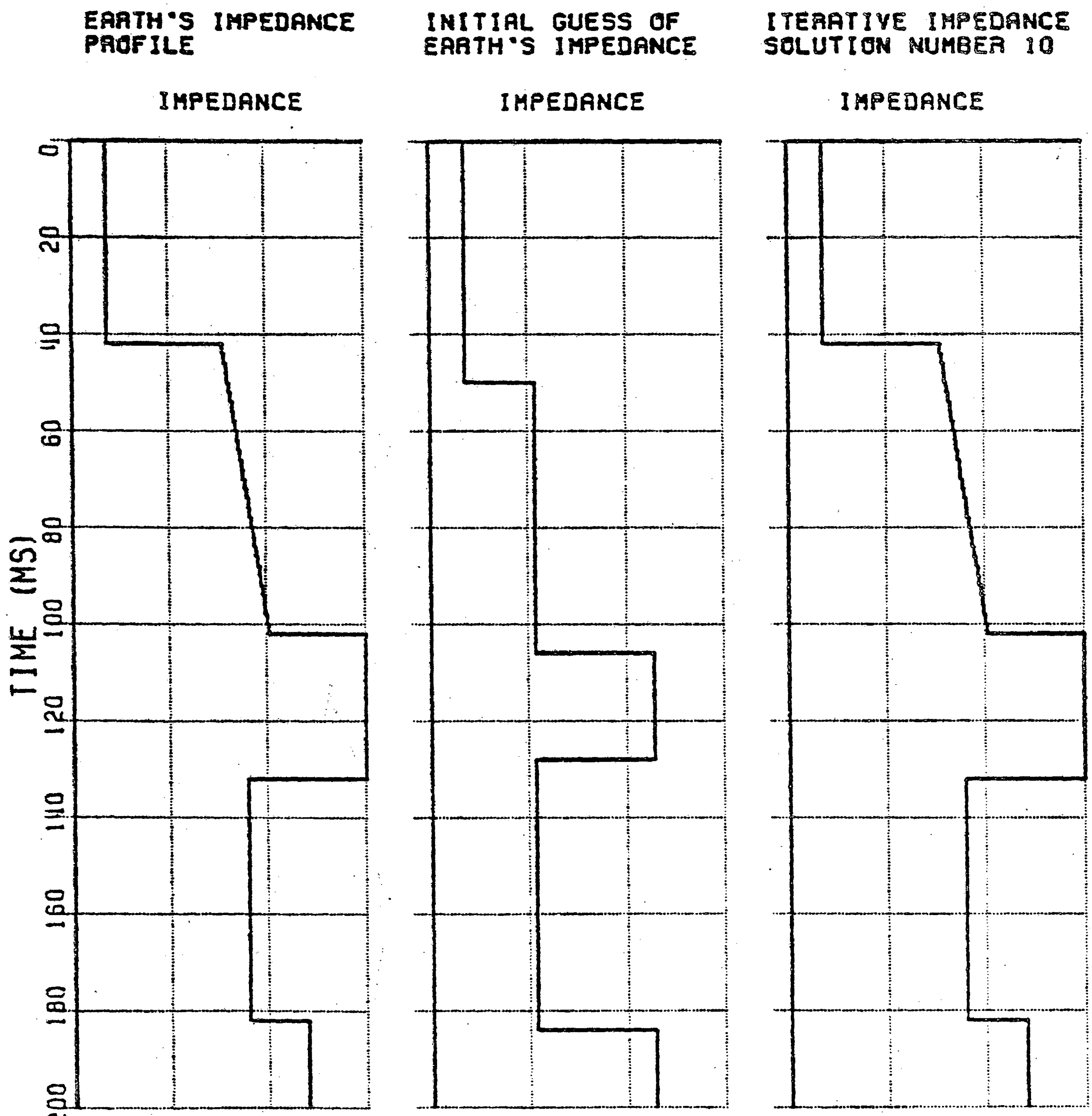


Figure 16A Inversion for an impedance profile that contains an impedance gradient in one interval.

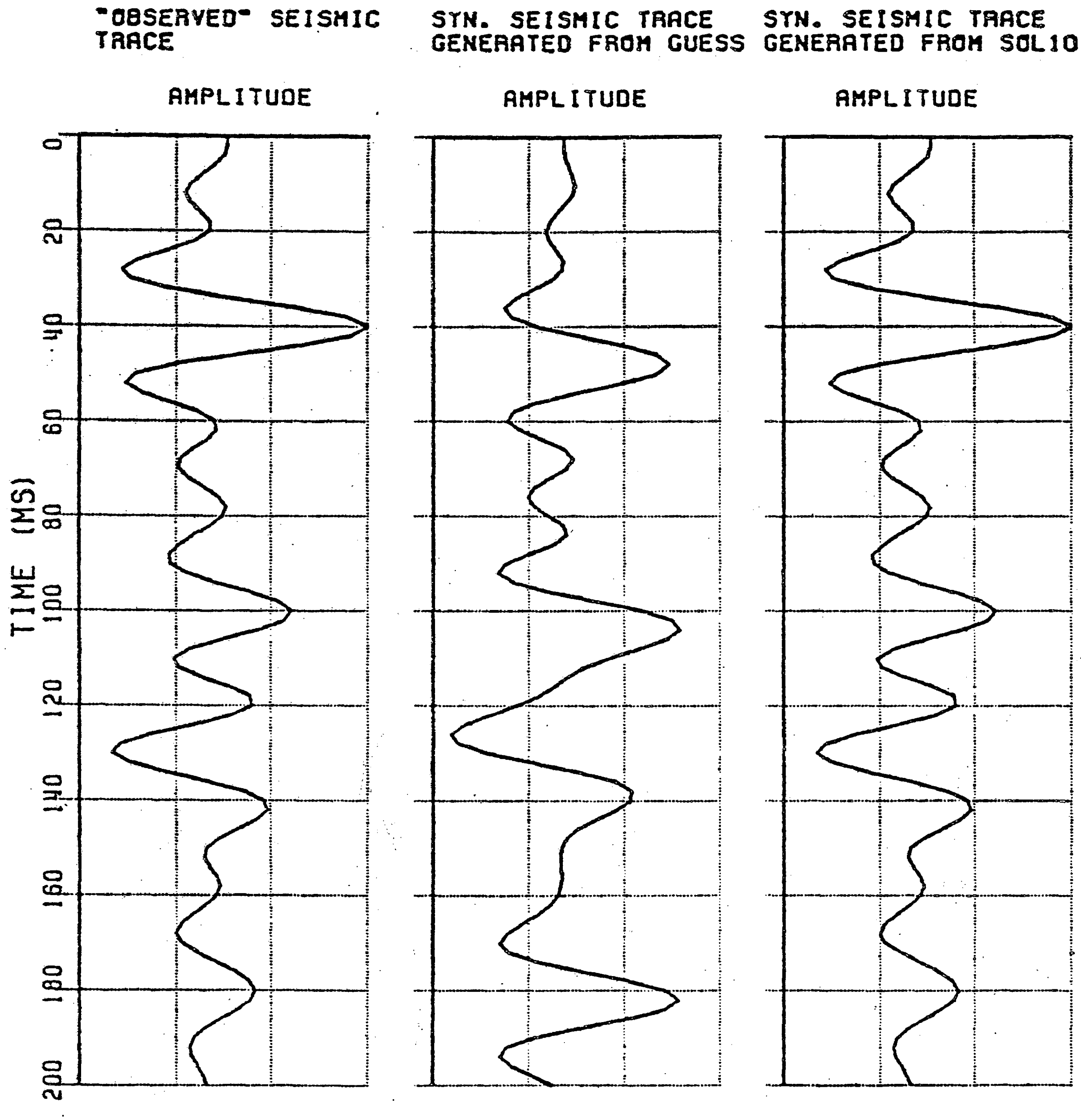


Figure 16B Inversion for an impedance profile that contains an impedance gradient in one interval.

largest arrivals. The boundaries in the guess were then moved to simulate an incorrect guess. The guess of the impedances within each interval is arbitrary except that the polarity of the impedance contrast at each boundary matches the polarity of the four arrivals used. The guess of the impedance gradient in each interval is zero.

When generating an initial guess, it is of some concern whether a given peak or trough is a real arrival or noise - a multiple arrival, side lobe of a real arrival or a tuning effect between two arrivals. Figures 17A and 17B show that one can construct an initial guess with the false assumption that noise is a true primary arrival. In the 'observed' data of this example there are two multiples arrivals - at 180 and 275 milliseconds. In generating the initial guess an impedance contrast was assigned to both of these multiples arrivals and in the final solution both of the contrasts were removed. Note also that the location of all boundaries in the guess have some error.

In Figures 18A and 18B the scale factor, along with the boundaries and impedances, is solved for. This example shows how the problem presented in Figure 9 is solved. Figure 9 shows the error that results when the data is incorrectly scaled. The "observed" trace of Figure 18B is the same as the observed trace of Figure 9 except Figure 18B has a source wavelet in it. The scale factor in both cases is 2.0, and in

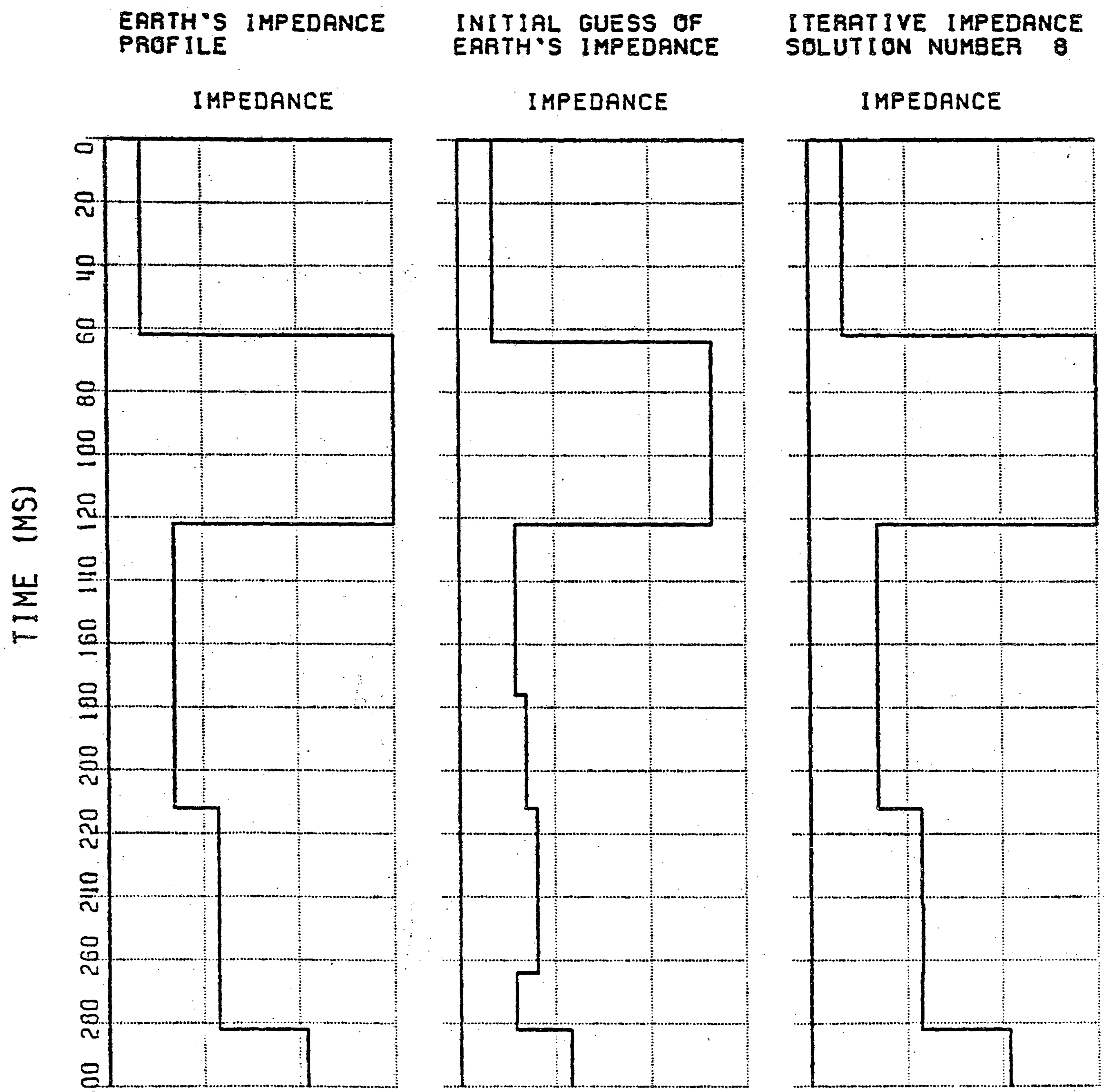


Figure 17A Inversion where initial guess has more boundaries than earth.

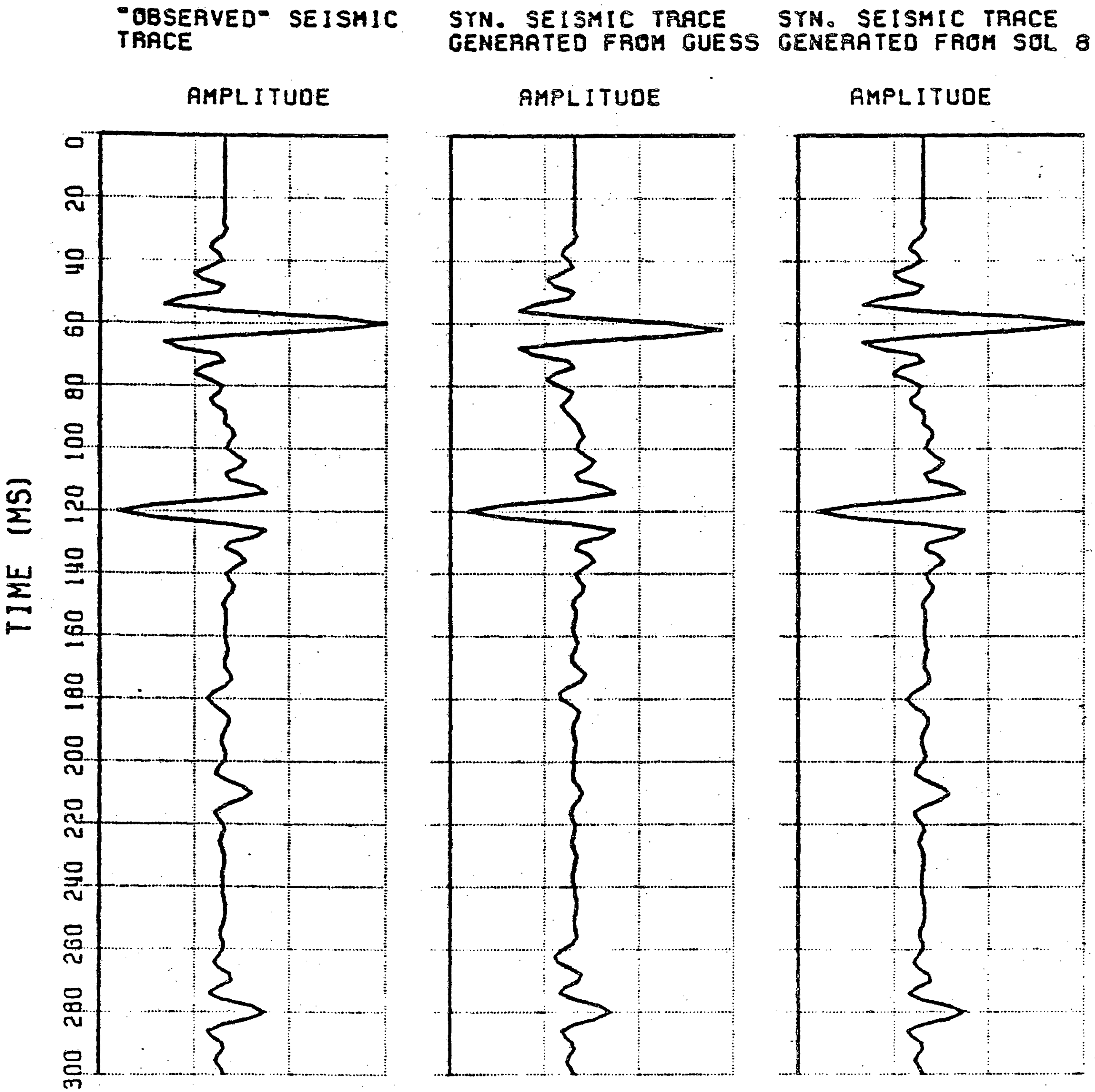


Figure 17B Inversion where initial guess has more boundaries than earth.

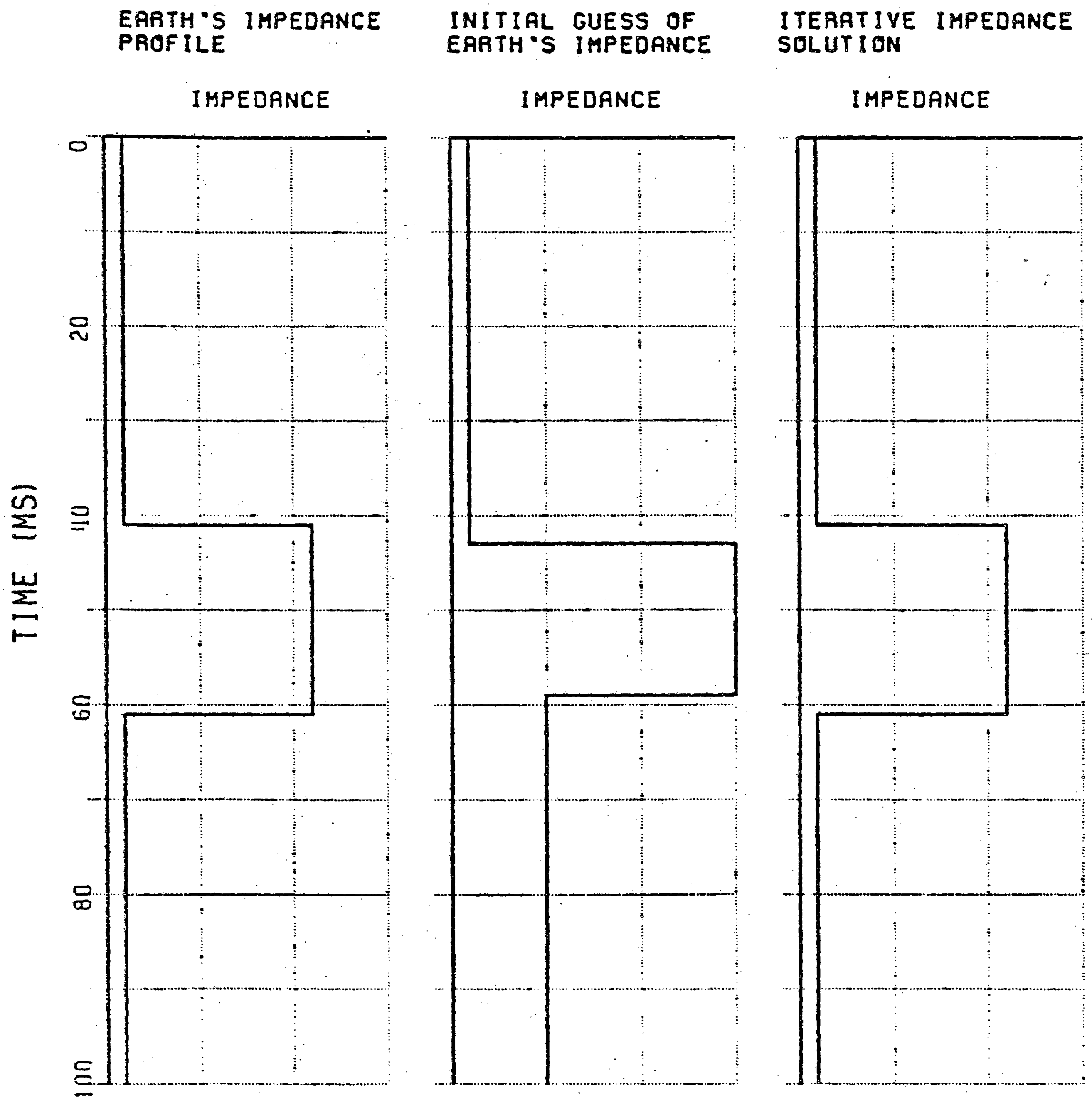


Figure 18A Inversion where scale factor is solved for.

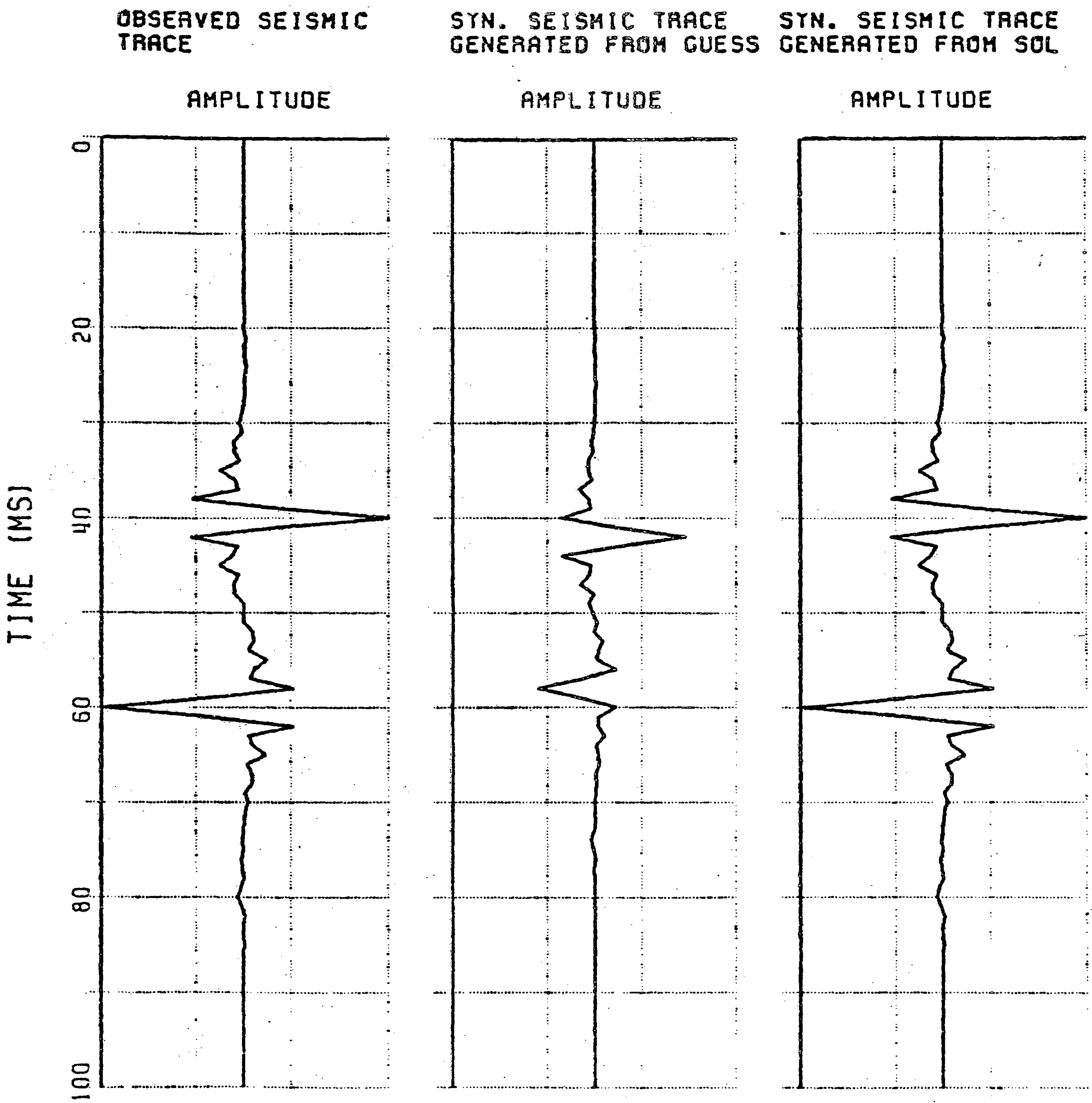


Figure 18B Inversion where scale factor is solved for.

the latter example it is solved for which removes the erroneous result of the former example.

Figure 19 through Figure 27 show a hypothetical example of how generalized linear inversion could be applied to an entire seismic line instead of one trace at a time as in all of the above examples. Figure 19 shows the geologic setting. Two wells have been drilled on the profile, and one has encountered a gas sand encased in shale. On the side of the figure is an impedance log as a function of time over the 300 millisecond window of interest. Figure 20 shows the two main categories of seismic traces that would be found in a seismic line shot over the profile in Figure 19. One trace is shot over the gas sand and the other traces misses it. Complicating things is the fact that a multiple arrives exactly where the top of the gas sand is expected. This example is noise free. To decide which of these traces really shows the gas sand, both are inverted using the initial guess shown in Figure 21. Note that this initial guess does have the gas sand. Figure 22 shows the results - both traces are inverted perfectly and show the occurrence (or lack of it) of the gas sand.

The above example was done in the absence of noise and is thus a bit unrealistic. Figure 23 is the same as Figure 19 except that noise has been added to make it more realistic. The noise is shown in Figure 24. This is filtered random noise where the filter used is the source wavelet for the

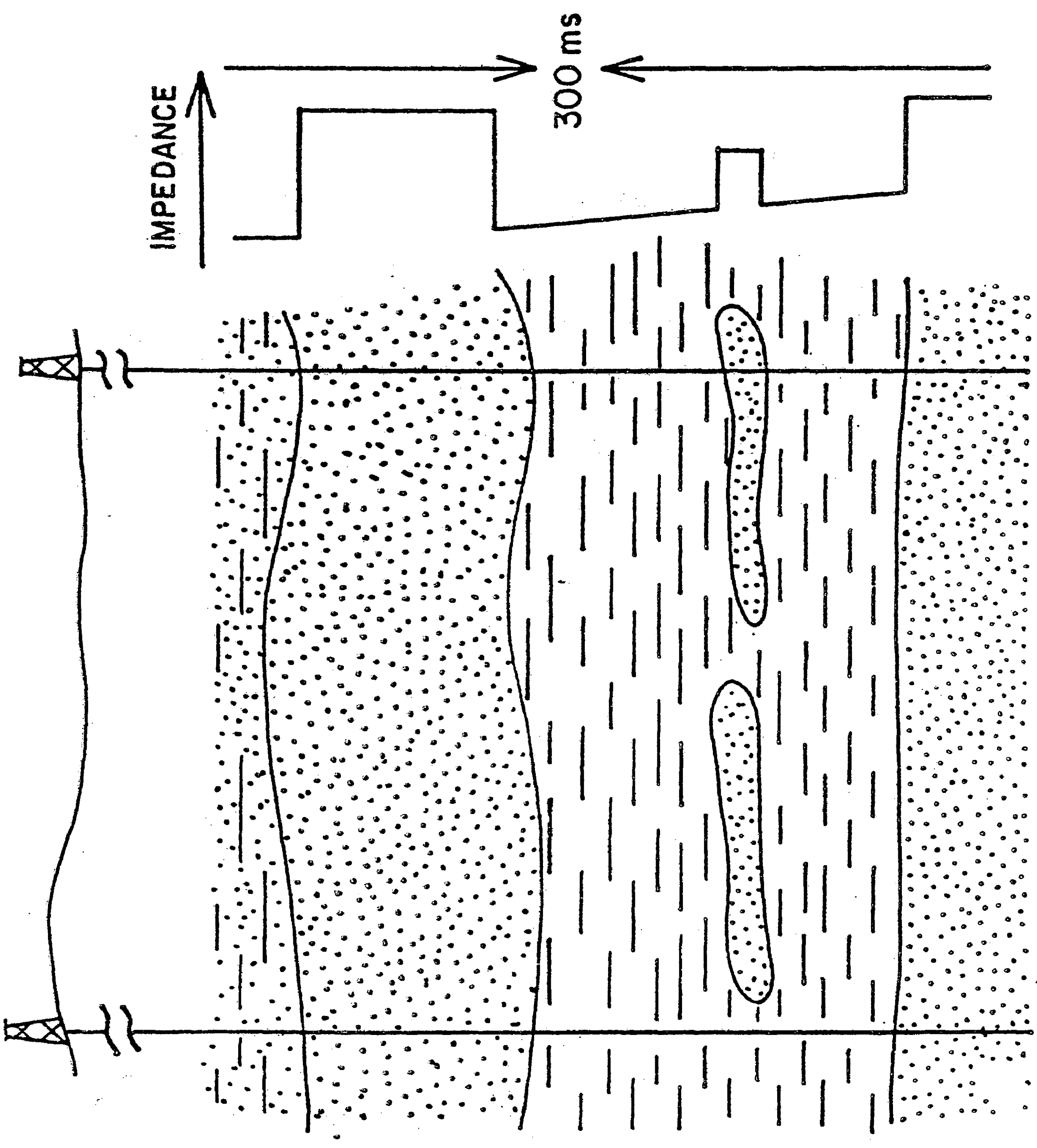


Figure 19 Geology setting for Figures 19-26.

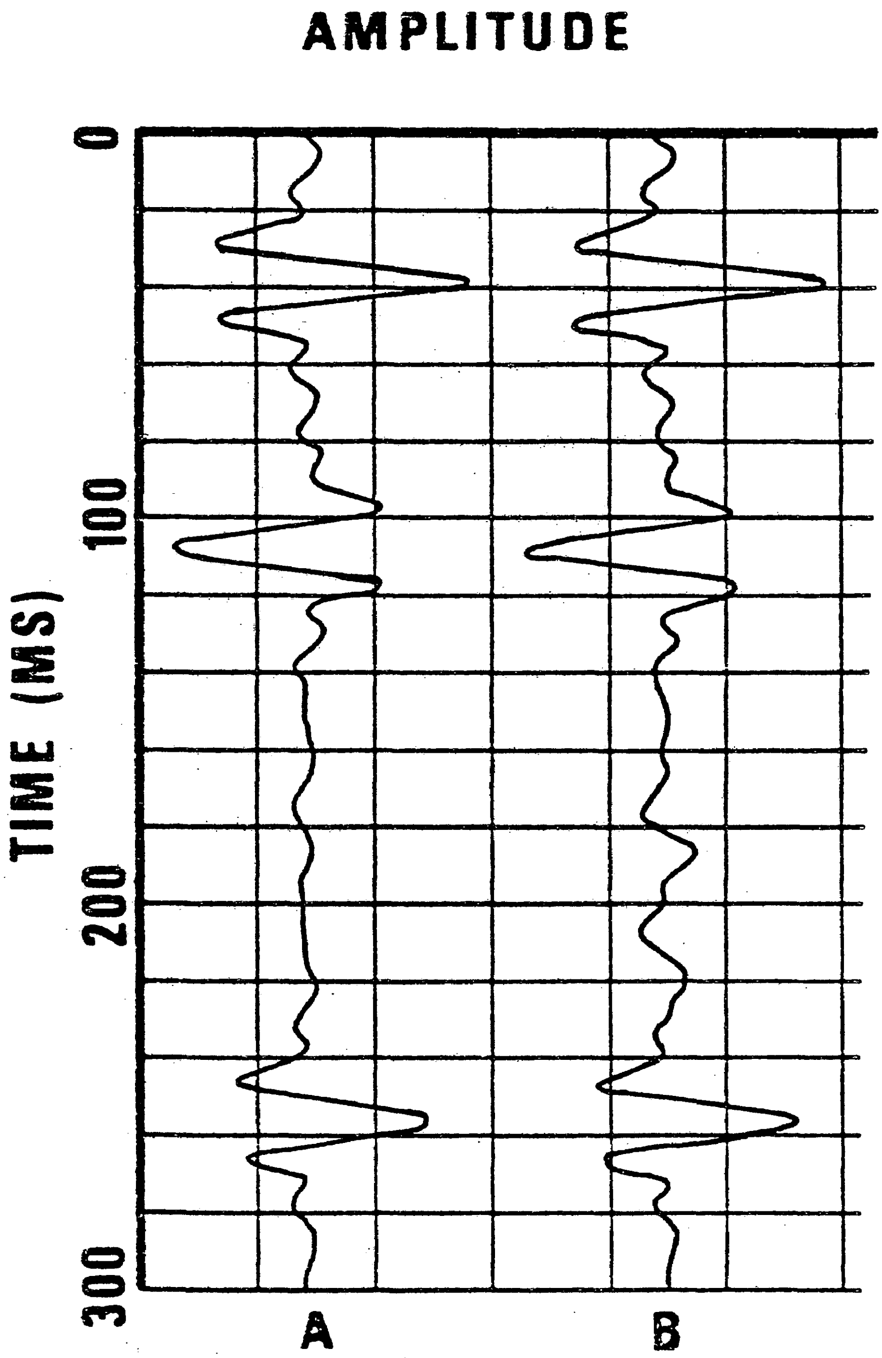


Figure 20 Two traces taken from a seismic line shot over Figure 18. Trace A does not sample the gas sand while trace B does.

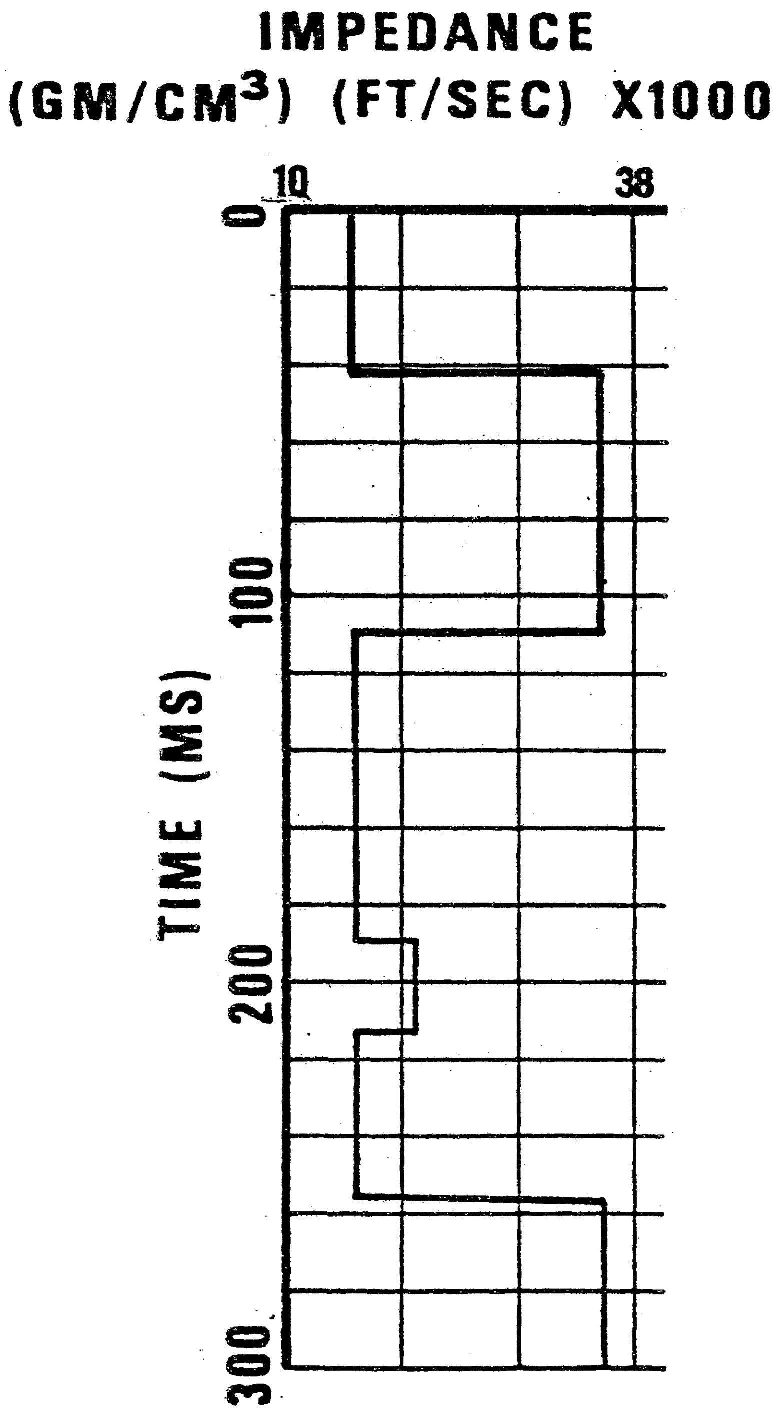


Figure 21 Initial guess used to invert both traces in Figure 19.

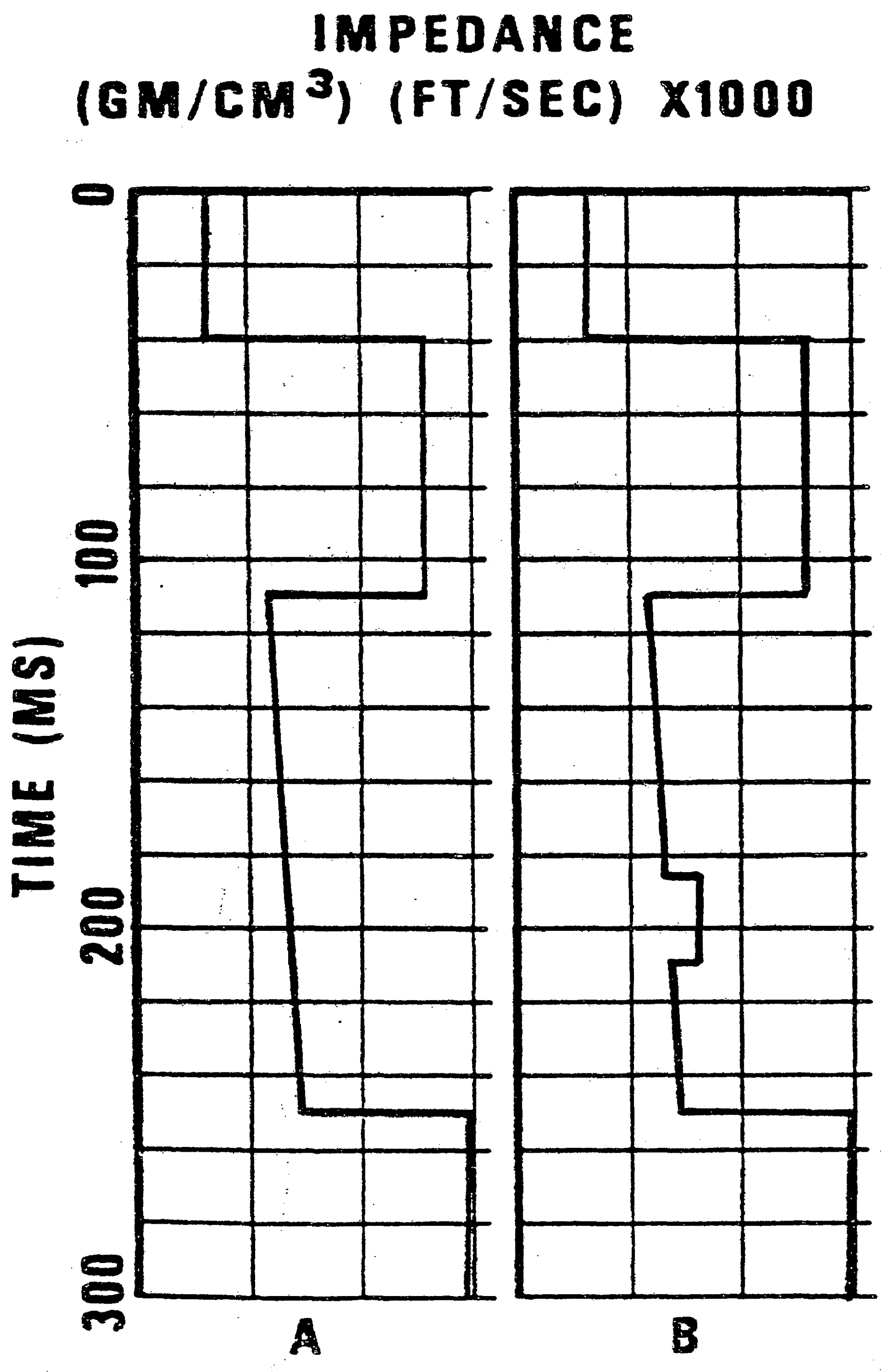


Figure 22 Inversion of both traces in Figure 19 using the initial guess of Figure 20. Trace A does not show the gas sand while trace B does.

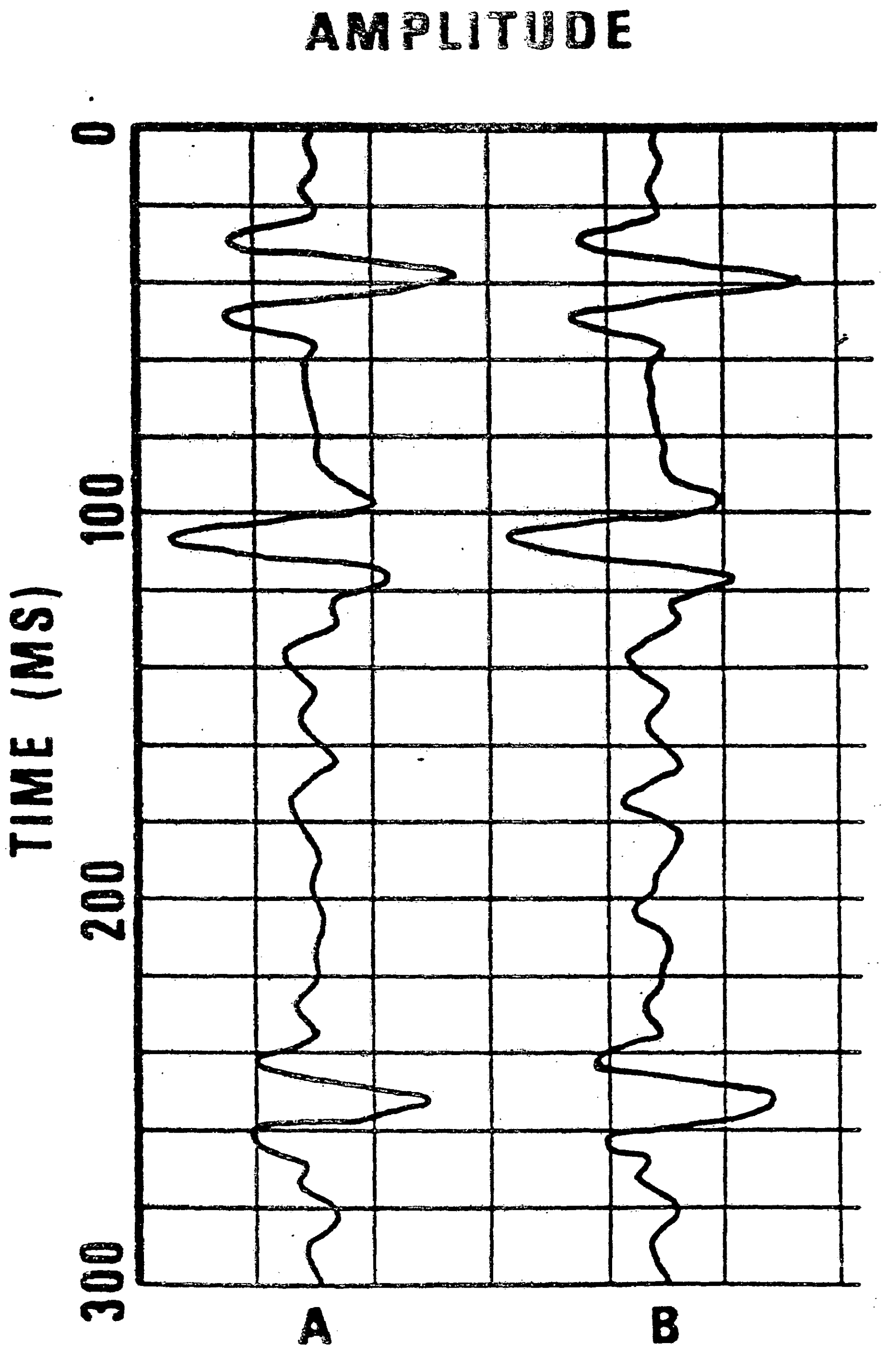


Figure 23 Two traces taken from a seismic line shot over Figure 18. The filtered random noise of Figure 23 is added to both traces. Trace A does not sample the gas while trace B does.

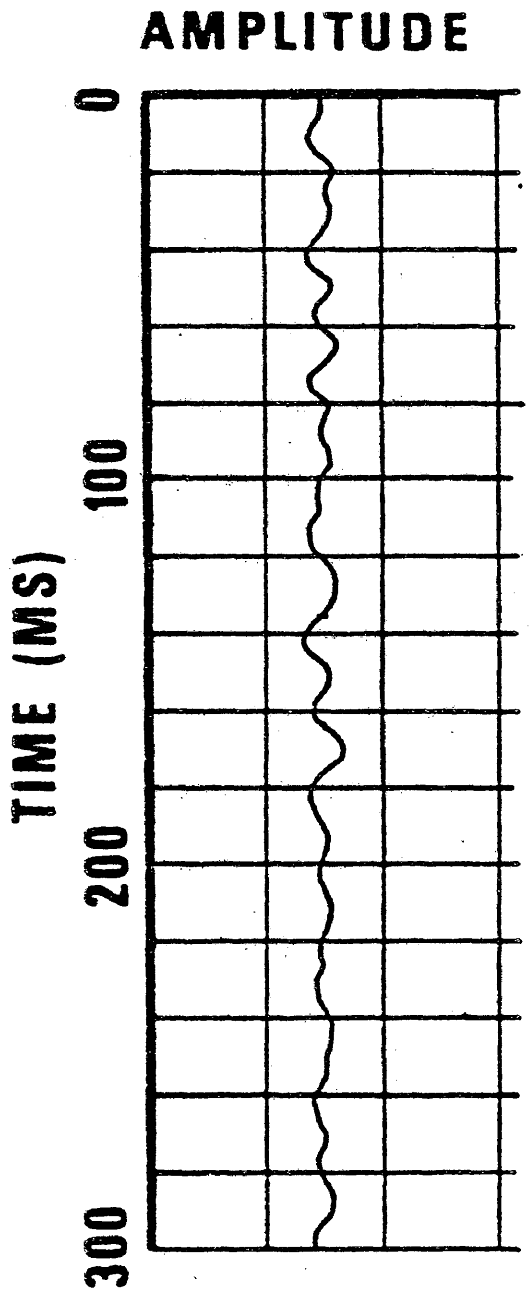


Figure 24 Filtered random noise that is added to Figure 20 to give Figure 23.

synthetic traces. Note that the noise has its highest amplitude exactly where the expected gas sand and multiple arrivals are. Both of these traces are inverted using the initial guess shown in Figure 21. The results of this inversion are shown in Figure 25. Note the deterioration of the results in the lower half of the window. The impedance gradients are unrealistically high giving rise to an impedance at the bottom of the window that is more than twice as high as in the real earth. Note that in the solution all boundary locations are correct and the impedance contrasts between lithologies are approximately correct.

To correct the errors found in Figure 25 some constraints were applied as in the manner discussed in the section on constraints and use of external information. The first constraint applied was to fix all impedance gradients to zero. The results of this are shown in Figure 26. This constraint has removed the undesirable results of Figure 25 but a new type of error results. Note that the slight impedance gradient that really exists in the model is absent in this inversion result. This may not seem like a problem in this example, but actual impedance logs show frequent and sometimes quite large impedance gradients within a single lithology. In such a case it may be very undesirable to constrain the gradients to zero. A different (and more realistic in this case) constraint is to fix the impedance of the last interval at its logged

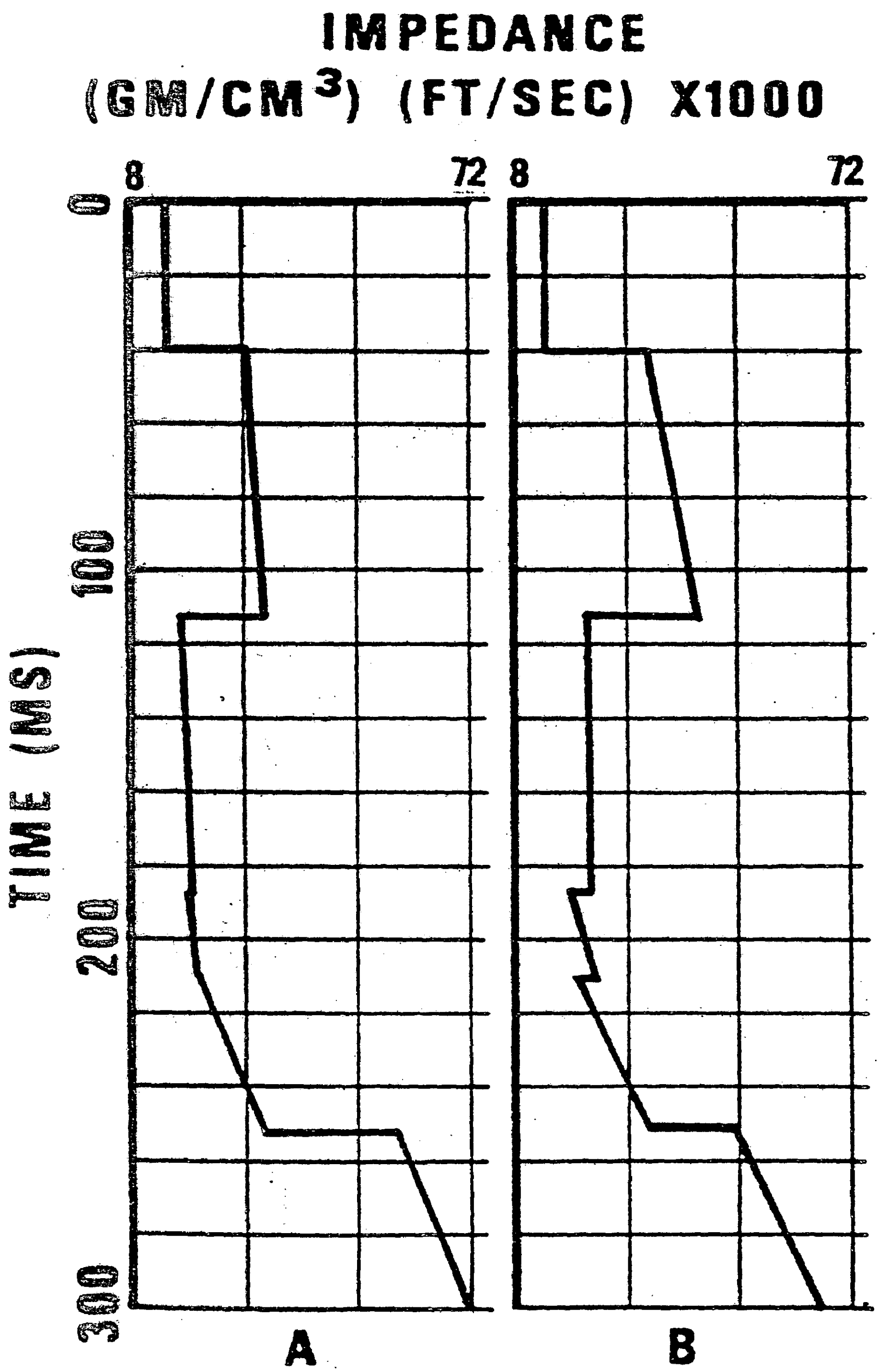


Figure 25 Inversion of both traces in Figure 22 using the initial guess of Figure 20. Inter-boundary impedance gradients are allowed in the solution.

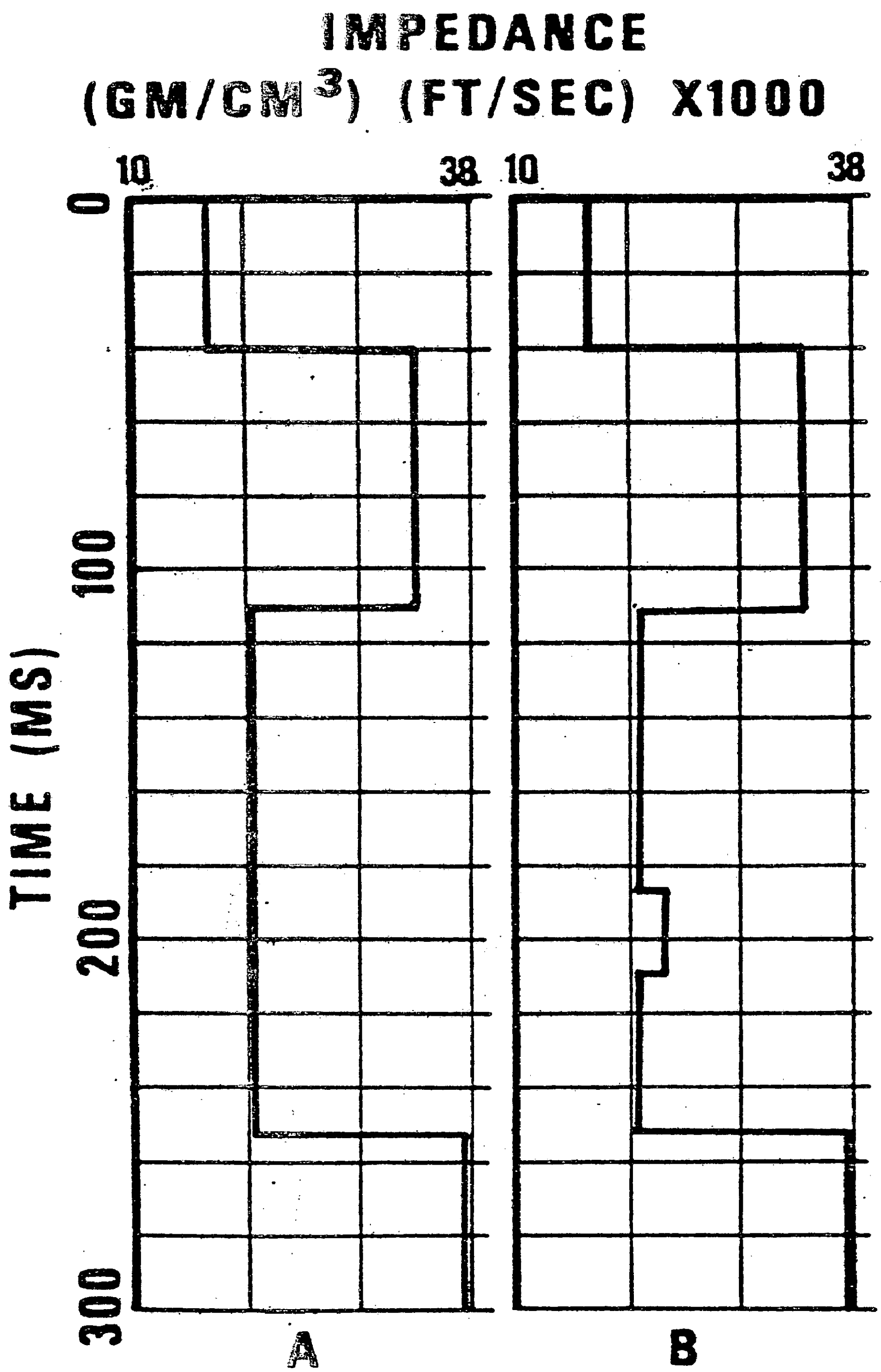


Figure 26 Inversion of both traces in Figure 22 using the initial guess of Figure 20. Inter-boundary impedance gradients are not allowed in the solution.

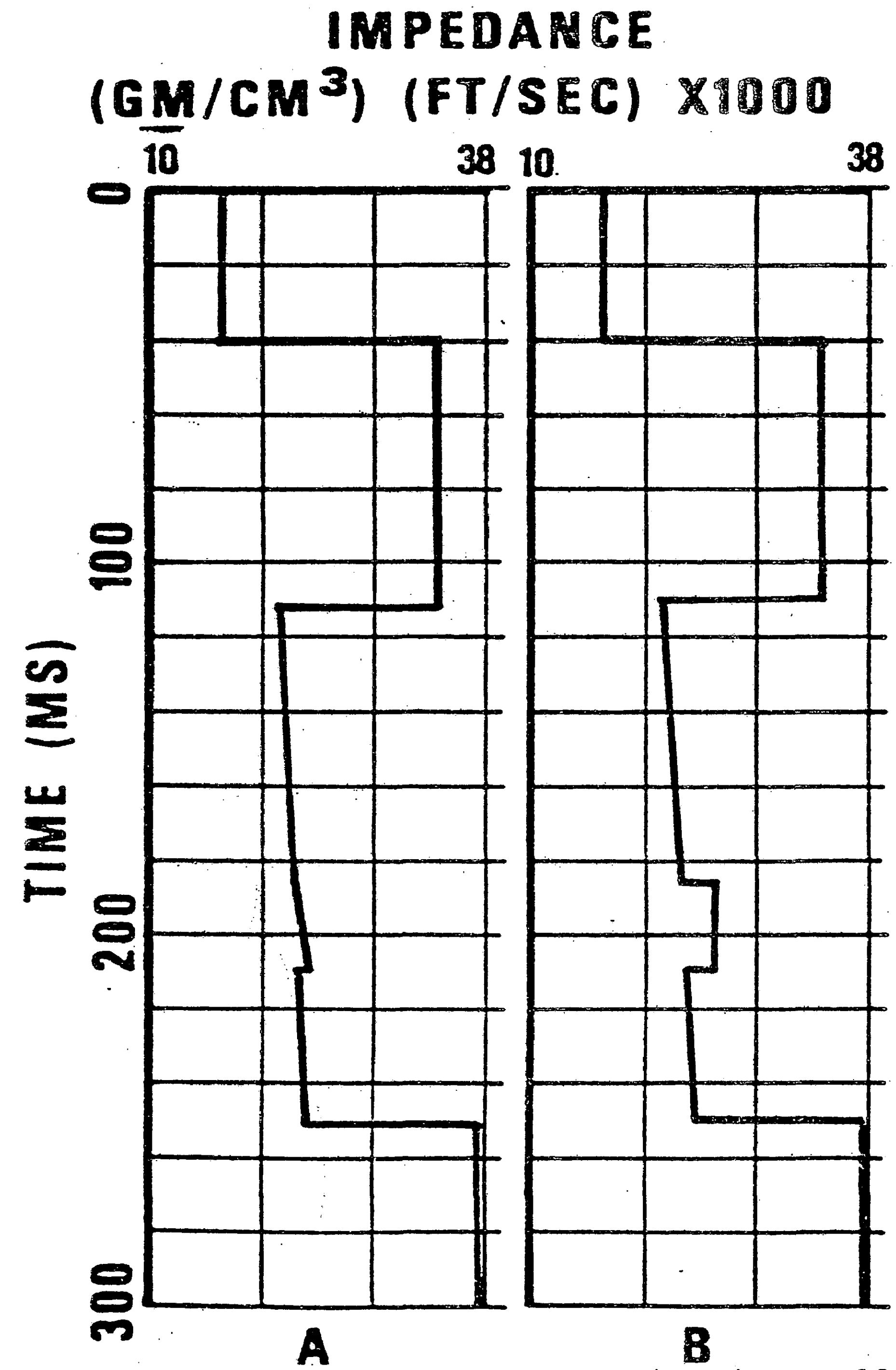


Figure 27 Inversion of both traces in Figure 22 using the initial guess of Figure 20. Inter-boundary impedance gradients are allowed and the solution for the last interval is fixed.

value. No boundary locations or impedance gradients are fixed. The results (Figures 27A and 27B) show a slight error compared to the correct answer, but there is no problem in identifying the gas sand.

Figures (28) through (35) show an example where a real seismic trace is inverted. Figure 28 shows the seismic trace to be inverted and an impedance log recorded near by. (All densities were assumed to be 2.0 gm/CM^3 to generate this impedance log.) The data shown are from a window, the top of which is at 1.0 second. The well was not logged below the 290 millisecond point in the window. The impedance log is shown here for comparison with the inversion results. The seismic data is a C.D.P. trace that has been processed to preserve true amplitude and zero phase charcteristics. The data were recorded at a one millisecond sample interval, resampled at two milliseconds and has been filtered with a 5-15-80-105 trapezoidal band-pass filter. This was the same filter that was used for a source wavelet in the inversion process. The wavelet was not solved for in the following examples.

The observed trace in Figure 28 was inverted using the initial guess found in Figure 29. The result is shown on the right of Figure 29. This initial guess was generated by assigning a boundary where impedance increases for every peak and a decrease for every trough found on the observed data. The impedance of every interval was arbitrary. The guess of the first (shallowest) interval is taken from the impedance

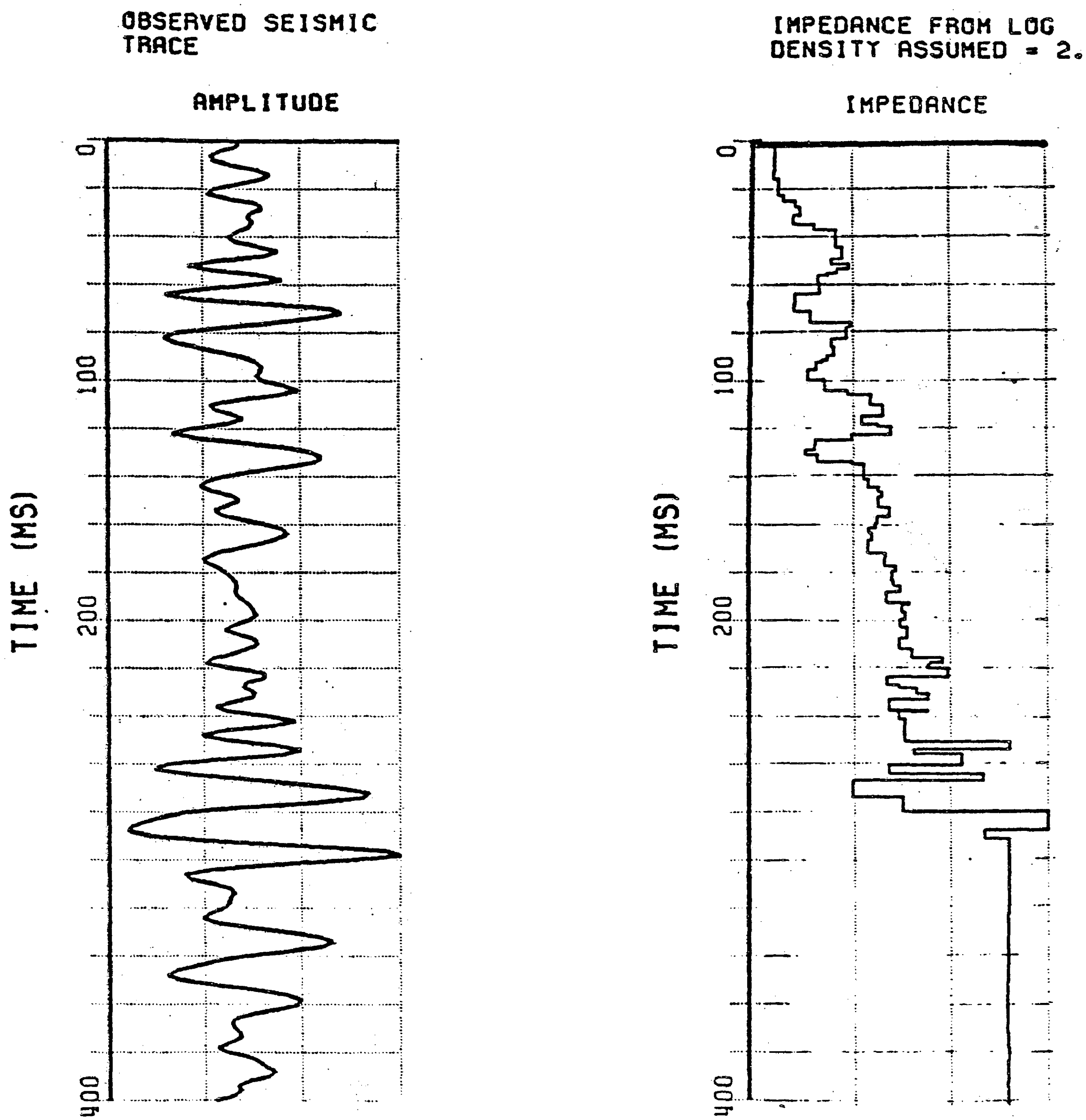


Figure 28 Field data example. Seismic trace from Williston basin and an associated impedance log.

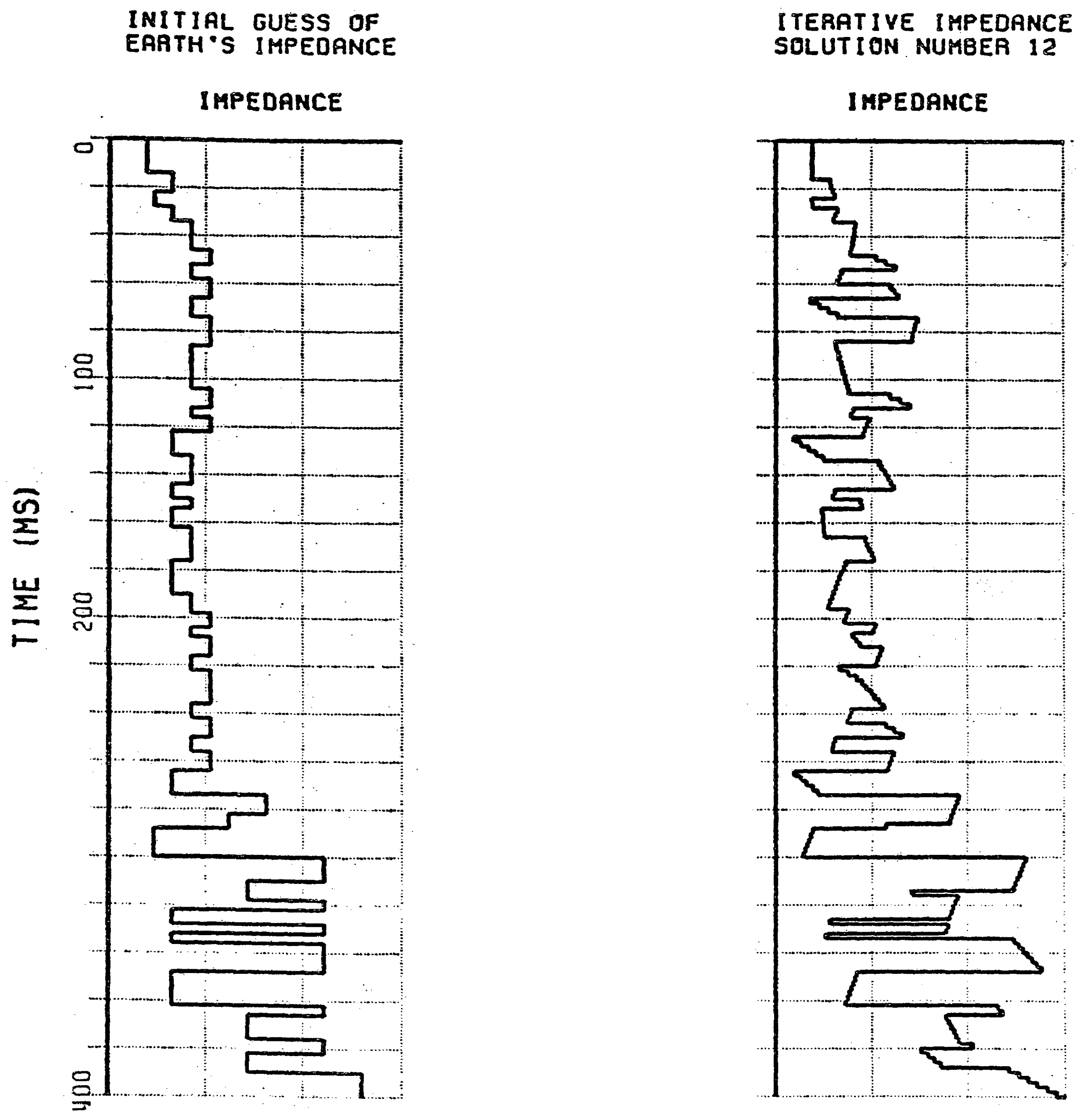


Figure 29 Initial guess used to invert the seismic trace of Figure 27 and the inversion results.

log and is fixed. The guess for all impedance gradients was zero. This example took 760 seconds and 12 iterations on a CDC Cyber 720 computer. To evaluate the results one can compare the impedance solution in Figure 29 to the real log in Figure 28. Another way to evaluate the results is shown in Figure 30 which shows a synthetic seismic trace generated from the impedance solution and the error between this synthetic seismic and the actual observed trace.

In Figure 31 the data are inverted again, this time using a different initial guess. This guess has the same boundary locations as the guess in Figure 29 but the impedance for each boundary is from the solution from Figure 29. Figure 32 evaluates the results of this inversion in the same manner that Figure 30.

The impedance solutions of Figures 29 and 31 show very good agreement with the actual log. Most of the boundary locations match exactly and the impedances have a good correspondence. One should remember that to generate the real impedance log all densities were assumed to be 2.0 gm/cm3 which introduces some error. The synthetic traces generated from the above impedance solutions also show good agreement with the observed trace.

It is a bit disturbing to see that the solutions in Figures 29 and 31 differ. This means that a solution is dependent on its initial guess. The reason for this difference stems from the fact that the boundary solutions are completed

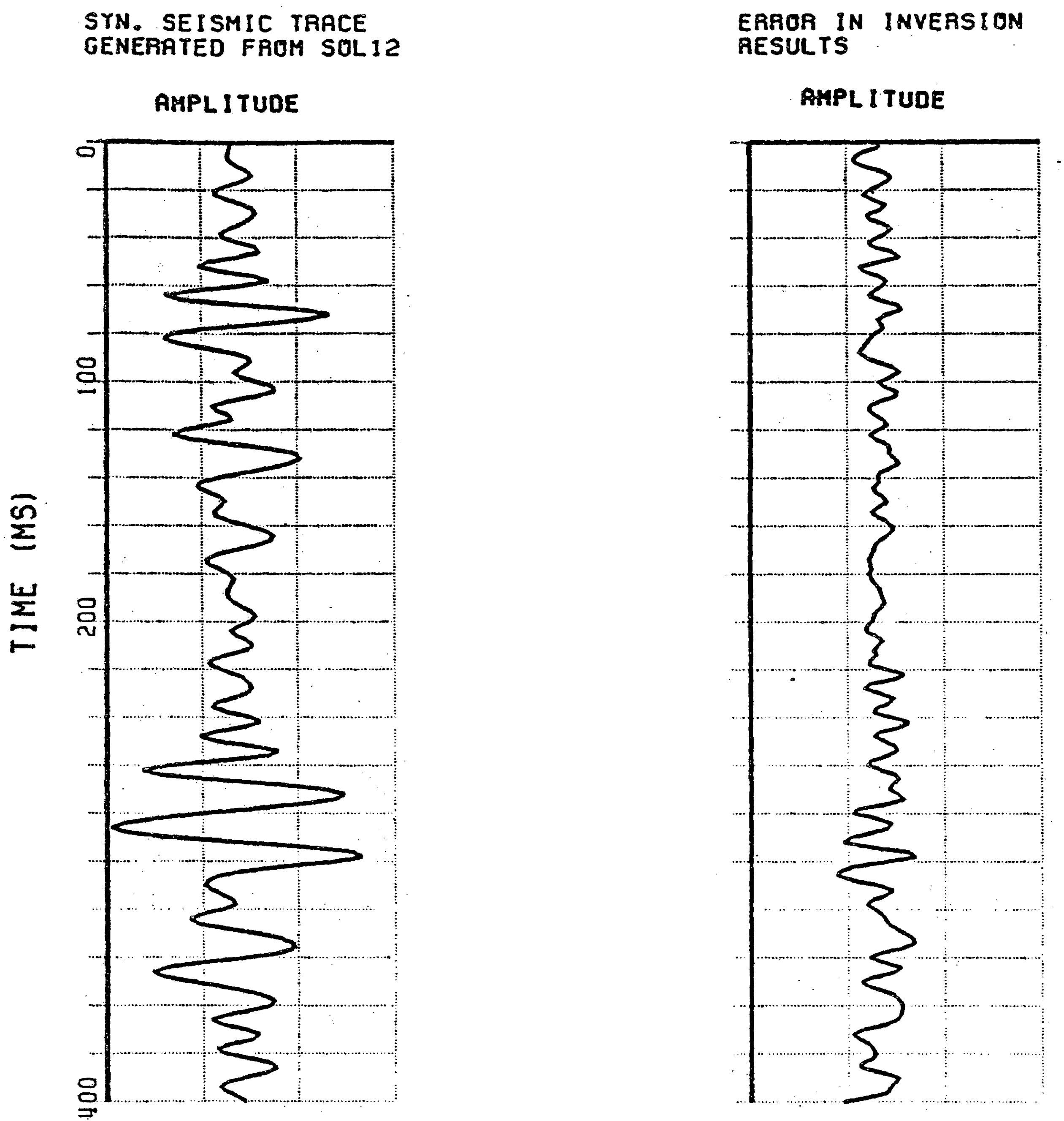


Figure 30 Synthetic seismic trace generated from the impedance solution in Figure 28 and the difference between this synthetic and the data being inverted.

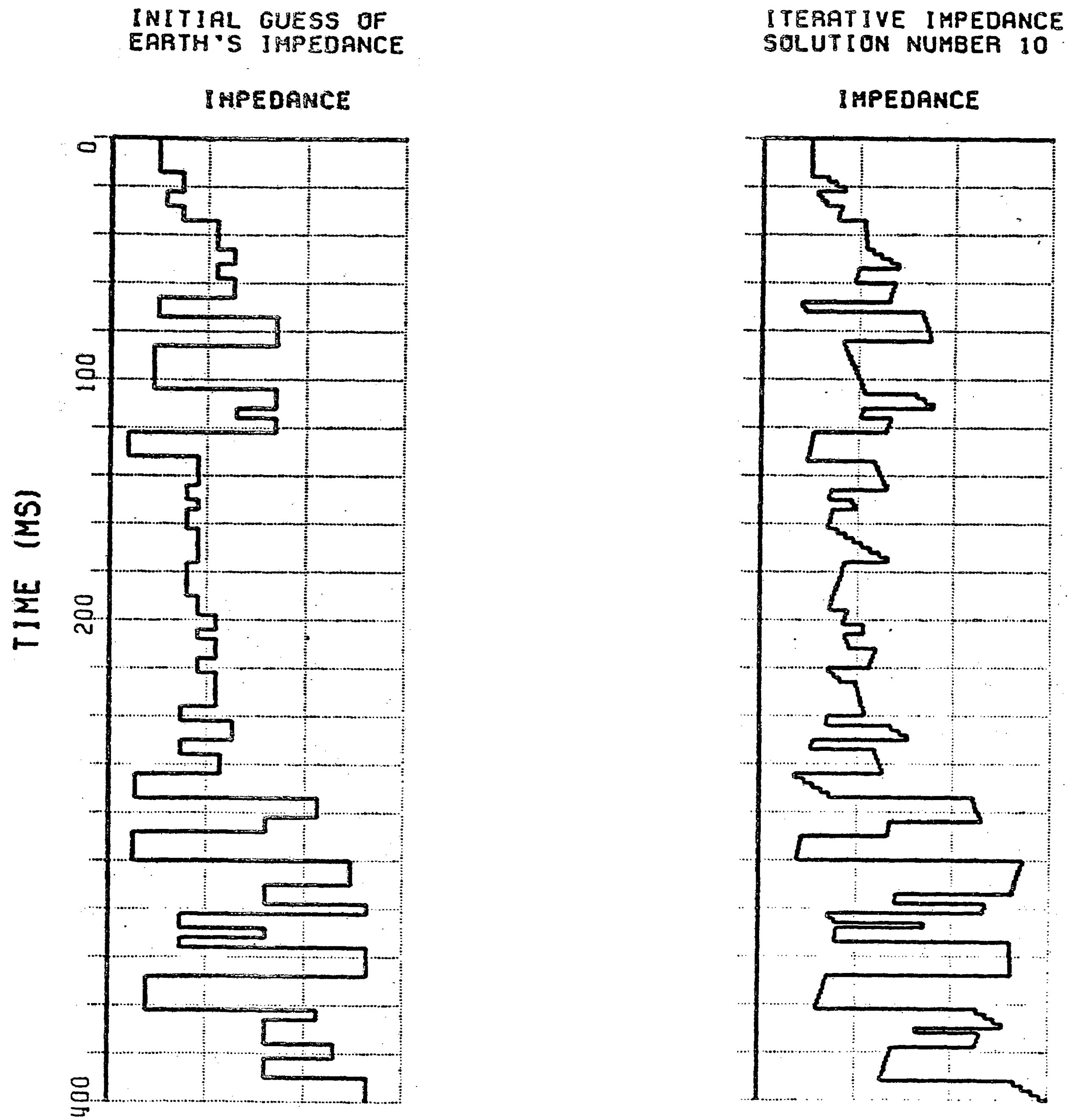


Figure 31 Initial guess used to invert the seismic trace of Figure 27 and the inversion results.

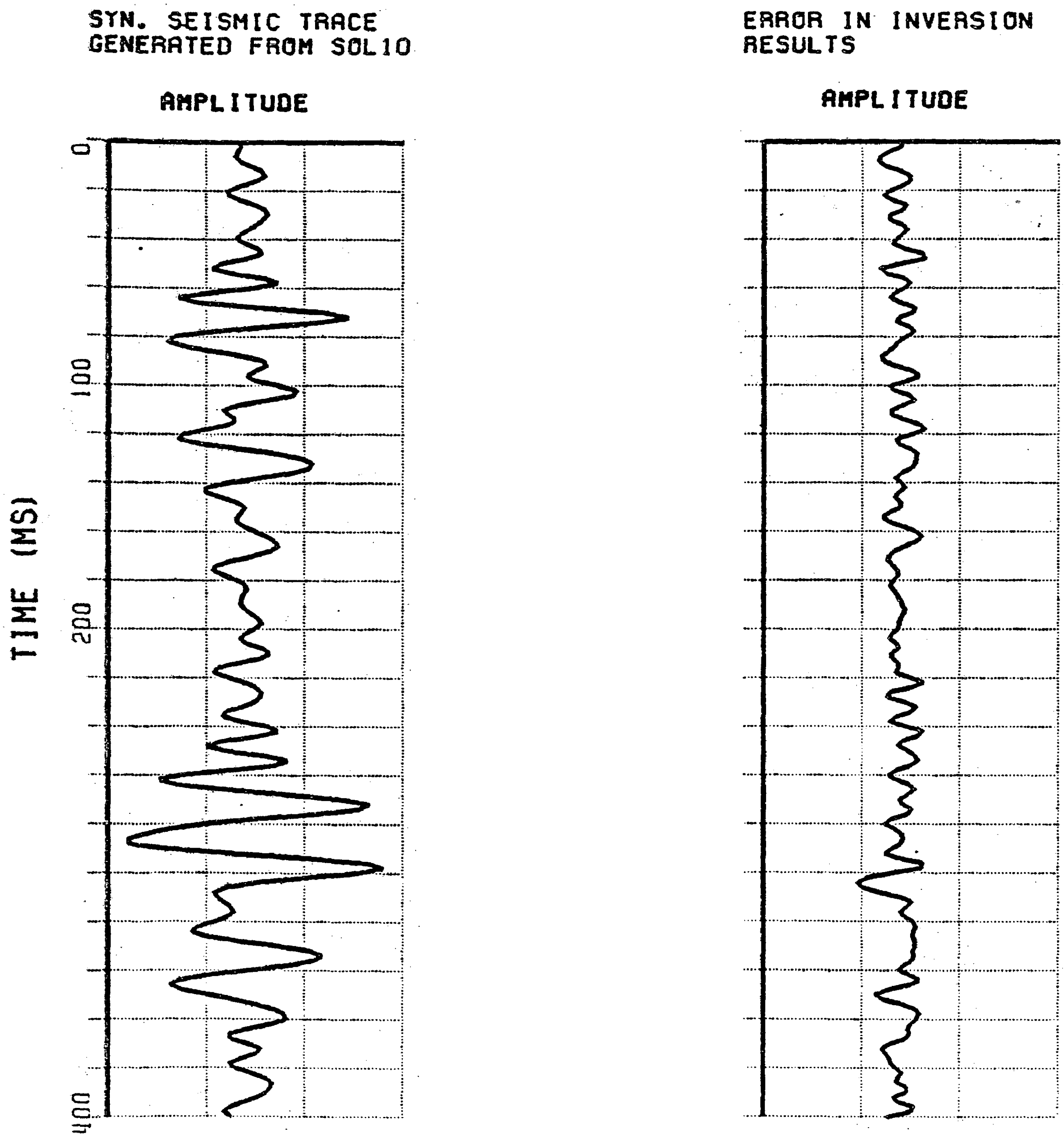


Figure 32 Synthetic seismic trace generated from the impedance solution of Figure 30 and the difference between this synthetic and the data being inverted.

before the impedance solutions are even contemplated* but impedance solutions can have a secondary effect on boundary solutions. This secondary effect is due to wavelet interference between different arrivals (primaries and all multiples) which can move the location of a given peak or trough.

In Figures 33 and 34 the data are inverted again, this time with a different initial guess. The initial guess of this example has a low frequency trend added to it. This trend was taken from the logged impedance values. In this example the amplitude and phase spectra of the source wavelet are solved for. The initial guess for the source wavelet is a 10-20-90-105 band-pass zero phase filter (Figure 35). This is the same source wavelet used in all previous examples - except the source wavelet was not solved for in the previous inversions, it was assumed known and fixed.

The solution for the source wavelet is a 18.1-29.5-95.6-104 band pass filter with the amplitude at 95.6 HZ 20% of the amplitude at 29.5 HZ. The phase of this filter is 6.7 degrees. Note that the amplitudes of the higher frequencies are much less in the solution wavelet than in the source wavelet. This is the sort of effect that is expected from inelastic attenuation. The phase of the solution wavelet is of the type observed by Fausset (1979).

*For an explanation of why this is done see section on parameterization.

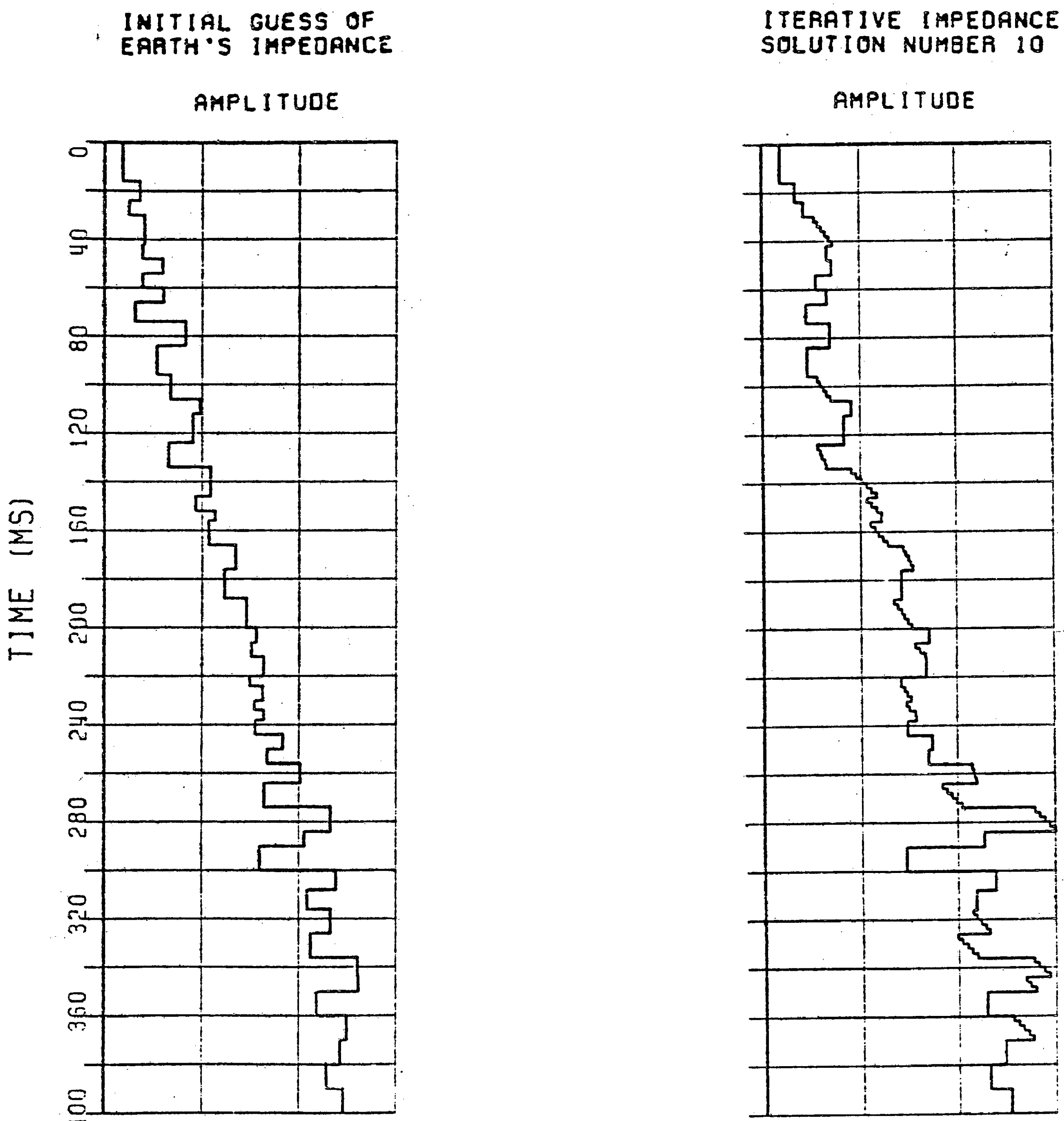


Figure 33 Initial guess and inversion results for seismic trace of Figure 27. The initial guess is the same as in Figure 28. The source wavelet is solved for in this example.

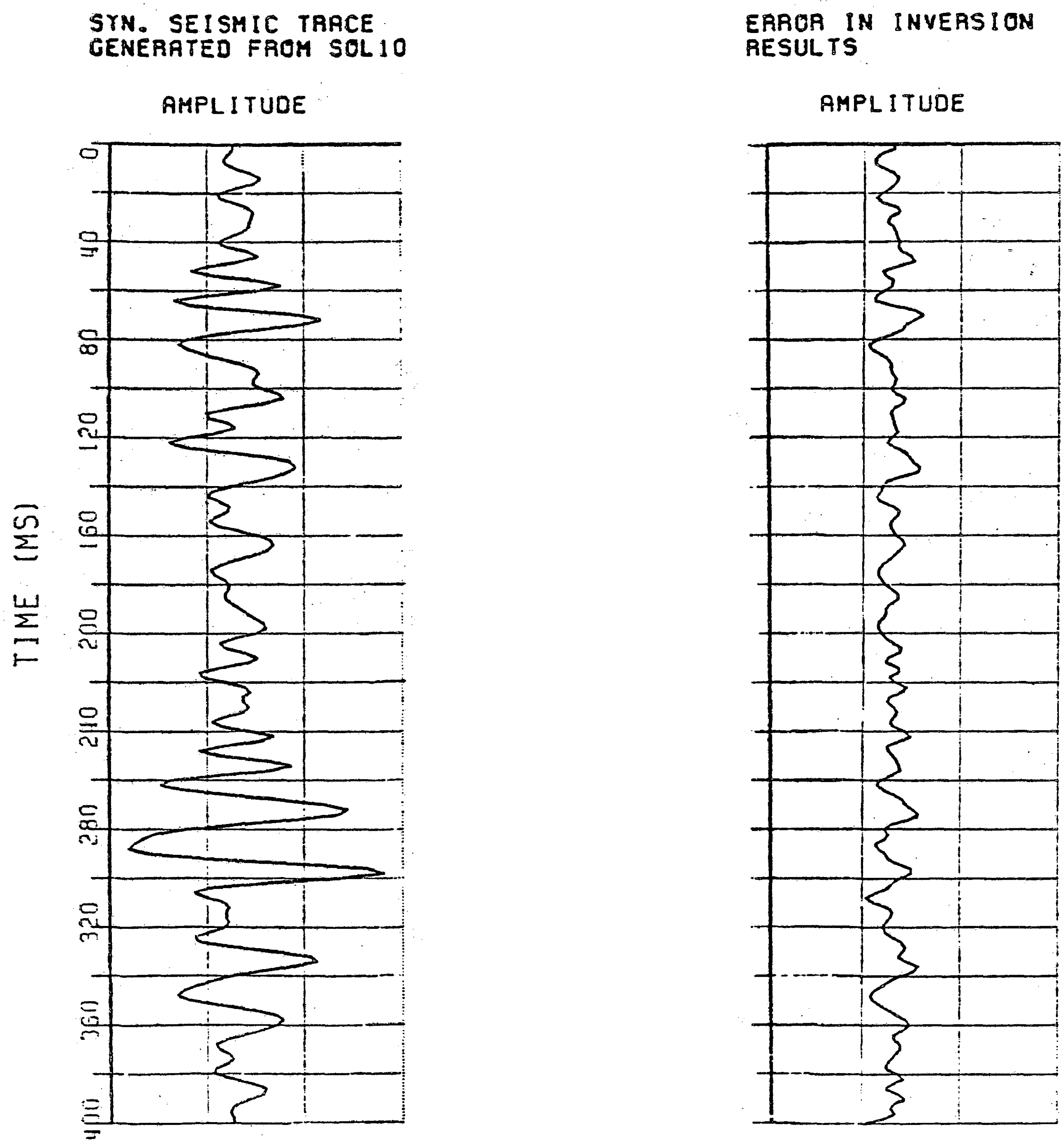


Figure 34 Synthetic seismic trace generated from the impedance solution of Figure 32 and the difference this synthetic and the data being inverted.

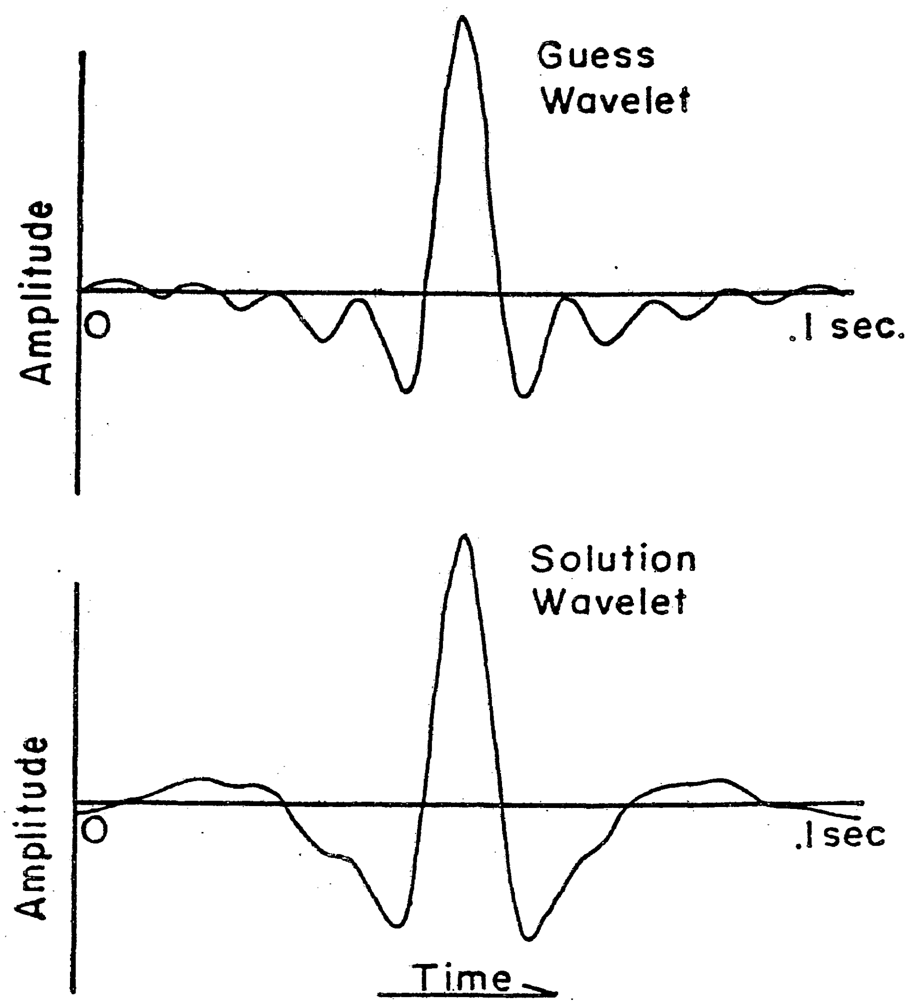
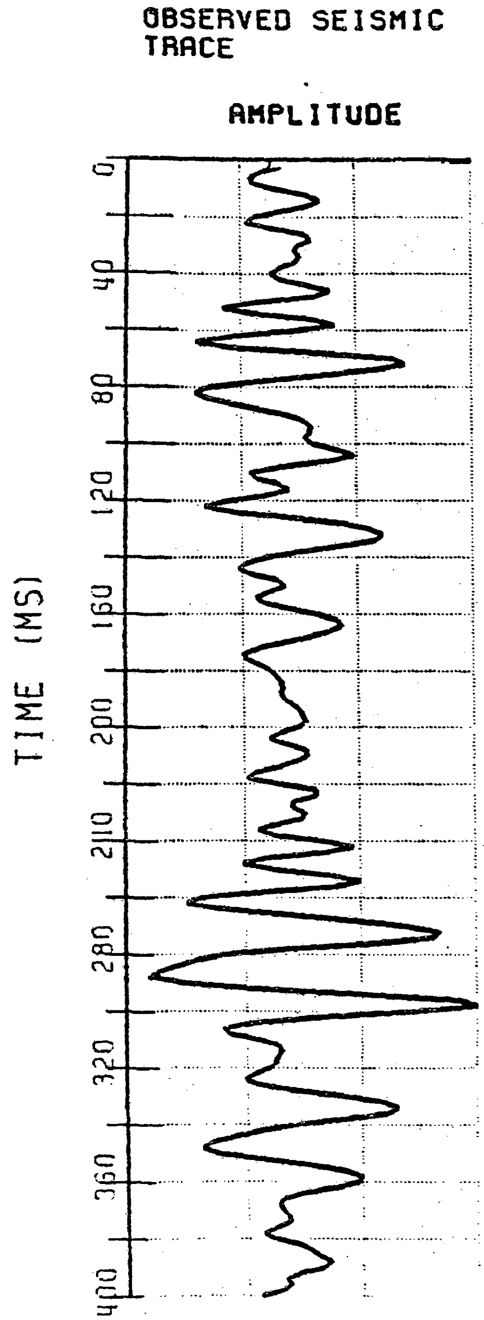


Figure 35 Guess and solution wavelet for the inversion example in Figures 32, 33.

Figures 36 and 37 are used to evaluate the inversion results. In Figure 36 the seismic trace being inverted is compared to a synthetic trace generated from the inversion result of Figure 33. This synthetic seismic trace is the same one displayed in Figure 34. In Figure 37 the inversion impedance result is compared to the logged impedance profile and to a conventional inversion result. This conventional inversion is similar to the Seislog-type inversion and has had a low frequency trend added to it.



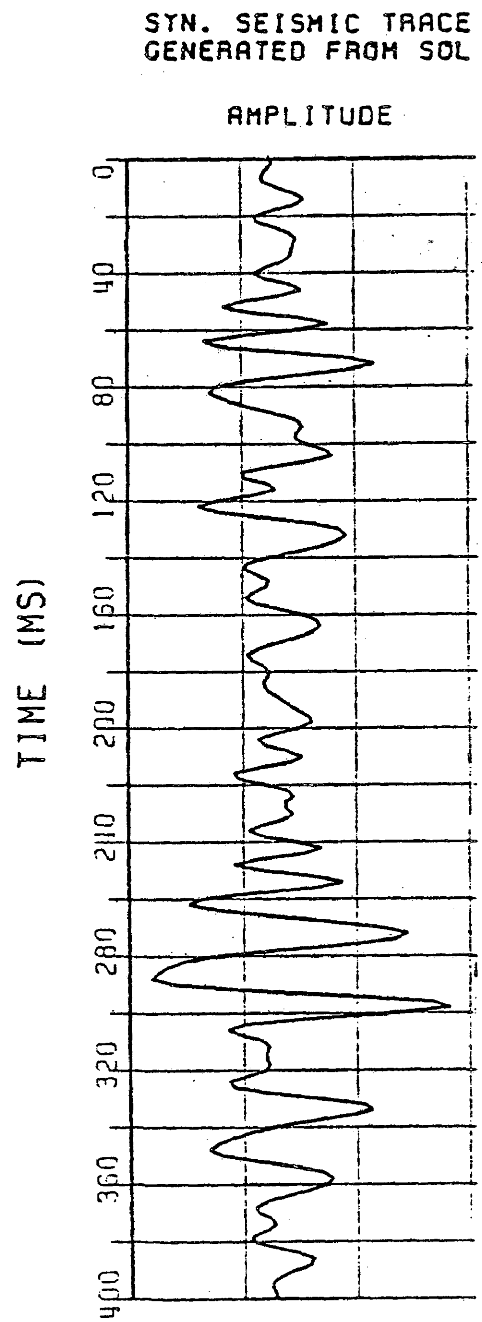


Figure 36 Comparison of the trace being inverted and a synthetic seismic trace generated from the inversion result.

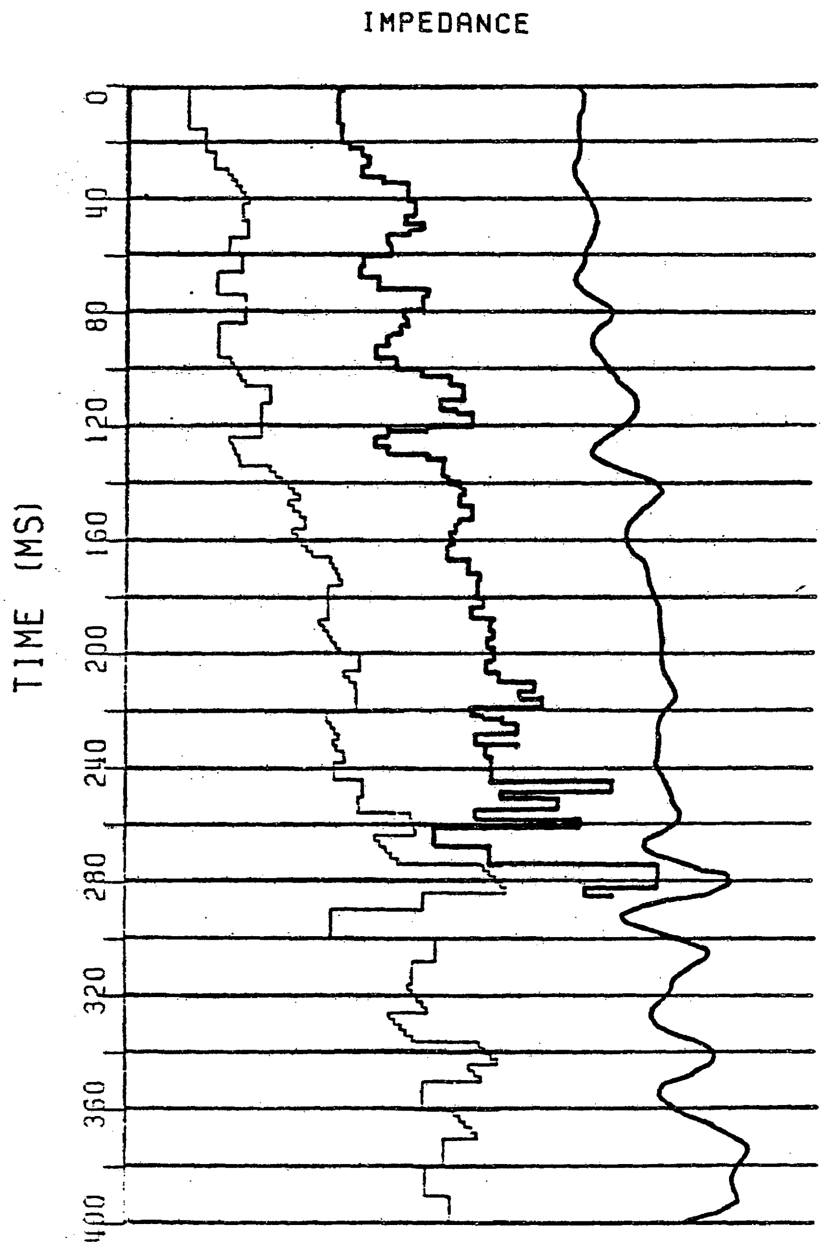


Figure 37. Comparison of (left to right) generalized linear inversion, logged impedance values, and conventional inversion result.

CONCLUSIONS

The well-known and well understood technique of generalized linear inversion has been applied to the problem of one-dimensional seismic inversion. This approach enables one to include in the results some assumptions about the form of those results. These assumptions are brought about by the method of parameterization, which forces all impedance boundaries to be discrete as opposed to continuous. This, in turn, has broadened the frequency content of the results beyond the band width of the input data.

The inversion problem is posed in terms of a set of simultaneous equations, which allows one to fix any of the impedance variables if they are well known. This approach is favorable to recursive inversion where errors tend to propagate and known impedance information cannot be easily utilized.

It has been shown that this technique can discriminate against the inversion of noise (Figure 15) but to do so, it is important that the user have some idea of what is noise and what is signal. In other cases (Figure 25) the solutions for the impedance gradients within one boundary will be dominated by the presence of noise. This effect can be lessened by the use of constraints, but it is questionable whether it is possible to extract reliable inter-boundary impedance gradient information in the presence of noise.

It is shown here how it is possible to solve for the source wavelet. It is also shown (Figure 14) that when the source wavelet is not solved for, the residual wavelet effects are negligible. If one makes a slightly incorrect guess for the source wavelet and does not solve for the wavelet with the impedances, the impedance result will be very similar to the case where the wavelet is solved for.

The generalized linear inversion technique has been tested extensively with synthetic data and the results are satisfactory. It is recommended, though, that more testing be done with field data in areas with good control.

The examples of inversion of real data used a large amount of computer time (760 seconds) and storage space (156,000 octal words). It is recognized that the computation time will drop by a large amount when the initial guess is close to the solution. This would be the case when inverting many traces on the same seismic section. The initial guess for the first trace inverted would be user supplied and the inversion of this trace would take a long time (as in the examples here). Inversion of all other traces would use for initial guess the solution from an adjacent trace for an initial guess. Since the guess would be closer to the solution the number of iterations needed would drop as would computation time.

It is felt that there are a number of software and hardware improvements that could speed up computation time. For example there is some symmetry in the sensitivity matrix and the square of it that can be taken advantage of in terms of computation time and storage space. Also attractive is the use of an array processor for matrix and array manipulation which would greatly speed up the inversion algorithm.

LIST OF REFERENCES CITED

Backus, G. and Gilbert, F., 1968, The Resolving Power of Gross Earth Data: Geophys. J. R. Astr. Soc. v. 16, pp. 169-205.

Backus, G. and Gilbert, F., 1970, Uniqueness in the Inversion of Inaccurate Gross Earth Data: Phil. Trans. Roy. Soc. London, v. 266, pp. 123-192.

Fausset, Neil E., 1979, Statistical Examination of Noise on Phase Estimation in Spiking Deconvolution: M.Sc. Thesis, T-2221, Colorado School of Mines, Golden CO.

Gjoystdal, H. and Ursin, B., 1981, Inversion of Reflection Times in Three Dimensions, Geophysics, v. 46, no. 7, pp. 472-983.

Goupillaud, P. L., 1961, An Approach to Inverse Filtering of Near-Surface Layer Effects from Seismic Records: Geophysics, v. 26, p. 754-760.

Larner, K.; Robinson, J.; Stone, D.; Treitel, S., 1977, Reflectivity as a Model Component, Seg. 3, S.E.G. School, Houston, Texas.

Marquardt, D. W., 1963, An Algorithm for Least Squares Estimation of Nonlinear Parameters: J. Soc. Indust. Appl. Math., v. 11, pp. 431-441.

Parker, R. L., 1977, Understanding Inverse Theory: Ann. Rev. Earth Planet, Sci., v. 5, pp. 35-64.

Robinson, Enders A., 1978, Multichannel Time Series Analysis with Digital Computer Programs: San Francisco, Holden Day.

Vandell, C., 1979, Generalized Linear Inverse Estimation of Acoustic Impedances from Seismic Reflection Data: M. Sc. Thesis, T-2223, Colorado School of Mines, Golden, CO.

Wiggins, R. A., 1972, The General Linear Inverse Problem: Implication of Surface Wave and Free Oscillations for Earth Structure: Rev. Geophysc. Space Phys., v. 10, pp. 251-255.

Wiggins, R. A.; Larner, K. L.; Wisecup, R. D., 1976, Residual Statics Analysis as General Linear Inverse Problem: Geophysics, v. 41, no. 5, pp. 922-938.

Wuenschel, P. C., 1960, Seismogram Synthesis Including Multiples and Transmission Coefficients: Geophysics, v. 25, p. 106-129.

APPENDIX A

Scale Factor and Amplitude Relationship

For any multiple train there are at least two associated primary arrivals if the source and receiver are not on an interface. The amplitudes of any two such primaries and the associated multiples are nonlinear functions of the reflection coefficients. There are two important consequences of this non-linearity that will be discussed in this appendix. They are: 1) If one tries to invert a scaled reflectivity function, exact inversion is impossible if the scale factor is not corrected for, 2) This scale factor can be calculated from the amplitudes of any primaries and associated multiples even when the reflection coefficients are unknown. These points will be illustrated here using a single layer earth model as shown in Figure 4.

From equation 13 we see that for the single layer case the amplitude of the first primary is:

$$A_{po} = C_o$$

and the amplitude of the second primary is:

$$A_{p1} = C_1(1-C_o^2)$$

In these expressions C_n is a reflection coefficient and A_{pn} is the amplitude of the nth primary arrival. Using the above relationships we can calculate what effect a scale factor has on any type of inversion by expressing the reflection coefficients in terms of the arrivals:

$$C_o = A_{po} \qquad 1)$$

$$C_1 = \frac{A_{p1}}{(1-C_o^2)} = \frac{A_{p1}}{(1-A_{po}^2)} \qquad 2)$$

These two expressions are the exact inversion result for the single layer case. If the amplitudes of the primaries are scaled by a factor x, the above expressions become:

$$C_o = XA_{po}$$

$$C_1 = \frac{XA_{p1}}{(1-X^2C_o^2)} = \frac{XA_{p1}}{(1-C_o^2)} \; \frac{1-C_o^2}{1-X^2C_o^2} \qquad 3)$$

Equation 3 shows that the reflection coefficients cannot be recovered from a scaled reflectivity function unless the scale factor is known. This is true for both absolute and relative amplitudes of the reflection coefficients. The first term on the right side of equation 3 is just a scaled version of equation 2. If this was the only term then one could recover the reflection coefficients in a relative sense - that is they would all be scaled by the same unknown amount x. The second term on the right side of equation 26 is the two way transmission loss due to the boundary immediately above. It is this term that makes recovery of the reflection coefficient's relative amplitudes from the reflectivity function impossible. Table 1 is a listing of the distortion due to this term as a function of the scale factor x. In this example C_o is 0.2.

Table 1

Inversion Error Due to Scale Factor

x	$(1-C_o^2)/(1-x^2C_o^2)$
0.5	.97
1	1
1.5	1.05
2	1.14
5	±00
10	∓.032
100	-0.002

The above table shows the inversion error when the reflectivity function is scaled by some unknown amount. It is possible to calculate what this scale factor is if one can identify two primaries and the multiple from them in the reflectivity function. Figure 4 shows the setting for which this is done here - that is the single layer case. From equation 13 we see that the magnitude of the first primary of Figure 4 is:

$$A_{po} = C_o$$

and the second primary arrival is:

$$A_{p_1} = C_1(1-C_o^2)$$

and the first multiple is:

$$A_{M_1} = C_oC_1^2(1-C_o^2).$$

Using the above relationships the magnitude of the multiple can be expressed in terms of the primary amplitudes:

$$A_{M_1} = \frac{A_{po}\ A_{p_1}^2}{(1-A_{po}^2)} \qquad 4)$$

To ascertain if the relfectivity function is scaled one could measure the magnitude of two primaries and the associated multiple. If the data are unscaled then the above equation would hold. If there is an unknown scale factor the above would be an inequality or:

$$A'_{M_1} = \frac{A'_{po}\ A'^2_{p_1}}{(1-A'^2_{p_1})}\ K \qquad 5)$$

or

$$K = A_{M_1}\ \frac{(1-A'_{po})}{A'_{po}\ A'^2_{p_1}}$$

Where the primes indicate scale amplitude values. Using the known values for K, A_{po}, A_{p_1} and A_{M_1} it is possible to solve for the scale factor x. It is important to note at this point that the following solution for the scale factor is not used in generalized linear inversion. Generalized inversion will in effect search for the scale factor required for equation 4 to be true. The following solution for the scale factor is included here only to illustrate that one can recover the scale factor when two primaries and a multiple are known.

If the reflectivity function has been scaled by a constant, we can replace the primed amplitudes in equation 5 with the unscaled values times the constant x:

$$XA_{M_1} = \frac{X\ A_{po}\ \ X^2 A_{p_1}^2\ K}{(1-X^2 A_{po}^2)}$$

or

$$K = \frac{A_{M_1}(1-X^2A_{po}^2)}{X^2A_{po}A_{p_1}^2} = \frac{A'_{M_1}(1-A'^2_{po})}{A'_{po}A'^2_{p_1}} \qquad 6$$

Since K is given by equation 5 the above equation can be solved for X if the unscaled amplitudes (A_{po}, A_{p_1} and A_{M_1}) are known. Since this is not the case we can substitute 4 in 6 to give:

$$K = \frac{A_{po}(A_{p1}^2)}{(1-A_{po}^2)} \qquad \frac{(1-X^2A_{po}^2)}{(X^2A_{po}A_{p_1}^2)}$$

or

$$K = \frac{(1-X^2A_{po}^2)}{X^2-X^2A_{po}^2)} = \frac{(1-A'^2_{po})}{X^2-A'^2_{po})}$$

This can be rewritten to give the scale factor X in terms of K, and A'^2_{po}:

$$X^2 = \frac{1-A'^2_{po}}{K} + A'^2_{po}$$

or if one has not yet measured K, substitute equation 6 in the above to give

$$X^2 = \frac{(1-A'^2_{po})}{} \quad \frac{A'_{po}A'^2_{p_1}}{A'_{M_1}(1-A'^2_{po})} + A'^2_{po}$$

or

$$X = \frac{A'_{po}A'^2_{pi}}{A'_{M_1}} + A'^2_{po} \quad ^{1/2}$$

APPENDIX B

This appendix contains all programs and subroutines, excluding plotting programs, used in this thesis.

```
C
C     PROGRAM INVER INVERTS A SINGLE SEISMIC TRACE USING GENERALIZED
C   LINEAR INVERSION.
C
C     FILE STRUCTURE
C     INPUT FILES:
C     INPUT ---------------------- TERMINAL
C     OUTPUT --------------------- TERMINAL
C     INVTES --------------------- TRACE TO BE INVERTED
C     CONV2 ---------------------- GUESS OF SOURCE WAVELET
C     INITIAL -------------------- INITIAL OF SOURCE WAVELET
C     MENUA ---------------------- FILE CONTAINING INVERSION PARAMETERS
C     NOPER ---------------------- NUMERICAL DERIVATIVE OPERATOR ( THE DOUBLET)
C
C
C     OUTPUT FILES:
C     LPOUT                        FILE CONTAINING INPUT TRACE ,IMPEDANCE GUESS,
C                                  SYNTHETIC GENERATED FROM GUESS,ITERATIVE
C                                  IMPEDANCE SOLUTION, SYNTHETIC GENERATED
C                                  FROM SOLUTION, AND DIFFERENCE BETWEEN INPUT
C                                  TRACE AND SYNTHETIC GENERATED FROM SOLUTION.
C
C
C                 FILE "MENUA" CONTAINS EXECUTION PARAMETERS AND IS OF THE
C                 FORM:
C   00012                MAX. NUMBER OF ITERATIONS.
C   NO                   DO YOU WANT A COMPLETE PRINTOUT?
C    000.00              DELTA P FOR BOUNDRY TRIPS.
C   NO                   SOLVE FOR PHASE ERRORS.
C   NO                   SOLVE FOR WAVELET.
C    3753664.00 00000.0 INITAL GUESS OF SCALE FACTORS.
C   NO                   SOLVE FOR CONSTANT SCALE FACTOR.
C   NO                   SOLVE FOR LINEAR SCALE FACTOR.
C   YES                  SOLVE FOR LINEAR IMPEDANCE CHANGES
C
C
C
C
C      LOGICAL VARIABLES:
C      LWAVE                    TRUE IS WAVELET IS TO BE SOLVED FOR
C      OUTSW                    TRUE IF LONG PRINTOUT IS WANTED (RESULTS
C                               OF EVERY ITERATION)
C      EXIT                     TRUE WHEN ITERATIONS FINISHED
C      LANSW                    USED WITH FUNCTION LANSW
C
C      LBOUND                   TRUE WHEN BOUNDARY ITERAIONS ARE NOT FINISHED
C      LPHAZE                   TRUE WHEN PHASE IS SOLVED FOR
C      LSCALE                   TRUE WHEN CONSTANT SCALE FACTOR IS SOLVED
C                               FOR.
C      LSCALR                   TRUE WHEN LINEAR (WITH TIME) SCALE FACTOR
C                               IS SOLVED FOR. NOT SUPPORTED IN CURRENT VERSION..C
C      LRMP                     TRUE IF IMPEDANCE GRADIENTS (WITHIN ONE INTERVAL
C                               ARE TO BE SOLVE FOR
C
C
C       ARRAYS:
C       A,PRT                  TWO ARRAYS THAT CONTAIN SENSITIVITY MATRIX
C                              AND IT'S SQUARE
C       OBSDAT                 OBSERVED TRACE
C       SEISIG                 CURRENT SYNTHETIC
C       INGESS                 IMPEDANCE PROFILE CREATED FROM DTOB,FIMPV
C       DTOB,FIMPV             DEPTHS AND IMPEDANCE OF EACH INTERVAL IN GUESS.
                               SECOND HALF OF FIMPV CONTAINS IMPEDANCE
```

```
C                                     GRADIENTS FOR EACH INTERVAL
C        DIFF                         CORRECTION VECTOR (ALSO ERROR VECTOR AT TIMES)
C
C
C        VARIABLES@D
C
C        NIMPV                        NUMBER OF IMPEDANCE VALUES
C        NINTS                        NUMBER OF INTERVALS IN GUESS
C        NPARAM                       NUMBER OF PARAMETERS TO BE SOLVED FOR.
C        F1,F2,F3,F4                  BANDPASS OF WAVELET
C        PHZ                          PHZ OF WAVELET.
C                                      LINEAR SCALE FACTOR (LINEARLY INCREASING
C                                      WITH TIME)
C
C*******************************************************************
      PROGRAM INVER(INPUT,OUTPUT,INVTES,CONV2,LPOUT,TAPE5=
     1INPUT,TAPE6=OUTPUT,TAPE7=INVTES,TAPE9=CONV2,TAPE
     210=LPOUT,INITAL,TAPE11=INITAL,MENUA,TAPE14=MENUA
     3,NOPER,TAPE12=NOPER,SEISLG,TAPE15=SEISLG)
      LOGICAL LWAVE,OUTSW,EXIT,LANSW,LNTD,LBOUND,LPHAZE,LSCALE,
     1LDF1,LSCALR,LRMP,LNODSE
      DIMENSION A(19000),SEISAV(220)
      LEVEL 3,A,SEISAV
      COMMON /ECSBLK/ A,SEISAV
      COMMON /MAIN/ SEISIG(220),INGESS(220),DIFF(220),OBSDAT(220),
     1PRT(19000),NPARP1,NPARAM,LBOUND,NPM,WAPSAV(8),WAPBES(8),DTOBES(99)
      COMMON /NUMTD/ LNTD,FILNTD(30),NFLNTD,NTDEL
      COMMON /CRTBLK/DTOB(99),FIMPV(99),NINTS,NIMPV,RMP1
     1,FIBEST(99),FISAVE(99),DIFSAV(99),DTOSAV(99)
      COMMON /WAVBLK2/ SAM,NWAVE,F1,F2,F3,F4,PHZ,LPHAZE,LWAVE,WAVE(100)
     1,NDEL,AF3,WSCALE,F1SAVE,F4SAVE,LDF1
      COMMON /SCLBLK/ LSCALE,SCALE,SCALR,LSCALR,LRMP
      COMMON /BCBLK/ITER,PCBOLD(99),TRIP(99),BNDTKP
      COMMON /WORK1/ WORK11(220),WORK12(220)
      COMMON /WORK2/ WORK21(220),WORK22(220)
      REAL  INGESS,LIMIT,LIMITD,LASSUM
  100 FORMAT (I5,/,I5)
 101  FORMAT(10X,10F12.2)
  102 FORMAT(I5,7F10.3)
  103 FORMAT(5F10.4)
  104 FORMAT(I5)
  105 FORMAT (T20,*ERROR IN INVERSION  RESULTS*)
  106 FORMAT (5F12.4)
 107  FORMAT(T20,*OBSERVED SEISMIC    TRACE*)
  109 FORMAT(*RMS ERROR= *,PE11.2,* DELTA RMS= *,PE11.2)
  110 FORMAT(*ITERATION         TIME PRT    TIME INVER*)
  111 FORMAT(I2,5X,2F12.2)
  112 FORMAT(////,1X,*RMS ERROR=*,PE11.2)
  113 FORMAT(*DO YOU WANT A COMPLETE PRINTOUT*)
  114 FORMAT(T20,*ITERATIVE IMPEDANCE SOLUTION NUMBER *,
     * I2,/,(5F12.4))
  115 FORMAT(T20,*SYN. SEISMIC TRACE  GENERATED FROM SOL*,I2,/,(5F12.4))
  116 FORMAT(*WHAT IS THE MAXIMUM NUMBER OF ITERATIONS ALLOWED*)
  120 FORMAT(5F12.4)
  121 FORMAT(T20,*INITIAL GUESS OF     EARTH'S IMPEDANCE *,/,(5F12.4))
  122 FORMAT(*CONVERGED BEFORE ERROR LIMIT REACHED*)
  123 FORMAT(*ENTER MIN. DELTA P FOR BOUNDRY CHANGES. 0=DEFAULT*)
  124 FORMAT(*DO YOU WANT TO SOLVE FOR PHASE ERROR?*)
  125 FORMAT(*DO YOU WISH TO SOLVE FOR CONSTANT SCALE IN OBS. DATA?*)
  126 FORMAT(*ENTER CONSTANT AND LINEAR SCALE FACTOR FOR OBSERVED DATA*)
  127 FORMAT(*DO YOU WANT TO SOLVE FOR THE WAVELET?*)
  128 FORMAT(T20,*SYN. SEISMIC TRACE  GENERATED FROM GUESS*,/,(5F12.4))
 129  FORMAT(F7.2)
```

```
 130   FORMAT(L3)
 131   FORMAT(A2,/,A2)
 135   FORMAT(F7.2,I5)
 138   FORMAT(T20,*SEISLOG INVERSION*)
 140   FORMAT(F11.2,F8.1)
 141   FORMAT(*DO YOU WISH TO SOLVE FOR LINEAR SCALE IN OBS. DATA?*)
 142   FORMAT(*DO YOU WANT TO SOLVE FOR LINEAR IMPEDANCE CHANGES?*)
 143   FORMAT(*INITAL GUESS OF SCALE FACTORS ARE *,F11.2,F11.2)
C
C*********************SET ERROR LIMITS TO STOP ITERATING
C
       LIMIT = 0.0001
       LIMITD = 0.00001
C
C********************** CHECK FOR MAXIMUM NUMBER OF
C**********************ITERATIONS.  CHECK IF COMPLETE PRINTED IS WANTED.
C
       WRITE(6,116)
       READ(14,104)MAXITER
       WRITE(6,104)MAXITER
       WRITE(6,113)
       OUTSW=LANSW(DUMMY)
       WRITE(6,130)OUTSW
       LWAVE=.TRUE.
       LNTD=.TRUE.
       WRITE(6,123)
       READ(14,129)BNDTRP
       WRITE(6,129)BNDTRP
       IF(BNDTRP.EQ.0.0)BNDTRP=.2
       DO 233 I=1,20
 233   TRIP(I)=BNDTRP
C
C*************READ IN OPERATOR TO BE USED IN ANALYTICAL DERIVATIVE
       READ(12,*)NFLNTD,NTDEL
       READ(12,103)(FILNTD(I),I=1,NFLNTD)
C
C
       WRITE(6,124)
       LPHAZE=LANSW(DUMMY)
       WRITE(6,130)LPHAZE
       WRITE(6,127)
       LDF1=LANSW(DUMMY)
       WRITE(6,130)LDF1
C      LIMIT=LIMIT*0.01
C      LIMITD=LIMITD*0.001
C
       READ(7,100)NPM,NIMPV
       READ(7,101)(OBSDAT(I),I=1,NIMPV)
C      WRITE(6,106)(OBSDAT(I),I=1,NIMPV)
C
C*******READ IN WAVELET USED WITH INITIAL GUESS
C
       READ(9,102)NWAVE,SAM,F1,F2,F3,F4,PHZ,AF3
       READ (9,103) (WAVE(I), I=1,NWAVE)
       NDEL=((NWAVE+1)/2)-1
       IF(PHZ.EQ.0.0.AND.LPHAZE)PHZ=0.1
       F4SAVE=F4
       F1SAVE=F1
C
C*******READ INITIAL GUESS OF SCALE AND SCALR
       READ(14,140)SCALE,SCALR
       WRITE(6,143)SCALE,SCALR
       WRITE(6,125)
```

```
      LSCALE=LANSW(DUMMY)
      WRITE(6,130)LSCALE
      WRITE(6,141)
      LSCALR=LANSW(DUMMY)
      WRITE(6,130)LSCALR
      WRITE(6,142)
      LRMP=LANSW(DUMMY)
      WRITE(6,130)LRMP
C*********WRITE INPUT DATA AND SYN TRACE TO OUTPUT FILE
      WRITE(10,104)NIMPV
      WRITE(10,107)
      WRITE(10,106)(OBSDAT(I),I=1,NIMPV)
C*******READ IN INTITIAL GUESS*****
      READ(11,*)NINTS,NIMPV,RMP1
      READ(11,120)(DTUB(I),I=1,NINTS)
      IEND=(2*NINTS)-1
      READ(11,120)(FIMPV(I),I=1,NINTS)
      ISTART=NINTS+1
      IEND=ISTART+NINTS-1
      READ(11,120)(FIMPV(I),I=ISTART,IEND)
C***********CHECK FOR ZEROS IN THE SECOND PART OF FIMPV
      DO 193 I=ISTART,IEND
 193  IF(FIMPV(I).EQ.0.0)FIMPV(I)=.1
C
C*********GENERATE SEISLOG GUESS
      WORK11(1)=FIMPV(1)
      DO 315 I=1,NIMPV
  315 WORK11(I+1)=WORK11(I)*(1.0+OBSDAT(I))/(1.0-OBSDAT(I))
      WRITE(15,138)
      WRITE(15,106)(WORK11(I),I=1,NIMPV)
C***********INITIALIZE PARAMETERS FOR FIRST PASS THROUGH ITERATION LOOP
C
      ITER=0
      LBOUND=.TRUE.
      LNOOSE=.FALSE.
      EXIT=.FALSE.
C*******GENERATE SYNTHETIC AND RMS FROM INITAL GUESS
C**CRTRMP GENERATES CONTINIOUS IMPED PROFILE FROM THICKNESS AND IMPEDANCES
      CALL CRTRMP(INGESS)
C****************GENERATE SYNTHETIC FROM INITIAL GUESS
C****************AND CALCULATE RMS ERROR
C
      CALL FWRD(INGESS,NIMPV,NPM,SEISIG,.TRUE.)
      CALL ERMS (OBSDAT,SEISIG,DIFF,NIMPV,EBEST)
      LASSUM=EBEST
      WRITE(6,144)EBEST
 144  FORMAT(*ERROR BETWEEN GUESS AND OBS= *,PE11.2)
      ENDFILE 6
C*************WRITE INITIAL GUESS AND SYNTHETIC TO PLOT FILE
      WRITE(13,121)(INGESS(I),I=1,NIMPV)
      WRITE(13,128)(SEISIG(I),I=1,NIMPV)
C***********BEGINING OF INVERSION LOOP
  205 CONTINUE
      ITER=ITER+1
      WRITE(6,899)ITER
 899  FORMAT(*                    ITERATIUN NO. *,I5)
CC
C
C*****************CALCULATE PARTIAL DERIVATIVE MATRIX
C
      TIME1 = SECOND(DUM)
      IF(.NOT.LBOUND)GO TO 211
      NPARAM=NINTS-1
```

```
      NPARP1=NPARAM
      CALL PRTBND (PRT,NPARAM,SEISIG)
      GO TO 212
  211 NPARAM=(2*NINTS)-2
      IF(.NOT.LRMP)NPARAM=NINTS-1
      NPARP1=NPARAM
      IF(LPHAZE)NPARP1=NPARP1+1
      IF(LSCALE)NPARP1=NPARP1+1
      IF(LSCALR)NPARP1=NPARP1+1
      IF(LDF1)NPARP1=NPARP1+5
      CALL PRTIMP (PRT,SEISIG,NPM,NPARAM)
  212 CONTINUE
C    COMPUTE TIME
      TIME2 = SECOND(DUM)
      TIME = TIME2 - TIME1
C
      CALL ERMS (OBSDAT,SEISIG,DIFF,NIMPV,EDUM)
C*****CALCULATE PRT TIMES DIFF
      DO 550 I=1,NPARP1
      SUM=0.0
      IJ=(I-1)*NIMPV+1
      DO 551 J=1,NIMPV
      SUM=SUM+(PRT(IJ)*DIFF(J))
 551  IJ=IJ+1
 550  WORK21(I)=SUM
C********CALCULATE A TRANS A    (A=PRT NOW)
      CALL TIMU(PRT,NIMPV,NPARP1,TRACE)
C*******WRITE A T A
C     DO 570 I=1,NPARP1
C     WRITE(6,575)I
C575  FORMAT(*ROW NUMBER *,I5)
C     ISTART=(I-1)*NPARP1+1
C     IEND=ISTART+NPARP1-1
C570  WRITE(6,571)(PRT(II),II=ISTART,IEND)
C571  FORMAT(10E11.2)
C******AUGMENT ATA WITH ATD
      ISTART=NPARP1*NPARP1
      DO 552 I=1,NPARP1
      IJ=I+ISTART
 552  PRT(IJ)=WORK21(I)
      WRITE(6,546)TRACE
 546  FORMAT(*TRACE OF ATA IS *,PE11.2)
      IF(.NOT.LBOUND)TRACE=TRACE*0.0001
      START=TRACE*10
      FINISH=START/10000.0
      RLAMDA=.000000000001
      TIME1 = SECOND(DUM)
      CALL SEARCH(FINISH,START,EBEST,CKBEST)
      WRITE(6,7890)EBEST
 7890 FORMAT(*ERROR RETURNED = *,PE11.2)
      IF(EBEST.LT.LASSUM) GO TO 979
      IF(.NOT.LBOUND) GO TO 975
      LBOUND=.FALSE.
      WRITE(6,8002)
 8002 FORMAT(*CANNOT IMPROVE ON LAST BOUNDRY ITERATION*)
      LNOOSE=.TRUE.
      GO TO 322
 975  CONTINUE
      WRITE(6,8001)
 8001 FORMAT(*CANNOT IMPROVE ON LAST IMPEDANCE ITERATION*)
      EXIT=.TRUE.
      GO TO 322
 979  CONTINUE
```

```
C********COPY BACK BEST PARAMETERS FOUND IN SEARCH
      IEND=2*NINTS
      DO 321 I=1,IEND
 321  FIMPV(I)=FIBEST(I)
      DO 319 I=1,NINTS
 319  DTOB(I)=DTOBES(I)
      IF(.NOT.LDF1) GO TO 629
      F1=WAPBES(1)
      F2=WAPBES(2)
      F3=WAPBES(3)
      F4=WAPBES(4)
      AF3=WAPBES(5)
  629 IF(LPHAZE)PHZ=WAPBES(6)
      IF(LSCALE)SCALE=WAPBES(7)
      IF(LSCALR)SCALR=WAPBES(8)
 322  CONTINUE
      TIME2 = SECOND(DUM)
      TIME1= TIME2 - TIME1
      DSUM=ABS(LASSUM-EBEST)
      LASSUM=EBEST
  215 WRITE (6,180)
  180 FORMAT (*                    TIME PRT    TIME INVER*)
      WRITE (6,190)TIME,TIME1
  190 FORMAT (7X,2F12.2)
C*******************WRITE RESULTS OF CORRECTIONS
      WRITE(6,715)
 715  FORMAT(*CORRECTED PARAMETERS ARE *)
      IEND=2*NINTS
      WRITE(6,701)(FIMPV(I),I=1,IEND)
 701  FORMAT(5F9.2)
C     IF(LBOUND)WRITE(6,701)(DTOB(I),I=1,NPARAM)
      WRITE(6,701)(DTOB(I),I=1,NPARAM)
      IF(LPHAZE.OR.LDF1.AND..NOT.LBOUND)WRITE(6,747)F1,F2,F3,F4,AF3,PHZ
 747  FORMAT(*F1,F2,F3,F4,AF3,PHZ= *,6F8.2)
      IF(LSCALE) WRITE(6,702)SCALE
      IF(LSCALR) WRITE(6,702)SCALR
  702 FORMAT(* SCALE FACTOR= *,F11.2)
      ENDFILE 6
C*******************CHECK TO BRANCH OUT OF LOOP
      WRITE(6,109)EBEST,DSUM
      WRITE(6,798)CKBEST
 798  FORMAT(*BEST LAMDA= *,PE11.2)
C
      IF(EBEST.LE.LIMIT.OR.ITER.EQ.MAXITER) EXIT=.TRUE.
      IF(DSUM.GT.LIMITD.OR.LBOUND)GO TO 209
      IF(LNODSE) GO TO 209
      WRITE(6,122)
      EXIT=.TRUE.
  209 CONTINUE
C
C*******************WRITE RESULTS (INTERMEDIATE AND/OR FINAL)
C
      IF (OUTSW.OR.EXIT) 207,208
  207 WRITE(10,114)ITER,(INGESS(I),I=1,NIMPV)
      WRITE(10,115)ITER,(SEISIG(I),I=1,NIMPV)
      IF(EXIT) GO TO 206
 208  GO TO 205
 206  CALL ERMS(OBSDAT,SEISIG,DIFF,NIMPV,SUM)
      WRITE(10,105)
      WRITE(10,106)(DIFF(I),I=1,NIMPV)
      WRITE (10,112) SUM
      STOP
      END
```

```
CCCCCCCCCCCCSUBROUTINE EERMS(EBEST,ERROR,RLAMDA)
C PURPOSE   SUBROUTINE EERMS EVALUATES ERRORS (RMS) FOR A
C           GIVEN DAMPING FACTOR. IF THE ERROR IS LOWER
C           THAN THE CURRENT BEST ERROR, EERMS WILL SAVE
C           THE PARAMETER CORRECTIONS MADE AND THE NEW
C           ERROR VECTOR.
C
C     SAMPLE CALL STATEMENT CALL EERMS(EBEST,ERROR,RLAMDA)
C    INPUT3D  RLAMDA      DAMPING FACTOR
C    OUTPUT3D EBEST       BEST (LOWEST) ERROR
C             ERROR      ERROR CALCULATED WITH CURRENT EXECUTION
C                        OF EERMS. IF THIS IS THE LOWEST IT IS ALSO
C                        SAVED IN EBEST.
C*******************************************************************
      SUBROUTINE EERMS(EBEST,ERROR,RLAMDA)
      LOGICAL LDF1,LBOUND,LPHAZE,LSCALE,LSCALR
      DIMENSION A(19000),SEISAV(220)
      LEVEL 3,A,SEISAV
      COMMON /ECSBLK/ A,SEISAV
      COMMON /MAIN/ SEISIG(220),INGESS(220),DIFF(220),SEISMD(220),
     1PRT(19000),NPARP1,NPARAM,LBOUND,NPM,WAPSAV(8),WAPBES(8),DTOBES(99)
      COMMON /CRTBLK/DTOB(99),FIMPV(99),NINTS,NIMPV,RMP1
     1,FIBEST(99),FISAVE(99),DIFSAV(99),DTCSAV(99)
      COMMON /WAVBLK2/ SAM,NWAVE,F1,F2,F3,F4,PHZ,LPHAZE,LWAVE,WAVE(100)
     1,NDEL,AF3,WSCALE,F1SAVE,F4SAVE,LDF1
      COMMON /SCLBLK/ LSCALE,SCALE,SCALR,LSCALR
      COMMON /WORK1/ WORK11(220),WORK12(220)
      COMMON /WORK2/ WORK21(220),WORK22(220)
C     DO 962 I=1,IEND
C962  FIBEST(I)=FIMPV(I)
C     IF(.NOT.LDF1) GO TO 929
C     WAPBES(1)=F1
C     WAPBES(2)=F2
C     WAPBES(3)=F3
C     WAPBES(4)=F4
C     WAPBES(5)=AF3
C 929 IF(LPHAZE)WAPBES(6)=PHZ
C     IF(LSCALE)WAPBES(7)=SCALE
C     IF(LSCALR)WAPBES(8)=SCALR
C     WRITE(6,3000)RLAMDA
C3000 FORMAT(*HAVE ENTERD EERMS. RLAMDA= *,1E11.2)
C********WRITE A TRANS A TO ECS BEFORE ADDING DAMPING FACTOR
      ILENTH=NPARP1*(1+NPARP1)
      CALL WRITEC(PRT(1),A(1),ILENTH)
C********ADD DAMPING FACTOR TO A TRANSPOSE A
      IEND=NPARP1*(NPARP1+1)
C     WRITE(6,4005)(PRT(I),I=1,IEND)
 4005 FORMAT(*PRT=*,10F8.4)
      DO 304 I=1,NPARP1
      II=I+(NPARP1*(I-1))
 304  PRT(II)=PRT(II)+RLAMDA
C************** SOLVE MATRIX EQUATION**********
      CALL SOLVE(PRT,NPARP1,1)
      DO 308 I=1,NPARP1
      II=I+(NPARP1*NPARP1)
 308  DIFF(I)=PRT(II)
C************** LIST PROPOSED CORRECTIONS
C     WRITE(6,700)
C700  FORMAT(*PARAMETER CORRECTIONS ARE*)
C     IEND=NPARAM
C     IF(.NOT.LBOUND)IEND=2*NINTS
C     WRITE(6,701)(DIFF(IP),IP=1,IEND)
C701  FORMAT(5F12.2)
```

```
C*********SAVE PARAMETERS BEFORE CORECTIONS ARE MADE
      IEND=2*NINTS
      DO 651 I=1,IEND
 651  FISAVE(I)=FIMPV(I)
      DO 652 I=1,NINTS
  652 DTOSAV(I)=DTOB(I)
      IF(.NOT.LDF1) GO TO 628
      WAPSAV(1)=F1
      WAPSAV(2)=F2
      WAPSAV(3)=F3
      WAPSAV(4)=F4
      WAPSAV(5)=AF3
  628 IF(LPHAZE)WAPSAV(6)=PHZ
      IF(LSCALE)WAPSAV(7)=SCALE
      IF(LSCALR)WAPSAV(8)=SCALR
C*******APCORR ADDS THE PARAMETER CORRECTIONS
      CALL APCORR(DIFF,NPARAM,LBOUND)
C********EVALUATE THE ERROR CALCULATED WITH THESE NEW CORRECTIONS.
C**CRTRMP GENERATES CONTINIOUS IMPED PROFILE FROM THICKNESS AND IMPEDANCES
      CALL CRTRMP(INGESS)
C*********GENERATE NEW WAVELET IF SOLVING FOR PHAZE
      IF(.NOT.LBOUND.AND.(LDF1.OR.LPHAZE))CALL WAVGEN2(WAVE)
C*****************GENERATE SYNTHETIC FROM UPDATED GUESS
C*****************AND CALCULATE RMS ERROR AND TANGENT WEIGHTED DIFFERENCE
C
      CALL FWRD(INGESS,NIMPV,NPM,WORK22,.FALSE.)
      CALL ERMS (SEISMD,WORK22,WORK21,NIMPV,ERROR)
      WRITE(6,3006)ERROR,EBEST
 3006 FORMAT(*ERROR,EBEST= *,PE11.2,3X,PE11.2)
      IF(ERROR.GE.EBEST)GO TO 610
C******NEXT SECTION IS EXECUTED IF ERROR IS BETTER.
C     WRITE(6,3007)
C3007 FORMAT(*NEW ERROR IS BETTER*)
      EBEST=ERROR
      DO 260 I=1,NINTS
 260  DTOBES(I)=DTOB(I)
      DO 261 I=1,NIMPV
 261  SEISIG(I)=WORK22(I)
C261  DIFFV(I)=WORK21(I)
      DO 262 I=1,IEND
 262  FIBEST(I)=FIMPV(I)
      IF(.NOT.LDF1) GO TO 629
      WAPBES(1)=F1
      WAPBES(2)=F2
      WAPBES(3)=F3
      WAPBES(4)=F4
      WAPBES(5)=AF3
  629 IF(LPHAZE)WAPBES(6)=PHZ
      IF(LSCALE)WAPBES(7)=SCALE
      IF(LSCALR)WAPBES(8)=SCALR
C**********COPY BACK OLDER PARAMETERS
  610 DO 612 I=1,IEND
 612  FIMPV(I)=FISAVE(I)
      DO 611 I=1,NINTS
 611  DTOB(I)=DTOSAV(I)
      IF(.NOT.LDF1) GO TO 630
      F1=WAPSAV(1)
      F2=WAPSAV(2)
      F3=WAPSAV(3)
      F4=WAPSAV(4)
      AF3=WAPSAV(5)
  630 IF(LPHAZE)PHZ=WAPSAV(6)
      IF(LSCALE)SCALE=WAPSAV(7)
```

```
      IF(LSCALR)SCALR=WAPSAV(8)
C********COPY  A TRANS A BACK FROM ECS
      ILENTH=NPARP1*(1+NPARP1)
      CALL READEC(PRT(1),A(1),ILENTH)
C     WRITE(6,3002)ERROR
C3002 FORMAT(*LEAVING EERMS. ERROR IS *,F10.5)
      RETURN
      END
C
C
C
C   SUBROUTINE FWRD CALCULATES THE FORWARD SOLUTION FROM AN INPUT
C   SET OF REFLECTION COEFFICIENTS.  THE FORWARD SOLUTION TAKES INTO
C   ACCOUNT PRIMARIES, MULTIPLES, AND TRANSMISSION LOSSES.
C
C
C  ****************************************************************
C
C   SAMPLE CALL STATEMENT:  CALL FWRD(RFCOEF,NPM,SEIS,MOLNTH,AN,NFILT,FILT
C              ,NDEL,ISR,NF1,NF2,NF3,NF4,PETPHZ)
C
C
C   INPUT:        RFCOEF = ARRAY CONTAINING REFLECTION COEFFICIENTS
C                    NPM = NUMBER OF REFLECTION COEFFICIENTS
C                 MOLNTH = NUMBER OF TIME UNITS TO BE CALCULATED FOR
C                          FORWARD SOLUTION
C                     AN = LOGICAL VARIABLE DETERMINING WHETHER THE FORWARD
C                          IS TO BE CONVOLVED WITH A WAVELET.
C                          IF AN =NEW THEN CALCULATE A NEW WAVELET USING
C                                NF1,NF2,NF3,NF4,PETPHZ AS PARAMERERS.
C                    ISR = SAMPLE RATE OF NEW WAVELET.
C      NF1,NF2,NF3,NF4= TRAPAZOIDAL BAND PASS OF NEW WAVELET.
C                 PETPHZ = PETURBED PHASE OF NEW WAVELET.
C                  NFILT =  NUMBER OF POINT IN THE WAVELET
C                   FILT = THE WAVELET.
C
C
C   OUTPUT:          SEIS = ARRAY CONTAINING FORWARD SOLUTION OR
C                          SEISMIC SYNTHETIC
C
C  ****************************************************************
C
C
C
      SUBROUTINE FWRD(RINPUT,NIMPV,NPM,SEIS,LSAVE)
      LOGICAL LWAVE,LSAVE,LSCALE,LSCALR
C     DIMENSION RFCOEF(220),SEIS(1),BONE(220),RINPUT(1),AONE(220)
      DIMENSION SEIS(1),RINPUT(1)
      DIMENSION A(19000),SEISAV(220)
      LEVEL 3,A,SEISAV
      COMMON /ECSBLK/ A,SEISAV
      COMMON /WAVBLK2/ SAM,NFILT,F1,F2,F3,F4,PHZ,LPHAZE,LWAVE,WAVE(100)
     1,NDEL,AF3,WSCALE,F1SAVE,F4SAVE,LOF1
      COMMON /SCLBLK/ LSCALE,SCALE,SCALR,LSCALR
      COMMON /WORK1/ AONE(220),BONE(220)
      COMMON /WORK2/ RFCOEF(220),WORK22(220)
C     WRITE(6,204)
C204  FORMAT(*HAVE ENTERED FWRD. ABOUT TO WRITE WAVE*)
C     WRITE(6,203)(WAVE(I),I=1,NFILT)
C203  FORMAT(*WAVE IN FWRD= *,5F10.4)
C     WRITE(6,202)
C202  FORMAT(*CALLING IPDRC FROM FWRD*)
      CALL IPDRC(RINPUT,NIMPV,RFCOEF)
```

```
      KEND = NPM - 1
      AONE(1) = 1.0
      BONE(1) = RFCOEF(NPM)
      DO 10   III=1,KEND
      I=KEND-III+1
      II = NPM - I
      DO 20   JJ=1,II
      J=II-JJ+1
   20 BONE(J+1) = BONE(J)
      BONE(1) = RFCOEF(I)*AONE(1)
      IF (II.LT.2)  GO TO 40
      DO 30   K=2,II
      SAVE = AONE(K)
      AONE(K) = AONE(K) + RFCOEF(I)*BONE(K)
      BONE(K) = RFCOEF(I)*SAVE + BONE(K)
   30 CONTINUE
   40 AONE(II+1) = RFCOEF(I) * BONE(II+1)
   10 CONTINUE
C     WRITE(6,201)
C201  FORMAT(*CALLING POLYDIV FROM FWRD*)
      CALL POLYDV (NPM,AONE,NPM,BONE,NIMPV,SEIS)
C**********SAVE RESULT BEFORE FILTERING IF DESIRED
      IF(.NOT.LSAVE) GO TO 47
      CALL WRITEC(SEIS(1),SEISAV(1),NIMPV)
C**********CONVOLVE RESULT WITH WAVELET IF DESIRED
   47 IF(.NOT.LWAVE)GOTO 50
C     WRITE(6,205)(SEIS(I),I=1,NIMPV)
C205  FORMAT(*SEIS = *,5F10.4)
C     WRITE(6,206)(WAVE(I),I=1,NFILT)
C206  FORMAT(*WAVE= *,5F10.4)
C     WRITE(6,207)(AONE(I),I=1,NIMPV)
C207  FORMAT(*AONE= *,5F10.4)
      CALL FOLD(NIMPV,SEIS,NFILT,WAVE,LC,AONE,NDEL)
      DO 45 I=1,NIMPV
   45 SEIS(I)=AONE(I)
C**********SCALE DATA BY CURRENT SCALE FACTOR IF SOLVING FOR SCALE
   50 CONTINUE
C  50 IF(.NOT.LSCALE)GO TO 52
      DO 53 I=1,NIMPV
   53 SEIS(I)=SEIS(I)*(SCALE+(I*SCALR))
   52 CONTINUE
      END
C
C
C
C   SUBROUTINE IPDRC CALCULATES THE REFLECTION COEFFICIENTS FROM
C   THE IMPEDANCE SERIES
C
C        THIS SUBROUTINE IS TAKEN FROM VANDELL(1979)
C ****************************************************************
C
C   SAMPLE CALL STATEMENT:  CALL IPDRC(MIPDLG,NIMPD,RCLG)
C
C
C   INPUT:        MIPDLG = ARRAY CONTAINING IMPEDANCE SERIES
C                  NIMPD = NUMBER OF IMPEDANCE LAYERS
C
C
C   OUTPUT:         RCLG = ARRAY CONTAINING REFLECTION COEFFICIENTS
C                          GENERATED BY IMPEDANCE LAYERS
C
C ****************************************************************
C
```

```
C
C
      SUBROUTINE IPDRC (MIPOLG,NIMPD,RCLG)
      DIMENSION MIPOLG(NIMPD),RCLG(NIMPD)
      REAL MIPOLG
      DO 10  I=2,NIMPD
      J = I-1
   10 RCLG(J) = (MIPOLG(I) - MIPOLG(J)) / (MIPOLG(I) + MIPOLG(J))
      RETURN
      END
C
C
C
C  SUBROUTINE ZERO STORES A FLOATING-POINT NUMBER ZERO
C  IN EACH STORAGE LOCATION OF AN ARRAY
C
C
C  (PROGRAM IS TAKEN FROM MULTICHANNEL TIME SERIES ANALYSIS
C    WITH DIGITAL COMPUTER PROGRAMS, ROBINSON,1967)
C
C ***************************************************
C
C  SAMPLE CALL STATEMENT:   CALL ZERO(LX,X)
C
C
C  INPUT:          LX = LENGTH OF ARRAY X
C                   X = ARRAY TO BE ZEROED
C
C
C  OUTPUT:          X = ZEROED ARRAY
C
C ***************************************************
C
C
C
      SUBROUTINE ZERO (LX,X)
      DIMENSION X(LX)
      IF (LX.LE.0) RETURN
      DO 1  I=1,LX
    1 X(I) = 0.0
      RETURN
      END
C
C
C
C  SUBROUTINE POLYDV DIVIDES ONE POLYNOMIAL BY ANOTHER
C
C
C  (PROGRAM IS TAKEN FROM MULTICHANNEL TIME SERIES ANALYSIS
C    WITH DIGITAL COMPUTER PROGRAMS, ROBINSON,1967)
C
C *************************************************************
C
C  SAMPLE CALL STATEMENT:   CALL POLYDV(N,DVS,M,DVD,L,Q)
C
C
C  INPUT:            N = LENGTH OF DIVISOR POLYNOMIAL
C                  DVS = ARRAY CONTAINING DIVISOR POLYNOMIAL
C                    M = LENGTH OF DIVIDEND POLYNOMIAL
C                  DVD = ARRAY CONTAINING DIVIDEND POLYNOMIAL
C                    L = LENGTH OF DESIRED OUTPUT QUOTIENT
C
```

```
C   OUTPUT          Q = ARRAY CONTAINING THE QUOTIENT OF DVD/DVS
C
C  *****************************************************************
C
C
C
      SUBROUTINE POLYDV(N,DVS,M,DVD,L,Q)
      DIMENSION DVS(N),DVD(M),Q(L)
      CALL ZERO (L,Q)
      CALL MOVE (MINO(M,L),DVD,Q)
      DO 10  I=1,L
      Q(I) = Q(I)/DVS(1)
      IF (I.EQ.L) RETURN
      K = I
      ISUB = MINO(N-1,L-I)
      DO 10  J=1,ISUB
      K = K + 1
   10 Q(K) = Q(K) - Q(I)*DVS(J+1)
      RETURN
      END
C
C
C
C   SUBROUTINE MOVE MOVES AN ARRAY FROM ONE STORAGE LOCATION
C   TO ANOTHER
C
C
C   (PROGRAM IS TAKEN FROM MULTICHANNEL TIME SERIES ANALYSIS
C     WITH DIGITAL COMPUTER PROGRAMS, ROBINSON,1967)
C
C  *****************************************************************
C
C   SAMPLE CALL STATEMENT:   CALL MOVE (LX,X,Y)
C
C
C   INPUT:          LX = LENGTH OF ARRAY X TO BE MOVED TO ARRAY Y
C                    X = ARRAY TO BE MOVED
C
C
C   OUTPUT:          Y = ARRAY THAT CONTAINS THE VALUES (LX IN NUMBER)
C                        ORIGINALLY CONTAINED IN ARRAY X
C
C  *****************************************************************
C
C
C
      SUBROUTINE MOVE (LX,X,Y)
      DIMENSION X(LX),Y(LX)
      DO 1  I=1,LX
    1 Y(I) = X(I)
      RETURN
      END
C
C
C
C   SUBROUTINE FOLD COMPUTES THE PRODUCT OF TWO POLYNOMIALS
C
C
C   (PROGRAM IS TAKEN FROM MULTICHANNEL TIME SERIES ANALYSIS
C     WITH DIGITAL COMPUTER PROGRAMS, ROBINSON,1967)
C
C  *****************************************************************
```

```
C
C   SAMPLE CALL STATEMENT:   CALL FOLD(LA,A,LB,B,LC,C)
C
C
C   INPUT:         LA = LENGTH OF FIRST POLYNOMIAL MULTIPLIER
C                   A = ARRAY CONTAINING THE FIRST POLYNOMIAL MULTIPLIER
C                  LB = LENGTH OF SECOND POLYNOMIAL MULTIPLIER
C                   B = ARRAY CONTAINING THE SECOND POLYNOMIAL MULTIPLIER
C                NDEL = NUMBER OF POINTS THAT OUTPUT IS DELAYED.
C
C   OUTPUT:        LC = LENGTH OF PRODUCT OF A*B (LA+LB-1)
C                   C = ARRAY CONTAINING THE POLYNOMIAL WHICH IS THE PRODUCT
C                       OF A*B
C
C  ****************************************************************
C
C
C
      SUBROUTINE FOLD (LA,A,LB,B,LC,C,NDEL)
      DIMENSION A(LA),B(LB),C(LC),TEMP(400)
C     WRITE(6,501)(A(I),I=1,LA)
C501  FORMAT(*LA= *,5F10.4)
C      WRITE(6,502)(B(I),I=1,LB)
C502  FORMAT(*LB= *,5F10.4)
      LC = LA + LB - 1
      CALL ZERO (LC,C)
      DO 1  I=1,LA
      DO 1  J=1,LB
      K = I + J - 1
    1 C(K) = C(K) + A(I)*B(J)
      LD=LA+NDEL
      DO 2 I=1,LD
 2    TEMP(I)=C(I+NDEL)
      DO 3 I=1,LA
 3    C(I)=TEMP(I)
      LC=LA
      RETURN
      END
C
C
C
C
C
C
C
C   SUBROUTINE ERMS COMPUTES THE DIFFERENCE BETWEEN THE OBSERVED AND
C   CALCULATED SEISMOGRAM AND CALCULATES THE RMS ERROR
C
C
C  ****************************************************************
C
C   SAMPLE CALL STATEMENT:   CALL ERMS(SEISMO,SEISIG,DIFF,MOLNTH,SUM)
C
C
C   INPUT:       SEISMO = ARRAY CONTAINING OBSERVED DATA TRACE
C                SEISIG = ARRAY CONTAINING CALCULATED DATA TRACE
C                MOLNTH = LENGTH IN TIME UNITS OF BOTH OBSERVED AND
C                         CALCULATED DATA TRACE
C
C
C   OUTPUT:        DIFF = ARRAY CONTAINING DIFFERENCE AT EACH TIME UNIT
C                         BETWEEN OBSERVED AND CALCULATED DATA TRACE
C                   SUM = RMS AMPLITUDE OF DIFFERENCE ARRAY
C
```

```
C ****************************************************************
C
C
C
      SUBROUTINE ERMS(SEISMD,SEISIG,DIFF,MDLNTH,SUM)
      DIMENSION SEISMD(1),SEISIG(1),DIFF(1)
      SUM = 0.0
      DO 10  I=1,MDLNTH
      DIFF(I) = SEISMD(I) - SEISIG(I)
   10 SUM = SUM + DIFF(I)**2
      IEND=NINTS*2
      SUM = (SUM/MDLNTH)**.5
      RETURN
      END
C
C
C
C
C*******************************************************
C
C  SUBROUTINE PRTIMP CALCULATES PARTIALS FOR IMPEDANCE VALUES,
C              WAVELET AND SCALE FACTOR.
C SAMPLE CALL STATEMENT    SUBROUTINE PRTIMP(A,SEISIG,NPM,NPARAM)
C    INPUT    SEISIG         CURRENT SEISMIC SYNTHETIC
C             NPARAM         NUMBER OF DERIVATIVE VECTORS TO CALCULATE
C    OUTPUT   A              PARTIAL DERIVATIVE MATRIX
C
C****************************************************************
      SUBROUTINE PRTIMP(A,SEISIG,NPM,NPARAM)
      LOGICAL LWAVE,LPHAZE,LSCALE,LDF1,LSCALR
      COMMON /CRTBLK/ DTOB(99),FIMPV(99),NINTS,NIMPV,RMP1
     1,FIBEST(99),FISAVE(99),DIFSAV(99),DTOSAV(99)
      COMMON /WAVBLK2/ SAM,NWAVE,F1,F2,F3,F4,PHZ,LPHAZE,LWAVE,WAVE(100)
     1,NDEL,AF3,WSCALE,F1SAVE,F4SAVE,LDF1
      COMMON /SCLBLK/ LSCALE,SCALE,SCALR,LSCALR
      DIMENSION A(1),SEISIG(1),IMPED(220),SEISIC(220)
      REAL IMPED
C
C
      IJ=1
      DO 10   I=1,NPARAM
      JJ=I+1
C     WRITE(6,1000)JJ,FIMPV(JJ)
C1000 FORMAT(*DOING PARTIAL FOR IMPED # *,I2,F10.2)
      PHOLD =FIMPV(JJ)
      FIMPV(JJ)=FIMPV(JJ)*1.1
C     IF(JJ.EQ.4)GO TO 638
C     IF(JJ.EQ.5) GO TO 638
C     IF(JJ.EQ.9) GO TO 638
C     IF(JJ.EQ.11) GO TO 638
      CALL CRTRMP(IMPED)
C
C********CALCULATE A SEISMIC SYNTHETIC FROM PERTURBED MODEL PARAMETERS
C
  100 CALL FWRD(IMPED,NIMPV,NPM,SEISIC,.FALSE.)
C
C  INNER LOOPS CALCULATE PARTIALS AT EACH TIME LAYER
C
   39 DO 41 J=1,NIMPV
      A(IJ)=(SEISIC(J)-SEISIG(J))/(FIMPV(JJ)-PHOLD)
   41 IJ=IJ+1
      GO TO 645
  638 CONTINUE
```

```
      DO 639 J=1,NIMPV
      A(IJ)=0.0
 639  IJ=IJ+1
 645  CONTINUE
C     ISTART=IJ-NIMPV+1
C     IEND=IJ
C     WRITE(6,3010)
C3010 FORMAT(*DOING IMPED INVERSION*)
C     WRITE(6,3004)(A(K),K=ISTART,IEND)
C3004 FORMAT(*PARTIALS ARE *,5E11.2)
C     WRITE(6,3000)(IMPED(IZ),IZ=1,NIMPV)
C3000 FORMAT(*PET GUESS *,5F11.3)
C     IEND=NPARAM*2
C     WRITE(6,3001)(FIMPV(IZ),IZ=1,IEND)
C3001 FORMAT(*IMPED VALS *,5F10.4)
C     WRITE(6,3002)(SEISIC(IZ),IZ=1,NIMPV)
C3002 FORMAT(*NEW MODEL *,5F10.4)
C     WRITE(6,3003)(SEISIG(IZ),IZ=1,NIMPV)
C3003 FORMAT(*OLD MODEL *,5F10.4)
      FIMPV(JJ)=PHOLD
   10 CONTINUE
C***************CALCULATE PARTIAL WITH RESPECT TO PHAZE IF WANTED
      IF(.NOT.LPHAZE) GO TO 11
      CALL WAVPAR(PHZ,NWAVE,WAVE,NIMPV,NDEL,IJ,A)
   11 CONTINUE
C********CALCULATE PARTIALS FOR CONSTANT SCALE FACTOR.
      IF(.NOT.LSCALE)GOTO 48
      DO 13 I=1,NIMPV
      A(IJ)=SEISIG(I)/(SCALE+(I*SCALR))
   13 IJ=IJ+1
   48 CONTINUE
C********CALCULATE PARTIALS FOR LINEAR SCALE FACTOR.
      IF(.NOT.LSCALR)GOTO 49
      WRITE(6,4579)SCALR
 4579 FORMAT(*DOING LINEAR SCALE PRT. SCALR= *,F10.4)
      DO 14 I=1,NIMPV
      A(IJ)=SEISIG(I)*I/(SCAL+(I*SCALR))
   14 IJ=IJ+1
   49 CONTINUE
C*********CALCULATE PARTIALS WITH RESPECT TO WAVELET
      IF(.NOT.LDF1) GO TO 50
      CALL WAVPAR(F1,NWAVE,WAVE,NIMPV,NDEL,IJ,A)
C     DO 65 I=1,NIMPV
C     J=ISAVE+I-1
C     WRITE(6,64)A(J),J
C64   FORMAT(*RETURNED FROM WAVPAR *,F12.6,I5)
C65   CONTINUE
      CALL WAVPAR(F2,NWAVE,WAVE,NIMPV,NDEL,IJ,A)
      CALL WAVPAR(F3,NWAVE,WAVE,NIMPV,NDEL,IJ,A)
      CALL WAVPAR(F4,NWAVE,WAVE,NIMPV,NDEL,IJ,A)
      CALL WAVPAR(AF3,NWAVE,WAVE,NIMPV,NDEL,IJ,A)
 50   CONTINUE
      RETURN
      END
C
C
C
C
C***********************************************************************
C
C SUBROUTINE WAVPAR CALCULATES THE DERIVATIVE OF THE SEIMIC TRACE
C            WITH RESPECT TO WAVELET PARAMETERS.
C
```

```
C     INPUT     PARAM     DERIVATIVE WITH RESPECT TO THIS PARAMETER IS
C                         CALCULATED.
C               NFILT     NUMBER OF POINTS IN WAVELET.
C               WAVE      SOURCE WAVELET.
C               NIMPV     NUMBER OF POINTS IN OUTPUT DERIVATIVE.
C               NDEL      DELAY IN WAVELET.(=NFILT/2 FOR ZERO PHASE)
C
C      OUTPUT   RETURN    PARTIAL DERIVATIVE.
C               IJ        CURRENT INDEX OF LAST POINT IN RETURN.
C
C*******************************************************
      SUBROUTINE WAVPAR(PARAM,NFILT,WAVE,NIMPV,NDEL,IJ,RETURN)
      DIMENSION WAVE(1),RETURN(1)
      DIMENSION A(19000),SEISAV(220)
      LEVEL 3,A,SEISAV
      COMMON /ECSBLK/ A,SEISAV
      COMMON /WORK1/ TEMP1(220),TEMP(220)
      COMMON /WORK2/ TEMP2(220),WORK22(220)
      PHOLD=PARAM
      PARAM=PARAM*1.095678
      CALL WAVGEN2(TEMP1)
      DENOM=PARAM-PHOLD
      DO 52 I=1,NFILT
      TEMP2(I)=(TEMP1(I)-WAVE(I))/DENOM
C     WRITE(6,60)I,TEMP2(I),TEMP1(I),WAVE(I)
C60   FORMAT(*I,DER,PET,ORGINAL= *,I5,3F12.5)
 52   CONTINUE
      PARAM=PHOLD
      CALL READEC(TEMP(1),SEISAV(1),NIMPV)
      CALL FOLD(NIMPV,TEMP,NFILT,TEMP2,LC,TEMP1,NDEL)
C     WRITE(6,61)(TEMP(I),I=1,NIMPV)
C61   FORMAT(*SEISAV *,5F8.3)
C     WRITE(6,62)(TEMP1(I),I=1,NIMPV)
C62   FORMAT(*OUTPUT OF CONV *,5F12.6)
      DO 53 I=1,NIMPV
      RETURN(IJ)=TEMP1(I)
 53   IJ=IJ+1
      RETURN
      END
C
C
C***************************************************************
C
C                SUBROUTINE PRTBND CALCULATES PARTIALS FOR BOUNDRIES
C     INPUT      SEISIG            CURRENT SEISMIC SYNTHETIC.
C                NPARAM            NUMBER OF BOUNDARIES.
C     OUTPUT     A                 PAPTIAL DERIVATIVE MATRIX.
C
C***************************************************************
      SUBROUTINE PRTBND(A,NPARAM,SEISIG)
      COMMON /CRTBLK/ DTOB(99),FIMPV(99),NINTS,NIMPV,RMP1
     1,FIBEST(99),FISAVE(99),DIFSAV(99),DTOSAV(99)
      COMMON /WORK1/ TEMP1(220),WORK12(220)
      DIMENSION A(1),SEISIG(1)
C
C  OUTER DO LOOP CALCULATES PARTIALS WITH RESPECT TO A SINGLE THICKNESS
      IJ=1
      DO 10  I=1,NPARAM
C     WRITE(6,1000)I,IEND1
C1000 FORMAT(*DOING PARTIAL FOR THICK # *,2I4)
      PHOLD =DTOB(I)
C
C*************DO NUMERICAL DERIVATIVE BY "CONVOLUTION" WITH DOUBLET
```

```
      CALL NTD(SEISIG(PHOLD),PHOLD,TEMP1,NIMPV).
C*******RESULT IS RETURNED IN TEMP**********
      DO 62 J=1,NIMPV
      A(IJ)=TEMP1(J)
   62 IJ=IJ+1
C     ISTART=IJ-NIMPV
C     IEND=IJ-1
C     WRITE(6,3004)(A(K),K=ISTART,IEND)
C3004 FORMAT(*DEPTH PARTIALS *,5F10.5)
   10 CONTINUE
      RETURN
      END
C
C
C
C*******************************************************************
C
C           SUBROUTINE APCORR ADDS PARAMETER CORRECTIONS
C
C   INPUT      DIFF   PARAMETER CORRECTIONS.
C              NPARAM NUMBER OF PARAMETERS.
C              LBOUND LOGICAL VARIABLE INDICATING IF BOUNDARY ITERATIONS
C                     ARE FINISHED.
C   OUTPUT     CORRECTED PARAMETERS FOUND IN ARRAYS DTOB,FIMPV.
C
C***********************************************************************
      SUBROUTINE APCORR(DIFF,NPARAM,LBOUND)
      LOGICAL LBOUND,LPHAZE,LSCALE,LDF1,LWAVE,LSCALR
      DIMENSION DIFF(1)
      COMMON /WAVBLK2/ SAM,NWAVE,F1,F2,F3,F4,PHZ,LPHAZE,LWAVE,WAVE(100)
     1,NDEL,AF3,WSCALE,F1SAVE,F4SAVE,LDF1
      COMMON /SCLBLK/ LSCALE,SCALE,SCALR,LSCALR
      COMMON /CRTBLK/ DTOB(99),FIMPV(99),NINTS,NIMPV,RMPL
     1,FIBEST(99),FISAVE(99),DIFSAV(99),DTOSAV(99)
      COMMON /WORK1/ WORK11(220),WORK12(220)
      COMMON /WORK2/ WORK21(220),WORK22(220)
C**********BE VERY CAREFUL ABOUT USING WORK ARRAYS
C*******DIFADB MAKES THE BOUNDRY CHANGES
      IF(.NOT.LBOUND)GO TO 710
      CALL DIFADB(DIFF,NPARAM)
C******THE FOLLOWING SWAP IS MADE SO LACK OF BOUNDRY CHANGES
C******CAN BE CHECKED FOR IN MAIN PROGRAM. DIFF IS CHANGED
C******FROM CORRECTION VECTOR TO ERROR VECTOR SOON IN
C******CALL TO FRMS IN EERMS.
      DO 130 I=1,NPARAM
  130 DIFSAV(I)=DIFF(I)
      GO TO 704
C*****************DIFADI MAKES IMPEDANCES CHANGES
  710 CALL DIFADI(DIFF,FIMPV,NPARAM)
      NPLUS=0
      IF(.NOT.LPHAZE)GO TO 703
      NPLUS=NPLUS+1
C*************DIFPHZ CHANGES THE LINEAR PHZ OFFSET
      CALL DIFPHZ(DIFF,PHZ,NPARAM)
C     WRITE(6,3043)PHZ,DIFF(NPARAM+1)
C3043 FORMAT(*NEW PHASE, CORRECTION= *,2F8.4)
C***********ADD CONSTANT SCALE CORRECTIONS
  703 IF(.NOT.LSCALE)GO TO 709
      NPLUS=NPLUS+1
C     WRITE(6,3045)DIFF(NPARAM+NPLUS),SCALE
C3045 FORMAT(*SCALE CORRECTION,OLD SCALE= *,2F10.4)
      SCALE=SCALE+DIFF(NPARAM+NPLUS)
C***********ADD LINEAR SCALE CORRECTIONS
```

```
 709  IF(.NOT.LSCALR) GO TO 711
      NPLUS=NPLUS+1
      WRITE(6,3046)DIFF(NPARAM+NPLUS),SCALR
 3046 FORMAT(*LINEAR SCALE CORRECTION.OLD SCALE *,2F10.4)
      SCALR=SCALR+DIFF(NPARAM+NPLUS)
C**********ADD WAVELET CORRECTIONS
 711  IF(.NOT.LDF1)GO TO 704
      CALL DIFADW(DIFF,NPARAM,NPLUS)
  704 CONTINUE
      RETURN
      END
C
C
C
C
C
C*****************************************************************
C
C     SUBROUTINE DIFADB WILL ADD PARAMETER CORRECTIONS FOR BOUNDRY LOCATIONS
C
C     INPUT       DIFF        PARAMETER CORRECTIONS
C                 NPARAM      NUMBER OF CORRECTIONS
C     OUTPUT      DTOB        CORRECTED BOUNDARY LOCATIONS.
C
C*****************************************************************
C
      SUBROUTINE DIFADB (DIFF,NPARAM)
      DIMENSION DIFF(1)
      COMMON /CRTBLK/ DTOB(99),FIMPV(99),NINTS,NIMPV,RMP1
     1,FIBEST(99),FISAVE(99),DIFSAV(99),DTOSAV(99)
      COMMON /BCBLK/ITER,PCBOLD(99),TRIP(99),BNDTRP
C**********THIS SECTION CALCULATES THE TRIP VALUE FOR EACH BOUNDRY
C**********LOCATION CHANGE
C*********DONT COMPUTE NEW TRIP VALUES IF THIS IS ITER #1
      IF(ITER.EQ.1) GO TO 302
C*********CHECK FOR SIGN CHANGES BETWEEN DIFF(I) AND PCBOLD(I)
      DO 301 I=1,NPARAM
      AVC=ABS(DIFF(I))
      AVO=ABS(PCBOLD(I))
      UNTYC=DIFF(I)/AVC
      UNTYO=PCBOLD(I)/AVO
      POM1=UNTYC/UNTYO
C*********IF POM1=1 , THEN THERE IS NO SIGN CHANGE AND BOUNDRY
C*********SHOULD CONTINUE TO BE MOVED. (IF TRIP VALUE ALLOWS IT)
      IF(POM1.EQ.1.0) GO TO 301
C*********IF POM=-1.0 THE THERE IS A SIGN CHANGE.  IF AVC .GT. AVO
C******THE BOUNDRY CHANGE IS ALLOWED.  IF AVO .GT. AVC THE BOUNDRY CHANGE
C******IS IGNORED. THE BOUNDRY CHANGES ARE ENFORCED BY SETTING TRIP(I).
      IF(AVC.GT.AVO) TRIP(I)=(AVC+AVO)/2.0
      IF(AVO.GT.AVC) TRIP(I)=AVO
      IF(TRIP(I).LT.BNDTRP)TRIP(I)=BNDTRP
 301  CONTINUE
 302  CONTINUE
C******SAVE DIFF(I) IN PCBOLD(I)
      DO 303 I=1,NPARAM
 303  PCBOLD(I)=DIFF(I)
C**********ADD PARAMETER CORRECTIONS
      DO 10 I=1,NPARAM
      TEMP=DIFF(I)
      DIFF(I)=0.0
      IF(TEMP.GT.TRIP(I))DIFF(I)=1.
      IF(TEMP.LT.-TRIP(I))DIFF(I)=-1.
   10 DTOB(I)=DTOB(I)+DIFF(I)
```

```
C***********CHECK FOR ZERO AND NEGATIVE LEANGTH INTERVALS
C***********CREATED BY THE ABOVE CORRECTIONS
      DO 400 I=2,NINTS
      J=I-1
      ID=DTOB(I)
      JD=DTOB(J)
      IF(ID.GT.JD) GO TO 400
C***********NEXT SECTION EXECUTED WHEN 0 OR NEG. LENGTH INTERVAL
C***********IS ENCOUNTERED
C     WRITE(6,402)(DTOB(K),K=1,NPARAM)
C402  FORMAT(*OLD DTOB= *,5F10.4)
C     WRITE(6,403)(DIFF(K),K=1,NPARAM)
C403  FORMAT(*OLD DIFF= *,5F10.4)
      DTOB(I)=DTOB(I)-DIFF(I)
      DTOB(J)=DTOB(J)-DIFF(J)
      DIFF(I)=0.0
      DIFF(J)=0.0
C     WRITE(6,404)(DTOB(K),K=1,NPARAM)
C404  FORMAT(*NEW DTOB= *,5F10.4)
C     WRITE(6,405)(DIFF(K),K=1,NPARAM)
C405  FORMAT(*NEW DIFF*,5F10.4)
 400  CONTINUE
      RETURN
      END
C
C*****************************************************************
C
C    SUBROUTINE  DIFADI WILL ADD PARAMETER CORRECTIONS TO IMPEDANCES
C
C    INPUT      DIFF          PARAMETER CORRECTIONS.
C               N             NUMBER OF CORRECTIONS.
C               FIMPV         VECTOR CONTAINING IMPEDANCES.
C    OUTPUT     FIMPV         CORREECTED IMPEDANCES.
C
C*****************************************************************
C
      SUBROUTINE DIFADI (DIFF,FIMPV,N)
      LOGICAL LRMP
      DIMENSION DIFF(1),FIMPV(1)
      COMMON /SCLBLK/ LSCALE,SCALE,SCALR,LSCALR,LRMP
C     WRITE(6,100)(FIMPV(I),I=1,N)
C100  FORMAT(*FIMPV(I) IN DIFADI= *,5F10.2)
      ICUT=N/2
      DO 10 I=1,N
      J=I+1
      IF(ABS(DIFF(I)).GT.5000.)DIFF(I)=5000.*DIFF(I)/ABS(DIFF(I))
      FIMPV(J)=FIMPV(J)+DIFF(I)
      IF(LRMP.AND.I.GT.ICUT.AND.FIMPV(J).GT.500.)FIMPV(J)=500.
  10  IF(LRMP.AND.I.GT.ICUT.AND.FIMPV(J).LT.0.10)FIMPV(J)=0.10
      RETURN
      END
C
C
C
C*****************************************************************
C
C           SUBROUTINE DIFADW ADD CORRECTIONS TO THE WAVELET
C     INPUT    DIFF     PARAMETER CORRECTIONS
C              NPARAM   NUMBER OF PARAMETERS.
C              NPLUS    LOCATION OF WAVELET CORRECTION PAST NPARAM
C     OUTPUT            CORRECTED VERSION OF F1,F2,F3,F4,PHZ,AF3.
C
C*****************************************************************
```

```
      SUBROUTINE DIFADW(DIFF,NPARAM,NPLUS)
      DIMENSION DIFF(1)
      COMMON /WAVBLK2/ SAM,NWAVE,F1,F2,F3,F4,PHZ,LPHAZE,LWAVE,WAVE(100)
     1,NDEL,AF3,WSCALE,F1SAVE,F4SAVE,LDF1
      NPLUS=NPLUS+1
      IK=NPARAM+NPLUS
C     WRITE(6,3046)DIFF(IK),F1
C3046 FORMAT(*F1 CORRECTION,OLD F1= *,2F12.7)
      IF(ABS(DIFF(IK)).GT.2.)DIFF(IK)=2.*DIFF(IK)/ABS(DIFF(IK))
      F1=F1+DIFF(IK)
      IF(F1.LT.F1SAVE) F1=F1SAVE
C**************CORRECT FREQUENCY NO. 2
      NPLUS=NPLUS+1
      IK=NPLUS+NPARAM
C     WRITE(6,3047)DIFF(IK),F2
C3047 FORMAT(*F2 CORRECTION,OLD F2= *,2F12.7)
      IF(ABS(DIFF(IK)).GT.5.0)DIFF(IK)=5.0*DIFF(IK)/ABS(DIFF(IK))
      F2=F2+DIFF(IK)
      IF(F2.LE.F1)F2=F1+1.0
C*******************CORRECT FREQUENCY NO. 3
      NPLUS=NPLUS+1
      IK=NPLUS+NPARAM
C     WRITE(6,3048)DIFF(IK),F3
C3048 FORMAT(*F3 CORRECTION,OLD F3= *,2F12.7)
      IF(ABS(DIFF(IK)).GT.10.) DIFF(IK)=10.*DIFF(IK)/ABS(DIFF(IK))
      F3=F3+DIFF(IK)
C*************CORRECT FREQUENCY NO. 4
      NPLUS=NPLUS+1
      IK=NPLUS+NPARAM
C     WRITE(6,3049)DIFF(IK),F4
C3049 FORMAT(*F4 CORRECTION,OLD F4= *,2F12.7)
      IF(ABS(DIFF(IK)).GT.5.)DIFF(IK)=5.*DIFF(IK)/ABS(DIFF(IK))
      F4=F4+DIFF(IK)
      IF(F4.GT.F4SAVE)F4=F4SAVE
      IF(F3.GT.F4)F3=F4-1.
C**************CORRECT THE AMPLITUDE AT FREQUENCY NO. 3
      NPLUS=NPLUS+1
C     WRITE(6,3050)DIFF(NPARAM+NPLUS),AF3
C3050 FORMAT(*AF3 CORRECTION,OLD AF3= *,2F12.7)
      IK=NPARAM+NPLUS
      IF(ABS(DIFF(IK)).GT.0.2) DIFF(IK)=0.2*DIFF(IK)/ABS(DIFF(IK))
      AF3=AF3+DIFF(IK)
      IF(AF3.GT.1.0)AF3=1.0
      RETURN
      END
C
C
C
C**********************************************************************
C
C       SUBROUTINE TIMU CALCULATES A TRANSPOSE A WHERE A IS A MATRIX PRT.
C
C     INPUT        PRT    M X N MATRIX TO BE SQUARED.
C                  NIMPV  N DIMENSION.
C                  NPARP1 M DIMENSION.
C
C     OUTPUT       PRT    SQUARED VERSION OF INPUT.
C                  TRACE  TRACE OF THE SQUARED MATRIX.
C
C**********************************************************************
      SUBROUTINE TIMU(PRT,NIMPV,NPARP1,TRACE)
      DIMENSION PRT(1)
      DIMENSION A(19000),SEISAV(220)
```

```
      LEVEL 3,A,SEISAV
      COMMON /ECSBLK/ A,SEISAV
      COMMON /WORK1/ TEMP1(220),TEMP2(220)
C******COPY PRT ONTO A. THIS LETS PRT BE OUTPUT OF TIMU
C     WRITE(6,100)
C100  FORMAT(*HAVE ENTERED TIMU*)
C     WRITE(6,101)
C101  FORMAT(* PRT MAXTIX IS*)
C     DO 102 IK=1,NPARP1
C     WRITE(6,103)IK
C103  FORMAT(* ROW # *,I5)
C     ISTAT=((IK-1)*NIMPV)+1
C     IEND=ISTAT+NIMPV-1
C     WRITE(6,104)(PRT(I),I=ISTAT,IEND)
C102  CONTINUE
C104  FORMAT(10E12.4)
      IEND=NIMPV*NPARP1
      CALL WRITEC(PRT(1),A(1),IEND)
      TRACE=0.0
      DO 10 I=1,NPARP1
      II=((I-1)*NIMPV)+1
      CALL READEC(TEMP1(1),A(II),NIMPV)
C     WRITE(6,105)I
C 105 FORMAT(*BACK FROM ECS ROW # *,I5)
C     WRITE(6,104)(TEMP1(L),L=1,NIMPV)
      DO 10 J=1,NPARP1
      II=((J-1)*NIMPV)+1
      CALL READEC(TEMP2(1),A(II),NIMPV)
      SUM=0.0
      DO 12 K=1,NIMPV
 12   SUM=SUM+TEMP1(K)*TEMP2(K)
      IF(I.EQ.J)TRACE=TRACE+SUM
      IJ=I+(J-1)*NPARP1
 10   PRT(IJ)=SUM
      RETURN
      END
      LOGICAL FUNCTION LANSW(DUMMY)
  100 FORMAT(A2)
 101  FORMAT(*ANSWER YES OR NO*)
 10   READ(14,100)ANS
      LANSW=.TRUE.
      IF(ANS.EQ.2HYE)RETURN
      LANSW=.FALSE.
      IF(ANS.EQ.2HNO)RETURN
      WRITE(5,101)
      GO TO 10
      END
C
C
C*************************************************************************
C
CSUBROUTINE CRTRMP GENERATES CONTINOUS IMPEDANCE PROFILE FROM PARAMETERS
C   GIVING DEPTH AND IMPEDANCE OF EACH INTERVAL.
C
C    INPUT          DTOB     BOTTOM TIME OF EACH INTERVAL.
C                   FIMPV    IMPEDANCE OF EACH INTERVAL.
C                            GRADIENT OF EACH INTERVAL IS IN SECOND HALF.
C
C    OUTPUT         OUTPUT   CONTINUOUS IMPEDANCE PROFILE.
C
C*************************************************************************
      SUBROUTINE CRTRMP (OUTPUT)
```

```
      LOGICAL LPRINT
      COMMON /CKTBLK/ DTOB(99),FIMPV(99),NINTS,NIMPV,RMP1
     1,FIBEST(99),FISAVE(99),DIFSAV(99),DTOSAV(99)
      DIMENSION OUTPUT(1)
C     WRITE(6,998)(FIMPV(I),I=1,60)
C998  FORMAT(*FIMPV= *,5F10.2)
      NTOTIM=1
      LPRINT =.FALSE.
      DO 80 I=1,NINTS
      OUTPUT(NTOTIM)=FIMPV(I)
      NTOTIM=NTOTIM+1
      ISTART=NTOTIM
      IEND=DTOB(I)
      IF(I.EQ.NINTS)IEND=NIMPV
      IF(ISTART.LE.IEND) GO TO 90
      LPRINT=.TRUE.
C     WRITE(6,110)I,ISTART,IEND
C 110 FORMAT(* NEGATIVE/ZERO LEANGTH INTERVAL FOR INTERVAL *,3I5)
      IBOMB=ISTART-IEND
      IF(IBOMB.EQ.1) GO TO 80
   90 RAMP=FIMPV(NINTS+I-1)
      IF(I.EQ.1)RAMP=RMP1
      DO 100 J=ISTART,IEND
      OUTPUT(NTOTIM)=OUTPUT(NTOTIM-1)+RAMP
      NTOTIM=NTOTIM+1
  100 CONTINUE
   80 CONTINUE
C     IF(LPRINT)WRITE(6,122)(OUTPUT(K),K=1,NIMPV)
C122  FORMAT(*BAD IMP= *,5F10.2)
 101  RETURN
      END
C
C
C
C*********************************************************************
C
C     SUBROUTINE NTD CALCULATES NUMERICAL THICKNESS DERIVATIVES BY
C                     REPLACEMENT WITH THE DOUBLET.
C
C    INPUT      AMP    AMPLITUDE AT ARRIVAL TIME WHERE DERIVATIVE IS TAKEN.
C               DEPTH LOCATION OF TIME WHERE DERIVATIVE IS TAKEN.
C               MOLNTH  LEANGTH OF SYNTHETIC SEISMIC TRACE AND OUTPUT.
C               FILNTD  DERIVATIVE OPERATOR (THE DOUBLET).
C
C    OUTPUT     TEMP   DERIVATIVE OF SYNTHETIC SEISMIC TRACE WITH
C                      RESPECT TO A BOUNDARY LOCATION.
C
C*********************************************************************
      SUBROUTINE NTD(AMP,DEPTH,TEMP,MOLNTH)
      DIMENSION TEMP(1)
      COMMON /NUMTD/ LNTD,FILNTD(30),NFLNTD,NTDEL
      CALL ZERO(MOLNTH,TEMP)
      ISTART=IFIX(DEPTH)-NTDEL
      IEND=ISTART+NFLNTD-1
      J=0
      DO 20 I=ISTART,IEND
      J=J+1
 100  FORMAT(*I,J,TEMP(I),FILNTD(J)= *,2I3,2F8.3)
   20 TEMP(I)=FILNTD(J)*AMP
      RETURN
      END
C
C
```

```
C*****************************************************************************
C
C          *SUBROUTINE DIFHZ WILL ADD PARAMETER CORRECTIONS FOR PHAZE
C
C   INPUT       DIFF       PARAMETER CORRECTIONS.
C               PHZ        PHASE VALUE TO BE CORRECTED.
C               NPARAM     LOCATION OF PARAMETER CORRECTION.
C   OUTPUT      PHASE      CORRECTED PHASE.
C.
C***********************************************************************
      SUBROUTINE DIFPHZ(DIFF,PHZ,NPARAM)
      DIMENSION DIFF(1)
      PHZ=PHZ+DIFF(NPARAM+1)
C     WRITE(6,10)PHZ,DIFF(NPARAM+1)
C  10 FORMAT(*NEW PHAZE, CORRECTION= *,2F10.4)
      RETURN
      END
C
C
C
C
C
C****************************************************************************
C
C     SUBROUTINE WAVGEN GENERATES A BAND PASS WAVELET OF ANY LINEAR
C                PHASE.  ALL CALCULATIONS ARE DONE IN TIME DOMAIN.
C
C       INPUT           F1,F2,F3,F4            TRAPAZOIDAL BANDPASS.
C                       AF3                    AMP AT AF3
C                       PHZ                    PHASE OF WAVELET
C
C       OUTPUT          TEMP                   OUTPUT WAVELET.
C
C****************************************************************************
      SUBROUTINE WAVGEN2(TEMP)
      LOGICAL LWAVE,LPHAZE
      DIMENSION TEMP(1)
      COMMON /WAVBLK2/ SAM,NWAVE,F1,F2,F3,F4,PHZ,LPHAZE,LWAVE,FILT(100)
     1,NDEL,AWF3,WSCALE,F1SAVE,F4SAVE,LOF1
      COMMON /WORK1/ PHZOPR(220),WORK11(220)
      COMMON /WORK2/ TEMP1(220),TEMP2(220)
C     WRITE(6,4999)F1,F2,F3,F4,AWF3,PHZ
C4999 FORMAT(*F1,2,3,4,AF3,PHZ= *,6F10.3)
      PHZLOC=PHZ/57.2957795
C
C************CONVERT FREQUENCY WAVLES TO ANGULAR FREQUENCY.
C     WRITE(6,5000)
C5000 FORMAT(*HAVE ENTERED WAVGENZ*)
      WF1=F1*6.2831853
      WF2=F2*6.2831853
      WF3=F3*6.2831853
      WF4=F4*6.2831853
C**********COMPUTE THE WEIGHTING COEFF.S
      AWF2=1.0
      C1=AWF2/(WF2-WF1)
      C2=((AWF3-AWF2)/(WF3-WF2))-(AWF2/(WF2-WF1))
      C3=(-AWF3/(WF4-WF3))-((AWF3-AWF2)/(WF3-WF2))
      C4=AWF3/(WF4-WF3)
C
C**************COMPUTE AREA OF BANDPASS ZONE
      H1=AWF2
      H2=AWF3
      IF(AWF3.LT.AWF2)GO TO 20
```

```
      H1=AWF3
      H2=AWF2
 20   CONTINUE
      AREA=((WF2-WF1)*.5*AWF2)+(H2*(WF3-WF2))+(.5*(WF3-WF2)*(H1-H2))
     1+((WF4-WF3)*.5*AWF3)
      AREA=AREA/3.14159
C*************CALCULATE THE WAVELET
      NCPT=(NWAVE+1)/2
      WSCALE=1/AREA
      TEMP1(NCPT)=1.0
      IEND=NCPT
      DO 30 I=2,IEND
      NN=I-1
      TIME=(NN)*SAM
      TEMP1(NCPT+NN)=-(C1*COS(WF1*TIME)+C2*COS(WF2*TIME)
     1+C3*COS(WF3*TIME)+C4*COS(WF4*TIME))*WSCALE/(TIME*TIME*3.14159)
      TEMP1(NCPT-NN)=TEMP1(NCPT+NN)
 30   CONTINUE
C     WRITE(6,5005)(TEMP1(I),I=1,NWAVE)
C5005 FORMAT(*ZERO PHZ WAVELET= *,5F12.6)
C*******************GENERATE CONVOLUTION OPERATOR FOR PHASE SHIFTING
      IF(.NOT.LPHAZE)GO TO 50
C     WRITE(6,5006)
C5006 FORMAT(*COMPUTING PHASE SHIFTER*)
      PHZOPR(NCPT)=COS(PHZLOC)/SAM
      IEND=NCPT
      DO 40 I=2,IEND
      NN=I-1
      TIME=NN*SAM
      PHZOPR(NCPT+NN)=-SIN(PHZLOC)/(3.14159*TIME)
      PHZOPR(NCPT-NN)=-PHZOPR(NCPT+NN)
 40   CONTINUE
C     WRITE(6,503)(PHZOPR(I),I=1,NWAVE)
C503  FORMAT(*CALL FOLD PHZ OPR= *,5F10.4)
      CALL FOLD(NWAVE,TEMP1,NWAVE,PHZOPR,LC,TEMP2,NDEL)
      DO 55 I=1,NWAVE
 55   TEMP(I)=TEMP2(I)
      GO TO 56
 50   CONTINUE
      DO 52 I=1,NWAVE
 52   TEMP(I)=TEMP1(I)
 56   CONTINUE
C***********REMOVE ANY DC OFFSET TO WAVELET AND RESCALE TO 1.0
      SUMM=0.0
      DO 51 I=1,NWAVE
 51   SUMM=SUMM+TEMP(I)
      AVE=SUMM/NWAVE
      SUMM=0.0
      WSCALE=1./(TEMP((NWAVE+1)/2)-AVE)
      DO 53 I=1,NWAVE
      TEMP(I)=(TEMP(I)-AVE)*WSCALE
 53   SUMM=SUMM+TEMP(I)
      AVE=SUMM/NWAVE
      RETURN
      END
C
C
C*****************************************************************
C
C
C     SUBROUTINE SOLVE WILL SOLVE A MATRIX EQUATION.
C
C        THIS SUBROUTINE WAS WRITTEN BY DR. CHARLES STOYER, C.S.M.
```

```
C
C           EQUATION SOLVED IS Y= B X     B IS A MATRIX
C                                         Y AND X ARE VECTORS
C
C           INPUT    A    SQUARE OF MATRIX B AUGMENTED WITH Y.
C                    N    DIMENSION OF A.  (MATRIX IS SQUARE)
C
C                    NP   DIMENSION OF OUTPUT
C
C           OUTPUT   A    SOLUTION VECTOR (X) WHEN NP=1.
C
C*****************************************************************************
      SUBROUTINE SOLVE (A,N,NP)
      DIMENSION A(N)
      IP=N*N+1
      II=1
      DO40I=1,N
      IF(ABS(A(II)).LT.1.E-17) GOTO 500
      X=1./A(II)
      A(II)=1.
      IP1=I+1
      II1=II+1
      IF(IP1.GT.N) GOTO 11
      DO10J=IP1,N
      A(II1)=X*A(II1)
      II1=II1+1
   10 CONTINUE
   11 IPE=IP
      DO20J=1,NP
      A(IPE)=A(IPE)*X
      IPE=IPE+1
   20 CONTINUE
      IF(IP1.GT.N) GOTO 31
      II1=II+1
      IS=II1+N
      DO30J=IP1,N
      X=A(IS-1)
      A(IS-1)=0.
      NMI=N-I
      IF(NMI.GT.0) CALL VMXU(A(IS),X,A(II1),NMI)
      IS=IS+N
      CALL VMXU(A(IPE),X,A(IP),NP)
      IPE=IPE+NP
   30 CONTINUE
   31 IP=IP+NP
      II=II+N+1
   40 CONTINUE
      IP=N*N+1
      II=IP-1
      JP=NP*(N-1)+IP
      IJ=N
      M=N-1
   50 IF(II.LE.1) RETURN
      IF(A(II).NE.1.) GOTO 510
      JJ=IJ
      IPE=IP
      DO60I=1,M
      X=A(JJ)
      A(JJ)=0.
      CALL VMXU(A(IPE),X,A(JP),NP)
      IPE=IPE+NP
      JJ=JJ-N
   60 CONTINUE
```

```
      IJ=IJ-1
      M=M-1
      JP=JP-NP
      II=II-N-1
      GOTO 50
  500 PRINT 1000,A(II)
 1000 FORMAT(* DIAGONAL ELEMENT = *,1PE13.4)
      RETURN
      STOP "SINGULAR"
  510 PRINT 1000, A(II)
      STOP "BAD DIAG"
      END
      SUBROUTINE VMXU(V,X,U,N)
      DIMENSION V(N),U(N)
      DO1I=1,N
      V(I)=V(I)-X*U(I)
    1 CONTINUE
      RETURN
      END
C
C
C*****************************************************************
C
C   SUBROUTINE SEARCH WILL SEARCH FOR THE OPTIMAL DAMPING FACTOR
C                      WITH WICH TO SOLVE MATRIX EQUATION.
C
C   THIS IS MODIFIED AFTER A VERSION WRITTEN BY DR. CHARLES STOYER. C.S.M.
C
C     INPUT       FINISH,START     END POINTS OF A REGION OVER WHICH TO SEARCH
C
C     OUTPUT      EBEST            BEST(LOWEST) ERROR
C                 CKBEST           DAMPING FACTOR ASSOCIATED WITH EBEST.
C
C*******************************************************************
      SUBROUTINE SEARCH(FINISH,START,EBEST,CKBEST)
      DIMENSION CK(4),ER(4)
      DATA BIAS/1.05/,SHALO/3./
C      INITIALIZE 4 VALUES
      ITER=1
      IF(FINISH.GE.START) STOP "FINISH.GE.START"
      CK(1)=START
      CK(4)=FINISH
      CALL EERMS(EBEST,ER(1),START)
      CALL EERMS(EBEST,ER(4),FINISH)
      DEL=10.**(ALOG10(FINISH/START)/3.)
      DELTA=DEL
      CKNEW=START*DEL
      CK(2)=CKNEW
      CALL EERMS(EBEST,ER(2),CKNEW)
      CKNEW=CKNEW*DEL
      INEW=3
C      CALCULATE NEW ERROR
   30 CK(INEW)=CKNEW
      CALL EERMS(EBEST,ER(INEW),CKNEW)
      K=0
      PRINT 4000,CK
      PRINT 4001,ER
 4000 FORMAT(*RLAMDA=I*,PE13.4)
 4001 FORMAT(* ERROR= *,PE13.4)
C      K WILL INDICATE ERROR MINIMUM
   41 EMIN=ER(1)
      DO50I=2,4
      IF(ER(I).GE.EMIN) GOTO 50
```

```
      K=I
      EMIN=ER(I)
   50 CONTINUE
      EMIN=EMIN*BIAS
      CKBEST=CK(K)
      IF(ER(1).EQ.ER(2).AND.ER(3).EQ.ER(4).AND.ER(2).EQ.ER(3)) RETURN
      IF(ITER.GT.5) RETURN
      ITER=ITER+1
      IF(K.GE.4) GOTO 100
      IF(K.GT.1) GOTO 200
C       K IS 1 (HIGHEST)
      CKNEW=CK(1)/DELTA
      INEW=1
      CK(4)=CK(3)
      CK(3)=CK(2)
      CK(2)=CK(1)
      ER(4)=ER(3)
      ER(3)=ER(2)
      ER(2)=ER(1)
      GOTO 30
C       K IS 4 (LOWEST)
  100 CKNEW=CK(4)*DELTA
      IF((ER(4)/ER(3)).GT.0.95) RETURN
      INEW=4
      CK(1)=CK(2)
      CK(2)=CK(3)
      CK(3)=CK(4)
      ER(1)=ER(2)
      ER(2)=ER(3)
      ER(3)=ER(4)
      GOTO 30
C       K IS 2 OR 3
  200 E1=ER(K+1)/ER(K)
      E2=ER(K-1)/ER(K)
      IF(E1.LT.SMALD .AND. E2.LT.SMALD) RETURN
      IF(CK(K+1)/CK(K).LT.0.9*CK(K)/CK(K-1)) GOTO 300
      CKNEW=SQRT(CK(K+1)*CK(K))
      INEW=K+1
      IF(K.EQ.3) GOTO 400
      CK(4)=CK(3)
      ER(4)=ER(3)
      GOTO 30
  300 INEW=K
      CKNEW=SQRT(CK(K-1)*CK(K))
      CK(4)=CK(3)
      CK(3)=CK(2)
      ER(4)=ER(3)
      ER(3)=ER(2)
      GOTO 30
  400 INEW=3
      CK(1)=CK(2)
      CK(2)=CK(3)
      ER(1)=ER(2)
      ER(2)=ER(3)
      GOTO 30
      END
```