

GENERALIZED LINEAR INVERSE ESTIMATION
OF ACOUSTIC IMPEDANCES FROM
SEISMIC REFLECTION DATA

by
Clark Vandell

CLOSED RESERVE
ARTHUR LAKES LIBRARY
COLORADO SCHOOL OF MINES
GOLDEN, COLORADO 80401

ProQuest Number: 11016585

All rights reserved

INFORMATION TO ALL USERS

The quality of this reproduction is dependent upon the quality of the copy submitted.

In the unlikely event that the author did not send a complete manuscript and there are missing pages, these will be noted. Also, if material had to be removed, a note will indicate the deletion.



ProQuest 11016585

Published by ProQuest LLC (2019). Copyright of the Dissertation is held by the Author.

All rights reserved.

This work is protected against unauthorized copying under Title 17, United States Code
Microform Edition © ProQuest LLC.

ProQuest LLC.
789 East Eisenhower Parkway
P.O. Box 1346
Ann Arbor, MI 48106 – 1346

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

Golden, Colorado

Date July 18, 1979

Signed: Clark Vandell
Clark Vandell

Approved: William A. Schneider
Dr. William A. Schneider

Golden, Colorado

Date July 19, 1979

George V. Keller
Head of Department
Dr. George V. Keller

CONTENTS

	Page
Abstract	
List of Figures	
List of Tables	
Acknowledgements	
Introduction	1
Forward Solution for the Inversion Scheme	5
Linearization of the Forward Solution	11
Singular Value Decomposition	14
Impedance Solutions from Singular Value Decomposition	17
Computer Program Scheme and Approximations	20
Inversion Results	27
Noiseless case	27
Random noise	52
Convolutional noise	94
Scaling noise	116
Errors	128
Feasability of Method	139
Conclusions	141
Appendix	
Programs and subroutines	142
Bibliography	188

ABSTRACT

Linear inverse theory is applied to synthetic seismic reflection data to calculate acoustic impedances as a function of depth. The earth is divided into discrete layers of equal thickness in travel time, a model which assumes the case of plane waves passing through plane parallel stratification. The forward solution for this model provides the relationship needed by inverse theory between the model parameters (impedances) and calculated values (synthetic seismogram). This forward solution is linearized by a Taylor series expansion, and the resulting linear problem is a set of m equations with n unknowns. The set of equations is reformulated by use of singular value decomposition and the resulting matrix equations are solved iteratively for the best solution in a least squares sense.

Solutions to this equation for model traces without noise are stable and converge rapidly. With random noise added, the solutions remain stable for even low S/N, but an irreducible error is encountered. Convolutional noise as well as scaling factor errors cause slower convergence and less stability, but solutions are still reached. Several acoustic impedance models

are used with very little difference observed with regard to the convergence, stability, and accuracy of the method.

LIST OF FIGURES

	Page
Figure 1-1 Layered model for the normal incidence synthetic seismogram.	7
Figure 1-2 Equivalent single layer representation of multilayer model in Figure 1-1.	7
Figure 1-3 Vector representation of sampled trace.	8
Figure 1-4 Forward solution steps from input impedance series.	10
Figure 1-5 General flowchart of inversion procedure.	21
Figure 1-6 Simple three layer model. Surface is excited with an impulse function.	23
Figure 2-1 Inversion results from model 1 with zero noise. The impedance model generates a reflection coefficient of .05 at each interface.	28
Figure 2-2 Model 1 RMS error for zero noise case.	29
Figure 2-3 The observed minus the calculated trace for successive iterations with model 1 and zero noise.	30
Figure 2-4 Inversion results for model 2 with zero noise. The impedance of each layer is incremented by a constant.	34
Figure 2-5 Model 2 RMS error for zero noise case.	35
Figure 2-6 The observed minus the calculated trace for successive iterations with model 2 and zero noise.	36
Figure 2-7 Inversion results for model 3 with zero noise. The impedance varies sinusoidally for this model.	37
Figure 2-8 Model 3 RMS error for zero noise case.	38

	Page	
Figure 2-9	The observed minus calculated trace for successive iterations with model 3 and zero noise.	39
Figure 2-10	Inversion results for Model 4 with zero noise. Impedance varies as a sinc for this model.	40
Figure 2-11	Model 4 RMS error for zero noise.	41
Figure 2-12	The observed minus calculated trace for successive iterations with model 4 and zero noise.	42
Figure 2-13	Inversion results for model 5 with zero noise. Impedances varies aperiodically in steps.	43
Figure 2-14	Model 5 RMS error for zero noise.	44
Figure 2-15	The observed minus calculated trace for successive iterations with model 5 and zero noise.	45
Figure 2-16	Inversion results for model 6 with zero noise. Impedance varies randomly.	46
Figure 2-17	Model 6 RMS error for zero noise.	47
Figure 2-18	The observed minus calculated trace for successive iterations with model 6 and zero noise.	48
Figure 2-19	Inversion results for model 7 with zero noise. Impedance varies periodically.	49
Figure 2-20	Model 7 RMS error for zero noise.	50
Figure 2-21	The observed minus calculated trace for successive iterations with model 7 and zero noise.	51
Figure 2-22	Inversion result for constant impedance model and .006216 RMS noise level.	56

	Page
Figure 2-23 Observed and calculated trace of final inversion iteration for constant impedance model and .006216 RMS noise level.	57
Figure 2-24 Inversion result for constant impedance model and .012985 RMS noise level.	58
Figure 2-25 Observed and calculated trace of final inversion iteration for constant impedance model and .012985 RMS noise level.	59
Figure 2-26 Inversion result for constant impedance model and .027841 RMS noise level.	60
Figure 2-27 Observed and calculated trace of final inversion iteration for constant impedance model and .027841 RMS noise level.	61
Figure 2-28 Inversion result for constant impedance model and .056962 RMS noise level.	62
Figure 2-29 Observed and calculated trace of final inversion iteration for constant impedance model and .056962 RMS noise level.	63
Figure 2-30 Inversion result for constant impedance model and .110179 RMS noise level.	64
Figure 2-31 Observed and calculated trace of final inversion iteration for constant impedance model and .110179 RMS noise level.	65
Figure 2-32 Inversion result for constant impedance model and .237170 RMS noise level.	66
Figure 2-33 Observed and calculated trace of final inversion iteration for constant impedance model and .237170 RMS noise level.	67
Figure 2-34 Inversion results for Model 1 with .006216 RMS noise level.	69
Figure 2-35 Model 1 RMS error for .006216 RMS noise level.	70

	Page
Figure 2-36 The observed minus calculated trace for successive iterations with model 1 and .006216 RMS noise level.	71
Figure 2-37 Inversion results for Model 3 with .006216 RMS noise level.	72
Figure 2-38 Model 3 RMS error for .006216 RMS noise level.	73
Figure 2-39 The observed minus calculated trace for successive iterations with model 3 and .006216 RMS noise level.	74
Figure 2-40 Inversion results for model 5 with .006216 RMS noise level.	75
Figure 2-41 Model 5 RMS error for .006216 RMS noise level.	76
Figure 2-42 The observed minus calculated trace for successive iterations with model 5 and .006216 RMS noise level.	77
Figure 2-43 Inversion results for model 6 with .006216 RMS noise level.	78
Figure 2-44 Model 6 RMS error for .006216 RMS noise level.	79
Figure 2-45 The observed minus calculated trace for successive iterations with model 6 and .006216 RMS noise level.	80
Figure 2-46 Inversion results for model 1 with .012985 RMS noise level.	81
Figure 2-47 Model 1 RMS error for .012985 RMS noise level.	82
Figure 2-48 The observed minus calculated trace for successive iterations with model 1 and .012985 RMS noise level.	83
Figure 2-49 Inversion results for Model 3 with .012985 RMS noise level.	84

	Page
Figure 2-50 Model 3 RMS error for .012985 RMS noise level	85
Figure 2-51 The observed minus calculated trace for successive iterations with model 3 and .012985 RMS noise level.	86
Figure 2-52 Inversion results for model 5 with .012985 RMS noise level.	87
Figure 2-53 Model 5 RMS error for .012985 RMS noise level.	88
Figure 2-54 The observed minus calculated trace for successive iterations with model 5 and .012985 RMS noise level.	89
Figure 2-55 Inversion results for model 6 with .012985 RMS noise level.	90
Figure 2-56 Model 6 RMS error for .012985 RMS noise level.	91
Figure 2-57 The observed minus calculated trace for successive iterations with model 6 and .012985 RMS noise level.	92
Figure 2-58 Convolutional wavelet with zero noise.	95
Figure 2-59 Convolutional wavelet with S/N = 1.	96
Figure 2-60 Convolutional wavelet with S/N = 2.	97
Figure 2-61 Convolutional wavelet with S/N = 4.	98
Figure 2-62 Amplitude and frequency spectrums for wavelet with zero noise.	99
Figure 2-63 Amplitude and frequency spectrums for wavelet with S/N = 1.	100
Figure 2-64 Amplitude and frequency spectrums for wavelet with S/N = 2.	101
Figure 2-65 Amplitude and frequency spectrums for wavelet with S/N = 4.	102

	Page
Figure 2-66	Inversion results for model 6 with uncorrupted convolutional wavelet. 104
Figure 2-67	Model 6 RMS error for uncorrupted convolutional wavelet. 105
Figure 2-68	The observed minus calculated trace for successive iterations with model 6 and uncorrupted convolutional wavelet. 106
Figure 2-69	Inversion results for model 6 and uncorrupted convolutional wavelet. 107
Figure 2-70	Model 6 RMS error for corrupted convolutional wavelet at S/N = 1. 108
Figure 2-71	The observed minus calculated trace for successive iterations with model 6 and corrupted convolutional wavelet at S/N = 1. 109
Figure 2-72	Inversion results for model 6 with corrupted convolutional wavelet at S/N = 2. 110
Figure 2-73	Model 6 RMS error for corrupted convolutional wavelet at S/N = 2. 111
Figure 2-74	The observed minus calculated trace for successive iterations with model 6 and corrupted convolutional wavelet at S/N = 2. 112
Figure 2-75	Inversion results for model 6 with corrupted convolutional wavelet S/N = 4. 113
Figure 2-76	Model 6 RMS error for corrupted convolutional wavelet at S/N = 4. 114
Figure 2-77	The observed minus calculated trace for successive iterations with model 6 and corrupted convolutional wavelet at S/N=4. 115
Figure 2-78	Inversion results for model 6 with .6 constant scaling factor. 119
Figure 2-79	Model 6 RMS error for .6 constant scaling factor. 120

	Page	
Figure 2-80	The observed minus calculated trace for successive iterations with model 6 and .6 constant scaling factor.	121
Figure 2-81	Inversion results for model 6 with $-.01$ linear scaling factor increment.	122
Figure 2-82	Model 6 RMS error for $-.01$ linear scaling factor increment.	123
Figure 2-83	The observed minus calculated trace for successive iterations with model 6 and $-.01$ linear scaling factor increment.	124
Figure 2-84	Inversion results for model 6 with $-.01$ exponential scaling factor increment.	125
Figure 2-85	Model 6 RMS error for $-.01$ exponential scaling factor increment.	126
Figure 2-86	The observed minus calculated trace for successive iterations with model 6 and $-.01$ exponential scaling factor increment.	127
Figure 2-87	Partial derivatives for last iteration on model 6 with zero noise.	131
Figure 2-88	Standard deviation of parameter corrections on last iteration on model 6 with zero noise.	132
Figure 2-89	Partial derivatives for last iteration on model 6 with .012985 RMS noise level.	133
Figure 2-90	Standard deviation of parameter corrections on last iteration on model 6 with .012985 RMS noise level.	134
Figure 2-91	Partial derivatives for last iteration on model 6 with .8 constant linear scaling factor.	135
Figure 2-92	Standard deviation of parameter corrections on last iteration on model 6 with .8 constant linear scaling factor.	136

	Page
Figure 2-93 Partial derivatives for last iteration on model 6 with corrupted convolutional wavelet at $S/N = 1$.	137
Figure 2-94 Standard deviation of parameter corrections on last iteration on model 6 with corrupted convolutional wavelet at $S/N = 1$.	138

LIST OF TABLES

		Page
Table 1	RMS Levels for Noise and Data	54
2	Random Noise Case 1 S/N Levels	68
3	Random Noise Case 2 S/N Levels	68
4	Computer CPU Time Requirements	140

ACKNOWLEDGEMENTS

I wish to express my gratitude to Dr. William A. Schneider of Colorado School of Mines who acted as my thesis advisor and also the members of my thesis committee, Drs. Frank A. Hadsell, David L. Butler, and James E. White.

It is also with thanks I acknowledge Dr. Charles H. Stoyer who spent considerable time helping me understand the inverse theory.

Finally a special thanks is extended to Daniel and Carol Vandell for their interest and encouragement.

INTRODUCTION

As processing technology has developed, more and more interest has been directed toward the area of seismic inversion. Many different approaches to inversion exist and are described in the literature. Kunetz, (1963), did some of the pioneering work, however his method solved for impedances sequentially and hence errors increased with depth and the method was very unstable. Bamberger et al, (1977,1978), attacked the problem via the one-dimensional wave equation. Fairly good results were achieved, however, there was some undesirable degradation of the latter part of the results in the presence of noise. Also the method was computationally costly with respect to computer time.

In another slightly different attempt, again using the one-dimensional wave equation, Gjevik et al, (1976,1978) transformed into the frequency domain and carried out their solution there. The accuracy of the solutions was good, but some instability was encountered. Most of these methods have the difficulty of lengthy computations. Lindseth, (1972), avoided the problem of time with the computationally fast Seislog* method but at the expense of the accuracy of other approaches.

* Registered trademark

Seismic inversion in the perfect sense would allow the physical earth parameters that determine the nature of the seismic data recorded in the field to be recovered exactly. These physical earth parameters, in the case of seismic exploration, are usually agreed upon as being the velocities and densities as a function of depth, and for the simpler one-dimensional problem, the impedances or product of the velocities and densities. With these parameters, a seismic synthetic can be generated using some forward modeling technique.

The forward solution has been known for some time but in 1961, Goupilland stated the solution in a particularly compact form for the case of plane waves propagating through plane parallel stratification. The basic idea that led to this compact representation was a division of the earth into parallel layers which were equal in depth expressed in travel time units. The fact that the forward solution can be expressed in a simple compact form is important in that this is the starting point for most inversion techniques. Linear inverse theory, in particular, requires that there be a functional relationship between the model parameters and the calculated values that we wish to compare with observed values. For all subsequent references to model parameters, it will be assumed, unless otherwise specified, that these parameters are just the impedances

associated with the different layers.

The object of the problem, then, is to investigate the relationship between a set of observations and a set of model parameters. When the difference between values calculated, using our forward modeling technique, and the values observed is minimized in a least squares sense, we have reached a solution.

One thing immediately obvious is that the model determines to a large extent how well the earth structure can be determined. Characteristics not included in the model can not be resolved as part of our earth model. Goupillard's forward technique assumes plane parallel boundaries. Many times this is a valid approximation over small distances however Bamberger, (1978), notes that this can be approximated even more closely by common recording point stacking which simulates a line source. An advantage of Goupillard's technique is that although it does assume plane boundaries, it includes the effect of all primaries, multiples and associated transmission losses.

With this forward modeling algorithm, it is possible to approach the problem of inversion. To implement linear theory, the forward solution, since it is not a linear expression, needs to be expanded in a Taylor series with non-linear terms dropped. The main effect of this will be just to cause the solution to be an approximation also. This necessitates the the need to iterate several times to converge to a solution.

For the solution of the set of simultaneous linear equations which arise from the linear inverse theory, singular value decomposition is used. The choice of this method is made for two basic reasons:

- (1) The method is stable, a feature important especially when solving a non-linear problem in a linear setting (Wiggins, 1972).
- (2) The eigenvalue decomposition yields some insight as to the relationships between the parameters, observations, and associated errors.

Because noise in its various forms is present on all seismic data to some degree, its effect is considered by adding it to various models and studying the result. Main considerations are stability, convergence and how errors are distributed in the model parameters.

The objectives of this study are to implement these ideas on inversion and apply them to some seismic models.

FORWARD SOLUTION FOR THE INVERSION SCHEME

As was mentioned in the introduction, the forward solution is the starting point in most inversion schemes. Since it is also required in linear inverse theory, we begin by examining Goupilland's formulation of the forward solution. The initial step is to break the earth into equal time thickness layers (Figure 1-1). We can note that while in terms of distance the layers may not be of equal thickness, they correspond to equal time units.

This formulation also allows to talk about discrete numbers rather than a continuum of points which in turn allows us to transform into the z-domain, with each Δt corresponding to one layer thickness in time (Figure 1-3).

As can be easily calculated for a single layer, the reflectivity series, as measured at the top interface (0) is just:

$$R_0(z) = \frac{c_0 + c_1 z}{1 + c_0 c_1 z} \quad (1)$$

Accordingly the synthetic seismogram can be computed by simply performing the polynomial division out to the required length. For the more general case, however, there are an arbitrary number of layers, not merely one. If the response from layer 1 could be expressed as a unit spike response, we would be back to a single layer model (Figure 1-2)

In fact this can be done and yields:

$$R_0(z) = \frac{c_0 + R_1 z}{1 + c_0 R_1(z)} \quad (2)$$

This expression results from merely replacing c_1 in equation 1 with R_1 , which is our generalization of the reflection response. Equation 2 still does not yield a means for calculating R_0 as we have no explicit equation for R_1 . It is possible, however, to repeat the above process and express R_1 in terms of R_2 , R_2 in terms of R_3 , etc until we reach the deepest layer interface.

Equation 2 can then be expressed as:

$$R_j(z) = \frac{c_j + R_{j+1} z}{1 + c_j R_{j+1} z} \quad j = 0, 1, 2, 3 \dots n-1 \quad (3)$$

where $R_n = c_n$ with c_n being the reflection coefficient of the bottom of the n^{th} layer.

Larner et al (1977) follow essentially the same development but note that equation 3 is computationally inefficient. By expressing $R_j(z)$ in terms of a ratio of two polynomials, a more efficient equation for computing synthetic seismograms can be gained.

By substituting the ratio of two polynomials, B_j/A_j for R_j , we have:

$$R_j = \frac{c_j A_{j+1} + B_{j+1} z}{A_{j+1} + c_j B_{j+1} z} = \frac{B_j}{A_j} \quad (4)$$

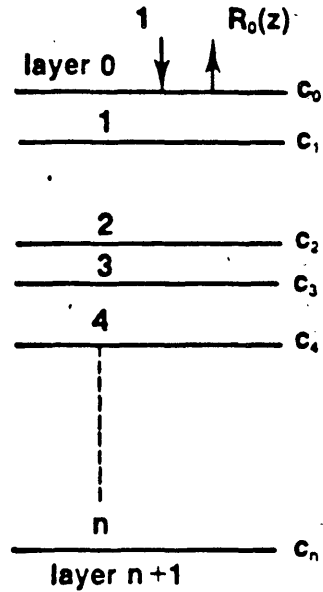


Figure 1-1

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

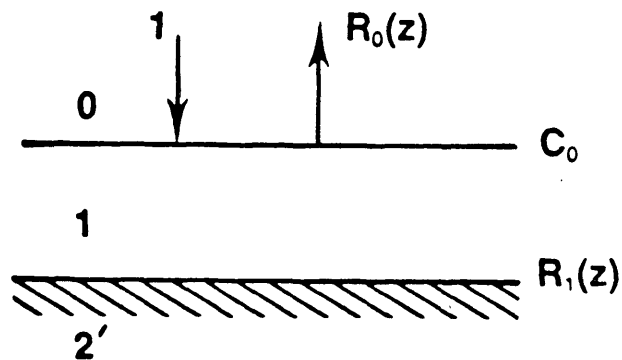


Figure 1-2

Equivalent single layer representation of the multilayer model in Figure 1. $R_1(z)$ is a generalization of c_1 . (After Larner et. al. 1977)

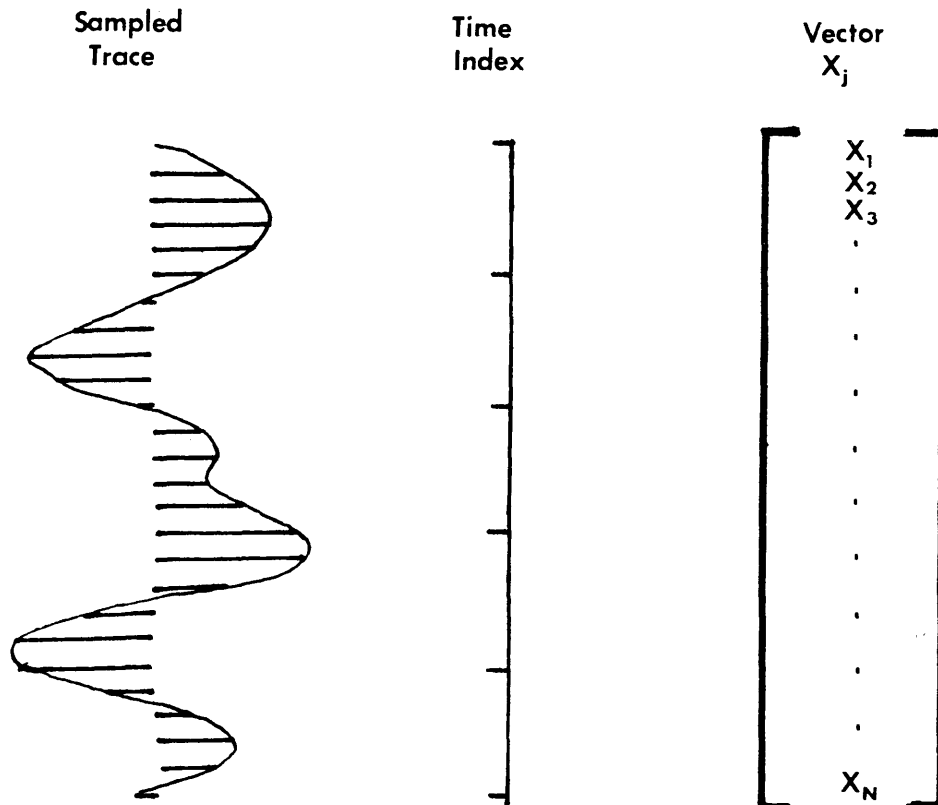


Figure 1-3 Vector representation of sampled trace.
(After Alam, 1977)

where $A_j = A_{j+1} + c_j B_{j+1} z$

$$j = n-1, n-2 \dots 1, 0$$

$$B_j = c_j A_{j+1} + B_{j+1} z$$

Since $B_n = c_n$ and $A_n = 1$ or $B_n/A_n = c_n$, we have our first values to solve the recursive equations for A and B and can therefore calculate the response at the surface.

Finally we note that this is very fast computationally and includes all primaries, multiples, and transmission losses for a normal incidence seismogram (Figure 1-4).

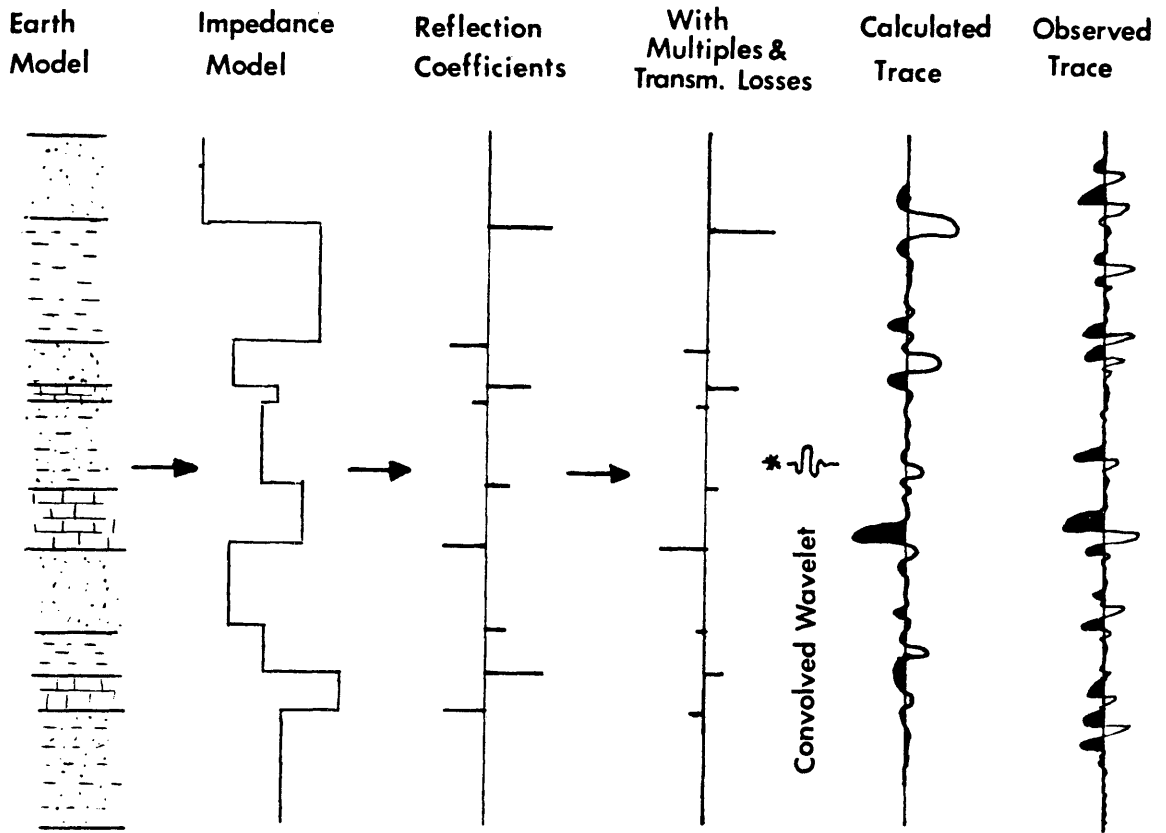


Figure 1-4 Forward solution steps from input impedance series.

LINEARIZATION OF THE FORWARD SOLUTION

Equation 4 is an equation expressing a synthetic seismogram in terms of the parameters (impedances) for which we wish to solve.

Very simply put:

$$F(P_i) = C_j \quad \begin{array}{l} i = 1, 2, \dots, m \\ j = 1, 2, \dots, n \end{array} \quad (5)$$

where P_i is a set of parameters

C_j is a set of observations

Unfortunately our expression of a synthetic seismogram in terms of impedances is not a linear expression as required by linear inverse theory. This can be remedied by expanding the forward solution expression in terms of a Taylor series.

By discarding all second and higher-ordered terms, we are left with

$$O_j - C_j = \sum_{i=1}^m \frac{\partial C_j}{\partial P_i} \Delta P_i \quad j = 1, 2, \dots, n \quad (6)$$

Here the subscript j represents time units in the same sense it did in equation 4 while i represents different model parameter indices. What we have done is expand equation 5 about some point near O_j or our observed values. Since our calculated values will not match the observed values, all our model parameters will be perturbed by some ΔP_i . By solving

for the ΔP 's, we can modify our initial guess of values for each parameter until they yield a forward solution that matches our observed values.

Wiggins (1972) refers to this procedure as the dual set of processes of parameterization of the model and linearization of the forward solution. Simply put, it describes the formulation of the forward model scheme in terms of the parameters we wish to invert to and subsequent linearization of the forward scheme.

Since equation 6 rests on a Taylor series expansion, the expression is only strictly true for small values of ΔP . Small depends largely on what type of observations we are considering. For the case of seismic reflection data, the partial derivatives vary very slowly, allowing initial guesses which can be relatively far from the solution impedances.

Equation 6 is a system of simultaneous linear equations which can also be expressed in matrix form:

$$A\Delta\vec{p} = \vec{d} \quad (7)$$

Matrix A contains the partial derivatives, $\Delta\vec{p}$ is a vector made up of the parameter corrections we need to solve for, and \vec{d} is a difference vector made up of the difference at each Δt time interval between our observed and calculated data.

More explicitly stated in matrix form, equation 7 can be expressed.

$$\begin{bmatrix} \frac{\partial C_1}{\partial P_1} & \frac{\partial C_1}{\partial P_2} & \cdots & \frac{\partial C_1}{\partial P_N} \\ \frac{\partial C_2}{\partial P_1} & \frac{\partial C_2}{\partial P_2} & \cdots & \frac{\partial C_2}{\partial P_N} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial C_M}{\partial P_1} & & & \frac{\partial C_M}{\partial P_N} \end{bmatrix} \begin{bmatrix} \Delta P_1 \\ \Delta P_2 \\ \vdots \\ \Delta P_N \end{bmatrix} = \begin{bmatrix} O_1 - C_1 \\ O_2 - C_2 \\ \vdots \\ O_M - C_M \end{bmatrix} \quad (8)$$

$m = 1, \dots, M$ observations $n = 1, \dots, n$ parameters

Without the presence of noise, the solution to equation 7 is unique.

Looking more closely at equation 7, it can be seen that the number of observations can be as large as we choose as long as it equals the number of parameters. Even when we consider N layers, although the synthetic seismogram will have primary events down to only $N\Delta t$, the multiples will continue indefinitely or until we truncate the series.

SINGULAR VALUE DECOMPOSITION

With equations 7 or 8, we are left with a system of simultaneous linear equations to solve for our unknowns. The system of equations

$$A\Delta\vec{p} = \vec{d}$$

can be solved in a least squares sense in several ways. Most straightforward would be to premultiply both sides of the equation by A^T yielding

$$A^T A \Delta\vec{p} = A^T \vec{d}$$

If both sides are multiplied by $(A^T A)^{-1}$, the solution is reached

$$\Delta\vec{p} = (A^T A)^{-1} A^T \vec{d} \quad (9)$$

The $\Delta\vec{p}$ vector contains the parameter changes needed to be made for each layer to force the calculated values into closer agreement with our observed data.

This particular solution rests on the supposition that $A^T A$ is not singular. This is equivalent to saying that we have at least M simultaneous independent linear equations to solve for M unknowns. If any of our M equations are not independent of each other, our problem is underconstrained and we cannot solve for all the unknowns. Although in our

case the problem is well-constrained, the matrix $A^T A$ may be very close to being singular. This can give rise to errors and stability problems. Lancsoz, (1961), looks at the problem in terms of spectral decomposition.

In all cases it is possible to decompose the matrix A into

$$A = U\Lambda V^T$$

where U is a set of orthonormalized eigenvectors associated with the eigenvectors of AA^T and our "observation" space, V is a set of orthonormalized eigenvectors of $A^T A$ associated with our "parameter" space, and Λ is a diagonal matrix containing the eigenvalues of $A^T A$. The eigenvectors can be viewed as components in our new observation and parameter space while the eigenvalues can be viewed as amplitudes associated with the various components. When a parameter is not able to be resolved from the system of linear equations, the decomposition will yield eigenvalues which are either very small or are in fact, zero.

Wiggins, (1972), notes two methods of handling eigenvalues which are either zero or very close to zero. One method is merely to truncate the U , Λ and V matrices appropriately, leaving the eigenvectors and eigenvalues associated with the poorly defined parameters out of the problem entirely.

One of the advantages of singular value decomposition lies in this very ability to distinguish which parameters we cannot solve for with the available observations. This particular method, however, gives us no information about our poorly defined parameters which have eigenvectors in the "null space" of A. There is no way to solve for the parameters associated with the truncated eigenvalues. This can be avoided by employing another method in which we adjust our Λ matrix by adding some small damping factor. This can best be illustrated by looking at the solution for $\Delta \vec{p}$.

IMPEDANCE SOLUTIONS FROM SINGULAR VALUE DECOMPOSITION

To solve the equation

$$A\Delta\vec{p} = \vec{d}$$

using singular value decomposition, we substitute the decomposition of A for A in the above equation giving

$$U\Lambda V^T\Delta\vec{p} = \vec{d}$$

As noted before, both U and V are orthonormalized eigenvector matrices. This means U and V have the following property,

$$U^T U = U U^T = V^T V = V V^T = I$$

where I is the identity matrix. We can premultiply both sides by U^T giving

$$\Lambda V^T\Delta\vec{p} = U^T\vec{d}$$

then multiply both sides by Λ^{-1}

$$V^T\Delta\vec{p} = \Lambda^{-1}U^T\vec{d}$$

and finally multiplying both sides by V yielding

$$\Delta\vec{p} = V\Lambda^{-1}U^T\vec{d} \tag{10}$$

By adding a small number or damping factor to the diagonal of the Λ^{-1} matrix, we can retain all eigenvalues and solve for every parameter. This procedure is equivalent to the

more familiar practice of adding a term to the diagonal of matrix A to prevent it from being singular. To show this we again start with the basic decomposition equation.

$$A = U\Lambda V^T$$

Taking the tranpose of both sides

$$A^T = V\Lambda U^T$$

therefore

$$A^T A = V\Lambda^2 V^T$$

If we now take equation 9 and add a r^2 term to the diagonal we reach

$$\hat{p} = \left[A^T A + r^2 I \right]^{-1} A^T \hat{d}$$

By taking this equation and substituting the appropriate decomposition of $A^T A$

$$\begin{aligned} \hat{p} &= \left[V\Lambda^2 V^T + V \overset{\hat{r}}{r^2} V^T \right]^{-1} V\Lambda U^T \hat{d} = \\ &\quad \left[V\Lambda_r^2 V^T \right]^{-1} V\Lambda U^T \hat{d} \end{aligned}$$

where $\Lambda_r^2 = \Lambda^2 r^2$

Further reducing this expression

$$\hat{p} = V\Lambda_r^2 V^T V\Lambda U^T \hat{d}$$

If we let $\Lambda' = \Lambda_r^{-2} \Lambda$

$$\text{then } \vec{p} = V \Lambda' U^T \vec{d} \quad (11)$$

We see that this is equal to equation 10 with the exception that Λ^{-1} is now replaced by Λ' .

Johansen (1977) showed that the best choice of r usually is in the range of the eigenvalues.

COMPUTER PROGRAM SCHEME AND APPROXIMATIONS

Equation 11 is the least squares solution to the initial problem set forth; that of determining a correction for each parameter so that the resulting calculated synthetic matches the observed values. The general procedure is outlined in the flowchart in figure 1-5.

Within the program there were several approximations used in calculating the values to insert into equation 11. First of all there is the matter of calculating values for the partial derivative matrix. In illustrating the number of calculations involved for an exact solution, we will consider the model in figure 1-6. It should be noted this is a model using reflectivity coefficients rather than impedances. This merely simplifies the example and the result is easily extended to an impedance model.

If we were to just compute the response of the above model to a unit impulse at the surface, the synthetic seismogram recorded at the surface would be:

$$\begin{aligned}
 R_0 = & c_0 + (1-c_0^2)c_1z + \left[(1-c_0^2)(1-c_1^2)c_2 + (1-c_0^2)(c_1^2)(-c_0) \right] z^2 \\
 & + \left[(1-c_0^2)(1-c_1^2)(1-c_2^2)c_3 + (1-c_0^2)(1-c_1^2)(c_2^2)(-c_1) + \right. \\
 & \left. (1-c_0^2)(1-c_1^2)(c_1)(-c_0) + (1-c_0^2)(c_1^3)(-c_0^2) \right] z^3
 \end{aligned}$$

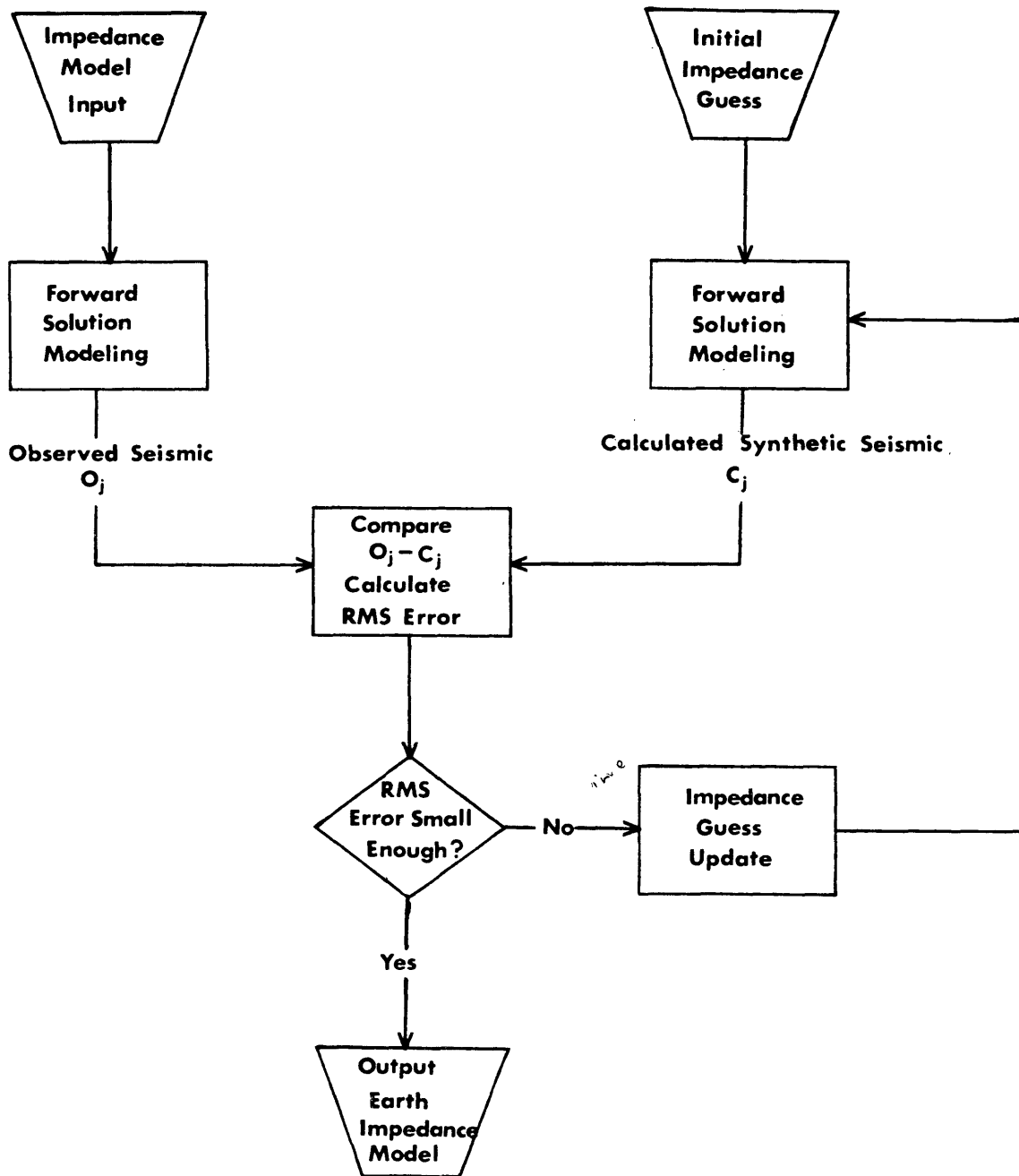


Figure 1-5 General flowchart of inversion procedure.

For time 0 or z^0 the response is given by one term, c_0 ; for time 1, one term; time 2, two terms; and time 3, four terms. If the model were extended to more layers it would become apparent that the number of terms needed to express the primary and all the multiples at a particular depth in time would increase as 2^{N-1} where N is equal to the depth in Δt units. For example, if the model contained 20 layers, the expression needed to give the multiple and primary response at time $20\Delta t$ would contain 2^{19} terms ect. Fortunately if we take equation 4

$$B_j/A_j = R_j$$

instead, we reduce the terms we need to express R_j .

Although this drastically reduces the number of terms we need to deal with, we are still left with a large number of calculations to perform.

Again using the model in figure 6 to illustrate the problem

$$\frac{\partial R_0}{\partial c_2} = \frac{\partial}{\partial c_2} \left(\frac{B_0}{A_0} \right) = A_0 \frac{\frac{\partial B_0}{\partial c_2} + B_0 \frac{\partial A_0}{\partial c_2}}{A_0^2}$$

This is one partial derivative we need to calculate. We can see that as A_0 is the result of a recursive calculation, it is the sum of many terms. We need to trace our way back

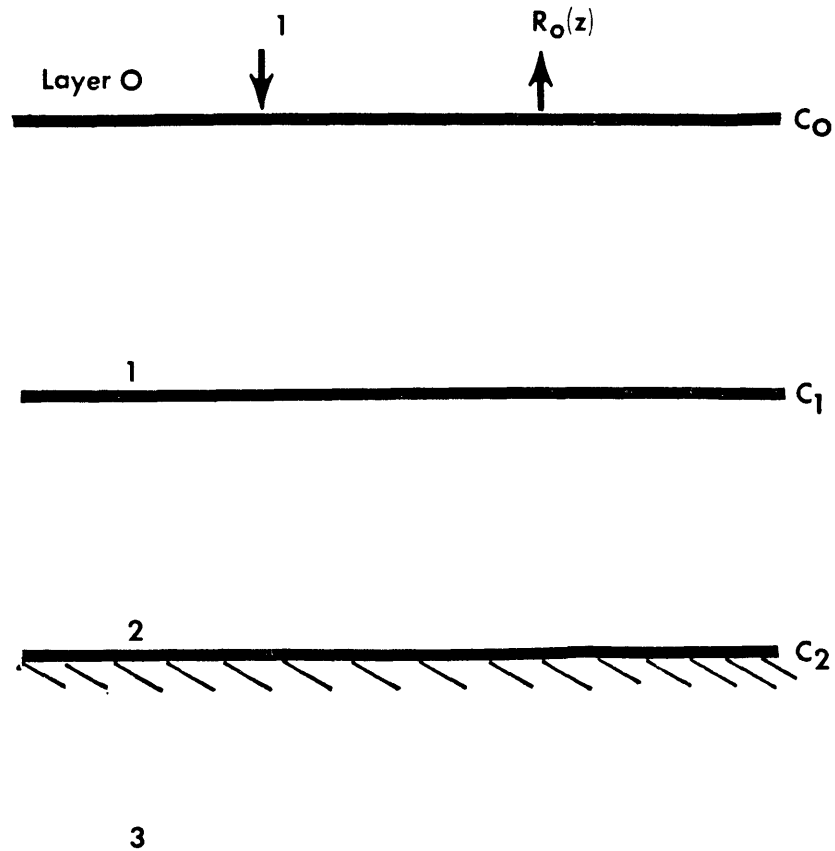


Figure 1-6 Simple three layer model. Surface is excited with an impulse function.

until we find where c_2 first appears, take the partial derivative, and then use this expression in the recursive scheme to proceed forward to A_0 again. This was in fact done in the program and gives an exact partial derivative.

As a faster alternative, a finite difference method was also tried. Since the calculated synthetic is a function of the model parameters,

$$F(P_i) = C_j$$

where j represents increments in time units (for observations) and P_i represents the different model parameters.

From the definition of a partial derivative,

$$\lim_{\Delta P_i \rightarrow 0} \frac{F(P_i + \Delta P_i) - F(P_i)}{\Delta P_i} = \frac{\partial C_j}{\partial P_i}$$

Varying ΔP_i from $.1P_i$ to $.001P_i$, very little difference was found from the values of the exact partials. For this reason, the finite difference partial derivations were used rather than going through the extensive calculations needed to produce the exact partials.

The other modification of equation 11 was logarithmic damping. In generalized linear inversion, many times the parameter corrections, although correct in sign, may be too large in magnitude. This proved to be true for the particular case of seismic inversion also. To reduce the magnitude of

the larger corrections and force a smoother convergence, log-tan weighting was used.

It was implemented as following Stoyer (personal communication, 1977)

$$\left[\frac{\partial \tan C_i}{\partial \log P_i} \right] \left[\Delta P_i \right] = \left[\tan(O_i - C_i) \right]$$

The parameter corrections are then added as:

$$\text{New } P_i = \text{Old } P_i \times 10^{\Delta P_i}$$

This particular damping technique is not the only one possible. It is a variation on the more familiar log-log damping. The variation was necessary since c_i , being the reflectivity series can have negative values.

A simpler but less effective method of damping that was tried was to merely limit the ΔP correction to some fixed value. This has the disadvantage of sometimes hindering convergence to a greater degree.

An algorithm developed by Stoyer (personal communication, 1977) was used with some modifications to choose the damping factor r . The routine checks the errors for various values of r and chooses the value which generates the minimum error.

Finally it should be noted that equation 11 requires a difference between a calculated synthetic trace and an observed synthetic trace. To generate an initial calculated synthetic

trace, an initial guess for the impedance layers is needed. A computationally very fast way to generate these initial impedances, other than guessing some constant value, is to implement a recursive inversion algorithm put forth by Lindseth (1972)

$$z(i+1) = z(i) \frac{1+R(i)}{1-R(i)}$$

This assumes the seismic trace consists of only reflection coefficients. Although inaccurate, it has proven to be a good initial guess.

INVERSION RESULTS

In order to test this method of inversion, seven models were used to evaluate stability and the rate of convergence. In addition, various forms of distortion were introduced to the calculated values derived from the model. First of all random noise was added to the signal and the results of the inversion were compared to the known model parameters. Again, adding a form of noise, a wavelet was convolved with the signal, and finally scaling was introduced. Each operation served to distort the reflectivity series and modify the final determination of the model parameters. Since our "observed" data is generated from a known impedance series, in all cases we know how much our inversion results deviate from the "correct" values.

First of all let us examine the noiseless case.

a) Noiseless case

Model 1 is an impedance series of forty layers which gives rise to a constant reflection coefficient of .05 at each impedance boundary.

In figure 2-1, the dashed line represents the initial guess of the impedance distribution.

As previously stated, the method of determining the initial guesses was simply a Seislog* inversion. This served the dual function of

*Registered trademark.

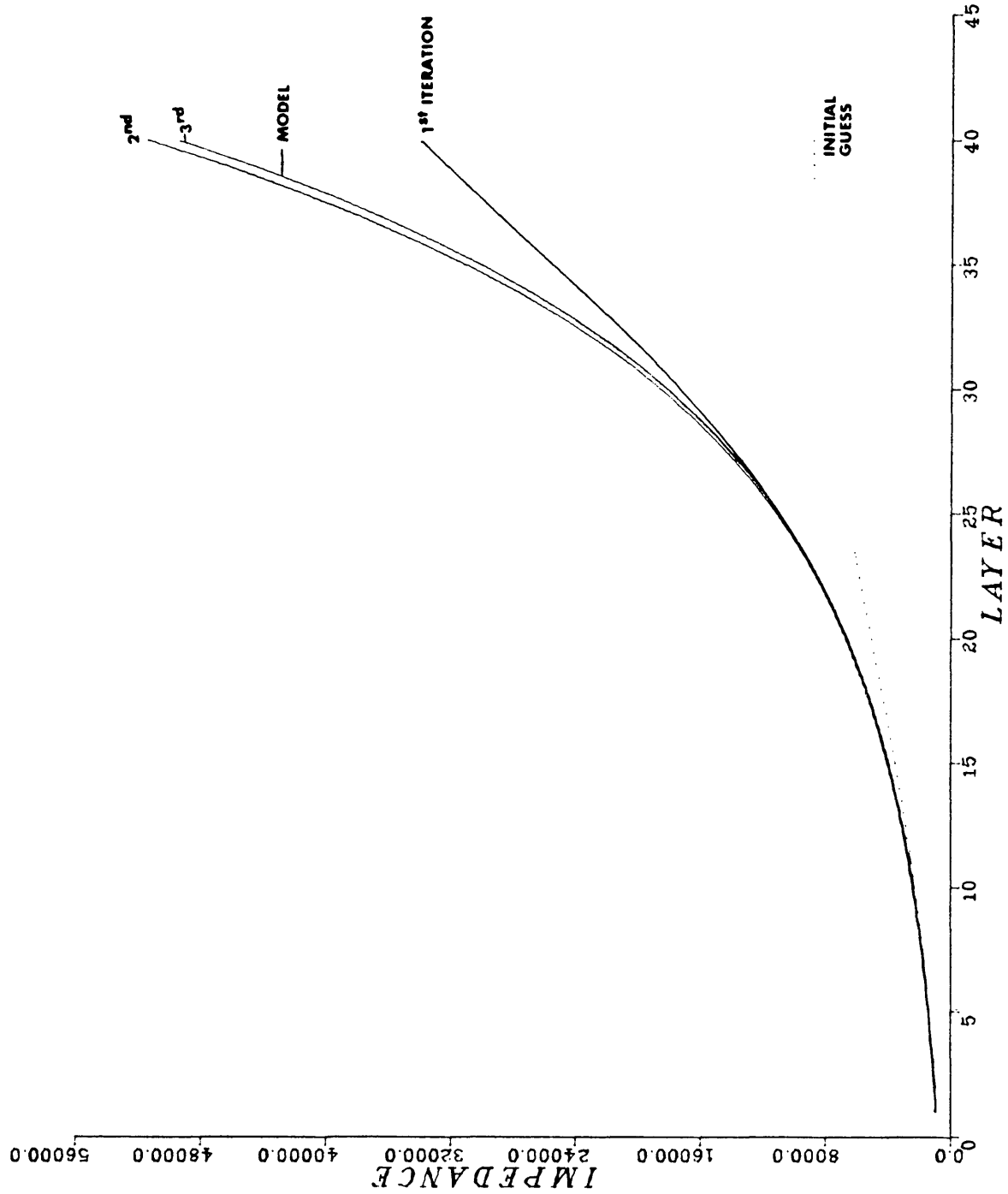


Figure 2-1 Inversion results for model 1 with zero noise. The impedance model generates a reflection coefficient of .05 at each interface.

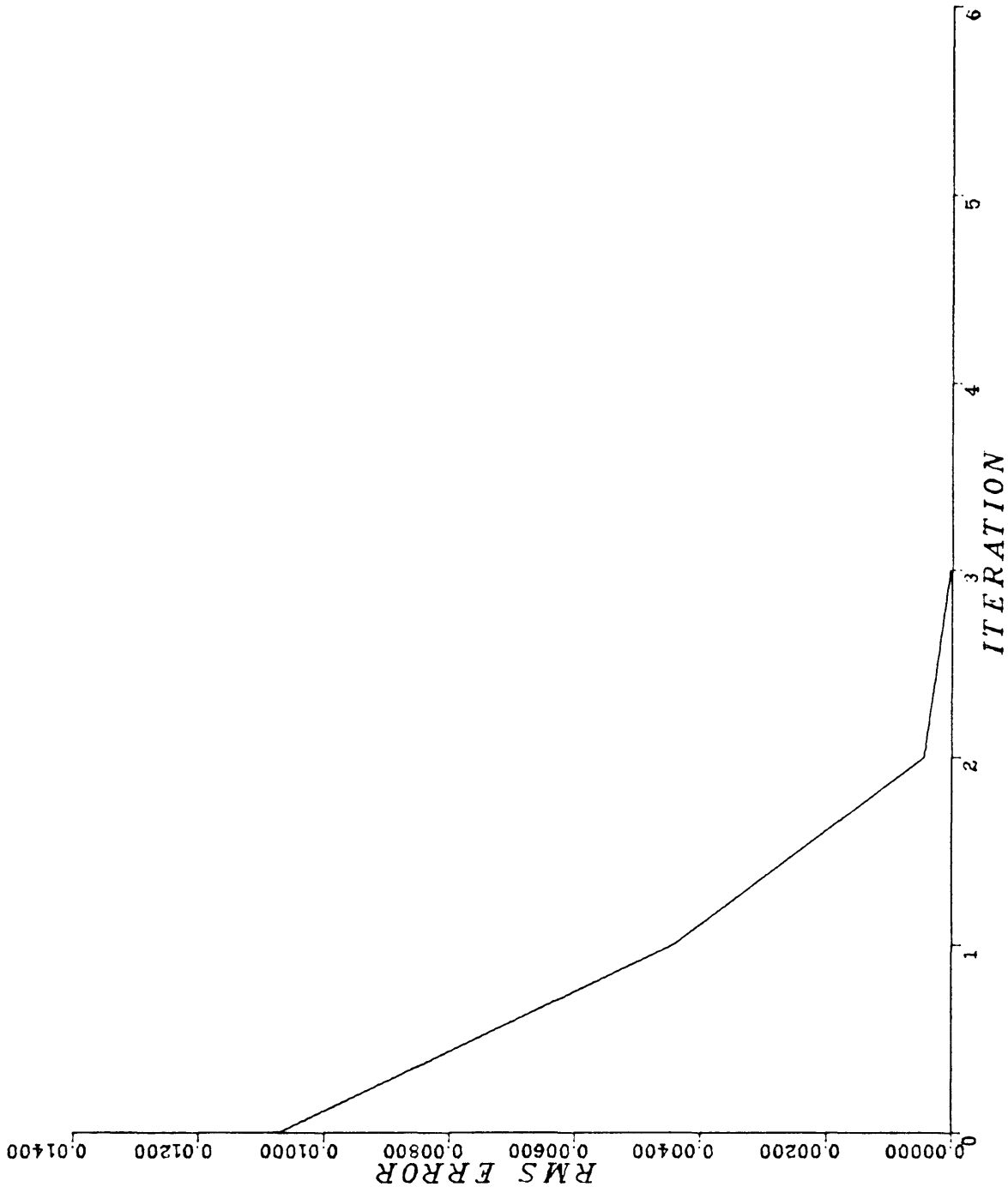


Figure 2-2 Model 1 RMS error for zero noise case.

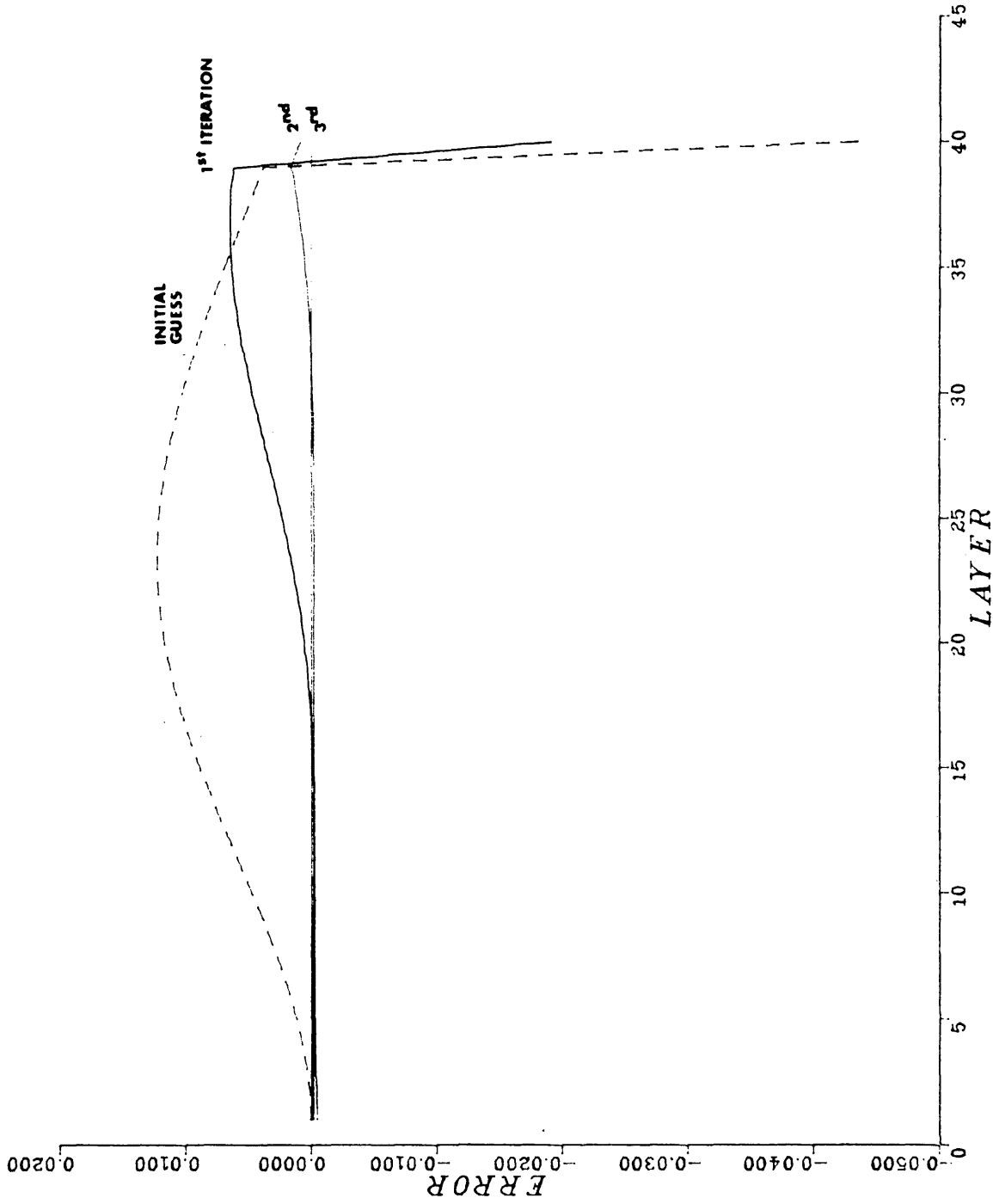


Figure 2-3 The observed minus the calculated trace for successive iterations with model 1 and zero noise.

generating a reasonable initial guess and at the same time comparing a commonly used inversion technique with the method outlined here.

In examining figure 2-1, one notices immediately the divergence of the initial guess from the true model parameters as the layers increase in number. This serves to illustrate what Bamberger et al (1978) commented on in their paper on inversion. That is, that multiples have a very noticeable effect and cannot be safely ignored. As the seislog inversion method neglects both multiples and transmission losses, it mistakenly assumes a leveling off of impedance contrast and the errors tend to be magnified with depth.

The iteration numbers are noted on the plot, and by the third iteration the values generated for the impedances by the program are indistinguishable from the model impedances. The RMS difference between the seismic trace, which was calculated from impedances generated by the inversion process, and the observed trace is referred to as the RMS error and is plotted in Figure 2-1. This name could be equally well applied to the RMS difference between the cal-

culated and model impedances except for the fact the true impedances of each layer in the earth are not known in practice. What we observe is a seismic trace composed of primaries, multiples and associated transmission losses - a reflectivity series, not an impedance series. For this reason the error in figure 2-3 is meant to be the difference between what we observe for a seismic trace and what we calculate using the impedances gained from the inversion process in the forward solution method. This particular display is also a bit misleading without a brief explanation. In figure 2-1, we noted an increasing discrepancy between the true and calculated impedances as we increased in depth. Figure 2-3, however, shows an error reaching a maximum at about layer 23 with an abrupt increase in error at layer 40. The decrease in the magnitude of the errors after layer 23 is due to the fact the energy reaching the lower levels is less. In fact, if the errors were computed as a function of the energy reaching each level, the ratio of error vs. energy would be increasing.

The abrupt change at time or layer 40 is due to the fact that this is a pure multiple. The model consists of 40 layers that create 39 interfaces which generate primaries. By the time we reach time $40\Delta t$, we are only receiving multiples. It is easy to see why an inversion technique which neglects multiples will give a bad result when faced with a signal only comprised of multiples.

Another feature evident from figure 2-3 is that the errors are reduced to zero more quickly at the shallower layers or times. This is true for all the models tested and seems to be a property of this inversion technique. At the same time, in the noiseless case, this is only a delay in error reduction, not a barrier. If sufficient iterations are performed, the error is reduced to zero for all units or layers.

Looking at model 2, figure 2-4, the same convergence is noted. Many of the comments directed toward model 1, are also applicable to all seven model. Figure 2-4 shows convergence at the third iteration and figure 2-5 illustrates the rapid rate of convergence.

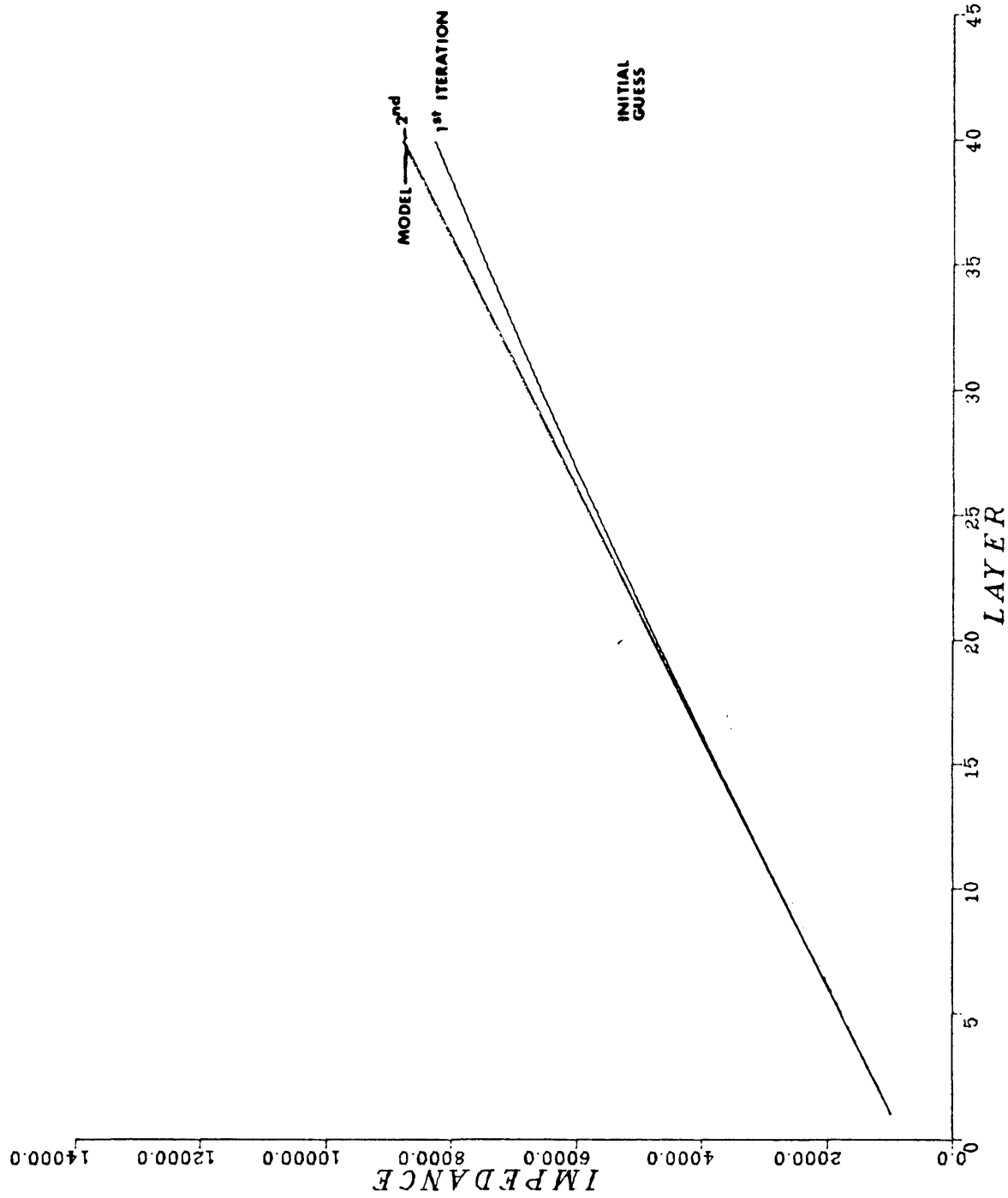


Figure 2-4 Inversion results for model 2 with zero noise. The impedance of each layer is incremented by a constant.

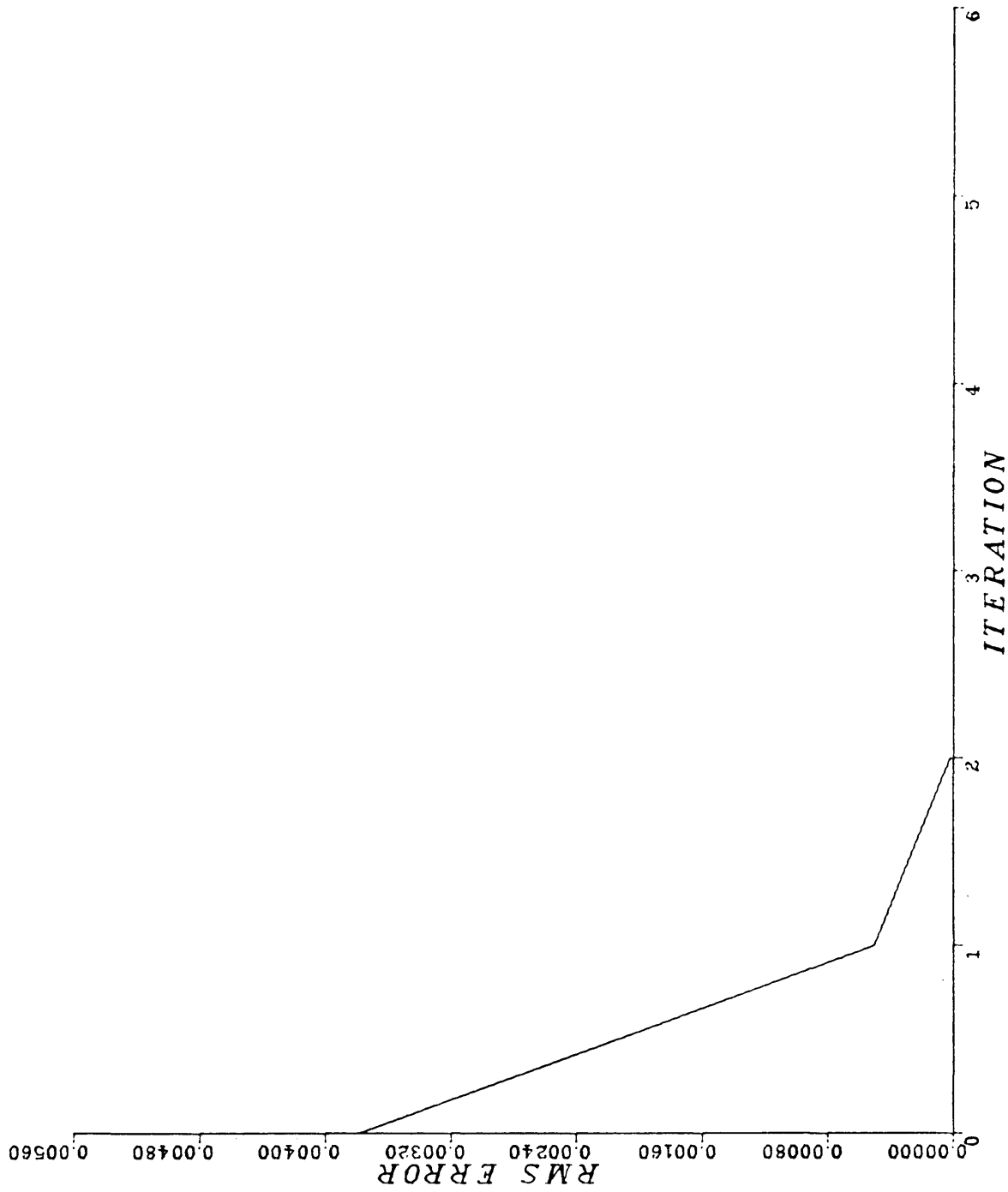


Figure 2-5 Model 2 RMS error for zero noise case.

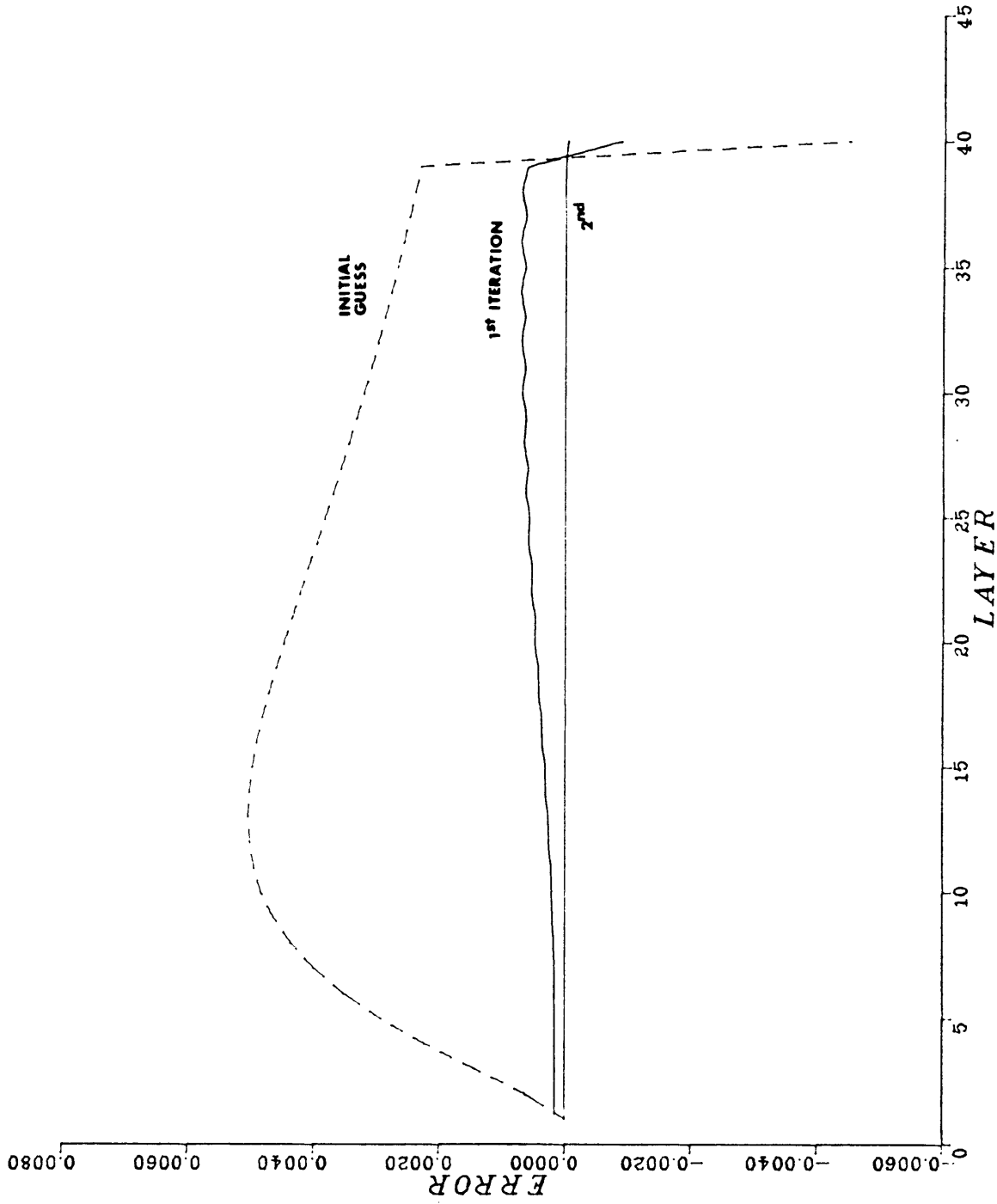


Figure 2-6 The observed minus the calculated trace for successive iterations with model 2 and zero noise.

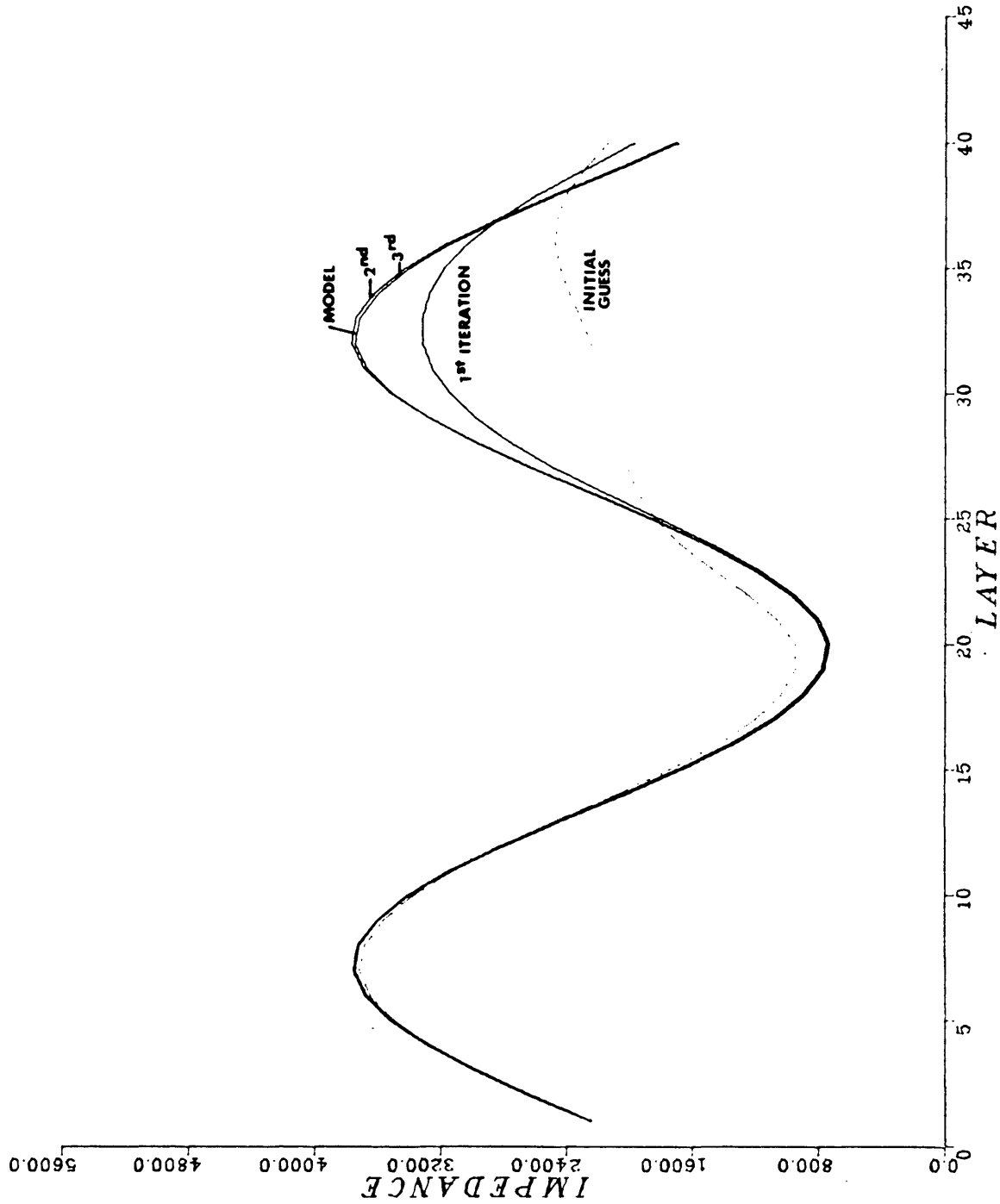


Figure 2-7 Inversion results for model 3 with zero noise. The impedance varies sinusoidally for this model.

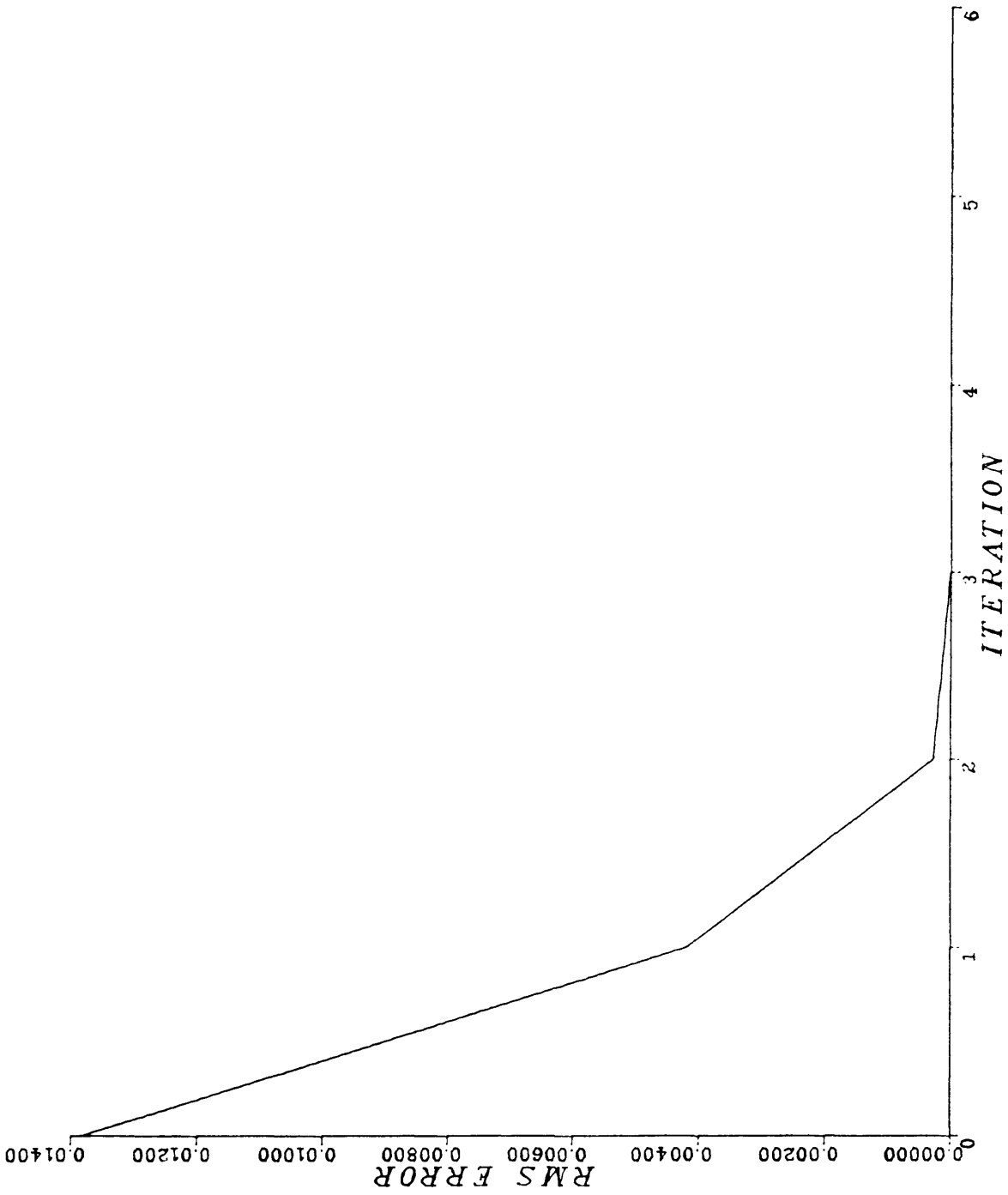


Figure 2-8 Model 3 RMS error for zero noise case.

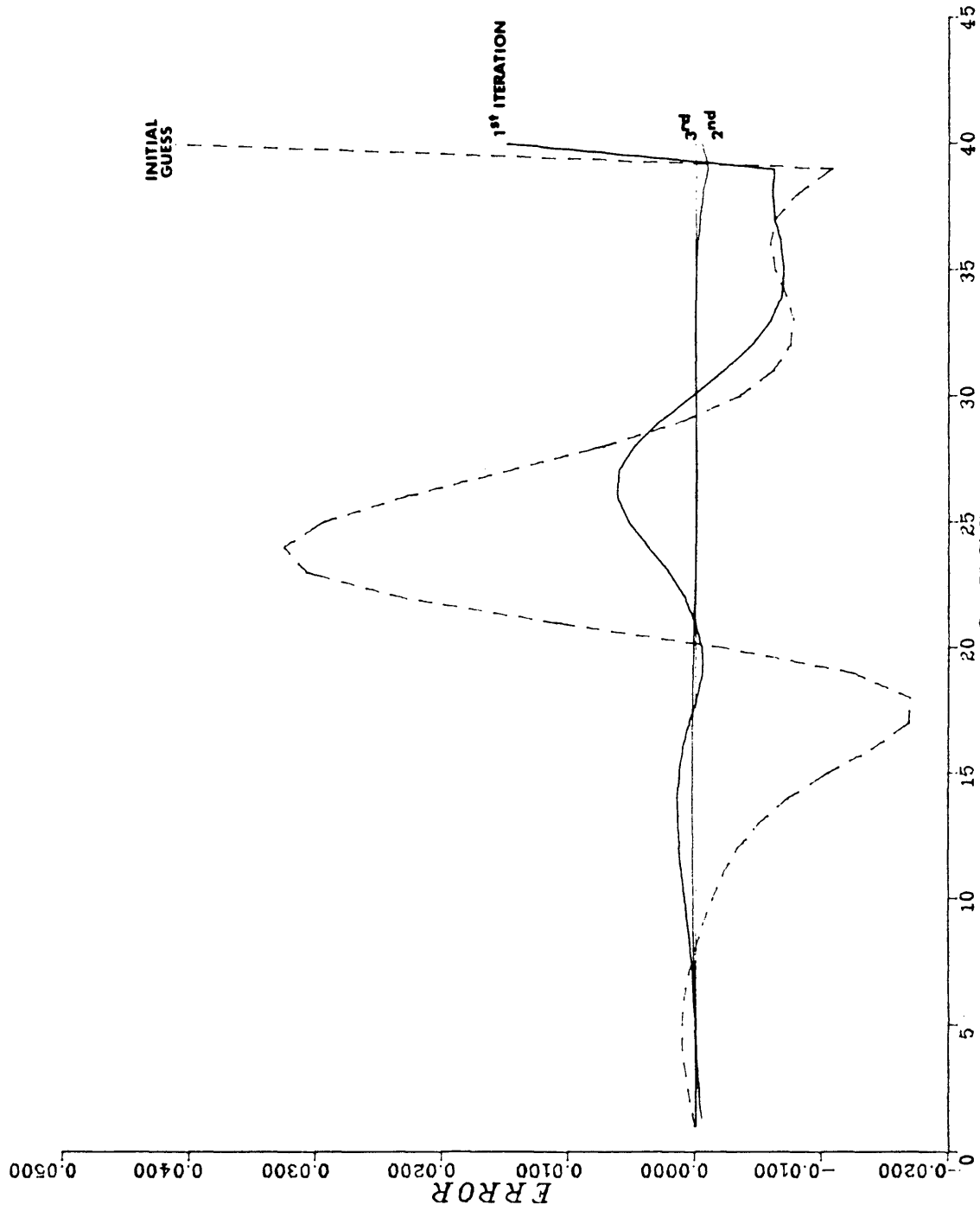


Figure 2-9 The observed minus calculated trace for successive iterations with model 3 and zero noise.

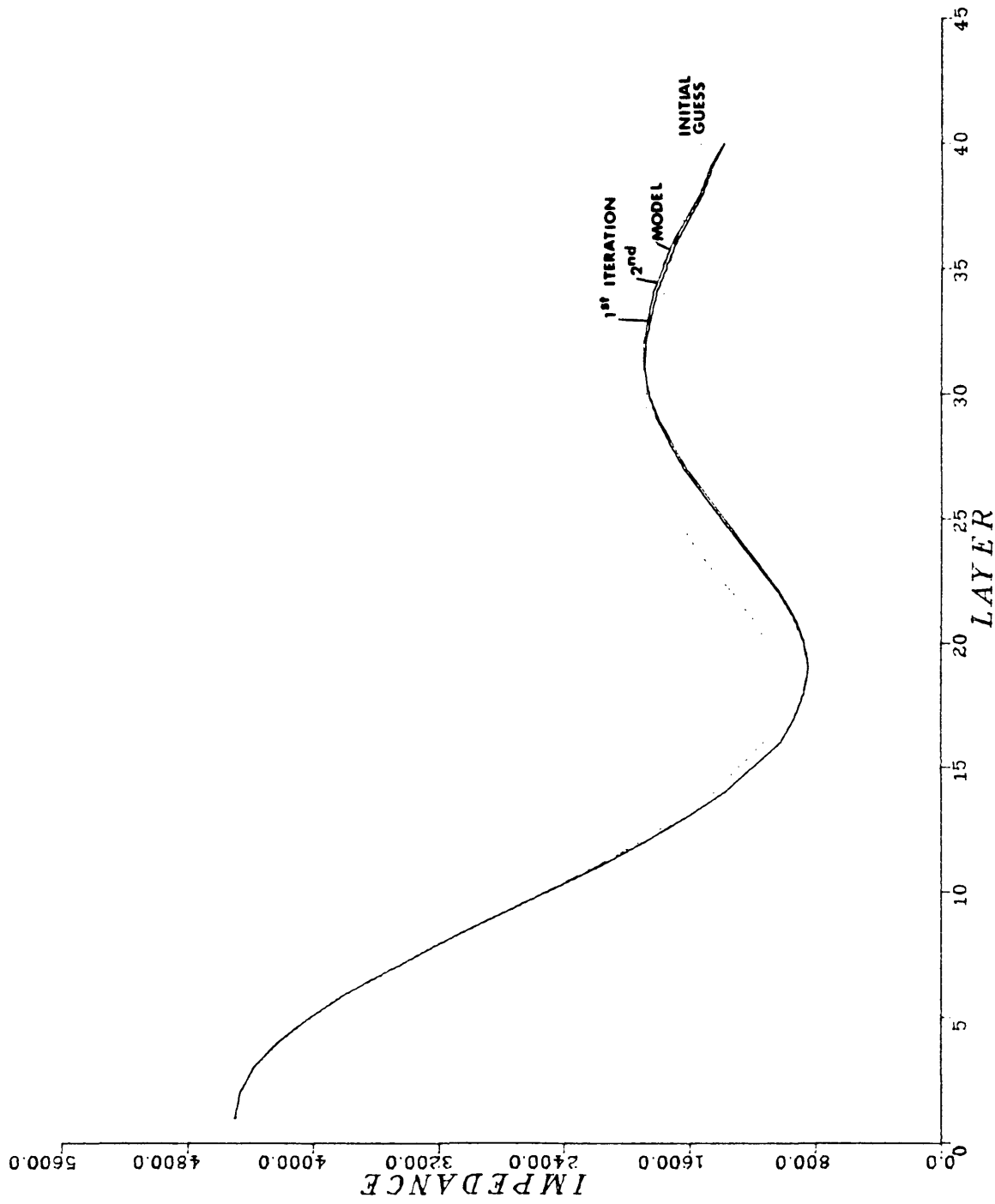


Figure 2-10 Inversion results for model 4 with zero noise. Impedance varies as a sinc function for this model.

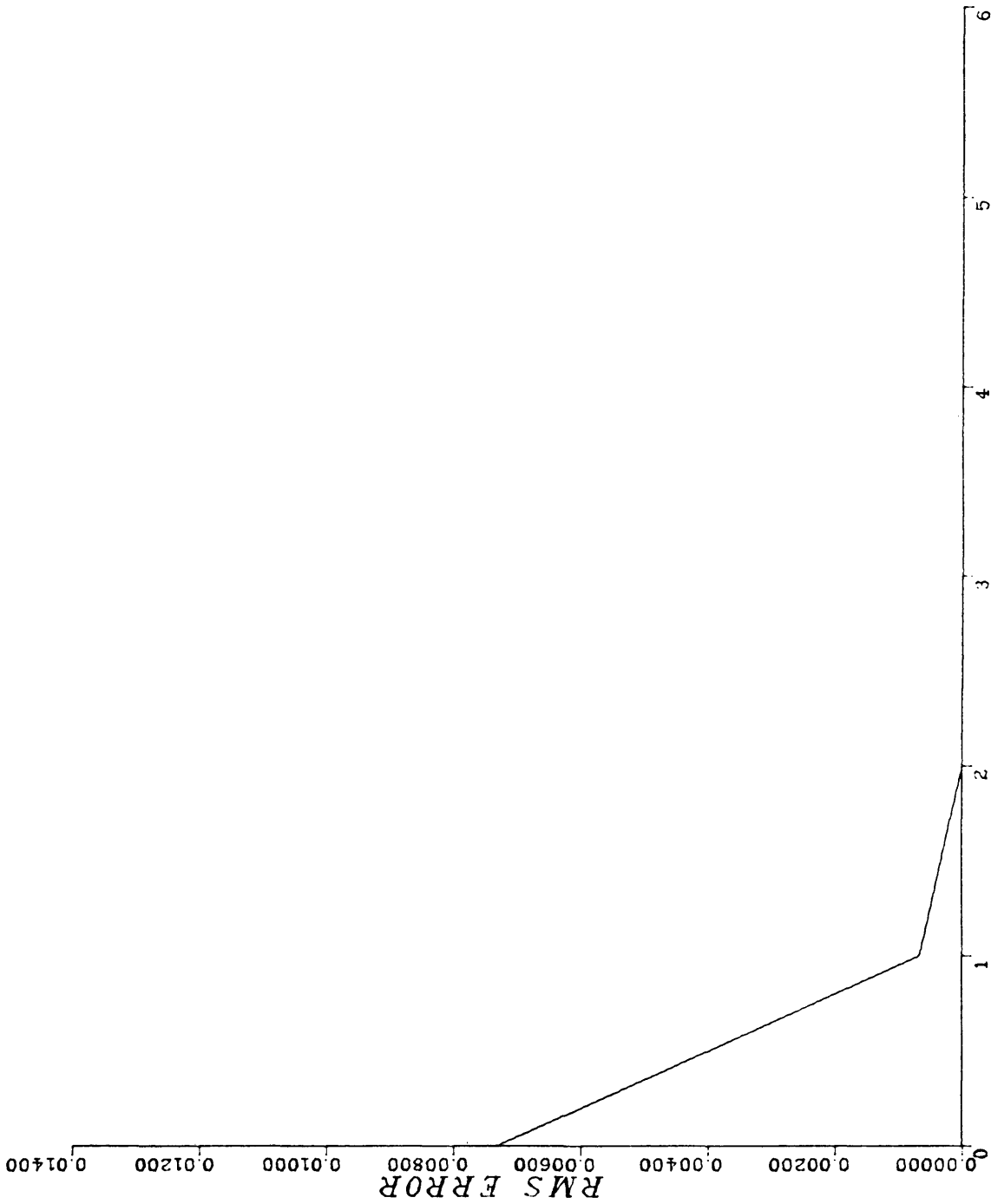


Figure 2-11 Model 4 RMS error for zero noise.

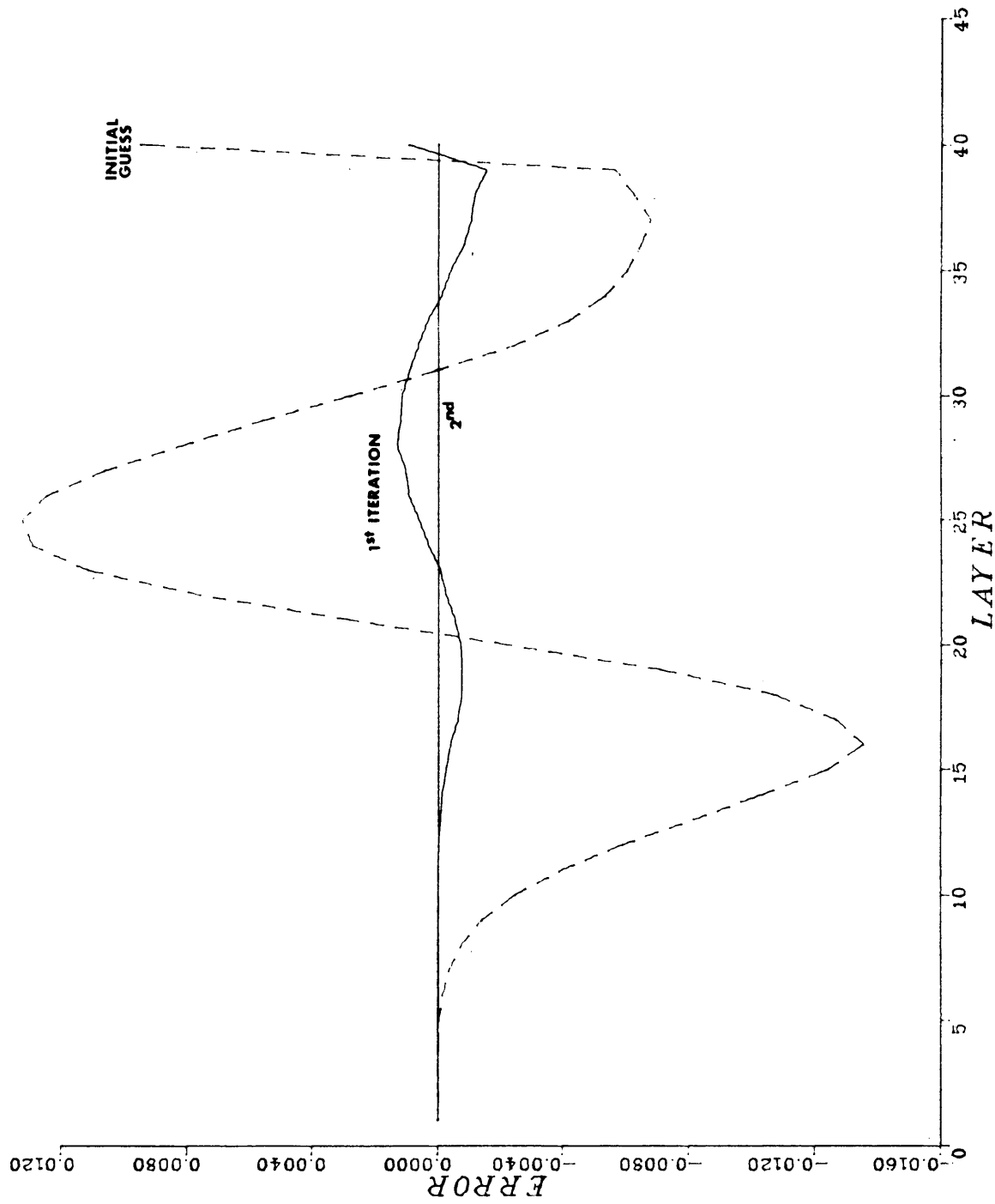


Figure 2-12 The observed minus calculated trace for successive iterations with model 4 and zero noise.

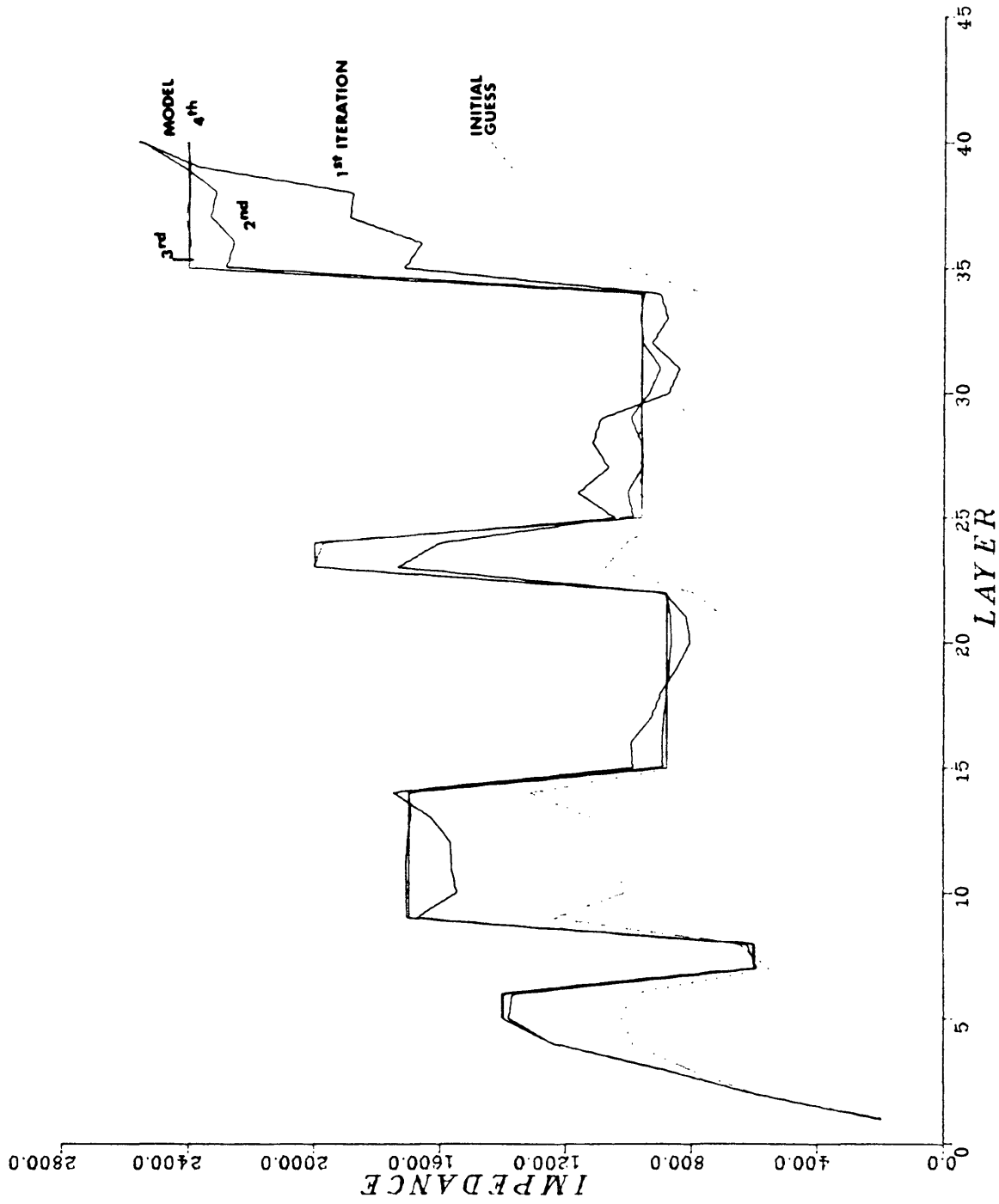


Figure 2-13 Inversion results for model 5 with zero noise. Impedance varies aperiodically in steps.

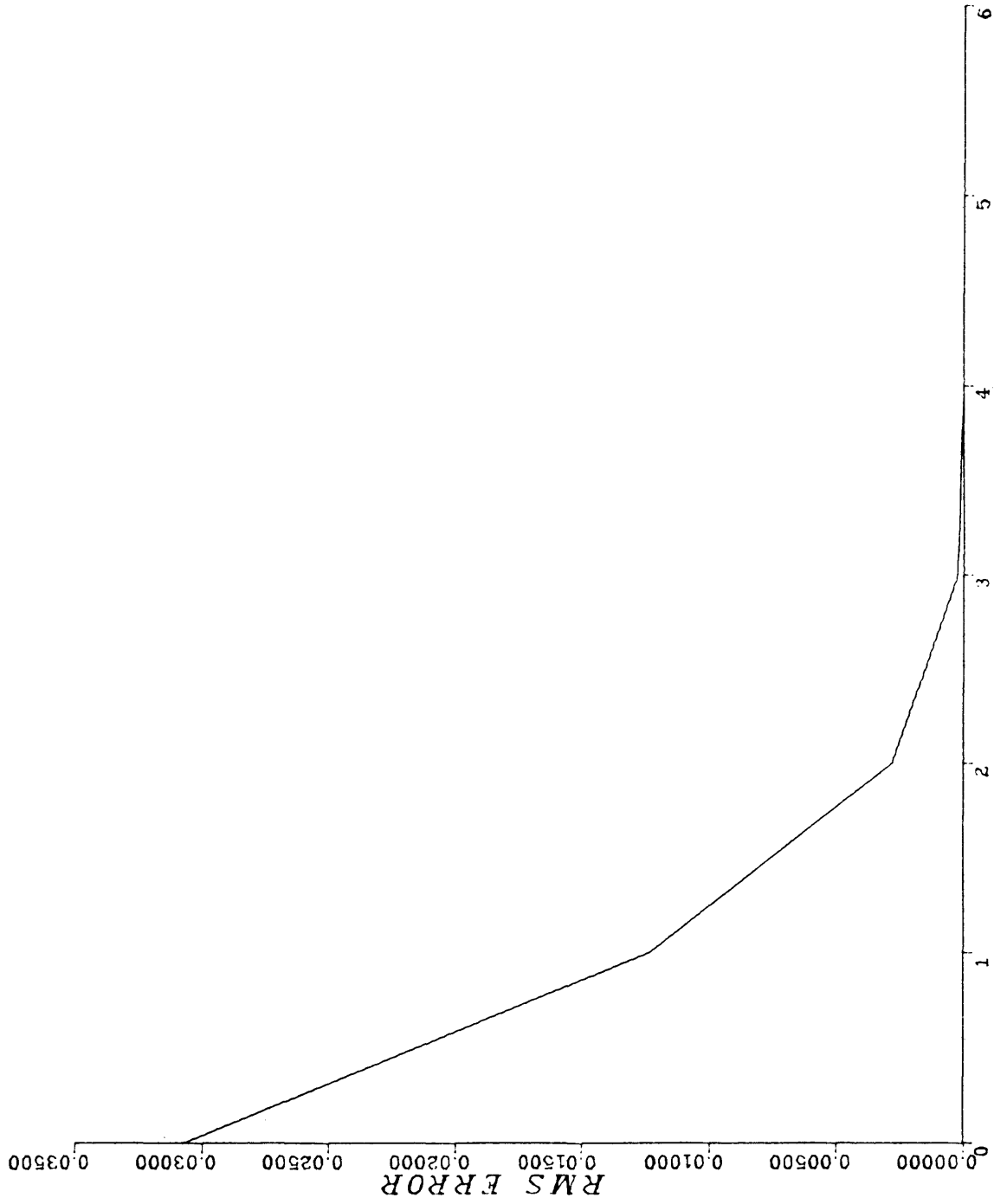


Figure 2-14 Model 5 RMS error for zero noise.

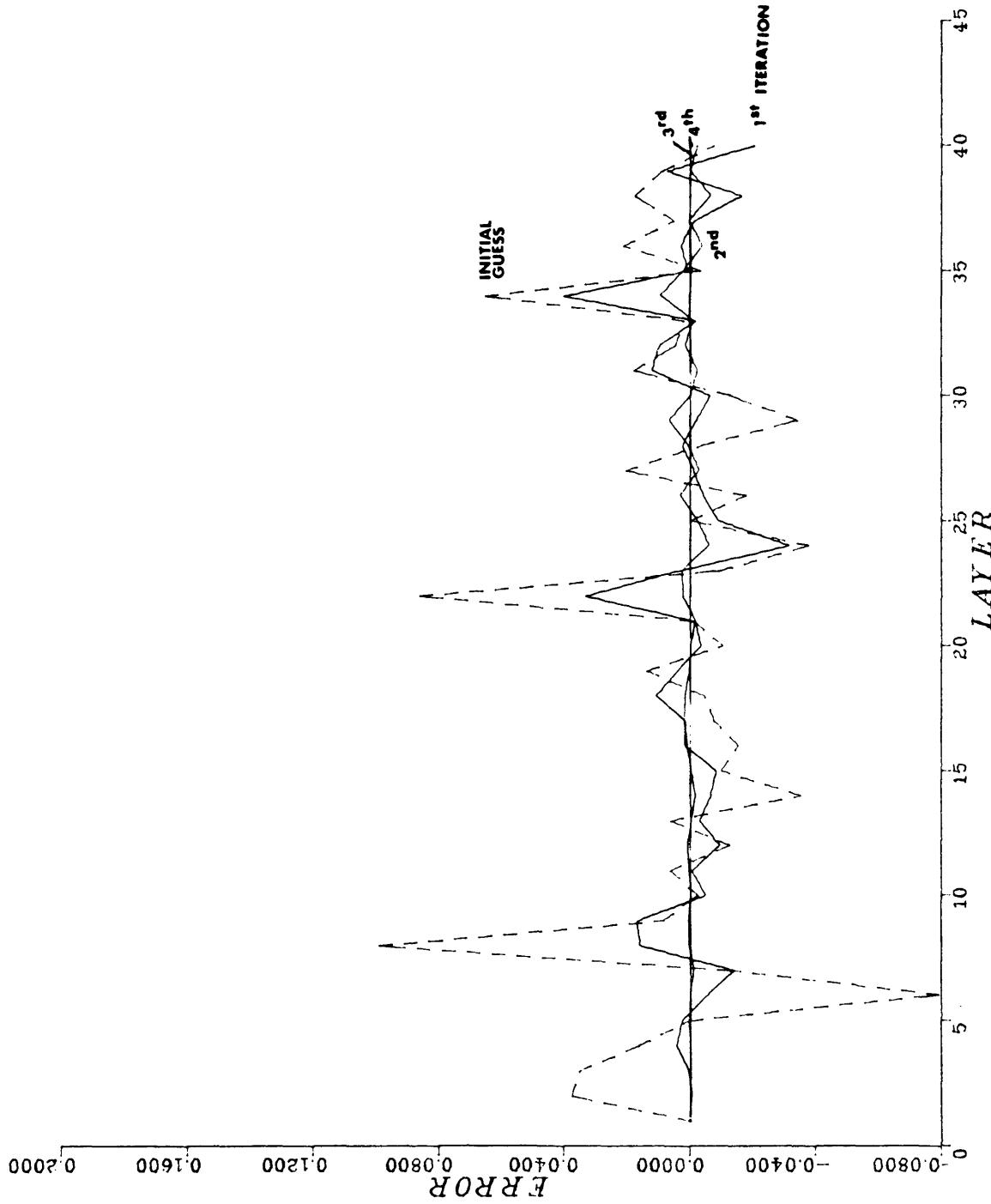


Figure 2-15 The observed minus calculated trace for successive iterations with model 5 and zero noise.

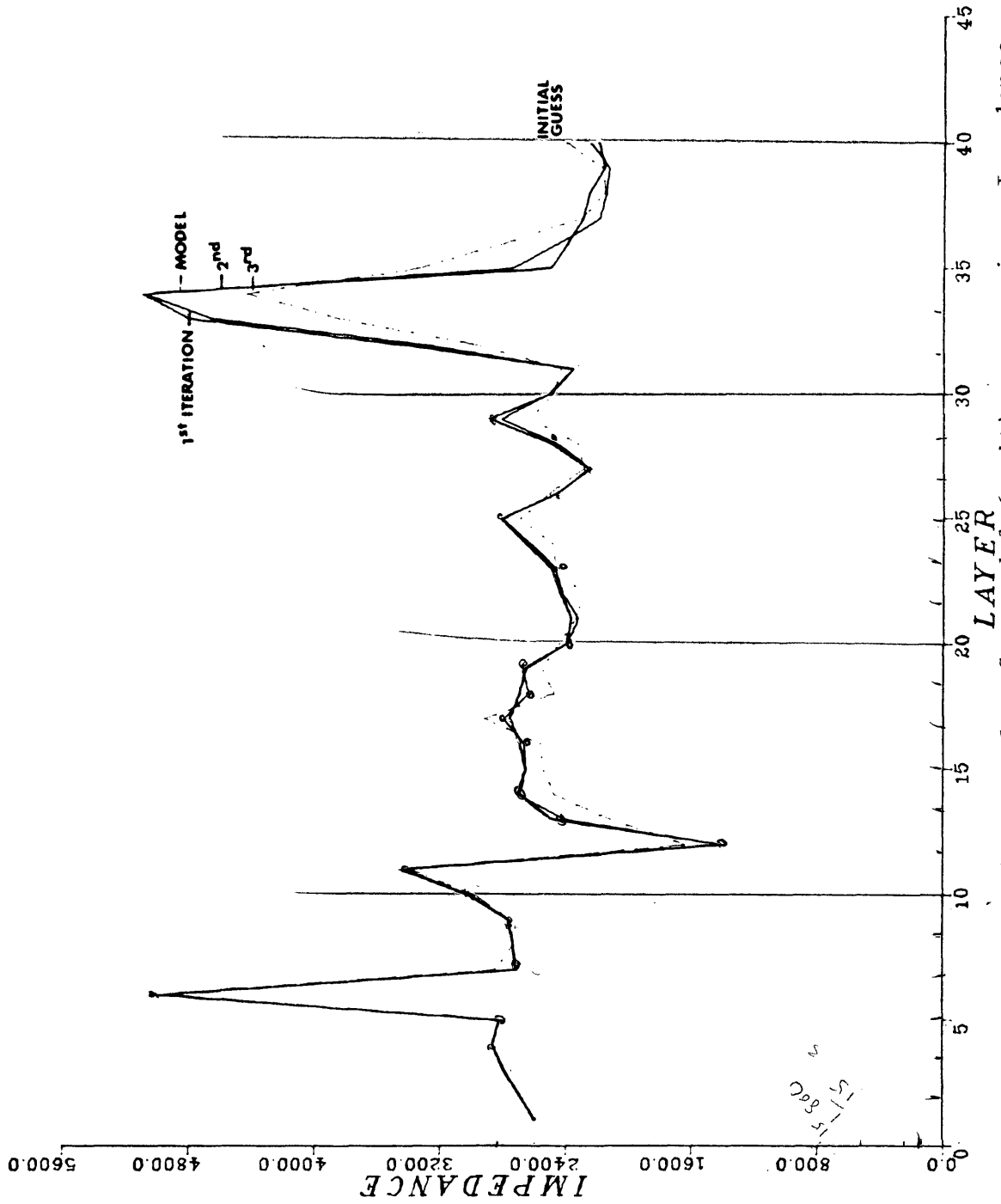


Figure 2-16 Inversion results for model 6 with zero noise. Impedance varies randomly.

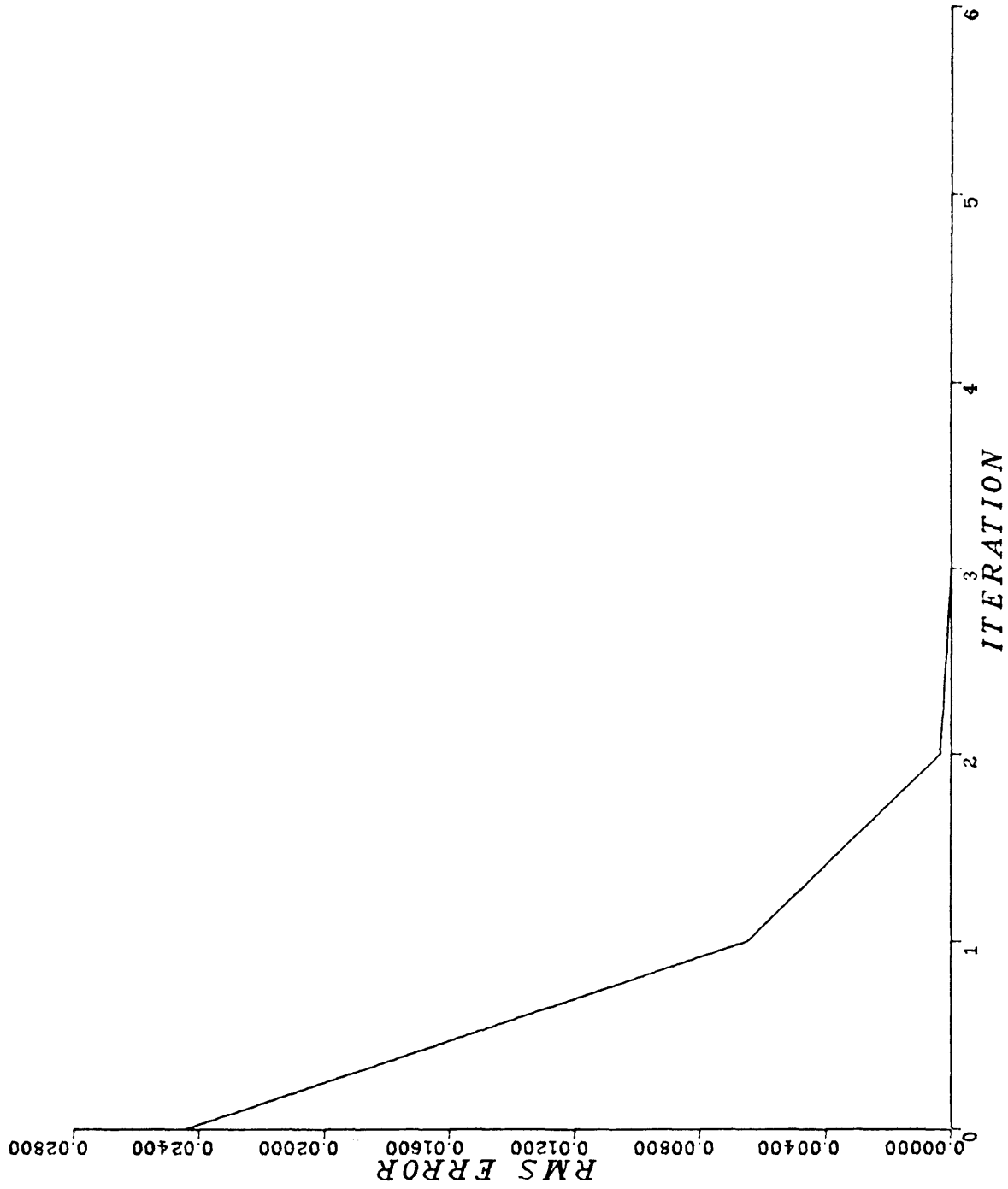


Figure 2-17 Model 6 RMS error for zero noise.

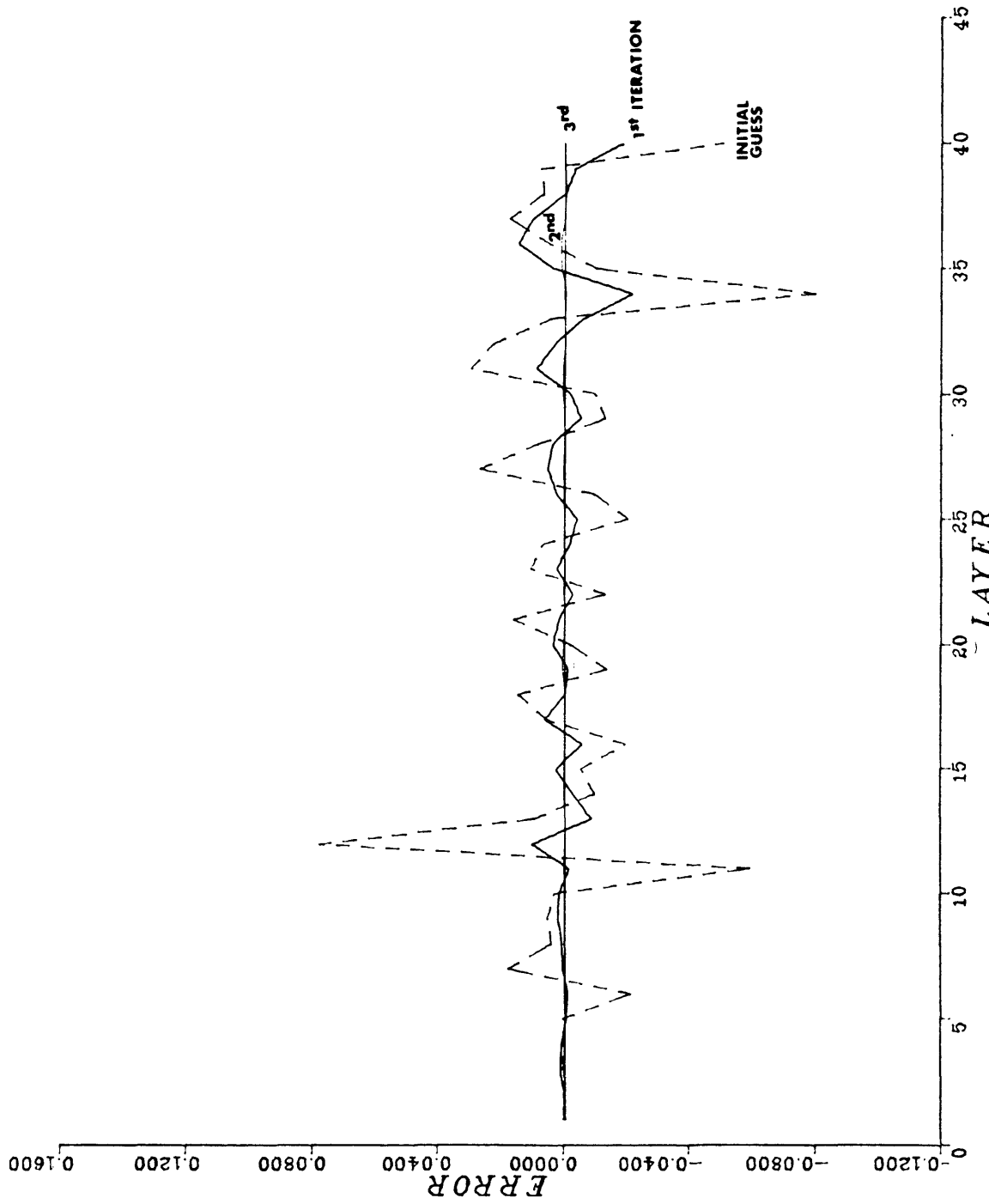


Figure 2-18 The observed minus calculated trace for successive iterations with model 6 and zero noise.

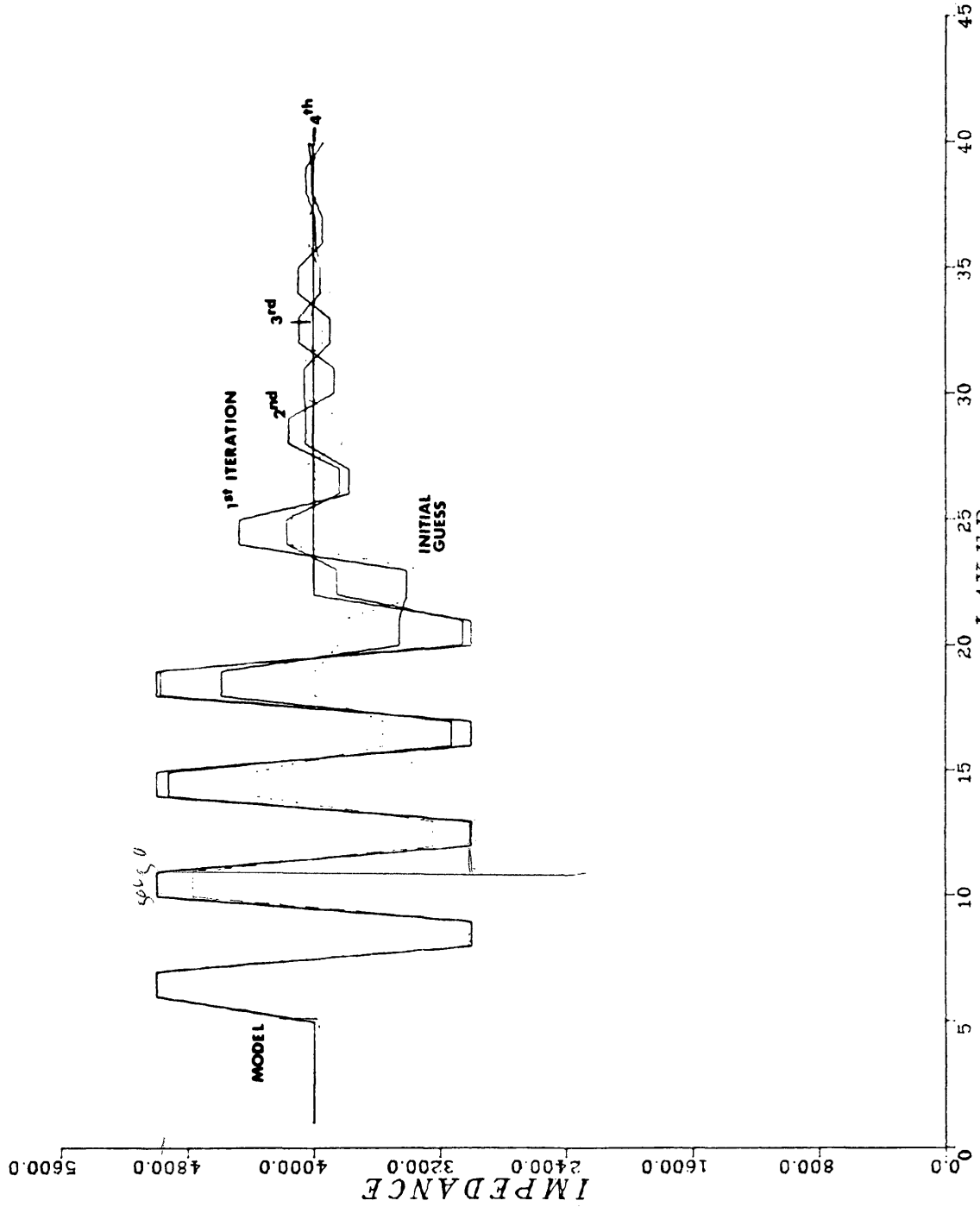


Figure 2-19 Inversion results for model 7 with zero noise. Impedance varies periodically.

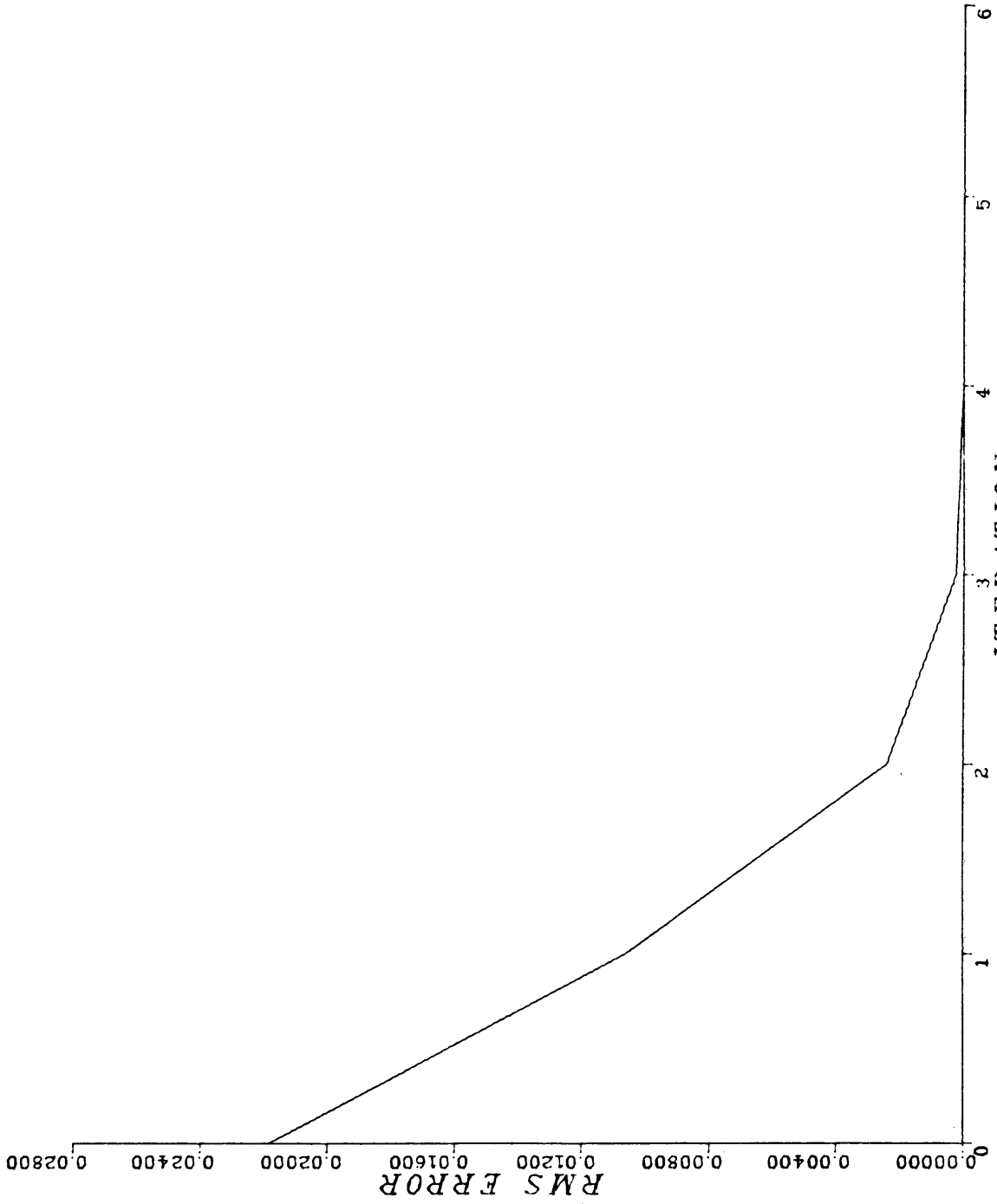


Figure 2-20 Model 7 RMS error for zero noise.

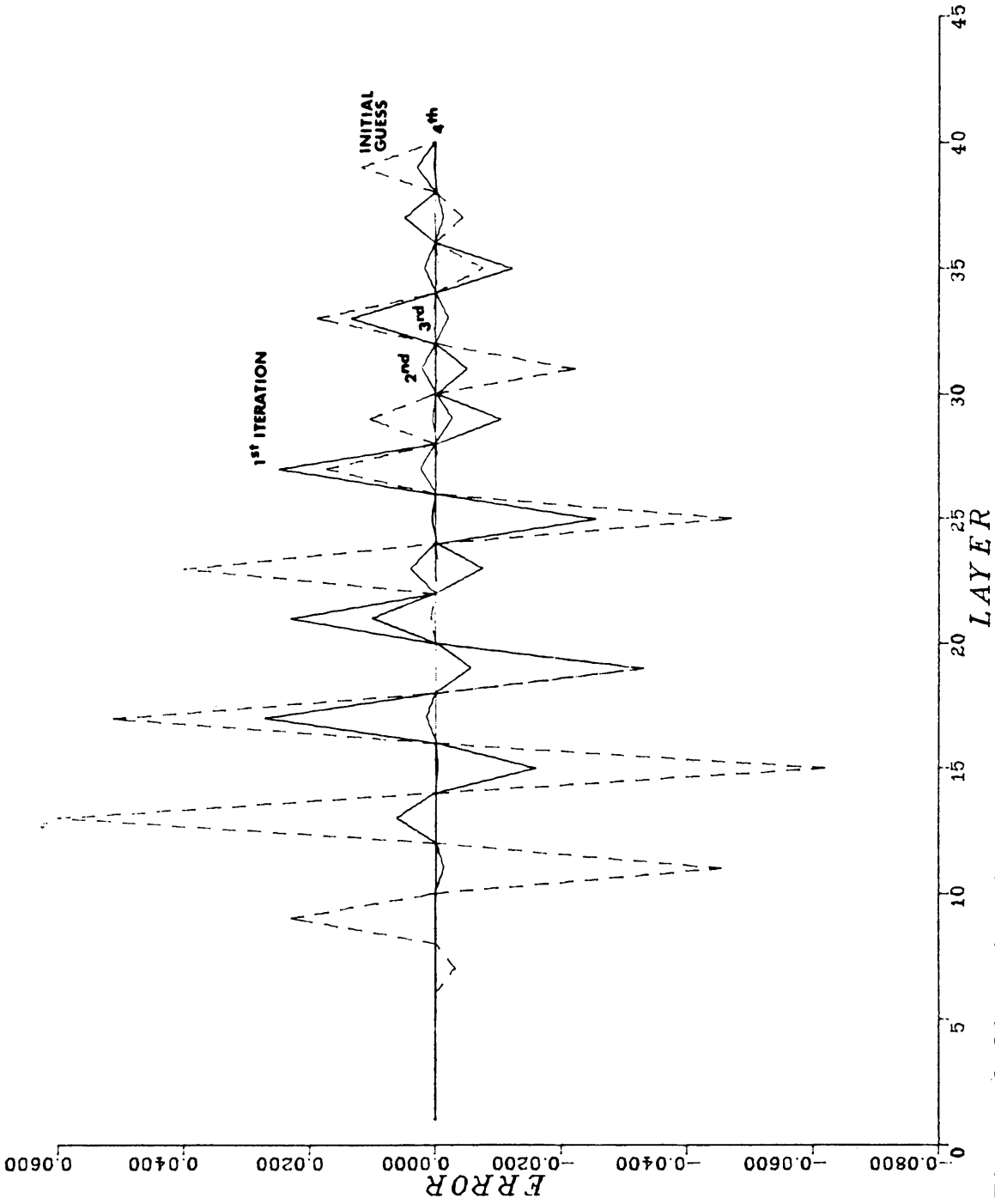


Figure 2-21 The observed minus calculated trace for successive iterations with model 7 and zero noise.

Models 3 and 4 represent periodic impedance distributions, but the rate of convergence is relatively unaffected. Convergence is reached in 2-3 iterations, and the fit to the model parameters is good.

Models 5 and 6 represent aperiodic models. The rate of convergence is slightly slower but not significantly so and the fit remains excellent.

Finally model 7 represents a periodic variation in impedances for about the first 20 layers, followed by 20 layers of constant impedance. At the last iteration, the fit is almost perfect with the multiple effects on the last 20 layers almost completely removed. As was noted in the previous section, the errors should be reducible to zero in the noiseless case and in fact, the results seem to indicate this is true. What about the case of additive noise, however? Does this method converge, and is it stable?

b) Random noise

Two separate paths were taken to investigate the effects of additive random noise. The first was to generate a model with a constant impedance for each layer and add noise to its response. The second was to add noise to our original models and

observe how the inverted impedances were perturbed.

As the first model is simpler, it probably best serves to illustrate the effects of noise so we will begin with it.

Table 1 lists the six different sets of noise added to our observed signal. Without the noise, the RMS data level is zero as would be expected for a constant impedance. The point for choosing this particular model now becomes clear as we see that in this case the noise is the only non-zero component of the signal. Figures 2-22 through 2-33 show the effects of the differing levels of noise. The two plots for each noise level show respectively the model and program generated impedances and the observed and calculated traces for the last iteration.

The things to notice are that the seislog initial guess is for the most part very close to the final iteration results. Secondly, except for the highest noise level case, the observed and calculated traces are very close. This means this inversion technique can find

TABLE 1

<u>Noise Value Range</u>	<u>RMS Noise Level</u>	<u>RMS Data Level</u>	<u>OBS - CALC RMS Error</u>
-.0125→.0125	.006216	0.00	.001912
-.025→.025	.012985	0.00	.002242
-.05 →.05	.027841	0.00	.003940
-.1 →.1	.056962	0.00	.000713
-.2 →.2	.110179	0.00	.004147
-.4 →.4	.237170	0.00	.131865

an impedance distribution which would give identical results to the effect of the additive noise. When in the first five cases the noise level doesn't generate an impedance distribution with high reflection coefficients, the multiples are not strong enough to give the inversion technique problems. In the last case, however, we notice that very early in the impedance distribution there is a large contrast. This in turn should generate strong multiples in the trace which are not there. This causes the routine to do a less efficient job in matching the calculated to the observed trace as we might guess, and a larger RMS error results.

For the second approach, two sets of noise corresponding to the two lowest levels in the previous example were added to each of the seven models. Due to the fact the RMS level of the data differed for each model, the resulting signal to noise levels also differed. Table 2 lists the first noise case and Table 3, the second. With both sets of noise, no problem was encountered with stability but as would be expected, the fit to the model parameters was not as good as in the noiseless case.

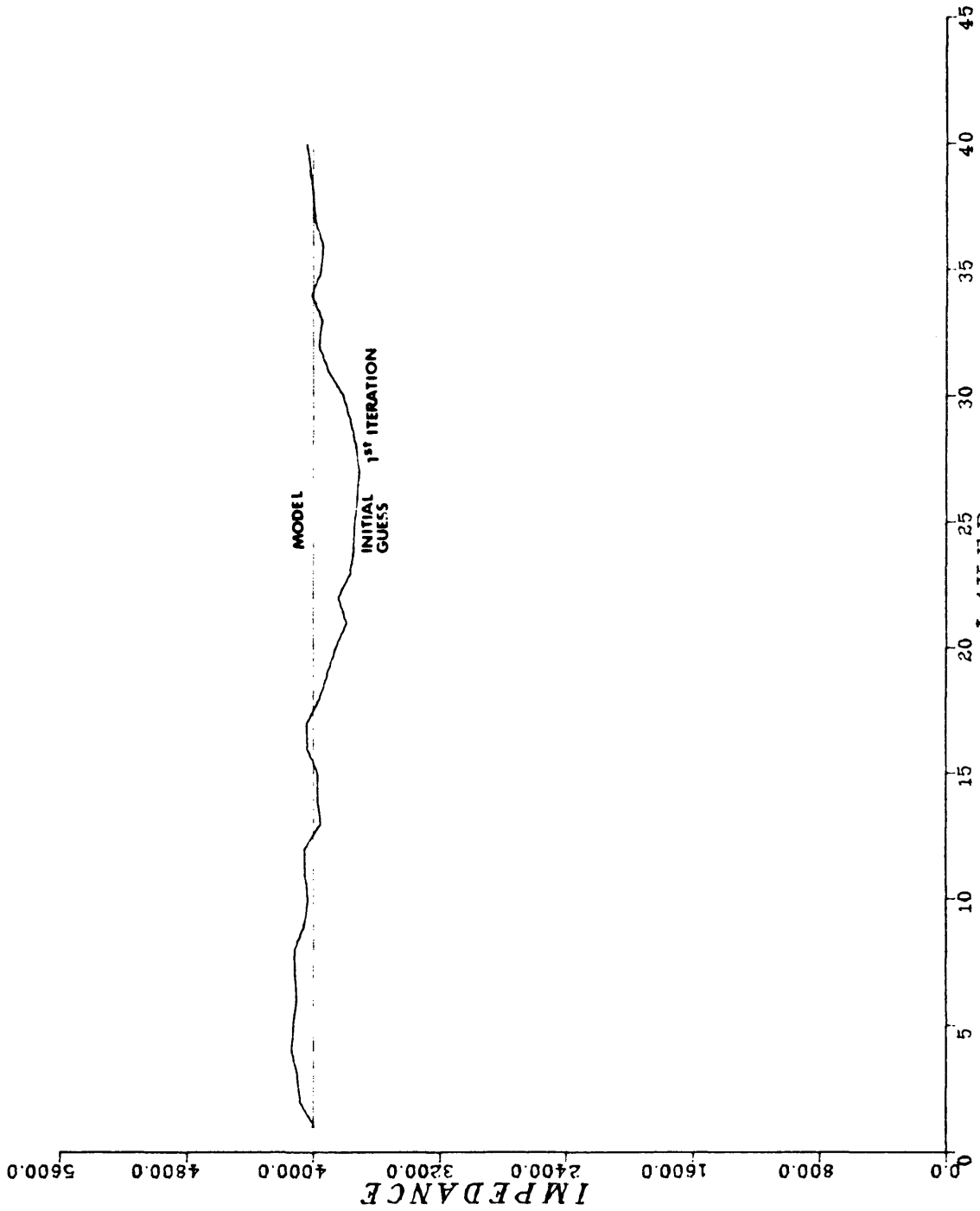


Figure 2-22 Inversion result for constant impedance model and .006216 RMS noise level.

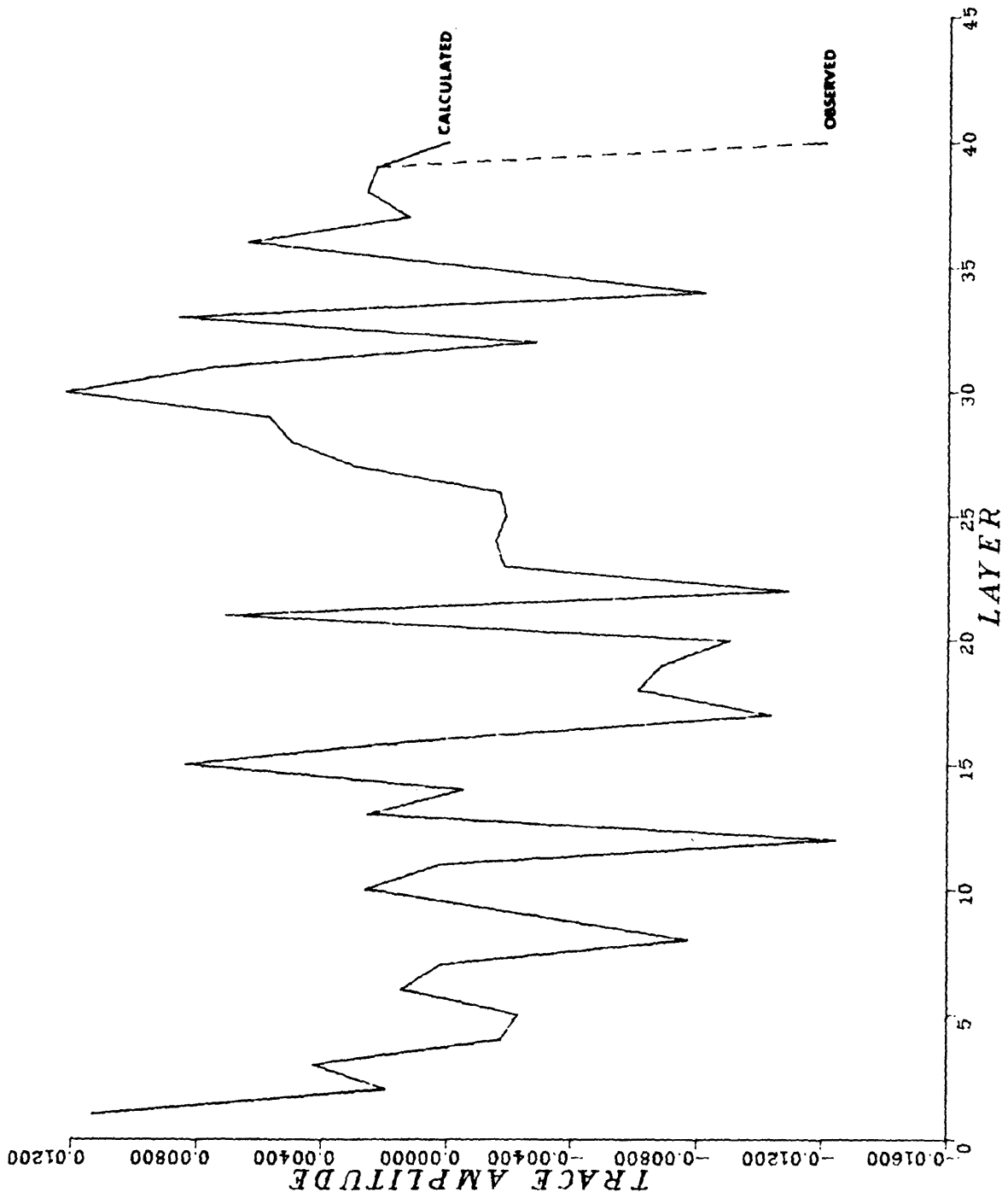


Figure 2-23 Observed and calculated trace of final inversion iteration for constant impedance model and .006216 RMS noise level.

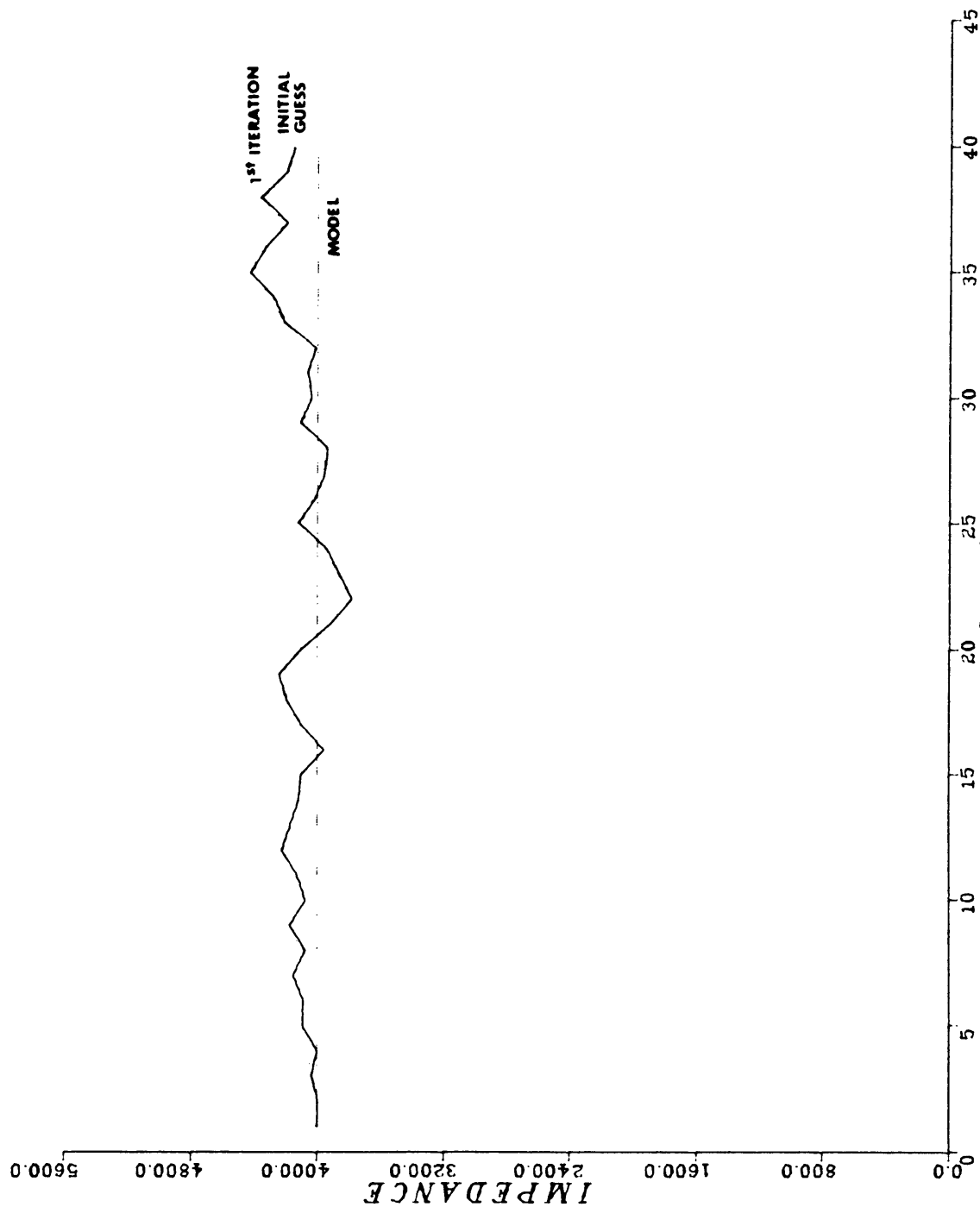


Figure 2-24 Inversion result for constant impedance model and .012985 RMS noise level.

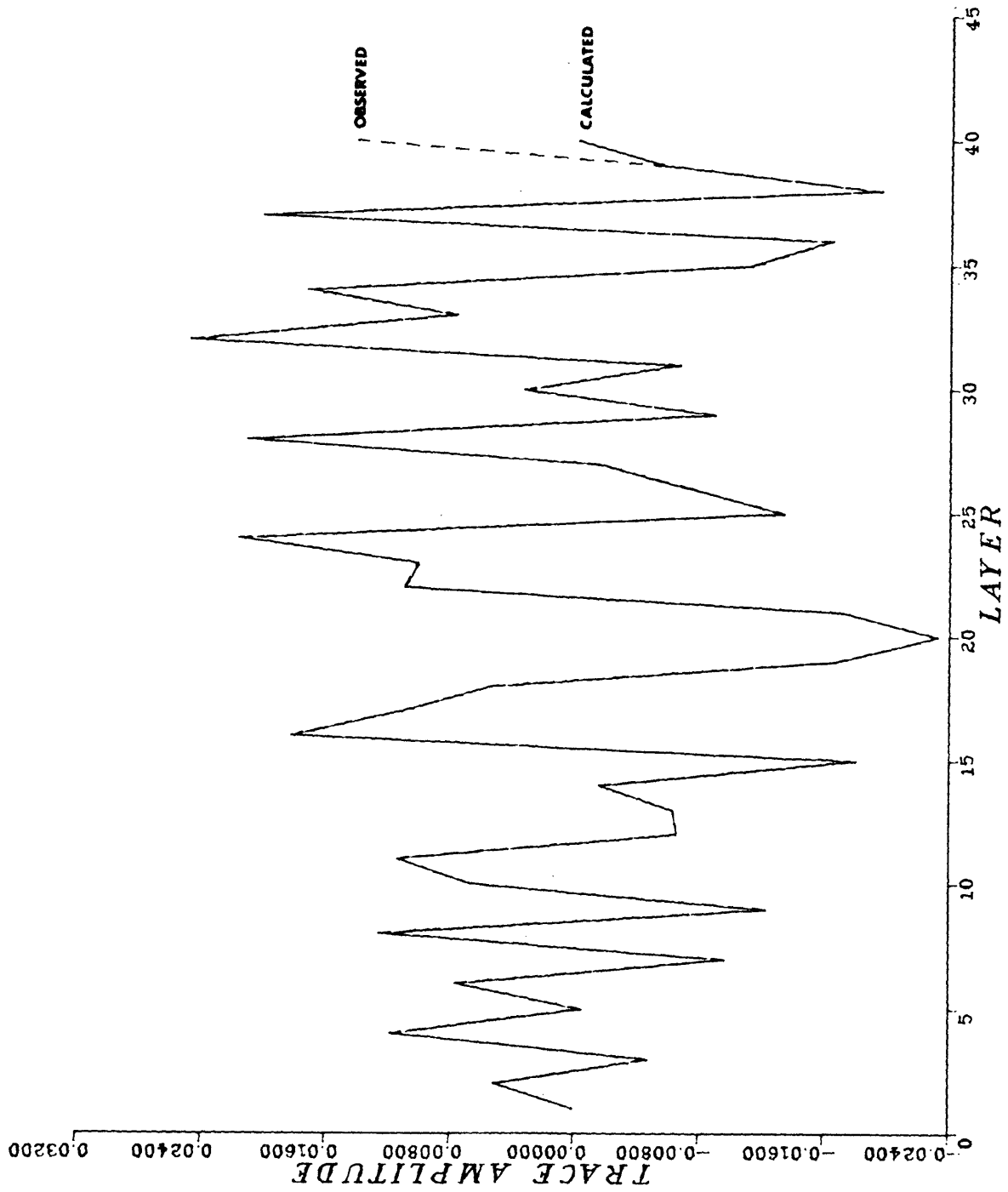


Figure 2-25 Observed and calculated trace of final inversion iteration for constant impedance model and .012985 RMS noise level.

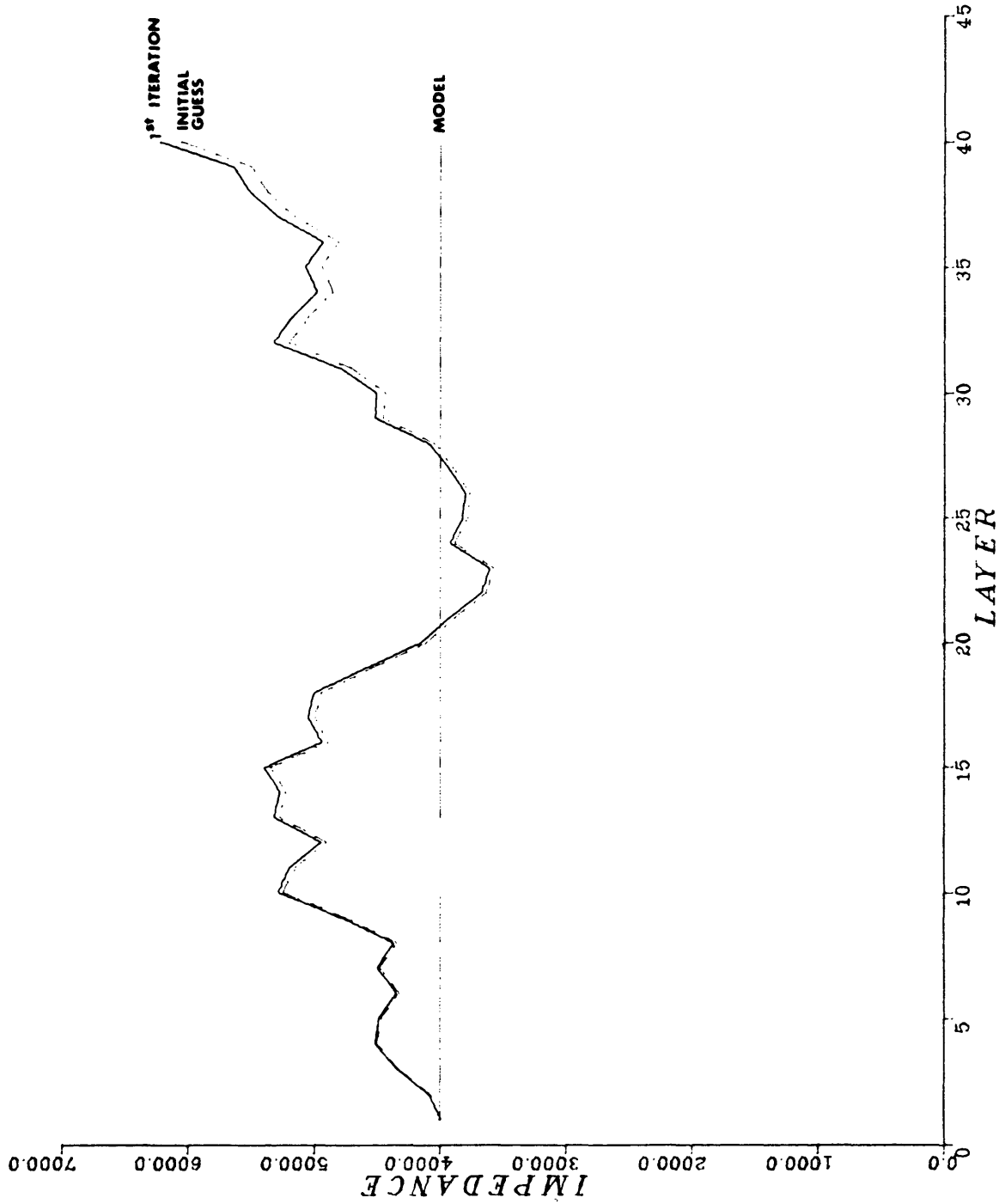


Figure 2-26 Inversion result for constant impedance model and .027841 RMS noise level.

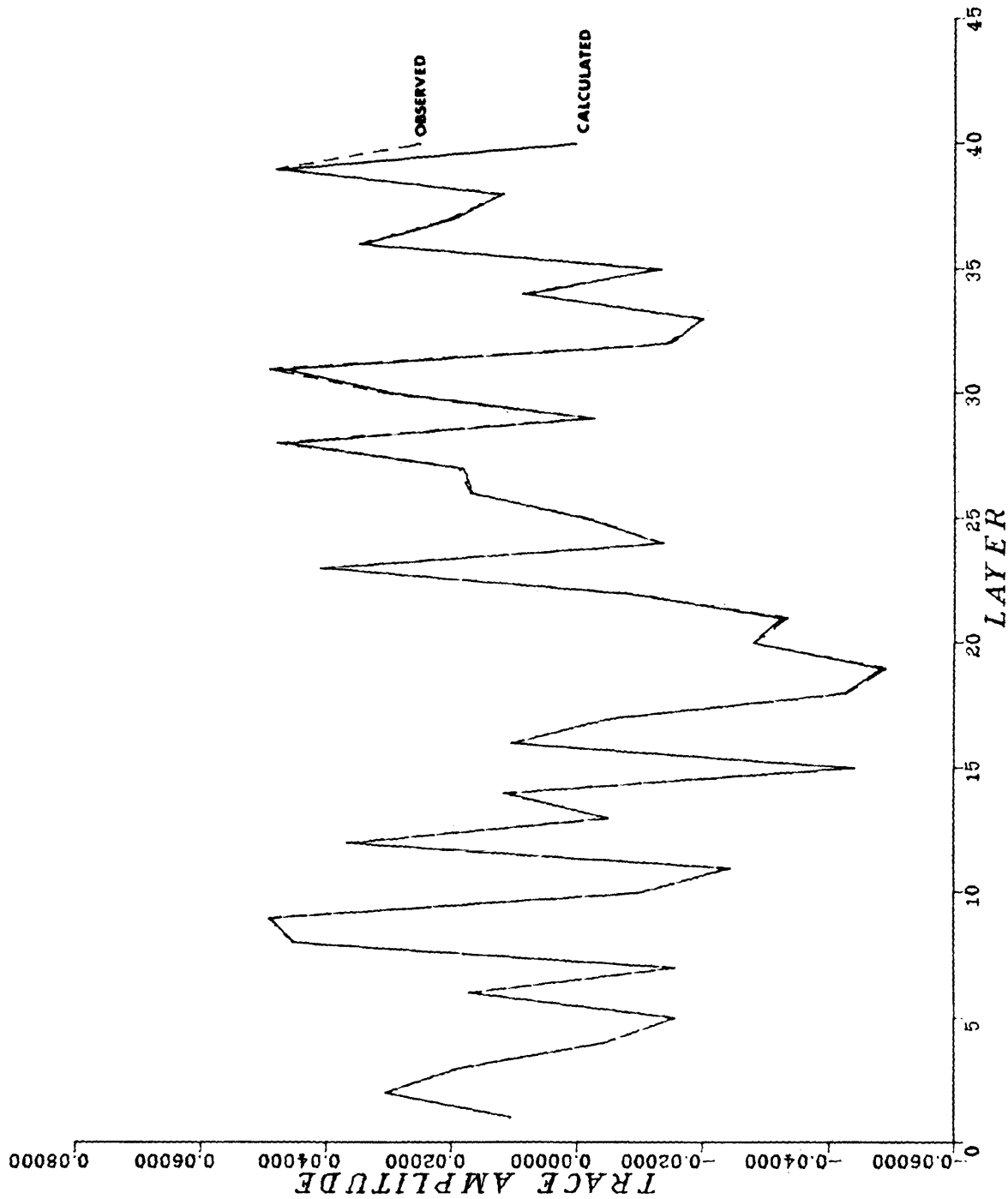


Figure 2-27 Observed and calculated trace of final inversion iteration for constant impedance model and .027841 RMS noise level.

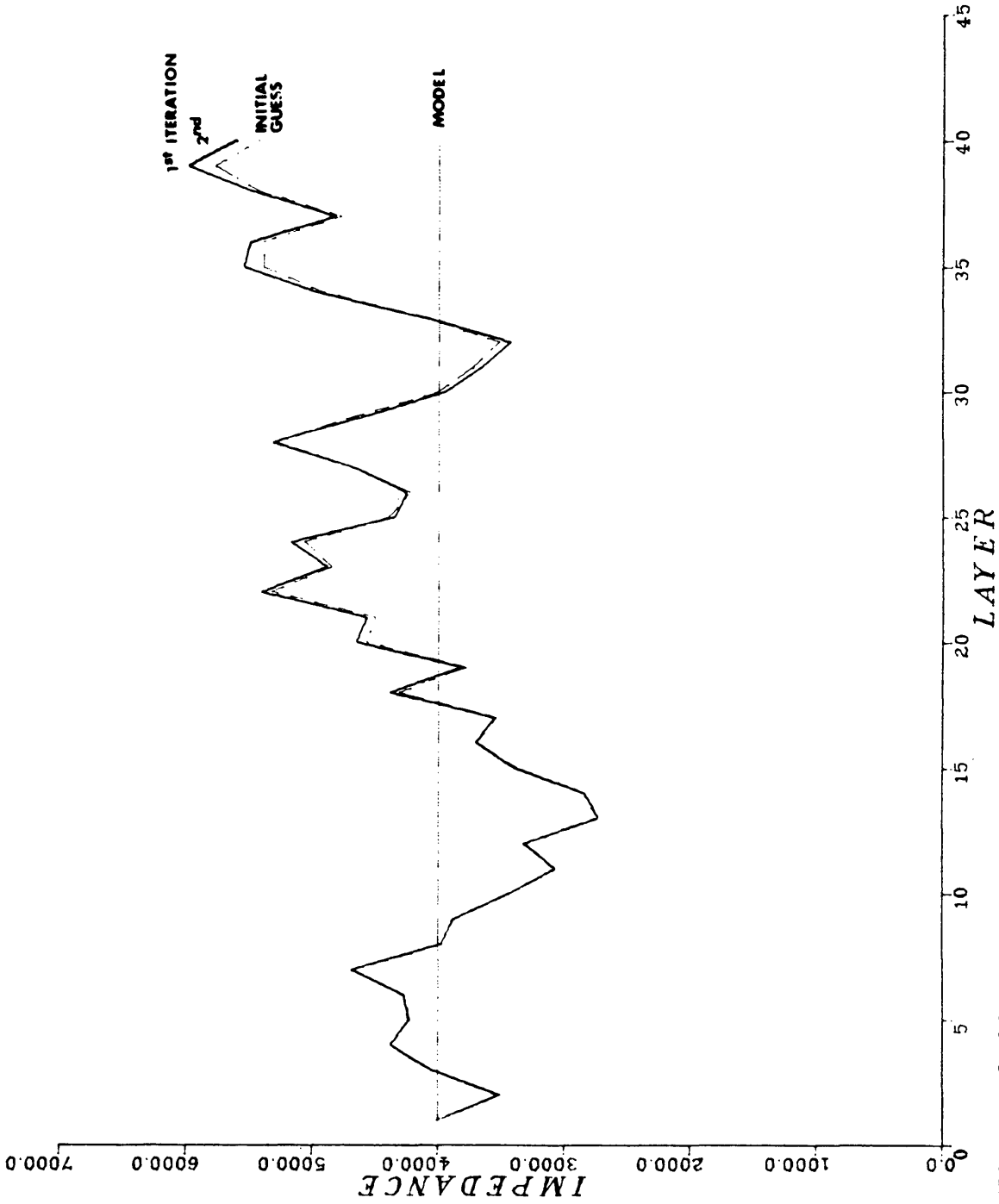


Figure 2-28 Inversion result for constant impedance model and .056962 RMS noise level.

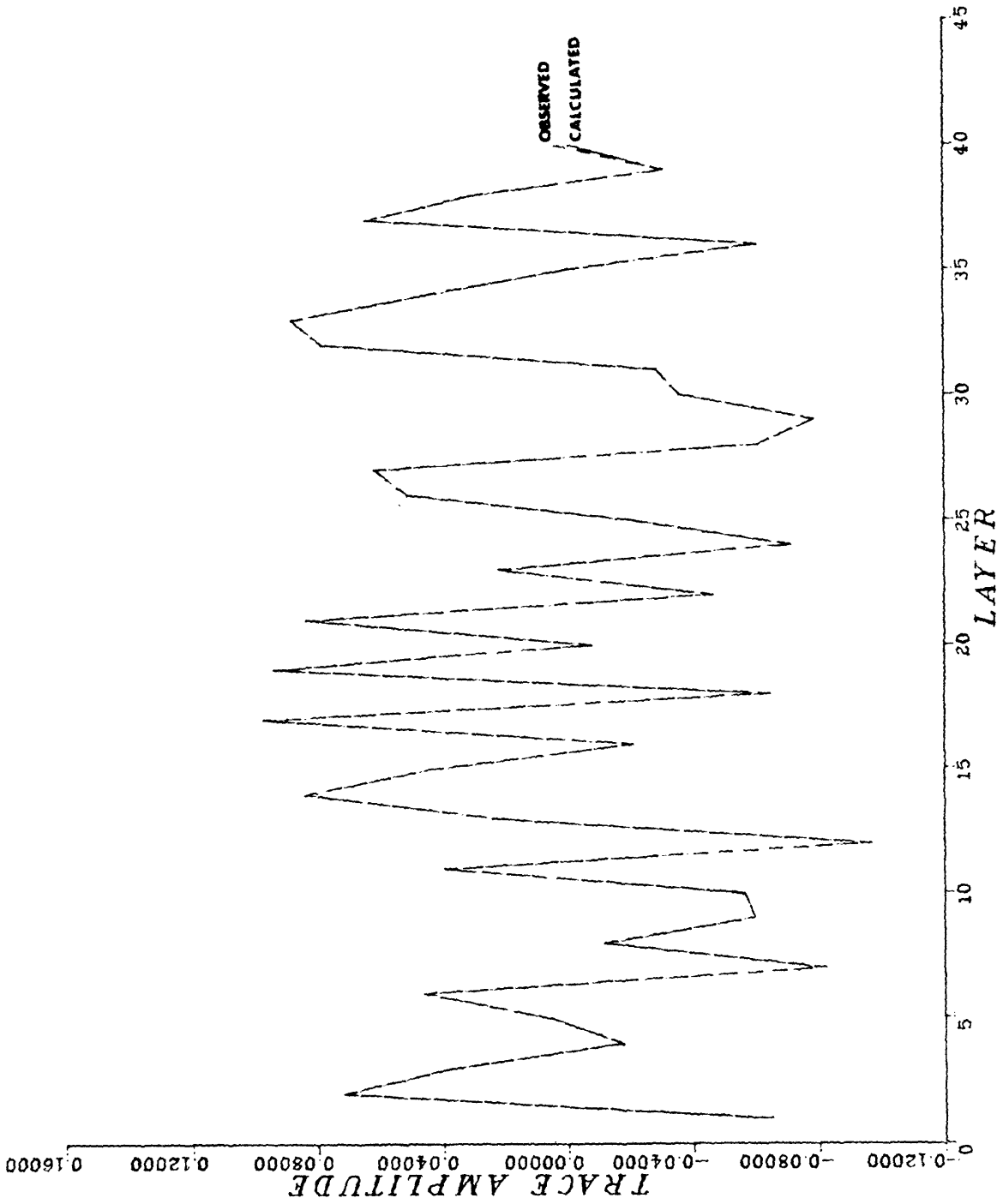


Figure 2-29 Observed and calculated trace of final inversion iteration for constant impedance model and .056962 noise level.

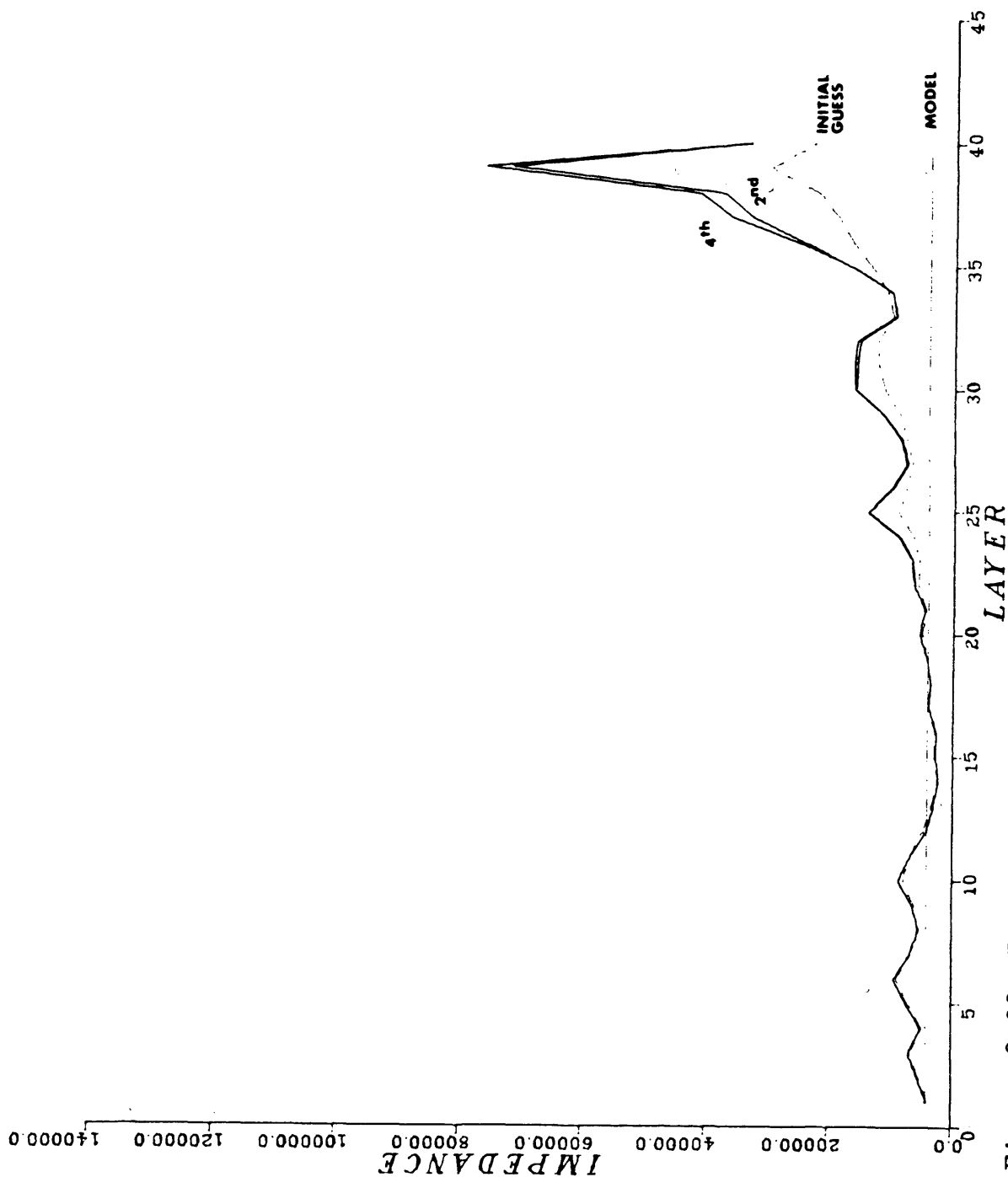


Figure 2-30 Inversion result for constant impedance model and .110179 RMS noise level.

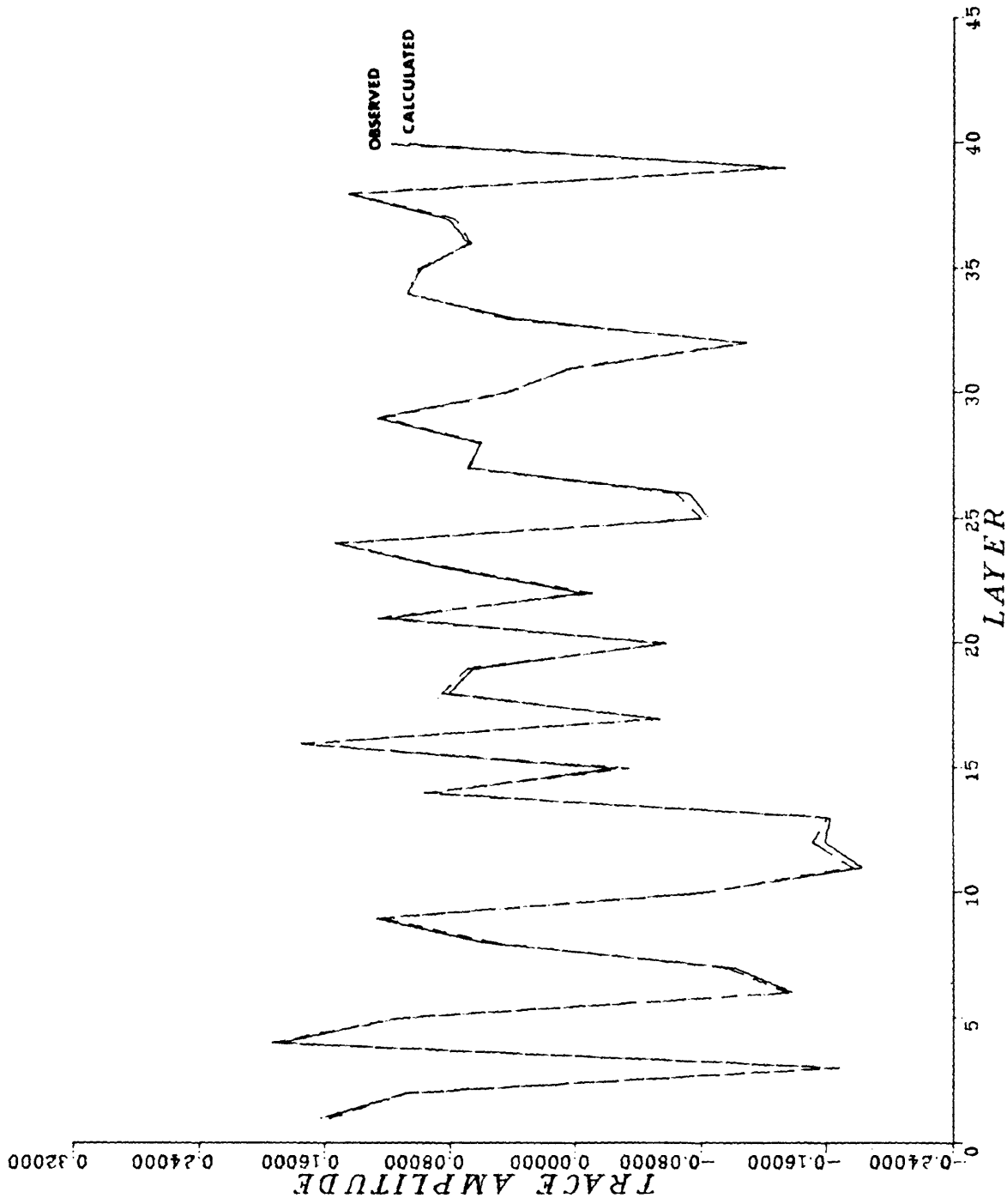


Figure 2-31 Observed and calculated trace of final inversion iteration for constant impedance model and .110179 RMS noise level.

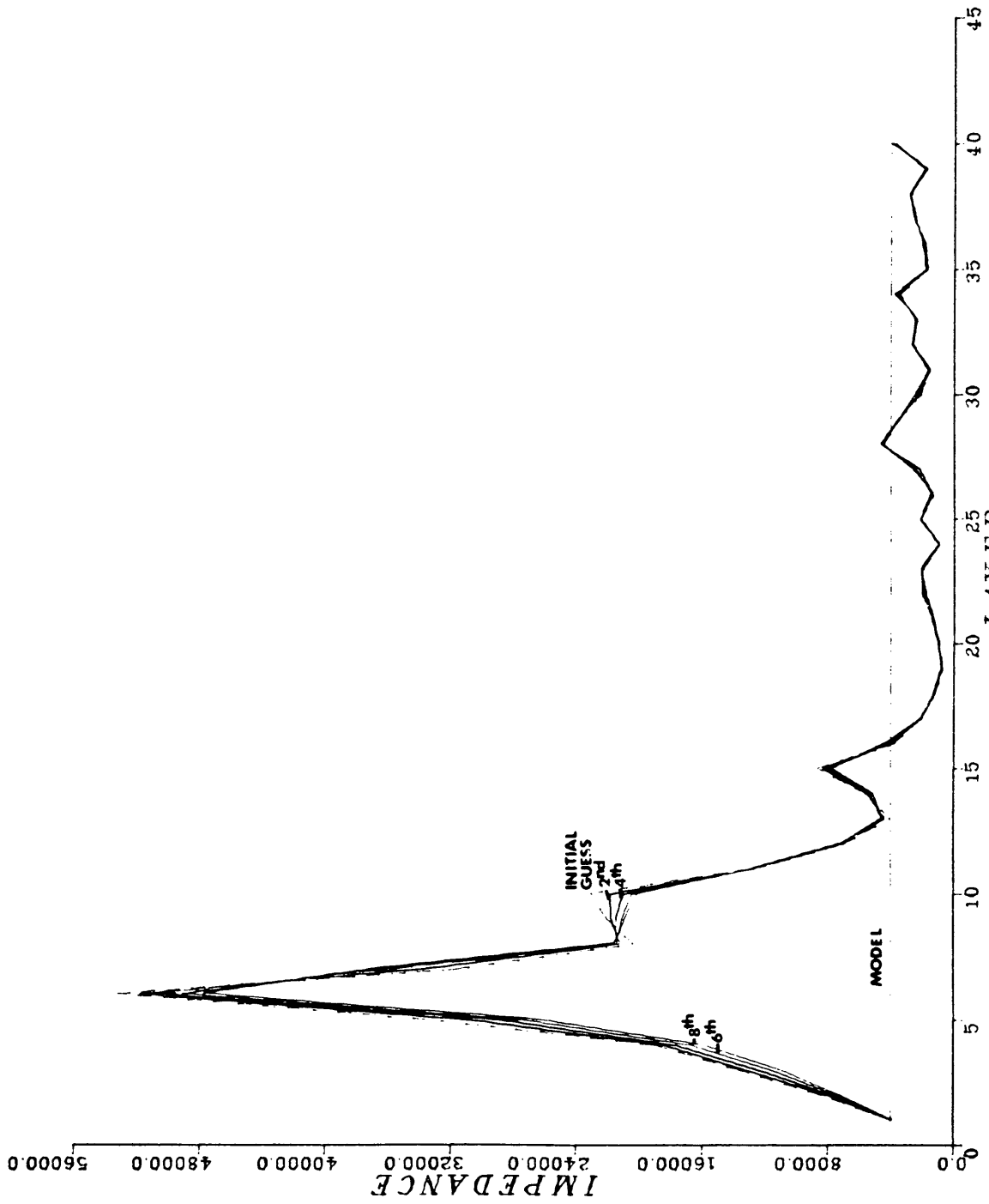


Figure 2-32 Inversion result for constant impedance model and .237170 RMS noise level.

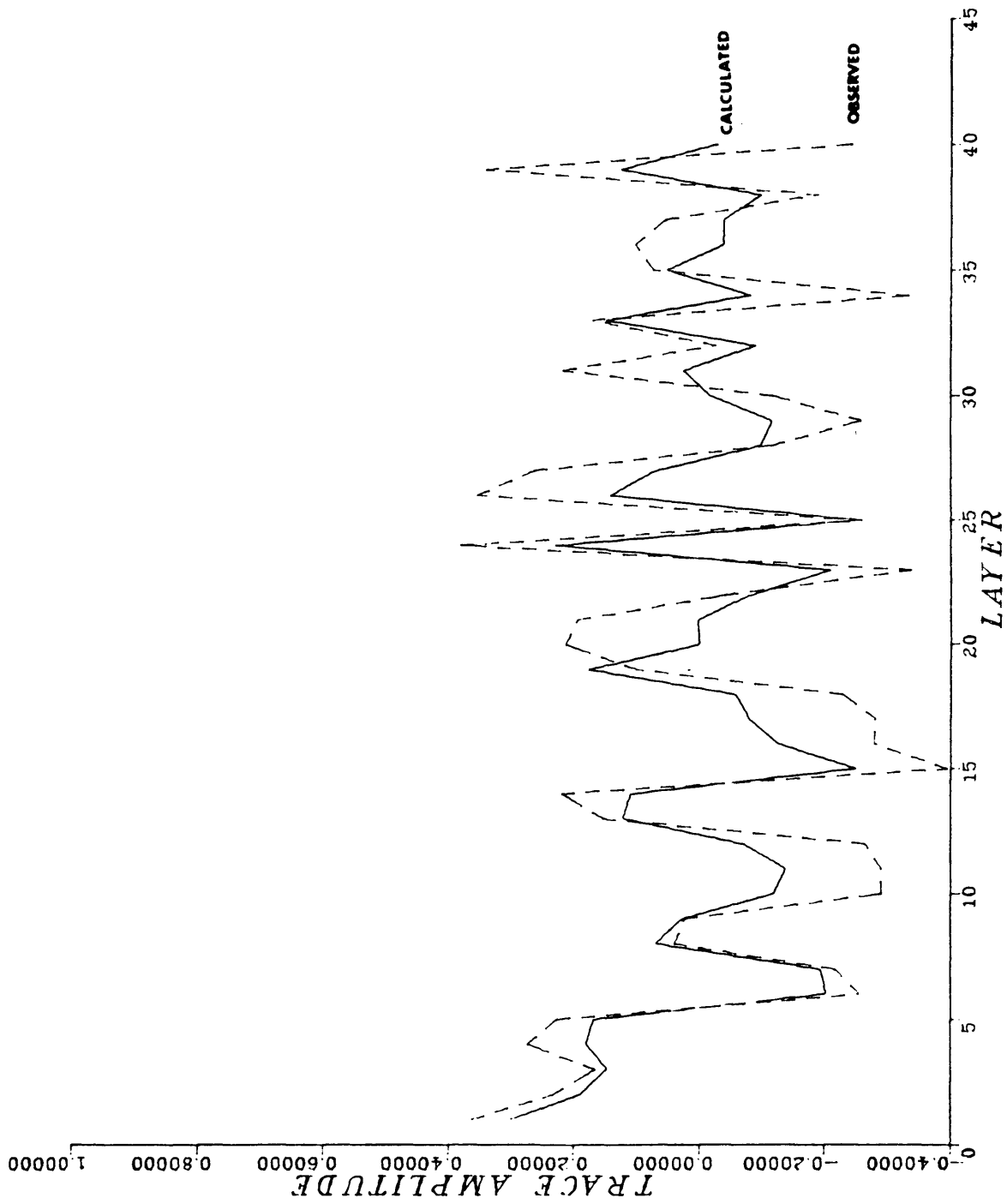


Figure 2-33 Observed and calculated trace of final inversion iteration for constant impedance model and .237170 RMS noise level.

TABLE 2

<u>Model #</u>	<u>Data RMS Level</u>	<u>Noise RMS Level</u>	<u>S/N</u>
1	.033021	.012985	2.5/1
2	.029603	.012985	2.3/1
3	.048041	.012985	3.7/1
4	.036163	.012985	2.8/1
5	.124314	.012985	9.6/1
6	.099556	.012985	7.7/1
7	.073689	.012985	5.7/1

TABLE 3

<u>Model #</u>	<u>Data RMS Level</u>	<u>Noise RMS Level</u>	<u>S/N</u>
1	.033021	.006216	5.3/1
2	.029603	.006216	4.8/1
3	.048041	.006216	7.7/1
4	.036163	.006216	5.8/1
5	.124314	.006216	20.0/1
6	.099556	.006216	16.0/1
7	.073689	.006216	11.9/1

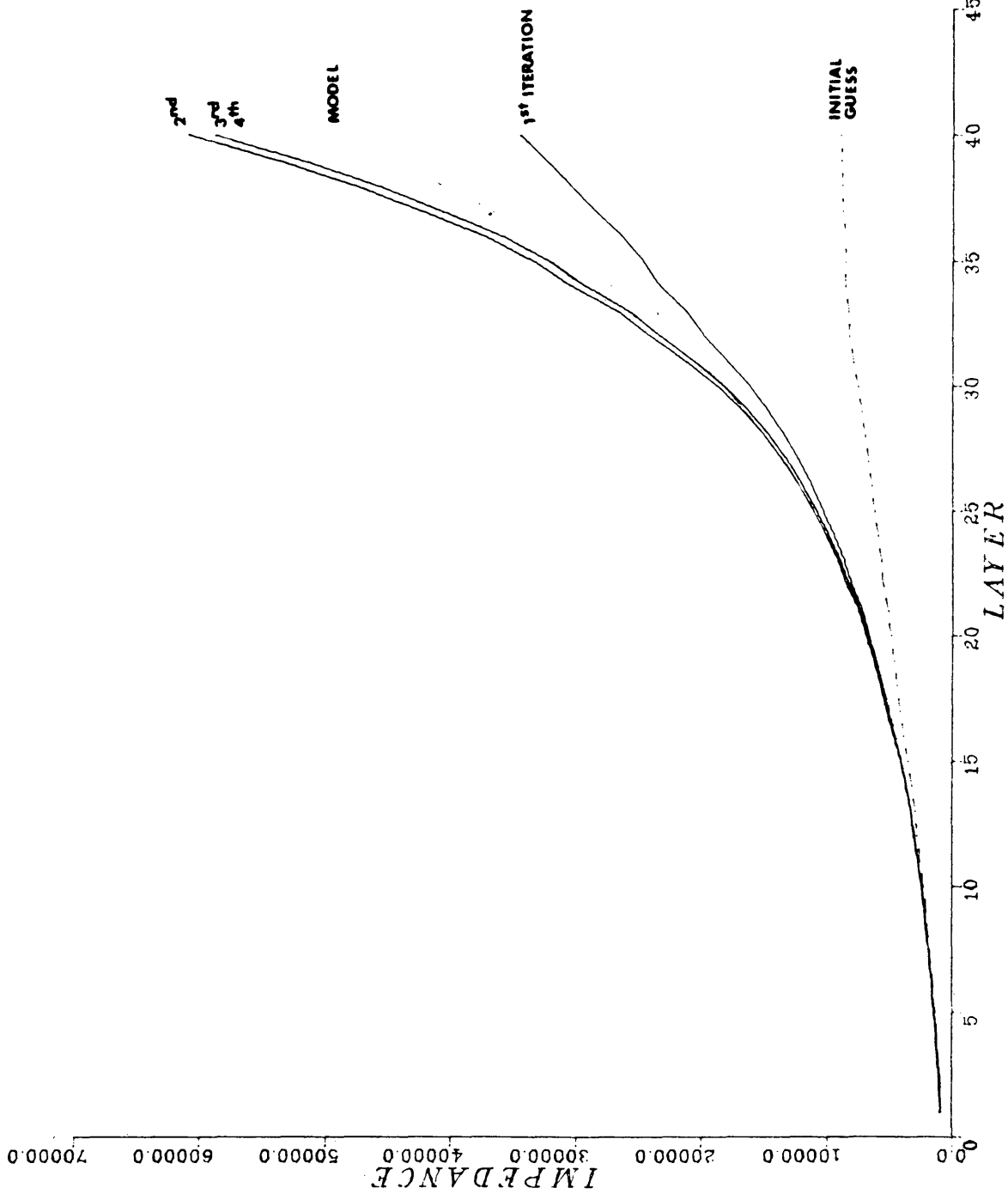


Figure 2-34 Inversion results for model 1 with .006216 RMS noise level.

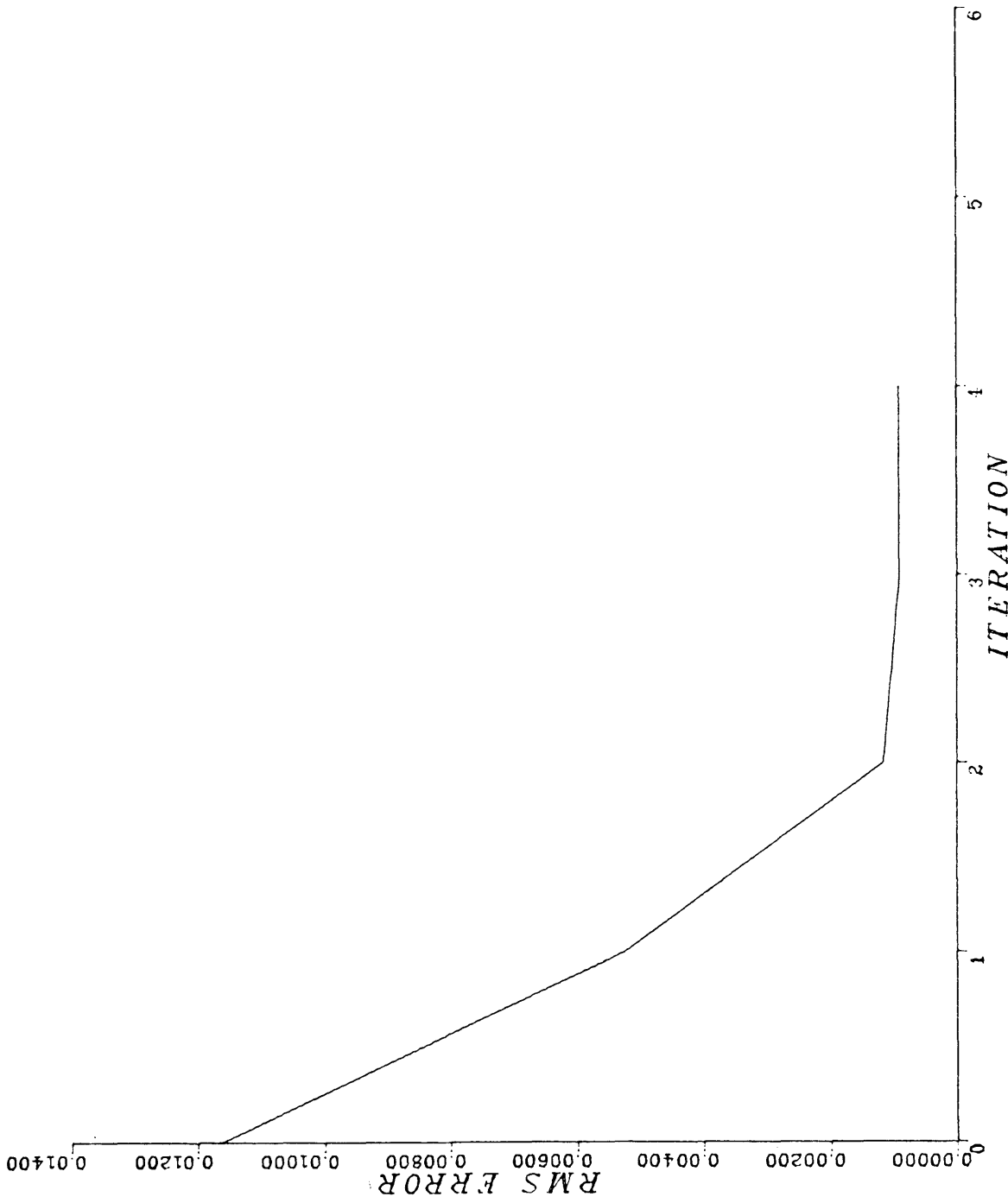


Figure 2-35 Model 1 RMS error for .006216 RMS noise level.

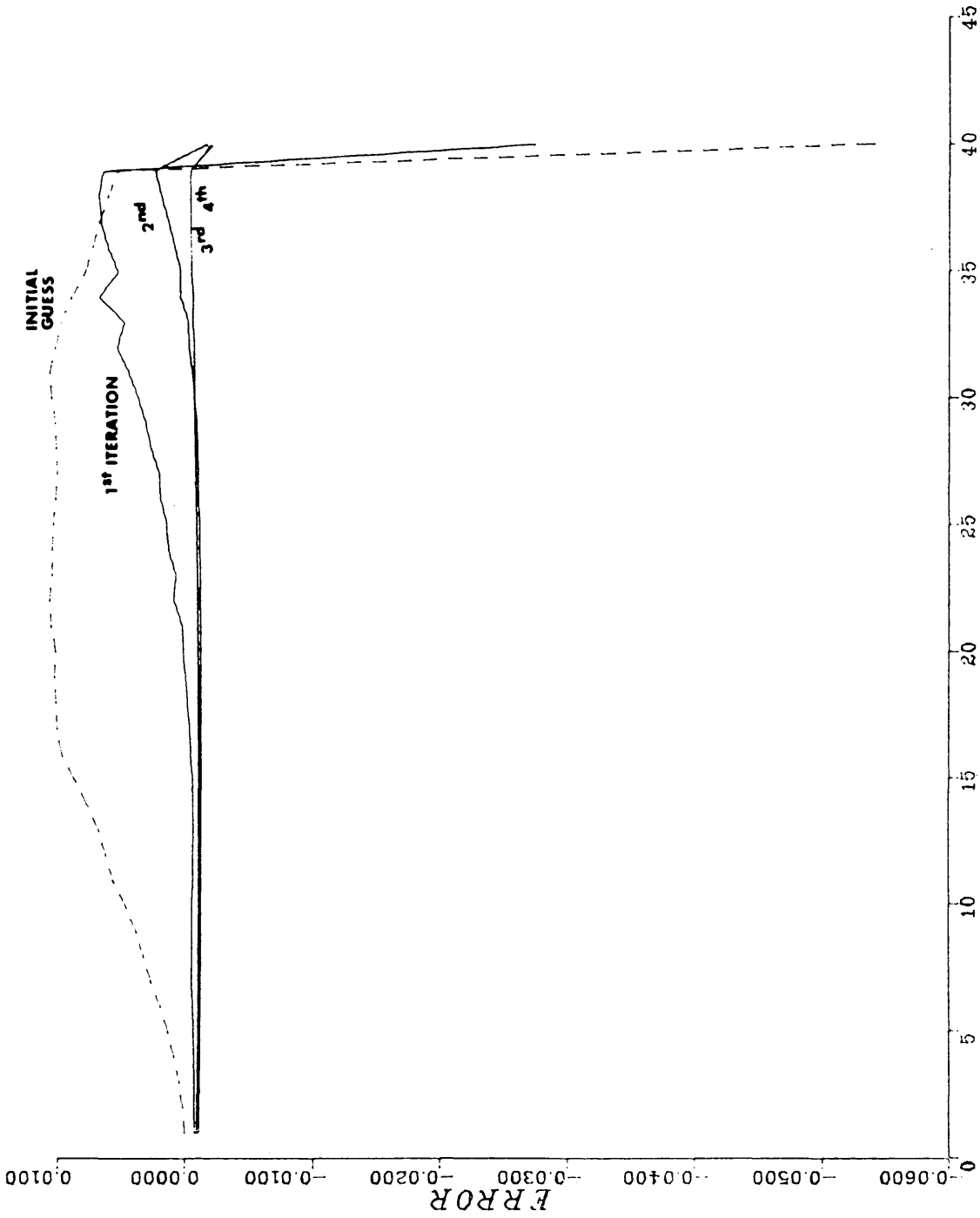


Figure 2-36 The observed minus calculated trace for successive iterations with model 1 and .006216 RMS noise level.

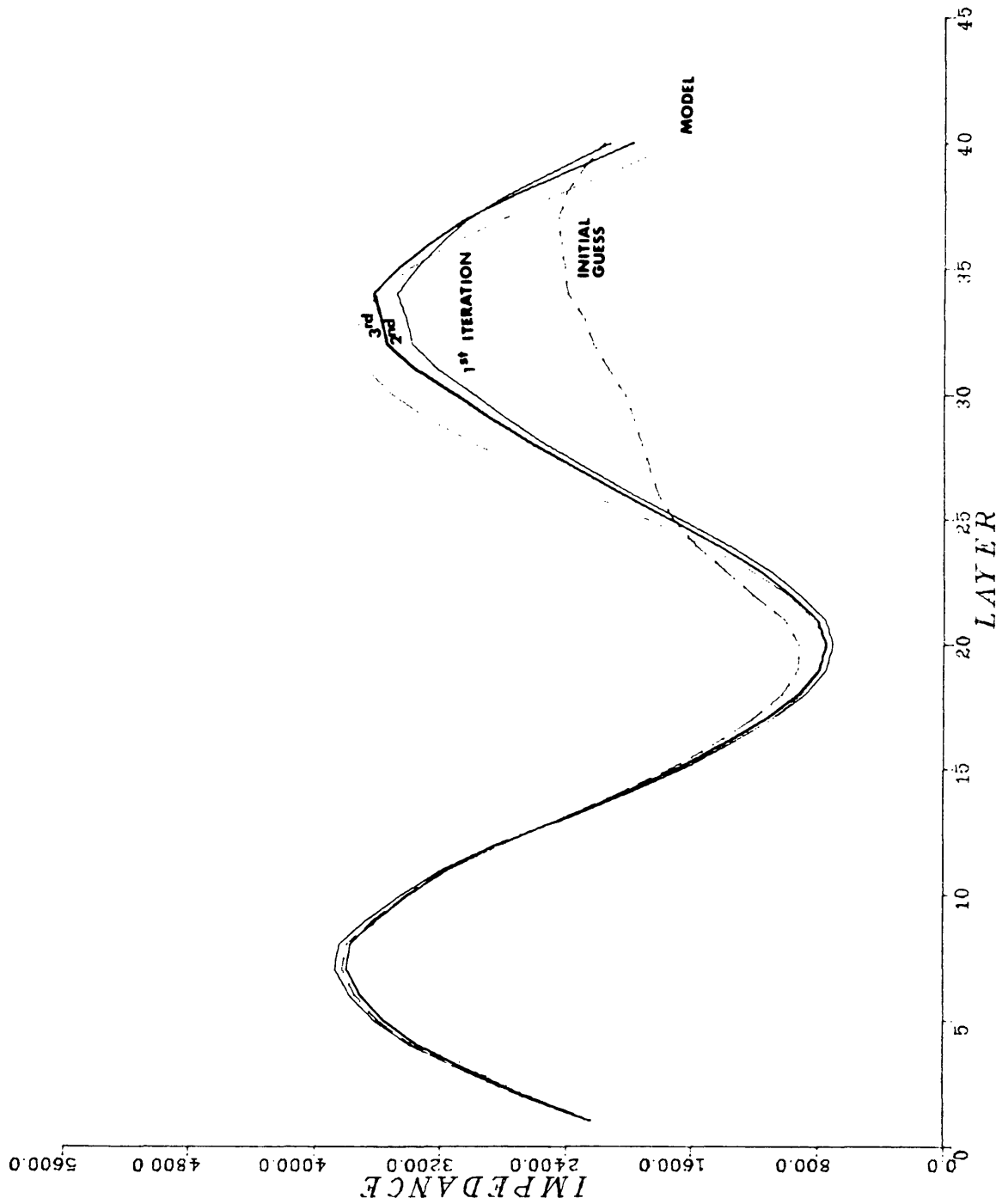


Figure 2-37 Inversion results for model 3 with .006216 RMS noise level.

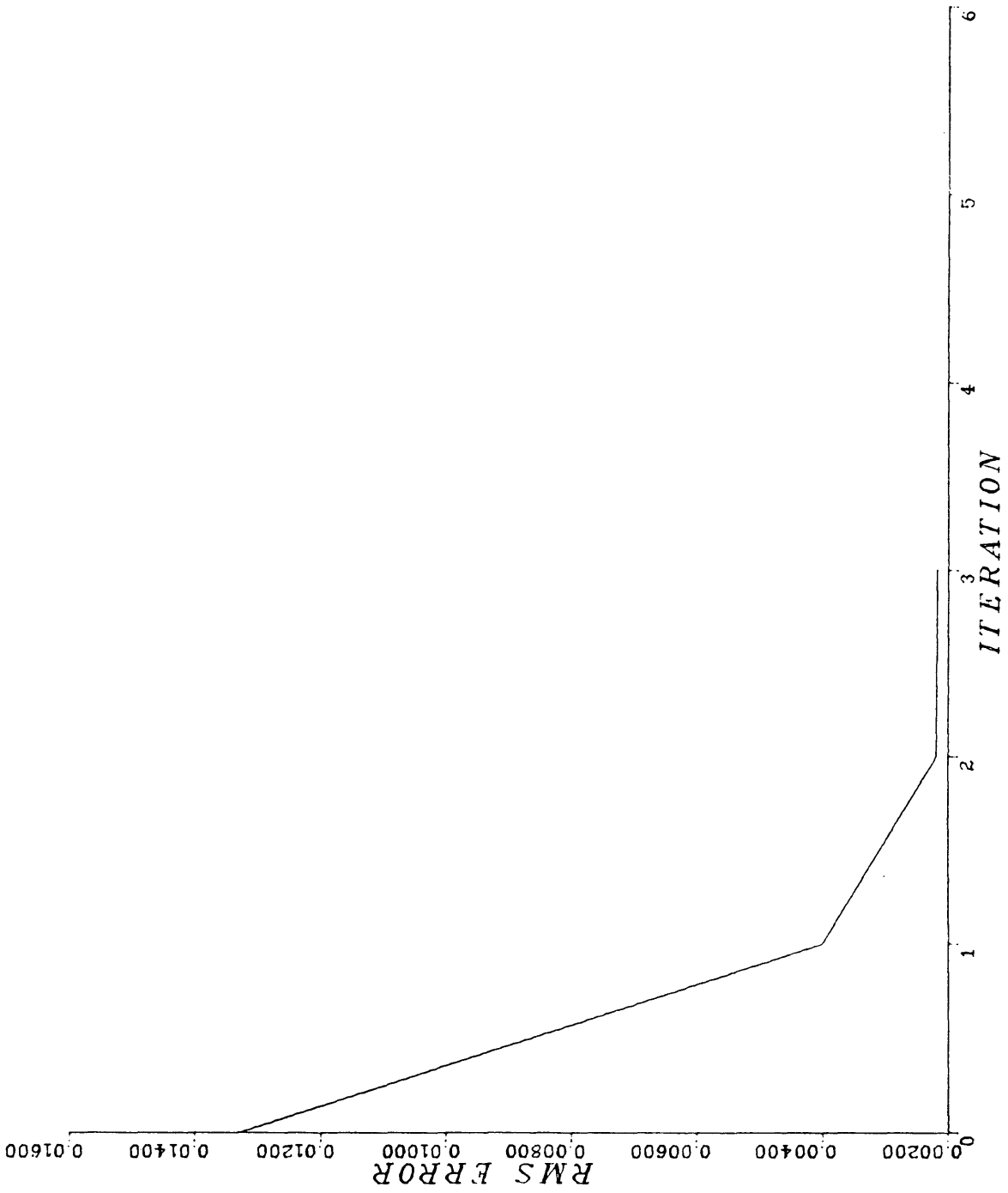


Figure 2-38 Model 3 RMS error for .006216 RMS noise level.

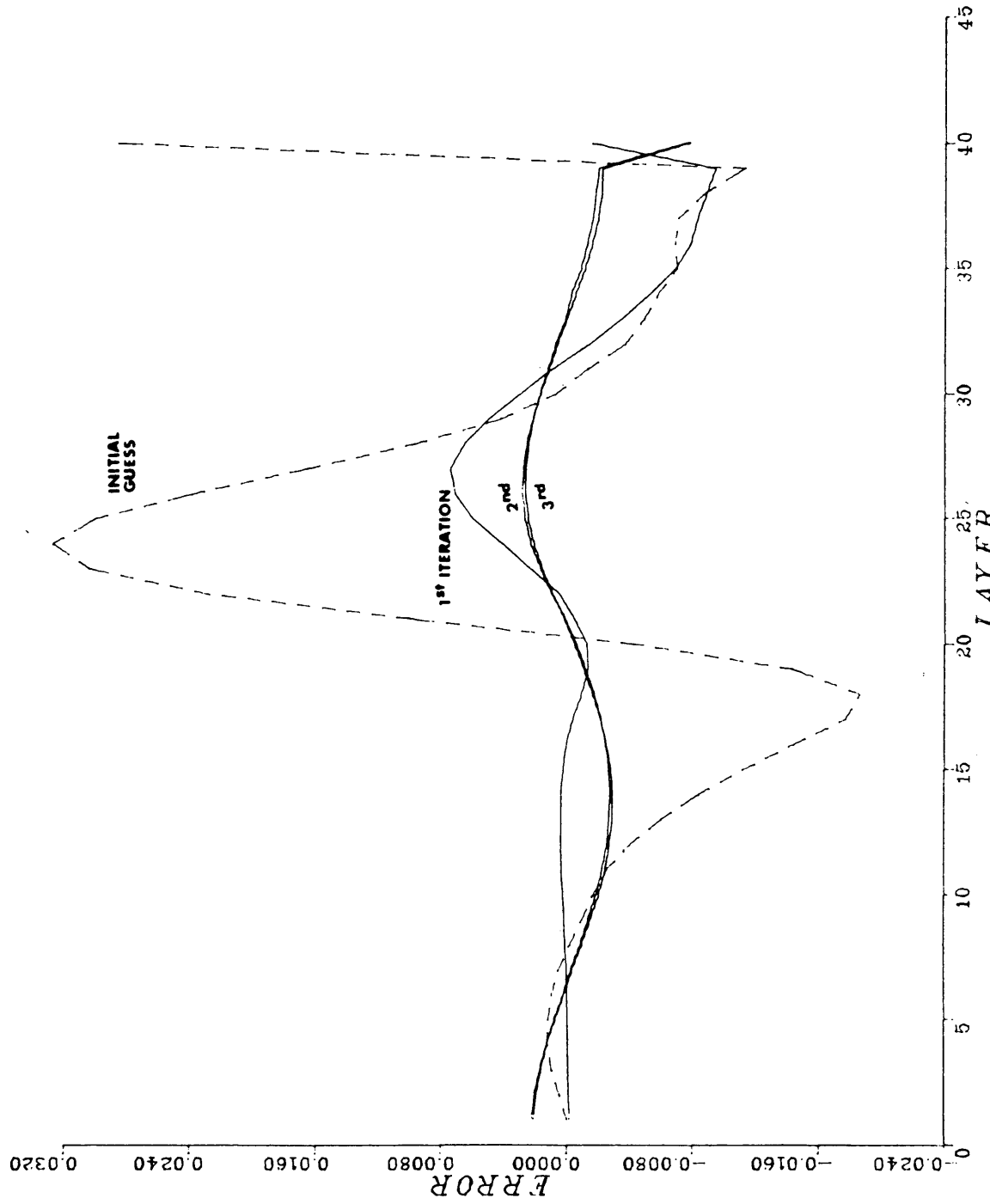


Figure 2-39 The observed minus calculated trace for successive iterations with model 3 and .006216 RMS noise level.

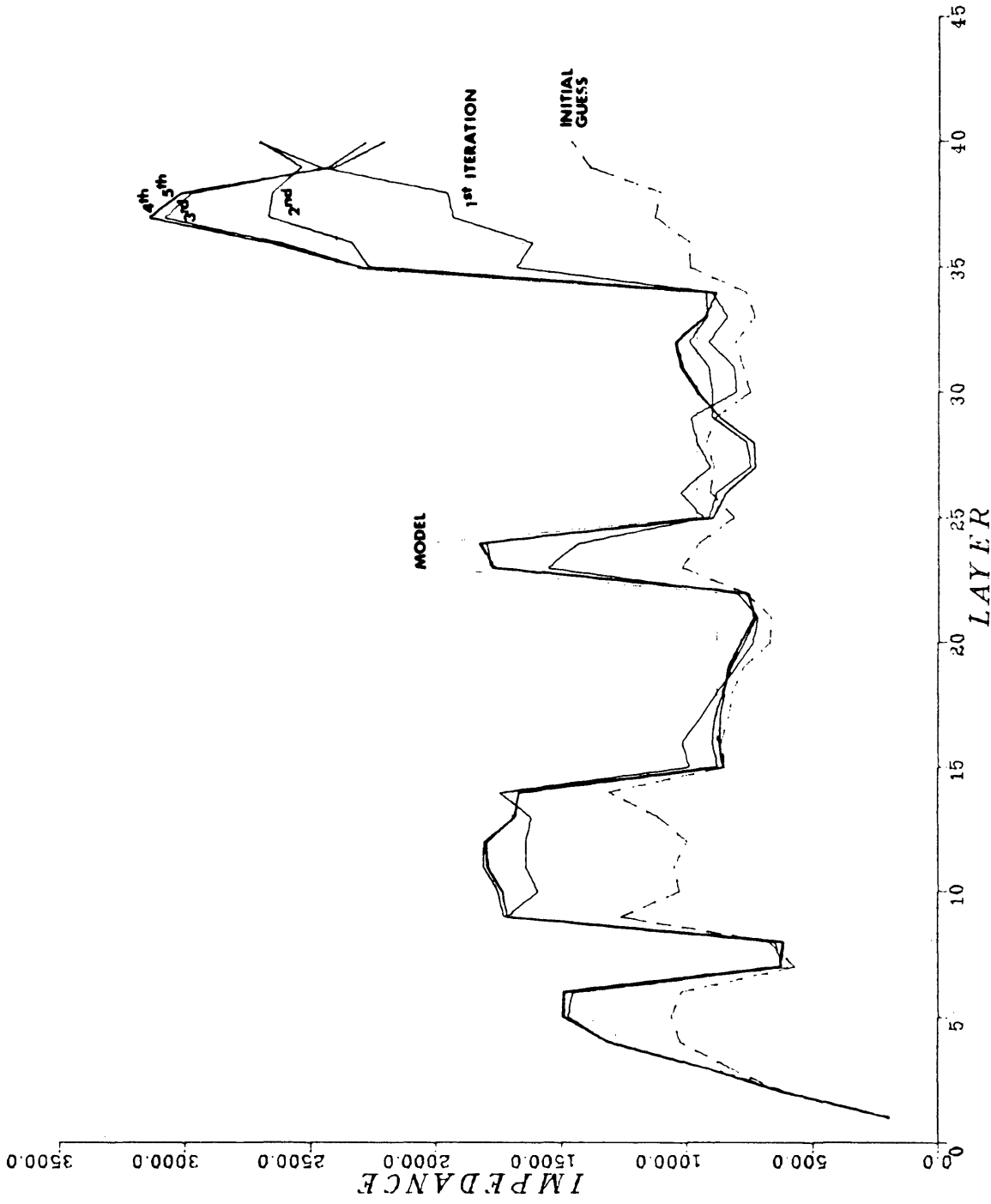


Figure 2-40 Inversion results for model 5 with .006216 RMS noise level.

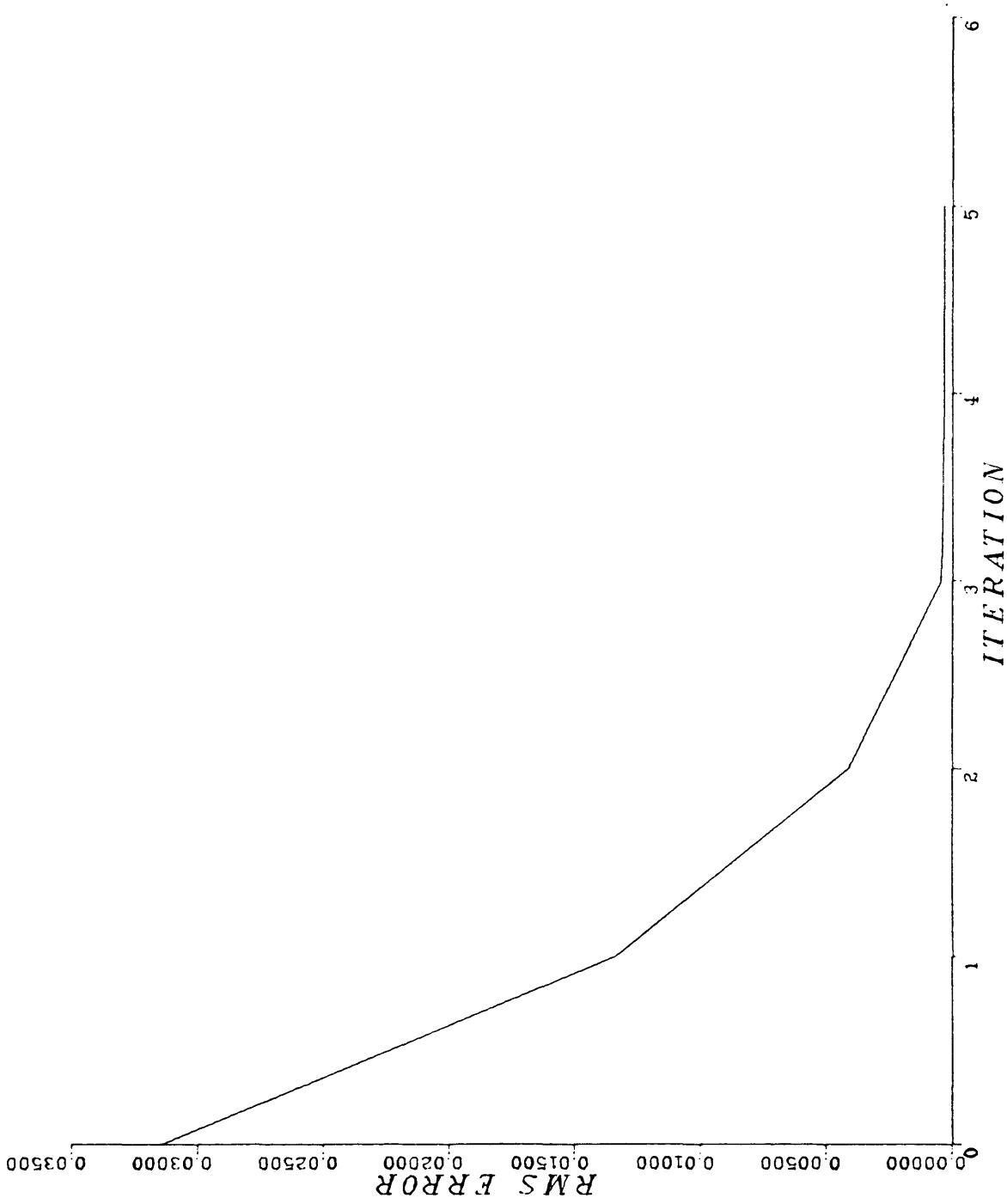


Figure 2-41 Model 5 RMS error for .006216 RMS noise level.

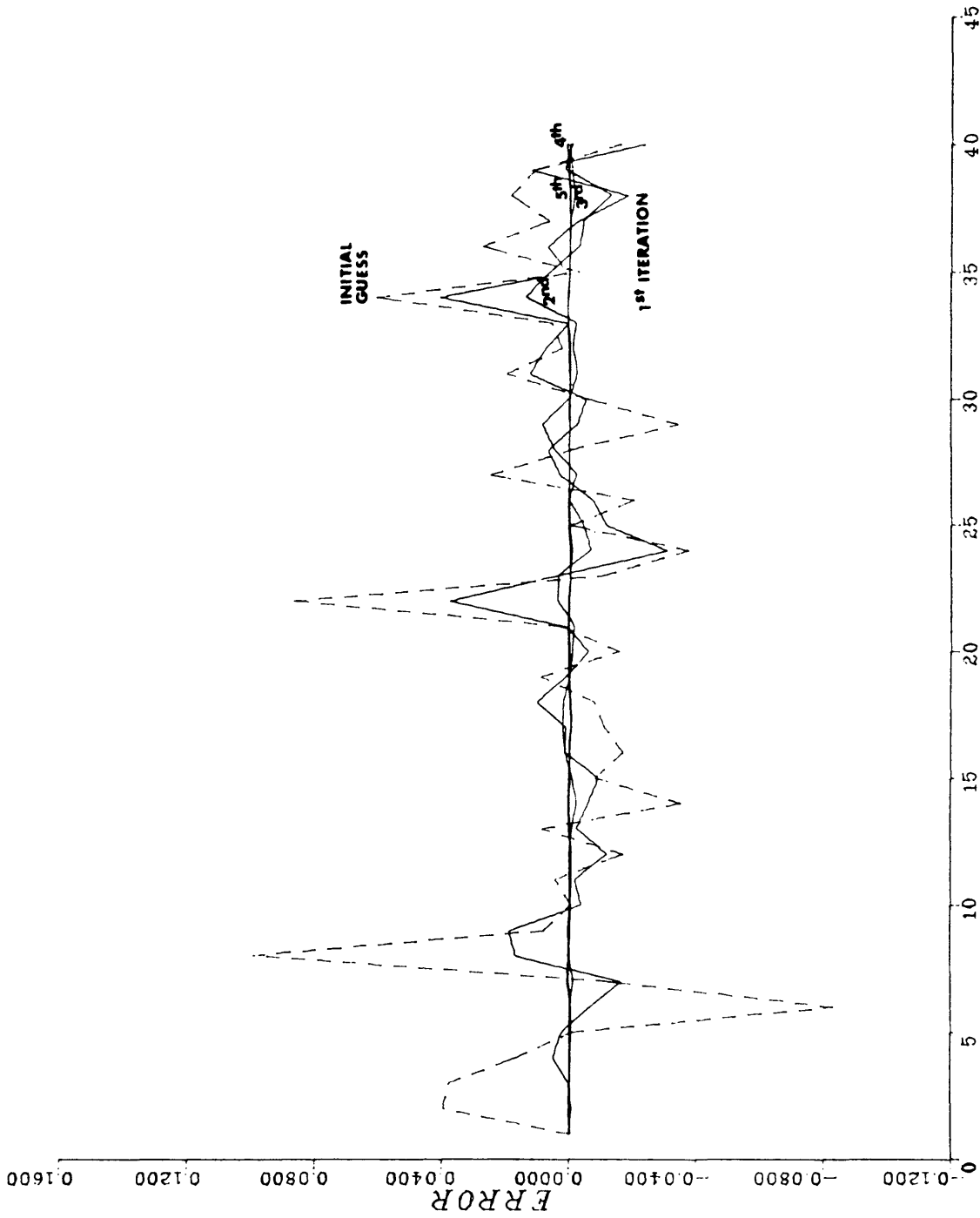


Figure 2-42 The observed minus calculated trace for successive iterations with model 5 and .006216 RMS noise level.

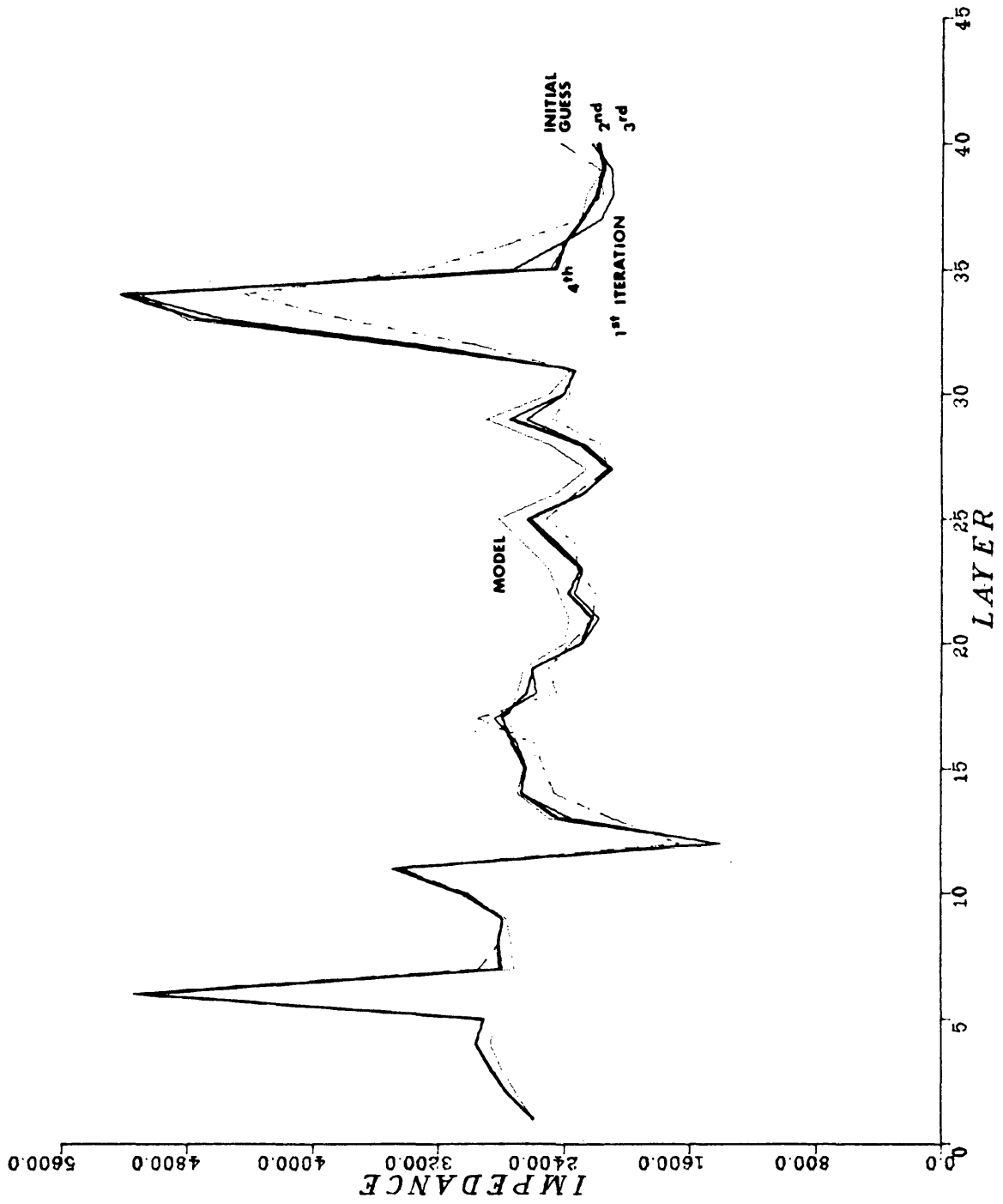


Figure 2-43 Inversion results for model 6 with .006216 RMS noise level.

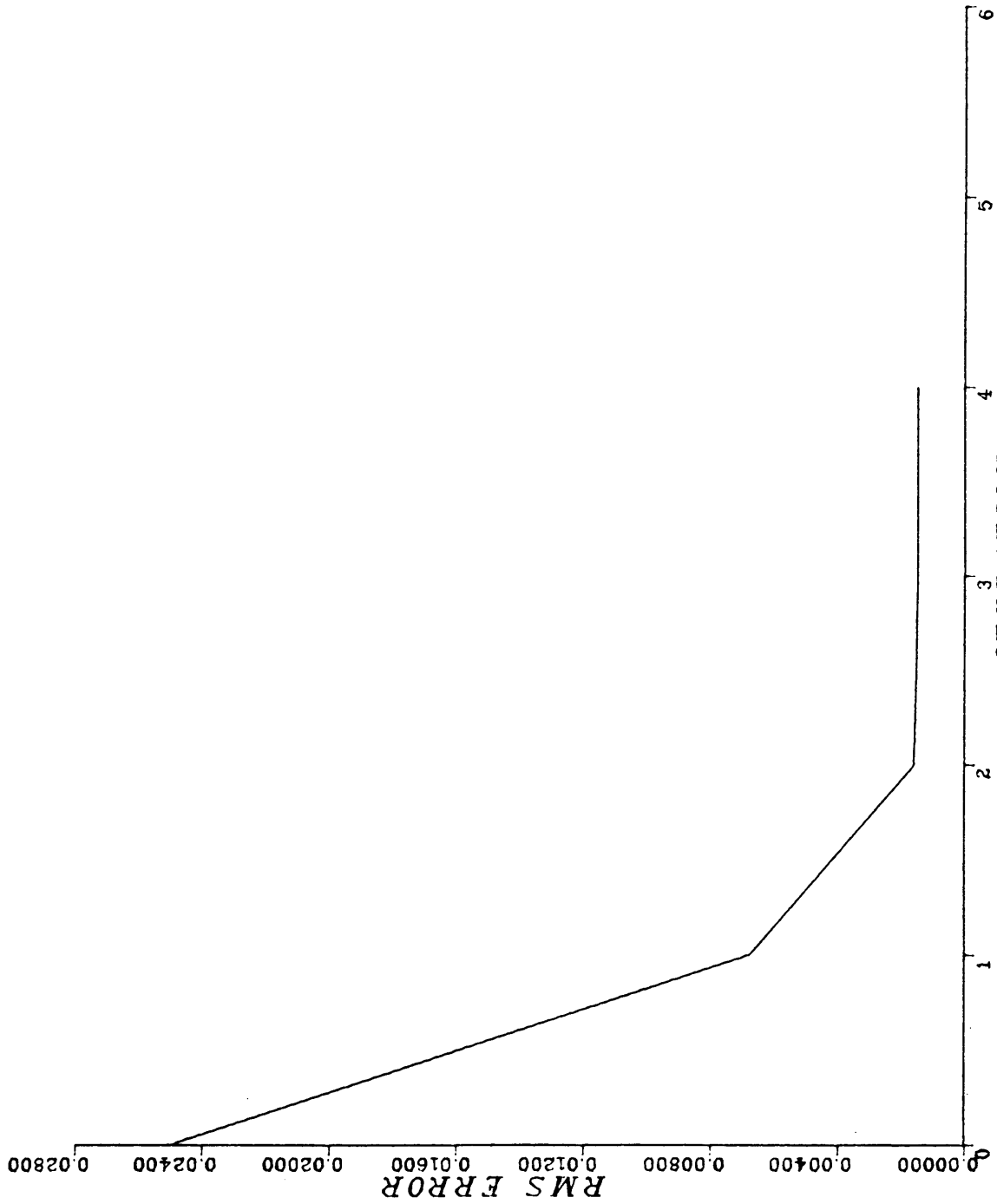


Figure 2-44 Model 6 RMS error for .006216 RMS noise level.

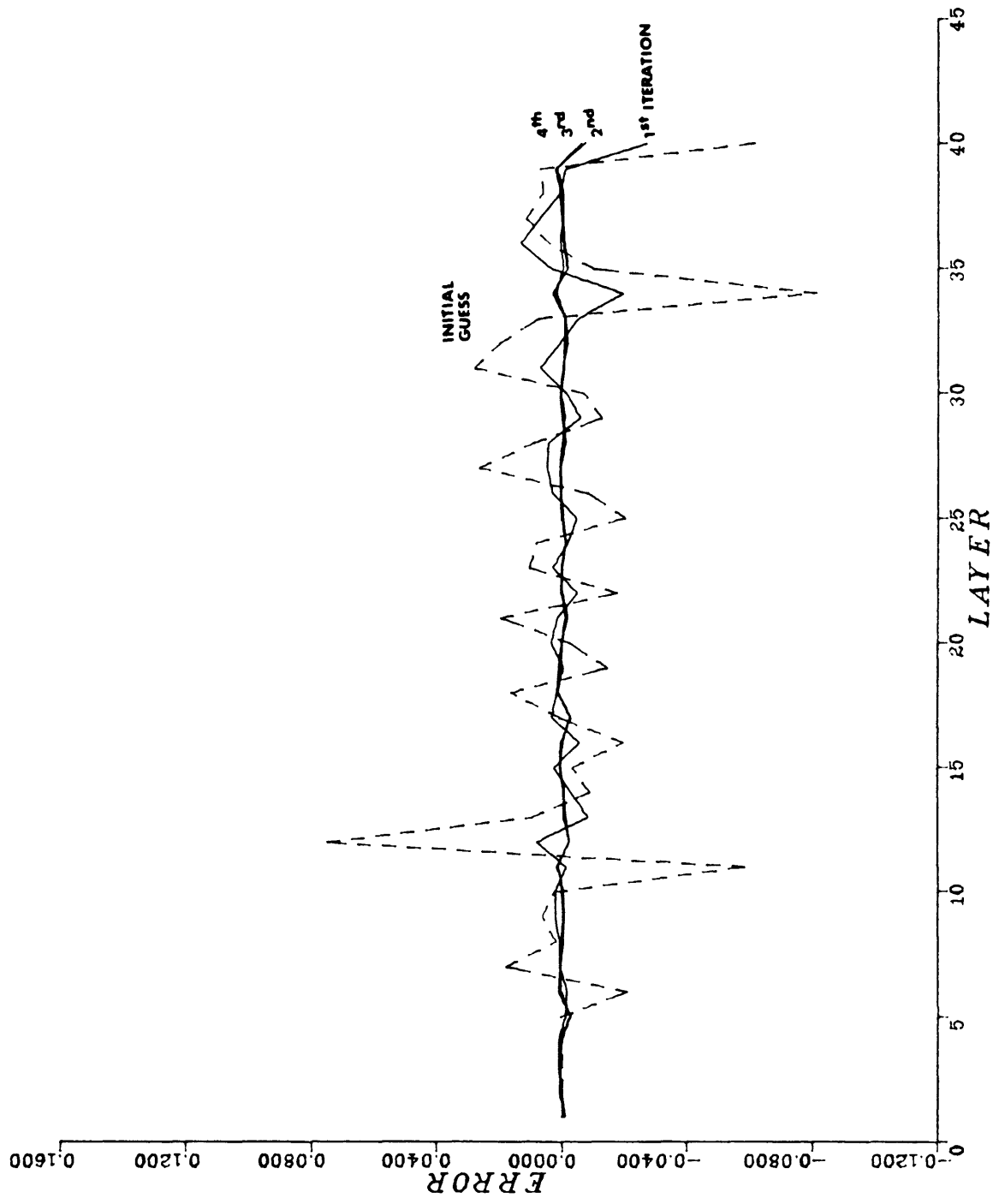


Figure 2-45 The observed minus calculated trace for successive iterations with model 6 and .006216 RMS noise level.

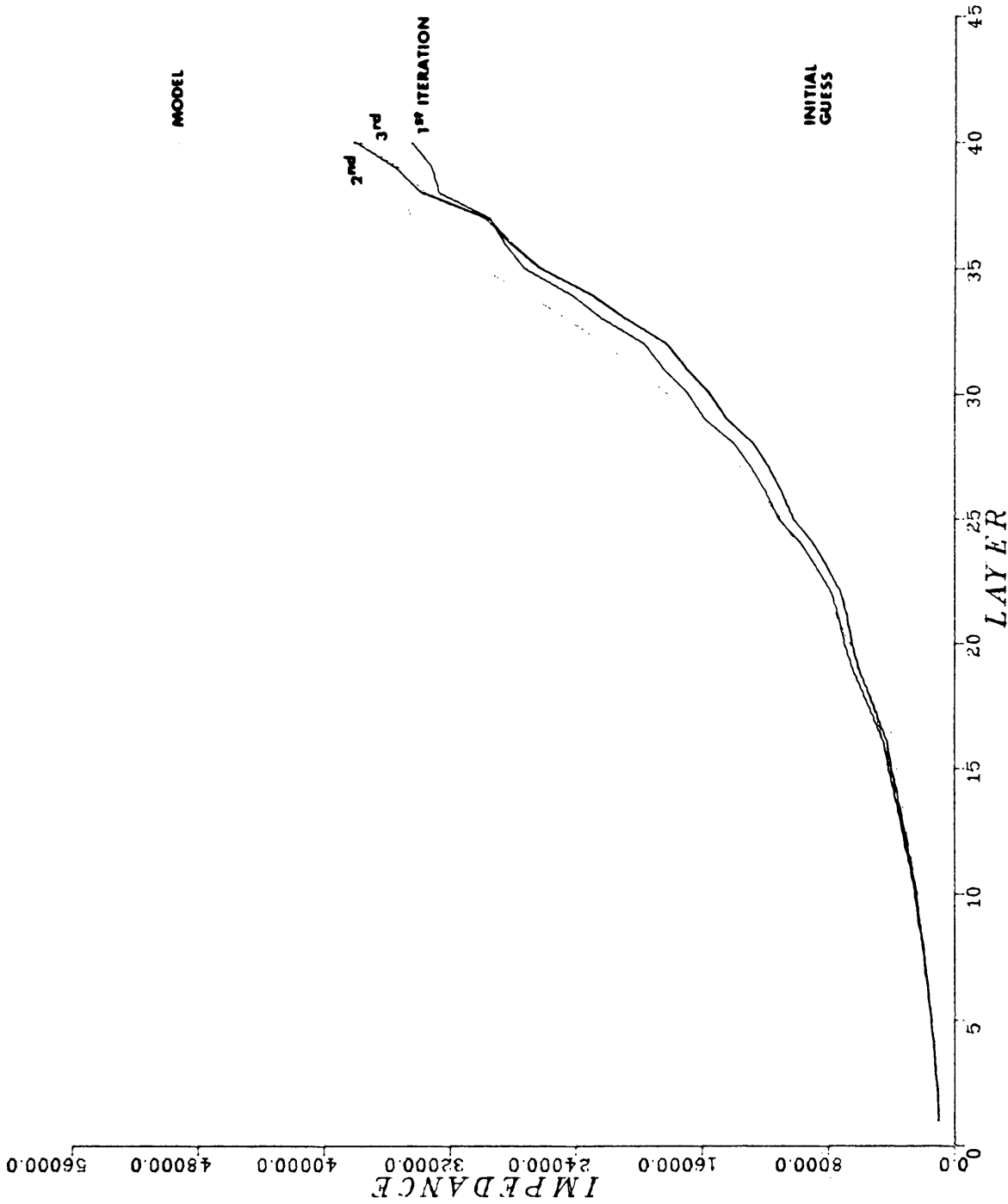


Figure 2-46 Inversion results for model 1 with .012985 RMS noise level.

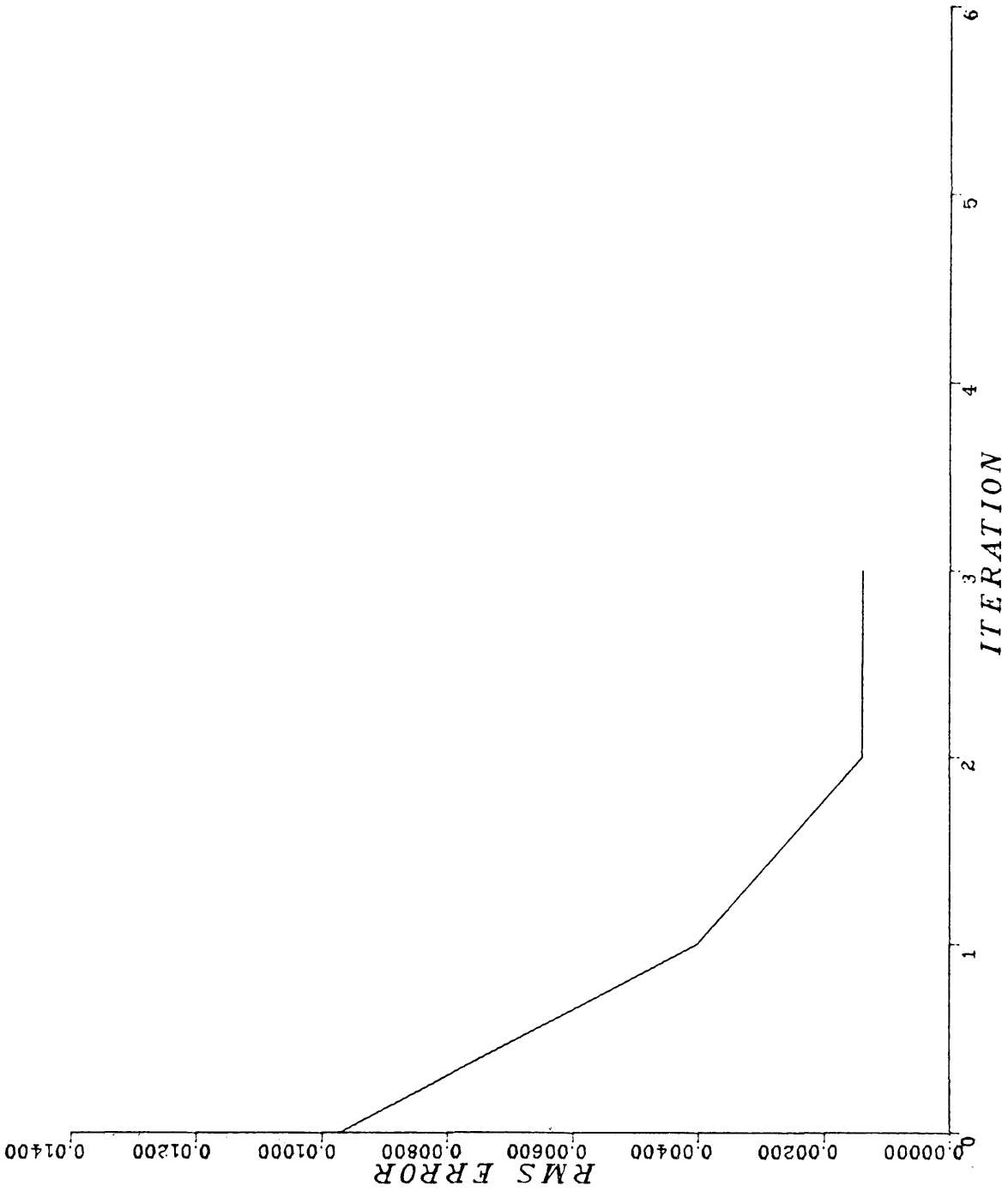


Figure 2-47 Model 1 RMS error for .012985 RMS noise level.

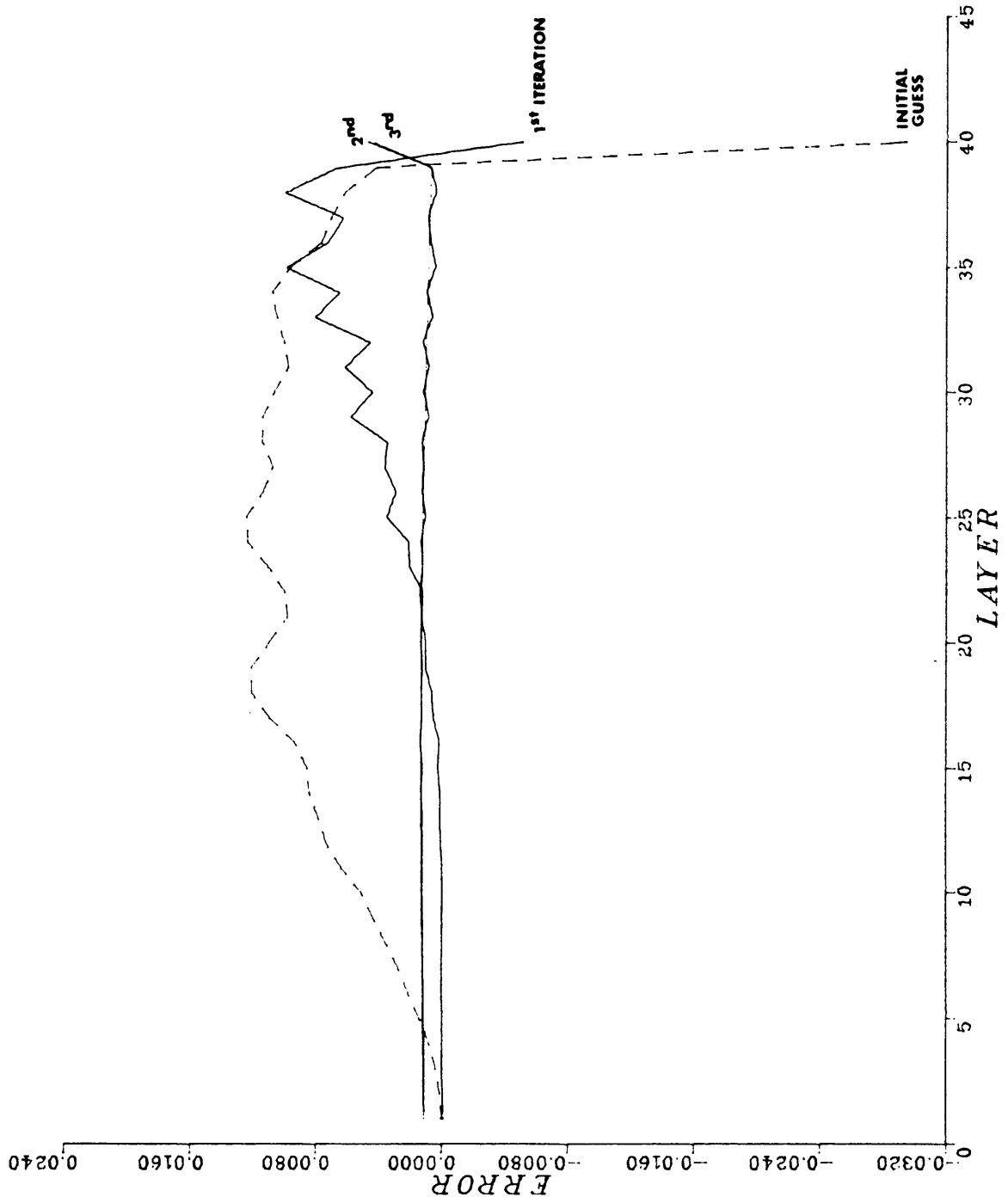


Figure 2-48 The observed minus calculated trace for successive iterations with model 1 and .012985 RMS noise level.

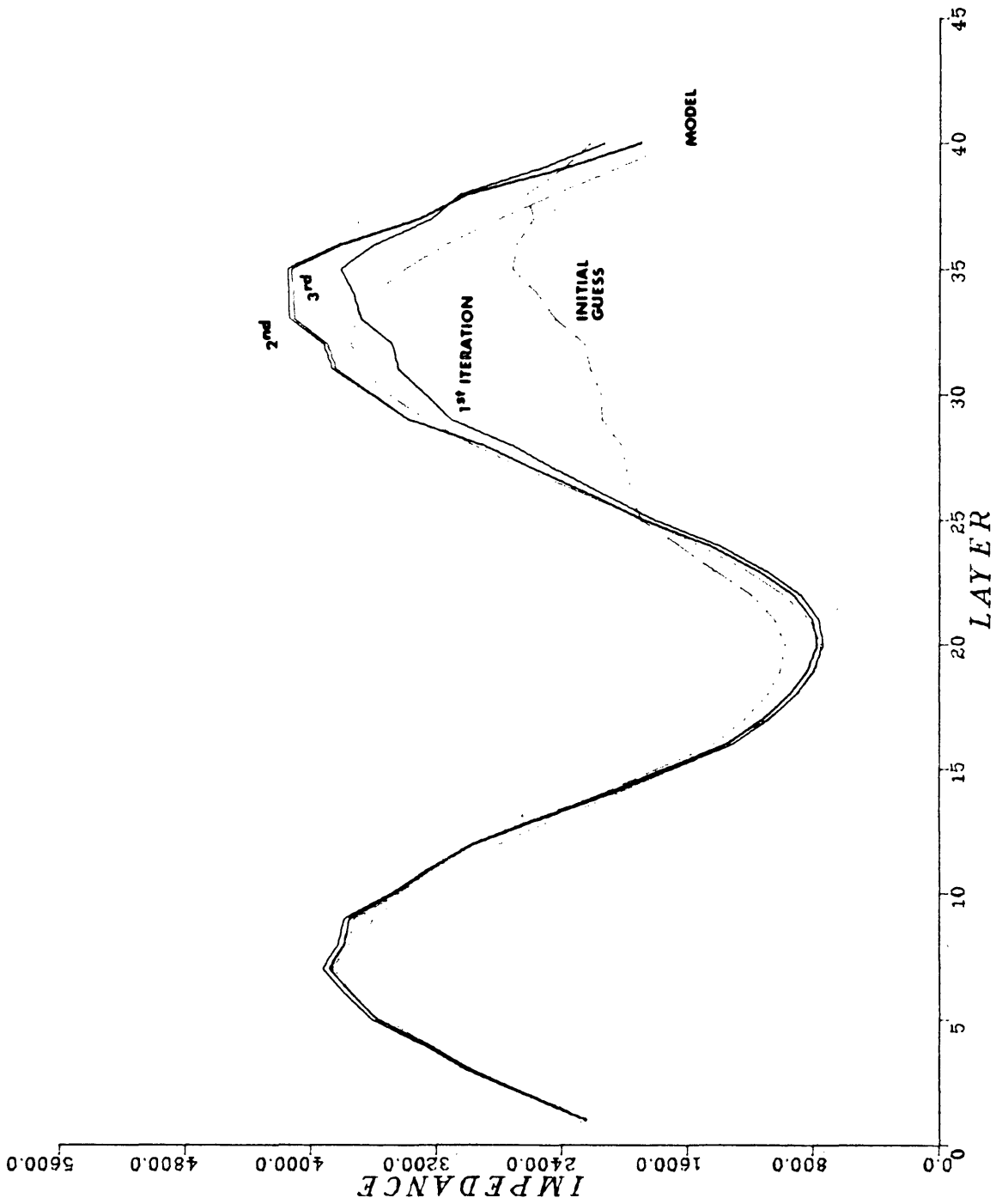


Figure 2-49 Inversion results for model 3 with .012985 RMS noise level.

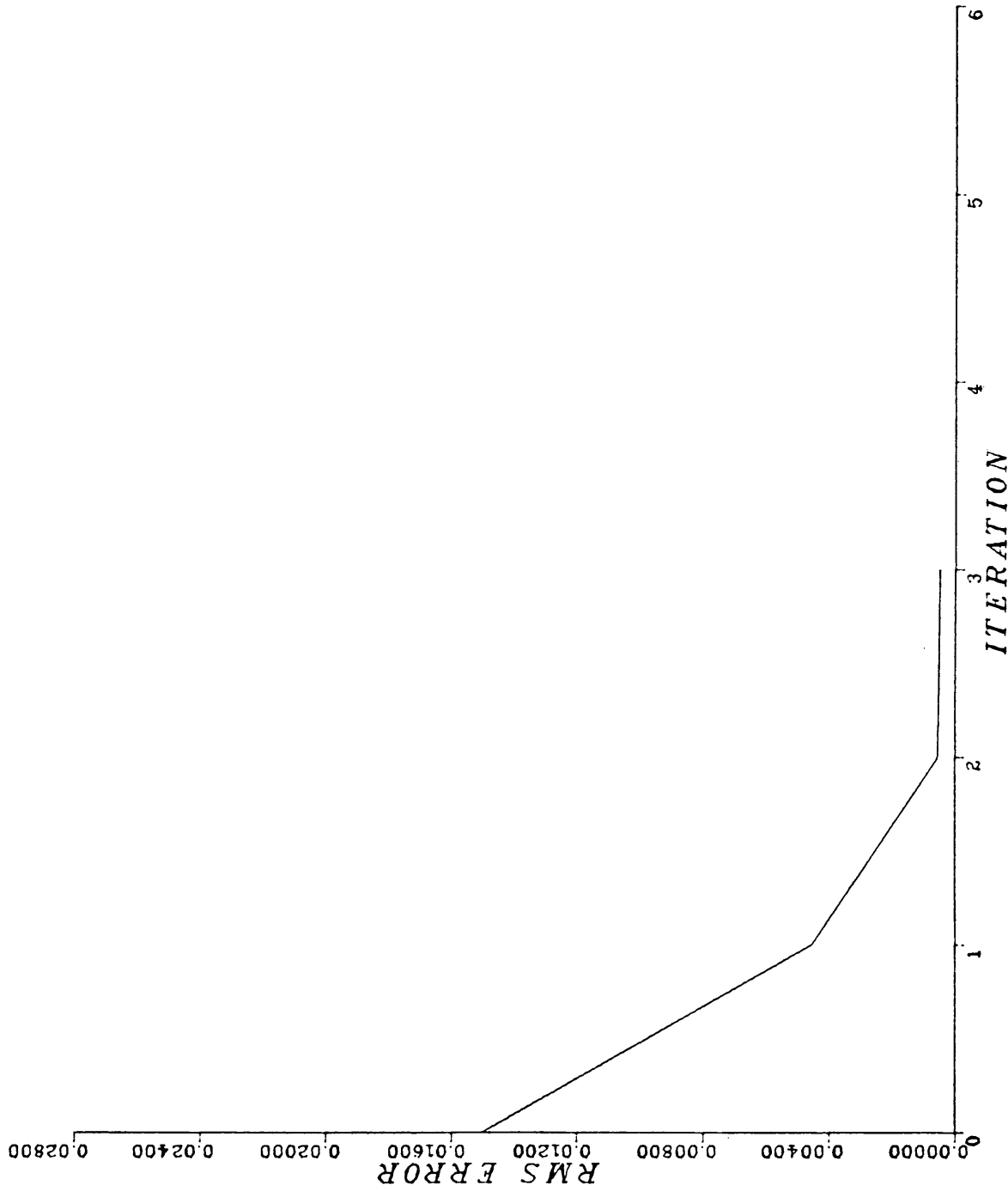


Figure 2-50 Model 3 RMS error for .012985 RMS noise level.

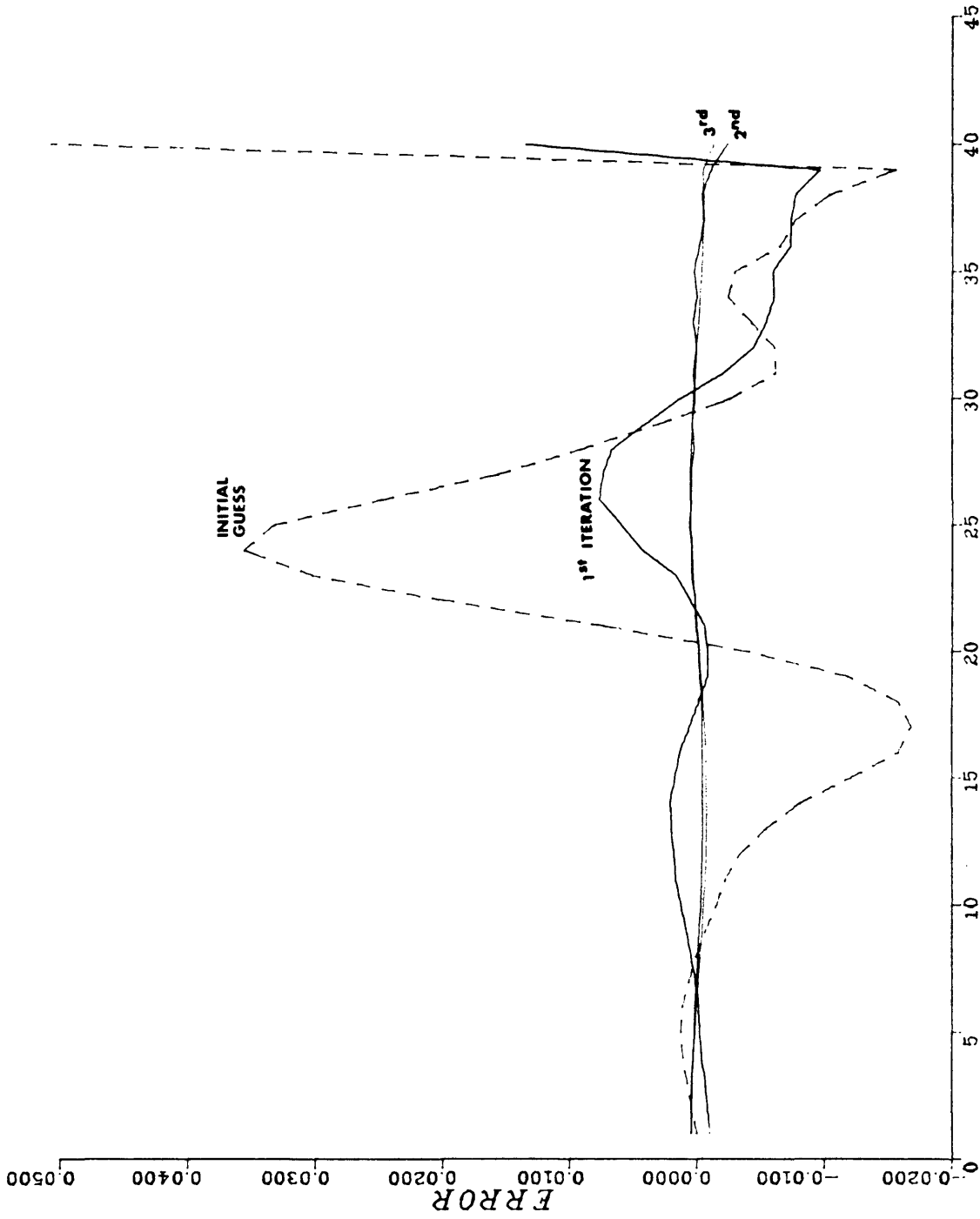


Figure 2-51 The observed minus calculated trace for successive iterations with model 3 and .012985 RMS noise level.

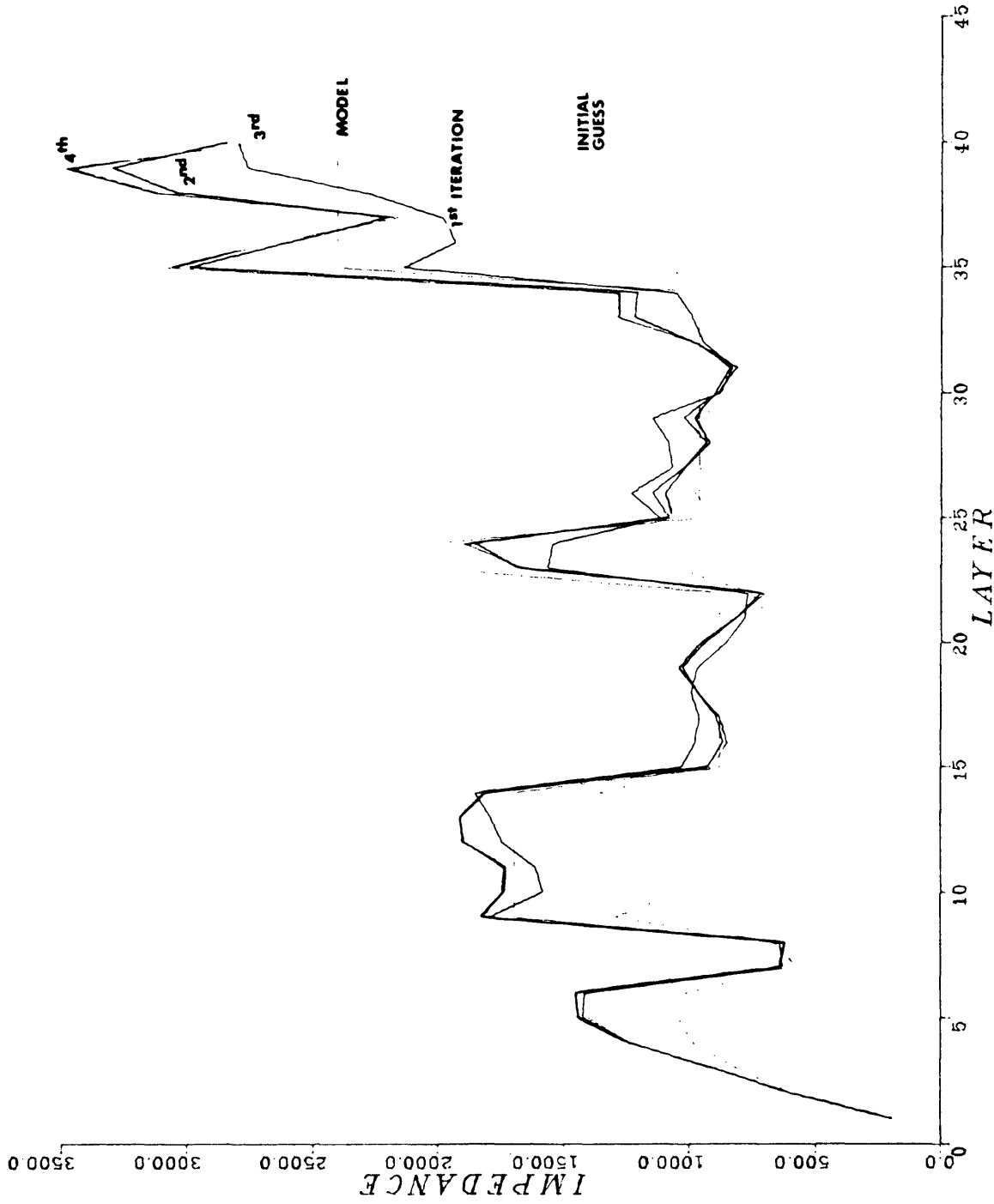


Figure 2-52 Inversion results for model 5 with .012985 RMS noise level.

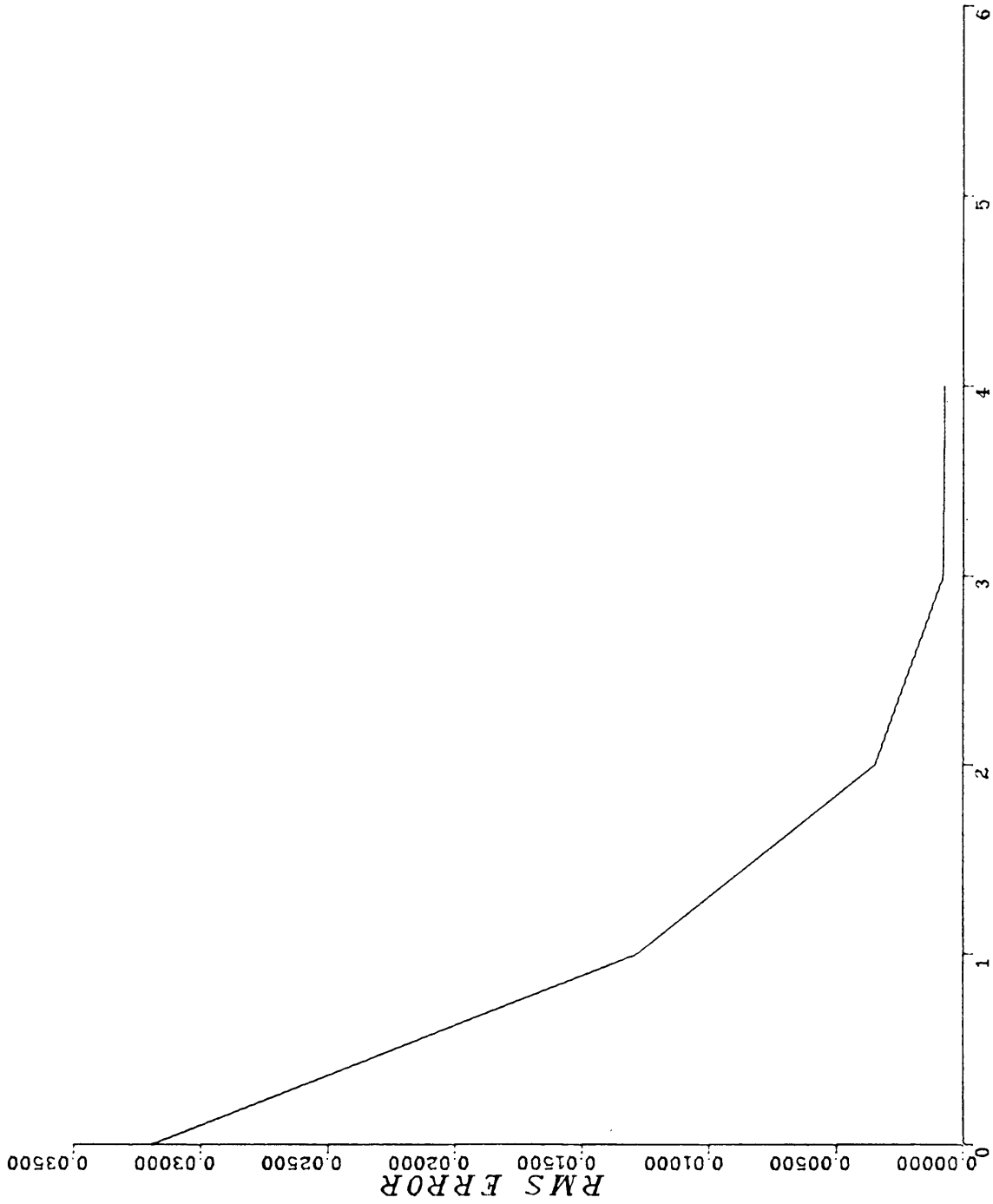


Figure 2-53 Model 5 RMS error for .012985 RMS noise level.

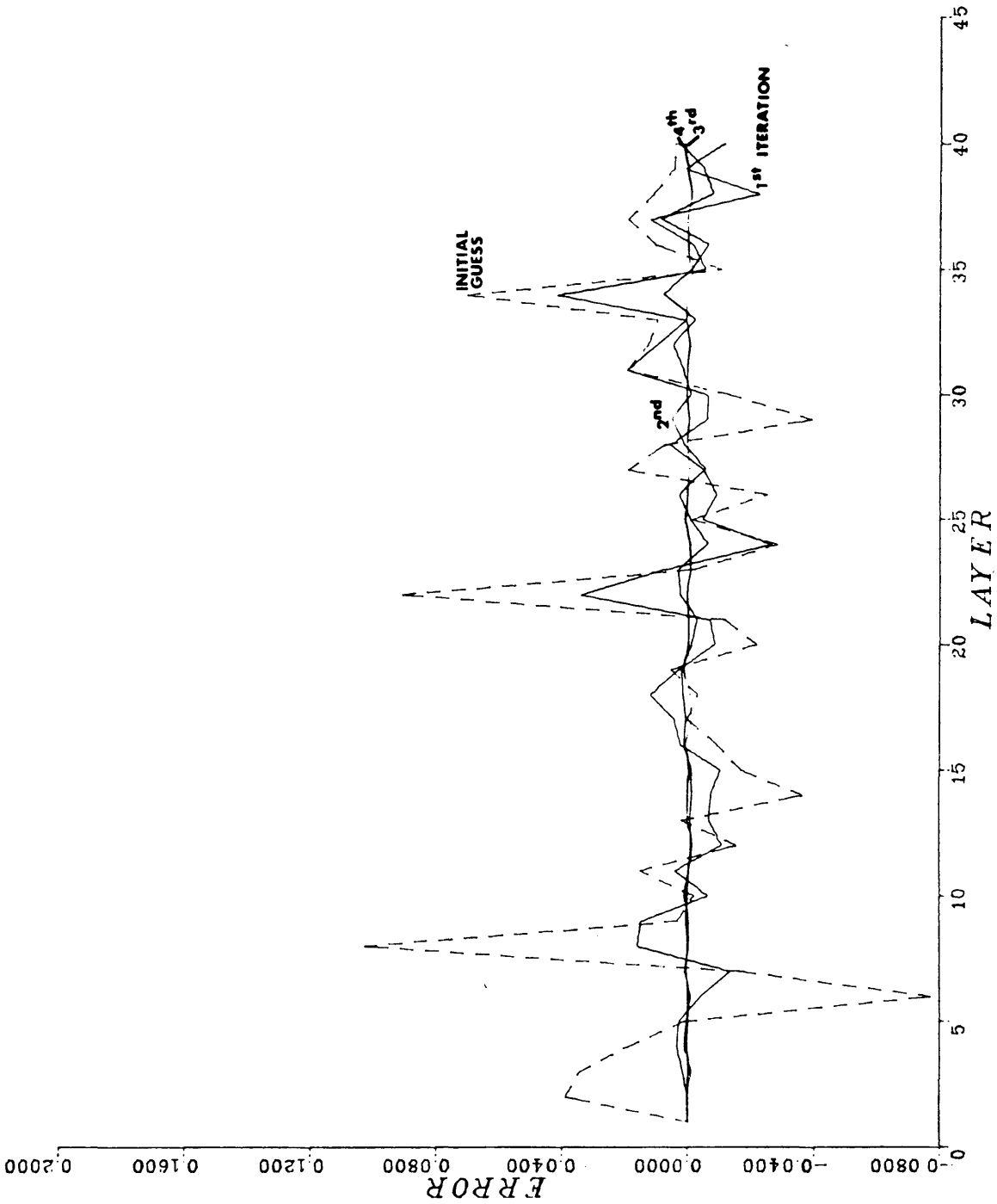


Figure 2-54 The observed minus calculated trace for successive iterations with model 5 and .012985 RMS noise level.

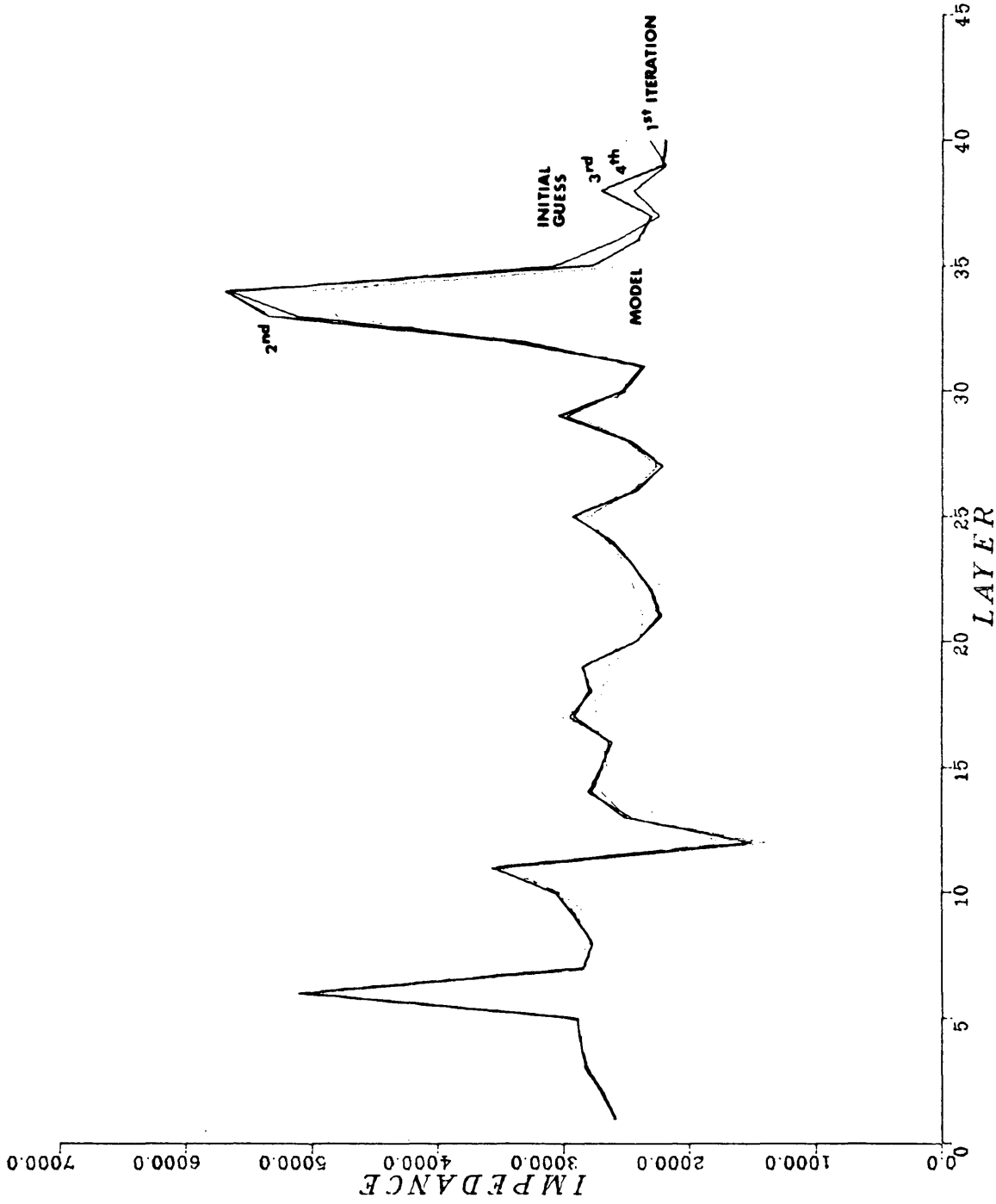


Figure 2-55 Inversion results for model 6 with .012985 RMS noise level.

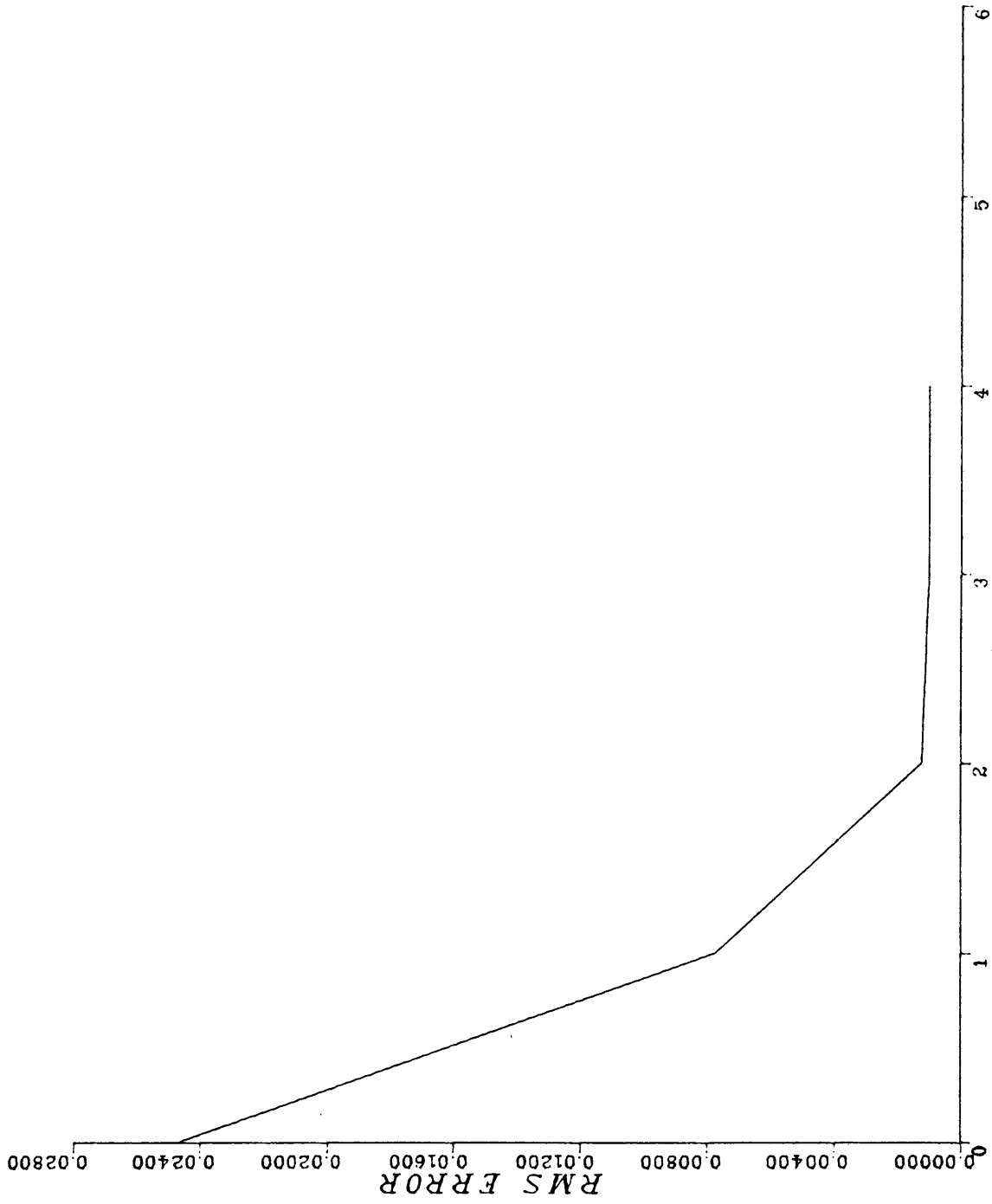


Figure 2-56 Model 6 RMS error for .012985 RMS noise level.

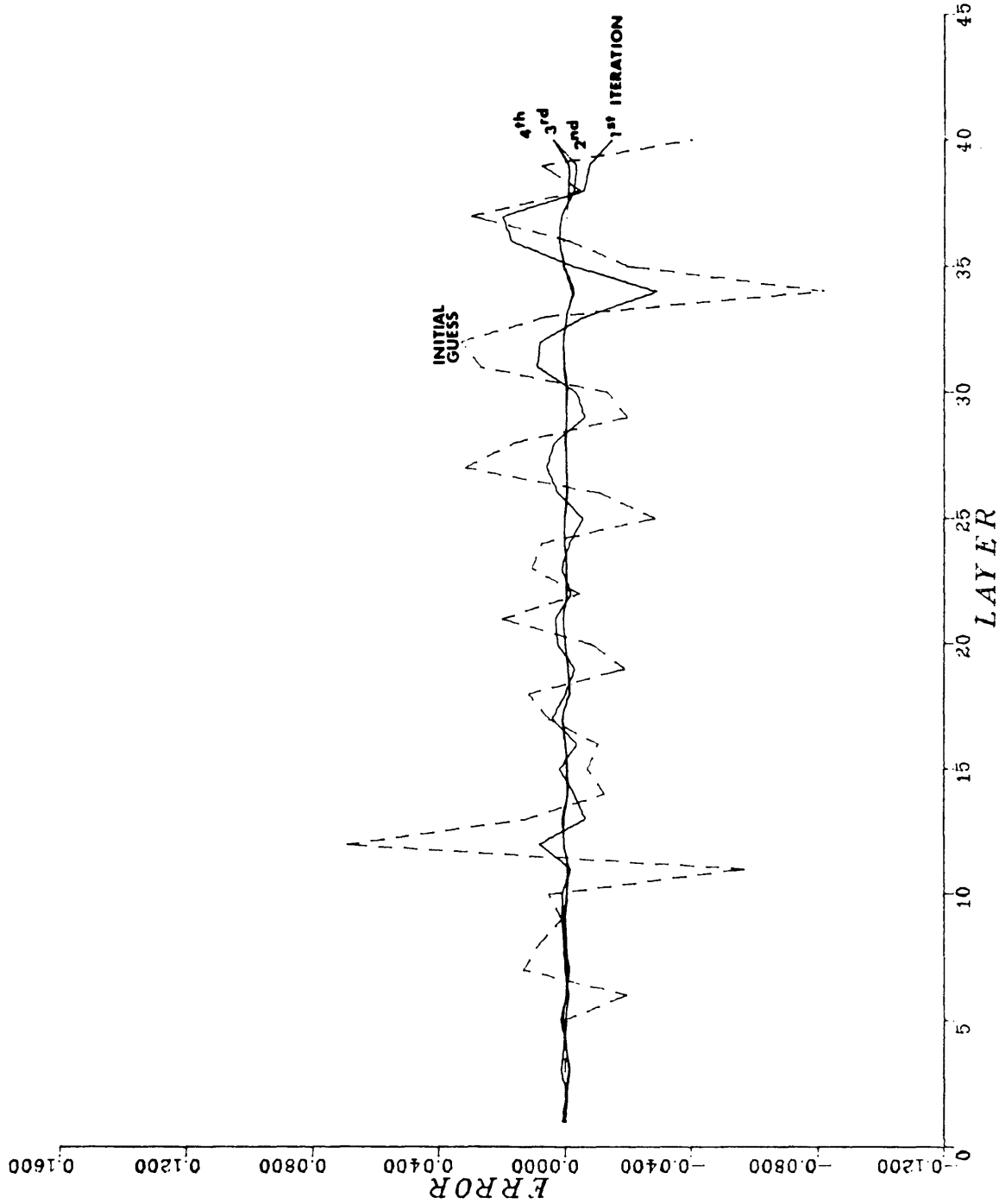


Figure 2-57 The observed minus calculated trace for successive iterations with model 6 and .012985 RMS noise level.

Using figure 2-34, which illustrates model 1, we observe that although the program does not converge to the model parameters, the results are still fairly good. In general, the plots showing the results for the other models indicate trends are well preserved with the inverted parameters oscillating around the model parameters. In particular, the model parameter determination does not diverge for the deeper layers. Bamberger et al (1976) commented that this particular problem was a major shortcoming of Kunetz's method (Kunetz, 1963) and caused it to be unstable.

The plots of error vs. time indicate the same behavior we note in the RMS error vs. iteration plots. That is, the error is reduced at each iteration until we reach some irreducible error. This error is evenly distributed between the various layers regardless of depth or impedance - a fact we would hope for with a least squares approach.

The method, then, is stable in the presence of random noise. For all models considered, the effects of the noise were most pronounced for model 5. At points in the model of constant

impedance for several layers, the noise had its maximum effect. This seems to agree with intuition, as at these points the model is generating no primaries. The inversion method has only the multiple energy to work with, which in the presence of noise, can be strongly corrupted.

c) Convolutional Noise

Up to this point, the assumption has been made that no convolutional noise has been present in the signal. In fact there normally is convolutional noise in the form of filtering or incomplete deconvolution. Considered now will be what effect a wavelet convolved with the reflectivity series will have.

In much of the literature on inversion of seismic data, the assumption is made that the source wavelet is known and can therefore be removed in deconvolution or left in the formulation of the solution as a known quantity. If this were not the case, however, how would the present technique respond?

To test this, the program was adapted to accept a wavelet which would be our theoretical source wavelet. This source wavelet is then

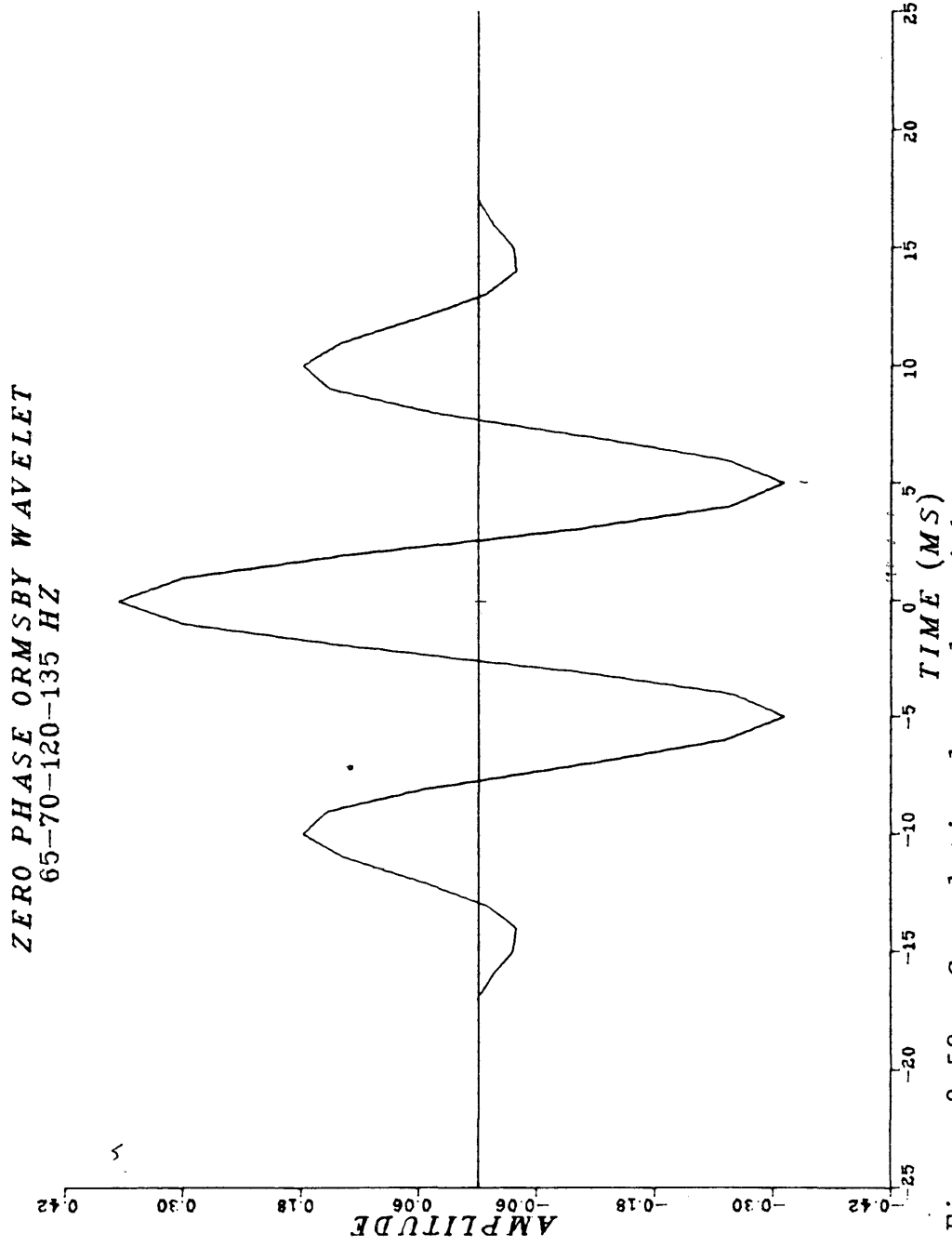


Figure 2-58 Convolutional wavelet with zero noise.

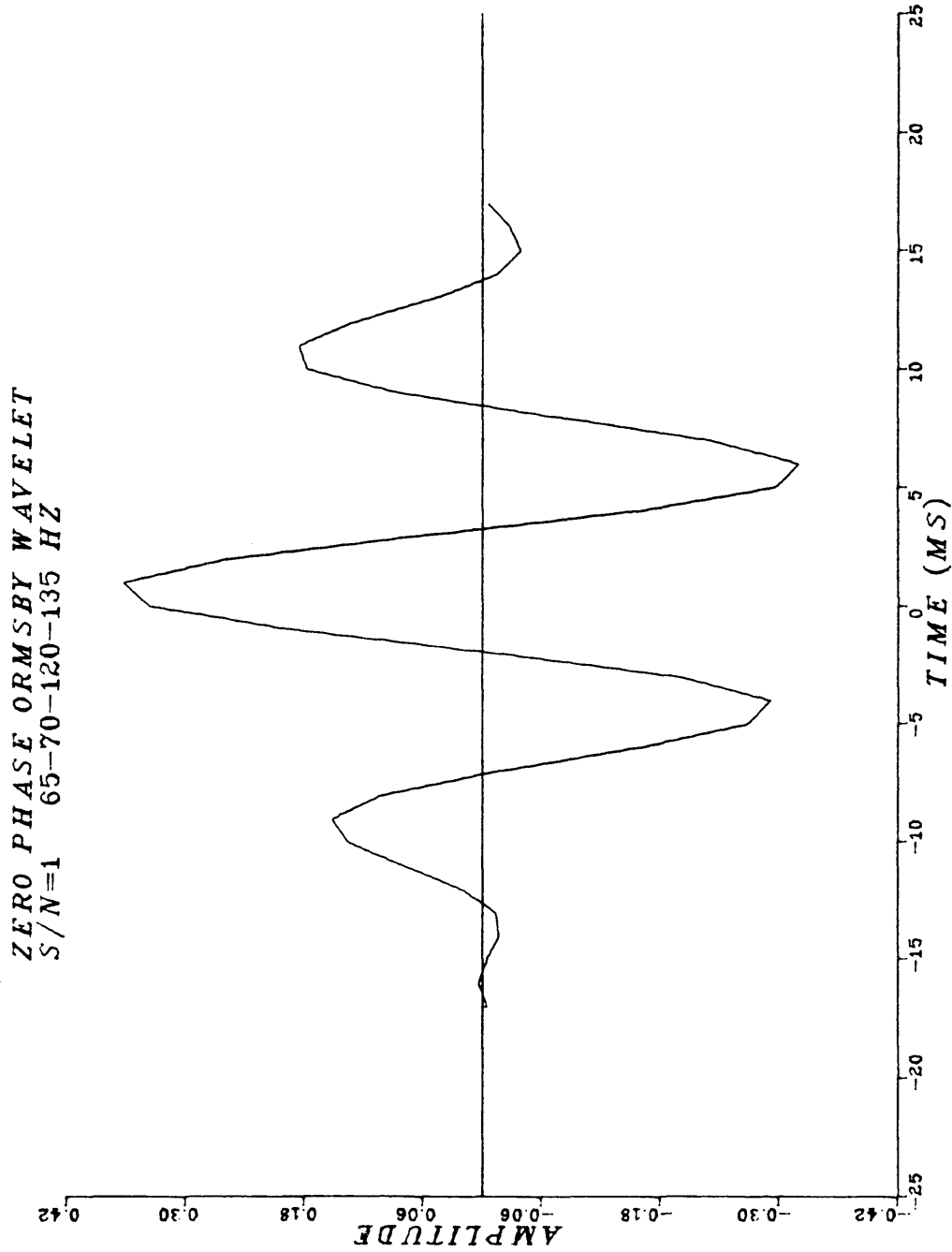


Figure 2-59 Convolutional wavelet with S/N = 1.

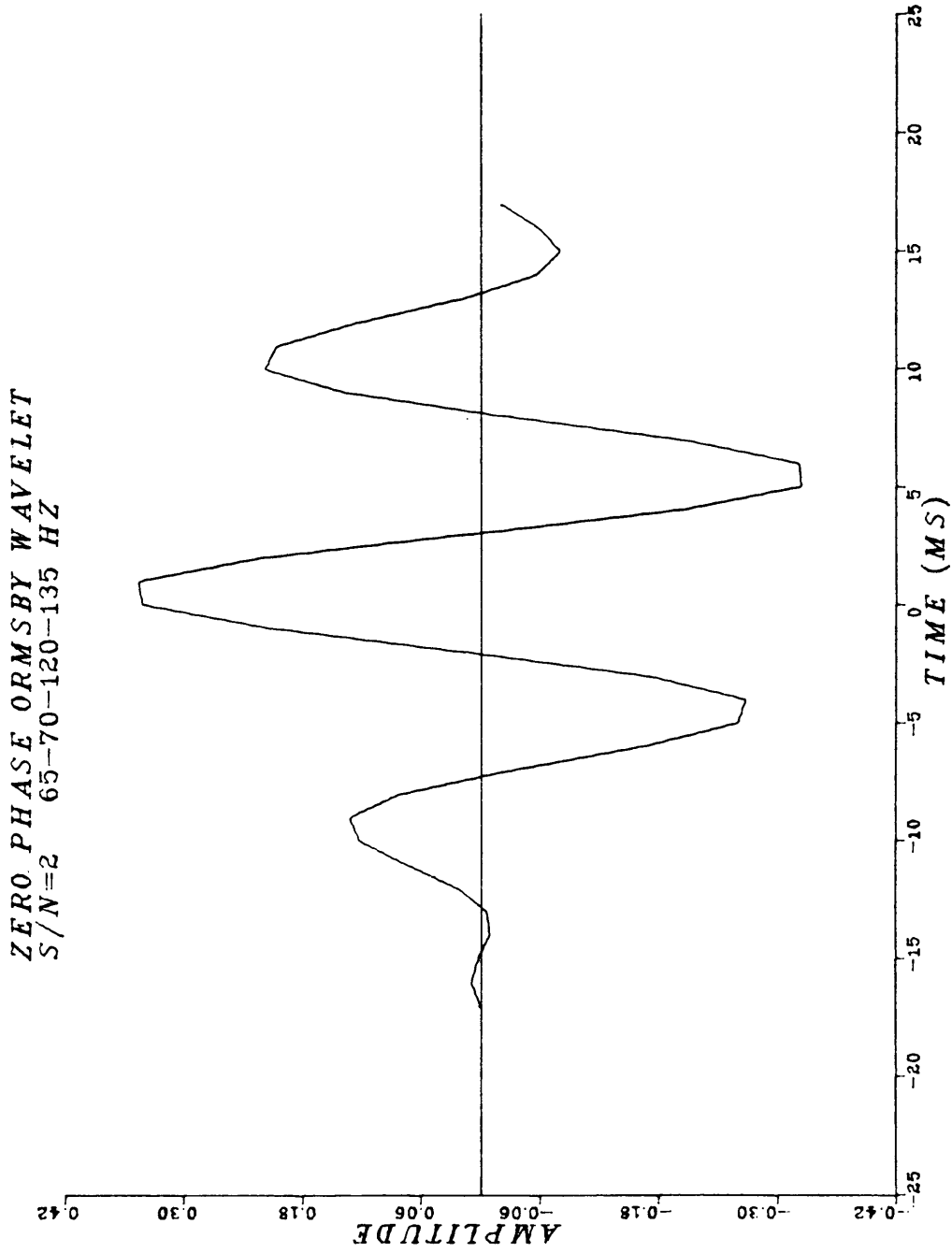


Figure 2-60 Convolutional wavelet with S/N = 2.

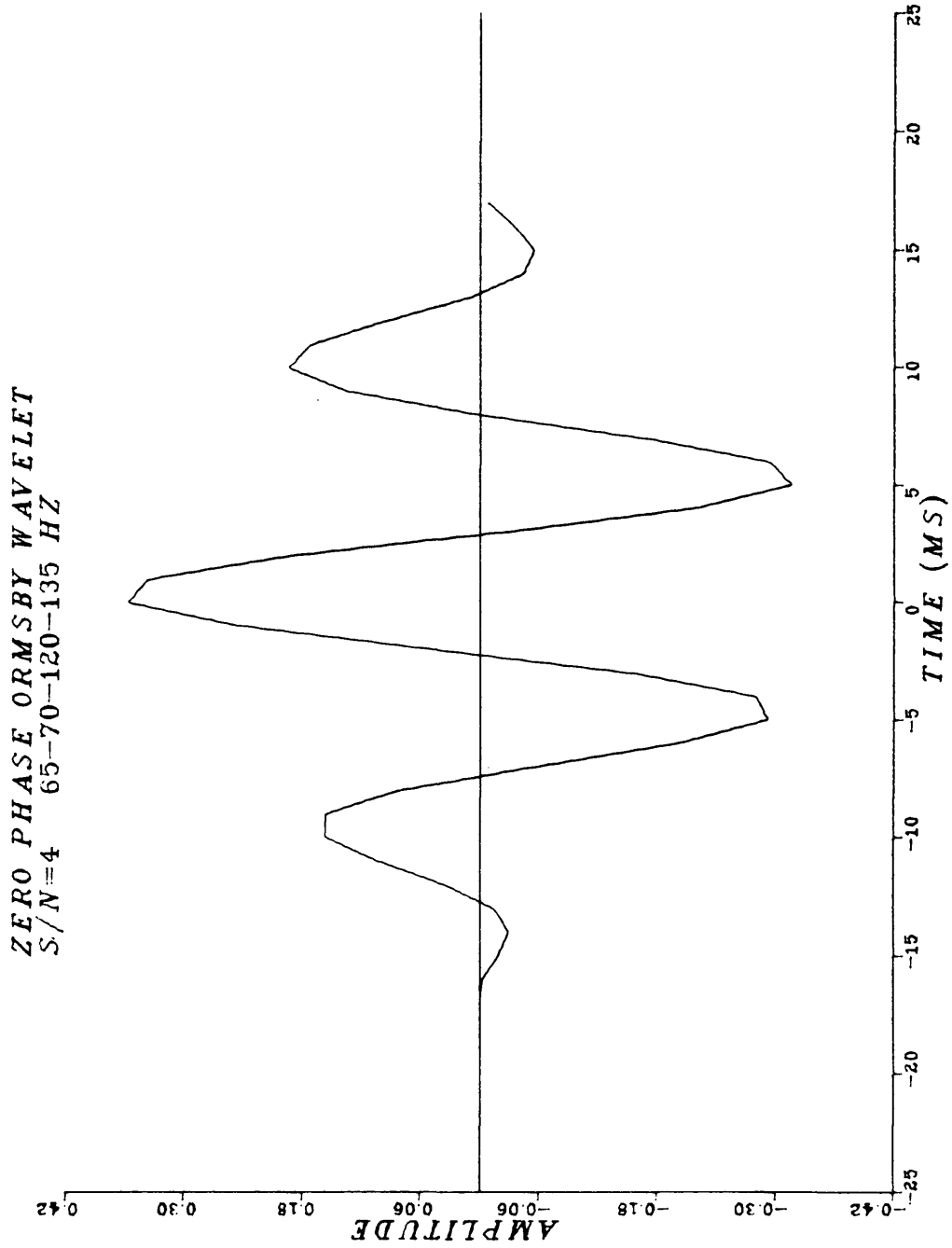


Figure 2-61 Convolutional wavelet with S/N = 4.

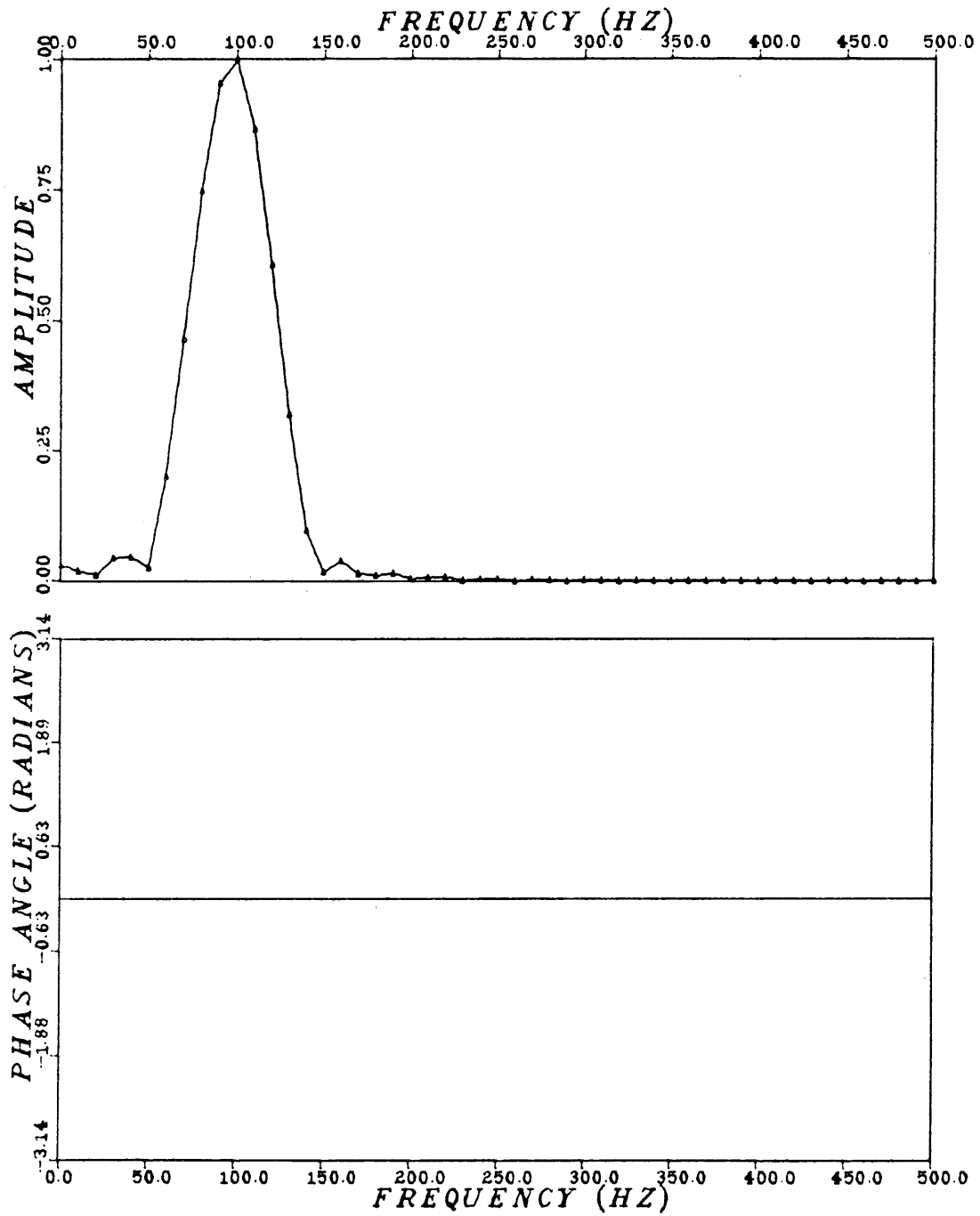


Figure 2-62 Amplitude and frequency spectrums for wavelet with zero noise.

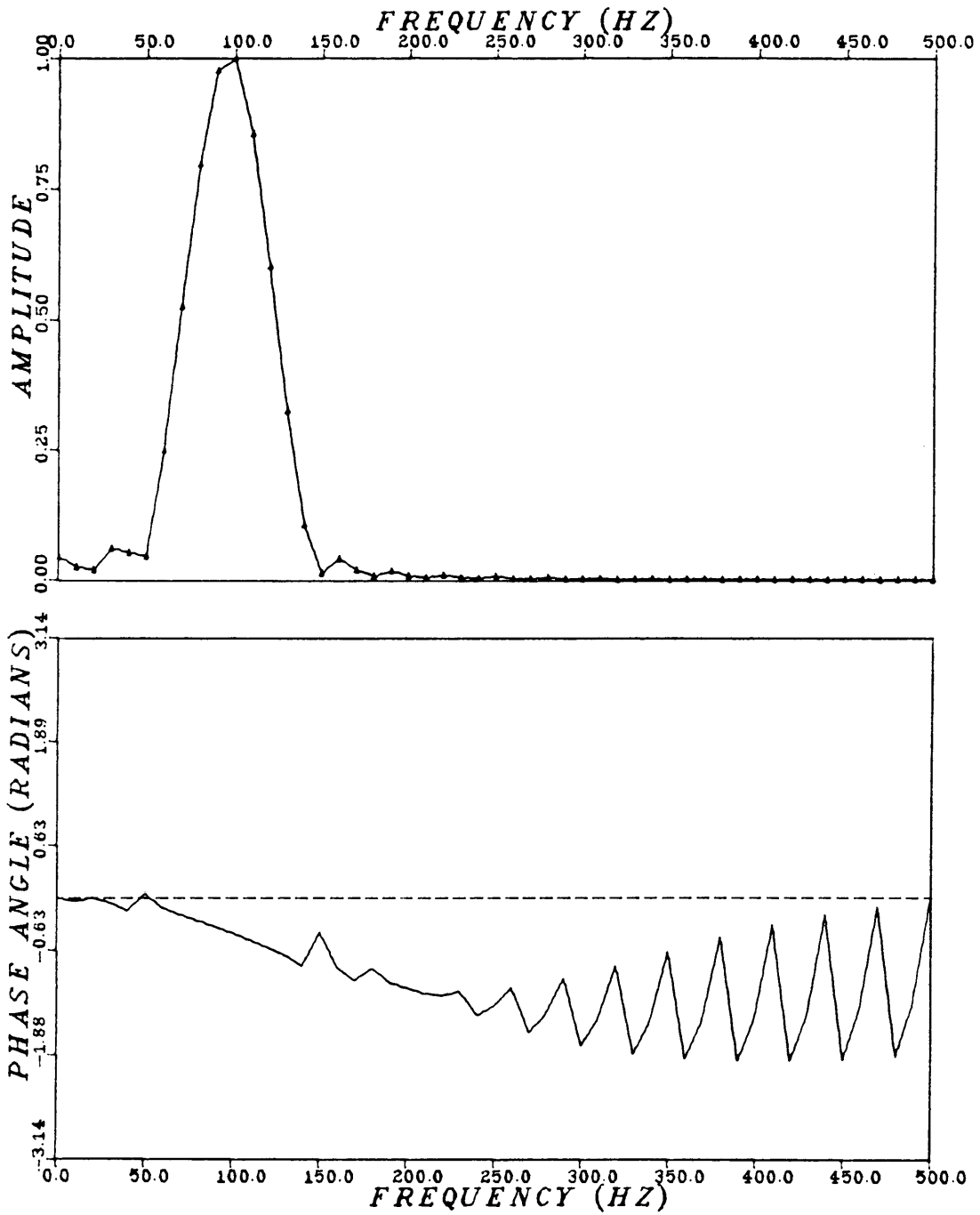


Figure 2-63 Amplitude and frequency spectrums for wavelet with S/N = 1.

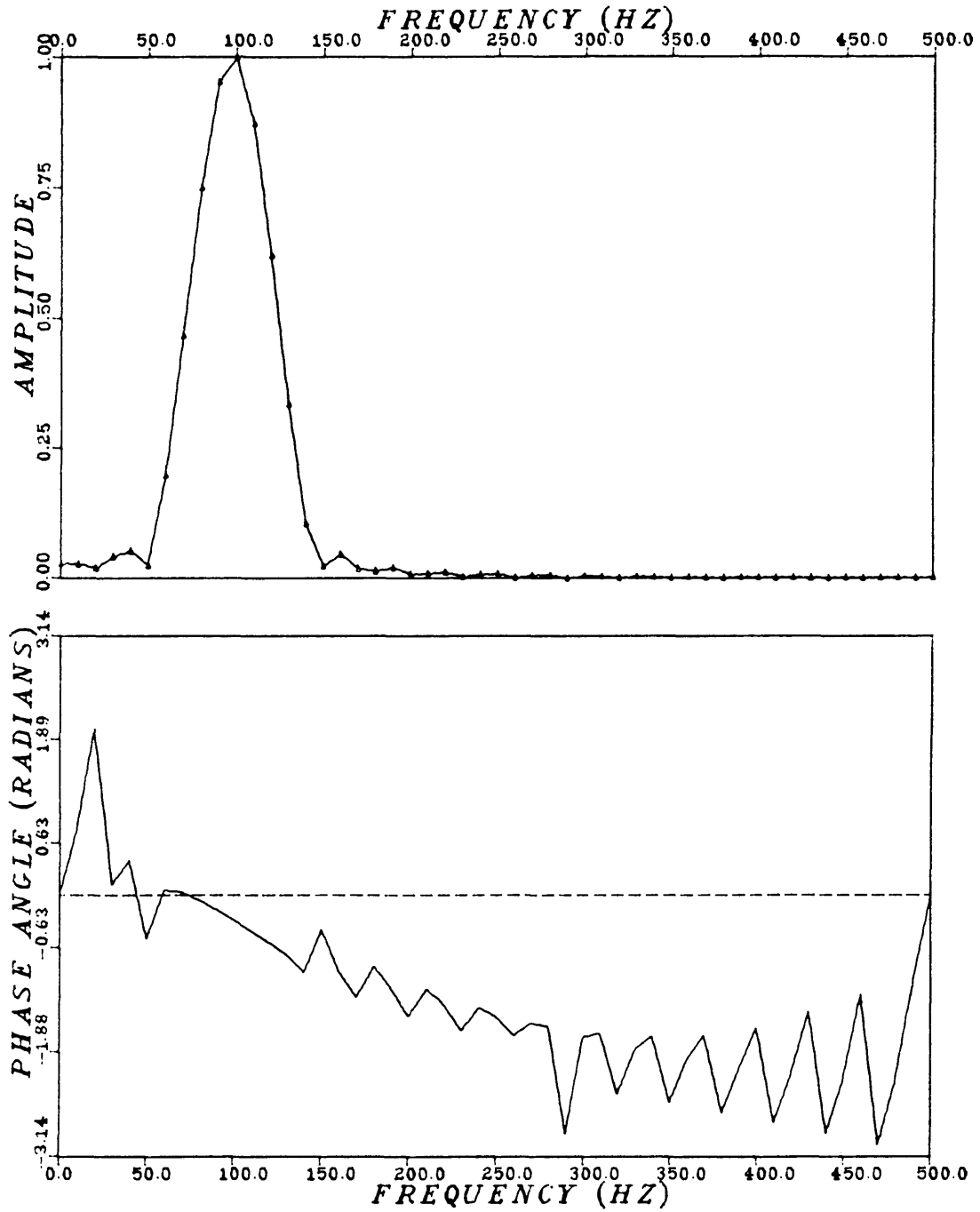


Figure 2-64 Amplitude and frequency spectrums for wavelet with S/N = 2.

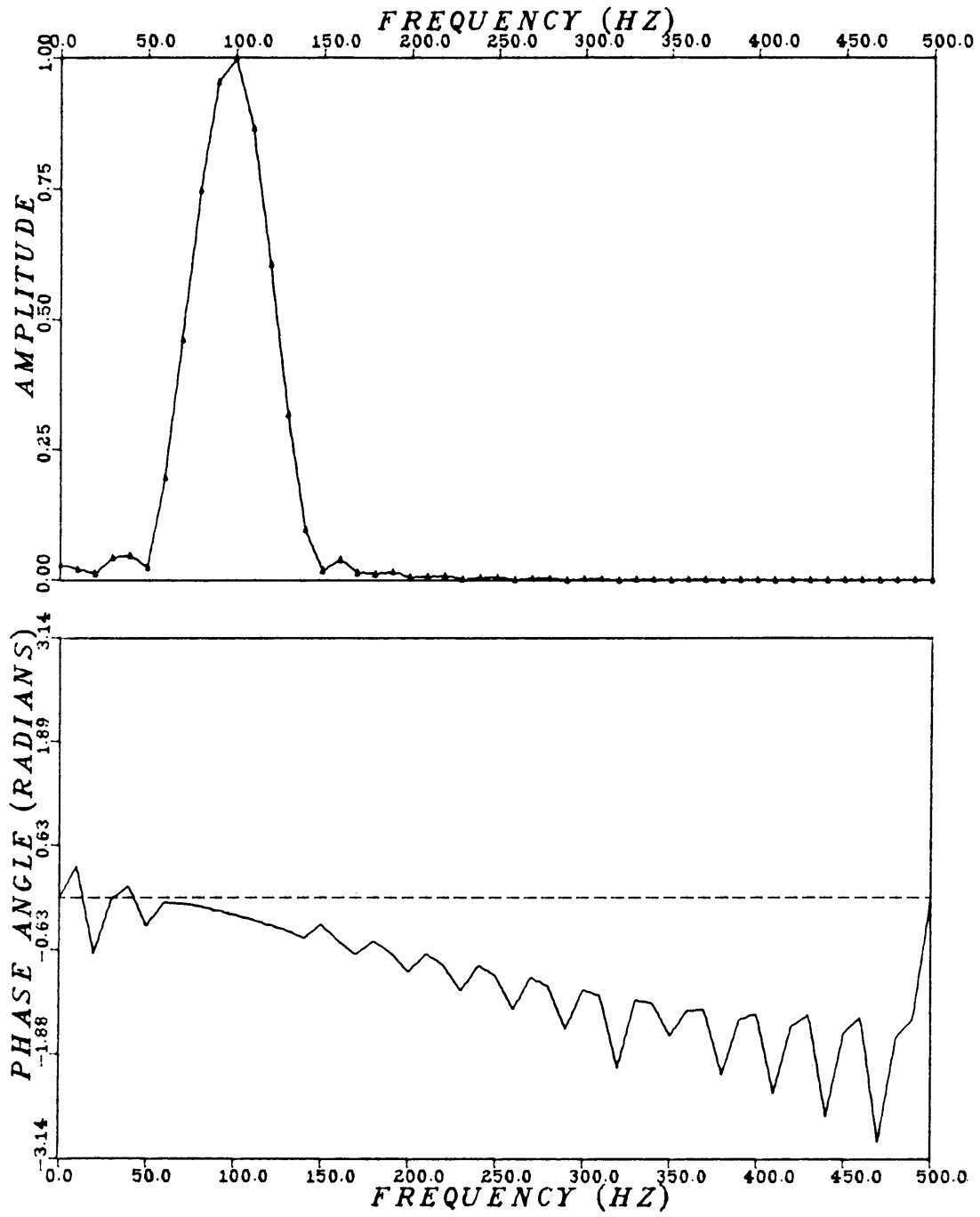


Figure 2-65 Amplitude and frequency spectrums for wavelet with S/N = 4.

convolved with our model to give the observed trace. The inversion scheme will then accept a second input wavelet that we say represents the source wavelet. If this second wavelet is identical to our first wavelet, the inversion scheme is again exact. However, if it is distorted, this simulates incorrect or incomplete knowledge of the source wavelet and errors result. Figures 2-58 through 2-61 illustrate the four wavelets used for testing purposes while 2-62 through 2-65 are their respective amplitude and frequency spectrums. Figure 2-58 represents the undistorted wavelet while 2-59 through 2-61 are corrupted wavelets with varying amounts of distortion.

The results are shown in figures 2-66 through 2-77. The distortions in the results of the inversion are apparent. The shift visible in the examples with the noisy wavelets can be explained by looking at the wavelets. Each is shifted about one time unit to later times. To match the observed and calculated traces, the program has only the calculated impedances to work with and accordingly shifts them to shallower times to offset the delayed wavelet.

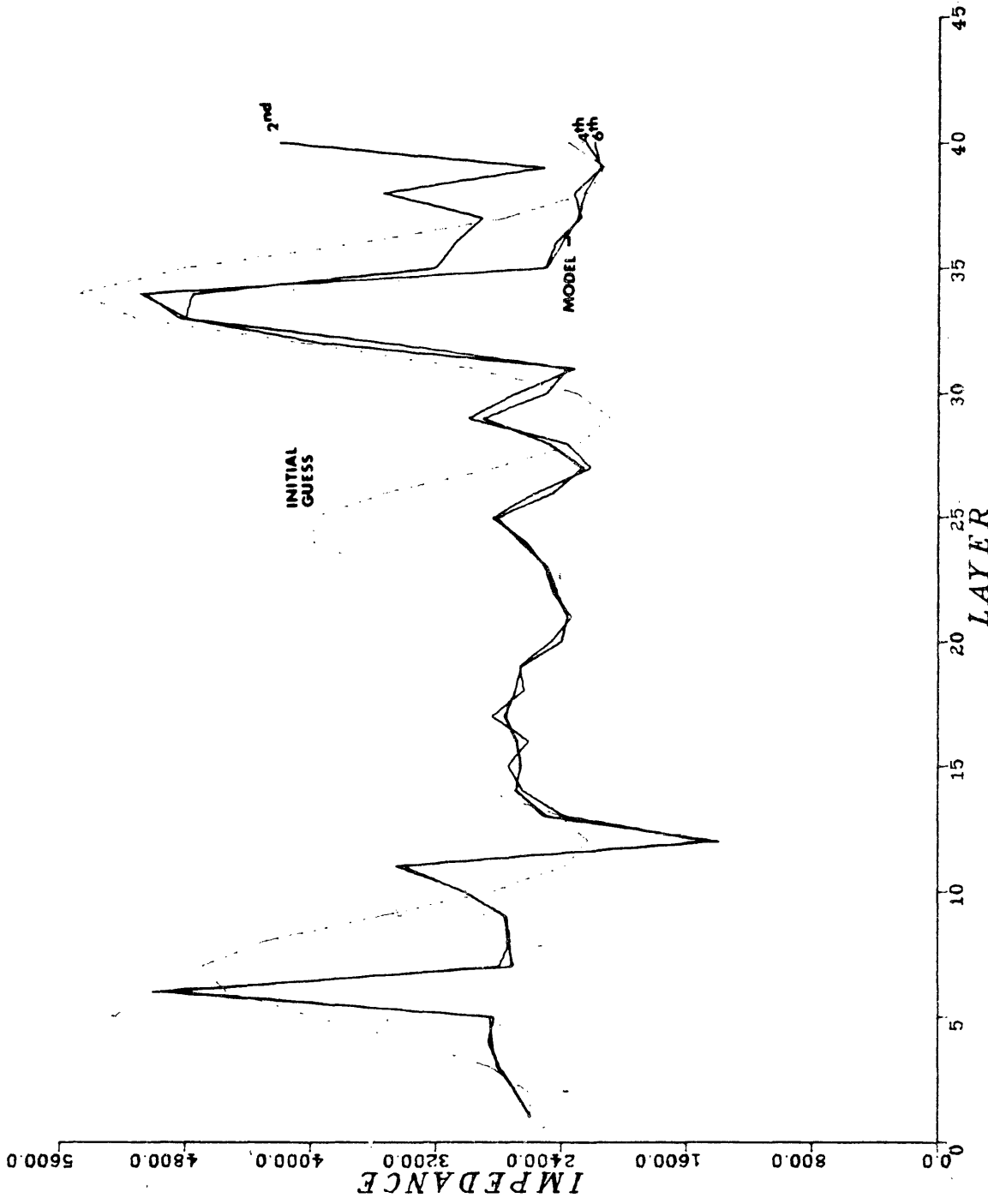


Figure 2-66 Inversion results for model 6 with uncorrupted convolutional wavelet.

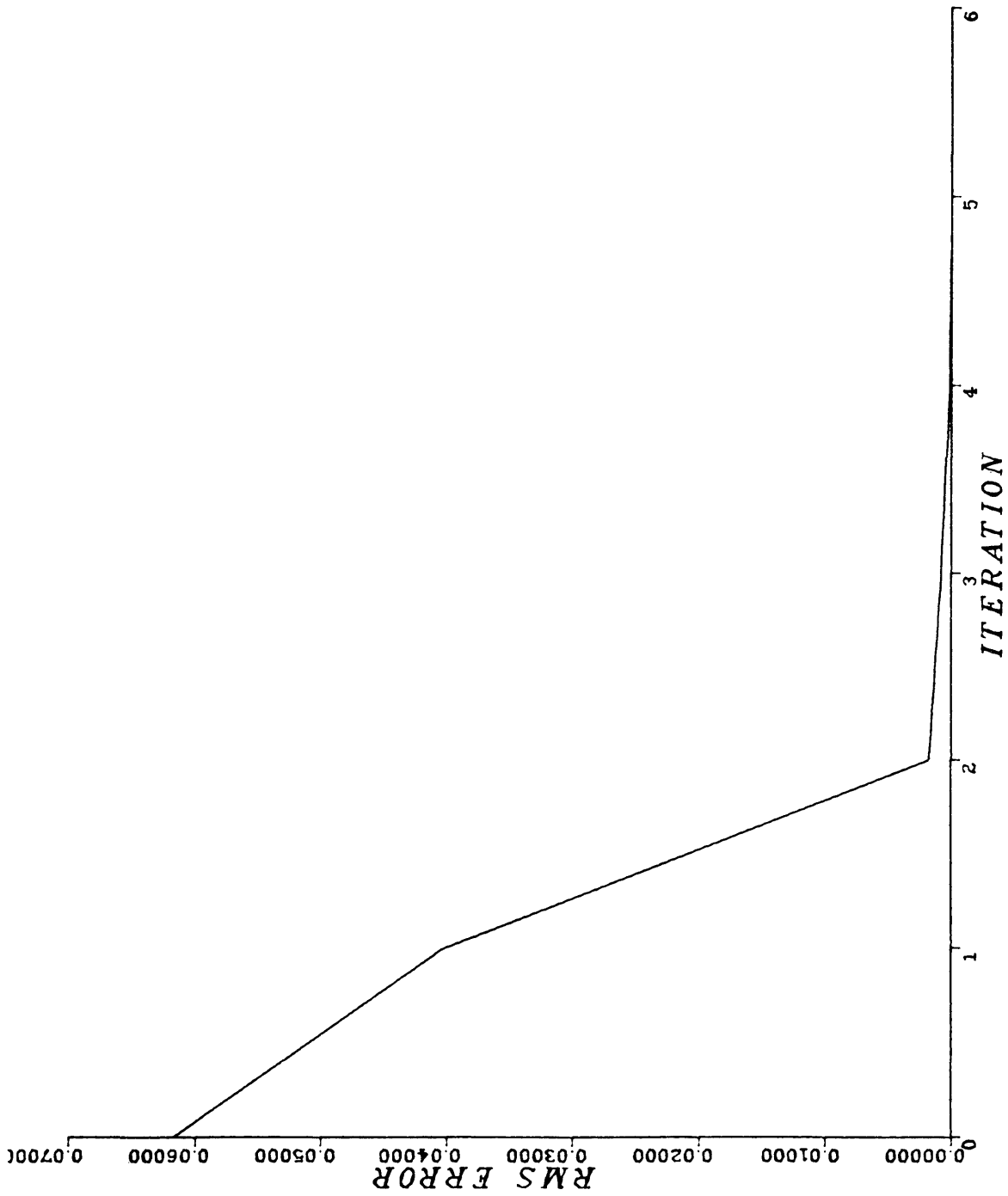


Figure 2-67 Model 6 RMS error for uncorrupted convolutional wavelet.

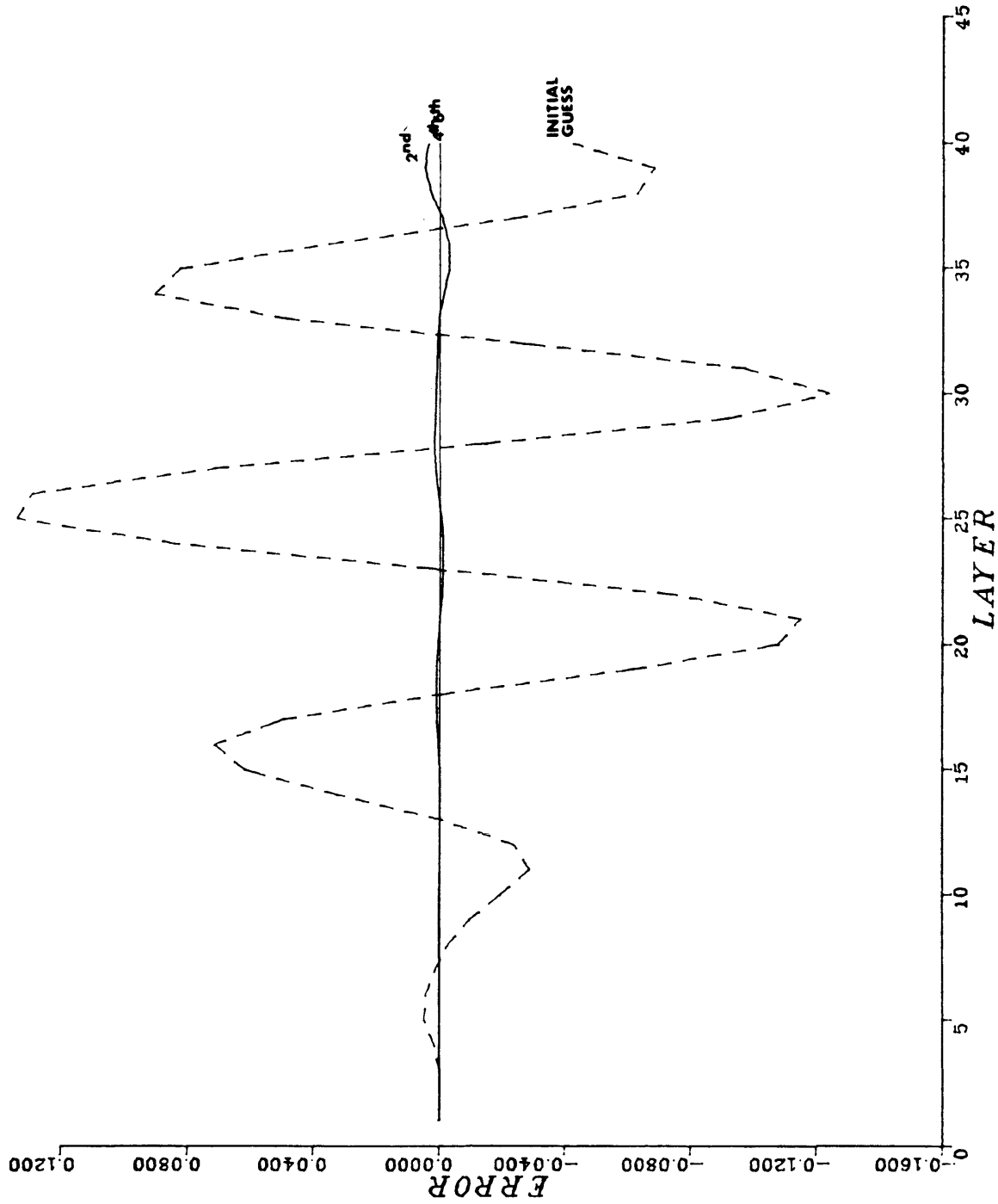


Figure 2-68 The observed minus calculated trace for successive iterations with model 6 and uncorrupted convolutional wavelet.

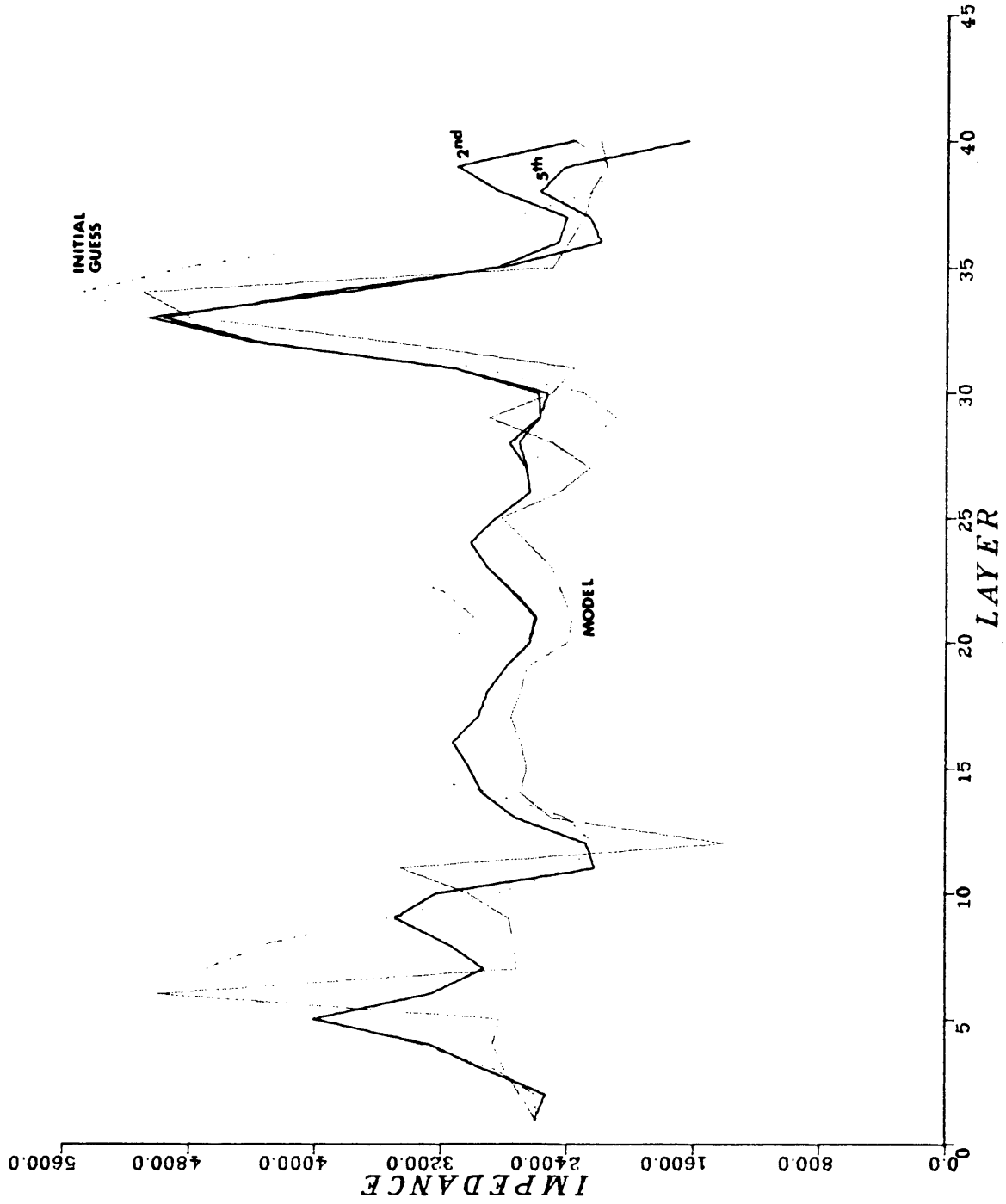


Figure 2-69 Inversion results for model 6 and uncorrupted convolutional wavelet at S/N = 1.

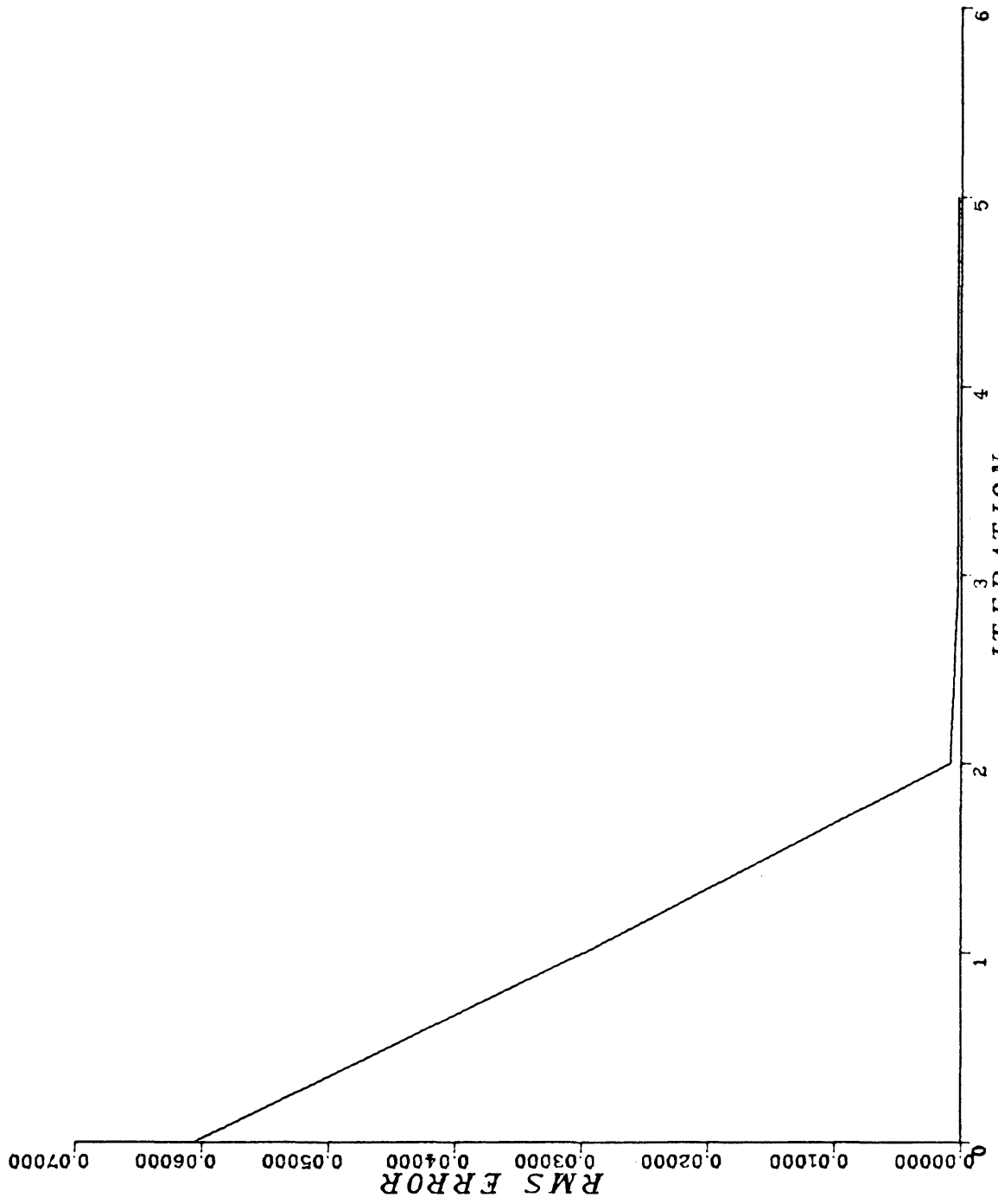


Figure 2-70 Model 6 RMS error for corrupted convolutional wavelet at $S/N = 1$.

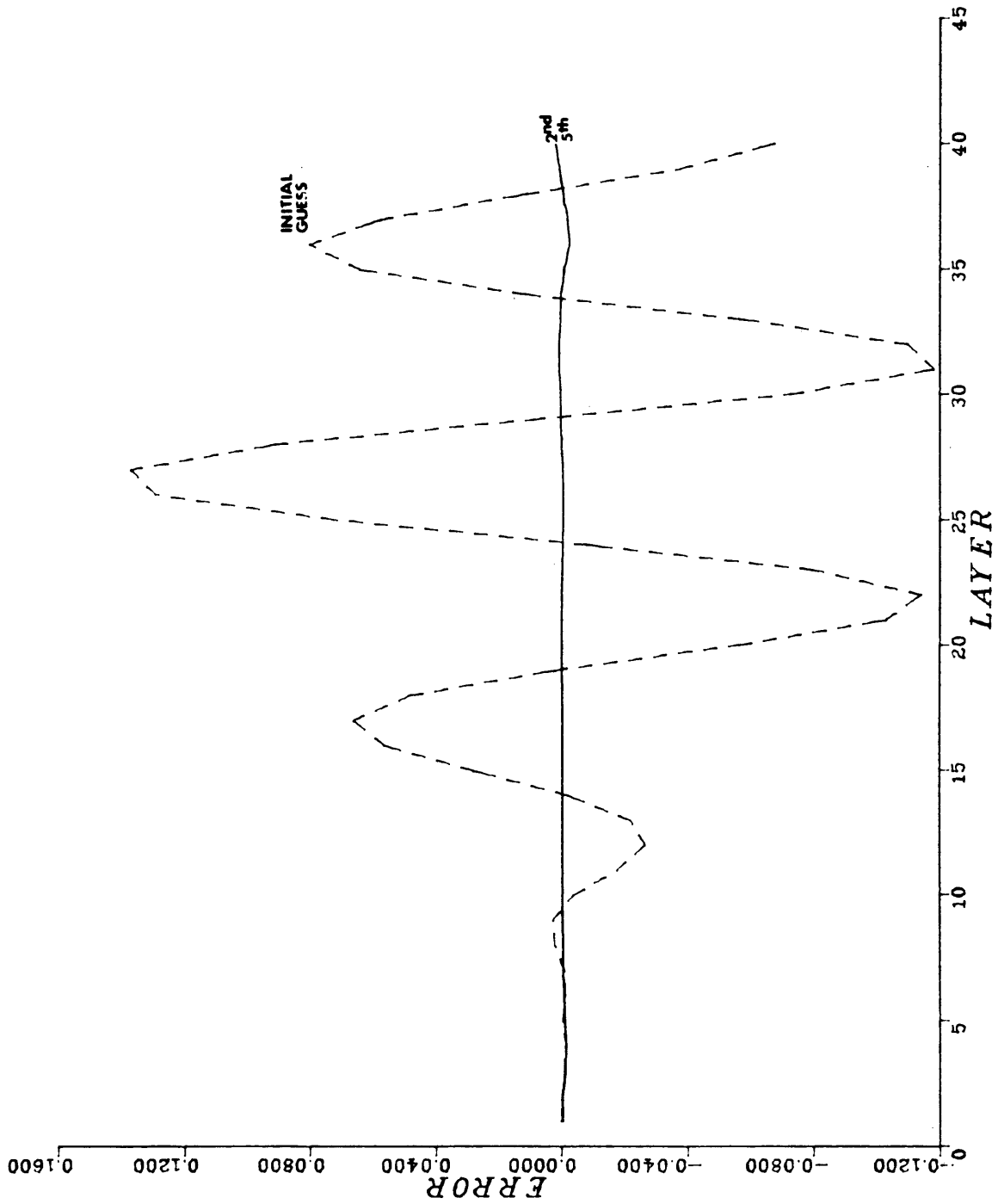


Figure 2-71 The observed minus calculated trace for successive iterations with model 6 and corrupted convolutional wavelet at S/N = 1.

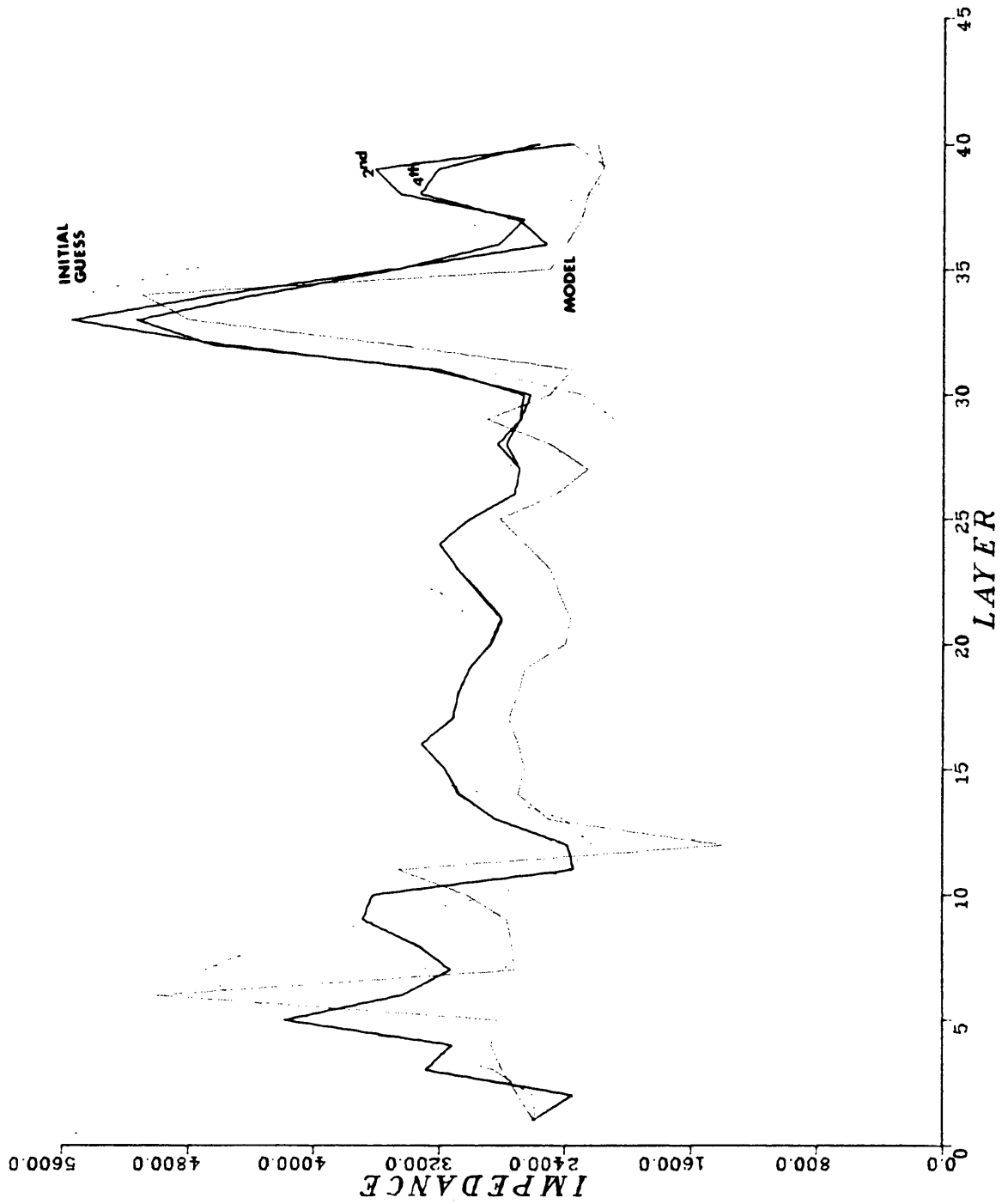


Figure 2-72 Inversion results for model 6 with corrupted convolutional wavelet at S/N = 2.

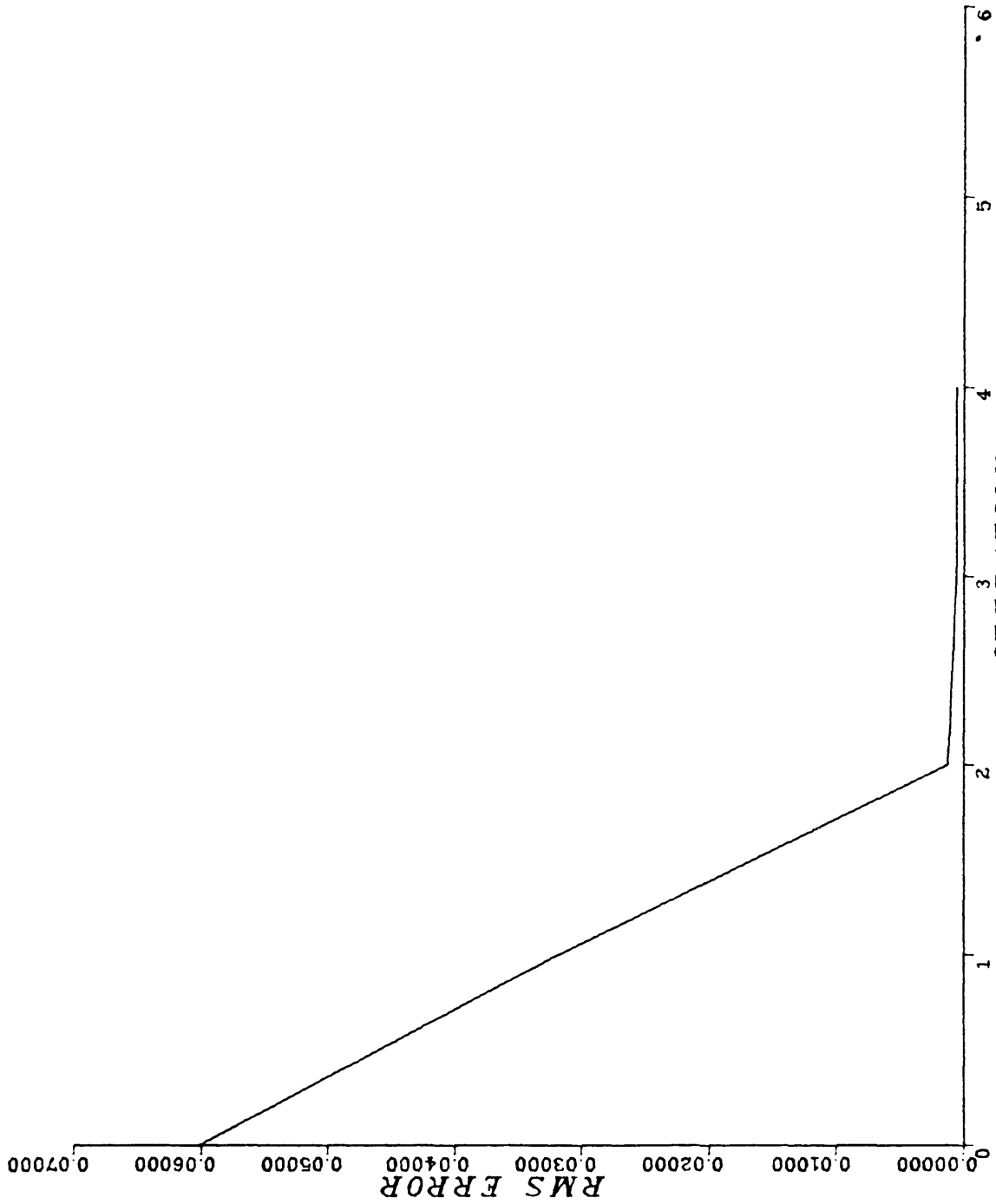


Figure 2-73 Model 6 RMS error for corrupted convolutional wavelet at $S/N = 2$.

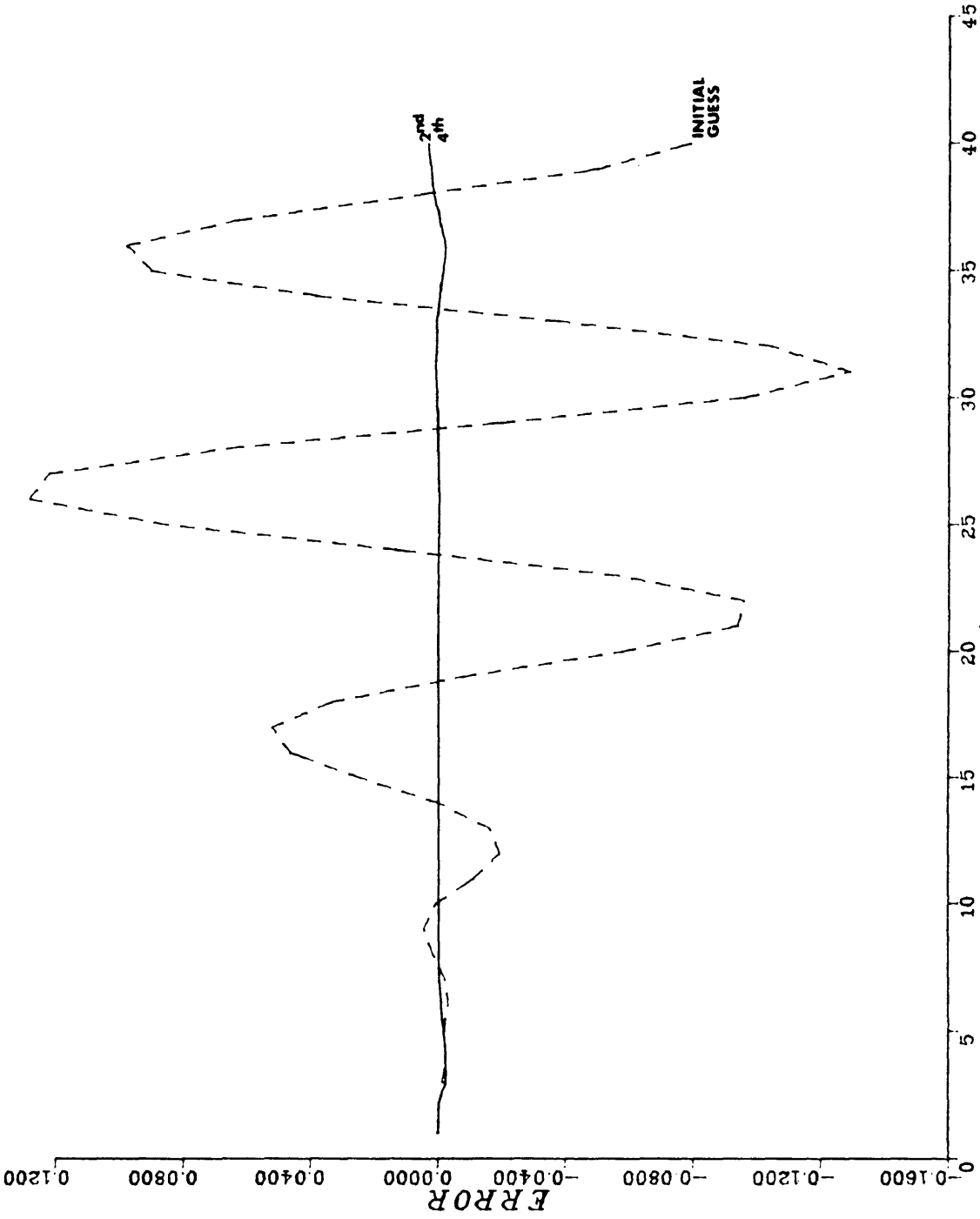


Figure 2-74 The observed minus calculated trace for successive iterations with model 6 and corrupted convolutional wavelet at $S/N = 2$.

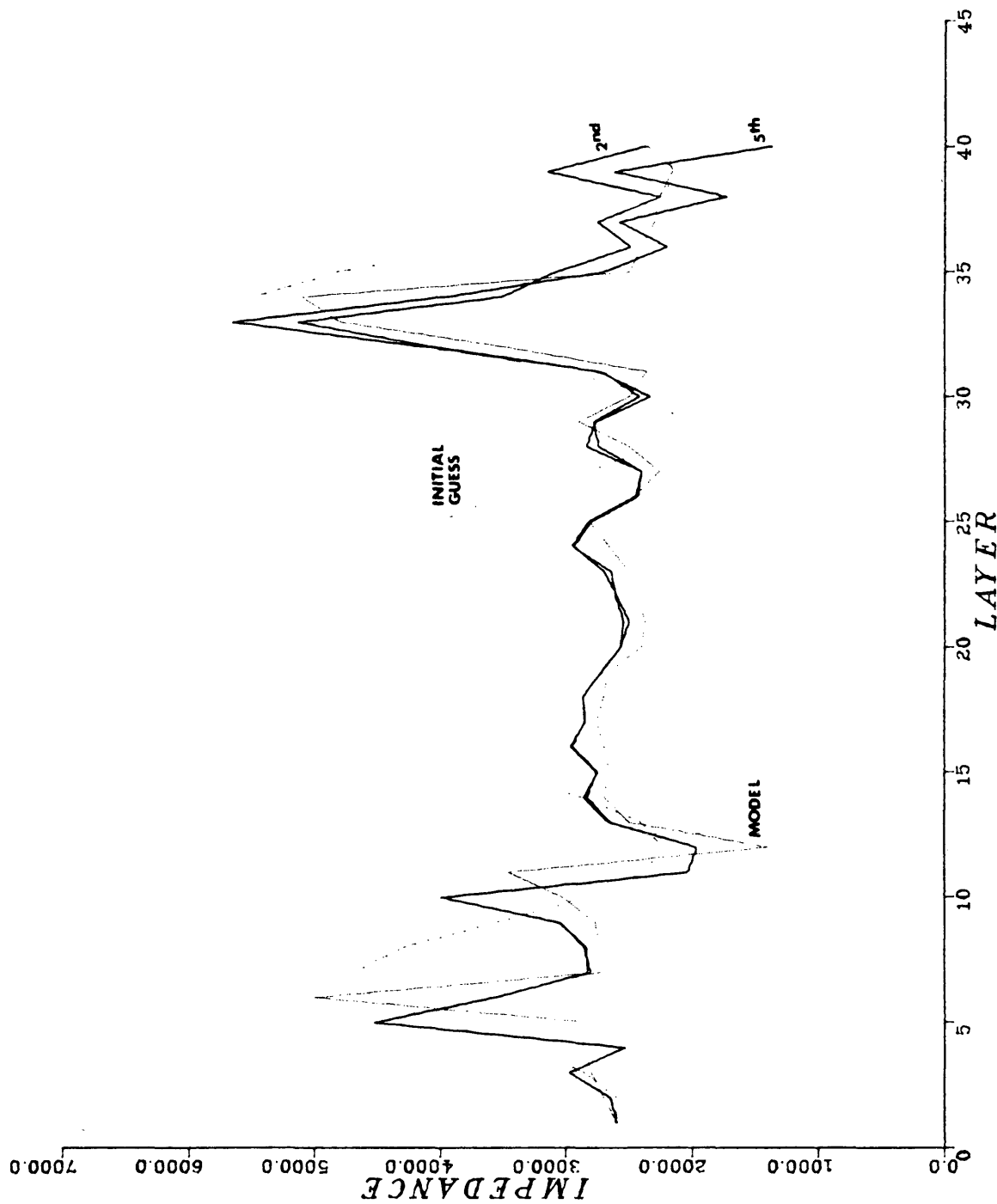


Figure 2-75 Inversion results for model 6 with corrupted convolutional wavelet S/N = 4.

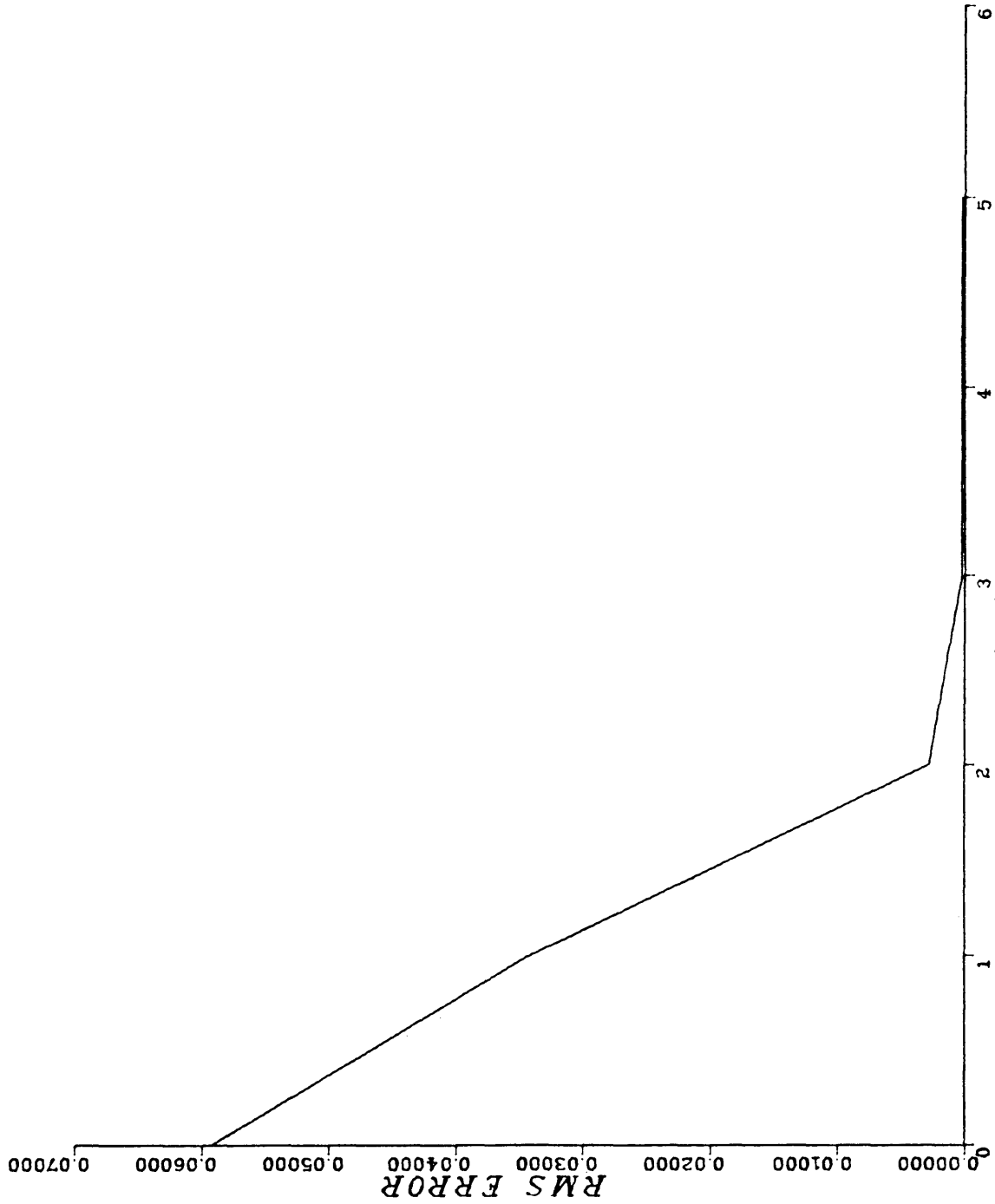


Figure 2-76 Model 6 RMS error for corrupted convolutional wavelet at $S/N = 4$.

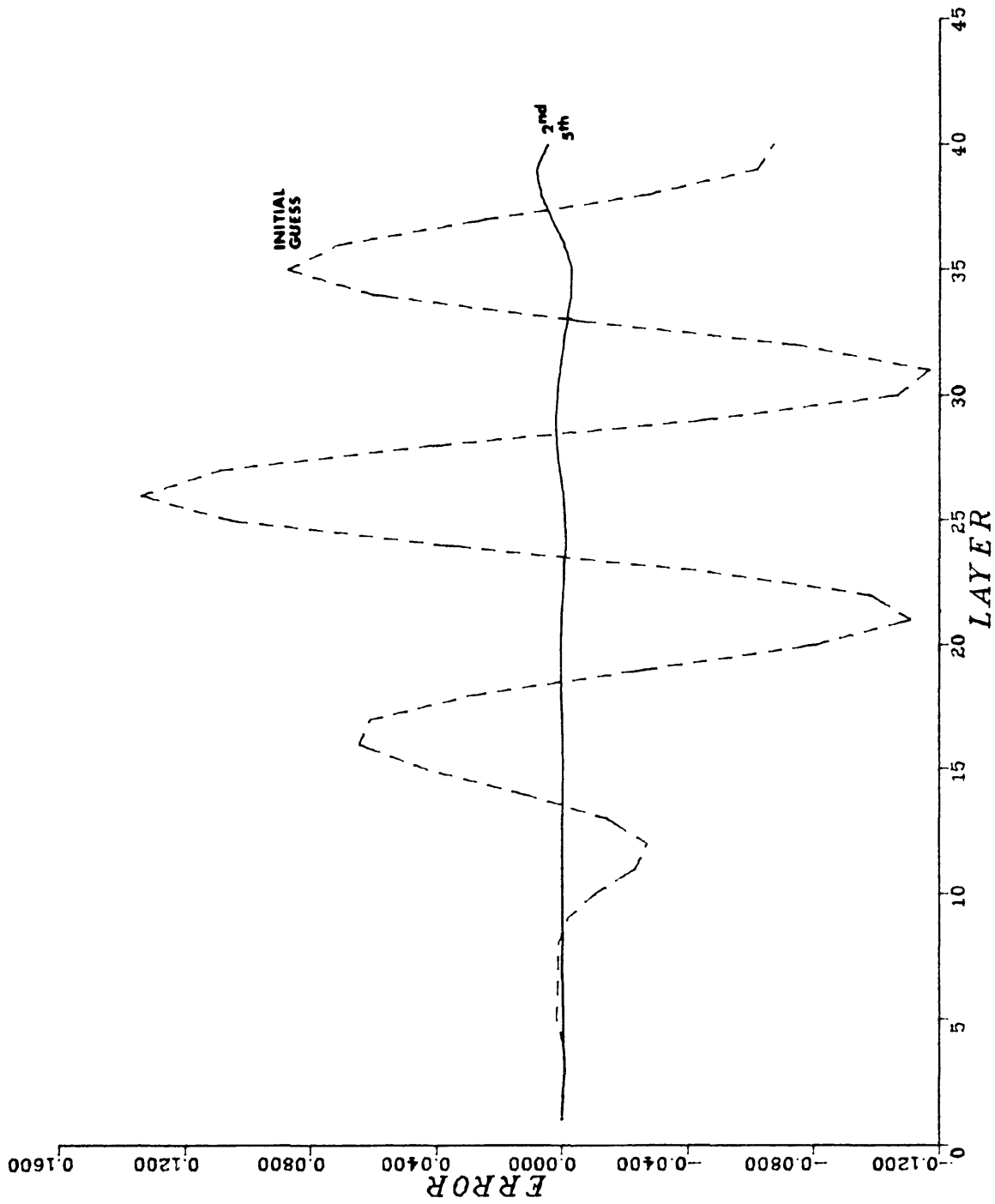


Figure 2-77 The observed minus calculated trace for successive iterations with model 6 and corrupted convolutional wavelet at S/N = 4.

The example here represents one of the more severe cases of distortion because of the wavelet length. In spite of this, the general trends are preserved and the errors are not exceedingly large. Stability can be a problem here, as the wavelet tends to strongly alter the partial derivative matrix. The statement made in a previous section about the partial derivatives varying slowly becomes less valid. Because of this, changes in the program such as increasing our increment in calculating the partial derivative matrix may enhance stability.

d) Scaling noise

Finally, we need to consider the effects of scaling problems. Unless certain reflection coefficients are known or impedances are known for certain layers, it is difficult to be sure the observed trace we have really represents a properly scaled reflectivity series. For this situation, three cases were considered.

Case 1 considers the situation where the entire trace is scaled by a constant other than one. Case 2 has the scaling factor changing linearly with the depth and case 3 has it changing exponentially with depth.

For the case of scaling, unlike random noise or convolutional noise, we can easily add another model parameter to our problem; the scaling factor. Up until now, all the model parameters we were trying to determine were impedances. The inversion technique however has no barriers which exclude adding the effects of other parameters. This is one of the powerful aspects of this approach to inversion. We merely need to add another column to the partial derivative matrix in equation 8 and extend $\Delta \vec{p}$ and \vec{d} one unit. This is automatically done by the program, and the results are interesting.

For case 1, the method converges exactly for a constant scaling factor of .6 and .8. The rate of convergence is slowed and not always constant. For cases 2 and 3, the number of iterations for convergence approaches eight and for one linear scaling factor, even this does not suffice. For the cases tested, it appears that although convergence occurs, its rate is considerably slowed and the scaling factor parameter is very sensitive to change. For that reason, when the initial guesses for

the impedances are significantly larger or smaller than the model impedances, this may force the parameter correction for the scaling factor in the wrong direction. This was not a problem encountered with the impedance model parameters.

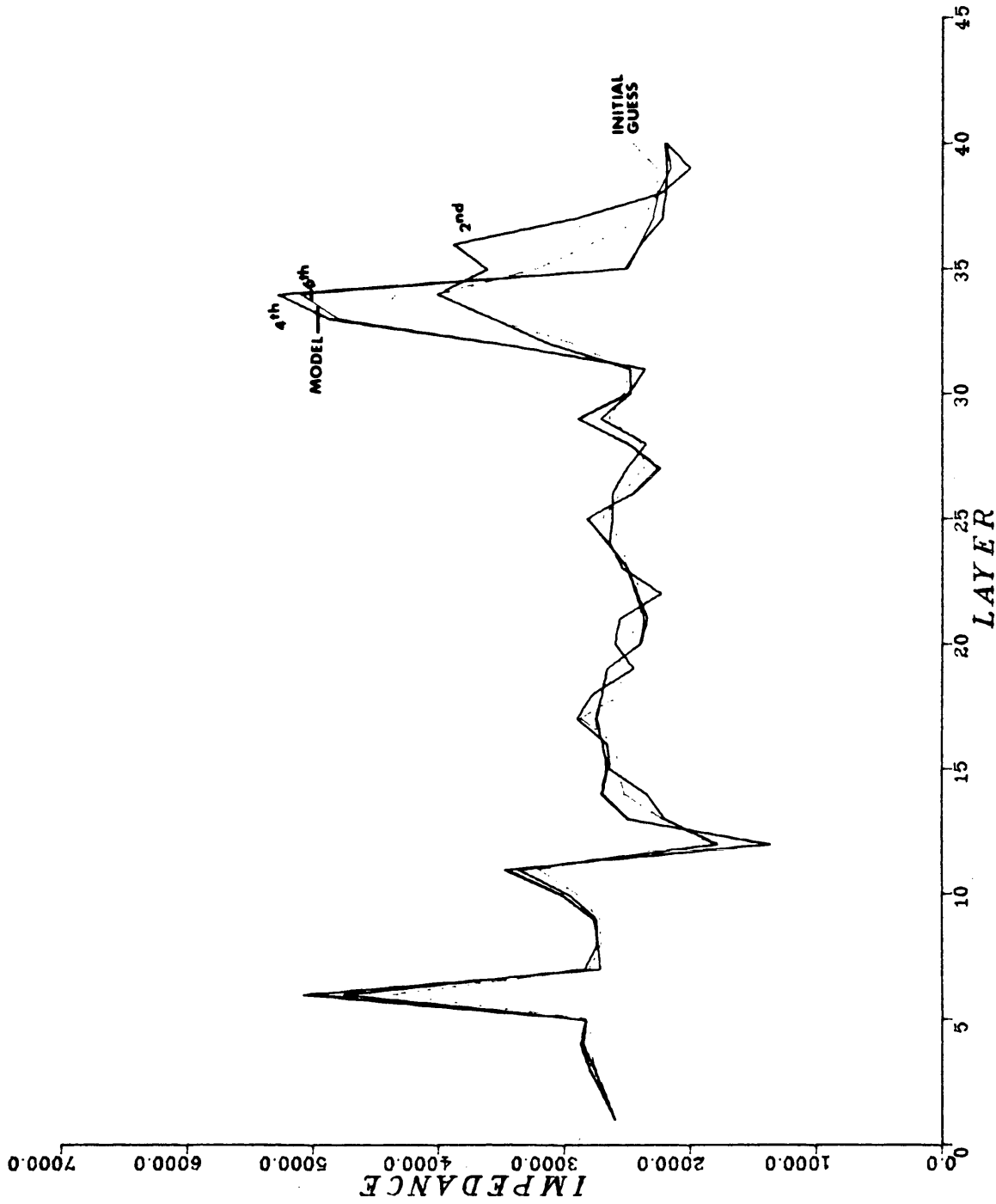


Figure 2-78 Inversion results for model 6 with .8 constant scaling factor.

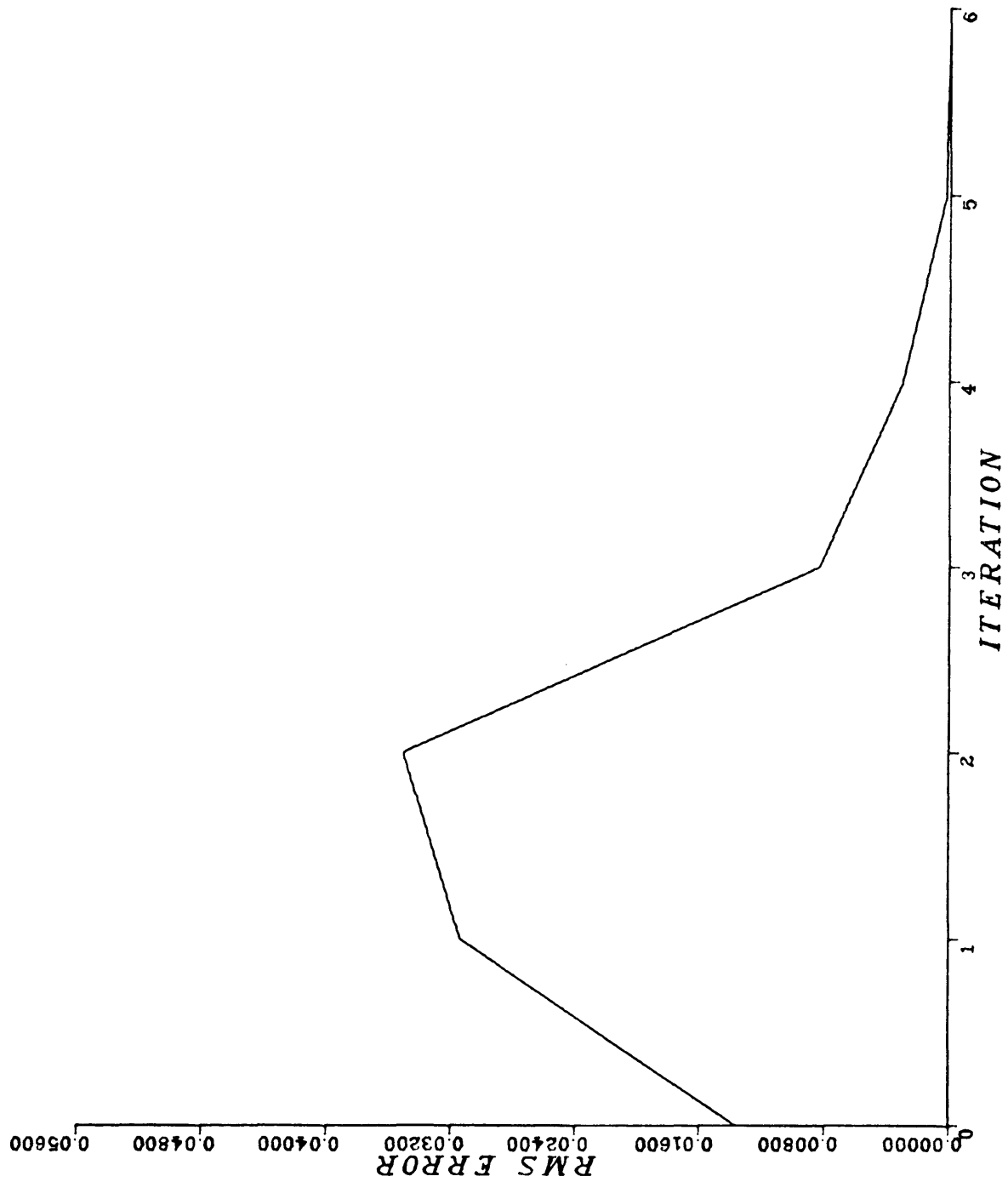


Figure 2-79 Model 6 RMS error for .8 constant scaling factor.

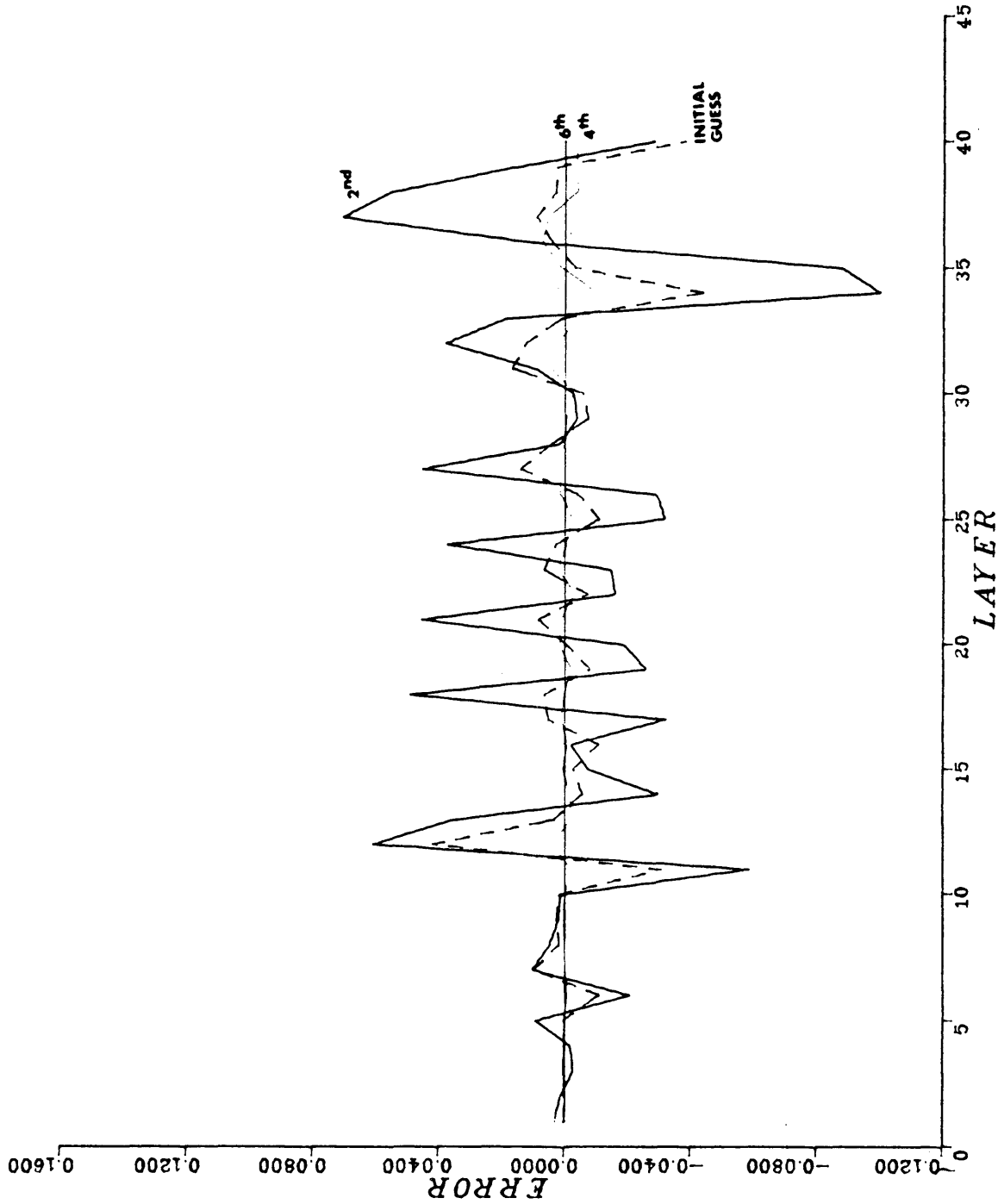


Figure 2-80 The observed minus calculated trace for successive iterations with model 6 and .8 constant scaling factor.

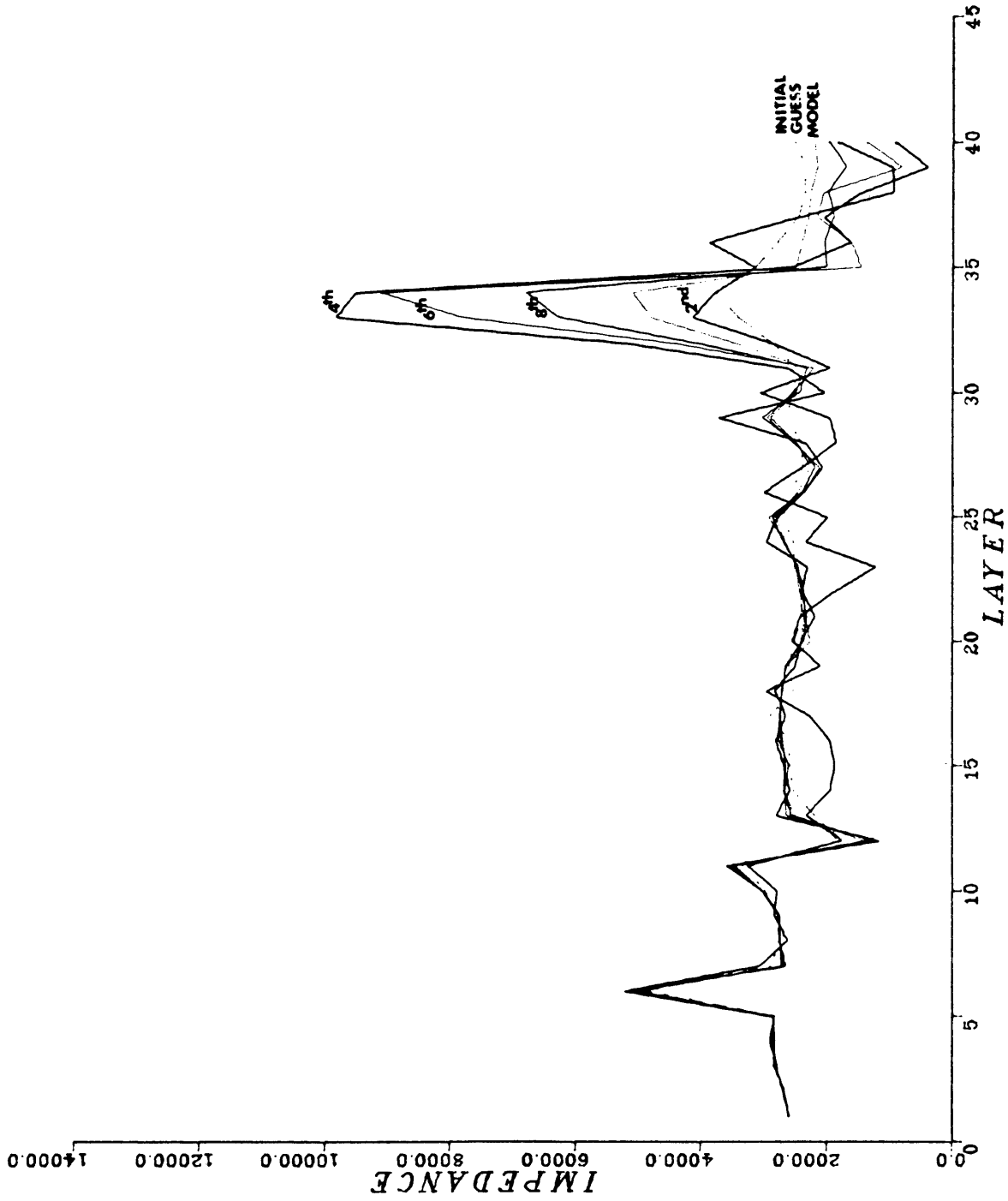


Figure 2-81 Inversion results for model 6 with -.01 linearly incremented scaling factor.

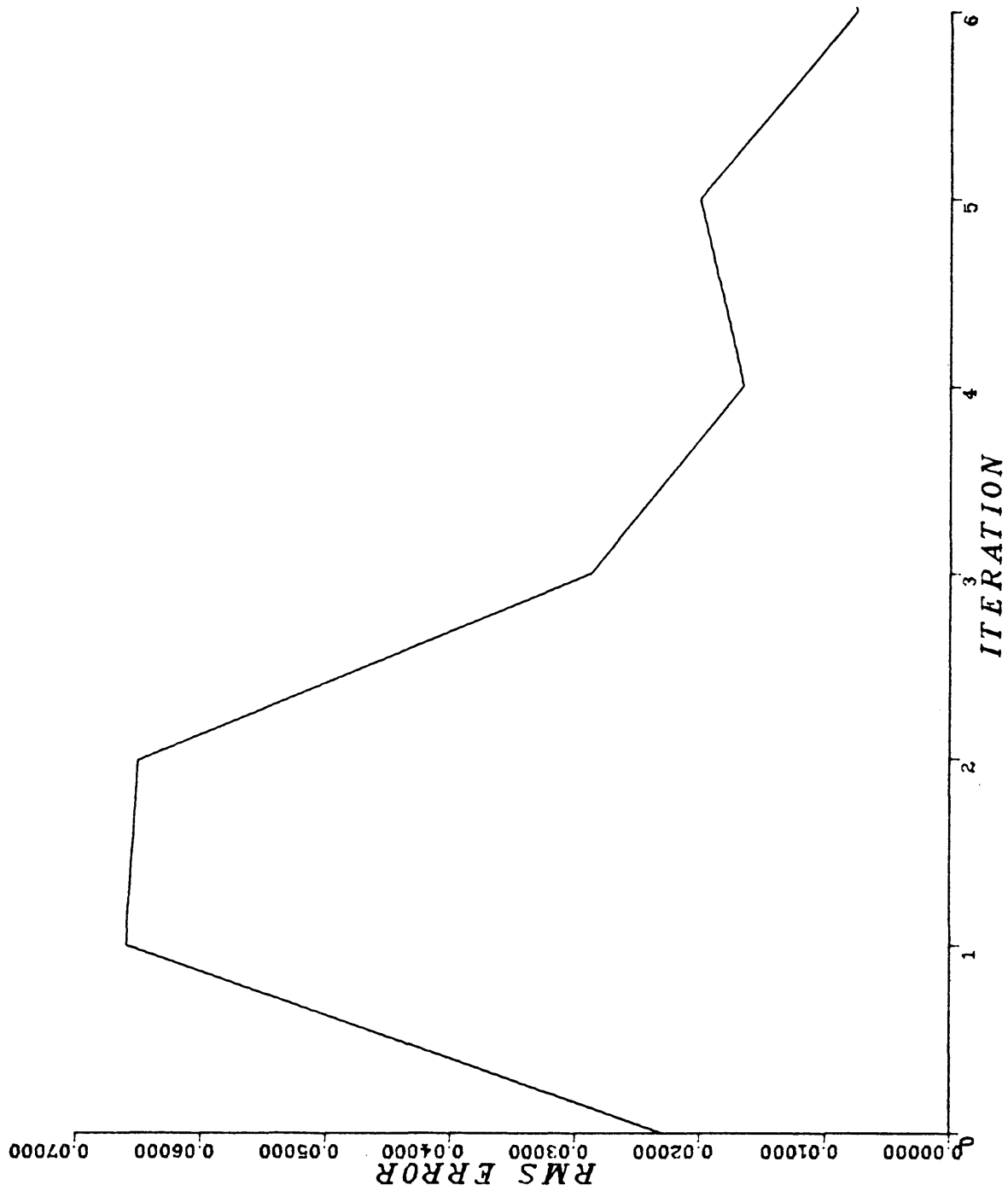


Figure 2-82 Model 6 RMS error for -.01 linearly incremented scaling factor.

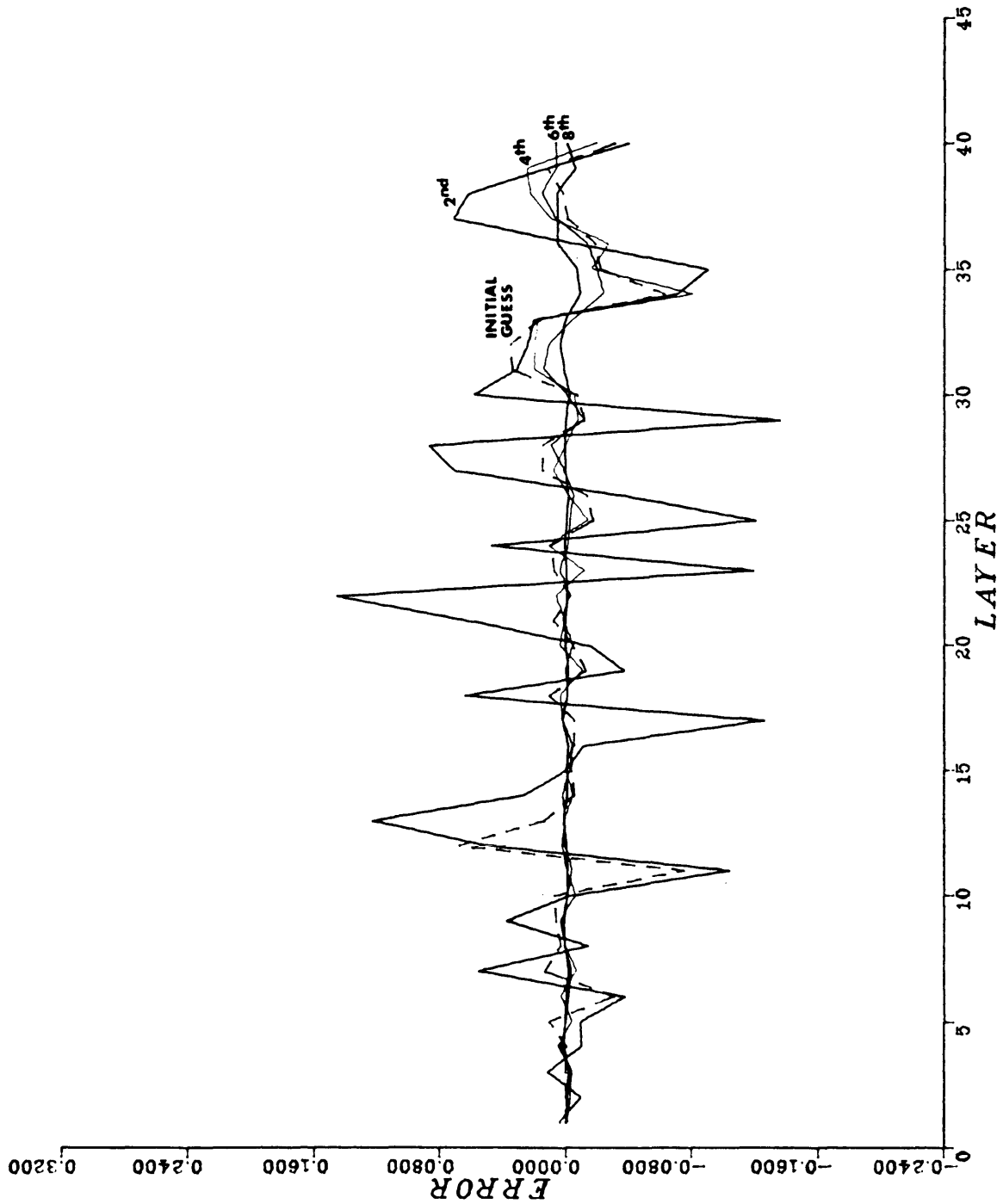


Figure 2-83 The observed minus calculated trace for successive iterations with model 6 and -.01 linearly incremented scaling factor.

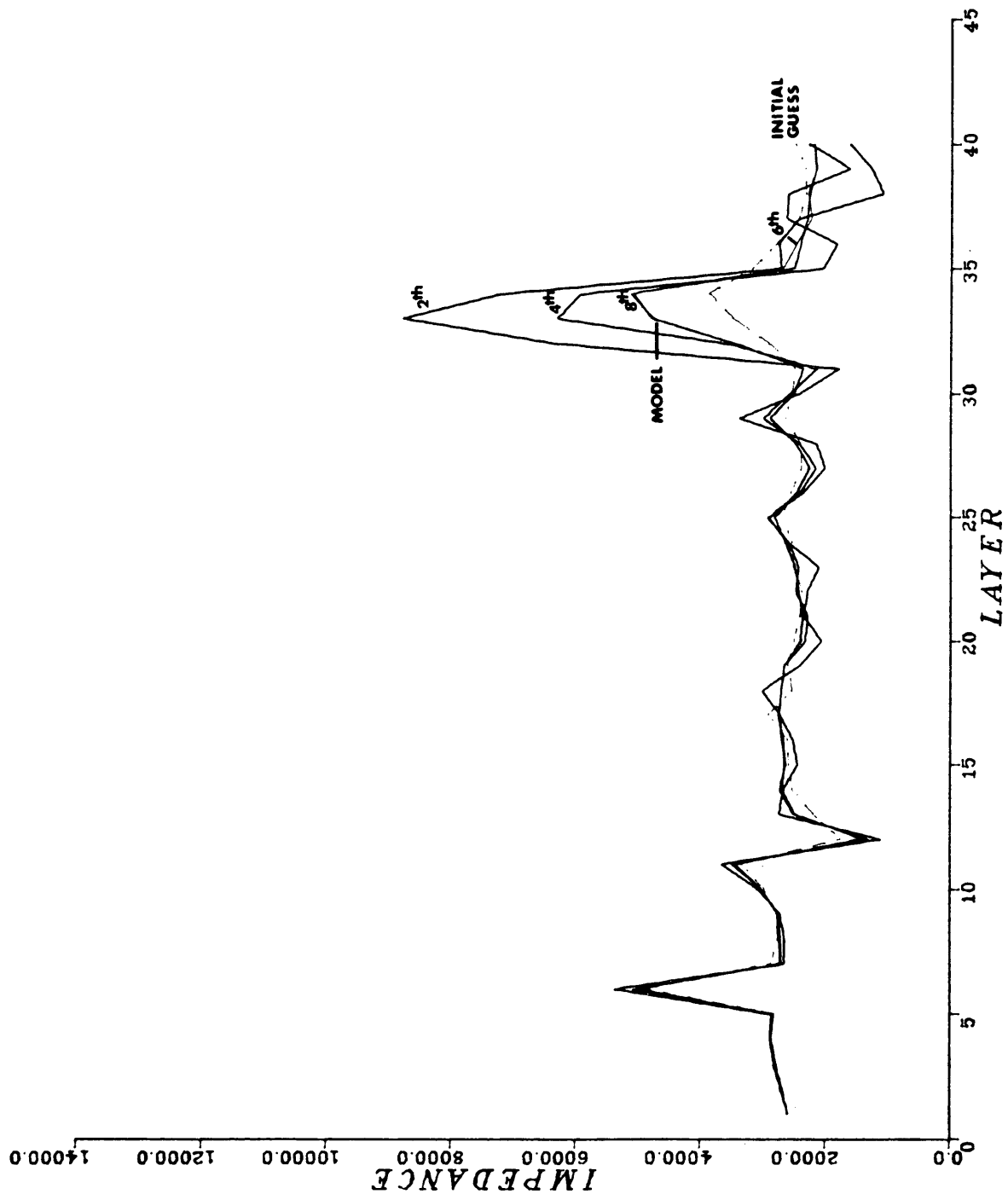


Figure 2-84 Inversion results for model 6 with $-.01$ exponentially incremented scaling factor.

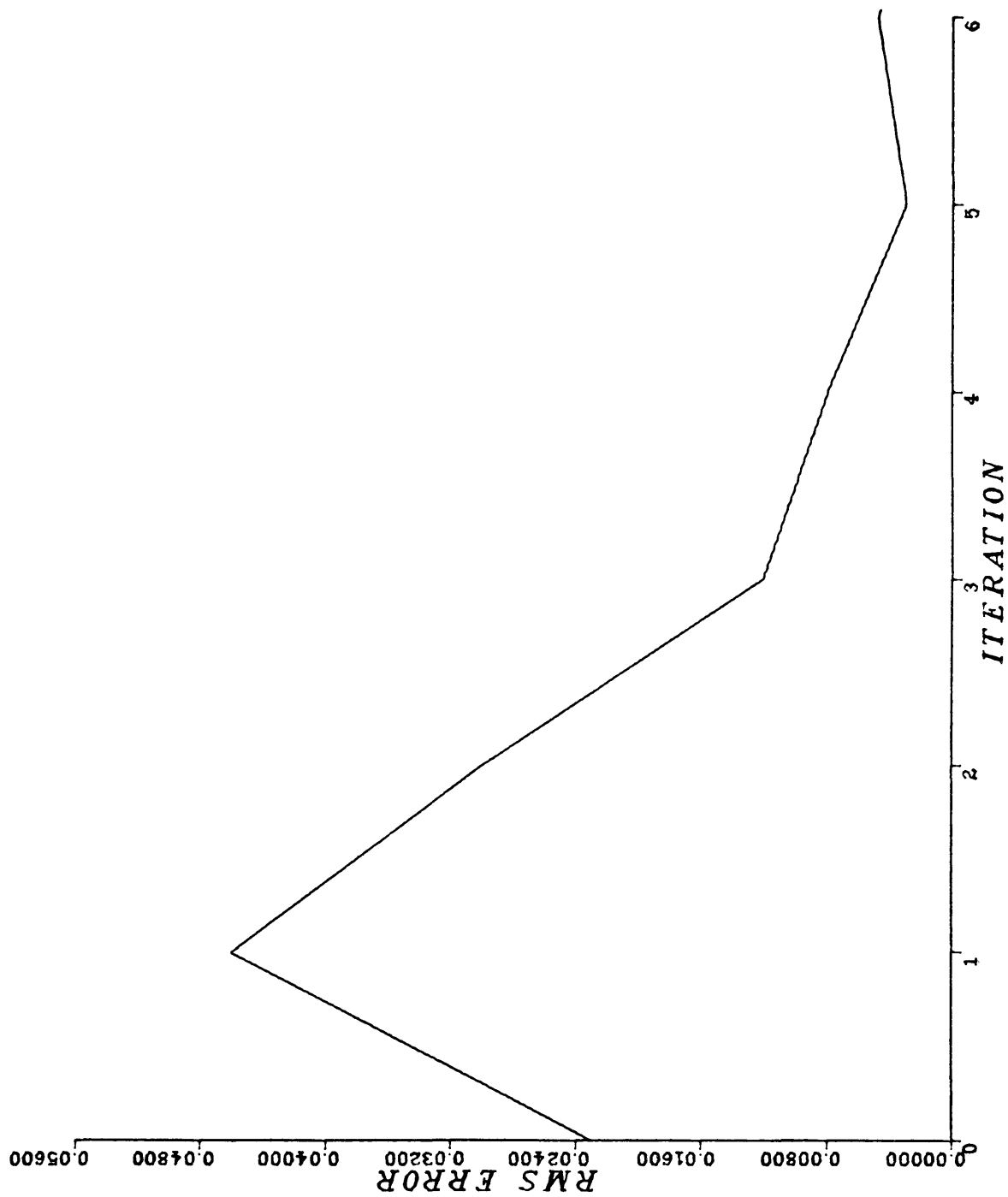


Figure 2-85 Model 6 RMS error for -.01 exponentially incremented scaling factor.

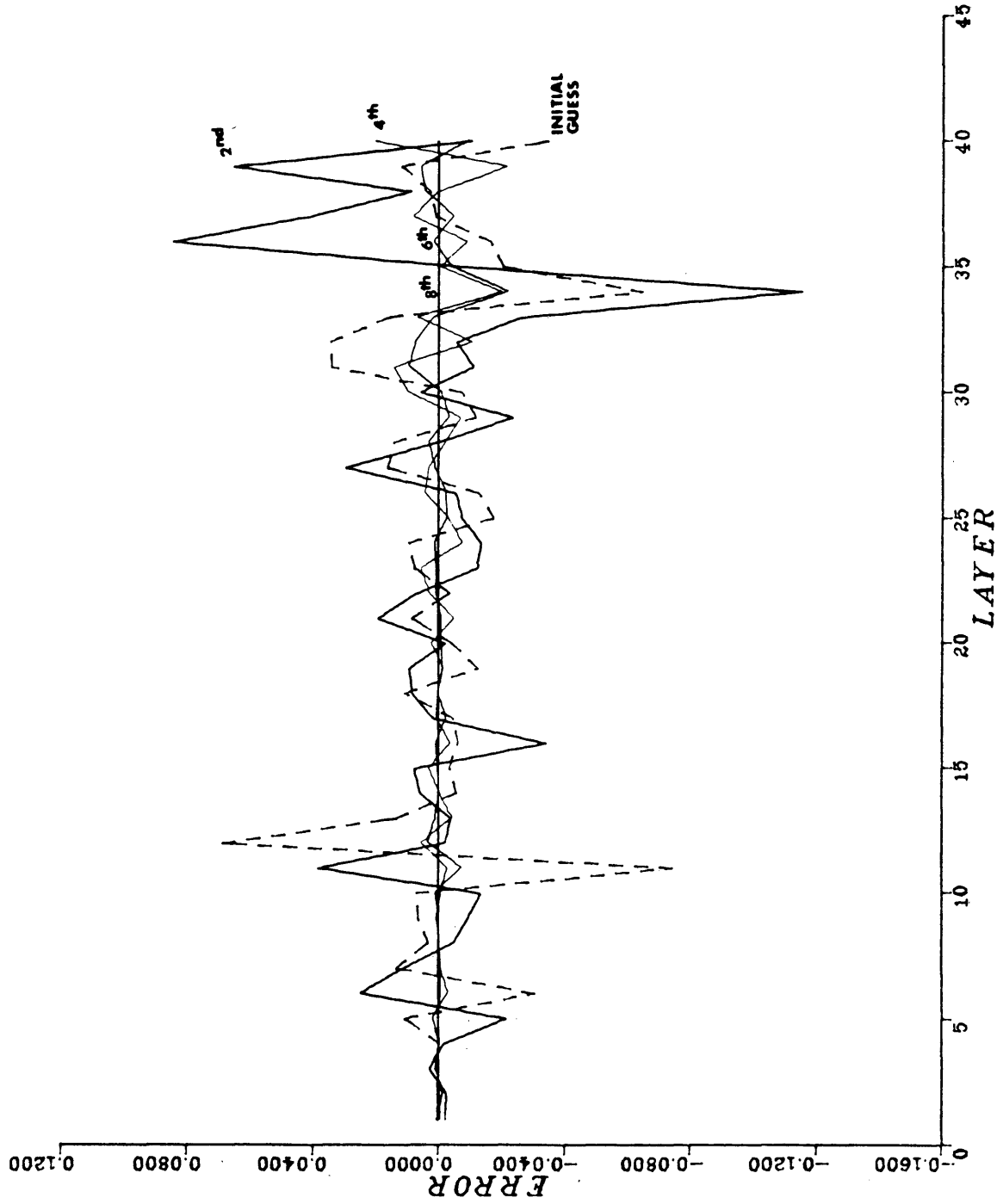


Figure 2-86 The observed minus calculated trace for successive iterations with model 6 and - 01 exponentially incremented scaling factor.

ERRORS

For most of the models and all of the noise cases considered, several plots were made to illustrate the error behavior. First of all, in each case, RMS error vs. iteration was plotted, and in all cases, with the exception of the scaling case, the decrease in error was monotonic. The case of the scaling factor differed in that the RMS error increased for the first iteration but decreased thereafter. This occurred basically because while the error in the scaling factor decreased, the impedance errors slightly increased.

The observed minus the calculated trace error or difference vector is also plotted. As mentioned before, what this mainly serves to illustrate is that the errors in the shallower layers decrease more quickly than those of the deeper layers. This results in the model parameters converging first at the shallower layers and then during later iterations at the deeper layers.

A form of sensitivity measurement which relates to error behavior is shown in the partial derivative plots. Selected plots are included rather than all as they are very similar from model to model, with the exception for the convolutional noise case where the values are shifted due to a shifted wavelet. As Wiggins (1976) mentions in his paper and Pelissier(1979) discusses in his thesis, the partial derivatives in matrix A of equation 7 relate directly to sensitivity. The larger the partial derivatives the larger their effect on the solution.

To produce a cleaner plot, only every sixth column of the partial derivative matrix of equation 8 is plotted. This results in having the partial derivative of the calculated values at each time layer with respect to parameter 1, parameter 7, ect. It can be seen the diagonal elements are the largest by far. For example, the derivative of time 7 is greatest when its with respect to the impedance of layer 7 and 8. What this says is that the primaries are mainly driving the inversion.

Finally, we need to use some information gained from singular value decomposition to relate the errors in observations to errors in the parameter correction determination. Both Wiggins (1976) and Johansen (1974) discuss determining error relations making use of this information. The basic equation used is

$$\sigma_{pj}^2 = \sum_{i=1}^N v_{ji} \left(\frac{\lambda_i}{\lambda_i^2 + r^2} \right)^2 \sigma_t^2$$

where

- σ_t^2 is the variance of the observational errors
- v_{ji} is the j^{th} component of eigenvector V_i
- λ_i is the i^{th} eigenvalue
- r is the damping factor

The equation relates the variance of the observations to the variance of the parameter corrections. This is important to note, since for our inversion, we are solving

for parameters corrections, not the actual parameters.

Some plots for representative models are shown with $\sigma_t^2 = .01$. In general, they are similar in character and two basic characteristics emerge. One is that the standard deviation is greater for lower impedances. This seems reasonable considering a small impedance change, if added to small impedances, will generate larger changes in reflection coefficients. Also there is a general decrease in the standard deviation as the time becomes greater.

This follows since the reflection series at deeper layers is less affected by the other layers. Fewer multiples are added into the signal at deeper layers or times. Random noise and other distorting factors have surprisingly little effect on the standard deviation of the parameter corrections other than the overall magnitude.

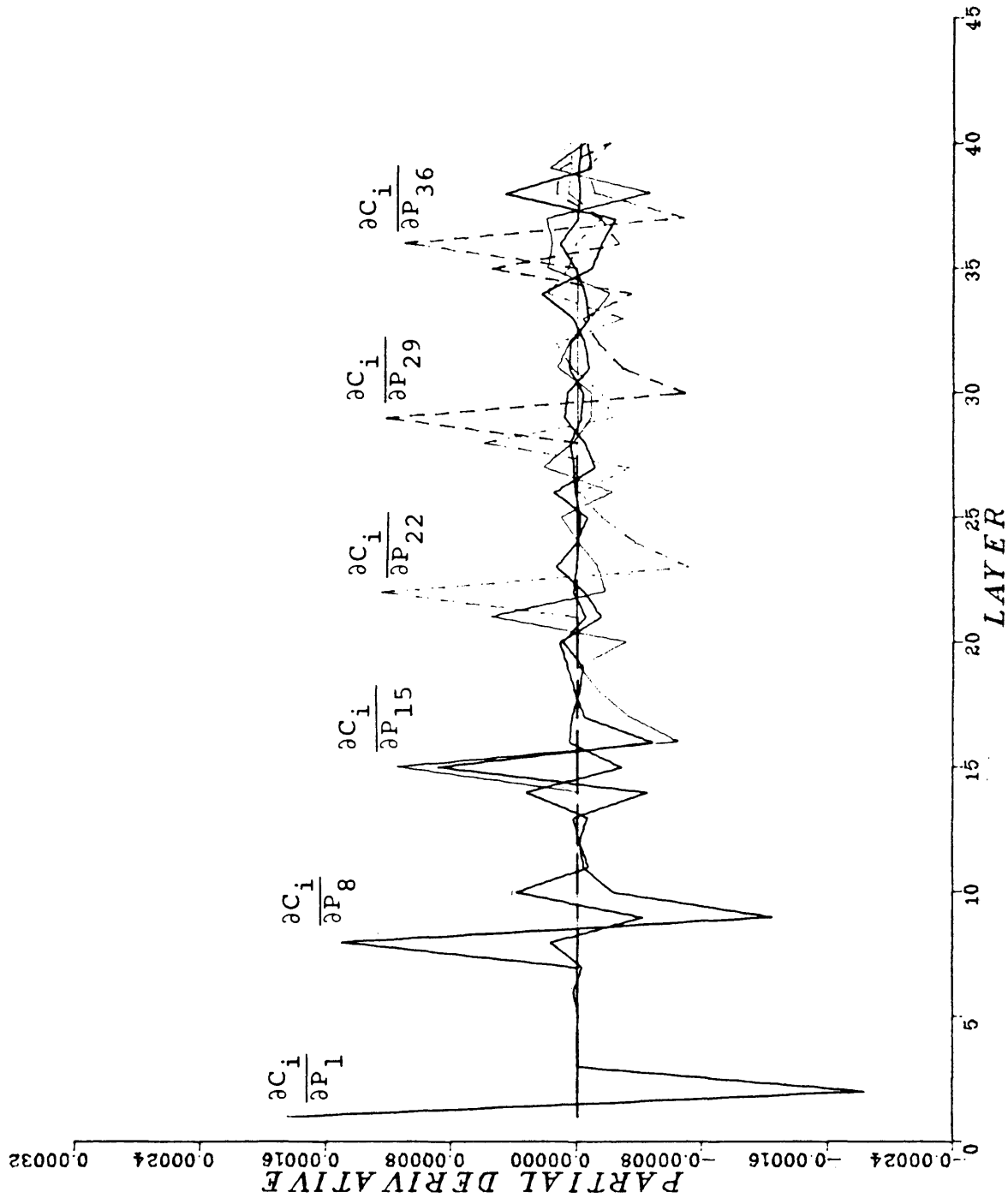


Figure 2-87 Partial derivatives for last iteration on model 6 with zero noise.

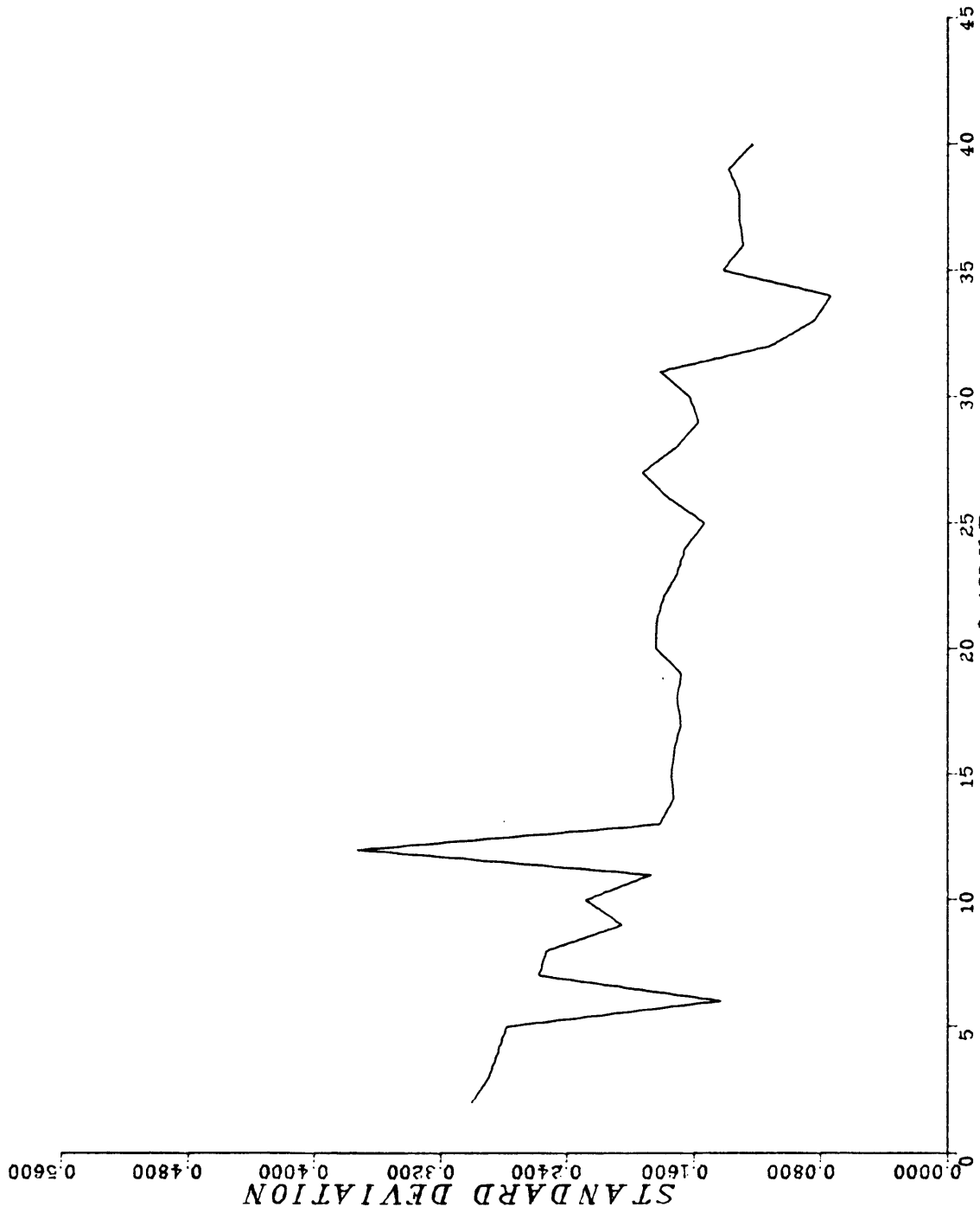


Figure 2-88 Standard deviation of parameter corrections on last iteration on model 6 with zero noise.

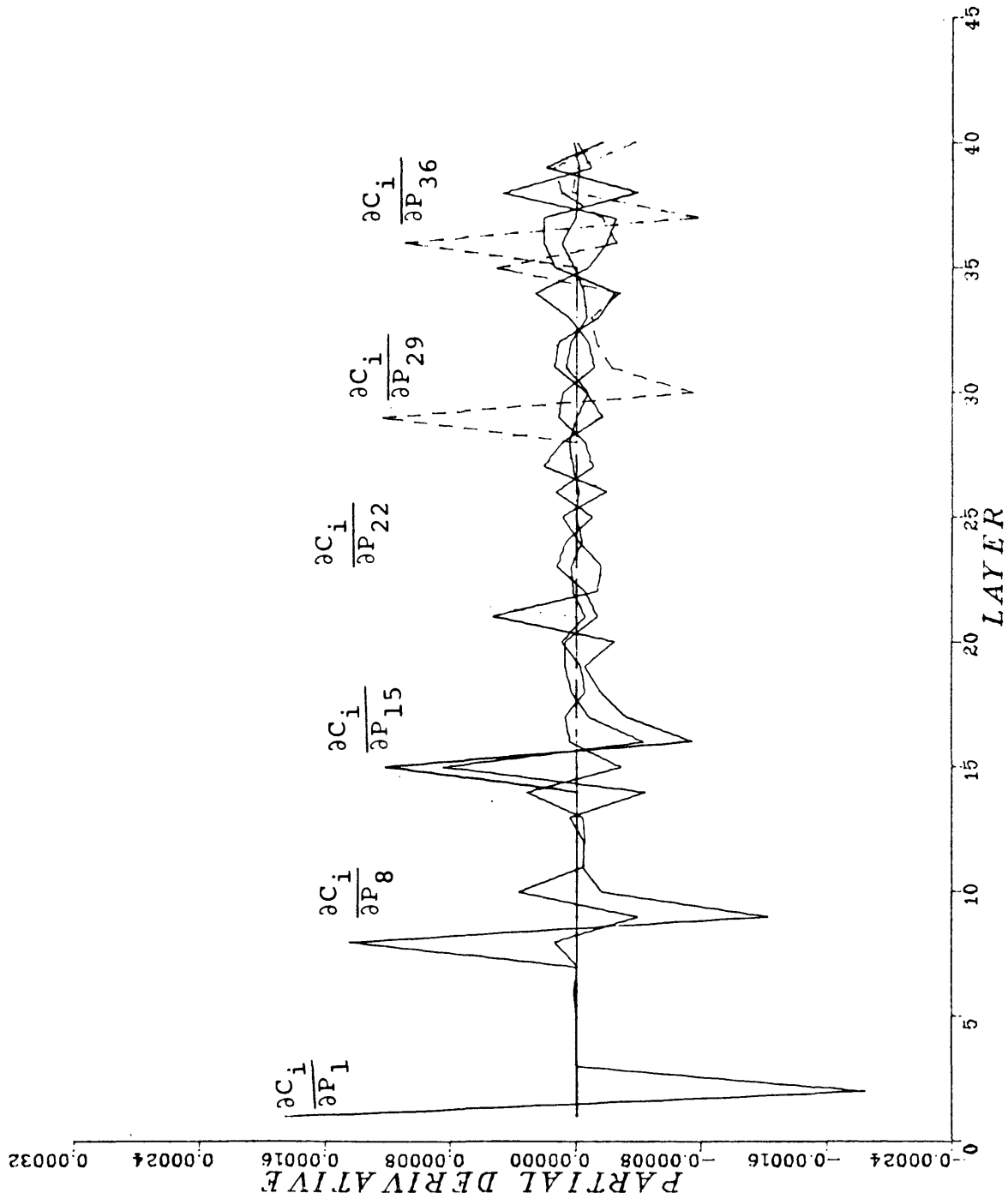


Figure 2-89 Partial derivatives for last iteration on model 6 with .012985 RMS noise level.

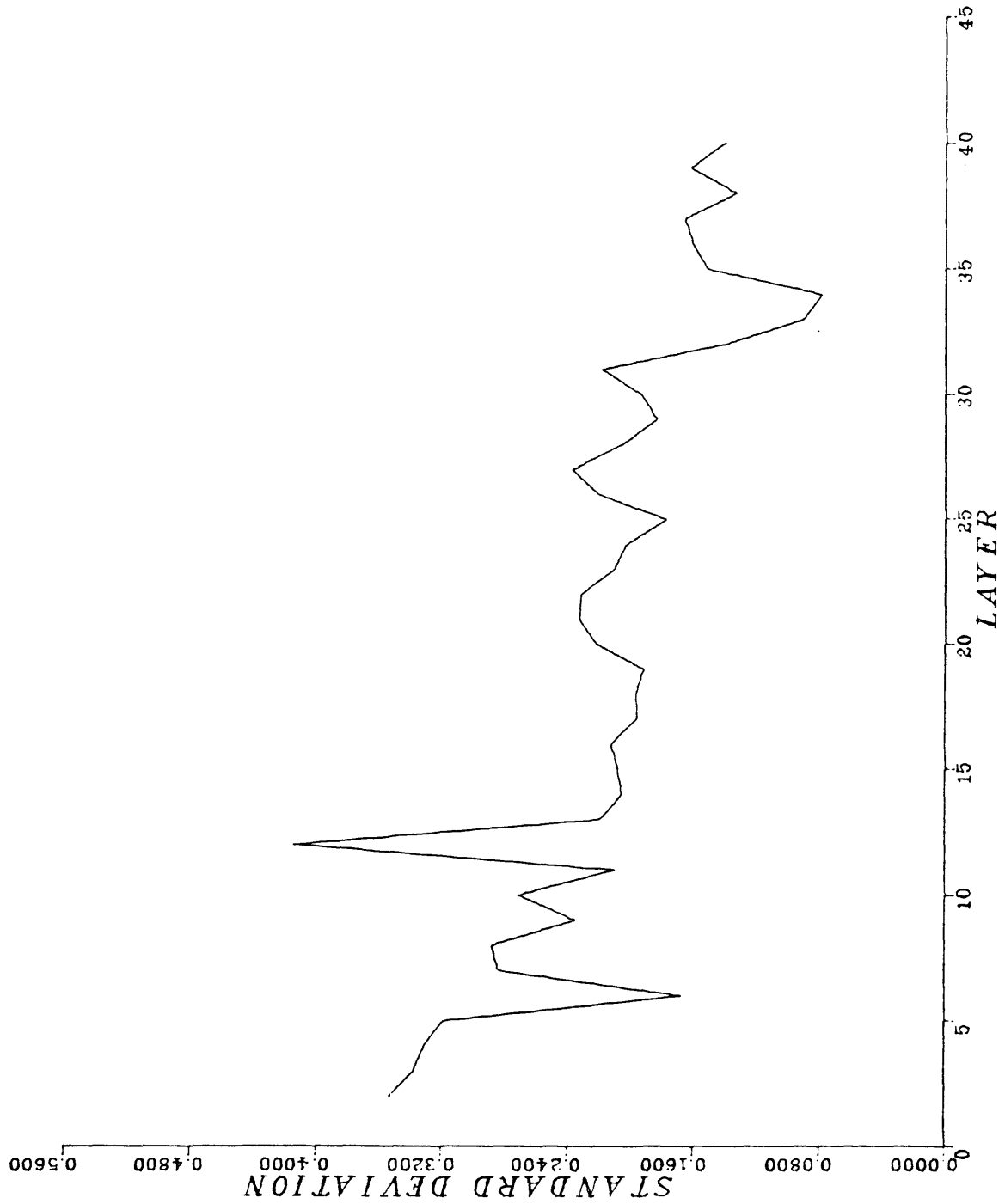


Figure 2-90 Standard deviation of parameter corrections on last iteration on model 6 with .012985 RMS noise level.

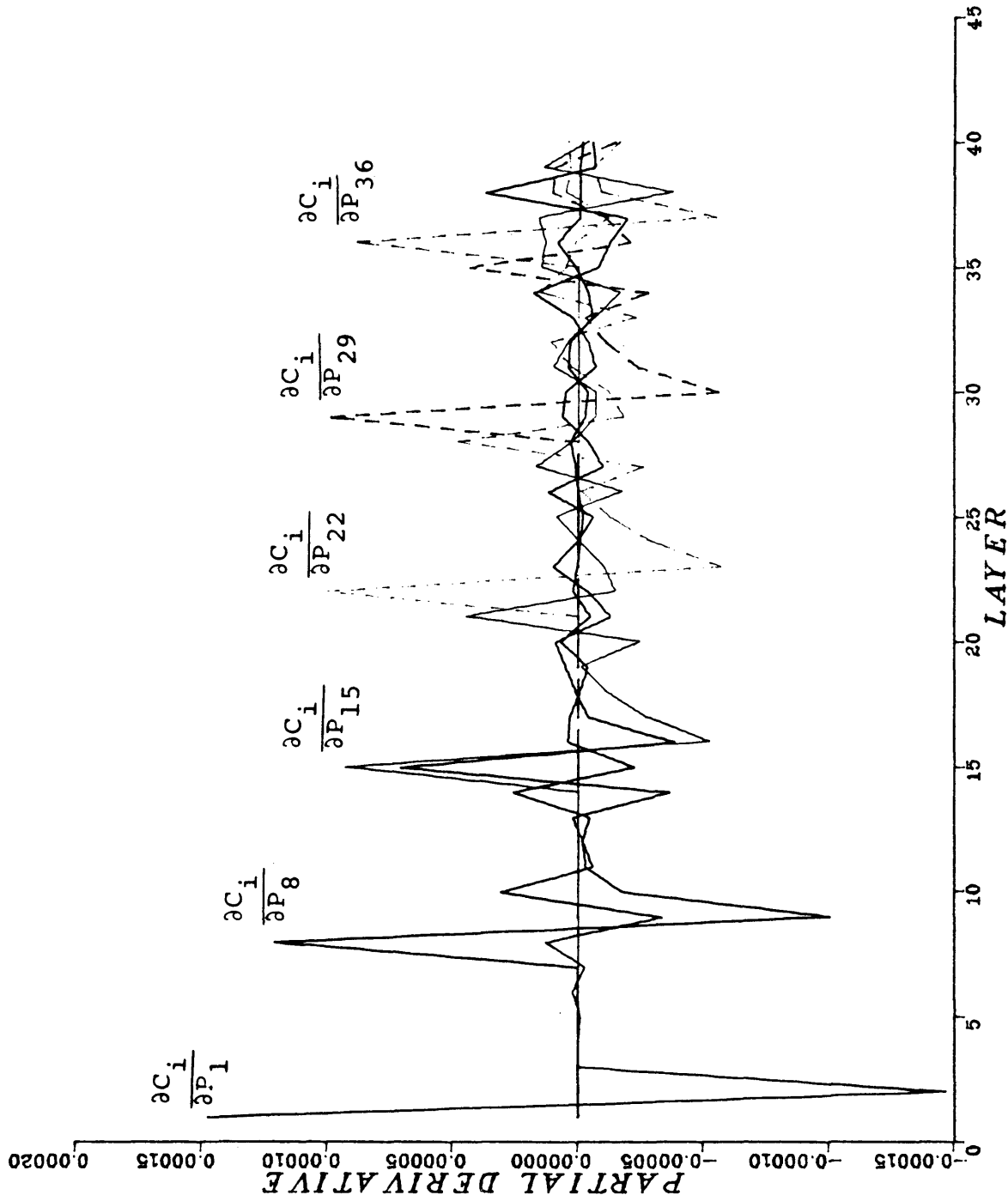


Figure 2-91 Partial derivatives for last iteration on model 6 with .8 constant linear scaling factor.

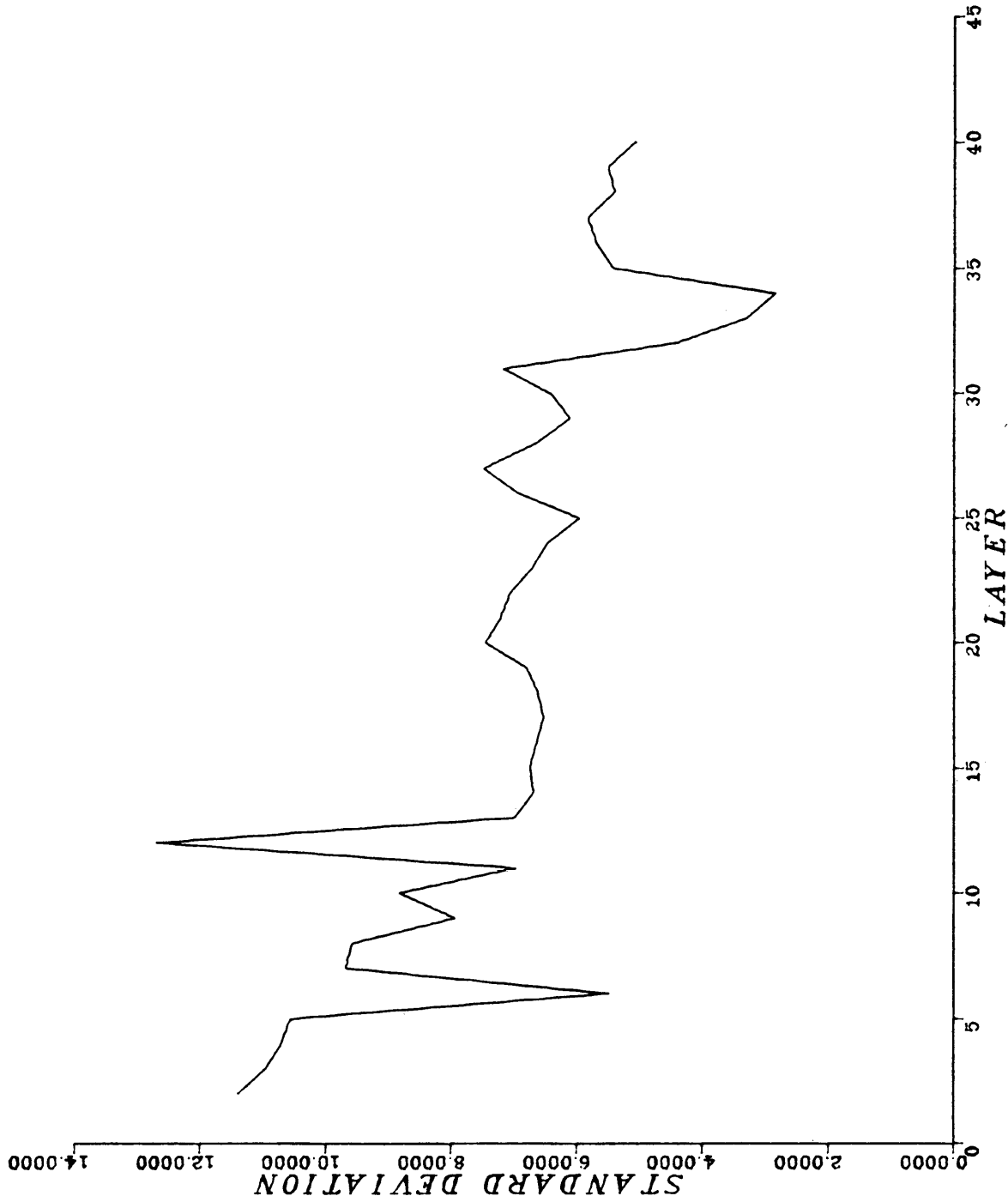


Figure 2-92 Standard deviation of parameter corrections on last iteration on model 6 with .8 constant linear scaling factor.

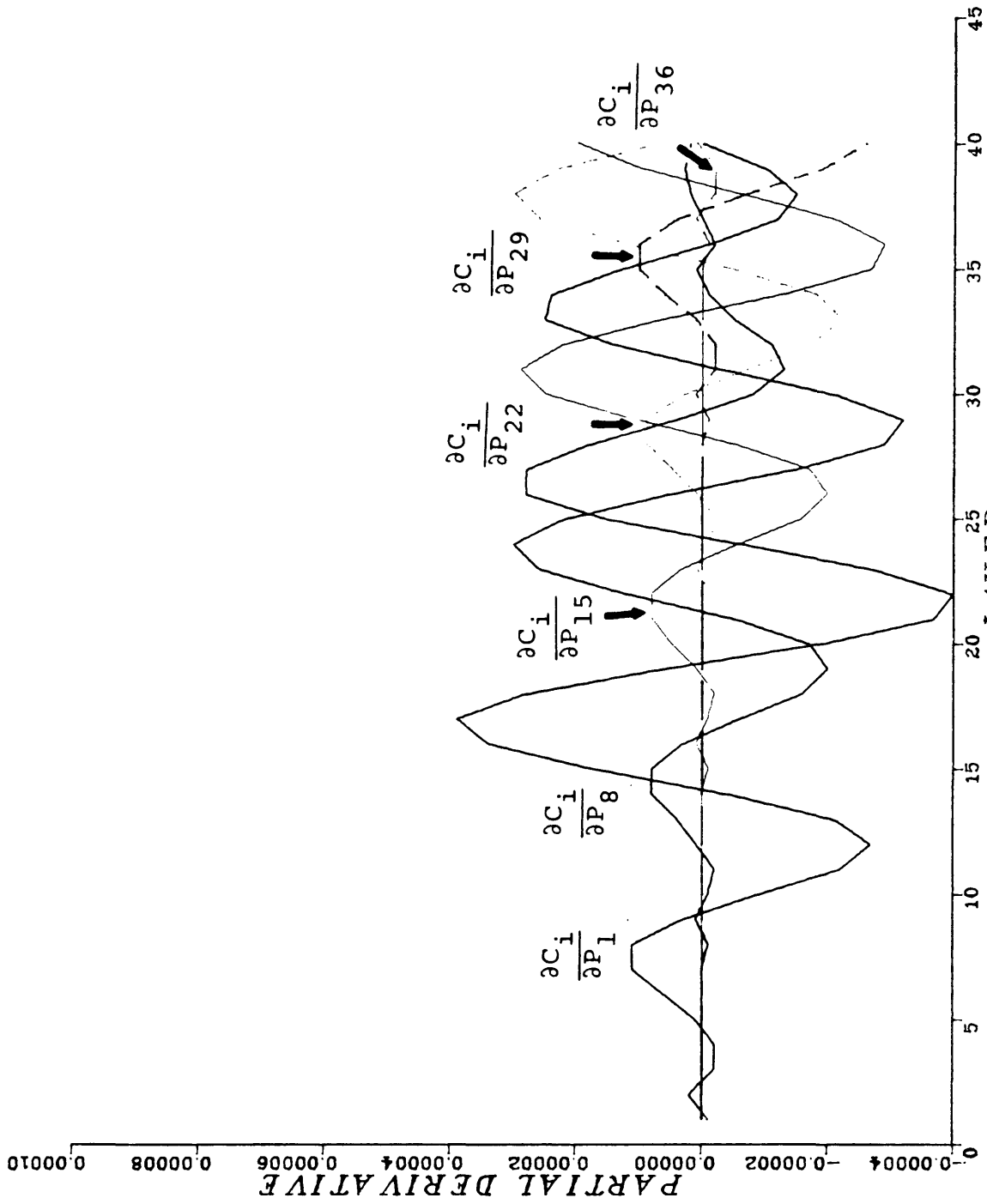


Figure 2-93 Partial derivatives for last iteration on model 6 with corrupted convolutional wavelet at S/N = 1.

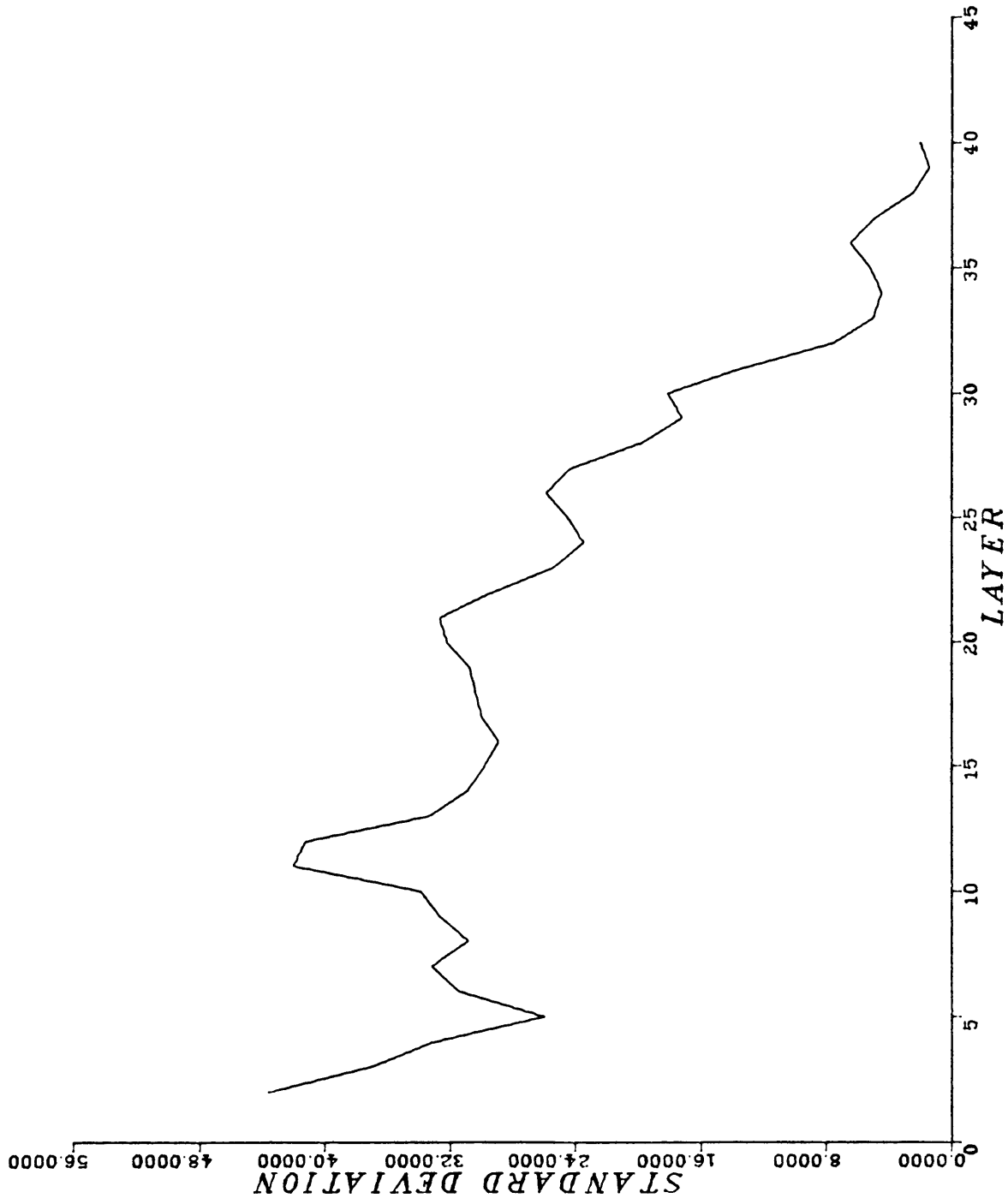


Figure 2-94 Standard deviation of parameter corrections on last iteration on model 6 with corrupted convolutional wavelet at S/N = 1.

FEASABILITY OF METHOD

Table 4 lists the CPU times required for models of varying numbers of layers. The singular value decomposition subroutine is from Golub et al (1970) and uses about 80% of the CPU time. From this study, for the majority of the cases, it appears the matrix equations are possibly stable enough to solve using other methods. This appears necessary as the requirements for computer time for actual field data exceed that available on present day computers. In short, while singular value decomposition has valuable qualities in solving the matrix equations, the computer time it requires seems to preclude its use for actual field data even for the larger and faster machines.

TABLE 4

<u># of Layers</u>	<u># of iterations</u>	<u>Time/iterations (Sec)</u>	<u>Total time (Sec)</u>
5	0	-	1.59
10	2	2.02 sec	4.03
20	3	5.67 sec	17.00
30	3	16.71	50.13
40	3	29.60	88.81
50	3	57.28	171.83

Note: Values are for Model 1 with no noise.

Data run on DEC 10 computer.

CONCLUSIONS

Using generalized linear inverse theory and Goupillard's recursive forward solutions, a quickly converging and stable technique for inversion can be implemented. Because of its approach to the solution of the matrix equations resulting from singular value decomposition, the method has good stability in the presence of noise.

Also because of its non-sequential approach to the solution of the parameter corrections, errors are not magnified with depth. At present, the method is hindered by the fact the computation time is relatively high. Approximately 80% of the CPU time is used to perform the singular value decomposition. The matrix equations, however, need not necessarily be solved in this method.

Finally, the formulation of the problem is readily adaptable to the addition of other parameters, a fact illustrated when scaling factors were included. This allows excellent flexibility while following the generalized linear inversion method.

APPENDIX

This appendix contains all programs and subroutines, excluding plot programs, used in this study. Documentation has been included at appropriate places in the listings to explain the function of each program in addition to its input and output.

C
 C
 C MAIN PROGRAM FOR SEISMIC INVERSION USING GENERALIZED
 C LINEAR INVERSE THEORY
 C
 C *****
 C
 C THIS PROGRAM PERFORMS THE INVERSION PROCESS AS FOLLOWS.
 C THE "OBSERVED DATA" IS CALCULATED FROM A SET OF IMPEDANCES
 C WHICH CHARACTERIZE THE EARTH MODEL. THE IMPEDANCES ARE ONE OF THE
 C INPUTS TO THE PROGRAM. IN THIS WAY, THE CORRECT MODEL PARAMETERS
 C ARE KNOWN. THE PROGRAM WILL THEN ACCEPT AN INITIAL GUESS AT THE
 C MODEL PARAMETERS OR WILL GENERATE ITS OWN USING THE OBSERVED
 C TRACE AND SEISLOG INVERSION. WITH THE INITIAL VALUES, THE PROGRAM
 C ITERATIVELY SOLVES THE INVERSION PROBLEM.
 C
 C
 C THE MAIN PROGRAM IS INTERACTIVE AND ASKS SEVERAL QUESTIONS
 C OF THE USER. THE ANSWERS DETERMINE WHICH OPTIONS AVAILABLE
 C IN THE PROGRAM WILL BE USED.
 C
 C WHAT IS YOUR INPUT FILE?
 C
 C ***** THE INPUT FILE SPECIFIED MUST CONTAIN THE NUMBER OF
 C REFLECTION COEFFICIENTS OR REFLECTION SURFACES IN THE
 C MODEL, THE NUMBER OF IMPEDANCES LAYERS IN THE MODEL,
 C THE NUMBER OF TIME UNITS TO BE CALCULATED FOR THE MODEL
 C SYNTHETIC SEISMOGRAM AND FINALLY THE IMPEDANCES OF EACH
 C UNIT TIME LAYER IN THE EARTH MODEL.
 C THE FORMAT MUST BE (I5,/,I5,/,I5,/, (8F12.2))
 C
 C ARE YOU INPUTTING INITIAL GUESS?
 C
 C ***** IF YES, THE DATA MUST BE PRESENT ON THE INPUT FILE
 C IN FORMAT (8F12.2). IF NO, THE INITIAL VALUES ARE
 C GENERATED IN THE PROGRAM WHICH USES A COMPUTATIONALLY
 C FAST SEISLOG INVERSION TECHNIQUE.
 C
 C DO YOU WANT A COMPLETE PRINTOUT?
 C
 C ***** A COMPLETE PRINTOUT CONTAINS ALL THE INTERMEDIATE ITERATION
 C RESULTS, OTHERWISE THE PRINTOUT CONTAINS ONLY THE FIRST
 C AND LAST ITERATION RESULTS.
 C
 C DO YOU WANT RANDOM NOISE ADDED TO SEISMIC SYNTHETIC?
 C
 C ***** THE RANDOM NOISE IS CONTAINED IN A FILE NAMED
 C NOISE.DAT. THE PROGRAM WILL CALCULATE AND PRINT OUT
 C ON THE TERMINAL THE RMS VALUE OF THE NOISE.
 C

C DO YOU WANT WAVELET CONVOLVED WITH SEISMIC SYNTHETIC?
C
C ***** THE SEISMIC SYNTHETIC DENOTES THE "OBSERVED TRACE" .
C THE WAVELET IS CONVOLVED WITH THE REFLECTIVITY SERIES
C TO PRODUCE A SYNTHETIC MODEL TRACE.
C
C DO YOU WANT WAVELET CONVOLVED WITH CALCULATED SYNTHETIC?
C
C ***** THIS ALLOWS THE CALCULATED SYNTHETIC TO BE CONVOLVED
C WITH EITHER NO WAVELET, THE SAME WAVELET AS IS
C CONVOLVED WITH THE "OBSERVED DATA" OR A DIFFERENT
C WAVELET.
C
C NAME THE FILE CONTAINING THE WAVELET TO BE CONVOLVED WITH
C SEISMIC SYNTHETIC?
C
C ***** IF A WAVELET IS TO BE CONVOLVED WITH THE "OBSERVED
C DATA" THE PROGRAM READS THE WAVELET COEFFICIENTS
C FROM THIS FILE.
C
C NAME THE FILE CONTAINING THE WAVELET TO BE CONVOLVED WITH
C THE CALCULATED SYNTHETIC?
C
C ***** THIS FILE MAY BE EITHER THE SAME OR DIFFERENT FROM
C THE FILE CONTAINING THE WAVELET TO BE CONVOLVED WITH
C THE "OBSERVED DATA".
C
C INDICATE HOW YOU WANT OBSERVED SYNTHETIC SCALED
C 1=CONSTANT,2=LINEAR,3=EXPONENTIAL,4=NONE
C
C ***** AFTER A SYNTHETIC MODEL TRACE REPRESENTING THE
C OBSERVED TRACE IS CALCULATED, IT MAY BE SCALED.
C THE FOLLOWING QUESTIONS WILL BE ASKED ONLY
C IF SCALING IS DESIRED.
C
C WHAT IS CONSTANT SCALING FACTOR?
C
C ***** THIS FACTOR IS MULTIPLIED TIMES EVERY TRACE COEFFICIENT.
C
C WHAT IS LINEAR INCREMENT FOR SCALING?
C
C ***** AT EACH SUCCESSION TIME LAYER THIS INCREMENT IS ADDED TO
C TO THE PREVIOUS SCALING FACTOR AND MULTIPLIED TIMES
C THAT TRACE COEFFICIENT.
C THE INITIAL SCALING FACTOR IS 1.0
C
C WHAT IS EXPONENTIAL INCREMENT FOR SCALING?
C
C ***** AT EACH SUCCESSION TIME LAYER THIS INCREMENT IS ADDED TO
C THE PREVIOUS EXPONENT SCALING FACTOR AND MULTIPLIED
C TIMES THAT TRACE COEFFICIENT.
C

```

C           THE INITIAL EXPONENT IS 0.0
C
C   NOTE:  IF SCALING IS PERFORMED, THE PROGRAM WILL ADD THIS AS
C           A MODEL PARAMETER TO BE SOLVED FOR.
C
C *****
C
C   DIMENSION MIPDLG(100),SEISIG(150),DIFF(150),INGESS(100)
C   DIMENSION SEISMD(150),RCLG(150),FILT(110),RAMDA(101)
C   DIMENSION PRT(10000),B(10000)
C   REAL MIPDLG,INGESS,LIMIT,LIMITD
C   DOUBLE PRECISION filFIL(10),ERR(10)
C   DOUBLE PRECISION FILNAM,FLNAME
C   DATA ANS5/'NO'/
C   DATA AYES/'YES'/,AND/'NO'/,IN,IOUT,IOUTTY/02,11,4/
C   DATA FIL/'FIL1.DAT','FIL2.DAT','FIL3.DAT','FIL4.DAT',
C & 'FIL5.DAT','FIL6.DAT','FIL7.DAT','FIL8.DAT',
C & 'FIL9.DAT','FIL10.DAT'/
C   DATA ERR/'ERR1.DAT','ERR2.DAT','ERR3.DAT','ERR4.DAT',
C & 'ERR5.DAT','ERR6.DAT','ERR7.DAT','ERR8.DAT',
C & 'ERR9.DAT','ERR10.DAT'/
C   OPEN (UNIT=20,FILE = 'RMS.DAT')
C   ITER = 0
C   HSUM = 10.0
C
C   SET LIMITS TO TERMINATE ITERATIONS
C
C   LIMIT = 0.00005
C   LIMITD = 0.0001
C
C   READ INPUT IMPEDANCES FOR EARTH MODEL
C
C   WRITE (IOUTTY,98)
C   98 FORMAT (1X,'WHAT IS YOUR INPUT FILE?')
C   ACCEPT 99, FILNAM
C   99 FORMAT (A10)
C   OPEN (UNIT=IN,FILE =FILNAM)
C   READ (IN,100) NPM,NIMPD,MDLNTH, (MIPDLG(I),I=1,NIMPD)
C 100 FORMAT (I5,/,I5,/,I5,/, (8F12.2))
C   MD = MDLNTH
C
C   WRITE INPUT MODEL IMPEDANCES TO FILE FOR PLOTTING
C
C   OPEN(UNIT=21,FILE=p=)FIL(1))
C   DO 17 I=1,NIMPD
C   HOLDI = I 11,0
C 17 WRITE(21,107) HOLDI,MIPDLG(I)
C 107 FORMAT (1X,F5.0,F12.2)
C   CLOSE (UNIT=21, FILE=FIL(1) )

```

```

C
C CALCULATE REFLECTIVITY SERIES FROM IMPEDANCE MODEL
C
      NPR = NIMPD
      CALL IPDRC (MIPDLG,NIMPD,RCLG)
      CALL FWRD (RCLG,NPM,SEISMD,MDLNTH)
C
C CHECK IF INITIAL GUESS IS INPUT AND IF COMPLETE
C PRINTOUT IS REQUESTED
C
      WRITE (IOUTTY,101)
101  FORMAT (1X," ARE YOU INPUTING INITIAL GUESS?")
      ACCEPT 102,ANS
102  FORMAT (A4)
      WRITE (IOUTTY,106)
106  FORMAT (1X," DO YOU WANT A COMPLETE PRINTOUT?")
      ACCEPT 102, ANS2
C
C WRITE MODEL IMPEDANCES AND OBSERVED SEISMIC SYNTHETIC TO
C FILE FOR SUBSEQUENT PRINTOUT
C
      12 WRITE (IOUT,109)
109  FORMAT (1X,T20," MODEL IMPEDANCES",//)
      WRITE (IOUT,111) (MIPDLG(I), I=1,NIMPD)
111  FORMAT (1X,(12F10.2))
      WRITE (IOUT,110)
110  FORMAT (1X,///,T20," OBSERVED SEISMIC SYNTHETIC",//)
      WRITE (IOUT,104) (SEISMD(J),J=1,MDLNTH)
104  FORMAT (1X,(12F10.6))
      WRITE (IOUTTY,112)
C
C CHECK IF RANDOM NOISE IS TO BE ADDED TO SEISMIC SYNTHETIC.
C IF YES, READ NOISE FILE,ADD TO DATA, AND CALCULATE RMS AMPLITUDE
C
112  FORMAT (1X,"DO YOU WANT RANDOM NOISE ADDED TO SEISMIC SYNTHETIC?")
      ACCEPT 102, ANS3
      IF (ANS3.EQ.AND) GO TO 20
      OPEN (UNIT=1, FILE="NOISE.DAT")
      READ (1,113) (RCLG(I), I=1,MDLNTH)
      CLOSE (UNIT=1, FILE="NOISE.DAT")
113  FORMAT (F12.8)
      SMNOIS = 0.0
      DO 21 J=1,MDLNTH
21   SMNOIS = SMNOIS + RCLG(J)**2
      SMNOIS = SQRT(SMNOIS/MDLNTH)
      WRITE (IOUTTY,114) SMNOIS
114  FORMAT (/,1X,"RMS NOISE LEVEL =",F9.6)
      SMDATA = 0.0
      DO 23 I=1,MDLNTH
23   SMDATA = SMDATA + SEISMD(I)**2
      SMDATA = SQRT(SMDATA/MDLNTH)

```

```

WRITE (IOUTTY,24) SMDATA
24 FORMAT (/,1X,"RMS DATA LEVEL",F9.6)
DO 22 I=1,MDLNTH
22 SEISMD(I) = SEISMD(I) + RCLG(I)
C
C WRITE OBSERVED SEISMIC SYNTHETIC WITH NOISE TO FILE
C FOR SUBSEQUENT PRINTOUT
C
WRITE (IOUT,115)
115 FORMAT (///,1X,T20,"OBSERVED SEISMIC SYNTHETIC WITH NOISE")
WRITE (IOUT,105) (SEISMD(J), J=1,MDLNTH)
C
C CHECK IF A WAVELET IS TO BE CONVOLVED WITH OBSERVED OR
C CALCULATED DATA
C
20 WRITE (IOUTTY,96)
96 FORMAT (1X,"DO YOU WANT WAVELET CONVOLVED WITH SEISMIC",
& " SYNTHETIC?")
ACCEPT 102, ANS4
IF (ANS4.EQ.AND), GO TO 30
WRITE (IOUTTY,117)
117 FORMAT (1X,"DO YOU WANT WAVELET CONVOLVED WITH CALCULATED",
& " SYNTHETIC?")
ACCEPT 102, ANS5
C
C ADJUST ITERATION LIMITS IF WAVELET CONVOLUTION IS PERFORMED
C
IF (ANS5.EQ.AYES) LIMIT = LIMIT * 0.1
IF (ANS5.EQ.AYES) LIMITD = LIMITD * 0.01
C
C READ IN WAVELET COEFFICIENTS, CONVOLVE, AND WRITE TO FILE
C
WRITE (IOUTTY,28)
28 FORMAT (1X,"NAME THE FILE CONTAINING THE WAVELET ",
& " TO BE CONVOLVED",/, " WITH THE SEISMIC SYNTHETIC")
ACCEPT 99, FLNAME
OPEN (UNIT=3, FILE=FLNAME)
READ (3,97) (FILT(I), I=1,35)
97 FORMAT (20X,6X,10X,F12.8)
CLOSE (UNIT=3, FILE=FLNAME)
CALL FOLD (MDLNTH,SEISMD,35,FILT,LC,SEISIG)
DO 95 I=1,LC
95 SEISMD(I) = SEISIG(I)
WRITE (IOUT,116)
116 FORMAT (///,1X,T20,"OBSERVED SEISMIC SYNTHETIC WITH",
& " CONVOLVED WAVELET")
WRITE (IOUT,105) (SEISMD(J), J=1,MDLNTH)
IF (ANS5.EQ.AND) GO TO 30
WRITE (IOUTTY,29)
29 FORMAT (1X,"NAME THE FILE CONTAINING THE WAVELET ",
& " TO BE CONVOLVED",/, " WITH THE CALCULATED SYNTHETIC")

```

```

ACCEPT 99, FLNAME
OPEN (UNIT=8, FILE=FLNAME)
READ (8,97) (FILT(I), I=1,35)
CLOSE (UNIT=8, FILE=FLNAME)
C
C SCALE DATA ACCORDING TO COMMAND
C
30 IF (ANS4.EQ.AYES) MD = LC
WRITE (IOUTTY,90)
90 FORMAT (1X,"INDICATE HOW YOU WANT OBSERVED SYNTHETIC SCALED",/,
& 1X,"1=CONSTANT,2=LINEAR,3=EXPONENTIAL,4=NONE")
ACCEPT 91, SCALE
91 FORMAT (F5.0)
SCL = SCALE
IF (SCL.NE.4.0) NPR = NIMPD + 1
IF (SCALE.EQ.1.0) GO TO 92 213
IF (SCALE.EQ.2.0) GO TO 93 214
IF (SCALE.EQ.3.0) GO TO 94 215
IF (SCALE.EQ.4.0) GO TO 27 216
92 WRITE (IOUTTY,81)
81 FORMAT (1X,"WHAT IS CONSTANT SCALING FACTOR?")
ACCEPT 91, SCALE
DO 82 I=1,MD
82 SEISMD(I) = SCALE * SEISMD(I)
GO TO 26
93 WRITE (IOUTTY,83)
83 FORMAT (1X,"WHAT IS LINEAR INCREMENT FOR SCALING?")
ACCEPT 91, SCALE
SCALER = 1.0
DO 84 I=1,MD
SEISMD(I) = SEISMD(I) * SCALER
84 SCALER = SCALER + SCALE
GO TO 26
94 WRITE (IOUTTY,85)
85 FORMAT (1X,"WHAT IS EXPONENTIAL INCREMENT FOR SCALING?")
ACCEPT 91, SCALE
SCALER = 0.0
DO 86 I=1,MD
SEISMD(I) = SEISMD(I) * EXP(SCALER)
86 SCALER = SCALER + SCALE
26 WRITE (IOUT,123)
123 FORMAT (///,1X,T20,"OBSERVED SEISMIC SYNTHETIC WITH",
& " SCALING FACTOR")
WRITE (IOUT,105) (SEISMD(J), J=1,MDLNTH)
C
C EITHER GENERATE OR READ INITIAL GUESSES FOR IMPEDANCES
C
27 IF (ANS.EQ.AYES) GO TO 10
INGESS(1) = MIPDLG(1)
IF (ANS4.EQ.AYES) GO TO 31
DO 11 I=1,NPM

```

```

11 INGESS(I+1) = INGESS(I) * (1.0 + SEISMD(I))/(1.0 - SEISMD(I))
GO TO 32
31 DO 13 I=1,NPM
13 INGESS(I+1)=INGESS(I) * (1.0 + SEISMD(I+17))/(1.0 - SEISMD(I+17))
32 IF (SCL.NE.4.0) INGESS(NPR) = -0.01
IF (SCL.EQ.1.0) INGESS(NPR) = 1.0
GO TO 25
10 READ (IN,103) (INGESS(I),I=1,NPR)
103 FORMAT (8F12.2)
C
C BEGINNING OF INVERSION ITERATION LOOP
C
25 IHOLD = ITER + 2
OPEN (UNIT = 21, FILE = FIL(IHOLD) )
OPEN (UNIT = 19, FILE = ERR(IHOLD))
DO 18 I=1,NPR
HOLDI = I
18 WRITE (21,107) HOLDI,INGESS(I)
CLOSE (UNIT = 21, FILE = FIL(IHOLD) )
C
C CALCULATE FORWARD SOLUTION FOR INITIAL GUESS PARAMETERS
C
CALL IPDRC (INGESS,NIMPD,RCLG)
CALL FWRD(RCLG,NPM,SEISIG,MDLNTH)
IF (ANS5.EQ.AND) GO TO 122
CALL FOLD(MDLNTH,SEISIG,35,FILT,LC,RCLG)
DO 121 N=1,MDLNTH
121 SEISIG(N) = RCLG(N)
122 SCALFC = INGESS(NPR)
IF (SCL.NE.4.0) CALL SCAL (SEISIG,SCALFC,SCL,MDLNTH)
DO 131 K=1,MDLNTH
DIFF(K) = SEISMD(K) - SEISIG(K)
HOLDK = K
131 WRITE (19,132) HOLDK,DIFF(K)
132 FORMAT (F5.0,F12.8)
CLOSE (UNIT=19, FILE=ERR(IHOLD))
CALL ERRMS (SEISMD,SEISIG,DIFF,MDLNTH,SUM)
DSUM = ABS(HSUM - SUM)
HSUM = SUM
HOLDI = ITER
WRITE (20,108) HOLDI,SUM
108 FORMAT (1X,F5.0,F10.6)
IF (ANS2.EQ.AND .AND. SUM.GT.LIMIT) GO TO 16
WRITE (IOUT,120) ITER, (INGESS(I),I=1,NIMPD)
C
C WRITE INTERMEDIATE ITERATION RESULTS TO FILE FOR PRINTOUT
C
120 FORMAT (///,1X,T20," ITERATIVE SOLUTION IMPEDANCE MODEL #",
& I3, //,(12F10.2))
IF (SCL.NE.4.0) WRITE (IOUT,125) INGESS(NPR)
125 FORMAT (/ ,1X,"SCALING FACTOR=",F10.5,/)

```

```

      WRITE (IOUT,130) ITER
130  FORMAT (///,T20," SEISMIC SYNTHETIC MODEL #",I3)
      WRITE (IOUT,105) (SEISIG(I),I=1,MDLNTH)
105  FORMAT (//,1X,(12F10.6))
C
C  CHECK TO BRANCH OUT OF LOOP
C
16  IF (DSUM .LT. LIMITD) GO TO 15
      IF (SUM.LE.LIMIT .OR. ITER.GE.8) GO TO 15
C
C  CALCULATE PARTIAL DERIVATIVE MATRIX AND SINGULAR VALUE
C  DECOMPOSITION SOLUTION
C
      CALL FDPART (PRT,INGESS,SEISIG,NPM,MDLNTH,NIMPD,FILT,ANSS,SCL,1)
      TIME1 = RUNTIM(DUM)
      CALL SVD(PRT,B,DIFF,SEISMD,SUM,NPM,MDLNTH,ITER,DF,INGESS,NIMPD,
&  FILT,ANSS,SCL,RAMDA)
      TIME2 = RUNTIM(DUM)
      TIME = TIME2 - TIME1
      ITER = ITER + 1
      WRITE (4,180)
180  FORMAT (1X," ITERATION           TIME IN SVD SUBROUTINE")
      WRITE (4,190) ITER,TIME
190  FORMAT (1X,/,I6,18X,F12.2,/)
C
C  ADD PARAMETER CORRECTIONS AND BRANCH TO BEGINNING OF LOOP
C
      CALL DIFADD(DIFF,INGESS,INGESS,NPR,SCL)
      GO TO 25
C
C  CALCULATE AND WRITE PARTIAL DERIVATIVES AND STANDARD
C  DEVIATION FOR PARAMETER CORRECTIONS FOR LAST ITERATION
C
15  ND = NPR - 1
      IOPEN = IHOLD - 1
      OPEN (UNIT=21, FILE=FILE(IOPEN))
      READ (21,146) (INGESS(I), I=1,NPR)
146  FORMAT (1X,5X,F12.2)
      CLOSE (UNIT=21, FILE=FILE(IOPEN))
      CALL IPDRC (INGESS,NIMPD,RCLG)
      CALL FWRD(RCLG,NPM,SEISIG,MDLNTH)
      IF (ANSS.EQ.AND) GO TO 152
      CALL FOLD(MDLNTH,SEISIG,35,FILT,LC,RCLG)
      DO 151 N=1,MDLNTH
151  SEISIG(N) = RCLG(N)
152  SCALFC = INGESS(NPR)
      IF (SCL.NE.4.0) CALL SCAL (SEISIG,SCALFC,SCL,MDLNTH)
      OPEN (UNIT=16, FILE="OBCAL.DAT")
      WRITE (16,134) (SEISMD(I),SEISIG(I), I=1,MDLNTH)
      CLOSE (UNIT=16, FILE="OBCAL.DAT")
      CALL FDPART (PRT,INGESS,SEISIG,NPM,MDLNTH,NIMPD,FILT,ANSS,SCL,2)

```

```
NI = 1
OPEN (UNIT=18, FILE="PARTL.DAT")
DO 133 KL=1, MDLNTH
NF = NI + NPR - 2
WRITE (18,134) (PRT(NA), NA=NI, NF, 7)
133 NI = NI + NPR - 1
134 FORMAT (12F10.6)
CLOSE (UNIT=18, FILE="PARTL.DAT")
NNN = NPM
IF (SCL.NE.4.0) NNN = NIMPD
CALL HYSVD(PRT, B, RAMDA, MDLNTH, NNN, .TRUE., .TRUE., 2)
DO 141 I=1, ND
RAMDA(I) = RAMDA(I) ** 2
141 RAMDA(I) = (RAMDA(I) * 0.01) / ((RAMDA(I) + DF**2) **2)
NC = 0
DO 142 J=1, ND
DIFF(J) = 0.0
DO 143 I=1, ND
143 DIFF(J) = (B(NC+I)**2) * RAMDA(I) + DIFF(J)
NC = NC + ND
142 DIFF(J) = SQRT(DIFF(J))
OPEN (UNIT=17, FILE="SDEV.DAT")
WRITE (17,144) (DIFF(J), J=1, ND)
144 FORMAT (10F12.6)
CLOSE (UNIT=17, FILE="SDEV.DAT")
WRITE (IOUT,140) SUM
140 FORMAT (////, 1X, "RMS ERROR=", F8.5)
CLOSE (UNIT=IN, FILE=FILNAM)
CLOSE (UNIT=20, FILE="RMS.DAT")
STOP
END
```

```

C
C
C
C SURROUTINE 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 SAMPLE CALL STATEMENT: CALL FWRD(RFCOEF,NPM,SEIS,MDLNTH)
C
C INPUT:          RFCOEF = ARRAY CONTAINING REFLECTION COEFFICIENTS
C                  NPM   = NUMBER OF REFLECTION COEFFICIENTS
C                  MDLNTH = NUMBER OF TIME UNITS TO BE CALCULATED FOR
C                          FORWARD SOLUTION
C
C OUTPUT:         SEIS = ARRAY CONTAINING FORWARD SOLUTION OR
C                     SEISMIC SYNTHETIC
C *****
C
C
C SUBROUTINE FWRD (RFCOEF,NPM,SEIS,MDLNTH)
C DIMENSION RFCOEF(NPM),SEIS(MDLNTH),AONE(100),BONE(100)
C KEND = NPM - 1
C AONE(1) = 1.0
C BONE(1) = RFCOEF(NPM)
C DO 10 I=KEND,1,-1
C   II = NPM - I
C   DO 20 J=II,1,-1
C 20 BONE(J+1) = BONE(J)
C   BONE(1) = RFCOEF(I)*AONE(1)
C   IF (II.LT.2) GO TO 40
C   DO 30 K=2,II
C   SAVE = AONE(K)
C   AONE(K) = AONE(K) + RFCOEF(I)*BONE(K)
C   BONE(K) = RFCOEF(I)*SAVE + BONE(K)
C 30 CONTINUE
C 40 AONE(II+1) = RFCOEF(I) * BONE(II+1)
C 10 CONTINUE
C CALL POLYDV (NPM,AONE,NPM,BONE,MDLNTH,SEIS)
C RETURN
C 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
C   OUTPUT          Q = ARRAY CONTAINING THE QUOTIENT OF DVD/DVS
C
C *****
C
C
C
C
C   SUBROUTINE POLYDV(N,DVS,M,DVD,L,Q)
C   DIMENSION DVS(N),DVD(M),Q(L)
C   CALL ZERO (L,Q)
C   CALL MOVE (MINO(M,L),DVD,Q)
C   DO 10  I=1,L
C   Q(I) = Q(I)/DVS(1)
C   IF (I.EQ.L) RETURN
C   K = I
C   ISUB = MINO(N-1,L-I)
C   DO 10  J=1,ISUB
C   K = K + 1
C 10 Q(K) = Q(K) - Q(I)*DVS(J+1)
C   RETURN
C   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
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
C SUBROUTINE FOLD (LA,A,LB,B,LC,C)
C DIMENSION A(LA),B(LB),C(LC)
C LC = LA + LB - 1
C CALL ZERO (LC,C)
C DO 1 I=1,LA
C DO 1 J=1,LB
C K = I + J - 1
1 C(K) = C(K) + A(I)*B(J)
C RETURN
C END
```



```

C
C
C
C   SUBROUTINE ERRMS COMPUTES THE TANGENT WEIGHTED DIFFERENCE BETWEEN
C   THE OBSERVED AND CALCULATED SEISMOGRAM AND CALCULATES THE RMS ERROR
C
C
C *****
C
C   SAMPLE CALL STATEMENT:  CALL ERRMS(SEISMD,SEISIG,DIFF,MDLNTH,SUM)
C
C
C   INPUT:      SEISMD = ARRAY CONTAINING OBSERVED DATA TRACE
C               SEISIG = ARRAY CONTAINING CALCULATED DATA TRACE
C               MDLNTH = LENGTH IN TIME UNITS OF BOTH OBSERVED AND
C                   CALCULATED DATA TRACE
C
C
C   OUTPUT:     DIFF = ARRAY CONTAINING TANGENT WEIGHTED DIFFERENCE
C                   BETWEEN THE OBSERVED AND CALCULATED DATA TRACE
C               SUM = RMS AMPLITUDE OF THE UNWEIGHTED DIFFERENCES
C
C *****
C
C
C   SUBROUTINE ERRMS(SEISMD,SEISIG,DIFF,MDLNTH,SUM)
C   DIMENSION SEISMD(MDLNTH),SEISIG(MDLNTH),DIFF(MDLNTH)
C   SUM = 0.0
C   DO 10  I=1,MDLNTH
C   DIFF(I) = SEISMD(I) - SEISIG(I)
C   SUM = SUM + DIFF(I)**2
C 10 DIFF(I) = TAN(SEISMD(I))-TAN(SEISIG(I))
C   SUM = (SUM/MDLNTH)**.5
C   RETURN
C   END

```

```

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(SEISMD,SEISIG,DIFF,MDLNTH,SUM)
C
C
C INPUT: SEISMD = ARRAY CONTAINING OBSERVED DATA TRACE
C SEISIG = ARRAY CONTAINING CALCULATED DATA TRACE
C MDLNTH = 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
C SUBROUTINE ERMS(SEISMD,SEISIG,DIFF,MDLNTH,SUM)
C DIMENSION SEISMD(MDLNTH),SEISIG(MDLNTH),DIFF(MDLNTH)
C SUM = 0.0
C DO 10 I=1,MDLNTH
C DIFF(I) = SEISMD(I) - SEISIG(I)
10 SUM = SUM + DIFF(I)**2
C SUM = (SUM/MDLNTH)**.5
C RETURN
C END

```

```

C
C
C
C SUBROUTINE FDPART CALCULATES THE PARTIAL DERIVATIVE MATRIX
C USING THE FINITE DIFFERENCE APPROACH
C
C *****
C
C SAMPLE CALL STATEMENT:
C   CALL FDPART(A,INGESS,SEISIG,NPM,MDLNTH,NIMPD,FILT,ANS5,SCL,IW)
C
C INPUT:  INGRESS = ARRAY CONTAINING CURRENT VALUES FOR MODEL
C          PARAMETERS (IMPEDANCES + SCALE FACTOR IF APPL.)
C          SEISIG = ARRAY CONTAINING SYNTHETIC SEISMOGRAM GENERATED
C          BY CURRENT MODEL PARAMETERS
C          NPM = NUMBER OF REFLECTION COEFFICIENTS IN MODEL
C          MDLNTH = NUMBER OF VALUES IN SYNTHETIC SEISMOGRAM
C          NIMPD = NUMBER OF IMPEDANCES IN MODEL
C          FILT = ARRAY CONTAINING COEFFICIENTS OF WAVELET TO BE
C          CONVOLVED WITH REFLECTIVITY SERIES TO PRODUCE
C          SYNTHETIC SEISMOGRAM
C          ANS5 = LOGICAL VARIABLE DETERMINING WHETHER FILT
C          MUST BE CONVOLVED WITH REFLECTIVITY SERIES TO
C          PRODUCE SYNTHETIC SEISMOGRAM
C          SCL = NUMBER CODE WHICH DENOTES WHAT TYPE OF GAIN MUST
C          BE APPLIED TO DATA TO PRODUCE SYNTHETIC SEISMOGRAM
C          IW = NUMBER CODE WHICH DENOTES WHETHER PARTIAL
C          DERIVATIVES CALCULATED ARE TO BE WEIGHTED
C              1 = WEIGHTED
C              2 = UNWEIGHTED
C
C OUTPUT:  A = ARRAY CONTAINING PARTIAL DERIVATIVE MATRIX IN
C          ONE DIMENSIONAL FORM; MDLNTH ROWS OF NPR-1
C          LENGTH.
C              NPR-1 = NPM FOR NO GAIN CASE
C              NPR-1 = NIMPD FOR GAIN CASE
C
C *****
C
C SUBROUTINE FDPART(A,INGESS,SEISIG,NPM,MDLNTH,NIMPD,FILT,ANS5,SCL,
C & IW)
C DIMENSION A(10000),INGESS(NPM),SEISIG(MDLNTH)
C DIMENSION SEISIC(150),RCLG(150),FILT(110)
C REAL INGRESS
C DATA AYES/'YES'/,AND/'NO'/
C NPR = NIMPD

```

```

      IF (SCL.NE.4.0) NPR = NIMPD + 1
C
C  OUTER DO LOOP CALCULATES PARTIALS WITH RESPECT TO A SINGLE PARAMETER
C
      DO 10 I=2,NPR
      IJ = I - 1
      PHOLD = INGESS(I)
C
C      CALCULATE A SEISMIC SYNTHETIC FROM PERTURBED MODEL PARAMETERS
C
      INGESS(I) = INGESS(I) * 1.1
      CALL IPDRC (INGESS,NIMPD,RCLG)
      CALL FWRD(RCLG,NPM,SEISIC,MDLNTH)
      IF (ANS5.EQ.AND) GO TO 16
      CALL FOLD(MDLNTH,SEISIC,35,FILT,LC,RCLG)
      DO 15 N=1,MDLNTH
15 SEISIC(N) = RCLG(N)
16 SCALFC = INGESS(NPR)
      IF (SCL.NE.4.0) CALL SCAL(SEISIC,SCALFC,SCL,MDLNTH)
      IF (I.EQ.NPR .AND. SCL.NE.4.0) GO TO 50
      IF (I.LE.2) GO TO 30
      IF (ANS5.EQ.AYES) GO TO 30
C
C  INSERT ALL ZERO PARTIAL DERIVATIVES INTO MATRIX
C
      L = I - 2
      DO 20 K=1,L
      A(IJ) = 0.0
20 IJ = IJ + NPR - 1
30 CONTINUE
      II = I - 1
C
C  INNER LOOPS CALCULATE PARTIALS AT EACH TIME LAYER
C
      IF (ANS5.EQ.AYES) II = 1
      DO 40 J=II,MDLNTH
      IF (IW.EQ.2) GO TO 39
      A(IJ) = (TAN(SEISIC(J))-TAN(SEISIG(J)))*24.15885793
      GO TO 40
39 A(IJ) = (SEISIC(J) - SEISIG(J)) / (INGESS(I) - PHOLD)
40 IJ = IJ + NPR - 1
10 INGESS(I) = PHOLD
      GO TO 70
50 INGESS(NPR) = PHOLD
      IJ = NIMPD
      DO 60 I=1,MDLNTH
      IF (IW.EQ.2) GO TO 59
      A(IJ) = (TAN(SEISIC(I))-TAN(SEISIG(I)))*24.15885793
      GO TO 60
59 A(IJ) = (SEISIC(I) - SEISIG(I)) / (0.01 * INGESS(NPR))
60 IJ = IJ + NIMPD

```

70 CONTINUE
RETURN
END

```

C
C
C   SUBROUTINE SCAL SCALES THE INPUT TRACE ACCORDING TO INPUT PARAMETERS
C
C *****
C
C   SAMPLE CALL STATEMENT:  CALL SCAL(SEIS,SCALFC,SCL,MDLNTH)
C
C   INPUT:      SEIS = ARRAY CONTAINING SERIES TO BE SCALED
C               SCALFC = SCALE FACTOR
C               SCL = NUMBER CODE DENOTING TYPE OF GAIN TO BE APPLIED
C                   1 = CONSTANT SCALING FACTOR
C                   2 = LINEARLY INCREASING SCALING FACTOR
C                   3 = EXPONENTIALLY INCREASING SCALING FACTOR
C                   4 = NO SCALING
C               MDLNTH = LENGTH OF SERIES TO BE SCALED
C
C   OUTPUT:     SEIS = ARRAY CONTAINING SCALED SERIES
C *****
C
C   SUBROUTINE SCAL(SEIS,SCALFC,SCL,MDLNTH)
C   DIMENSION SEIS(MDLNTH)
C   IF (SCL.EQ.1.0) GO TO 10
C   IF (SCL.EQ.2.0) GO TO 20
C   IF (SCL.EQ.3.0) GO TO 30
C
C   CONSTANT SCALING
C
C   10 DO 15 KK=1,MDLNTH
C      15 SEIS(KK) = SEIS(KK) * SCALFC
C      GO TO 40
C
C   LINEARLY INCREASING SCALING
C
C   20 SCALER = 1.0
C      DO 25 KK=1,MDLNTH
C         SEIS(KK) = SEIS(KK) * SCALER
C   25 SCALER = SCALER + SCALFC
C      GO TO 40
C
C   EXPONENTIALLY INCREASING SCALING
C
C   30 SCALER = 0.0
C      DO 35 KK=1,MDLNTH
C         SEIS(KK) = SEIS(KK) * EXP(SCALER)

```

```
35 SCALER = SCALER + SCALFC  
40 CONTINUE  
   RETURN  
   END
```

```

C
C
C
C   SUBROUTINE SVD SOLVES THE MATRIX INVERSION EQUATION BY
C   SINGULAR VALUE DECOMPOSITION
C
C
C *****
C
C   SAMPLE CALL STATEMENT:
C     CALL SVDC(PRT,V,DIFF,SEISMD,SUM,NPM,MDLNTH,ITER,DF,INGESS,
C             NIMPD,FILT,ANS5,SCL,RAMDA)
C
C   INPUT:   PRT = ARRAY CONTAINING THE PARTIAL DERIVATIVE MATRIX
C             OF THE INVERSION EQUATION
C             DIFF = ARRAY CONTAINING THE DIFFERENCE MATRIX OF THE
C             INVERSION EQUATION
C             SEISMD = ARRAY CONTAINING OBSERVED DATA TRACE
C             SUM = RMS AMPLITUDE OF UNWEIGHTED DIFFERENCE MATRIX
C             NPM = NUMBER OF REFLECTION COEFFICIENTS IN MODEL
C             MDLNTH = NUMBER OF VALUES IN SYNTHETIC SEISMOGRAM
C             ITER = ITERATION NUMBER OF INVERSION LOOP
C             INGESS = ARRAY CONTAINING CURRENT VALUES FOR MODEL
C             PARAMETERS (IMPEDANCES + SCALE FACTOR IF APPL.)
C             NIMPD = NUMBER OF IMPEDANCES IN MODEL
C             FILT = ARRAY CONTAINING COEFFICIENTS OF WAVELET TO BE
C             CONVOLVED WITH REFLECTIVITY SERIES TO PRODUCE
C             SYNTHETIC SEISMOGRAM
C             ANS5 = LOGICAL VARIABLE DETERMINING WHETHER FILT
C             MUST BE CONVOLVED WITH REFLECTIVITY SERIES TO
C             PRODUCE SYNTHETIC SEISMOGRAM
C             SCL = NUMBER CODE WHICH DENOTES WHAT TYPE OF GAIN MUST
C             BE APPLIED TO DATA TO PRODUCE SYNTHETIC SEISMOGRAM
C
C   OUTPUT:  PRT = ARRAY CONTAINING U MATRIX FROM SINGULAR VALUE
C             DECOMPOSITION
C             V = ARRAY CONTAINING V MATRIX FROM SINGULAR VALUE
C             DECOMPOSITION
C             DIFF = ARRAY CONTAINING WEIGHTED CORRECTIONS FOR EACH
C             PARAMETER
C             DF = BEST DAMPING FACTOR
C             RAMDA = ARRAY CONTAINING EIGENVALUES FROM SINGULAR VALUE
C             DECOMPOSITION
C
C *****
C
C   SUBROUTINE SVDC(PRT,V,DIFF,SEISMD,SUM,NPM,MDLNTH,ITER,DF,INGESS,

```

```

& NIMPD,FILT,ANSS,SCL,RAMDA)
  DIMENSION PRT(10000),V(10000),DIFF(MDLNTH),SEISMD(MDLNTH)
  DIMENSION CK(3),ER(3),ERR(101),RAMDA(101),HOLD(100),FILT(110)
  REAL INGESS(100)
  NPR = NPM
  IF (SCL.NE.4.0) NPR = NIMPD
  WRITE (4,900)
900 FORMAT (1X,' ENTERING HYSVD')
C
C SINGULAR VALUE DECOMPOSITION
C
  CALL HYSVD(PRT,V,RAMDA,MDLNTH,NPR,.TRUE.,.TRUE.,1)
  WRITE (4,901)
901 FORMAT (1X,' EXITING HYSVD')
C
C PERFORM FIRST PART OF MATRIX MULTIPLICATION FOR SOLUTION
C
  DO 10 I=1,NPR
  HOLD(I) = 0.0
  N = I
  DO 10 J=1,MDLNTH
  HOLD(I) = HOLD(I) + PRT(N)*DIFF(J)
10 N = N + NPR
  IHOLD = ITER
  IF (ITER.GT.0 .AND. DF.GT.RAMDA(NPR)) ITER = 0
  IF (ITER.GT.0) RAMDA(NPR+1) = DF
C
C CALCULATE BEST DAMPING FACTOR
C
  CALL TESTR(RAMDA,V,NPR,CK,ERR,SUM,ITER,SEISMD,HOLD,INGESS,MDLNTH,
& NIMPD,FILT,ANSS,SCL)
  CALL SEARCH (CK,ERR,RAMDA,V,SEISMD,NPR,MDLNTH,HOLD,INGESS,
& NIMPD,FILT,ANSS,SCL)
  ITER = IHOLD
  DF = CK(2)
  WRITE (4,187) DF
187 FORMAT (1X,F12.6)
C
C PERFORM LAST PART OF MATRIX MULTIPLICATION FOR SOLUTION
C
  DO 20 I=1,NPR
  RAMDA(I) = RAMDA(I) / (RAMDA(I)**2 + DF**2)
20 HOLD(I) = RAMDA(I) * HOLD(I)
  N = 0
  DO 30 I=1,NPR
  DIFF(I) = 0.0
  DO 40 J=1,NPR
40 DIFF(I) = V(N+J)*HOLD(J) + DIFF(I)
30 N = N + NPR
  RETURN
  END

```

```

C
C
C SUBROUTINE DIFADD ADDS THE PARAMETER CORRECTION TO THE IMPEDANCE PARA
C
C *****
C
C SAMPLE CALL STATEMENT: CALL DIFADD (DIFF,INGESH,INGESS,NPR,SCL)
C
C
C INPUT:   DIFF = ARRAY CONTAINING WEIGHTED PARAMETER CORRECTIONS
C          INGRESS = MODEL PARAMETERS BEFORE CORRECTION
C          NPR = NUMBER OF MODEL PARAMETERS
C          SCL = NUMBER CODE WHICH DENOTES WHAT TYPE OF GAIN MUST
C               BE APPLIED TO DATA TO PRODUCE SYNTHETIC SEISMOGRAM
C
C
C OUTPUT:  INGESH = MODEL PARAMETERS WITH PARAMETER CORRECTIONS
C
C *****
C
C
C
C SUBROUTINE DIFADD (DIFF,INGESH,INGESS,NPR,SCL)
C DIMENSION DIFF(NPR),INGESH(100),INGESS(100)
C REAL INGRESS, INGESH
C IF (SCL.NE.4.0 .AND. DIFF(NPR-1).LE.-0.10) DIFF(NPR-1) = -0.10
C IF (SCL.NE.4.0 .AND. DIFF(NPR-1).GE.0.10) DIFF(NPR-1) = 0.10
C DO 10 I=2,NPR
C   J = I - 1
C   IF (SCL.NE.4.0 .AND. DIFF(J).LE.-0.20) DIFF(J) = -0.20
C   IF (SCL.NE.4.0 .AND. DIFF(J).GE.0.20) DIFF(J) = 0.20
C 10 INGESH(I) = INGRESS(I) * (10**DIFF(I-1))
C RETURN
C END

```

```

C
C
C
C SUBROUTINE TESTR FINDS THREE POINTS BETWEEN WHICH THE BEST
C DAMPING FACTOR LIES. THE SUBROUTINE SEARCHES THROUGH THE
C EIGENVALUES INITIALLY.
C
C
C (WITH THE EXCEPTION OF MINOR CHANGES, THE PROGRAM IS THAT
C OF DR. CHARLES STOYER, CSM)
C
C *****
C
C SAMPLE CALL STATEMENT: CALL TESTR(RAMDA,V,NPM,CK,ERR,SAVE,
C ITTER,SEISMD,HOLD,INGESS,MDLNTH,NIMPD,FILT,ANS5,SCL)
C
C
C INPUT: RAMDA = ARRAY CONTAINING EIGENVALUES FROM SINGULAR
C VALUE DECOMPOSITION
C V = ARRAY CONTAINING V MATRIX FROM SINGULAR VALUE
C DECOMPOSITION
C NPM = NUMBER OF REFLECTION COEFFICIENTS IN MODEL
C SAVE = RMS ERROR OF PREVIOUS ITERATION
C ITTER = ITERATION NUMBER OF INVERSION LOOP
C SEISMD = ARRAY CONTAINING OBSERVED DATA TRACE
C HOLD = ARRAY CONTAINING FIRST PART OF MATRIX
C MULTIPLICATION FOR SOLUTION
C INGRESS = ARRAY CONTAINING CURRENT VALUES FOR MODEL
C PARAMETERS (IMPEDANCES + SCALE FACTOR IF APPL.)
C MDLNTH = NUMBER OF VALUES IN SYNTHETIC SEISMOGRAM
C NIMPD = NUMBER OF IMPEDANCES IN MODEL
C FILT = ARRAY CONTAINING COEFFICIENTS OF WAVELET TO
C BE CONVOLVED WITH REFLECTIVITY SERIES TO
C PRODUCE SYNTHETIC SEISMOGRAM
C ANS5 = LOGICAL VARIABLE DETERMINING WHETHER FILT
C MUST BE CONVOLVED WITH REFLECTIVITY SERIES TO
C PRODUCE SYNTHETIC SEISMOGRAM
C SCL = NUMBER CODE WHICH DENOTES WHAT TYPE OF GAIN MUST
C BE APPLIED TO DATA TO PRODUCE SYNTHETIC SEISMOGRAM
C
C
C OUTPUT: CK = ARRAY CONTAINING THREE POINTS BETWEEN WHICH
C THE BEST DAMPING FACTOR LIES.
C ERR = ARRAY CONTAINING THE ERRORS FOUND BY USING THE
C THREE DAMPING FACTORS IN ARRAY CK
C
C *****
C
C
C
C SUBROUTINE TESTR(RAMDA,V,NPM,CK,ERR,SAVE,ITTER,SEISMD,HOLD,

```

```

&  INGESS,MDLNTH,NIMPD,FILT,ANS5,SCL)
REAL  INGESS(100)
DIMENSION  RAMDA(101),SEISMD(100),HOLD(100),FILT(110)
DIMENSION  CK(3),ERR(101)
DIMENSION  V(10000),A(10000)
NE = NPM
ITER = 0
WRITE (4,1111)
1111  FORMAT(' ENTERING TESTR')
C
C  SEARCH AMONG EIGENVALUES FOR THREE POINTS BETWEEN WHICH LIES
C  THE BEST DAMPING FACTOR
C
      IF (ITTER.GT.0)  NE = NE + 1
      IT=NE
      SAVE10=2.*SAVE
      INE=NE+1
      DO 10 I=1,NF
      INE=INE-1
      ERR(INE)=ERMSS(RAMDA,V,SEISMD,RAMDA(INE),NPM,MDLNTH,HOLD,INGESS,
&  NIMPD,FILT,ANS5,SCL)
      IF(ERR(INE) .GT. SAVE10)GO TO 20
      IF(.9999*ERR(INE) .GE. ERR(IT))GO TO 10
      IT=INE
10  CONTINUE
20  CK(2)=RAMDA(IT)
      ER2=ERR(IT)
      IF(IT .NE. NE)GO TO 30
C
C  IF MINIMUM EIGENVALUE IS NOT SMALL ENOUGH, GENERATE SMALLER NUMBER
C
      CK(1)=CK(2)*.2
      ER1=ERMSS(RAMDA,V,SEISMD,CK(1),NPM,MDLNTH,HOLD,INGESS,NIMPD,
&  FILT,ANS5,SCL)
25  IF (ER1.GE.ER2)  GO TO 40
      CK(3) = CK(2)
      ER3 = ER2
      CK(2) = CK(1)
      ER2 = ER1
      CK(1) = CK(2)*0.2
      ER1 = ERMSS(RAMDA,V,SEISMD,CK(1),NPM,MDLNTH,HOLD,INGESS,NIMPD,
&  FILT,ANS5,SCL)
      ITER = ITER + 1
      IF(ITER.LT.7)  GO TO 25
      GO TO 40
30  CK(1)=RAMDA(IT+1)
      ER1=ERR(IT+1)
40  IF(IT .NE. 1)  GO TO 50
C
C  IF LARGEST EIGENVALUE IS NOT LARGE ENOUGH, GENERATE LARGER VALUE
C

```

```
      CK(3)=CK(2)*10.
      ER3=ERMSS(RAMDA,V,SEISMD,CK(3),NPM,MDLNTH,HOLD,INGESS,NIMPD,
&      FILT,ANS5,SCL)
      GO TO 60
50    IF (ITER.GE.1) GO TO 60
      CK(3)=RAMDA(IT-1)
      ER3=ERR(IT-1)
C
C  INSERT ERRORS INTO ARRAY
C
50    ERR(1)=ER1
      ERR(2)=ER2
      ERR(3)=ER3
      RETURN
      END
```

```

C
C
C
C SUBROUTINE SEARCH TAKES THE THREE POINTS BETWEEN WHICH LIES THE
C BEST DAMPING FACTOR AND INTERPOLATES TO FIND THE LEAST ERROR
C DAMPING FACTOR
C
C
C (WITH THE EXCEPTION OF MINOR CHANGES, THE PROGRAM IS THAT
C OF DR. CHARLES STOYER, CSM)
C
C *****
C
C SAMPLE CALL STATEMENT: CALL SEARCH(CK,ER,AMDA,V,SEISMD,NPM,
C MDLNTH,HOLD,INGESS,NIMPD,FILT,ANSS,SCL)
C
C
C INPUT: CK = ARRAY CONTAINING THREE POINTS BETWEEN WHICH THE
C BEST DAMPING FACTOR LIES
C ER = ARRAY CONTAINING THE ERRORS FOUND BY USING THE
C THREE DAMPING FACTORS IN ARRAY CK
C RAMDA = ARRAY CONTAINING EIGENVALUES FROM SINGULAR
C VALUE DECOMPOSITION
C V = ARRAY CONTAINING V MATRIX FROM SINGULAR VALUE
C DECOMPOSITION
C SEISMD = ARRAY CONTAINING OBSERVED DATA TRACE
C NPM = NUMBER OF REFLECTION COEFFICIENTS IN MODEL
C MDLNTH = NUMBER OF VALUES IN SYNTHETIC SEISMOGRAM
C HOLD = ARRAY CONTAINING FIRST PART OF MATRIX
C MULTIPLICATION FOR SOLUTION
C INGESS = ARRAY CONTAINING CURRENT VALUES FOR MODEL
C PARAMETERS (IMPEDANCES + SCALE FACTOR IF APPL.)
C NIMPD = NUMBER OF IMPEDANCES IN MODEL
C FILT = ARRAY CONTAINING COEFFICIENTS OF WAVELET TO
C PRODUCE SYNTHETIC SEISMOGRAM
C ANSS = LOGICAL VARIABLE DETERMINING WHETHER FILT
C MUST BE CONVOLVED WITH REFLECTIVITY SERIES TO
C PRODUCE SYNTHETIC SEISMOGRAM
C SCL = NUMBER CODE WHICH DENOTES WHAT TYPE OF GAIN MUST
C BE APPLIED TO DATA TO PRODUCE SYNTHETIC SEISMOGRAM
C
C
C OUTPUT: CK = ARRAY CONTAINING THREE POINTS WITH THE CENTER
C POINT BEING THE BEST DAMPING FACTOR
C ER = ARRAY CONTAINING THE ERRORS FOUND BY USING THE
C THREE DAMPING FACTORS IN ARRAY CK
C
C *****
C
C
C
C

```

```

      SUBROUTINE SEARCH(CK,ER,RAMDA,V,SEISMD,NPM,MDLNTH,HOLD,INGESS,
&  NIMPD,FILT,ANS5,SCL)
      IMPLICIT LOGICAL(L)
      REAL INGESS(100)
      DIMENSION RAMDA(101),SEISMD(100),HOLD(100),FILT(110)
      DIMENSION CK(3),ER(3)
      DIMENSION V(10000),A(10000)
      WRITE(4,1111)
1111  FORMAT(' ENTERING SEARCH')
      ITER=0
      SVP=0.
      SVM=0.

C
C  CHECK TO SEE IF THE MIDDLE POINT OF THE THREE POINTS BETWEEN WHICH
C  THE BEST DAMPING FACTOR LIES IS THE LEAST ERROR POINT
C
      LTREND=LMOND(ER)
      IF(LTREND)GO TO 100
10    CONTINUE

C
C  CHECK TO SEE IF CONVERGENCE IS REACHED IN DETERMINING
C  MINIMUM ERROR POINT
C
      IF(LSHAL(ER))GO TO 200
      ITER=ITER+1
      IF(ITER .GT. 9)GO TO 200

C
C  GENERATES INTERMEDIATE POINTS IN SEARCH MINIMUM
C
      CKP=SQRT(SQRT(CK(3)*CK(2)*CK(2)*CK(2)))
      CKM=(CK(1)*CK(1)*CK(2)*CK(2)*CK(2))**.2

C
C  CALCULATES ERRORS FOR INTERPOLATED POINTS
C
      IF(CKP .NE. SVP)ERP=ERMSS(RAMDA,V,SEISMD,CKP,NPM,MDLNTH,
&  HOLD,INGESS,NIMPD,FILT,ANS5,SCL)
      SVP=CKP
      IF(CKM .NE. SVM)ERM=ERMSS(RAMDA,V,SEISMD,CKM,NPM,MDLNTH,
&  HOLD,INGESS,NIMPD,FILT,ANS5,SCL)
      SVM=CKM
      IF(ERP .GT. ERM)GO TO 20
      IF(ERP .GT. ER(2))GO TO 15

C
C  ACCORDING TO WHERE LEAST ERROR POINT IS,
C  ASSIGNS NEW VALUES TO THREE POINTS AND THEIR ERRORS
C
      CK(1)=CK(2)
      ER(1)=ER(2)
      CK(2)=CKP
      ER(2)=ERP
      GO TO 10

```

```
15      CK(1)=CKM
        ER(1)=ERM
        CK(3)=CKP
        ER(3)=ERP
        GO TO 10
20      IF(ERM .GT. ER(2))GO TO 15
        CK(3)=CK(2)
        ER(3)=ER(2)
        CK(2)=CKM
        ER(2)=ERM
        GO TO 10
100     CONTINUE
        IF(ER(2) .LT. ER(1))GO TO 110
        CK(2)=CK(1)
        ER(2)=ER(1)
        RETURN
110     IF( ER(2) .LT. ER(3))RETURN
        CK(2)=CK(3)
        ER(2)=ER(3)
200     RETURN
        END
```



```

C
C
C  FUNCTION ERMSS COMPUTES THE RMS ERROR FOR DIFFERENT DAMPING FACTORS
C
C *****
C
C  SAMPLE CALL STATEMENT:  ERMSS(RAMDA,V,SEISMD,DF,NPR,MDLNTH,
C                          HOLD,INGESS,NIMPD,FILT,ANS5,SCL)
C
C
C  INPUT:      RAMDA = ARRAY CONTAINING EIGENVALUES FROM SINGULAR
C              VALUE DECOMPOSITION
C              V = ARRAY CONTAINING V MATRIX FROM SINGULAR VALUE
C              DECOMPOSITION
C              SEISMD = ARRAY CONTAINING OBSERVED DATA TRACE
C              DF = DAMPING FACTOR
C              NPR = NUMBER OF PARAMETERS CORRECTIONS
C              MDLNTH = NUMBER OF VALUES IN SYNTHETIC SEISMOGRAM
C              HOLD = ARRAY CONTAINING FIRST PART OF MATRIX
C                   MULTIPLICATION FOR SOLUTION
C              INGRESS = ARRAY CONTAINING CURRENT VALUES FOR MODEL
C                   PARAMETERS (IMPEDANCES + SCALE FACTOR IF APPL.)
C              NIMPD = NUMBER OF IMPEDANCES IN MODEL
C              FILT = ARRAY CONTAINING COEFFICIENTS OF WAVELET TO
C                   PRODUCE SYNTHETIC SEISMOGRAM
C              ANS5 = LOGICAL VARIABLE DETERMINING WHETHER FILT
C                   MUST BE CONVOLVED WITH REFLECTIVITY SERIES TO
C                   PRODUCE SYNTHETIC SEISMOGRAM
C              SCL = NUMBER CODE WHICH DENOTES WHAT TYPE OF GAIN MUST
C                   BE APPLIED TO DATA TO PRODUCE SYNTHETIC SEISMOGRAM
C
C
C  OUTPUT:     ERMSS = ERROR ASSOCIATED WITH INPUT DAMPING FACTOR
C
C *****
C
C  FUNCTION ERMSS(RAMDA,V,SEISMD,DF,NPR,MDLNTH,HOLD,INGESS,NIMPD,
C &  FILT,ANS5,SCL)
C  DIMENSION RAMDA1(100),DELTAP(100),HOLD1(100),INGESH(100)
C  DIMENSION SEISIC(150),RCLG(150),FILT(110)
C  DIMENSION RAMDA(101),V(10000),SEISMD(MDLNTH),HOLD(100)
C  DIMENSION INGRESS(100)
C  REAL INGRESS,INGESH
C  DATA ANO/'NO'/
C  NPM = NIMPD - 1
C  N = 0
C
C  SOLVE FOR PARAMETER CORRECTIONS WITH INPUT DAMPING FACTOR
C

```

```

      DO 10 I=1,NPR
      RAMDA1(I) = RAMDA(I) / (RAMDA(I)**2 + DF**2)
10  HOLD1(I) = RAMDA1(I)*HOLD(I)
      DO 20 I=1,NPR
      DELTAP(I) = 0.0
      DO 30 J=1,NPR
30  DELTAP(I) = V(N+J)*HOLD1(J) + DELTAP(I)
20  N = N + NPR
      NPR1 = NIMPD
      IF (SCL.NE.4.0) NPR1 = NIMPD + 1
C
C  ADD CORRECTIONS TO CURRENT MODEL PARAMETERS
C
      CALL DIFADD(DELTAP,INGESH,INGESS,NPR1,SCL)
C
C  CALCULATE FORWARD SOLUTION
C
      CALL IPDRC (INGESH,NIMPD,RCLG)
      CALL FWRD(RCLG,NPM,SEISIC,MDLNTH)
      IF (ANS5.EQ.AND) GO TO 25
      CALL FOLD(MDLNTH,SEISIC,35,FILT,LC,RCLG)
      DO 21 K=1,MDLNTH
21  SEISIC(K) = RCLG(K)
25  SCALFC = INGESS(NPR1)
      IF (SCL.NE.4.0) CALL SCAL(SEISIC,SCALFC,SCL,MDLNTH)
C
C  CALCULATE ERROR BETWEEN OBSERVED DATA TRACE
C  AND CALCULATED SYNTHETIC
C
      CALL ERMSC(SEISMD,SEISIC,DELTAP,MDLNTH,ERMSS)
      RETURN
      END

```

```

C ***** START OF SVD *****
C
C SUBROUTINE HYSVD (A,V,S,M,N,WITHU,WITHV,IW)
C
C INTEGER M, N, P
C REAL R, W, CS, SN, TOL, F, X, EPS, G, T, Y
C REAL ETA, H, Q, Z
C INTEGER I, J, K, L, L1, N1, NP
C LOGICAL WITHU, WITHV
C DIMENSION S(N)
C DIMENSION A(10000),V(10000)
C
C -----
C
C THIS IS A TRANSLATION OF A CDC 6600 FORTRAN PROGRAM TO IBM 360
C FORTRAN IV. THIS SUBROUTINE USES SHORT PRECISION ARITHMETIC.
C A LONG PRECISION VERSION IS AVAILABLE UNDER THE NAME EDSVDC.
C
C THIS SUBROUTINE REPLACES EARLIER SUBROUTINES WITH THE SAME NAME,
C 689 6 % 80% S OF A COMPLEX ARITHMETIC PROGRAM, PUBLISHED
C AS ALGORITHM 358. THIS CURRENT PROGRAM IS FASTER, MORE ACCURATE
C AND LESS OBSCURE IN DESCRIBING ITS CAPABILITIES.
C
C ORIGINAL PROGRAMMER= R. C. SINGLETON
C 360 VERSION BY= J. G. LEWIS
C LAST REVISION OF THIS SUBROUTINE= 4 DECEMBER 1973
C
C -----
C
C ADDITIONAL SUBROUTINE NEEDED= ROTATE
C
C -----
C
C THIS SUBROUTINE COMPUTES THE SINGULAR VALUE DECOMPOSITION
C OF A REAL M*N MATRIX A, I.E. IT COMPUTES MATRICES U, S, AND V
C SUCH THAT
C
C 
$$A = U * S * VT,$$

C WHERE
C U IS AN M*N MATRIX AND  $UT*U = I$ , (UT=TRANSDPOSE
C OF U),
C V IS AN N*N MATRIX AND  $VT*V = I$ , (VT=TRANSDPOSE
C OF V),
C AND S IS AN N*N DIAGONAL MATRIX.
C
C DESCRIPTION OF PARAMETERS=
C
C A = REAL ARRAY. A CONTAINS THE MATRIX TO BE DECOMPOSED.
C THE ORIGINAL DATA ARE LOST. IF WITHV=.TRUE., THEN
C THE MATRIX U IS COMPUTED AND STORED IN THE ARRAY A.

```

```

C
C   M,N = INTEGER VARIABLES.  THE NUMBER OF ROWS AND COLUMNS
C   IN THE MATRIX STORED IN A.  (NG=MG=100.  IF IT IS
C   NECESSARY TO SOLVE A LARGER PROBLEM, THEN THE
C   AMOUNT OF STORAGE ALLOCATED TO THE ARRAY T MUST
C   BE INCREASED ACCORDINGLY.)  IF MLT N , THEN EITHER
C   TRANSPOSE THE MATRIX A OR ADD ROWS OF ZEROS TO
C   INCREASE M TO N.
C
C   P = INTEGER VARIABLE.  IF PC0, THEN COLUMNS N+1, . . . ,
C   N+P OF A ARE ASSUMED TO CONTAIN THE COLUMNS OF AN M*P
C   MATRIX B.  THIS MATRIX IS MULTIPLIED BY UT, AND UPON
C   EXIT, A CONTAINS IN THESE SAME COLUMNS THE N*P MATRIX
C   UT*B.  (PC=0)
C
C   WITHU, WITHV = LOGICAL VARIABLES.  IF WITHU=.TRUE., THEN
C   THE MATRIX U IS COMPUTED AND STORED IN THE ARRAY A.
C   IF WITHV=.TRUE., THEN THE MATRIX V IS COMPUTED AND
C   STORED IN THE ARRAY V.
C
C   S = REAL ARRAY.  S(1), . . . , S(N) CONTAIN THE DIAGONAL
C   ELEMENTS OF THE MATRIX S ORDERED SO THAN S(I)&S(I+1),
C   I=1, . . . , N-1.
C
C   V = REAL ARRAY.  V CONTAINS THE MATRIX V.  IF WITHU
C   AND WITHV ARE NOT BOTH =.TRUE., THEN THE ACTUAL
C   PARAMETER CORRESPONDING TO A AND V MAY BE THE SAME.
C
C   THIS SUBROUTINE IS A REAL VERSION OF A FORTRAN SUBROUTINE
C   BY BUSINGER AND GOLUB, ALGORITHM 359= SINGULAR VALUE
C   DECOMPOSITION OF A COMPLEX MATRIX, COMM. ACM, V. 12,
C   NO. 10, PP. 564-565 (OCT. 1969).
C   WITH REVISIONS BY RC SINGLETON, MAY 1972.
C   -----
C
C   DIMENSION  T(250)
C   DATA P/C/
C
C   ECLIPSE CONSTANTS
C   DATA ETA,TOL/1.9E-7,1.1E-30/
C
C   DATA ETA,TOL /1.5E-15,1.E-250/
C
C   ETA (16**-6) AND TOL (16**-59) ARE MACHINE DEPENDENT CONSTANTS
C   FOR IBM 360/370 COMPUTERS (SHORT FORM ARITHMETIC).
C   ETA IS THE MACHINE EPSILON (RELATIVE ACCURACY):
C   TOL IS THE SMALLEST REPRESENTABLE REAL DIVIDED BY ETA.
C
C   NP = N + P
C   N1 = N + 1

```

```

C
C   HOUSEHOLDER REDUCTION TO BIDIAGONAL FORM
      G = 0.0
      EPS = 0.0
      L = 1
10   T(L) = G
      K = L
      L = L + 1
C
C   ELIMINATION OF A(I,K), I=K+1, . . . , M
      S(K) = 0.0
      Z = 0.0
      KK = (K-1)*N + K
      IK = KK
      DO 20 I = K,M
          Z = Z + A(IK)**2
20   IK = IK + N
      IF (Z.LT.TOL) GOTO 50
      G = SQRT(Z)
      F = A(KK)
      IF (F.GE.0.0) G = - G
      S(K) = G
      H = G * (F - G)
      A(KK) = F - G
      IF (K.EQ.NP) GOTO 50
      KS = (K-1)*N
      KSAV = KS + K
      KL = KS + L
      DO 41 J = L,NP
          F = 0
          IK = KSAV
          IJ = KL
          DO 30 I = K,M
              F = F + A(IK)*A(IJ)
              IK = IK + N
              IJ = IJ + N
30   F = F/H
          IJ = KL
          IK = KSAV
          DO 40 I = K,M
              A(IJ) = A(IJ) + F*A(IK)
              IJ = IJ + N
40   IK = IK + N
41   KL = KL + 1
C
C   ELIMINATION OF A(K,J), J=K+2, . . . , N
50   EPS = AMAX1(EPS,ABS(S(K)) + ABS(T(K)))
      IF (K.EQ.N) GOTO 100
      G = 0.0
      Z = 0.0
      KL = (K-1)*N + L

```

```

      KJ = KL
      DO 60 J = L,N
        Z = Z + A(KJ)**2
60     KJ = KJ + 1
      IF (Z.LT.TOL) GOTO 10
      G = SQRT(Z)
      F = A(KL)
      IF (F.GE.0.0) G = - G
      H = G * (F - G)
      A(KL) = F - G
      KJ = KL
      DO 70 J = L,N
        T(J) = A(KJ)/H
70     KJ = KJ + 1
      LL = (L-1)*N + L
      DO 91 I = L,M
        F = 0
        KJ = KL
        IJ = LL
        DO 80 J = L,N
          F = F + A(KJ)*A(IJ)
          KJ = KJ + 1
80     IJ = IJ + 1
        IJ = LL
        DO 90 J = L,N
          A(IJ) = A(IJ) + F*T(J)
90     IJ = IJ + 1
91     LL = LL + N
C
      GOTO 10
C
C     TOLERANCE FOR NEGLIGIBLE ELEMENTS
100  EPS = EPS*ETA
C
C     ACCUMULATION OF TRANSFORMATIONS
      IF (.NOT.WITHV) GOTO 160
      K = N
      GOTO 140
110  IF (T(L).EQ.0.0) GOTO 140
      KL = (K-1)*N + L
      H = A(KL)*T(L)
      LL = (L-1)*N + L
      DO 131 J = L,N
        Q = 0
        IJ = LL
        KI = KL
        DO 120 I = L,N
          Q = Q + A(KI)*V(IJ)
          IJ = IJ + N
120     KI = KI + 1
        Q = Q/H

```

```

        IJ = LL
        KI = KL
        DO 130 I = L,N
            V(IJ) = V(IJ) + Q*A(KI)
            IJ = IJ + N
130         KI = KI + 1
131         LL = LL + 1
140         KJ = (K-1)*N + 1
        DO 150 J = 1,N
            V(KJ) = 0
150         KJ = KJ + 1
        KK = (K-1)*N + K
        V(KK) = 1.0
        L = K
        K = K - 1
        IF (K.NE.0) GOTO 110
C
160 K = N
    IF (.NOT.WITHU) GOTO 230
    G = S(N)
    IF (G.NE.0.0) G = 1.0/G
    GO TO 210
170     KJ = (K-1)*N + L
    DO 180 J = L,N
        A(KJ) = 0
180     KJ = KJ + 1
    G = S(K)
    IF (G.EQ.0.0) GOTO 210
    KK = (K-1)*N + K
    H = A(KK)*G
    LSAV = (L-1)*N
    LL = LSAV + L
    KL = KK + L - K
    DO 201 J = L,N
        Q = 0
        IK = LSAV + K
        IJ = LL
        DO 190 I = L,M
            Q = Q + A(IK)*A(IJ)
            IK = IK + N
190         IJ = IJ + N
        Q = Q/H
        IJ = KL
        IK = KK
        DO 200 I = K,M
            A(IJ) = A(IJ) + Q*A(IK)
            IJ = IJ + N
200         IK = IK + N
            LL = LL + 1
201         KL = KL + 1
    G = 1.0/G

```

```

210   KK = (K-1)*N + K
      JK = KK
      DO 220 J = K,M
          A(JK) = A(JK)*G
220   JK = JK + N
      A(KK) = A(KK) + 1.0
      L = K
      K = K - 1
      IF (K.NE.0) GOTO 170
C
C   QR DIAGONALIZATION
      K = N
C
C   TEST FOR SPLIT
230   L = K
240   IF (ABS(T(L)).LE.EPS) GOTO 290
      L = L - 1
      IF (ABS(S(L)).GT.EPS) GOTO 240
C
C   CANCELLATION
      CS = 0.0
      SN = 1.0
      L1 = L
      L = L + 1
      DO 280 I = L,K
          F = SN*T(I)
          T(I) = CS*T(I)
          IF (ABS(F).LE.EPS) GOTO 290
          H = S(I)
          W = SQRT(F*F + H*H)
          S(I) = W
          CS = H/W
          SN = - F/W
          IF (WITHU) CALL HYROT(A(L1), A(I), CS, SN, M, N)
          IF (NP.EQ.N) GOTO 280
          L1J = (L1-1)*N + N1
          IJ = (I-1)*N + N1
          DO 270 J = N1,NP
              Q = A(L1J)
              R = A(IJ)
              A(L1J) = Q*CS + R*SN
              A(IJ) = R*CS - Q*SN
              L1J = L1J + 1
270   IJ = IJ + 1
280   CONTINUE
C
C   TEST FOR CONVERGENCE
290   W = S(K)
      IF (L.EQ.K) GOTO 360
C
C   ORIGIN SHIFT

```

```

X = S(L)
Y = S(K-1)
G = T(K-1)
H = T(K)
F = ((Y - W)*(Y + W) + (G - H)*(G + H))/(2.0*H*Y)
G = SQRT(F*F + 1.0)
IF (F.LT.0.0) G = - G
F = ((X - W)*(X + W) + (Y/(F + G) - H)*H)/X
C
C
QR STEP
CS = 1.0
SN = 1.0
L1 = L + 1
DO 350 I = L1,K
  G = T(I)
  Y = S(I)
  H = SN*G
  G = CS*G
  W = SQRT(H*H + F*F)
  T(I-1) = W
  CS = F/W
  SN = H/W
  F = X*CS + G*SN
  G = G*CS - X*SN
  H = Y*SN
  Y = Y*CS
  IF (WITHV) CALL HYROT(V(I-1), V(I), CS, SN, N, N)
  W = SQRT(H*H + F*F)
  S(I-1) = W
  CS = F/W
  SN = H/W
  F = CS*G + SN*Y
  X = CS*Y - SN*G
  IF (WITHU) CALL HYROT(A(I-1), A(I), CS, SN, M, N)
  IF (N.EQ.NP) GOTO 350
  IM1J = (I-2)*N + N1
  IJ = IM1J + N
  DO 340 J = N1, NP
    Q = A(IM1J)
    R = A(IJ)
    A(IM1J) = Q*CS + R*SN
  A(IJ) = R*CS - Q*SN
  IM1J = IM1J + 1
340 IJ = IJ + 1
350 CONTINUE
C
  T(L) = 0.0
  T(K) = F
  S(K) = X
  GOTO 230
C

```

```

C      CONVERGENCE
360    IF (W.GE.0.0) GOTO 380
        S(K) = - W
        IF (.NOT.WITHV) GOTO 380
        JK = K
        DO 370 J = 1,N
            V(JK) = - V(JK)
370    JK = JK + N
380    K = K - 1
        IF (K.NE.0) GO TO 230

C
IF (IW.EQ.2) RETURN
C
SORT SINGULAR VALUES
DO 450 K = 1,N
    G = -1.0
    DO 390 I = K,N
        IF (S(I).LT.G) GOTO 390
        G = S(I)
        J = I
390    CONTINUE
    IF (J .EQ. K) GO TO 450
    S(J) = S(K)
    S(K) = G
    IF (.NOT.WITHV) GOTO 410
    IK = K
    IJ = J
    DO 400 I = 1,N
        Q = V(IJ)
        V(IJ) = V(IK)
        V(IK) = Q
        IK = IK + N
400    IJ = IJ + N
410    IF (.NOT.WITHU) GOTO 430
        IJ = J
        IK = K
        DO 420 I = 1,M
            Q = A(IJ)
            A(IJ) = A(IK)
            A(IK) = Q
            IJ = IJ + N
420    IK = IK + N
430    IF (N.EQ.NP) GOTO 450
        JI = (J-1)*N + N1
        KI = (K-1)*N + N1
        DO 440 I = N1,NP
            Q = A(JI)
            A(JI) = A(KI)
            A(KI) = Q
            JI = JI + 1
440    KI = KI + 1
450    CONTINUE

```

C

RETURN
END

```
      SUBROUTINE  HYROT  (X, Y, CS, SN, N, NCOL)
      INTEGER N
      REAL      X(10000), Y(10000), CS, SN
C
C
      REAL      XX
      INTEGER J
C
C
      NM = (N-1)*NCOL + 1
      DO 10 J = 1, NM, NCOL
          XX = X(J)
          X(J) = XX*CS + Y(J)*SN
10      Y(J) = Y(J)*CS - XX*SN
      RETURN
      END
```

BIBLIOGRAPHY

- Bamberger, A.; Chavent, G.; Hemon, Ch.; and Lailly, P., 1978, Inversion of normal incidence seismograms: SEG preprint, Box 3098, Tulsa, OK.
- Bamberger, A.; Chavent, G.; Lailly, P., 1977, Une application de la theorie du controle a un probleme inverse de sismique: Ann. Geophys., t 33, fasc. 1/2, p 183-200
- Braile, L. W., 1973, Inversion of crustal seismic refraction and reflection data: Jour. of Geophys. Research, V. 78, No. 32, p. 7738-7744
- Gjevik, B.; Nilsen, A.; and Høyen, J., 1976, An attempt at the inversion of reflection data: Geophysical Prospecting, V. 24, p. 492-505
- Gjevik, B. and Nilsen, A., 1978, Inversion of reflection data: Geophysical Prospecting, V. 26, p. 421-432
- Golub, G. H., and Reinsch, C., 1970, Singular value decomposition and least squares solutions: Numer. Math., V. 11, p. 403-420
- Goupillaud, P. L., 1961, An approach to inverse filtering of near-surface layer effects from seismic records: Geophysics, V. 26, p. 754-760
- Hilterman, F. J.; Schneider, W. A.; Alam, A., 1977, $W(t)$, The wavelet as a model component: Seg. 2, SEG Continuing Education Symposium.
- Johansen, H. K., 1977, A man/computer interpretation system for resistivity sounding over a horizontally stratified earth: Geophys. Prosp., V. 25, p. 667-691
- Kunetz, G., 1963, Quelques exemples d'analyse d'enregistrements sismiques, Geophysical Prospecting, V. 11, No. 4, p. 409
- Lancsoz, C., 1961, Linear differential operators: London, D. Van Nostrand Co. Ltd., p. 100-161
- Larner, K.; Robinson, J.; Stone, D.; Treitel, S., 1977, $R(t)$, reflectivity as a model component, Seg. 3, SEG School, Houston, TX.

- Lavergne, M., and Willm, C., 1977, Inversion of seismograms and pseudo velocity logs, Geophysical Prospecting, V. 25, p. 231-250
- Parker, R. L., 1977, Understanding inverse theory: Ann. Rev. Earth Planet. Sci., V. 5, p. 35-64
- Pelissier, M. A., 1979, SH channel waves: Parameter sensitivity and linear inversion for a deep coal seam: M.Sc. Thesis, T-2199, Colorado School of Mines, Golden, CO.
- Robinson, Enders A., 1978, Multichannel time series analysis with digital computer programs: San Francisco, Holden Day.
- Stoyer, C., 1978, Damping factor determination: Golden, Colorado, Personal communication.
- Wiggins, R. A.; Lerner, K. L.; Wisecup, R.D., 1976, Residual statics analysis as general linear inverse problem: Geophysics, v.41, no.5, p. 922-938
- Wiggins, R. A., 1972, The general linear inverse problem: Implication of surface waves and free oscillations for earth structure: Rev. Geophys. Space Phys., v.10, p. 251-285
- Wuenschel, P. C., 1960, Seismogram synthesis including multiples and transmission coefficients: Geophysics, v.25, p. 106-129.