Report of Investigations 9159

A Tomographic Computer Program With Constraints To Improve Reconstructions for Monitoring In Situ Mining Leachate

By Daryl R. Tweeton

UNITED STATES DEPARTMENT OF THE INTERIOR
Donald Paul Hodel, Secretary

BUREAU OF MINES
David S. Brown, Acting Director

Library of Congress Cataloging in Publication Data:

Tweeton, Daryl R.
A tomographic computer program with constraints to improve reconstructions for monitoring in situ mining leachate.

(Report of investigations ; 9159)

Bibliography: p. 30.

Supt. of Docs. no.: I 28.23: 9159.

1. Hydrometallurgy—Data processing. 2. Leaching—Data processing. 3. Tomography. I. Title. II. Series: Report of investigations (United States. Bureau of Mines) ; 9159.

TN23.U43 [TN688] 622 s [622'.2] 87-600412

CONTENTS

ILLUSTRATIONS

UNIT OF MEASURE ABBREVIATIONS USED IN THIS REPORT

kb	kilobyte	MHz	megahertz
km/s	kilometer per second	ms	millisecond
m	meter	pct	percent

A TOMOGRAPHIC COMPUTER PROGRAM WITH CONSTRAINTS TO IMPROVE RECONSTRUCTIONS FOR MONITORING IN SITU MINING LEACHATE

By Daryl R. Tweeton[1]

ABSTRACT

The Bureau of Mines is investigating possible applications of geophysics to monitor leachate during in situ mining. Tomographic reconstruction of seismic or electromagnetic crosshole data appears applicable. The tomographic program BOMTOM (Bureau of Mines tomography) was written and then used to investigate reconstructions with synthetic travel time data. The calculations confirmed that without constraints, crosshole data do not provide unique solutions. The solution is not determined solely by the data, but is also influenced by the velocity guesses starting the iterative solution procedure.

The effectiveness of constraints in combatting nonuniqueness was examined. The requirement that the velocity not vary laterally at the top and bottom of the pixel grid was effective. This constraint is realistic when the top and bottom of the grid represent dry and saturated regions, respectively. Limiting the maximum velocity improved reconstructions, but may not be applicable in practice.

BOMTOM uses the simultaneous iterative reconstruction technique (SIRT) with straight ray paths. The program is efficient in use of memory, and can be run on a personal computer. A source code in FORTRAN is listed. A source code in BASICA and compiled executable codes in both languages are available from the Bureau's Twin Cities (MN) Research Center.

[1]Research physicist, Twin Cities Research Center, Bureau of Mines, Minneapolis, MN.

INTRODUCTION

One of the goals of the Bureau of Mines in situ mining research program is to develop improved methods of monitoring the movement of the leachate. The current research emphasis is on in situ mining in fractured rock. Determining the location of the leachate is important for both economic and environmental reasons. Knowing the shape of the leachate-saturated region is necessary to select the spacing between injection wells that will give adequate coverage with a minimum number of wells. Determining that shape in a relatively small test area can provide the basis for selecting the well spacing in a larger production area. Improved methods of monitoring the location of the leachate can give an earlier warning of the start of its migration to areas where it might be environmentally harmful.

The conventional method for locating leachate during in situ mining is to sample monitor wells. However, wells are expensive to construct in deep, hard-rock deposits, so drilling enough wells to show the shape of the leachate-saturated region is costly. Therefore, the Bureau is evaluating geophysical methods for locating the leachate. A geophysical system could provide more reliable leachate monitoring in a more efficient and economical manner.

The depth at which in situ mining is generally conducted makes crosshole geophysical techniques more promising than surface techniques. However, most crosshole techniques give either sufficient resolution or penetration, but not both. For in situ mining of deep deposits, the distance between injection wells will probably be at least 30 m. The needed penetration is at least the well-to-well spacing, but resolution no larger than a tenth of that spacing is desirable. When used with geophysical methods having sufficient penetration, tomographic reconstruction provides a mathematical method of obtaining adequate resolution.

The applicability of geophysical methods differs for in situ mining above and below the water table. For use above the water table, the Bureau is considering seismic crosshole surveying with tomographic reconstruction of the velocity distribution between the boreholes. The degree of saturation affects the seismic velocity. For use below the water table, a method that responds to the difference between leachate and ground water is needed. Certain electromagnetic methods appear to be applicable, and will be evaluated further in future research.

The Bureau has investigated other techniques. In previous tests (1)[2] at an in situ uranium mine, neither four-terminal resistivity measurements nor audiomagnetotellurics reliably showed the location of leachate replacing ground water.

Tomography has been applied to geophysical investigations by a number of researchers. Ramirez (2) reported the application of tomography to electromagnetic (5-40 MHz) attenuation crosshole data with separations up to 30 m. He used the same SIRT tomographic technique as Dines and Lytle (3) used for analyzing electromagnetic (50 MHz) crosshole data. Dines and Lytle measured both attenuation and velocity, and found attenuation to be more diagnostic. Applications to seismic reflection data were made by Chiu (4) and by Bishop (5). Analyses of seismic crosshole data were performed by Phillips (6), Gustavsson (7), and Peterson (8-9). Foss (10) applied tomography to detection of coal mine hazards with ground-penetrating-radar, but reported that tomography did not conclusively show the clay vein of interest.

The objective of the research described in this report was to develop and test a tomographic computer program suitable for application to monitoring of leachate migration during in situ mining. The ability to run the program on a personal computer was desirable so that small mining companies could use the procedure on

[2]Underlined numbers in parentheses refer to items in the list of references preceding the appendixes at the end of this report.

their own facilities. The program had to be easy to use so people could perform the calculations with a minimum of delay for familiarization with the program.

The decision to write a tomographic program followed unsuccessful attempts to find a suitable program. Available programs were unsuitable for various reasons, such as requiring excessive computer memory, being unreliable with imperfect data, or not providing tests to check the uniqueness of solutions. Therefore, a new tomographic program BOMTOM (Bureau of Mines tomography) was written to serve the Bureau's needs and those of mining companies wishing to apply tomography. It was designed to offer options needed for in situ mining applications and yet run on a personal computer. It was intended for seismic data analysis, and so was tested with synthetic data representing in situ mining above the water table. The program was refined by options that the tests indicated were needed, and also by options to make it easier to use as recommended by users of preliminary versions. The resulting program offers advantages in computer memory requirements, execution time, ease of use, and options for testing and dealing with nonuniqueness.

ACKNOWLEDGMENTS

The author wishes to thank John Peterson, University of California, Berkeley, for providing tomographic data and the program published in his thesis; Nigel Wattrus, Arco Oil and Gas, Plano, TX, for providing tomographic programs used in his thesis research; and Leo Akkila, Geoseismo, for providing the VIBROVISION demonstration program and results. The ability to compare Bureau results with those from other programs was very helpful in verifying BOMTOM.

TOMOGRAPHIC METHODS

Tomographic reconstruction of geophysical data is receiving more interest as a method of obtaining adequate resolution where transmitters and receivers must be widely separated. The general procedures involved in tomography have been described elsewhere (3, 8-9, 11-13). Only a brief discussion, as it relates to this special application, will be given.

Tomographic reconstruction as applied to crosshole data is a mathematical process for constructing a two-dimensional representation of a field such as seismic velocity between the boreholes. Figure 1 shows a simplified diagram of the transmitter and receiver locations. The measurement procedure consists of setting the transmitter at one position, stepping the receiver through the receiver positions, then moving the transmitter to its next position and repeating the series of receiver locations. This procedure is repeated until all of the desired ray paths have been generated. Repeating the receiver positions precisely is not necessary if the program provides sufficient flexibility for the input data format. (See "Discussion of Tomographic Program BOMTOM" section.)

A helpful summary of the advantages and disadvantages of the various tomographic mathematical methods was given by Worthington (11). He grouped them into four categories:

1. Matrix inversion.
2. Fourier transform methods.
3. Convolutional methods or filtered back-projection.
4. Algebraic reconstruction techniques.

Matrix inversion involves dividing the region of interest into cells (pixels), setting up linear equations relating the velocity in the individual pixels to the experimental travel times, and then solving the system of equations by direct matrix inversion techniques. This method cannot be applied where there is no exact solution, such as when there are more travel times than pixels and the data are

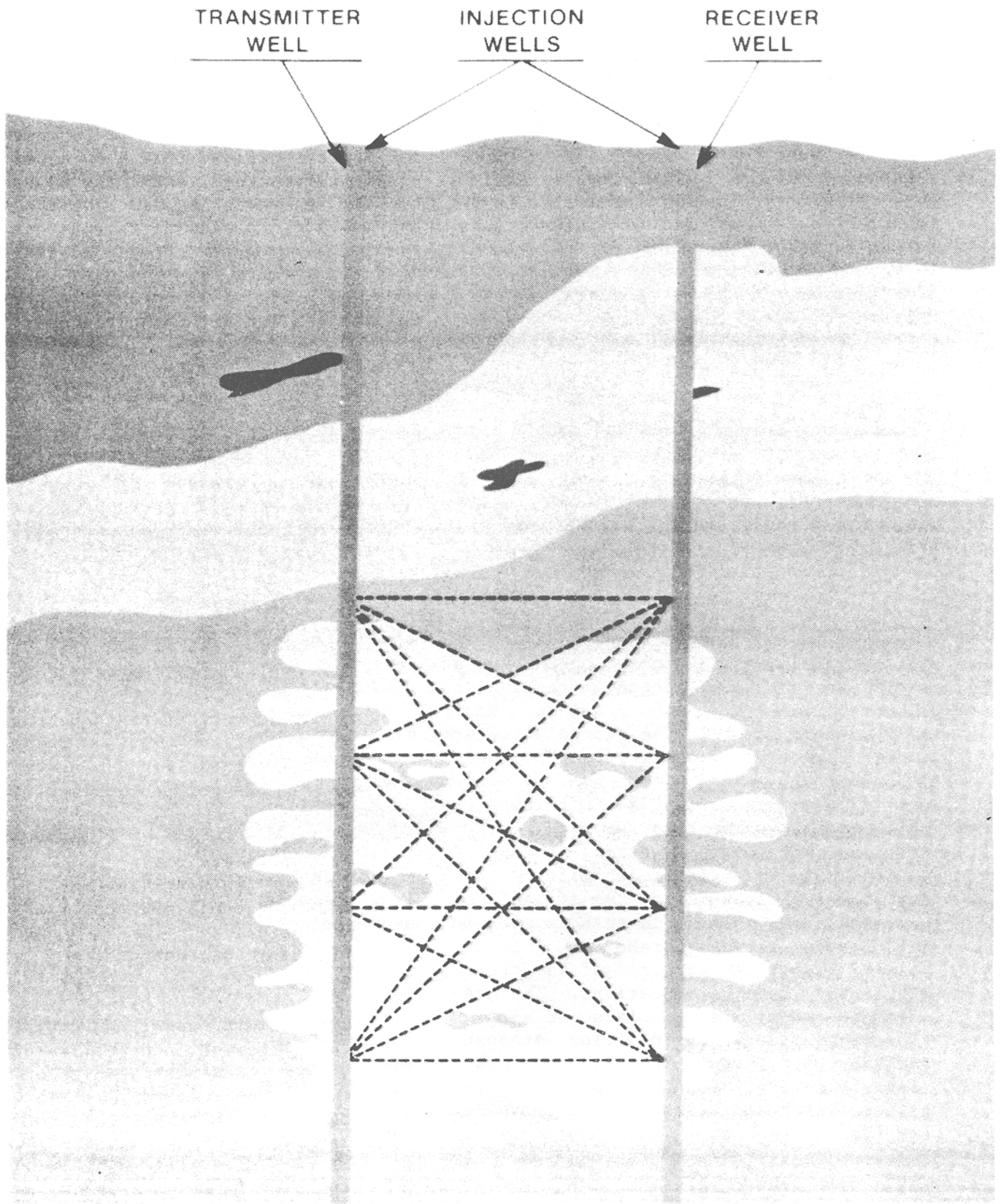


FIGURE 1.—Transmitter and receiver locations in boreholes.

imperfect. Such overdetermination with imperfect data is common in tomography, so this method is generally not applicable. An additional disadvantage is the very large size of the array of the coefficients of the unknowns. These disadvantages make it unsuitable for the desired application.

Neither Fourier transform nor convolutional methods are well suited for reconstructions with typical geophysical tomographic data. Both methods lack adequate ability to use irregularly spaced data. Those wishing to read more about these methods are referred to reference 11.

Algebraic reconstruction techniques (ART) start with the same set of pixels and linear equations as used in matrix inversion techniques. However, the solution is found by iterations rather than by direct matrix inversion. An initial guess is made for the velocity in each pixel. The corresponding travel times for the ray paths are then calculated and compared with the experimental data. The differences between the calculated and experimental travel times are used in calculating correction factors to be applied to the initial guesses. This procedure is repeated until some convergence or limit criterion is reached.

The correction factors can be calculated for each ray path separately or can be made for all paths simultaneously. In the latter case, the procedure is called the simultaneous iterative reconstruction technique (SIRT). Some authors treat SIRT as a special case of ART; others categorize them as two separate series expansion techniques. Series expansion refers to the solution being expressed in a series in terms of the velocities in the pixels. In this report, SIRT and ART are distinct.

REASONS FOR SELECTING SIRT

Criteria used by Bureau researchers for choosing a tomographic method included the following:

1. It should be a known, proven method.
2. It should converge to a solution with a minimum of special input, such as specifying in advance the optimum number of iterations. It should not diverge after having converged close to a solution.
3. The solution should not depend on the order in which the data are input.
4. The method should make full use of available information. For example, it should not disregard the lengths of the path segments within pixels.
5. It should be applicable to irregularly spaced data.
6. It should be feasible to use with a personal computer having 640 kb random access memory.
7. It should be applicable to inconsistent data, for which there is no exact solution.
8. It should be applicable to overdetermined solutions, that is, more data than unknowns.

The one method that satisfied all of these criteria was SIRT. As explained previously, the two common iterative techniques are ART (algebraic reconstruction technique) and SIRT (simultaneous iterative reconstruction technique). ART converges somewhat faster and requires less memory, but SIRT offers the important advantage that the calculated answer does not depend on the order in which the experimental data are input.

Ivansson (12) discussed the difference in the convergence properties of ART and SIRT. ART converges to a solution satisfying the experimental point that is input last. If the equations are somewhat inconsistent, the usual case with experimental data, then the solution to which ART converges depends on which datum point is input last. This situation can adversely affect the results if the last datum point has a large error. In contrast, SIRT treats all data equally. All data points have equal influence on the solution, unless other weighting is specified.

APPLICATIONS TO MONITORING LEACHATE

A vital question when applying tomography is whether the solution is unique. Is the reconstruction the only solution or just one of many that would give equally good fits to the data? Circumventing the problem of nonuniqueness by selecting the solution giving the smallest pixel-to-pixel variability in seismic velocity is not satisfactory for in situ mining applications. The change in velocity caused by leachate could occur over a small distance. The problem must be approached in terms of the data plus appropriate constraints determining the solution.

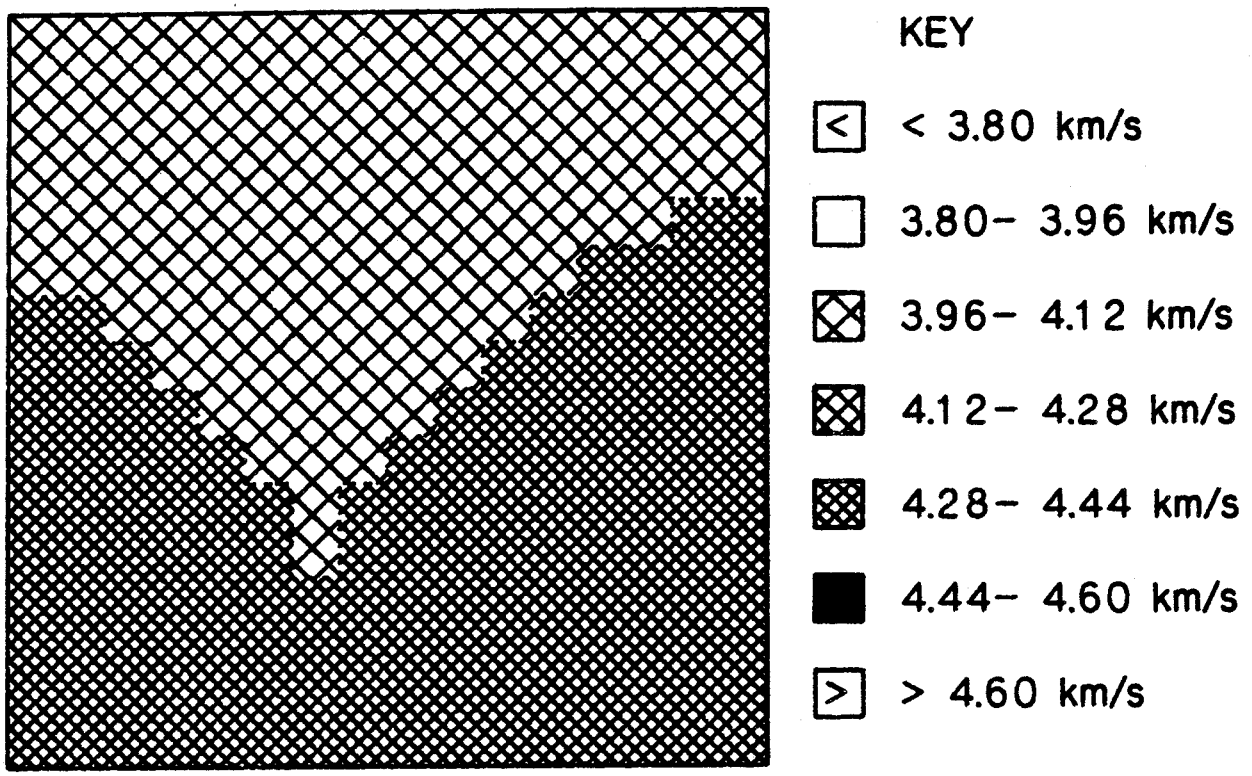
Clearly, if there are more unknowns (pixel velocities) than equations (ray paths with measured travel times), then there are an infinite number of solutions that can fit the data. However, even when the number of pixels is less than the number of ray paths, the solution is not necessarily unique. One test of uniqueness is to vary the initial guesses for the velocities in the pixels. If the solution is unique, it will be independent of the initial guesses. The distribution of velocities among the pixels must be changed in this test; changing all initial guesses by the same factor is not adequate. An alternative test is to assign velocities to pixels to establish a synthetic model, calculate the corresponding travel times, and use those data as input to BOMTOM. If a different solution than the model is obtained, the solution is not unique. The converse is not true, however. Obtaining the same solution as the model does not by itself guarantee uniqueness.

Calculations were performed with BOMTOM to examine the solutions obtained with synthetic travel time data representing an in situ leaching operation above the water table. Synthetic data are more effective than real data for examining the effect of nonuniqueness because an exact solution is known to exist. Departures from the known solution can be attributed to the reconstruction procedure rather than an uncertain amount of experimental error.

The transmitter and receiver were assumed to be in injection wells, as in figure 1. The leachate would move down as it moves out, so the leachate surface would be lower between the wells. The calculations used a 10-pct increase in velocity to represent the effect of saturation. The 10-pct increase is based on experiments by Thill (14). BOMTOM uses straight rays because a 10-pct variation does not require ray bending (3). Geological discontinuities in the types of ore bodies being considered are not expected to be large enough between wells to require ray bending. If geological conditions do require it, programs that perform ray bending are available (13, 15).

The calculations were made assuming 16 transmitter locations and 16 receiver locations spaced symmetrically in the left and right boreholes, respectively. There were 16 rows of pixels, spaced so that transmitter and receiver locations were in the middle of rows. Calculations were first made for 16 columns of pixels. This number of locations and pixels was chosen because it was adequate and convenient for displaying the results. Real data could have fewer or considerably more locations and pixels. Figure 2 displays the model distribution of velocities used in generating the synthetic travel time data.

Figures include graphical displays for a quick qualitative illustration of the distributions and numerical displays for examining the reconstructions in more detail. For uniformity, the graphical display keys in this series of calculations are identical, even when some velocities do not appear in a particular display. Graphical displays are based on output directly from the computer, without rounding to the tabular accuracy. This difference can create some apparent discrepancies. For example, 3.958 and 3.962 would be plotted differently, but would both be 3.96 in a numerical display.




```
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.40 4.40
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.40 4.40 4.40 4.40
4.40 4.40 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.00 4.00 4.00 4.00 4.00 4.40 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.40 4.00 4.00 4.00 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.40 4.40 4.00 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.40 4.40 4.00 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40
```

FIGURE 2.—Model velocity distribution for a 16- by 16-pixel grid. Top, graphical display; bottom; numerical display.

An artifact appeared that would hinder reliable location of the leachate. With 16 columns, the number of unknowns equals the number of data points. There could be a unique solution if all the equations were independent, but there was not. As shown in figure 3, an artifact of a high velocity occurred near the dip in the leachate between the wells. One might expect that reducing the number of unknowns by reducing the number of pixels would provide a better solution, so calculations were performed with 11 columns. The model velocity distribution is shown in figure 4. With 11 columns, there are more data points than pixels. If all the equations were independent, the solution would be unique, in fact, overdetermined. However, figure 5 shows that the artifact was as serious with 11 columns as with 16. There was a definite problem in obtaining reconstructions that allowed reliable monitoring of the leachate, with a special difficulty in reproducing the depth of the dip between the wells.

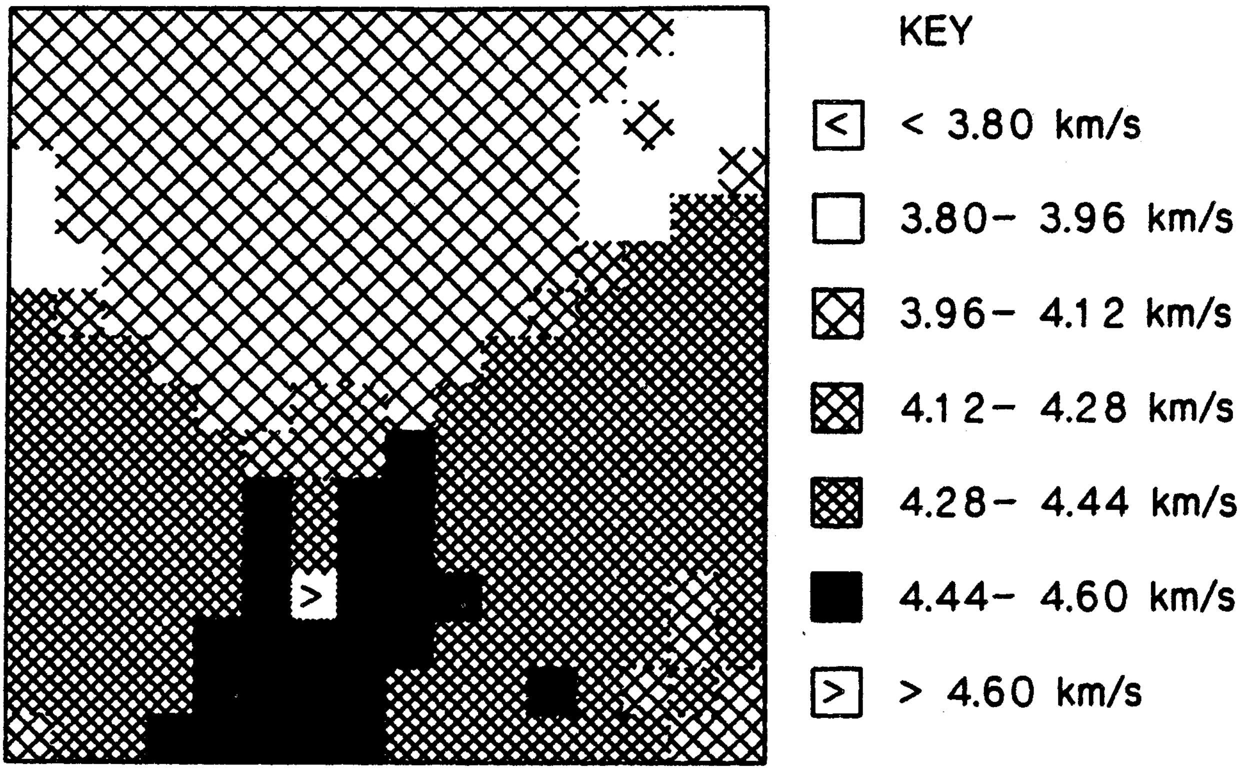
CONSTRAINTS TO REDUCE EFFECT OF NONUNIQUENESS

Calculations with synthetic data showed that solutions were not unique even when the number of pixels was half that of the travel times. Therefore, it was desirable to apply constraints to try to reduce the uncertainty in the solution. One source of uncertainty was that the position and width of a vertical strip passing completely through the region of ray paths cannot be determined by crosshole surveying (12). Figure 6 demonstrates three different distributions of velocity that yield identical travel times. If the initial guesses for velocity are uniform, as in the top distribution, then BOMTOM calculates a solution that is uniform. However, if the initial velocity guesses are close to the middle or bottom distributions, then BOMTOM reconstructs a distribution similar to the corresponding initial guesses. The solution depends on the initial guesses, not only on the data.

The situation is similar to that in four-terminal resistivity measurements, where the data do not contain enough information to allow both the thickness and the resistivity of layers to be determined independently. In tomography, the velocity and the width of a vertical strip passing through the region of ray paths cannot both be determined, nor can its horizontal position. Thus, a solution obtained from crosshole data with no constraints is only one of an infinite number that would fit the data equally well, and depends on the initial velocity guesses.

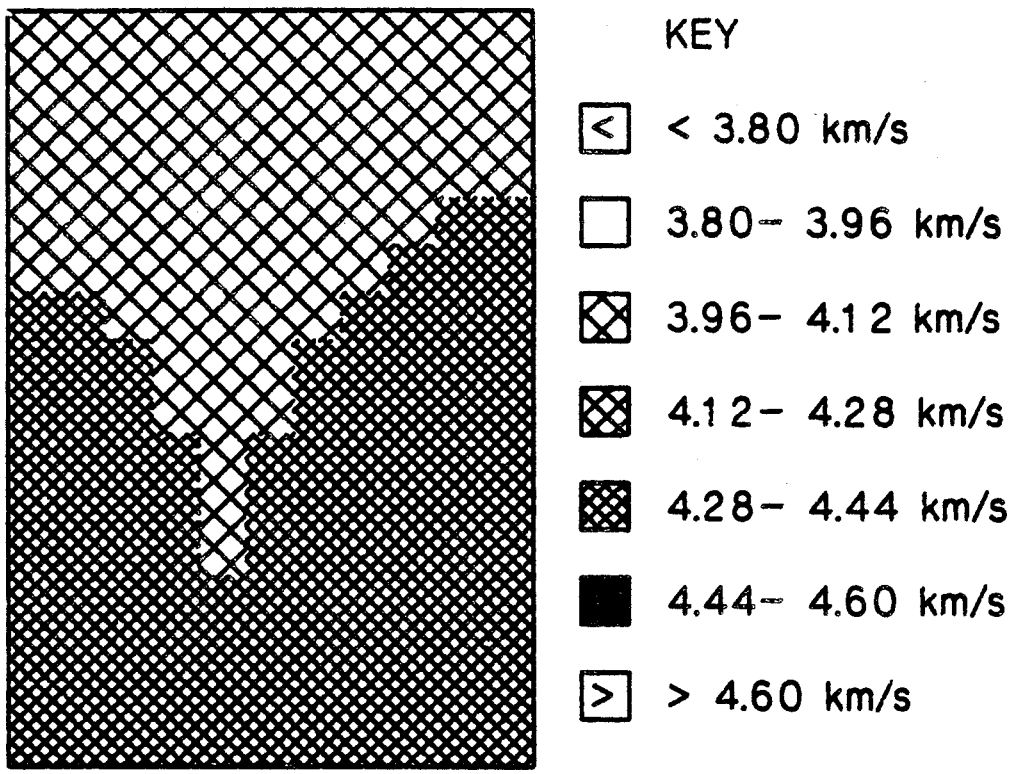
The effect of this source of nonuniqueness on the reliability of reconstructions is not limited to cases where there is a contrasting strip in the actual velocity distribution. It does not arise from inadequate approximations such as too few data points or too few iterations. It is much more general, affecting all unconstrained reconstructions of crosshole data. The effect on reliability occurs because the nonuniqueness of the solution allows artifacts to be produced during the iterative solution procedure. A solution with artifacts can fit the experimental travel times as well as the actual velocity distribution does, no matter how accurate are the data, how many data points there are, or how many iterations are performed.

The effectiveness of several constraints in combatting this source of nonuniqueness was examined. One constraint consisted of requiring that there were rows of pixels at the top and/or bottom of the grid where the velocity did not vary laterally. This constraint was expected to prevent artifacts of vertical strips of contrasting velocity passing through the ray path region. It will be satisfied in many applications in in situ mining above the water table because the top of the pixel grid will represent a region with no leachate and the bottom will represent a saturated region. An option was added to BOMTOM for designating the number of rows at the top and bottom in which the velocity for all pixels in the row is set equal to the average velocity for that row. The option does not specify the average velocity nor that the average velocity for any row is related to the velocity in any other row. A second constraint consisted



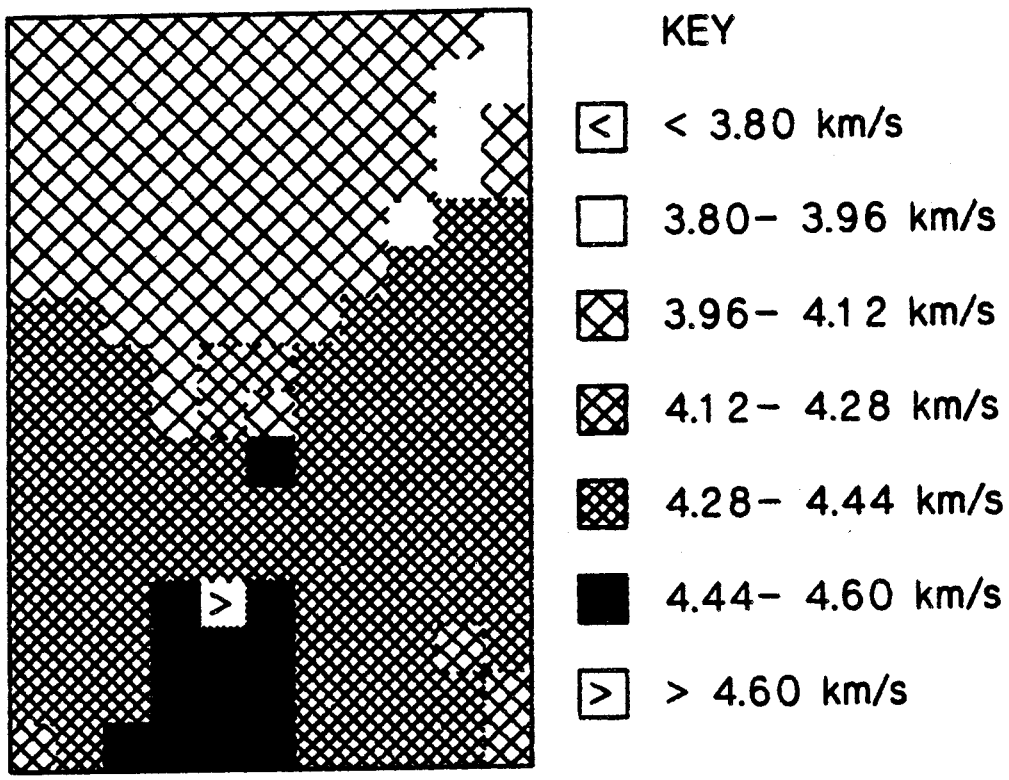
4.07	4.02	4.02	4.03	4.02	4.01	4.01	4.01	4.01	4.01	4.01	4.00	3.98	3.98	3.92	3.87
4.01	4.04	4.03	4.02	4.01	4.02	4.02	4.02	4.01	4.01	4.00	4.02	3.98	3.93	3.94	3.88
4.00	4.02	4.02	4.03	4.04	4.01	4.04	4.02	4.03	4.02	4.01	3.98	3.95	3.96	3.91	3.93
3.95	4.00	4.02	4.04	4.04	4.05	4.04	4.05	4.04	4.01	4.02	3.96	3.94	3.95	3.88	4.01
3.95	3.96	4.02	4.04	4.06	4.07	4.05	4.05	4.03	4.02	3.99	3.98	3.91	3.95	4.30	4.37
3.92	3.95	4.00	4.06	4.07	4.07	4.07	4.05	4.03	4.04	3.99	3.96	4.25	4.33	4.32	4.35
4.40	4.26	4.02	4.03	4.10	4.08	4.10	4.07	4.05	4.02	4.00	4.27	4.30	4.32	4.33	4.35
4.34	4.33	4.34	4.06	4.08	4.10	4.08	4.11	4.07	4.03	4.34	4.31	4.28	4.32	4.33	4.31
4.37	4.31	4.37	4.35	4.10	4.11	4.14	4.14	4.10	4.35	4.32	4.30	4.33	4.32	4.32	4.34
4.36	4.35	4.35	4.38	4.38	4.14	4.21	4.15	4.48	4.38	4.32	4.34	4.30	4.32	4.29	4.33
4.38	4.35	4.34	4.35	4.37	4.48	4.39	4.52	4.44	4.38	4.34	4.33	4.30	4.32	4.30	4.33
4.35	4.39	4.32	4.38	4.37	4.46	4.40	4.48	4.48	4.41	4.37	4.31	4.29	4.31	4.30	4.30
4.39	4.34	4.33	4.36	4.43	4.52	4.65	4.56	4.46	4.47	4.37	4.32	4.32	4.31	4.27	4.28
4.35	4.33	4.36	4.39	4.50	4.48	4.55	4.55	4.46	4.42	4.42	4.36	4.32	4.28	4.25	4.31
4.31	4.29	4.40	4.43	4.49	4.51	4.52	4.52	4.40	4.41	4.43	4.44	4.37	4.25	4.28	4.24
4.19	4.36	4.39	4.47	4.49	4.50	4.50	4.50	4.41	4.41	4.41	4.42	4.42	4.32	4.24	4.14

FIGURE 3.—Reconstruction with a 16- by 16-pixel grid and no constraints. Top, graphical display; bottom, numerical display.




```
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.40 4.40
4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.00 4.40 4.40 4.40
4.40 4.40 4.00 4.00 4.00 4.00 4.00 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.00 4.00 4.00 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.00 4.00 4.00 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.00 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.00 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.00 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40
4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40 4.40
```

FIGURE 4.—Model velocity distribution for an 11- by 16-pixel grid. Top, graphical display; bottom, numerical display.



4.04	4.02	4.03	4.03	4.03	4.00	3.98	3.98	4.00	3.98	3.88
4.01	4.03	4.02	4.04	4.03	4.01	3.99	3.98	3.98	3.94	3.93
3.99	4.02	4.02	4.04	4.05	4.03	4.00	3.98	3.97	3.92	3.98
3.98	4.01	4.02	4.06	4.05	4.04	4.01	3.97	3.96	3.90	4.00
3.98	3.99	4.02	4.07	4.07	4.04	4.02	3.96	3.94	4.29	4.39
3.99	3.96	4.02	4.09	4.09	4.05	4.00	3.96	4.32	4.31	4.36
4.40	4.33	4.01	4.09	4.09	4.10	3.97	4.35	4.31	4.32	4.34
4.40	4.33	4.39	4.09	4.12	4.13	4.33	4.34	4.31	4.32	4.33
4.39	4.35	4.36	4.07	4.18	4.11	4.33	4.34	4.32	4.32	4.33
4.38	4.36	4.35	4.44	4.28	4.47	4.36	4.34	4.30	4.31	4.34
4.37	4.36	4.36	4.40	4.34	4.42	4.41	4.33	4.29	4.32	4.33
4.37	4.35	4.36	4.40	4.32	4.43	4.42	4.33	4.29	4.32	4.31
4.37	4.35	4.36	4.45	4.69	4.47	4.41	4.34	4.32	4.29	4.30
4.34	4.35	4.39	4.49	4.59	4.51	4.38	4.36	4.35	4.27	4.29
4.30	4.35	4.42	4.52	4.57	4.46	4.35	4.42	4.37	4.28	4.25
4.25	4.35	4.46	4.54	4.54	4.46	4.38	4.38	4.39	4.31	4.16

FIGURE 5.—Reconstruction with an 11- by 16-pixel grid and no constraints. Top, graphical display; bottom, numerical display.

of setting the velocities near the boreholes to known values and not allowing them to change. This constraint could be applied to in situ mining if sonic logs were run in the boreholes.

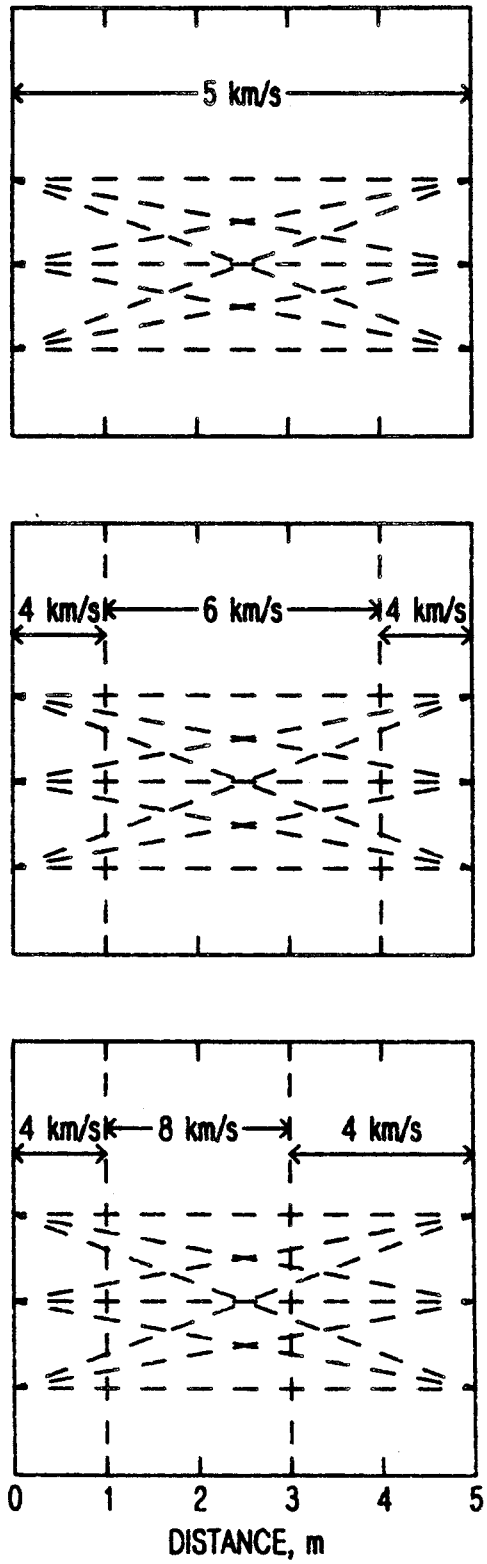


FIGURE 6.—Three velocity distributions yielding identical travel times.

A pixel grid of 10 rows and 5 columns was used in this part of the study. The input travel times were those for a uniform 5-km/s velocity distribution with 10 transmitters and 10 receivers evenly spaced. Thus, there were twice as many ray paths as pixels. To test the effectiveness of the constraints, the initial velocity guesses included a highly contrasting vertical strip. The model velocity distribution, the initial guesses, and the reconstructions with the constraints are shown in figure 7. An ideal reconstruction would give the uniform 5-km/s model velocity distribution in *7A*. As expected, the reconstruction with no constraints (*7C*) was poor. The initial vertical strip was not only still present, it actually increased as the program found an alternative velocity distribution yielding the same travel times. Reconstructions with one row at the top (*7D*) or with one row at both top and bottom (*7E*) where the velocity did not vary laterally were not satisfactory. Fixing the velocity in the boreholes (*7F*) was not beneficial when used by itself. The combination of fixed borehole velocities and one row at both top and botttom where the velocity did not vary laterally (*7G*) was slightly better than no constraints, but not satisfactory. Two laterally constant rows at the top and at the bottom (*7H*) were more effective than the combination in *7G*. Obtaining a good reconstruction required the combination of two laterally constant rows at top and bottom with fixed velocities in the boreholes (*7I*). These results demonstrated the desirability of surrounding the region to be investigated by as large an area of known and constrained velocities as possible. They also demonstrated the effectiveness of the combination of laterally constant rows and fixed borehole velocities as constraints to combat nonuniqueness of solutions.

With those results giving some hope of dealing with the artifact described previously, the effectiveness of the constraints on the synthetic in situ data was examined. The two cases considered were those corresponding to figure *7H* and

A

5.00	5.00	5.00	5.00	5.00
5.00	5.00	5.00	5.00	5.00
5.00	5.00	5.00	5.00	5.00
5.00	5.00	5.00	5.00	5.00
5.00	5.00	5.00	5.00	5.00
5.00	5.00	5.00	5.00	5.00
5.00	5.00	5.00	5.00	5.00
5.00	5.00	5.00	5.00	5.00
5.00	5.00	5.00	5.00	5.00
5.00	5.00	5.00	5.00	5.00

B

5.00	4.00	8.00	4.00	5.00
5.00	4.00	8.00	4.00	5.00
5.00	4.00	8.00	4.00	5.00
5.00	4.00	8.00	4.00	5.00
5.00	4.00	8.00	4.00	5.00
5.00	4.00	8.00	4.00	5.00
5.00	4.00	8.00	4.00	5.00
5.00	4.00	8.00	4.00	5.00
5.00	4.00	8.00	4.00	5.00
5.00	4.00	8.00	4.00	5.00

C

5.13	4.08	8.33	4.08	5.13
5.13	4.08	8.33	4.08	5.13
5.13	4.08	8.33	4.08	5.13
5.13	4.08	8.33	4.08	5.13
5.13	4.08	8.33	4.08	5.13
5.13	4.08	8.33	4.08	5.13
5.13	4.08	8.33	4.08	5.13
5.13	4.08	8.33	4.08	5.13
5.13	4.08	8.33	4.08	5.13
5.13	4.08	8.33	4.08	5.13

D

5.00	5.00	5.00	5.00	5.00
5.11	4.17	7.67	4.17	5.11
5.09	4.11	8.27	4.11	5.09
5.13	4.08	8.40	4.08	5.13
5.12	4.07	8.43	4.07	5.12
5.12	4.08	8.38	4.08	5.12
5.13	4.08	8.32	4.08	5.13
5.13	4.08	8.30	4.08	5.13
5.14	4.08	8.30	4.08	5.14
5.15	4.07	8.29	4.07	5.15

E

5.00	5.00	5.00	5.00	5.00
5.10	4.19	7.63	4.19	5.10
5.08	4.12	8.25	4.12	5.08
5.11	4.09	8.43	4.09	5.11
5.10	4.07	8.54	4.07	5.10
5.10	4.07	8.54	4.07	5.10
5.11	4.09	8.43	4.09	5.11
5.08	4.12	8.25	4.12	5.08
5.10	4.19	7.63	4.19	5.10
5.00	5.00	5.00	5.00	5.00

F

5.00	4.14	8.57	4.14	5.00
5.00	4.14	8.57	4.14	5.00
5.00	4.14	8.57	4.14	5.00
5.00	4.14	8.57	4.14	5.00
5.00	4.14	8.57	4.14	5.00
5.00	4.14	8.57	4.14	5.00
5.00	4.14	8.57	4.14	5.00
5.00	4.14	8.57	4.14	5.00
5.00	4.14	8.57	4.14	5.00
5.00	4.14	8.57	4.14	5.00

G

5.00	5.00	5.00	5.00	5.00
5.00	4.28	7.36	4.28	5.00
5.00	4.23	7.88	4.23	5.00
5.00	4.21	7.99	4.21	5.00
5.00	4.20	8.04	4.20	5.00
5.00	4.20	8.04	4.20	5.00
5.00	4.21	7.99	4.21	5.00
5.00	4.23	7.88	4.23	5.00
5.00	4.28	7.36	4.28	5.00
5.00	5.00	5.00	5.00	5.00

H

5.00	5.00	5.00	5.00	5.00
4.99	4.99	4.99	4.99	4.99
5.16	4.58	5.70	4.58	5.16
5.25	4.30	6.49	4.30	5.25
5.32	4.16	7.00	4.16	5.32
5.32	4.16	7.00	4.16	5.32
5.25	4.30	6.49	4.30	5.25
5.16	4.58	5.70	4.58	5.16
4.99	4.99	4.99	4.99	4.99
5.00	5.00	5.00	5.00	5.00

I

5.00	5.00	5.00	5.00	5.00
5.00	5.00	5.00	5.00	5.00
5.00	4.91	5.18	4.91	5.00
5.00	4.87	5.26	4.87	5.00
5.00	4.86	5.28	4.86	5.00
5.00	4.86	5.28	4.86	5.00
5.00	4.87	5.26	4.87	5.00
5.00	4.91	5.18	4.91	5.00
5.00	5.00	5.00	5.00	5.00
5.00	5.00	5.00	5.00	5.00

FIGURE 7.—Effectiveness of constraints combatting nonuniqueness from contrasting vertical strip in initial velocity guesses. An ideal reconstruction would reproduce the uniform 5-kms velocity distribution in *A*. *A*, model velocity distribution used for generating synthetic travel times; *B*, velocity distribution used for initial guess; reconstruction with—*C*, no constraints; *D*, one averaged row at top; *E*, one averaged row at top and bottom; *F*, known fixed velocities in boreholes; *G*, one averaged row at top and bottom, and fixed velocities in boreholes; *H*, two averaged rows at top and bottom; *I*, two averaged rows at top and bottom, and fixed velocities in boreholes.

7I. All tests were done with 11 columns and 16 rows of pixels, using synthetic travel times generated by the velocity distribution in figure 4. Enough iterations were performed to ensure that the reconstruction would not benefit from additional iterations.

The reconstruction with two laterally constant rows at top and at bottom is shown in figure 8. The initial guesses were a uniform 4.2 km/s, since reconstructions with in situ mining data are likely to be started with a uniform velocity guess. The artifact of high velocity near the dip was smaller than when no constraints were used.

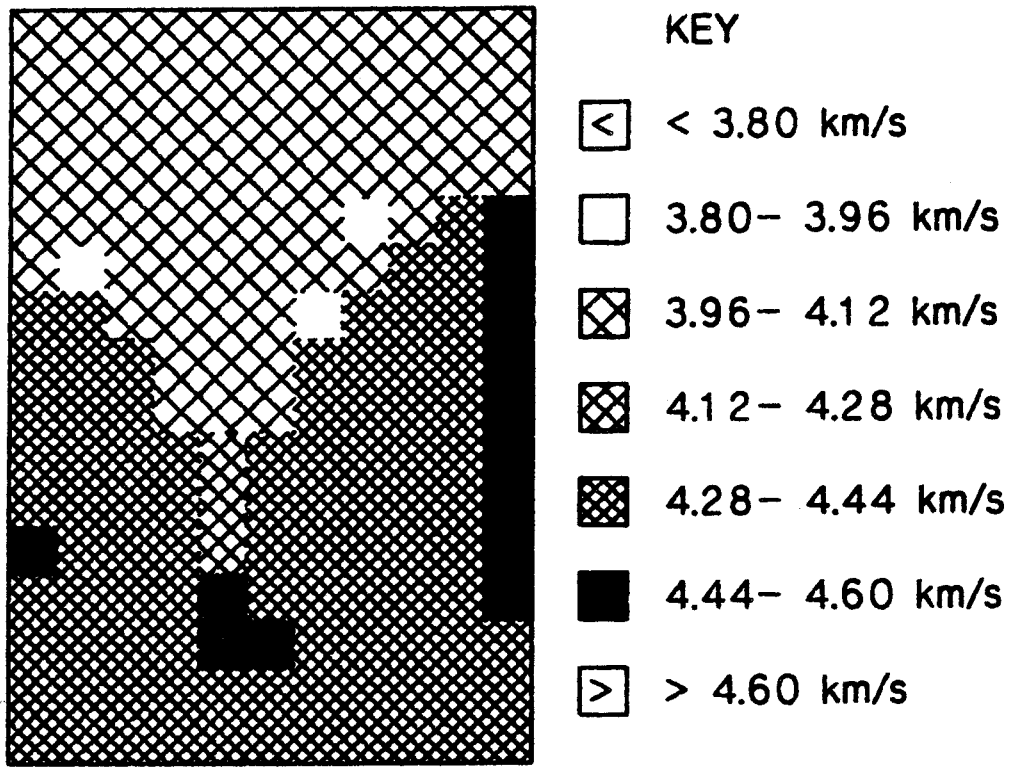
The reconstruction with the combination of the preceding constraint plus fixed borehole velocities is shown in figure 9. In this case, the reconstruction benefited only slightly by the additional constraint. These results indicate that the benefits of constraints should be examined on a case by case basis. The fixing of borehole velocities that helped for the initial velocity with contrasting vertical strip shown in figure 7 was less helpful for the uniform initial velocity used here.

One additional constraint was tried. It consisted of limiting the calculated velocity to a specified range. A very good reconstruction, shown in figure 10, was obtained with no constraint other than limiting the maximum calculated velocity to 4.4 km/s. The reconstruction with 16 columns was nearly as good. Additional calculations were performed with 11 columns and adding the previously described constraints. Fixing the velocity in the boreholes to known values improved the reconstruction only slightly. When all the constraints were applied, fixing the velocity in the boreholes, having two rows at top and bottom with no lateral variation in velocity, plus limiting the maximum velocity to 4.4 km/s, the reconstruction was nearly identical to the model velocity distribution. However, this constraint was not helpful unless the upper limit was set quite close to the maximum velocity in the model. Limiting the velocity range to 3.8 to 4.6 km/s did not provide additional improvement when used with fixed borehole velocities and laterally invariant top and bottom rows. In practice, one may not know the limits precisely enough to help the reconstruction significantly.

As stated previously, a unique solution is independent of the initial velocity guesses. To examine the uniqueness further, reconstructions were calculated using a first guess known to be very different from the model velocity distribution. The reconstruction used the travel times for the 11- by 16-pixel velocity model shown previously in figure 4. However, the initial velocity guess was 6.0 km/s in the top half of the grid and 3.0 km/s in the bottom half. This pattern reversed the order of the model, where the higher velocities were in the lower portion of the grid, and also included a wider range of velocities. Reconstructions were performed with no constraints, constraints at the top and bottom, and constraints at top, bottom and sides.

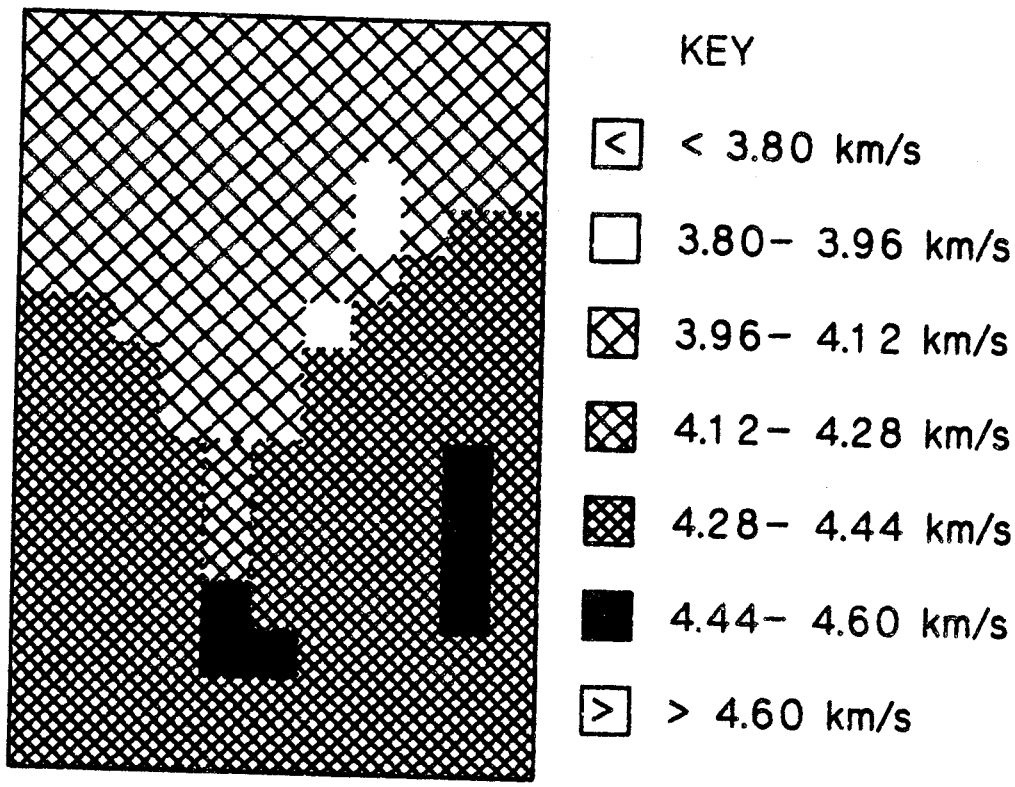
The reconstruction with no constraints is shown in figure 11. As expected, BOMTOM was able to fit the data with a reconstruction that resembled neither the model velocity distribution nor the reconstruction obtained with a uniform 4.2-km/s initial guess. This reconstruction would not allow reliable determination of the location of the leachate.

Again, constraints were helpful. The reconstruction with two laterally invariant rows at the top and bottom is shown in figure 12. The reconstruction is closer to the model, and would allow a useful, if not exact, determination of leachate location. The additional constraint of fixing the velocities in the boreholes to known values improved the reconstruction slightly more, as shown in figure 13. The reconstruction is sufficiently close to the model to provide a useful indication of the location of the leachate, including the depth of the dip. It is quite close to that obtained with those constraints and a uniform 4.2-km/s initial guess. Thus, this reconstruction is sufficiently constrained so the results are determined primarily by the data, not by the initial velocity guess.



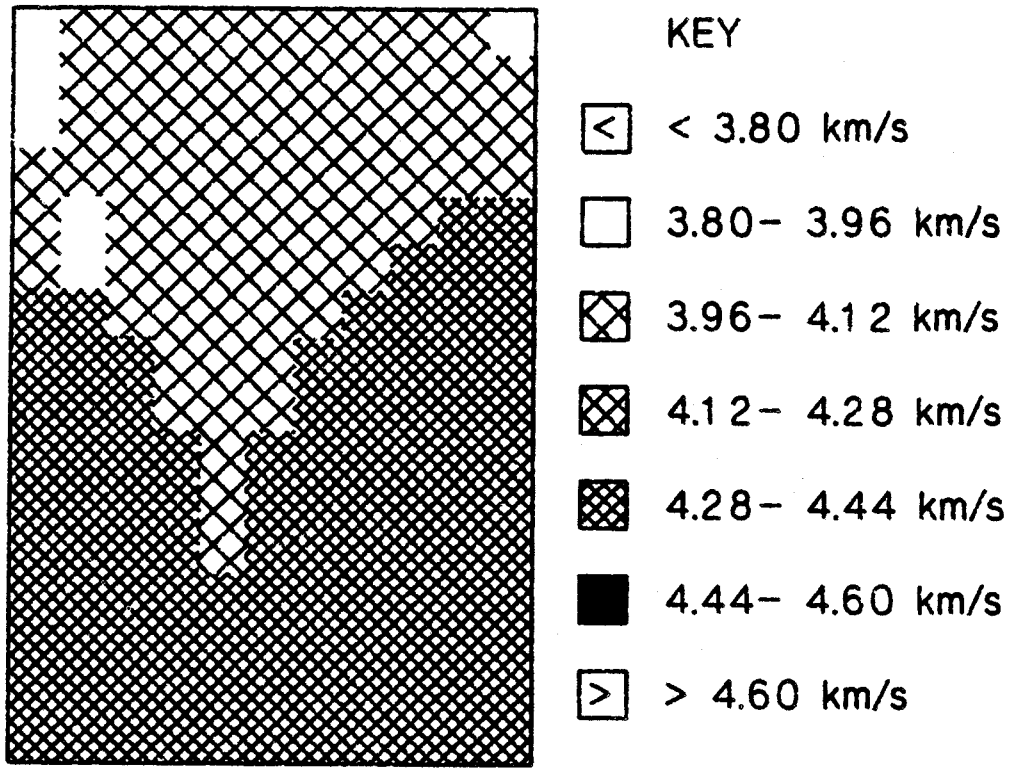
4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
4.00	3.99	4.00	3.99	4.02	4.02	4.00	3.97	3.99	4.01	4.04
4.00	3.98	3.99	4.00	4.03	4.03	4.00	3.96	3.98	4.00	4.07
4.00	3.97	3.98	4.02	4.04	4.03	3.99	3.96	3.97	4.38	4.50
4.01	3.96	3.98	4.04	4.05	4.04	3.97	3.96	4.37	4.39	4.49
4.41	4.34	3.98	4.04	4.05	4.06	3.94	4.37	4.36	4.40	4.48
4.42	4.34	4.37	4.03	4.07	4.06	4.32	4.37	4.36	4.41	4.47
4.43	4.35	4.35	4.01	4.11	4.04	4.33	4.36	4.36	4.41	4.47
4.43	4.36	4.34	4.38	4.17	4.40	4.35	4.35	4.35	4.41	4.48
4.44	4.37	4.34	4.34	4.21	4.37	4.39	4.34	4.34	4.42	4.47
4.44	4.38	4.34	4.32	4.19	4.37	4.40	4.33	4.34	4.43	4.46
4.44	4.39	4.35	4.32	4.57	4.41	4.40	4.33	4.36	4.43	4.44
4.42	4.40	4.37	4.35	4.47	4.45	4.38	4.36	4.38	4.43	4.42
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40

FIGURE 8.—Reconstruction with an 11- by 16-pixel grid, constraints at top and bottom. Top, graphical display; bottom, numerical display.



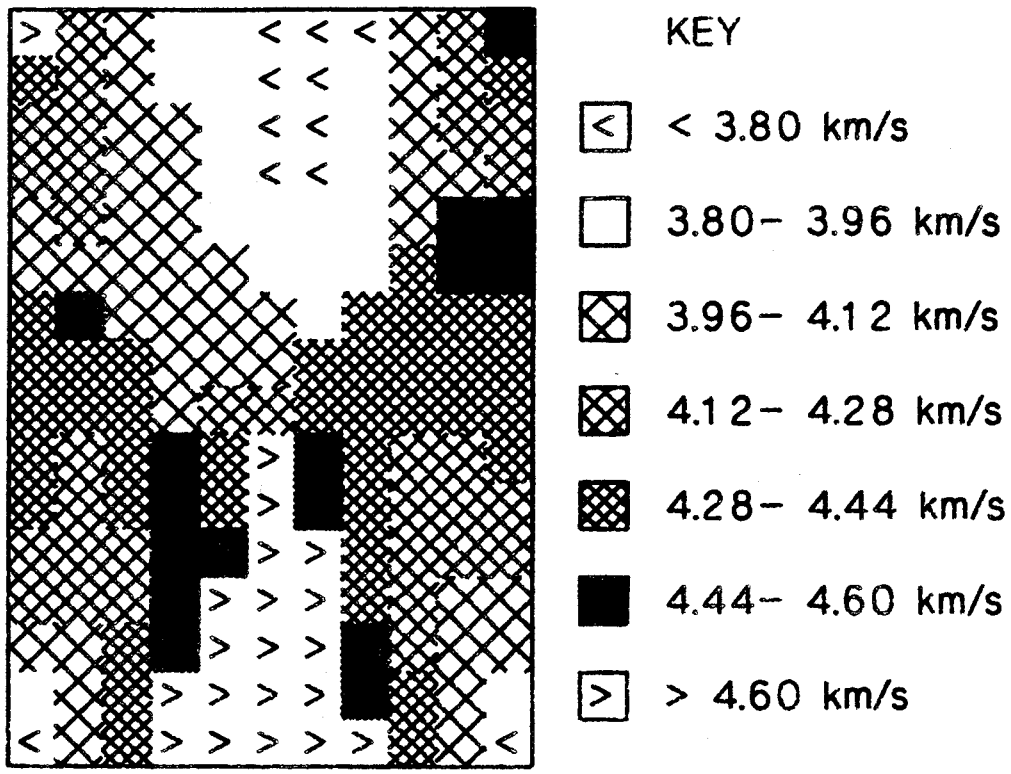
4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
4.00	3.99	3.99	3.99	4.02	4.02	4.00	3.96	3.99	4.02	4.00
4.00	3.98	3.99	4.00	4.03	4.03	4.00	3.96	3.98	4.02	4.00
4.00	3.97	3.99	4.02	4.04	4.03	3.99	3.96	3.98	4.43	4.40
4.00	3.96	3.99	4.03	4.04	4.04	3.97	3.97	4.37	4.43	4.40
4.40	4.35	3.98	4.04	4.05	4.05	3.94	4.37	4.38	4.43	4.40
4.40	4.35	4.37	4.03	4.07	4.05	4.31	4.37	4.38	4.43	4.40
4.40	4.36	4.36	4.01	4.11	4.03	4.32	4.37	4.37	4.43	4.40
4.40	4.38	4.35	4.37	4.16	4.39	4.35	4.36	4.36	4.44	4.40
4.40	4.39	4.34	4.34	4.20	4.36	4.39	4.34	4.36	4.45	4.40
4.40	4.41	4.35	4.32	4.19	4.37	4.40	4.33	4.36	4.45	4.40
4.40	4.41	4.36	4.32	4.56	4.42	4.40	4.33	4.37	4.45	4.40
4.40	4.41	4.37	4.35	4.47	4.45	4.38	4.36	4.39	4.43	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40

FIGURE 9.—Reconstruction with an 11- by 16-pixel grid, constraints at top, bottom, and sides. Top, graphical display; bottom, numerical display.



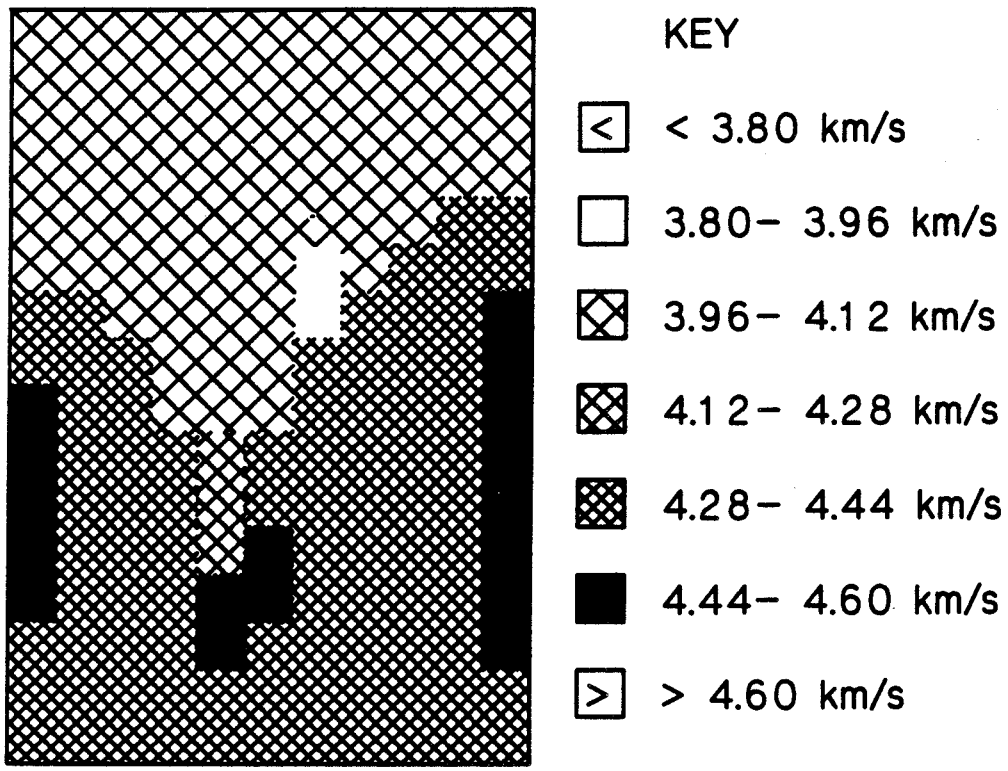
3.92	3.96	4.00	4.05	4.05	4.02	4.00	4.00	4.00	3.98	3.96
3.94	3.96	3.99	4.04	4.06	4.02	4.00	4.00	4.00	3.97	3.98
3.95	3.96	3.98	4.04	4.06	4.03	4.00	3.99	3.99	3.97	4.00
3.97	3.96	3.98	4.03	4.06	4.03	4.00	3.99	3.99	3.97	4.00
3.98	3.96	3.98	4.03	4.06	4.05	3.99	3.99	3.99	4.37	4.40
3.99	3.96	3.98	4.02	4.05	4.06	3.98	3.99	4.38	4.38	4.40
4.40	4.35	3.98	4.01	4.05	4.06	3.97	4.40	4.38	4.39	4.40
4.40	4.36	4.37	3.99	4.06	4.05	4.37	4.40	4.38	4.39	4.39
4.40	4.38	4.37	3.98	4.08	4.02	4.38	4.39	4.39	4.40	4.39
4.40	4.39	4.38	4.37	4.09	4.40	4.39	4.39	4.39	4.40	4.39
4.39	4.40	4.38	4.37	4.07	4.39	4.40	4.40	4.39	4.40	4.39
4.39	4.40	4.39	4.38	4.04	4.40	4.40	4.40	4.40	4.40	4.39
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40

FIGURE 10.—Reconstruction with an 11- by 16-pixel grid, constrained maximum velocity. Top, graphical display; bottom, numerical display.



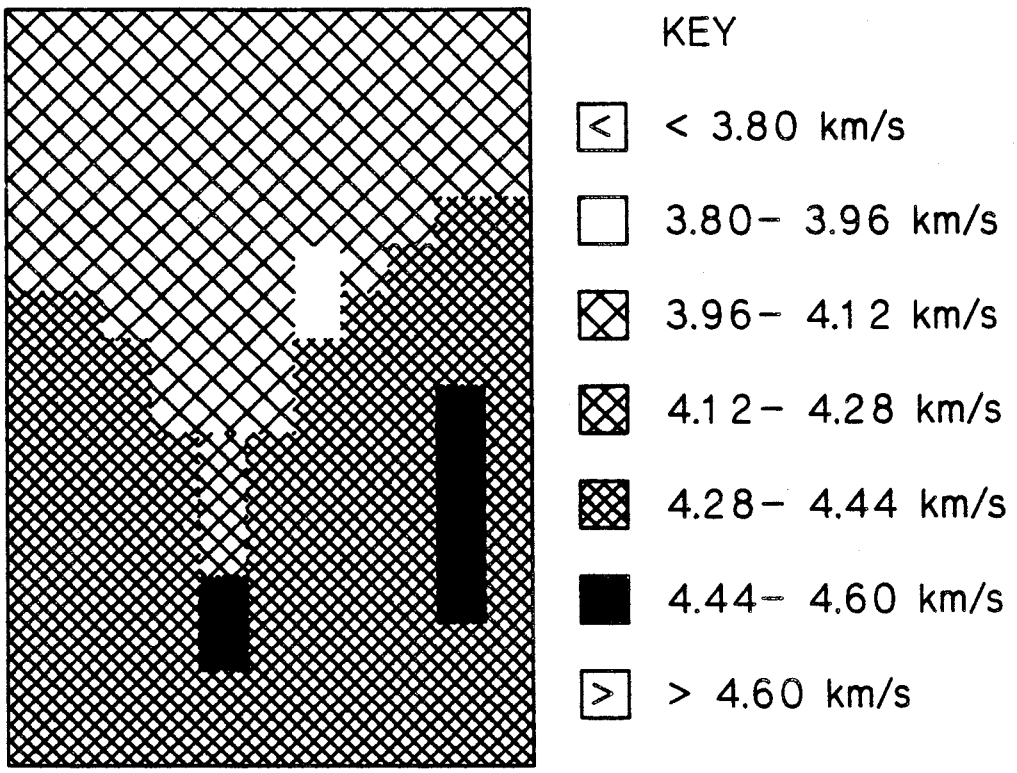
4.62	4.25	4.10	3.85	3.85	3.80	3.75	3.75	4.04	4.21	4.50
4.42	4.26	4.09	3.93	3.84	3.78	3.75	3.83	4.04	4.19	4.36
4.28	4.23	4.10	3.98	3.86	3.79	3.77	3.88	4.05	4.15	4.25
4.18	4.20	4.10	4.03	3.89	3.79	3.80	3.92	4.04	4.11	4.17
4.11	4.15	4.08	4.05	3.94	3.82	3.83	3.94	4.03	4.48	4.52
4.07	4.10	4.07	4.07	3.98	3.89	3.86	3.95	4.40	4.44	4.45
4.44	4.45	4.04	4.06	4.04	3.98	3.90	4.35	4.37	4.40	4.39
4.41	4.38	4.41	4.06	4.11	4.07	4.33	4.35	4.33	4.34	4.35
4.39	4.32	4.37	4.05	4.20	4.13	4.40	4.34	4.29	4.28	4.32
4.35	4.26	4.32	4.45	4.30	4.63	4.49	4.34	4.25	4.22	4.29
4.30	4.21	4.29	4.45	4.39	4.69	4.56	4.35	4.23	4.17	4.24
4.23	4.17	4.27	4.46	4.45	4.75	4.62	4.37	4.21	4.13	4.18
4.15	4.14	4.27	4.50	4.99	4.79	4.66	4.41	4.22	4.08	4.10
4.04	4.11	4.29	4.57	4.99	4.81	4.66	4.47	4.24	4.05	3.99
3.90	4.09	4.34	4.67	4.95	4.82	4.63	4.56	4.29	4.04	3.85
3.73	4.09	4.38	4.86	4.86	4.75	4.64	4.64	4.32	4.06	3.67

FIGURE 11.—Reconstruction with an 11- by 16-pixel grid, alternative initial velocity guess, no constraints. Top, graphical display; bottom, numerical display.



4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
3.99	3.98	4.01	4.01	4.02	3.98	3.99	4.01	4.00	4.00	4.01
3.97	3.98	4.01	4.03	4.02	3.97	3.98	4.02	3.99	4.00	4.01
3.97	3.99	4.01	4.04	4.02	3.97	3.97	4.01	3.99	4.40	4.42
3.98	3.98	4.00	4.04	4.02	4.00	3.95	4.00	4.39	4.40	4.43
4.39	4.37	3.99	4.02	4.04	4.03	3.94	4.39	4.38	4.41	4.44
4.42	4.36	4.38	4.00	4.06	4.04	4.34	4.38	4.36	4.41	4.46
4.47	4.35	4.36	3.98	4.10	4.03	4.36	4.36	4.34	4.40	4.49
4.49	4.35	4.34	4.35	4.14	4.42	4.40	4.34	4.33	4.40	4.52
4.50	4.36	4.34	4.31	4.16	4.43	4.42	4.32	4.32	4.41	4.52
4.49	4.38	4.34	4.29	4.15	4.44	4.43	4.31	4.33	4.42	4.51
4.47	4.39	4.36	4.29	4.54	4.45	4.42	4.31	4.34	4.43	4.48
4.44	4.41	4.37	4.33	4.48	4.44	4.40	4.36	4.36	4.43	4.44
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40

FIGURE 12.—Reconstruction with an 11- by 16-pixel grid, alternative initial velocity guess, constraints at top and bottom. Top, graphical display; bottom, numerical display.



4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00	4.00
4.00	3.98	4.00	4.01	4.02	3.99	3.99	4.00	4.00	4.00	4.00
4.00	3.97	4.01	4.02	4.02	3.98	3.98	4.01	3.99	4.01	4.00
4.00	3.97	4.00	4.04	4.03	3.99	3.97	4.01	3.99	4.41	4.40
4.00	3.97	4.00	4.04	4.03	4.00	3.96	4.01	4.39	4.42	4.40
4.40	4.37	3.99	4.03	4.04	4.02	3.94	4.40	4.38	4.42	4.40
4.40	4.38	4.38	4.01	4.06	4.03	4.33	4.39	4.37	4.43	4.40
4.40	4.39	4.37	3.98	4.09	4.03	4.35	4.36	4.36	4.45	4.40
4.40	4.41	4.36	4.34	4.13	4.42	4.38	4.34	4.35	4.46	4.40
4.40	4.42	4.36	4.31	4.15	4.41	4.41	4.31	4.35	4.47	4.40
4.40	4.42	4.36	4.29	4.14	4.42	4.42	4.31	4.35	4.47	4.40
4.40	4.42	4.37	4.30	4.53	4.43	4.41	4.32	4.36	4.46	4.40
4.40	4.42	4.38	4.34	4.47	4.43	4.39	4.36	4.38	4.44	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40

FIGURE 13.—Reconstruction with an 11- by 16-pixel grid, alternative initial velocity guess, constraints at top, bottom, and sides. Top, graphical display; bottom, numerical display.

SMOOTHING

Smoothing consists of replacing the velocity in each pixel with a weighted average velocity, averaged over that pixel and its neighbors. Possible benefits of smoothing were examined. Smoothing has been reported to provide a better looking reconstruction with field data (6). The purpose of smoothing is to average random inconsistencies in data, so smoothing would not be beneficial with self-consistent synthetic data. The inconsistencies with field data arise only partially from experimental errors in measuring the travel times and positions. They occur also because the real velocity distribution can vary continuously with position. Real data come from a very large number of very small pixels. Tomography uses fewer, larger pixels, each containing a constant velocity.

To simulate experimental errors, the synthetic travel times were rounded to 0.1 ms by changing the output format for the data file. To provide a finer pixel grid, a grid of 61 columns and 61 rows was used with 16 transmitters and 16 receivers spaced evenly in the boreholes. Figure 14 shows the distribution of velocities used in generating the travel times. The velocity range for this series of calculations was changed slightly to obtain graphical velocity increments that better portrayed the gradation in velocities near boundaries of wet and dry zones using the larger number of pixels.

Figures 15 and 16 show the reconstruction obtained using a 61- by 61-pixel grid without and with smoothing, respectively. All other options and initial velocity guesses were the same for both runs. The smoothing option in BOMTOM used weighting factors of 20, 4, and 1 for the pixel whose velocity was being replaced, its nearest neighbors, and its corner neighbors, respectively. Smoothing was performed after each iteration. With more pixels than ray paths, some of the pixels do not contain any rays. Without smoothing, there is no basis for changing the velocity in them. With smoothing, those pixels have their velocity adjusted by the effect of their neighbors. The reconstruction without smoothing is so uneven that it would be difficult to interpret. With smoothing, patterns are more evident.

Another question is whether it is better to use more pixels with smoothing or fewer pixels without smoothing. Figure 17 shows the reconstruction using a 16- by 16-pixel grid and no smoothing. It would be difficult to interpret such an uneven distribution. In this example, smoothing provided a reconstruction that was easier to interpret.

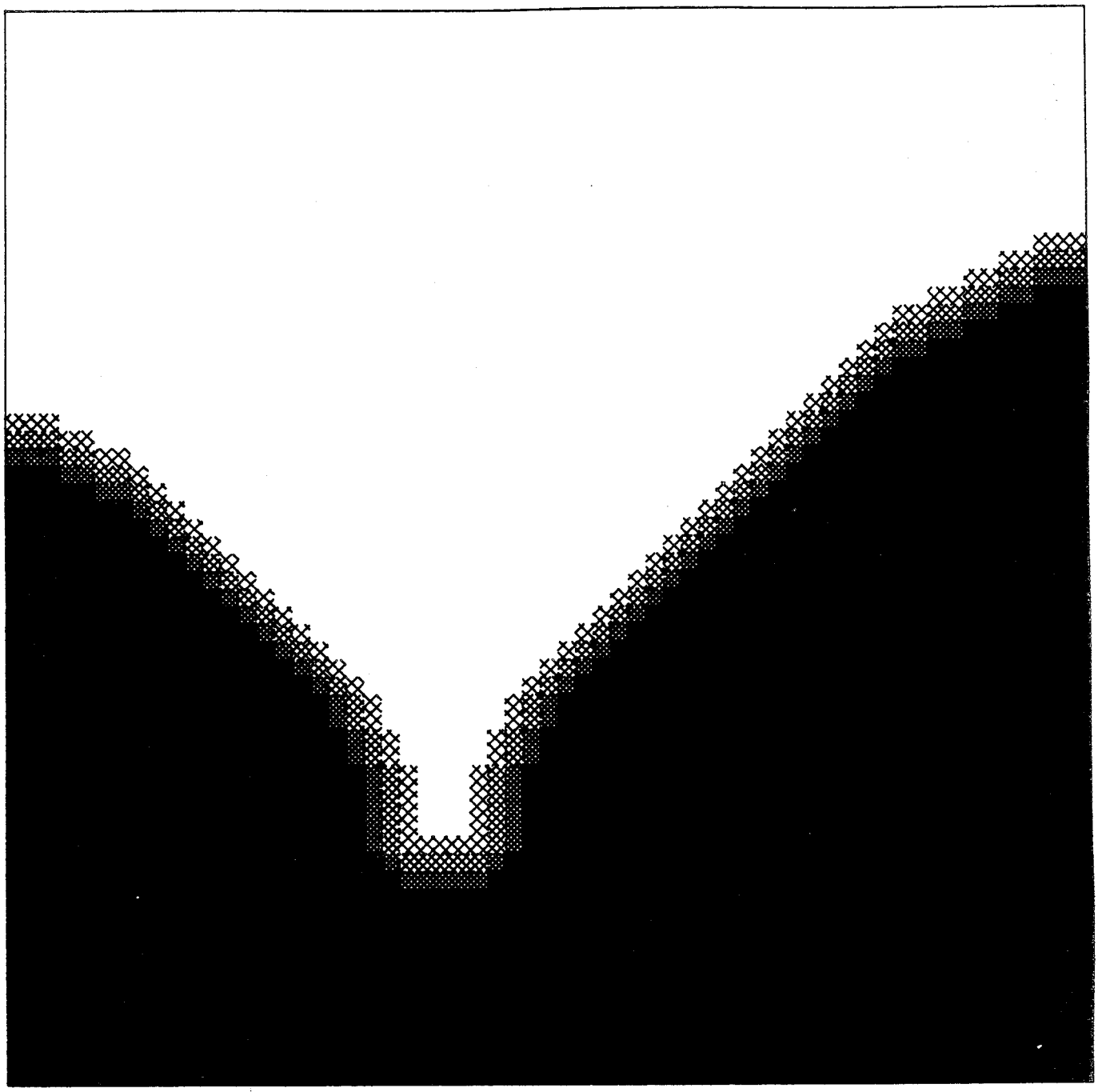
There may be situations where smoothing will not be beneficial. The price for the improvement in appearance is that differences may be smeared out and harder to locate. The benefit of smoothing becomes greater as the inconsistency of the data becomes greater. For highly inconsistent data, smoothing may be necessary to obtain convergence (3). Thus, smoothing is an important option, though it can reduce resolution in some instances.

DISCUSSION OF TOMOGRAPHIC PROGRAM BOMTOM

BOMTOM is available in both FORTRAN and BASICA, as was requested by mining companies. Appendix A contains a listing in FORTRAN. A source code in BASICA and compiled executable programs in both languages are available. The FORTRAN program was compiled with a Ryan-McFarland[3] compiler, version 2.11. Programs may be requested from the author at Bureau of Mines, 5629 Minnehaha Avenue South, Minneapolis, MN 55417. Suggestions for improving the program are invited.

Figure 18 is a flow diagram showing the main sections of the program. Program comments explain how to input data and run the program. BOMTOM allows the user to specify interactively the names of the input file, the output file for printing, and the output data file that can be used as an input file when resuming a run. The number of files can become large, so a systematic method of naming is helpful.

[3]Reference to specific products does not imply endorsement by the Bureau of Mines.



KEY

< 3.70 km/s

3.70– 3.82 km/s

3.82– 3.94 km/s

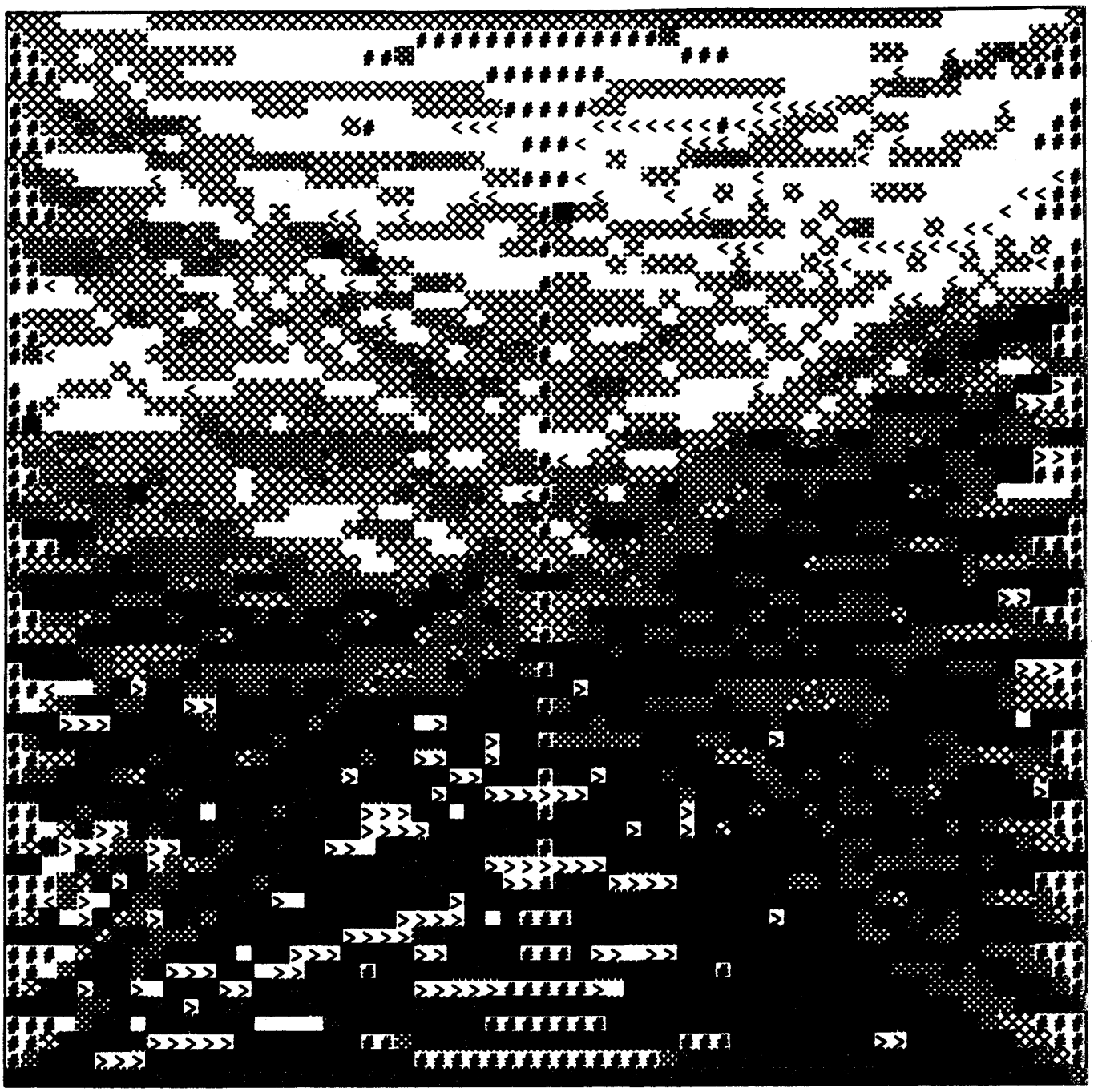
3.94– 4.06 km/s

4.06– 4.18 km/s

4.18– 4.30 km/s

> 4.30 km/s

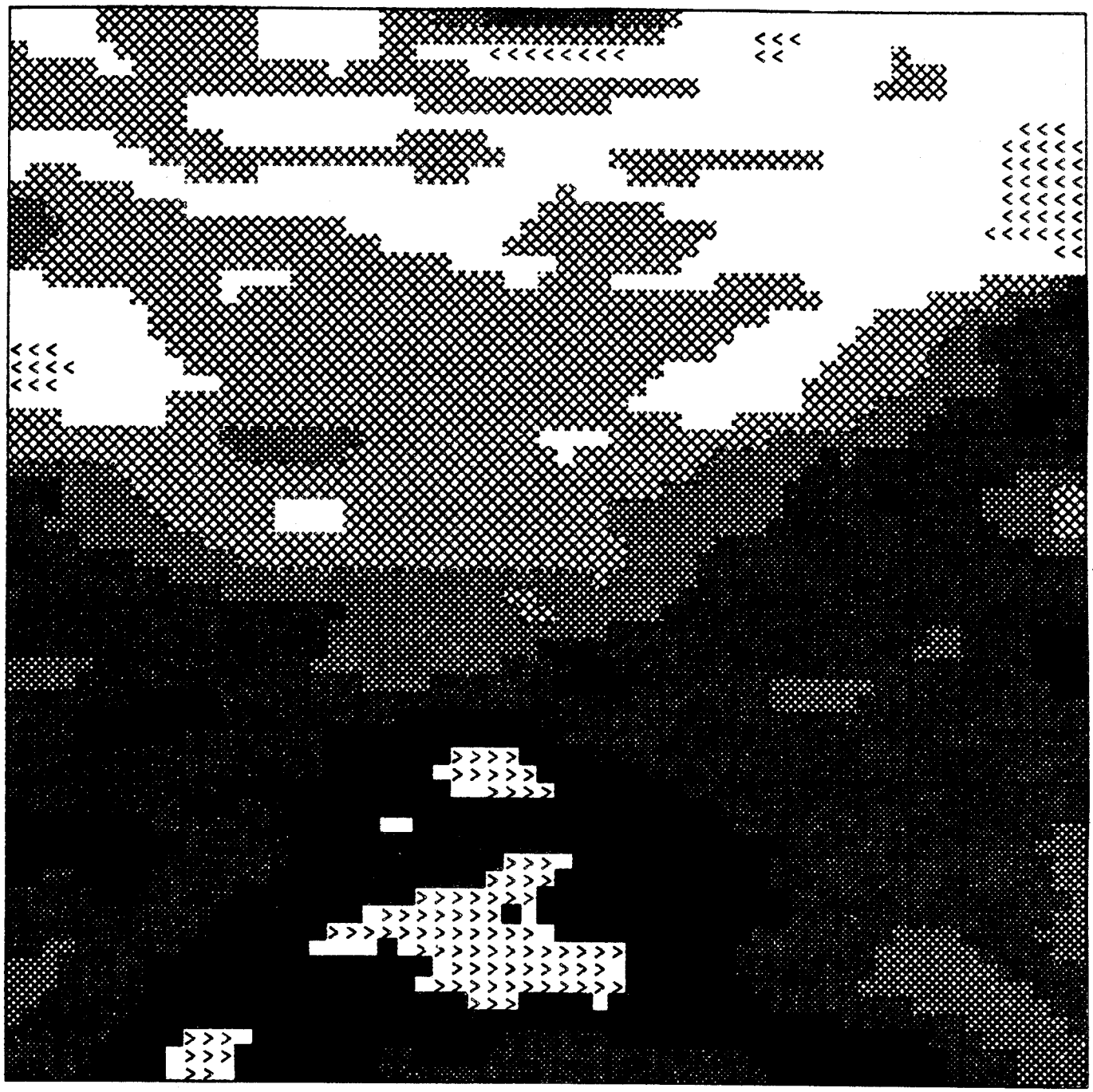
FIGURE 14.—Model velocity distribution for a 61- by 61-pixel grid.



KEY

< 3.70 km/s	4.06– 4.18 km/s
3.70– 3.82 km/s	4.18– 4.30 km/s
3.82– 3.94 km/s	> 4.30 km/s
3.94– 4.06 km/s	Sum of path fractions = 0

FIGURE 15.—Reconstruction of rounded data, 61- by 61-pixel grid, no smoothing.



KEY

< 3.70 km/s

3.70– 3.82 km/s

3.82– 3.94 km/s

3.94– 4.06 km/s

4.06– 4.18 km/s

4.18– 4.30 km/s

> 4.30 km/s

FIGURE 16.—Reconstruction of rounded data, 61- by 61-pixel grid, with smoothing.

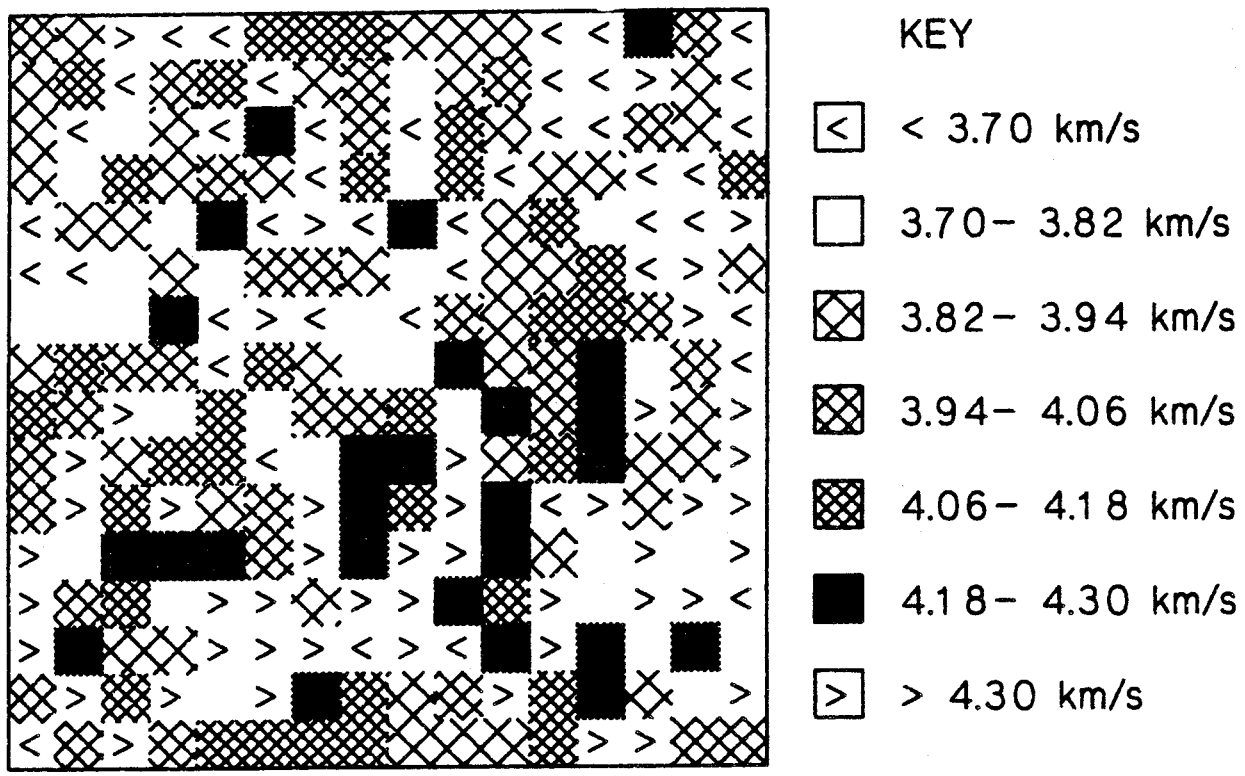


FIGURE 17.—Reconstruction of rounded data, 16- by 16-pixel grid, no smoothing.

For example, using the same descriptive name for the two output files except giving one the extension .OUT and the other the extension .DAT helps to keep the two files associated. To assist new users in creating the input file, BOMTOM provides interactive guidance. If an input file is specified that BOMTOM cannot find, BOMTOM asks if the user wants to create a file with that name. If the user does, BOMTOM starts a file with options set to typical values, and then asks for data input step by step. After the input file is created, the user can choose to continue running the program or stop. Creating the input file with a word processor is faster, but users may find BOMTOM's guidance helpful at first.

BOMTOM uses rectangular pixels. The size of the pixel is determined by the number of rows and columns of pixels that are specified for the width and height to be considered. The width and height are specified by the positions of the centers of the first (top left) and last (bottom right) pixels. BOMTOM assumes for purposes of labeling that the transmitter and receiver are in the left and right boreholes, respectively.

BOMTOM allows either of two formats for inputting transmitter and receiver positions and travel times. The first format is to enter all the positions with an identification number assigned to each position, then enter the travel times along with the corresponding identifiers for the transmitter and receiver positions. The second format is to enter the positions and the corresponding travel time for each ray path on one line. The first method is more common, as it requires fewer numbers to be entered. It is well suited for data where most of the receiver positions are repeated for each transmitter position. Input files for synthetic data (ITRMAX = -1) and input files created interactively are in that format. The second method provides complete flexibility because positions are input for every path separately. For either method, the identification numbers do not need to be sequential. If some positions are not used, there is no need to renumber the identifiers.

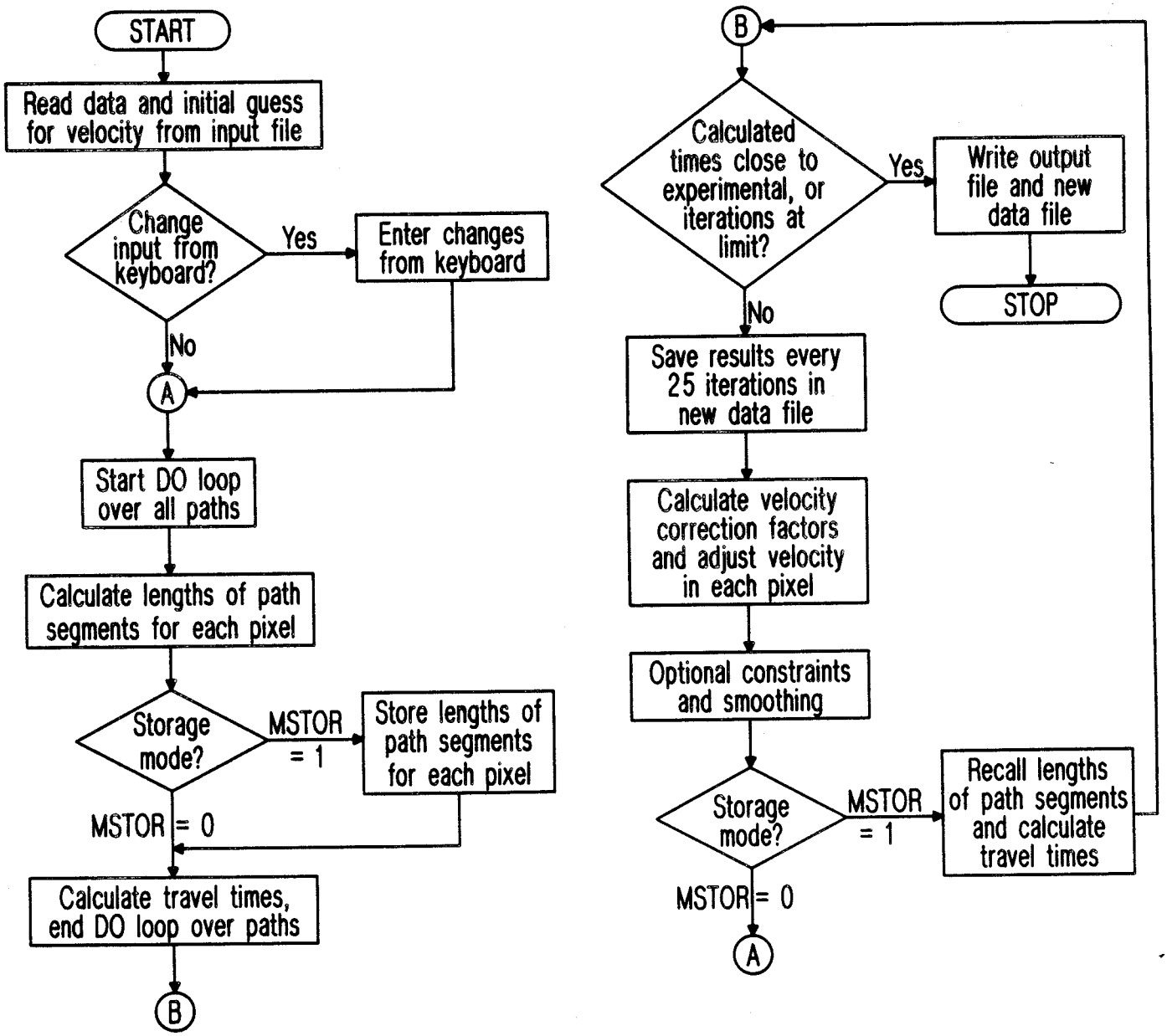


FIGURE 18.—Flow diagram of BOMTOM.

The positions are three-dimensional, with the z axis pointing down along the borehole axis. BOMTOM corrects for non-planar boreholes by projecting the ray paths onto the plane defined by the positions of the centers of the first and last pixels and the direction of the z axis. The positions of the first and last pixels should be chosen carefully so that the appropriate plane for projection is used and the transmitter and receiver positions are centered in pixels as well as possible. Centering positions in pixels provides the advantage that the pixel grid column does not change with depth for typical borehole deviations.

BOMTOM provides several other options. A weighting factor can be assigned to each travel time to account for differences in accuracy or reliability of the data. Because list-directed input format is used, the weighting factors cannot simply be left out; BOMTOM must be told whether the weighting factors are used by specifying MWT as 0 or 1 for not using or using weighting factors.

The user can choose either multiplicative or additive correction factors by specifying MCOR as 0 or 1, respectively. The factors generally give similar results. In some cases additive correction factors have given slightly faster convergence. Convergence can be accelerated by adjusting the relaxation parameter, RELAX, but this adjustment demands the user's time and may not be worthwhile if only a few runs are planned. If RELAX is too large the solution is degraded. The convergence and solution are very sensitive to RELAX with multiplicative correction factors.

The constraint that the velocity is not varied for pixels containing a transmitter or receiver is activated by setting MBORE = 1. The pixels at the borehole are those for which the velocity is most probably known. An option for fixing the velocity in columns at the sides is also provided by setting NFXRHT and NFXLFT equal to the desired number of columns. The program is easy to modify to fix the velocities in other pixels because the array LGLCOR controls whether the velocity is varied for a corresponding pixel. If LGLCOR(PIXEL) is set to FALSE in FORTRAN or 0 in BASICA, then the velocity in the corresponding pixel will not be varied.

An option, MDIF, is provided to allow comparisons with prior runs. The preferred tomographic procedure is to acquire data before the leachate is injected, then compare the reconstructions for times after injection with those before injection. Taking the difference will help to compensate for the effects of geology. To use this option, set MDIF = 1, and specify the data file for the prior run with which the comparison will be made when the program asks for it. Since comparing is done with the data file for the prior run, no retyping of data is necessary. The only precaution is not to write over the data file.

To aid testing the uniqueness of solutions, BOMTOM provides an option for changing the initial velocity guesses. The value of INVOPT, 1, 2, or 3, determines how the initial guesses for velocity will be input. The initial guesses can be changed from the keyboard. If INVOPT = 1, then the initial guess is uniform for all pixels. If INVOPT = 2, then the initial guesses are input row by row. This option is convenient for changing the pattern of the initial guesses to test uniqueness without having to input a new guess for every pixel individually. For example, one run could be done with the initial velocity guesses higher in the top half of the grid than in the bottom half and the next run could be done with that relationship reversed, with no need to edit the input file. If INVOPT = 3, the initial guesses are input pixel by pixel. This option is the usual setting for reading data files that have been generated by BOMTOM, as when resuming or comparing runs. BOMTOM sets INVOPT = 3 when generating a data file.

Three different velocity ranges can be input. The user can specify limits for calculated velocities. If a velocity is outside the range, it is set to the closer limit. The test for being outside the limit is performed after optional averaging or smoothing. The velocity limits for scaled and graphical displays can also be specified. The ranges are specified separately for velocities and differences in velocities. If the input value of any of these velocity parameters is -99., BOMTOM proceeds with no limit on the corresponding velocity. Display limits are then taken to be the minimum and maximum calculated by BOMTOM. Thus, setting all of the velocity limits equal to -99. is a safe procedure when the appropriate limits are not known.

The steps followed by BOMTOM after inputting data are as follows:

1. For the first path (travel time), determine the pixel row and column for the corresponding transmitter and receiver locations.

2. Calculate the length of the path segment in each pixel, and calculate the travel time for that path segment in that pixel, using the initial guess for velocity. Accumulate the travel times corresponding to each pixel.

3. Take the accumulated travel times for the entire path as the calculated travel time. Compare it with the experimental travel time. Calculate the difference, DELTA, and the ratio, RATIO.

4. Do steps 1 through 3 for all paths. Calculate correction factors for the velocity in all pixels. BOMTOM can use either multiplicative or additive correction factors, weighted according to the fraction of the total path length in a pixel that came from a particular path. This assures that a path which just cuts a corner of a pixel does not influence the velocity for that pixel as much as a path that passes all the way through it, and that a long path does not have more total influence than a shorter one.

5. Employ, if desired, smoothing and/or constraints for reducing the effects of nonuniqueness.

6. Recalculate the travel times using the new guesses for the velocities. If MSTOR = 0, the lengths of path segments in each pixel are calculated every iteration. If MSTOR = 1, the lengths are calculated the first iteration only, then stored and recalled as needed. Use MSTOR = 1 if there is sufficient memory because the execution is faster. Very large pixel grids or BASICA may require MSTOR = 0.

7. Check whether any of the criteria for stopping iterations have been satisfied. Iteration stops if the number of iterations exceeds ITRMAX or if the average of the absolute values of the DELTA's is less than TOLABS. It also stops if the decrease in that value is less than TOLINC for two successive iterations. Two successive iterations are used in this test because the amount of improvement per iteration can fluctuate. If none of these criteria are satisfied, then repeat steps 1 through 7. When one is satisfied, then go to step 8.

8. Save the results in a data file specified by the user that can serve as an input file when resuming a run. This file avoids the need to start over when the user wants additional iterations. During running, results are saved every 25 iterations in the specified data file, and can be displayed graphically on the screen. Thus, the program can be interrupted and no more than 25 iterations will be lost. To stop the program, hold CONTROL-BREAK until the program tries to write to the screen after completing a save.

9. Print the results. The calculated velocities are the primary output. A scaled display is also given where all the velocities are represented by integers from 0 to 9. This display can serve as input for the user's graphical display, or as a compact display of relative velocities. A simple graphical output that can be printed on a dot matrix printer using ASCII characters is provided as an option. (Some dot matrix printers cannot print these characters.) The character for the lowest velocities within the specified range is a blank. To mark the edge of the grid, BOMTOM prints a horizontal line in the pixels at the top or bottom of the grid that would otherwise be blank. Velocities below and above the specified range are represented by < and >, respectively. Pixels for which the sum of the fractions of path lengths (see step 4) is less than that specified by SMFPLL are represented by #. When the grid width is less than 40 pixels, graphical characters are printed twice to provide a better proportioned display.

The techniques for calculating the lengths of the path segments in each pixel and for associating segments with pixels merit some discussion. Together, they take the largest arrays and a large share of the execution time. This program was written with the goal of saving array space to permit running it on a personal computer. One way to save a great deal of space is to recalculate the lengths every iteration instead of storing and recalling them. To avoid excessive execution time, such calculations need to be performed more efficiently than the standard procedure.

In the usual technique for calculating the segments (see program in reference 9), the path is considered pixel by pixel. In each pixel a calculation is made to determine whether the path leaves the side, top or bottom of that pixel. The length of the segment in the pixel is then calculated, and the index for the next pixel is determined. The entry point for the next pixel is calculated, and the process is repeated until the end of the ray is reached.

That procedure provides a compact program, and is satisfactory when the segments are calculated only once. However, if the segment lengths are recalculated during each iteration to save space, then a faster method is desirable. BOMTOM calculates each point where the path will cross a horizontal grid line of the pixels. It then calculates the corresponding column and row. The length of rays in pixels where no crossing of a horizontal grid line occurs can be calculated quickly in a DO LOOP. Subroutines were avoided in the iterative calculations because calling a subroutine uses extra time. (Passing variables to subroutines is a common source of errors when modifying a program, so avoiding subroutines also facilitates inserting modifications.)

To save space when the path segments are stored and recalled, the segments are stored in the one-dimensional array, DP. DP is filled sequentially with segments, so it contains no blanks. The one-dimensional array, IDPIX, stores the pixel number for the corresponding path segment. It, too, is filled sequentially and contains no blanks.

SUMMARY AND CONCLUSIONS

Crosshole seismic or electromagnetic surveying with tomographic reconstruction is promising as an aid to monitoring leachate during in situ mining. However, crosshole data do not provide unique reconstructions. One cause of nonuniqueness is that crosshole data cannot determine the position of a vertical strip passing completely through the zone crossed by the ray paths, and cannot determine independently its width and velocity. The nonuniqueness persists even when the number of travel times exceeds the number of pixels.

Calculations were performed with synthetic data of the type expected from in situ mining above the water table, with a dip in the leachate level between injection wells. The seismic velocity distribution was modeled by attributing a velocity increase of 10 pct to the leachate.

The simulations generated a high-velocity artifact near the dip between the wells. Constraining the velocity at the top and bottom of the pixel grid not to vary laterally reduced the artifact. Applying the additional constraint of fixing the velocities in the boreholes at known values reduced the artifact slightly more. The reconstruction improved significantly when the upper velocity limit was close to the highest velocity in the model. However, the appropriate upper limit may not be well known for field data.

The simulations demonstrated the need to consider nonuniqueness when performing such calculations, and the desirability of surrounding the region to be investigated by an area of known or constrained velocities.

REFERENCES

1. Kehrman, R. F. Detection of Lixiviant Excursions With Geophysical Resistance Measurements During In Situ Mining (contract J0188080, Westinghouse Electric Corp.). BuMines OFR 5-81, 1979, 156 pp.; NTIS PB 81-171324.

2. Ramirez, A. L. Recent Experiments Using Geophysical Tomography in Fractured Granite. Proc. of the IEEE, v. 74, No. 2, Feb. 1986, pp. 347-352.

3. Dines, K. A., and R. J. Lytle. Computerized Geophysical Tomography. Proc. of the IEEE, v. 67, No. 7, July 1979, pp. 1065-1073.

4. Chiu, S. K. L., E. R. Kanasewich, and S. Phadke. Three-Dimensional Determination of Structure and Velocity by Seismic Tomography. Geophysics, v. 51, No. 8, Aug. 1986, pp. 1559-1571.

5. Bishop, T. N., K. P. Bube, R. T. Cutler, R. T. Langan, P. L. Love, J. R. Resnick, R. T. Shuey, D. A. Spindler, and H. W. Wyld. Tomographic Determination of Velocity and Depth in Laterally Varying Media. Geophysics, v. 50, No. 6, June 1985, pp. 903-923.

6. Phillips, W. S., P. A. Johnson, and J. N. Albright. Cross-Hole Tomography at Rifle, Colorado. Poster Paper 1.13, Soc. Expl. Geol. Annu. Meeting, 1986, pp. 228-230.

7. Gustavsson, M., S. Ivansson, and J. Pihl. Seismic Borehole Tomography--Measurement System and Field Studies. Proc. of the IEEE, v. 74, No. 2, Feb. 1986, pp. 339-346.

8. Peterson, J. N., B. N. P. Paulson, and T. V. McEvilly. Applications of Algebraic Reconstruction Techniques to Crosshole Seismic Data. Geophysics, v. 50, No. 10, Oct. 1985, pp. 1566-1580.

9. Peterson, J. E., Jr. The Application of Algebraic Reconstruction Techniques to Geophysical Problems. Ph.D. Thesis, Lawrence Berkeley Laboratory, Univ. CA, LBL-21498, Apr. 1986, 188 pp.

10. Foss, M. F., and R. J. Leckenby. Coal Mine Hazard Detection Using In-Seam Ground-Penetrating-Radar Transillumination. BuMines RI 9062, 1987, 27 pp.

11. Worthington, M. H. An Introduction to Geophysical Tomography. First Break, v. 2, pt. 11, Nov. 1984, pp. 20-26.

12. Ivansson, S. Seismic Borehole Tomography--Theory and Computational Methods. Proc. of the IEEE, v. 74, No. 2, Feb. 1986, pp. 328-338.

13. Wattrus, N. J. Two-Dimensional Velocity Anomaly Reconstruction by Seismic Tomography. Ph.D. Thesis, Univ. MN, Minneapolis, MN, 1984, 245 pp.

14. Thill, R. E., and J. A. Jessop. Effects of Water Saturation on Acoustic Wave Velocity. Soc. Min. Eng. AIME preprint 86-148, 1986, 13 pp.

15. GEOSEISMO OY (Pasuunakuja 1, SF-00420 Helsinki, Finland). Program VIBROVISION. 1986.

APPENDIX A.-- LISTING OF PROGRAM BOMTOM

```
      PROGRAM BOMTOM
C*************************************************************************
C*************************************************************************
C  BOMTOM (BUREAU OF MINES TOMOGRAPHY)IS A SIMULTANEOUS ITERATIVE
C   RECONSTRUCTION TECHNIQUE (SIRT) TOMOGRAPHIC PROGRAM FOR CROSSHOLE
C   SEISMIC TRAVEL TIMES.  BOMTOM WILL GENERATE SYNTHETIC TRAVEL TIMES
C   IF THE ITERATION LIMIT EQUALS -1.
C  INPUT, OUTPUT AND DATA FILE NAMES ARE ENTERED FROM THE KEYBOARD. INTERACTIVE
C   INSTRUCTIONS ARE GIVEN FOR CREATING A NEW INPUT FILE IF THE SPECIFIED INPUT
C   FILE IS NOT FOUND.  THERE IS AN OUTPUT FILE WITH TITLES FOR PRINTING, AND A
C   DATA FILE IN WHICH RESULTS ARE SAVED EVERY 25 ITERATIONS AND AT THE END.
C   THIS DATA FILE CAN BE USED AS THE INPUT FILE WHEN RESUMING A RUN.
C  OPTIONS CAN BE CHANGED FROM THE KEYBOARD INTERACTIVELY.  OPTIONS INCLUDE
C   APPLYING CONSTRAINTS, SMOOTHING, AND CHANGING INITIAL VELOCITY GUESSES.
C  EXECUTION CAN BE STOPPED BY HOLDING CONTROL-BREAK UNTIL BOMTOM WRITES TO
C   THE SCREEN, EVERY 25 ITERATIONS.
C  CORRECTION FACTORS ARE WEIGHTED BY THE FRACTION OF A GIVEN RAY IN A GIVEN
C   PIXEL (CELL) DIVIDED BY THE SUM OF FRACTIONS OF RAY PATHS IN THAT PIXEL,
C   AND BY AN OPTIONAL WEIGHT ACCOUNTING FOR RELATIVE RELIABILITY OF DATA.
C  WRITTEN BY DARYL TWEETON, U S BUREAU OF MINES, MINNEAPOLIS, 1986.
C*************************************************************************
C  OPERATION OF PROGRAM                                  LABEL NUMBERS
C   READ OR CREATE INPUT FILE                              100-299
C   WRITE OPTIONS AND DATA IN OUTPUT FILE                  300-399
C   CALCULATE GENERAL GEOMETRIC FACTORS                    400-499
C   CALCULATE LENGTHS OF PATH SEGMENTS                     500-599
C   ACCUMULATE CORRECTION FACTORS FOR EACH PIXEL           600-649
C   APPLY WEIGHTED CORRECTION FACTORS TO VELOCITIES        650-699
C   AVERAGING AND SMOOTHING OPTIONS                        700-799
C   SAVE RESULTS IN DATA FILE                              800-899
C   WRITE RESULTS IN OUTPUT FILE                           900-999
C*************************************************************************
C  DESCRIPTION OF DATA INPUT READ FROM INPUT FILE.
C
C  UNITS FOR LENGTH, TRAVEL TIMES, AND VELOCITIES MUST BE SELF-CONSISTENT.
C  USUAL UNITS ARE LENGTH IN METERS, TIME IN MILLISECONDS, VELOCITY IN KM/SEC.
C  FOR LABELING IN THE PROGRAM, TRANSMITTERS ARE ASSUMED TO BE ON LEFT AND
C   RECEIVERS ON RIGHT SIDE.
C  PIXEL 1 IS AT TOP LEFT.  PIXEL 2 IS NEXT PIXEL IN TOP ROW.
C
C  ALL DATA ARE IN LIST-DIRECTED FORMAT.  INPUT NUMBERS MUST BE SEPARATED BY
C   A SPACE OR COMMA.  THEY DO NOT NEED TO BE IN PARTICULAR COLUMNS.
C  *************************************************************************
C  HEADER
C
C  ONE LINE OF COMMENTS FOR THE TOP OF OUTPUT FILE.  LEAVE FIRST COLUMN BLANK.
C*************************************************************************
C  MDATA     MWT     MSTOR     MCOR     MBORE     MSMTH     MDIF
C
```

```
C  USE MDATA=0 TO INPUT TRAVEL TIMES SEPARATELY FROM TRANSMITTER AND RECEIVER
C   POSITIONS WITH POSITION IDENTIFICATION NUMBERS.  IDENT NUMBERS NEED NOT
C   BE SEQUENTIAL, BUT MUST BE INTEGERS WITH ABSOLUTE VALUE LESS THAN 32767.
C  USE MDATA=1 FOR POSITIONS AND CORRESPONDING TIMES ON THE SAME LINE.
C  SET MWT=1 TO USE INDIVIDUAL WEIGHTING FACTORS FOR ACCURACY OF TRAVEL TIMES.
C  USE MSTOR=0 TO RECALCULATE PATH SEGMENTS EACH ITERATION OR =1 TO STORE AND
C   RECALL PATH SEGMENTS (MORE MEMORY BUT FASTER EXECUTION). SEE COMMENTS
C   FOLLOWING DIMENSION STATEMENT.
C  USE MCOR=0 FOR MULTIPLICATIVE CORRECTION FACTORS AND =1 FOR ADDITIVE.
C   THEY GIVE SIMILAR RESULTS.
C  SET MBORE=1 TO FIX VELOCITIES IN PIXELS CONTAINING TRANSMITTER OR RECEIVER
C   POSITIONS TO THE INITIAL VALUES.  WOULD BE USED WITH BOREHOLE SONIC LOGS.
C  SET MSMTH=1 FOR SMOOTHING AFTER EACH ITERATION.  WEIGHTING IS 20, 4, 1
C   FOR PIXEL, ITS NEAREST NEIGHBORS, AND CORNER NEIGHBORS, RESPECTIVELY.
C  SET MDIF=1 TO FIND DIFFERENCE IN VELOCITIES BETWEEN THIS AND A PRIOR RUN.
C  ************************************************************************
C  ITRMAX    ITRSAV     TOLABS      TOLINC     RELAX
C
C  ITRMAX = MAXIMUM NUMBER OF ITERATIONS ALLOWED.  SET ITRMAX = -1 TO GENERATE
C   SYNTHETIC TRAVEL TIMES FOR THE INITIAL VELOCITIES.
C  ITRSAV = NUMBER OF ITERATIONS THAT HAVE BEEN PERFORMED; 0 WHEN STARTING
C   A NEW RUN, NOT 0 WHEN CONTINUING RUN USING THE DATA FILE AS INPUT.
C  TOLABS AND TOLINC ARE CONVERGENCE CRITERIA FOR ENDING ITERATIONS. ITERATIONS
C   STOP IF THE AVERAGE ABSOLUTE VALUE OF THE DIFFERENCE BETWEEN CALCULATED AND
C   EXPERIMENTAL TRAVEL TIMES IS LESS THAN TOLABS, OR IF THE DECREASE IN THAT
C   AVERAGE ERROR IS LESS THAN TOLINC FOR TWO SUCCESSIVE ITERATIONS.
C  RELAX = RELAXATION PARAMETER.  LEAVING RELAX=1.0 IS SAFE, THOUGH
C   CONVERGENCE MAY BE QUICKER IF IT IS ADJUSTED.
C  ************************************************************************
C  IPDAT      IPIVG      IPFRC      IPERR      IPSEG     IPGRA     ISGRA
C
C  IPDAT=1 CAUSES PRINTING OF POSITIONS, TRAVEL TIMES, AND WEIGHTS.
C  IPIVG=1 CAUSES PRINTING OF INITIAL VELOCITY GUESSES.
C  IPFRC=1 CAUSES PRINTING OF THE SUM OF THE FRACTIONS OF PATHS IN EACH PIXEL.
C  IPERR=1 CAUSES PRINTING OF THE ERRORS (DELTA) FOR EACH TRAVEL TIME.
C  IPSEG=1 CAUSES PRINTING OF SEGMENT LENGTHS FOR EACH PATH AND PIXEL.
C  IPGRA=1 CAUSES PRINTING OF GRAPHICAL DOT MATRIX DISPLAY OF VELOCITIES.
C  ISGRA=1 CAUSES GRAPHICAL SCREEN DISPLAY OF VELOCITIES EVERY 25 ITERATIONS.
C   ANY VALUE OTHER THAN 1 SUPPRESSES THE CORRESPONDING PRINTING OR DISPLAY.
C  ************************************************************************
C  POSP1X    POSP1Y    POSP1Z    POSPNX   POSPNY    POSPNZ

C  POSP1X, POSP1Y, AND POSP1Z ARE THE X, Y, AND Z COORDINATES OF THE CENTER
C   OF PIXEL 1 (THE TOP LEFT PIXEL).
C  POSPNX, POSPNY, AND POSPNZ ARE THE X, Y, AND Z COORDINATES OF THE CENTER
C   OF PIXEL NPIXH*NPIXV (THE BOTTOM RIGHT PIXEL).
C  THEY SHOULD BE SET SO THAT THE TRANSMITTERS AND RECEIVERS ARE CENTERED AS
C   WELL AS POSSIBLE IN PIXELS.  IF PIXEL 1 IS TOO FAR ABOVE OR TO THE LEFT OF
C   THE PATHS, THEN PIXEL INDICES MAY BE LARGER THAN THE DIMENSION. IF PIXEL 1
C   IS BELOW OR TO THE RIGHT OF PATHS, THEN PIXEL INDICES WILL BE NEGATIVE.
C   CONVERSE DIRECTIONS APPLY TO PIXEL NPIXH*NPIXV.
C  ************************************************************************
```

```
C  NPIXH      NPIXV      NLITOP     NLIBOT      NFXLFT      NFXRHT
C
C  NPIXH=NUMBER OF COLUMNS OF PIXELS (COUNTING HORIZONTALLY).
C  NPIXV=NUMBER OF ROWS OF PIXELS (COUNTING VERTICALLY).
C   IF DESIRED PIXEL HEIGHT IS EPIXV, SET NPIXV=1+(POSPNZ-POSP1Z)/EPIXV.
C  NLITOP AND NLIBOT ARE THE NUMBER OF LATERALLY INVARIANT ROWS AT TOP AND
C   BOTTOM OF THE GRID.  THE VELOCITY IN EACH OF THOSE ROWS IS SET EQUAL TO
C   THE AVERAGE FOR THAT ROW AFTER EACH ITERATION.
C  NFXLFT AND NFXRHT ARE THE NUMBER OF COLUMNS AT THE LEFT AND RIGHT SIDES OF
C   THE GRID IN WHICH THE VELOCITIES ARE FIXED AT THE INITIAL VALUES.
C  ********************************************************************************

C  THE INPUT OF TRANSMITTER AND RECEIVER LOCATIONS DEPENDS ON MDATA AND ITRMAX.
C   ************************************
C   IF MDATA = 0 OR ITRMAX = -1, USE THE FOLLOWING FORM:
C
C   NPST         NPSR         ADJDIS
C
C   NPST, NPSR ARE THE NUMBER OF TRANSMITTER, RECEIVER POSITIONS
C   ADJDIS IS ADDED TO ALL HORIZONTAL DISTANCES BETWEEN BOREHOLES.  IT IS ADDED
C    TO THE RECEIVER POSITION.
C   ********
C   IDPST(IPST)   POSTX(IPST)   POSTY(IPST)   POSTZ(IPST)
C    (ONE LINE FOR EACH IPST = 1 TO NPST)
C   IDPST(IPST) IS THE IDENTIFICATION NUMBER OF THE TRANSMITTER POSITION.
C    THEY DO NOT NEED TO BE SEQUENTIAL.
C   POSTX, POSTY, AND POSTZ ARE X, Y, AND Z COORDINATES OF TRANSMITTER
C    POSITIONS.  Z INCREASES WITH DEPTH.
C   ********
C   IDPSR(IPSR)   POSRX(IPSR)   POSRY(IPSR)   POSRZ(IPSR)
C    (ONE LINE FOR EACH IPSR = 1 TO NPSR)
C   IDPSR(IPSR) IS THE IDENTIFICATION NUMBER OF THE RECEIVER POSITION.
C   POSRX, POSRY, AND POSRZ ARE X, Y, AND Z COORDINATES OF RECEIVER POSITIONS.
C   ********
C   NEXPTT       ADJTT
C
C   NEXPTT IS THE NUMBER OF EXPERIMENTAL TRAVEL TIMES.
C   ADJTT IS ADDED TO ALL EXPERIMENTAL TRAVEL TIMES.
C   ********
C   IF ITRMAX = -1, SKIP THIS INPUT
C   IDEXTT(IEXPTT)  IDEXTR(IEXPTT)  EXPTT(IEXPTT)    (WEIGHT(IEXPTT), IF MWT=1)
C    (ONE LINE FOR EACH IEXPTT = 1 TO NEXPTT)
C   IDEXTT AND IDEXTR ARE THE IDENTIFICATION NUMBERS OF THE TRANSMITTER AND
C    RECEIVER POSITIONS FOR THE TRAVEL TIME EXPTT(IEXPTT)
C   WEIGHT(IEXPTT) IS AN OPTIONAL WEIGHTING FACTOR FOR ACCURACY OF DATA.
C    CORRECTION FACTORS ARE MULTIPLIED BY THE CORRESPONDING WEIGHT. IF ALL DATA
C    ARE EQUALLY RELIABLE, SET EVERY WEIGHT = 1.0 OR SET MWT=0.
C   ************************************
C   IF MDATA = 1, USE THE FOLLOWING FORM:
C
C   NEXPTT      ADJDIS      ADJTT
C   *******
```

```
C   IDPATH(IEXPTT)  POSTX(IEXPTT)  POSTY(IEXPTT)  POSTZ(IEXPTT)  POSRX(IEXPTT)
C    POSRY(IEXPTT)  POSRZ(IEXPTT)  EXPTT(IEXPTT)    (WEIGHT(IEXPTT),IF MWT = 1)
C    (ONE LINE FOR EACH IEXPTT = 1 TO NEXPTT)
C   IDPATH(IEXPTT) IS THE IDENTIFICATION NUMBER OF A PATH AND
C    THE CORRESPONDING TRAVEL TIME. THEY DO NOT NEED TO BE SEQUENTIAL.
C   EXPTT(IEXPTT) IS THE EXPERIMENTAL TRAVEL TIME FOR PATH IEXPTT.
C   ************************************************************************

C  SMFPLL   VLOCAL   VHICAL    VLODSP    VHIDSP    VLODIF    VHIDIF    INVOPT
C
C  SMFPLL IS THE LOWER LIMIT FOR THE SUM OF FRACTIONS OF PATH LENGTHS IN A
C   PIXEL (SEGMENT LENGTH IN PIXEL/LENGTH OF PATH FROM TRANS TO REC).  IF THE
C   SUM IS LESS THAN OR EQUAL SMFPLL, THEN THE VELOCITY FOR THE PIXEL IS
C   DISPLAYED AS ###.
C
C  VLOCAL AND VHICAL ARE THE LOW AND HIGH LIMITS FOR THE CALCULATED VELOCITIES.
C   IF A VELOCITY IS OUT OF THAT RANGE, IT IS SET = THE LIMIT.  IF NO LIMIT IS
C   DESIRED, SET THE LIMIT TO -99.
C
C  A COMPACT DISPLAY WITH VELOCITIES SCALED TO THE RANGE 0 TO 9 IS GENERATED,
C    AND A DOT MATRIX GRAPHICAL DISPLAY IS AN OPTION (SET IPGRA =1).
C  IF VLODSP (OR VHIDSP) IS -99., THEN THE LOWEST (OR HIGHEST) VELOCITY
C   DISPLAYED IS THE MINIMUM (OR MAXIMUM) CALCULATED BY BOMTOM.
C  IF VLODSP (OR VHIDSP) IS NOT - 99., THEN THE LOWEST (OR HIGHEST) VELOCITY
C   DISPLAYED IS VLODSP (OR VHIDSP).
C  WHEN DIFFERENCES BETWEEN THE VELOCITIES IN THIS RUN AND A PRIOR RUN ARE
C   DISPLAYED, VLODSP AND VHIDSP ARE REPLACED BY VLODIF AND VHIDIF.
C
C  INVOPT SPECIFIES THE OPTION FOR INPUTTING THE INITIAL GUESSES OF VELOCITIES.
C
C  IF INVOPT = 1, INPUT IS UNIFORM AND THE FORMAT IS:
C   V(1)
C   BOMTOM SETS INITIAL VELOCITY = V(1) FOR EVERY PIXEL.
C
C  IF INVOPT = 2, INPUT IS ROW BY ROW AND THE FORMAT IS:
C   IROWV     V(IROWV)                      (ONE LINE FOR EACH ROW, NPIXV LINES)
C   IROWV IS THE ROW NUMBER.  ROWS CAN BE INPUT IN ANY ORDER.
C   BOMTOM SETS INITIAL VELOCITIES = V(IROWV) FOR ALL PIXELS IN THAT ROW.
C    THIS OPTION FACILITATES VARYING THE PATTERN OF INITIAL GUESSES TO TEST THE
C    UNIQUENESS OF A SOLUTION.
C
C  IF INVOPT = 3, INPUT IS PIXEL BY PIXEL AND THE FORMAT IS:
C   IROWV   (V(I),I=1,NPIXH)              (ONE LINE FOR EACH ROW, NPIXV LINES)
C   IROWV IS THE ROW NUMBER.
C   V(I) IS THE INITIAL VELOCITY FOR ROW IROWV, COLUMN I.
C   THIS OPTION IS USED WHEN RESUMING A RUN USING A DATA FILE AS INPUT.
C*******END OF DESCRIPTION OF DATA READ FROM INPUT FILE************************
C******DEFINITIONS OF SOME OTHER VARIABLES USED IN BOMTOM**********************
C
C  DPIXH=DISTANCE FROM CENTER OF LEFT PIXEL TO CENTER OF RIGHT PIXEL.
C  DPIXV=DISTANCE FROM CENTER OF TOP PIXEL TO CENTER OF BOTTOM PIXEL.
C
```

```
C  POSTH AND POSRH ARE THE HORIZONTAL COORDINATES OF THE TRANSMITTER AND
C   RECEIVER IN THE COORDINATE SYSTEM OF THE PIXEL GRID.
C
C  ICOLT AND ICOLR ARE THE PIXEL COLUMNS CONTAINING POSTH AND POSRH.
C  IROWT AND IROWR ARE THE PIXEL ROWS CONTAINING POSTZ AND POSRZ.
C
C  NACROS IS THE NUMBER OF TIMES A PATH CROSSES PIXEL ROW BOUNDARY LINES.
C  TCROS(I), I=1,NACROS, IS THE HORIZONTAL DISTANCE FROM A TRANSMITTER
C   TO A POINT WHERE THE PATH CROSSES A PIXEL ROW BOUNDARY LINE.
C  ICOL(I) IS THE COLUMN IN WHICH CROSSING I OCCURS.
C  EDCROS(I) IS THE HORIZONTAL DISTANCE FROM THE LEFT EDGE OF ICOL(I)
C   TO CROSSING I.
C
C  D(J),J=1 TO BETWEEN SQRT(NPIXH**2+NPIXV**2) AND 2*SQRT(NPIXH**2+NPIXV**2),
C   IS THE LENGTH OF A PATH SEGMENT IN A PIXEL.  J DOES NOT DESIGNATE A
C   PARTICULAR PIXEL.
C  SMFPIX(IPIXEL),IPIXEL=1 TO NPIXH*NPIXV,IS THE SUM OF THE FRACTION OF PATH
C   LENGTHS IN PIXEL(IPIXEL).
C  IPIXSG(IPIXEL),IPIXEL=1 TO NPIXH*NPIXV, IS AN ARRAY KEEPING TRACK OF WHICH
C   PATH SEGMENTS ARE IN WHICH PIXELS.
C  LGLCOR(IPIXEL),IPIXEL=1 TO NPIXH*NPIXV, IS A LOGICAL ARRAY.  IF LGLCOR(I)
C   IS FALSE, THEN V(IPIXEL) IS NOT CHANGED.
C
C  DELT(I)=EXPTT(I)-CALCULATED TRAVEL TIME FOR PATH I.
C  RATIO(I)=EXPTT(I)/CALCULATED TRAVEL TIME FOR PATH I.
C*******************************************************************************
C*******************************************************************************
      IMPLICIT INTEGER*2(I-N)
      INTEGER*4 IARSG,IARSGS,NTOTSG,IOS
      CHARACTER*12 INFIL,OUTFIL,DATFIL,OLDFIL
      CHARACTER*79 HEADER,HEADF
      CHARACTER*5 VALPHA
      CHARACTER*1 VSCALE
      LOGICAL*1 LGLCOR
      LOGICAL FLXT
C
       DIMENSION IVSCAL(65),VALPHA(65),VSCALE(65),
     1 ICOL(65),TCROS(65),EDCROS(65),   COR(4000),IPIXSG(4000),
     2  LGLCOR(4000),SMFPIX(4000),V(4000),VSMTH(4000),WTSMF(4000),
     3 DELT(455),EXPTT(455),EXTADJ(455),IDEXTT(455),IDEXTR(455),
     4  IDPATH(455),RATIO(455),WEIGHT(455),
     5 IDPST(455),POSTX(455),POSTY(455),POSTZ(455),
     6 IDPSR(455),POSRX(455),POSRY(455),POSRZ(455),
     7 D(150),IPP(455),DPPATH(455),DP(24000),IDPIX(24000)
C
C  IF ALL OPTIONS IN SECOND LINE OF INPUT ARE SET TO 0, THEN DIMENSIONS ARE:
C               IVSCAL(NPIXH),VALPHA(NPIXH),VSCALE(NPIXH),
C ICOL(NPIXV),TCROS(NPIXV),EDCROS(NPIXV), COR(NPIXH*NPIXV),IPIXSG(NPIXH*NPIXV),
C  LGLCOR(NPIXH*NPIXV),SMFPIX(NPIXH*NPIXV),V(NPIXH*NPIXV),VSMTH(1),WTSMF(1),
C DELT(NEXPTT),EXPTT(NEXPTT),EXTADJ(NEXPTT),IDEXTT(NEXPTT),IDEXTR(NEXPTT),
C  IDPATH(1),RATIO(NEXPTT),WEIGHT(1),
C IDPST(NPST),POSTX(NPST),POSTY(NPST),POSTZ(NPST),
```

```
C IDPSR(NPSR),POSRX(NPSR),POSRY(NPSR),POSRZ(NPSR),
C D(ISG), ISG IS LESS THAN 2*SQRT(NPIXH**2 + NPIXV**2) AND IS GIVEN IN OUTPUT.
C IPP(1),DPPATH(1),DP(1),IDPIX(1)
C
C  IF NOT ALL OPTIONS ARE SET TO 0, THEN ALTER DIMENSIONS AS FOLLOWS:
C IF MDATA=1, IDPATH(NEXPTT),POSTX(NEXPTT),POSTY(NEXPTT),POSTZ(NEXPTT),
C    POSRX(NEXPTT),POSRY(NEXPTT),POSRZ(NEXPTT)
C IF MWT=1, WEIGHT(NEXPTT),WTSMF(NPIXH*NPIXV)
C IF MSTOR=1, IPP(NEXPTT),DPPATH(NEXPTT),DP(IARSG),IDPIX(IARSG),
C  WHERE IARSG IS LESS THAN ISG*NEXPTT  AND IS GIVEN IN OUTPUT.
C IF MSMTH=1, VSMTH(NPIXH*NPIXV)
C
      DATA IYES,JYES /1HY,1Hy/
      DATA MDATA,MWT,MSTOR,MCOR,MBORE,MSMTH,MDIF/0,0,1,1,0,0,0/
      DATA ITRMAX,ITRSAV,TOLABS,TOLINC,RELAX/200,0,0.0,0.0,1.0/
      DATA IPDAT,IPIVG,IPFRC,IPERR,IPSEG,IPGRA,ISGRA/0,1,0,0,0,1,1/
      DATA SMFPLL,VLOCAL,VHICAL,VLODSP,VHIDSP,VLODIF,VHIDIF,INVOPT/
     1 0.0,-99.,-99.,-99.,-99.,-99.,-99.,3/
  100 MDFLG=0
      ICRFLG=0
  101 WRITE(*,102)
  102 FORMAT(' ENTER NAME OF EXISTING INPUT FILE OR OF NEW FILE TO BE CR
     1EATED: ')
      READ(*,103) INFIL
  103 FORMAT(A12)
      INQUIRE(FILE=INFIL,EXIST=FLXT,ERR=128,IOSTAT=IOS)
      IF(FLXT) THEN
       OPEN(3,FILE=INFIL,STATUS='OLD',ERR=128,IOSTAT=IOS)
       GO TO 123
      END IF
  104 WRITE(*,105) INFIL
  105 FORMAT(' CREATE A NEW INPUT FILE NAMED ',A12,' ? (Y/N): ')
      READ (*,124) MYN
      IF(MYN.NE.IYES .AND. MYN.NE.JYES) GO TO 101
      WRITE(*,107)
  107 FORMAT(' ENTER ONE LINE OF HEADING FOR INPUT FILE',/,1X)
      READ(*,306) HEADER
      WRITE(*,*)('A NEW INPUT FILE HAS BEEN STARTED, WITH OPTIONS SET FO
     1R NO CONSTRAINTS.')
      WRITE(*,108)
  108 FORMAT(/,' ENTER X,Y,Z COORDINATES OF CENTER OF TOP PIXEL ON TRANS
     1MITTER SIDE (POSP1X,     POSP1Y,POSP1Z). THEY ARE USUALLY THE SAME
     2 AS COORDINATES OF TOP TRANS POSITION  FOR SYMMETRIC TRANS, REC PO
     3SITIONS.',/,1X)
      READ(*,*) POSP1X,POSP1Y,POSP1Z
      WRITE(*,109)
  109 FORMAT(' ENTER X,Y,Z COORDINATES OF CENTER OF BOTTOM PIXEL ON RECE
     1IVER SIDE (POSPNX,     POSPNY,POSPNZ). THEY ARE USUALLY THE SAME A
     2S COORDINATES OF BOTTOM REC POSITION FOR SYMMETRIC TRANS, REC POSI
     3TIONS.',/,1X)
      READ(*,*) POSPNX,POSPNY,POSPNZ
      WRITE(*,110)
```

```
110 FORMAT(' ENTER NUMBER OF PIXEL COLUMNS,ROWS (NPIXH,NPIXV).',/,' IF
   1 DESIRED PIXEL HEIGHT IS EPIXV, SET NPIXV=1+(POSPNZ-POSP1Z)/EPIXV.
   2',/,1X)
    READ(*,*) NPIXH,NPIXV
    NPXTOT=NPIXH*NPIXV
    WRITE(*,*)('ENTER NUM OF TRANS,REC POSITIONS (NPST,NPSR) ')
    READ(*,*) NPST,NPSR
    WRITE(*,111)
111 FORMAT(' ENTER DISTANCE CORRECTION TO BE ADDED TO ALL RECEIVER POS
   1ITIONS (ADJDIS)',/,1X)
    READ(*,*) ADJDIS
    DO 113 IPST=1,NPST
    WRITE(*,112)
112 FORMAT(' ENTER IDENT NUM, X,Y,Z COORDINATES OF TRANS POSITION (IDP
   1ST,POSTX,POSTY,POSTZ)',/,1X)
113 READ(*,*) IDPST(IPST),POSTX(IPST),POSTY(IPST),POSTZ(IPST)
    DO 115 IPSR=1,NPSR
    WRITE(*,114)
114 FORMAT(' ENTER IDENT NUM, X,Y,Z, COORDINATES OF REC POSITION (IDPS
   1R,POSRX,POSRY,POSRZ)',/,1X)
115 READ(*,*) IDPSR(IPSR),POSRX(IPSR),POSRY(IPSR),POSRZ(IPSR)
    WRITE(*,*)('ENTER NUM OF TRAVEL TIMES (NEXPTT) ')
    READ(*,*) NEXPTT
    WRITE(*,116)
116 FORMAT(' ENTER TIME CORRECTION TO BE ADDED TO EACH TRAVEL TIME (AD
   1JTT)',/,1X)
    READ(*,*) ADJTT
    DO 118 IEXPTT=1,NEXPTT
    WRITE(*,117)
117 FORMAT(' ENTER TRANS IDENT NUM, REC IDENT NUM, TRAVEL TIME (IDEXTT
   1,IDEXTR,EXPTT)',/,1X)
118 READ(*,*) IDEXTT(IEXPTT),IDEXTR(IEXPTT),EXPTT(IEXPTT)
    WRITE(*,*)('ENTER GUESS FOR UNIFORM INITIAL VELOCITY (V) ')
    READ(*,*) V(1)
    DO 119 IPIXEL=2,NPXTOT
119 V(IPIXEL)=V(1)
    OPEN(4,FILE=INFIL,STATUS='NEW',ERR=128,IOSTAT=IOS)
    ICRFLG=-99
    GO TO 805
120 WRITE(*,121) INFIL
121 FORMAT(1X,A12,' HAS BEEN CREATED. RUN PROGRAM ? (Y/N): ')
    READ (*,124) MYN
    IF(MYN.NE.IYES .AND. MYN.NE.JYES) THEN
     STOP
    END IF
    ICRFLG=0
    OPEN(3,FILE=INFIL,STATUS='OLD',ERR=128,IOSTAT=IOS)
123 WRITE(*,*)('ENTER NAME OF OUTPUT FILE FOR PRINTING: ')
    READ(*,103) OUTFIL
    INQUIRE(FILE=OUTFIL,EXIST=FLXT,ERR=128,IOSTAT=IOS)
    IF(.NOT.FLXT) GO TO 125
    WRITE(*,*)('FILE ALREADY EXISTS.  WRITE OVER IT?  (Y/N): ')
```

```
      READ (*,124) MYN
  124 FORMAT(A1)
      IF(MYN.NE.IYES .AND. MYN.NE.JYES) GO TO 123
  125 OPEN(2,FILE=OUTFIL,ERR=128,IOSTAT=IOS)
  126 WRITE(*,*)('ENTER NAME OF DATA FILE FOR SAVING RESULTS: ')
      READ(*,103) DATFIL
      INQUIRE(FILE=DATFIL,EXIST=FLXT,ERR=128,IOSTAT=IOS)
      IF(.NOT.FLXT) GO TO 130
      WRITE(*,*)('FILE ALREADY EXISTS.  WRITE OVER IT?  (Y/N): ')
      READ (*,124) MYN
      IF(MYN.NE.IYES .AND. MYN.NE.JYES) GO TO 126
      GO TO 130
  128 WRITE(*,129) IOS
  129 FORMAT(/,' ERROR #',I5,' IN INQUIRE OR OPEN STATEMENT.')
      GO TO 101
C
C******START READING INPUT FILE, LET OPERATOR EXAMINE SOME LINES*******
  130 READ(3,132) HEADER
      IF(MDFLG.NE.-999) WRITE(*,132) HEADER
  132 FORMAT(1X,A79)
      READ(3,*) MDATA,MWT,MSTOR,MCOR,MBORE,MSMTH,MDIF
      IF(MDFLG.NE.-999)WRITE(*,134)MDATA,MWT,MSTOR,MCOR,MBORE,MSMTH,MDIF
  134 FORMAT(' MDATA=',I1,'   MWT=',I1,'   MSTOR=',I1,
     1'   MCOR=',I1,'   MBORE=',I1,'   MSMTH=',I1,'    MDIF=',I1)
      READ(3,*) ITRMAX,ITRSAV,TOLABS,TOLINC,RELAX
      IF(MDFLG.NE.-999) WRITE(*,136) ITRMAX,ITRSAV,TOLABS,TOLINC,RELAX
  136 FORMAT(' ITRMAX=',I5,'  ITRSAV=',I5,'  TOLABS=',F10.7,
     1'  TOLINC=',F10.7,'  RELAX=',F7.3)
      READ(3,*) IPDAT,IPIVG,IPFRC,IPERR,IPSEG,IPGRA,ISGRA
      READ(3,*) POSP1X,POSP1Y,POSP1Z,POSPNX,POSPNY,POSPNZ
      READ(3,*) NPIXH,NPIXV,NLITOP,NLIBOT,NFXLFT,NFXRHT
      NPXTOT=NPIXH*NPIXV
       IF(MDATA.NE.1 .OR.ITRMAX.EQ.-1) THEN
       READ(3,*) NPST,NPSR,ADJDIS
       DO 138 IPST=1,NPST
  138  READ(3,*) IDPST(IPST),POSTX(IPST),POSTY(IPST),POSTZ(IPST)
       DO 140 IPSR=1,NPSR
  140  READ(3,*) IDPSR(IPSR),POSRX(IPSR),POSRY(IPSR),POSRZ(IPSR)
       READ(3,*) NEXPTT,ADJTT
        IF(ITRMAX.EQ.-1) THEN
        DO 141 IPST=1,NPST
        DO 141 IPSR=1,NPSR
        IDEXTT((IPST-1)*NPSR+IPSR)=IDPST(IPST)
  141   IDEXTR((IPST-1)*NPSR+IPSR)=IDPSR(IPSR)
        GO TO 150
        END IF
        IF(MWT.NE.1) THEN
        DO 142 IEXPTT=1,NEXPTT
  142   READ(3,*) IDEXTT(IEXPTT),IDEXTR(IEXPTT),EXPTT(IEXPTT)
        ELSE
        DO 144 IEXPTT=1,NEXPTT
  144   READ(3,*) IDEXTT(IEXPTT),IDEXTR(IEXPTT),EXPTT(IEXPTT),
```

```
     1   WEIGHT(IEXPTT)
         END IF
        ELSE
        READ(3,*) NEXPTT,ADJDIS,ADJTT
        IF(MWT.NE.1) THEN
        DO 146 IEXPTT=1,NEXPTT
   146  READ(3,*)IDPATH(IEXPTT),POSTX(IEXPTT),POSTY(IEXPTT),POSTZ(IEXPTT)
     1,POSRX(IEXPTT),POSRY(IEXPTT),POSRZ(IEXPTT),EXPTT(IEXPTT)
         ELSE
        DO 148 IEXPTT=1,NEXPTT
   148  READ(3,*)IDPATH(IEXPTT),POSTX(IEXPTT),POSTY(IEXPTT),POSTZ(IEXPTT)
     1,POSRX(IEXPTT),POSRY(IEXPTT),POSRZ(IEXPTT),EXPTT(IEXPTT),
     2 WEIGHT(IEXPTT)
         END IF
        END IF
   150 READ(3,*) SMFPLL,VLOCAL,VHICAL,VLODSP,VHIDSP,VLODIF,VHIDIF,INVOPT
       GO TO (156,160,164)INVOPT
       WRITE(*,154)
       WRITE(2,154)
   154 FORMAT(' INVOPT SPECIFIED INCORRECTLY, PROGRAM TERMINATED.',/,
     1 ' THIS ERROR IS USUALLY CAUSED BY THE WRONG NUMBER OF INPUT',/,
     2'   PARAMETERS SOMEWHERE BEFORE INVOPT IS READ.')
       GO TO 999
   156 READ(3,*) V(1)
        DO 158 IPIXEL=2,NPXTOT
   158  V(IPIXEL) = V(1)
       GO TO 168
   160  DO 162 IPIXV=1,NPIXV
        READ(3,*) IROWV, V(1+(IROWV-1)*NPIXH)
         DO 162 IPIXH=2,NPIXH
   162   V(IPIXH+(IROWV-1)*NPIXH) = V(1+(IROWV-1)*NPIXH)
       GO TO 168
   164  DO 166 IPIXV=1,NPIXV
   166  READ(3,*)IROWV,(V(IPIXH+(IROWV-1)*NPIXH),IPIXH=1,NPIXH)
   168 CLOSE(3)
       IF(MDFLG.EQ.-999) GO TO 992
       VLODFP=VLODIF
       VHIDFP=VHIDIF
C***********END OF READING INPUT FILE*********************************************
C***********START MODIFYING INPUT USING KEYBOARD, IF DESIRED.*****************
   170 NPXTOT=NPIXH*NPIXV
       DFACX=POSPNX-POSP1X
       DFACY=POSPNY-POSP1Y
       DPIXHS=(DFACX*DFACX + DFACY*DFACY)
       DPIXH=SQRT(DPIXHS)
       DPIXV=POSPNZ-POSP1Z
       EPIXH=DPIXH/(NPIXH-1)
       EPIXV=DPIXV/(NPIXV-1)
       WRITE(*,308) HEADER
       WRITE(*,*)(' CHANGE INPUT?  RUN PROGRAM OR ENTER NUMBER OF ANY OPT
      1ION YOU WANT TO CHANGE.')
       WRITE(*,*)('NUM  OPTION')
```

```
      WRITE(*,*)(' 1   RUN PROGRAM')
      IF(MSTOR.NE.1)WRITE(*,*)(' 2   MSTOR = 0, RECALCULATE PATH SEGMENT
     1S EACH ITERATION')
      IF(MSTOR.EQ.1)WRITE(*,*)(' 2   MSTOR = 1, STORE AND RECALL PATH SE
     1GMENTS')
      IF(MBORE.NE.1)WRITE(*,*)(' 3   MBORE = 0, VEL AT TRANS AND REC POS
     1 ARE VARIED UNLESS FIXED BY OPT 12')
      IF(MBORE.EQ.1)WRITE(*,*)(' 3   MBORE = 1, VELOCITIES AT TRANS AND
     1 REC POS ARE FIXED')
      IF(MSMTH.NE.1) WRITE(*,*)(' 4   MSMTH = 0, NO SMOOTHING')
      IF(MSMTH.EQ.1) WRITE(*,*)(' 4   MSMTH = 1, PERFORM SMOOTHING')
      IF(MDIF.NE.1)WRITE(*,*)(' 5   MDIF = 0, WILL NOT BE COMPARED WITH
     1 PRIOR RUN')
      IF(MDIF.EQ.1)WRITE(*,*)(' 5   MDIF = 1, WILL BE COMPARED WITH PRIO
     1R RUN')
      WRITE(*,173) ITRMAX,TOLABS,TOLINC,RELAX,IPDAT,IPIVG,IPFRC,IPERR,
     1IPSEG,IPGRA,ISGRA,NPIXH,NPIXV,EPIXH,EPIXV,NLITOP,NLIBOT,NFXLFT,
     2NFXRHT,ADJDIS,ADJTT,SMFPLL
  173 FORMAT('  6   ITRMAX (ITERATION LIMIT) = ',I6,/,     ' 7   TOLABS,
     1TOLINC (ABSOLUTE,INCREMENTAL ERROR TOL FOR CONV) = ',F7.4,F8.4,
     2/,'  8   RELAX (RELAXATION PARAMETER) = ',F9.5,/,'  9   IPDAT,IPIV
     3G,IPFRC,IPERR,IPSEG,IPGRA,ISGRA (1 OUTPUT: DATA,INITIAL',/,'
     4 VEL,PATH FRAC,ERRORS,SEG LENGTHS,VEL GRAPH,SCRN VEL GRAPH)= ',I1,
     56I2,/,' 10   NPIXH,NPIXV (NUMBER OF COLUMNS,ROWS OF PIXELS) = ',2I
     64,/,'          (PIXEL WIDTH,HEIGHT =',2F8.3,')',/,
     7    ' 11   NLITOP,NLIBOT (NUM OF LATERALLY INVARIANT ROWS AT TOP,B
     8OTTOM) =',2I3,/,' 12   NFXLFT,NFXRHT (NUM OF COL WITH FIXED VEL AT
     8 LEFT,RIGHT SIDES) =',2I3,/,' 13   ADJDIS,ADJTT (DISTANCE, TIME AD
     9DED) = ',2F10.4,/,' 14   SMFPLL (LOWER LIMIT OF PATH FRACTIONS FOR
     9 DISPLAY) = ',F8.4)
      WRITE(*,174) VLOCAL,VHICAL
  174 FORMAT(' 15   VLOCAL,VHICAL (LIMITS OF CALC VELOCITY,UNLESS = -99.
     1) =',2F9.3)
      WRITE(*,175) VLODSP,VHIDSP
  175 FORMAT(' 16   VLODSP,VHIDSP (VELOCITY LIMITS IN GRAPHS AND SCALED
     1DISPLAYS',/,'       UNLESS = -99., THEN LIMITS ARE MIN,MAX VELOCIT
     1IES) =',2F9.3)
      WRITE(*,176) VLODIF,VHIDIF
  176 FORMAT(' 17   VLODFP,VHIDFP (VELOCITY DIFFERENCE LIMITS IN GRAPHS)
     1 =',2F9.3)
      WRITE(*,177) INVOPT
  177 FORMAT(' 18   INVOPT (OPTION FOR INPUT OF INITIAL VELOCITY GUESS:
     1 1 FOR',/,'             UNIFORM, 2 FOR ROW BY ROW, 3 FOR PIXEL BY
     2PIXEL) = ',I2)
      WRITE(*,*)('ENTER OPTION NUMBER. ')

      READ(*,*)IFLG
      GO TO(280,180,185,190,195,200,205,210,215,220,225,228,230,235,
     1 238,240,243,245) IFLG
      WRITE(*,*) (' INCORRECT OPTION NUMBER.  REENTER OPTION, 1 TO 18')
      GO TO 170
  180 IF(MSTOR.EQ.1) MTEMP=0
```

```
      IF(MSTOR.NE.1) MTEMP=1
      MSTOR=MTEMP
      GO TO 170
  185 IF(MBORE.EQ.1) MTEMP=0
      IF(MBORE.NE.1) MTEMP=1
      MBORE=MTEMP
      GO TO 170
  190 IF(MSMTH.EQ.1) MTEMP=0
      IF(MSMTH.NE.1) MTEMP=1
      MSMTH=MTEMP
      GO TO 170
  195 IF(MDIF.EQ.1) MTEMP=0
      IF(MDIF.NE.1) MTEMP=1
      MDIF=MTEMP
      GO TO 170
  200 WRITE(*,202) ITRMAX
  202 FORMAT(' CURRENT ITRMAX IS',I5,/,'  ENTER NEW ITRMAX  ')
      READ(*,*) ITRMAX
      GO TO 170
  205 WRITE(*,207) TOLABS,TOLINC
  207 FORMAT(' CURRENT TOLABS,TOLINC ARE',2F8.4,/,
     1'   ENTER NEW TOLABS,TOLINC ')
      READ(*,*) TOLABS,TOLINC
      GO TO 170
  210 WRITE(*,212) RELAX
  212 FORMAT(' CURRENT RELAX IS',F10.6,/,'  ENTER NEW RELAX ')
      READ(*,*) RELAX
      GO TO 170
  215 WRITE(*,217) IPDAT,IPIVG,IPFRC,IPERR,IPSEG,IPGRA,ISGRA
  217 FORMAT(' CURRENT IPDAT,IPIVG,IPFRC,IPERR,IPSEG,IPGRA,ISGRA ARE',
     1 7I2,/,'   ENTER NEW IPDAT,IPIVG,IPFRC,IPERR,IPSEG,IPGRA,ISGRA ')
      READ(*,*)IPDAT,IPIVG,IPFRC,IPERR,IPSEG,IPGRA,ISGRA
      GO TO 170
  220 WRITE(*,222) NPIXH,NPIXV
  222 FORMAT(' CURRENT NPIXH,NPIXV ARE',2I4,/,
     1 '   ENTER NEW NPIXH,NPIXV ')
      READ(*,*) NPIXH,NPIXV
      IF (INVOPT.EQ.1) GO TO 255
      WRITE(*,*)('IF NPIXH,NPIXV ARE CHANGED AND INVOPT=2 OR 3, BE SURE
     1INITIAL VEL GUESSES ARE   FOR CURRENT PIXEL GRID SIZE')
      GO TO 245
  225 WRITE(*,227) NLITOP,NLIBOT
  227 FORMAT(' CURRENT NLITOP,NLIBOT ARE',2I3,/,
     1 '   ENTER NEW NLITOP,NLIBOT ')
      READ(*,*) NLITOP,NLIBOT
      GO TO 170
  228 WRITE(*,229) NFXLFT,NFXRHT
  229 FORMAT(' CURRENT NFXLFT,NFXRHT ARE',2I3,/,
     1 '   ENTER NEW NFXLFT,NFXRHT ')
      READ(*,*) NFXLFT,NFXRHT
      GO TO 170
  230 WRITE(*,232) ADJDIS,ADJTT
```

```
232 FORMAT(' CURRENT ADJDIS,ADJTT ARE',2F9.4,/,
   1'   ENTER NEW ADJDIS,ADJTT ')
    READ(*,*) ADJDIS,ADJTT
    GO TO 170
235 WRITE(*,237) SMFPLL
237 FORMAT(' CURRENT SMFPLL IS',F7.4,/,'  ENTER NEW SMFPLL ')
    READ(*,*) SMFPLL
    GO TO 170
238 WRITE(*,239) VLOCAL,VHICAL
239 FORMAT(' CURRENT VLOCAL,VHICAL ARE',2F10.4,/,
   1 '   ENTER NEW VLOCAL,VHICAL  ')
    READ(*,*) VLOCAL,VHICAL
    GO TO 170
240 WRITE(*,242) VLODSP,VHIDSP
242 FORMAT(' CURRENT VLODSP,VHIDSP ARE',2F10.4,/,
   1 '   ENTER NEW VLODSP,VHIDSP  ')
    READ(*,*) VLODSP,VHIDSP
    GO TO 170
243 WRITE(*,244) VLODFP,VHIDFP
244 FORMAT(' CURRENT VLODIF,VHIDIF ARE',2F10.4,/,
   1 '   ENTER NEW VLODIF,VHIDIF  ')
    READ(*,*) VLODFP,VHIDFP
    GO TO 170
245 WRITE(*,247) INVOPT
247 FORMAT(' CURRENT INVOPT IS ',I3,/,' ENTER NEW INVOPT (1 UNIFORM; 2
   1 ROW BY ROW; OR 3 PIXEL BY PIXEL) ')
    READ(*,*) INVOPT
    GO TO (252,258,264) INVOPT
    WRITE(*,250)
250 FORMAT(' INVOPT NOT SPECIFIED CORRECTLY.  TRY AGAIN.')
    GO TO 245
252 WRITE(*,254) V(1)
254 FORMAT(' CURRENT INITIAL GUESS IS ',F8.4,/,'  ENTER NEW INITIAL GU
   1ESS ')
    READ(*,*) V(1)
255  DO 256  IV=2,NPIXH*NPIXV
256  V(IV) = V(1)
    GO TO 170
258 WRITE(*,*)(' ENTER PIXEL ROW NUMBER. NEG ENTRY ENDS THIS INPUT ')
    READ(*,*) IPIXV
    IF(IPIXV.LT.0) GO TO 170
    WRITE(*,260) V(1+(IPIXV-1)*NPIXH)
260 FORMAT(' CURRENT INITIAL GUESS IS ',F8.4,/,'  ENTER NEW INITIAL GU
   1ESS ')
    READ(*,*) VTR
     DO 262 IPIXH=1,NPIXH
262  V(IPIXH+(IPIXV-1)*NPIXH)=VTR
    GO TO 258
264 WRITE(*,*)(' ENTER PIXEL ROW NUMBER. NEG ENTRY ENDS THIS INPUT ')
     READ(*,*) IPIXV
     IF(IPIXV.LT.0) GO TO 170
266 WRITE(*,*)(' ENTER PIXEL COLUMN NUMBER ')
```

```
      READ(*,*) IPIXH
      WRITE(*,268) V(IPIXH+(IPIXV-1)*NPIXH)
  268 FORMAT(' CURRENT INITIAL GUESS IS ',F8.4,/,'  ENTER NEW INITIAL GU
     1ESS ')
      READ(*,*) V(IPIXH+(IPIXV-1)*NPIXH)
      GO TO 264
C***********END OF MODIFYING INPUT USING KEYBOARD*****************************
C*******CHECK FILE FOR PRIOR RUN IF COMPARISONS WILL BE MADE*******************
  280 IF (MDIF.NE.1 .OR. MDFLG.EQ.-999) GO TO 300
  282 WRITE(*,283)
  283 FORMAT(' ENTER NAME OF FILE OF PRIOR RUN FOR COMPARING VELOCITIES:
     1 ')
      READ(*,284) OLDFIL
  284 FORMAT(A12)
      INQUIRE(FILE=OLDFIL,EXIST=FLXT,ERR=128,IOSTAT=IOS)
      IF(FLXT) GO TO 289
  286 WRITE(*,288)
  288 FORMAT(' FILE FOR PRIOR RUN WAS NOT FOUND.',/,
     1 ' CHECK SPELLING, EXTENSION, AND PATH.  THEN TRY AGAIN.')
      GO TO 282
  289 OPEN(3,FILE=OLDFIL,STATUS='OLD',ERR=286,IOSTAT=IOS)
  290 READ(3,132) HEADF
      WRITE(*,132) HEADF
      READ(3,*) MDADF,MWTDF,MSTRDF,MCRDF,MBRDF,MSMTDF,MDIFDF
      WRITE(*,134) MDADF,MWTDF,MSTRDF,MCRDF,MBRDF,MSMTDF,MDIFDF
      READ(3,*) ITMDF,ITSDF,TLADF,TLIDF,RELDF
      WRITE(*,136) ITMDF,ITSDF,TLADF,TLIDF,RELDF
      CLOSE(3)
      WRITE(*,*)(' IS THIS THE CORRECT FILE FOR PRIOR RUN?  (Y/N): ')
      READ(*,292) MYN
  292 FORMAT(A1)
      IF(MYN.NE.IYES .AND. MYN.NE.JYES) GO TO 282
C***********PRINT INPUT PARAMETERS*********************************************
  300 WRITE(2,302)
  302 FORMAT(1H1)
      WRITE(*,304)
  304 FORMAT(' ENTER ONE LINE OF HEADING FOR OUTPUT',/,1X)
      READ(*,306) HEADER
  306 FORMAT(A79)
      WRITE(2,308) HEADER
  308 FORMAT(1X,A79)
      IF(ITRMAX.EQ.-1) THEN
      WRITE(2,*)('SYNTHETIC TRAVEL TIME GENERATION FROM BOMTOM')
      GO TO 309
      END IF
      IF(MDFLG.NE.-999) WRITE(2,*)('TOMOGRAPHIC RECONSTRUCTION FROM BOMT
     1OM')
      IF(MDFLG.EQ.-999) WRITE(2,*)('PARAMETERS OF PRIOR RUN USED FOR COM
     6PARISON')
  309 WRITE(2,310) INFIL
  310 FORMAT(/,' INPUT FILE IS ',A12)
      WRITE(2,312) OUTFIL
```

```
312 FORMAT(' THIS OUTPUT FILE IS ',A12)
    WRITE(2,313) DATFIL
313 FORMAT(' OUTPUT DATA FILE IS ',A12)
    IF(MDATA.NE.1) WRITE(2,*)('MDATA = 0, POSITIONS AND TIMES ARE INPU
   1T SEPARATELY')
    IF(MDATA.EQ.1) WRITE(2,*)('MDATA = 1, POSITIONS AND TIMES ARE INPU
   1T TOGETHER')
    IF(MWT.NE.1) WRITE(2,*)('MWT = 0, NO WEIGHTING FACTORS FOR ACCURAC
   1Y OF TRAVEL TIMES')
    IF(MWT.EQ.1) WRITE(2,*)('MWT = 1, USE WEIGHTING FACTORS FOR ACCURA
   1CY OF TRAVEL TIMES')
    IF(MSTOR.NE.1)WRITE(2,*)('MSTOR = 0, RECALCULATE PATH SEGMENTS')
    IF(MSTOR.EQ.1)WRITE(2,*)('MSTOR = 1, STORE AND RECALL PATH SEGMENT
   1S')
    IF(MCOR.NE.1)WRITE(2,*)('MCOR = 0, MULTIPLICATIVE CORRECTION FACTO
   1RS')
    IF(MCOR.EQ.1)WRITE(2,*)('MCOR = 1, ADDITIVE CORRECTION FACTORS')
    IF(MBORE.NE.1)WRITE(2,*)('MBORE = 0, VELOCITIES AT TRANS AND REC P
   1OS ARE VARIED UNLESS COL VEL FIXED')
    IF(MBORE.EQ.1)WRITE(2,*)('MBORE = 1, VELOCITIES AT TRANS AND REC P
   1OS ARE FIXED')
    IF(MSMTH.NE.1) WRITE(2,*)('MSMTH = 0, NO SMOOTHING')
    IF(MSMTH.EQ.1) WRITE(2,*)('MSMTH = 1, PERFORM SMOOTHING')
    IF(MDIF.NE.1)WRITE(2,*)('MDIF = 0, WILL NOT BE COMPARED WITH PRIOR
   1 RUN')
    IF(MDIF.EQ.1) WRITE(2,314) OLDFIL
314 FORMAT(' MDIF = 1, WILL BE COMPARED WITH ',A12)
    WRITE(2,316) IPDAT,IPIVG,IPFRC,IPERR,IPSEG,IPGRA,ISGRA
316 FORMAT(' IPDAT,IPIVG,IPFRC,IPERR,IPSEG,IPGRA,ISGRA = ',I2,',',I2,
   1 ',',I2,',',I2,',',I2,',',I2,',',I2,',',I2)
    WRITE(2,320)NEXPTT,POSP1X,POSP1Y,POSP1Z,POSPNX,POSPNY,POSPNZ,
   1 NPIXH,NPIXV,EPIXH,EPIXV,NLITOP,NLIBOT,NFXLFT,NFXRHT,TOLABS,
   2 TOLINC,RELAX
320 FORMAT(/,' NUMBER OF EXPERIMENTAL TRAVEL TIMES = ',I3,/,
   1' CENTER OF PIXEL 1 IS AT X =',F10.4,'  Y =',F10.4,'  Z =',F10.4,
   2/,' CENTER OF LAST PIXEL IS AT X =',F10.4,'  Y =',F10.4,'  Z =',
   3 F10.4,/,' NUMBER OF COLUMNS, ROWS OF PIXELS = ',I3,',',I3,/,
   4' PIXEL WIDTH, HEIGHT = ',F9.4,',',F9.4,/,
   5' NUMBER OF LATERALLY INVARIANT ROWS AT TOP, BOTTOM = ',I2,',',I2,
   6/,' NUMBER OF FIXED VELOCITY COLUMNS AT LEFT, RIGHT = ',I2,',',I2,
   7  /,' ABSOLUTE ERROR TOLERANCE FOR CONVERGENCE = ',F9.6,
   8  /,' INCREMENTAL ERROR TOLERANCE FOR CONVERGENCE = ',F9.6,
   9       /,' RELAXATION PARAMETER =',F9.5)
    WRITE(2,322) ADJDIS,ADJTT
322 FORMAT(' DISTANCE ADJUSTMENT ADDED BETWEEN BOREHOLES = ',F10.4,/,
   1 ' TIME ADJUSTMENT ADDED TO ALL TRAVEL TIMES = ',F10.4)
    IF(VLOCAL.EQ.-99. .AND. VHICAL.EQ.-99.) WRITE(2,*)
   6 ('NO LIMITS ON CALCULATED VELOCITIES.')
    IF(VLOCAL.NE.-99.) WRITE(2,323) VLOCAL
323 FORMAT(' LOWEST ALLOWED VELOCITY =',F8.3)
    IF(VHICAL.NE.-99.) WRITE(2,324) VHICAL
324 FORMAT(' HIGHEST ALLOWED VELOCITY =',F8.3)
```

```
    IF(ITRMAX.EQ.-1)GO TO 400
328 IF(IPDAT.NE.1)GO TO 365
     IF(MDATA.NE.1) THEN
     WRITE(2,330) NPST,NPSR
330  FORMAT(//,' NUMBER OF TRANSMITTER, RECEIVER POSITIONS =',I4,',',
   1 I4)
     WRITE(2,*)('POSITIONS LISTED HAVE NOT BEEN ADJUSTED BY ADJDIS')
     WRITE(2,332)
332  FORMAT(/,' TRANS IDENT        X              Y              Z')
     DO 334 IPST=1,NPST
334  WRITE(2,336) IDPST(IPST),POSTX(IPST),POSTY(IPST),POSTZ(IPST)
336  FORMAT(I10,3F15.4)
     WRITE(2,338)
338  FORMAT(/,'   REC IDENT        X              Y              Z')
     DO 340 IPSR=1,NPSR
340  WRITE(2,336) IDPSR(IPSR),POSRX(IPSR),POSRY(IPSR),POSRZ(IPSR)
     WRITE(2,342) NEXPTT
342  FORMAT(//,' NUMBER OF TRAVEL TIMES =',I6)
     WRITE(2,344)
344  FORMAT(' TRAVEL TIMES LISTED HAVE NOT BEEN ADJUSTED BY ADJTT',/)
      IF(MWT.NE.1) THEN
      WRITE(2,*)('TRANS ID   REC ID    TRAVEL TIME        TRANS ID   R
   1EC ID    TRAVEL TIME')
      NEXP2=NEXPTT/2
      DO 346 IEXP2=1,NEXP2
      ISUM2=IEXP2+NEXP2
346   WRITE(2,348) IDEXTT(IEXP2),IDEXTR(IEXP2),EXPTT(IEXP2),
   1   IDEXTT(ISUM2),IDEXTR(ISUM2),EXPTT(ISUM2)
348   FORMAT(1X,I7,I9,F15.5,9X,I7,I9,F15.5)
      IF(ISUM2.LT.NEXPTT)WRITE(2,349) IDEXTT(NEXPTT),IDEXTR(NEXPTT),
   1   EXPTT(NEXPTT)
349   FORMAT(41X,I7,I9,F15.5)
      ELSE
      WRITE(2,*)('TRANS IDENT   REC IDENT      TRAVEL TIME  WEIGHT')
      DO 350 IEXPTT=1,NEXPTT
350   WRITE(2,352) IDEXTT(IEXPTT),IDEXTR(IEXPTT),EXPTT(IEXPTT),
   1  WEIGHT(IEXPTT)
352   FORMAT(1X,I9,I13,F18.5,F7.3)
      END IF
     ELSE
     WRITE(2,354)
354  FORMAT(//,' POSITIONS AND TRAVEL TIMES LISTED HAVE NOT BEEN ADJUS
   1TED BY ADJDIS AND ADJTT')
      IF(MWT.NE.1) THEN
     WRITE(2,356)
356  FORMAT(/,'PATH ID  TRANS X   TRANS Y   TRANS Z    REC X    REC Y
   1 REC Z   TRAVEL TIME')
     DO 358 IEXPTT=1,NEXPTT
358  WRITE(2,360)IDPATH(IEXPTT),POSTX(IEXPTT),POSTY(IEXPTT),
   1 POSTZ(IEXPTT),POSRX(IEXPTT),POSRY(IEXPTT),POSRZ(IEXPTT),
   2 EXPTT(IEXPTT)
360  FORMAT(1X,I4,3F10.4,3F10.4,F9.5)
```

```
      ELSE
      WRITE(2,361)
361   FORMAT(/,' PATH   TRANS X   TRANS Y   TRANS Z    REC X     REC Y
     1   REC Z  TRAV TIME  WT')
      DO 362 IEXPTT=1,NEXPTT
362   WRITE(2,363)IDPATH(IEXPTT),POSTX(IEXPTT),POSTY(IEXPTT),
     1 POSTZ(IEXPTT),POSRX(IEXPTT),POSRY(IEXPTT),POSRZ(IEXPTT),
     2 EXPTT(IEXPTT),WEIGHT(IEXPTT)
363   FORMAT(1X,I4,3F10.4,3F10.4,F9.5,F5.2)
      END IF
      END IF
365   IF(IPIVG.NE.1) GO TO 369
      WRITE(2,366)
366   FORMAT(/,' INITIAL GUESSES FOR VELOCITIES FOR EACH PIXEL')
      DO 367 IPIXV=1,NPIXV
367   WRITE(2,368) (V(IPIXH+(IPIXV-1)*NPIXH),IPIXH=1,NPIXH)
368   FORMAT(1X,61F5.2)
369   IF(IPGRA.NE.1) GO TO 395
      IF(VLODSP.EQ.-99.) THEN
        VMIN=10000.
        DO 370 IPIXEL=1,NPXTOT
        VMIN=AMIN1(VMIN,V(IPIXEL))
370     CONTINUE
      ELSE
        VMIN=VLODSP
      END IF
      IF(VHIDSP.EQ.-99.) THEN
        VMAX=-10000.
        DO 371 IPIXEL=1,NPXTOT
        VMAX=AMAX1(VMAX,V(IPIXEL))
371     CONTINUE
      ELSE
        VMAX=VHIDSP
      END IF
      VDIFF=VMAX-VMIN
      DIV1=VMIN+0.2*VDIFF
      DIV2=VMIN+0.4*VDIFF
      DIV3=VMIN+0.6*VDIFF
      DIV4=VMIN+0.8*VDIFF
      WRITE(2,981)
      WRITE(2,*)('  GRAPH OF INITIAL VELOCITY GUESSES')
      WRITE(2,*)(' VELOCITY LIMITS FOR A PATTERN ARE ON EACH SIDE OF IT.
     1')
      WRITE(2,982) VMIN,CHAR(32),CHAR(32),DIV1,CHAR(176),CHAR(176),
     1 DIV2,CHAR(177),CHAR(177),DIV3,CHAR(178),CHAR(178),DIV4,CHAR(219),
     2 CHAR(219),VMAX
      DO 390 IPIXV=1,NPIXV
      DO 384 IPIXH=1,NPIXH
      IPIXEL=IPIXH+(IPIXV-1)*NPIXH
        IF(V(IPIXEL).LT.VMIN) IASCI=60
        IF(V(IPIXEL).GE.VMIN .AND. V(IPIXEL).LT.DIV1) IASCI=32
        IF(IASCI.EQ.32 .AND. IPIXV.EQ.1 .AND. IPIXH.EQ.2) IASCI=196
```

```
        IF(V(IPIXEL).GE.DIV1 .AND. V(IPIXEL).LT.DIV2) IASCI=176
        IF(V(IPIXEL).GE.DIV2 .AND. V(IPIXEL).LE.DIV3) IASCI=177
        IF(V(IPIXEL).GT.DIV3 .AND. V(IPIXEL).LE.DIV4) IASCI=178
        IF(V(IPIXEL).GT.DIV4 .AND. V(IPIXEL).LE.VMAX) IASCI=219
        IF(V(IPIXEL).GT.VMAX) IASCI=62
         IF(IPIXV.EQ.1 .AND. IASCI.EQ.32) IASCI=196
         IF(IPIXV.EQ.NPIXV .AND. IASCI.EQ.32) IASCI=95
        WRITE(VSCALE(IPIXH),985) CHAR(IASCI)
  384 CONTINUE
      IF(NPIXH.LT.40) WRITE(2,987) (VSCALE(IPIXH),VSCALE(IPIXH),
     1 IPIXH=1,NPIXH)
      IF(NPIXH.GE.40) WRITE(2,987) (VSCALE(IPIXH),IPIXH=1,NPIXH)
  390 CONTINUE
  395 IF(ITRFLG.EQ.-100) GO TO 930
C***********END OF PRINTING INPUT PARAMETERS**********************************
C***********BEGIN CALCULATION OF NEW VELOCITIES*******************************
  400 IF(MDFLG.EQ.-999) GO TO 992
      AVERO=10000.
      DELERO=10000.
      ITRPRN=0
      ITRNEW=0
      IFLGPT=0
      ISFLG=0
      IF(ISGRA.EQ.1) ISFLG = -99
      DO 420 IPIXEL=1,NPXTOT
      LGLCOR(IPIXEL)=.TRUE.
      WTSMF(IPIXEL)=0.0
  420 SMFPIX(IPIXEL)=0.0
C*******FIX VELOCITY IN DESIGNATED NUMBER OF COLUMNS*****
      DO 422 IFXLFT=1,NFXLFT
      DO 422 IPIXV=1,NPIXV
      IPIXEL = IFXLFT + (IPIXV-1)*NPIXH
  422 LGLCOR(IPIXEL) = .FALSE.
      DO 424 IFXRHT=1,NFXRHT
      DO 424 IPIXV=1,NPIXV
      IPIXEL = NPIXH-(IFXRHT-1) + (IPIXV-1)*NPIXH
  424 LGLCOR(IPIXEL) = .FALSE.
C*************************************************************
  430 IARSG=0
      NTOTSG=0
      DO 435 I=1,NPXTOT
  435 COR(I)=0.
C*******START MAIN DO LOOP FOR CALCULATION OF PATH SEGMENTS*****
      DO 640 IEXPTT=1,NEXPTT
      CAL = 0.0
       IF(MSTOR.EQ.1 .AND. ITRNEW.GT.0) GO TO 610
       IF(MDATA.NE.1) THEN
       IPST=IDEXTT(IEXPTT)
       IF(IDPST(IPST).EQ.IDEXTT(IEXPTT)) GO TO 448
       DO 440 IPST=1,NPST
  440  IF(IDPST(IPST).EQ.IDEXTT(IEXPTT)) GO TO 448
       WRITE(*,445) IDEXTT(IEXPTT),IEXPTT
```

```
      WRITE(2,445) IDEXTT(IEXPTT),IEXPTT
  445 FORMAT(' DID NOT FIND TRANSMITTER POSITION NUMBER',I4,
     1 ' FOR TRAVEL TIME',I6)
      GO TO 999
  448 IPSR=IDEXTR(IEXPTT)
      IF(IDPSR(IPSR).EQ.IDEXTR(IEXPTT)) GO TO 460
  449 DO 450 IPSR=1,NPSR
  450 IF(IDPSR(IPSR).EQ.IDEXTR(IEXPTT)) GO TO 460
      WRITE(*,455) IDEXTR(IEXPTT),IEXPTT
      WRITE(2,455) IDEXTR(IEXPTT),IEXPTT
  455 FORMAT(' DID NOT FIND RECEIVER POSITION NUMBER',I4,
     1 ' FOR TRAVEL TIME',I6)
      GO TO 999
  460 POSTXA=POSTX(IPST)
      POSTYA=POSTY(IPST)
      POSTZA=POSTZ(IPST)
      POSRXA=POSRX(IPSR)
      POSRYA=POSRY(IPSR)
      POSRZA=POSRZ(IPSR)
      ELSE
      POSTXA=POSTX(IEXPTT)
      POSTYA=POSTY(IEXPTT)
      POSTZA=POSTZ(IEXPTT)
      POSRXA=POSRX(IEXPTT)
      POSRYA=POSRY(IEXPTT)
      POSRZA=POSRZ(IEXPTT)
      END IF
       IF(ADJDIS.NE.0.) THEN
       DFACX=POSRXA-POSTXA
       DFACY=POSRYA-POSTYA
       DPTHH=SQRT(DFACX*DFACX + DFACY*DFACY)
       POSRXA=POSRXA+DFACX*ADJDIS/DPTHH
       POSRYA=POSRYA+DFACY*ADJDIS/DPTHH
       END IF
      DFACX=POSTXA-POSP1X
      DFACY=POSTYA-POSP1Y
      HYP1S=DFACX*DFACX + DFACY*DFACY
      DFACX=POSTXA-POSPNX
      DFACY=POSTYA-POSPNY
      HYPNS=DFACX*DFACX + DFACY*DFACY
      POSTH=.5*(HYP1S-HYPNS+DPIXHS)/DPIXH
      POSTV=POSTZA-POSP1Z
      DFACX=POSRXA-POSP1X
      DFACY=POSRYA-POSP1Y
      HYP1S=DFACX*DFACX + DFACY*DFACY
      DFACX=POSRXA-POSPNX
      DFACY=POSRYA-POSPNY
      HYPNS=DFACX*DFACX + DFACY*DFACY
      OFFNH=.5*(HYPNS-HYP1S+DPIXHS)/DPIXH
      POSRH=DPIXH-OFFNH
      POSRV=POSRZA-POSP1Z
```

```
      IROWT=NINT(POSTV/EPIXV)+1
      IROWR=NINT(POSRV/EPIXV)+1
      ICOLT=NINT((POSTH+.00001)/EPIXH)+1
      ICOLR=NINT((POSRH-.00001)/EPIXH)+1
       IF(IROWT.LT.1 .OR. IROWR.LT.1) THEN
       WRITE(*,466) IEXPTT
       WRITE(2,466) IEXPTT
  466  FORMAT(/,' IROWT OR IROWR IS LESS THAN 1 FOR TRAVEL TIME NUMBER '
     1 ,I5,/,' MOVE PIXEL 1 (TOP LEFT) UPWARD.  EXECUTION TERMINATED.')
       END IF
       IF(IROWT.GT.NPIXV .OR. IROWR.GT.NPIXV) THEN
       WRITE(*,468) IEXPTT
       WRITE(2,468) IEXPTT
  468  FORMAT(/,' IROWT OR IROWR IS GREATER THAN NPIXV FOR TRAVEL TIME N
     1UMBER ',I5,/,' MOVE THE LAST PIXEL (BOTTOM RIGHT) DOWNWARD.  EXECU
     1TION TERMINATED.')
       END IF
       IF(ICOLT.LT.1) THEN
       WRITE(*,470) ICOLT,IEXPTT
       WRITE(2,470) ICOLT,IEXPTT
  470  FORMAT(/,' ICOLT = ',I6,' FOR TRAVEL TIME NUMBER ',
     1 I5,/,' MOVE PIXEL 1 (TOP LEFT) TO LEFT.  EXECUTION TERMINATED.')
       GO TO 999
       END IF
       IF(ICOLR.GT.NPIXH) THEN
       WRITE(*,472) ICOLR,IEXPTT
       WRITE(2,472) ICOLR,IEXPTT
  472  FORMAT(/,' ICOLR = ',I6,' ,WHICH IS GREATER THAN NPIXH, FOR TRAVE
     1L TIME NUMBER ',I5,/,' MOVE THE LAST PIXEL (BOTTOM RIGHT) TO RIGHT
     2.  EXECUTION TERMINATED.')
       GO TO 999
       END IF
      NSCROS=IROWR-IROWT
      NACROS=IABS(NSCROS)
      IA=0
      IF(NSCROS.GT.0)IA=1
      IF(NSCROS.LT.0)IA=-1

      DFACH=POSRH-POSTH
      DFACV=POSRV-POSTV
      DPATH=SQRT(DFACH*DFACH + DFACV*DFACV)
      DSEC=DPATH/DFACH
      DFACX=POSRXA-POSTXA
      DFACY=POSRYA-POSTYA
      DFACZ=POSRZA-POSTZA
      DPTHTL=SQRT(DFACX*DFACX + DFACY*DFACY + DFACZ*DFACZ)
      EXTADJ(IEXPTT)= (EXPTT(IEXPTT)+ADJTT)*DPATH/DPTHTL
      IF(MWT.NE.1) WTFAC=1./DPATH
      IF(MWT.EQ.1) WTFAC=WEIGHT(IEXPTT)/DPATH
  480 SLOPE=ABS(DFACV/DFACH)
      DO 488 ICROS=1,NACROS
      TCROS(ICROS)=(.5*EPIXV-POSTV*IA+
```

```
     1(IROWT-1)*EPIXV*IA+(ICROS-1)*EPIXV)/SLOPE
      E1CROS=TCROS(ICROS)-(ICOLT-.5)*EPIXH+POSTH
      IF(E1CROS.GT.0)GO TO 485
      ICOL(ICROS)=ICOLT
      IF(ICROS.EQ.1) EDCROS(ICROS)=TCROS(ICROS)-POSTH+(ICOLT-1)*EPIXH
      IF(ICROS.GT.1)EDCROS(ICROS)=TCROS(ICROS)-TCROS(ICROS-1)
      GO TO 488
  485 ICOL(ICROS)=INT(E1CROS/EPIXH)+1+ICOLT
      EDCROS(ICROS)=E1CROS-(ICOL(ICROS)-1-ICOLT)*EPIXH
  488  CONTINUE
  490 ICOL(NACROS+1)=ICOLR
C
C*****START CALCULATIONS FOR COLUMN ICOLT*****
  500  ISG=1
       JPIXH=ICOLT
       ICROS=1
      IF (ICOL(ICROS).EQ.JPIXH) THEN
        D(1)=TCROS(ICROS)*DSEC
  505   IF(ICROS.EQ.NACROS) GO TO 510
          IF(ICOL(ICROS+1).NE.ICOL(ICROS)) GO TO 510
            ISG=ISG+1
            D(ISG)=(TCROS(ICROS+1)-TCROS(ICROS))*DSEC
            ICROS=ICROS+1
          GO TO 505
  510   ISG=ISG+1
        D(ISG)=((ICOLT-.5)*EPIXH-POSTH-TCROS(ICROS))*DSEC
        ICROS=ICROS+1
      ELSE
        D(1)=((ICOLT-.5)*EPIXH-POSTH)*DSEC
      END IF
      ISGT=ISG
      DO 520 JSG=1,ISGT
      IPIXEL=ICOLT+(IROWT-1)*NPIXH+IA*(JSG-1)*NPIXH
      CAL=CAL+D(JSG)/V(IPIXEL)
      IF(MBORE.EQ.1) LGLCOR(IPIXEL)=.FALSE.
  520 IPIXSG(JSG)=IPIXEL
C*****END OF CALCULATIONS FOR COLUMN ICOLT*****
C*****START CALCULATIONS FOR COLUMN ICOLT+1 THROUGH ICOLR-1*****
      JPIXH=ICOLT+1
C COLUMNS WHERE PATH DOES NOT CROSS PIXEL ROW BOUNDARY.
  525   DO 530 JJPIXH=JPIXH,ICOL(ICROS)-1
          ISG=ISG+1
          D(ISG)=EPIXH*DSEC
          IPIXEL=JJPIXH+(IROWT-1)*NPIXH+(ICROS-1)*IA*NPIXH
          CAL=CAL+D(ISG)/V(IPIXEL)
  530     IPIXSG(ISG)=IPIXEL
        JPIXH=ICOL(ICROS)
        IF(JPIXH.EQ.ICOLR) GO TO 550
C  PIXEL ABOVE POINT WHERE PATH CROSSES PIXEL ROW BOUNDARY.
  535 ISG=ISG+1
      D(ISG)=EDCROS(ICROS)*DSEC
      IPIXEL=ICOL(ICROS)+(IROWT-1)*NPIXH+(ICROS-1)*IA*NPIXH
```

```
      CAL=CAL+D(ISG)/V(IPIXEL)
      IPIXSG(ISG)=IPIXEL
      IF (ICROS.EQ.NACROS) GO TO 545
  540   IF (ICOL(ICROS+1).NE.ICOL(ICROS)) GO TO 545
          ISG=ISG+1
          D(ISG)=(EDCROS(ICROS+1)-EDCROS(ICROS))*DSEC
          IPIXEL=ICOL(ICROS)+(IROWT-1)*NPIXH+ICROS*IA*NPIXH
          CAL=CAL+D(ISG)/V(IPIXEL)
          IPIXSG(ISG)=IPIXEL
          ICROS=ICROS+1
        GO TO 540
C  PIXEL BELOW POINT WHERE PATH CROSSES PIXEL ROW BOUNDARY.
  545 ISG=ISG+1
      D(ISG)=(EPIXH-EDCROS(ICROS))*DSEC
      IPIXEL=ICOL(ICROS)+(IROWT-1)*NPIXH+ICROS*IA*NPIXH
      CAL=CAL+D(ISG)/V(IPIXEL)
      IPIXSG(ISG)=IPIXEL
        JPIXH=ICOL(ICROS)+1
        ICROS=ICROS+1
        GO TO 525

C*****START CALCULATIONS FOR COLUMN ICOLR*****
  550 ISGR=ISG+1
      IF(ICROS.GT.NACROS) GO TO 565
        ISG=ISG+1
        D(ISG)=EDCROS(ICROS)*DSEC
  555     IF (ICROS.GE.NACROS) GO TO 560
           ISG=ISG+1
           D(ISG)=(EDCROS(ICROS+1)-EDCROS(ICROS))*DSEC
           ICROS=ICROS+1
          GO TO 555
  560   ISG=ISG+1
       D(ISG)=(POSRH-DPIXH+(NPIXH-ICOLR+.5)*EPIXH-EDCROS(NACROS))
     1 *DSEC
       GO TO 567
  565   ISG=ISG+1
        D(ISG)=(POSRH-DPIXH+(NPIXH-ICOLR+.5)*EPIXH)*DSEC

  567 NSG=ISG
      DO 570 JSG=ISGR,NSG
      IPIXEL=ICOLR+(IROWR-1)*NPIXH-IA*(ISG-JSG)*NPIXH
      CAL=CAL+D(JSG)/V(IPIXEL)
      IF(MBORE.EQ.1) LGLCOR(IPIXEL)=.FALSE.
  570 IPIXSG(JSG)=IPIXEL
C*****END CALCULATIONS FOR COLUMN ICOLR*****
      NTOTSG=NTOTSG+NSG
C*****PRINT PATH SEGMENT LENGTHS IF IPSEG = 1 ******
      CONTINUE
       IF (IPSEG.EQ.1 .AND. ITRNEW.EQ.0) THEN
        IFLGPT=1
        IF(MDATA.NE.1)THEN
        WRITE(2,580)
```

```
  580   FORMAT(' TRANS IDENT, ROW, COL    REC IDENT, ROW, COL   PIXEL
     1   PATH SEG LENGTH')
         DO 583 JD=1,NSG
         JJ=IPIXSG(JD)
  583    WRITE(2,585) IDEXTT(IEXPTT),IROWT,ICOLT,IDEXTR(IEXPTT),IROWR,
     1 ICOLR,JJ,D(JD)
  585    FORMAT(1X,I9,I7,I5,I11,I7,I5,I7,7X,G11.5)
         ELSE
         WRITE(2,590)
  590    FORMAT(' PATH NUM  TRANS ROW,COL  REC ROW,COL   PIXEL  PATH SEG
     1 LENGTH')
         DO 593 JD=1,NSG
         JJ=IPIXSG(JD)
  593    WRITE(2,595) IDPATH(IEXPTT),IROWT,ICOLT,IROWR,ICOLR,JJ,D(JD)
  595 FORMAT(1X,I5,I11,I5,I11,I5,I5,6X,G11.5)
       END IF
      END IF
C****************************************
       IF(ITRMAX.EQ.-1) THEN
       EXPTT(IEXPTT)=CAL
       GO TO 636
       END IF
C*****CALCULATE SUMS OF CORRECTION FACTORS FOR EACH PIXEL*******
      IF(MSTOR.NE.1 .OR. ITRNEW.NE.0) GO TO 627
      DO 605 JD=1,NSG
      IARSG=IARSG+1
      DP(IARSG)=D(JD)
  605 IDPIX(IARSG)=IPIXSG(JD)
      IPP(IEXPTT)=NSG
      DPPATH(IEXPTT)=DPATH
      GO TO 627
  610 CONTINUE
      IARSGS=IARSG
      DO 615 JD=1,IPP(IEXPTT)
      IARSG=IARSG+1
      IDP=IDPIX(IARSG)
  615 CAL=CAL+DP(IARSG)/V(IDP)
      DELT(IEXPTT)=EXTADJ(IEXPTT)-CAL
      RATIO(IEXPTT)=EXTADJ(IEXPTT)/CAL
      IARSG=IARSGS
      DO 620 JD=1,IPP(IEXPTT)
      IARSG=IARSG+1
      IF(MWT.NE.1) WTFAC=1./DPPATH(IEXPTT)
      IF(MWT.EQ.1 )WTFAC=WEIGHT(IEXPTT)/DPPATH(IEXPTT)
      IDP=IDPIX(IARSG)
      IF(MCOR.NE.1) COR(IDP)=COR(IDP)+DP(IARSG)*RATIO(IEXPTT)*
     1 WTFAC
  620 IF(MCOR.EQ.1) COR(IDP)=COR(IDP)+DP(IARSG)*DELT(IEXPTT)*
     1 WTFAC/DPPATH(IEXPTT)
      GO TO 640
  627 DELT(IEXPTT)=EXTADJ(IEXPTT)-CAL
      RATIO(IEXPTT)=EXTADJ(IEXPTT)/CAL
```

```
      DO 630 JV=1,NSG
      JJ=IPIXSG(JV)
      IF(MCOR.NE.1) COR(JJ)=COR(JJ)+D(JV)*RATIO(IEXPTT)*WTFAC
      IF(MCOR.EQ.1) COR(JJ)=COR(JJ)+D(JV)*DELT(IEXPTT)*WTFAC/
     1 DPATH
      IF(ISG.GT.IPMAX) IPMAX=NSG
  630 CONTINUE
  635 IF(ITRNEW.GT.0) GO TO 640
  636 IF(NSG.GT.IPMAX) IPMAX=NSG
      DO 637 JV=1,NSG
      JJ=IPIXSG(JV)
      IF(MWT.EQ.1) WTSMF(JJ)=WTSMF(JJ)+D(JV)*WEIGHT(IEXPTT)/DPATH
  637 SMFPIX(JJ)=SMFPIX(JJ)+D(JV)/DPATH
  640 CONTINUE
C*****END OF CALCULATION OF SUMS OF CORRECTION FACTORS FOR PIXELS*************
      IF(ITRMAX.EQ.-1) THEN
      IFLGPT = 1
      GO TO 800
      END IF
C***********CHECK FOR SUFFICIENT CONVERGENCE.***********************************
  650 IFLGPT = 0
      AVERN=0.
      DO 660 IJ=1,NEXPTT
  660 AVERN=AVERN+ABS(DELT(IJ))
      AVERN=AVERN/NEXPTT
      IF (AVERN.LT.TOLABS .OR. ITRSAV.GE.ITRMAX) IFLGPT = 1
      DELERN=AVERO-AVERN
      IF(DELERN.LT.TOLINC .AND. DELERO.LT.TOLINC) IFLGPT = 1
      AVERO=AVERN
      DELERO=DELERN
      IF(IFLGPT.EQ.1 .OR. ITRPRN.EQ.25) GO TO 800
C
C*****MAKE CORRECTIONS TO VELOCITIES*****
  680 DO 690 IPIXEL=1,NPXTOT
      IF(.NOT.LGLCOR(IPIXEL)) GO TO 690
      IF(SMFPIX(IPIXEL).LT..00001) GO TO 690
      IF(MWT.NE.1) COR(IPIXEL)=COR(IPIXEL)*RELAX/SMFPIX(IPIXEL)
      IF(MWT.EQ.1) COR(IPIXEL)=COR(IPIXEL)*RELAX/WTSMF(IPIXEL)
      IF(MCOR.NE.1) V(IPIXEL)=V(IPIXEL)/COR(IPIXEL)
      IF(MCOR.EQ.1) V(IPIXEL)=V(IPIXEL)/(1.+ V(IPIXEL)*COR(IPIXEL))
  690 CONTINUE
      ITRSAV=ITRSAV+1
      ITRPRN=ITRPRN+1
      ITRNEW=ITRNEW+1
C*****AVERAGE OVER DESIGNATED NUMBER OF ROWS AT TOP AND BOTTOM********
      DO 725 ICTOP=1,NLITOP
      VAVTOP=0.
      NPX=0
      DO 710 IPIXH=1,NPIXH
      IPIXEL = IPIXH + (ICTOP-1)*NPIXH
      IF(SMFPIX(IPIXEL).LT..00001) GO TO 710
      NPX=NPX+1
```

```
      VAVTOP=VAVTOP+V(IPIXEL)
  710 CONTINUE
       IF(NPX.GT.0) THEN
       VAVTOP=VAVTOP/NPX
       DO 720 IPIXH=1,NPIXH
       IPIXEL = IPIXH + (ICTOP-1)*NPIXH
  720  IF(LGLCOR(IPIXEL)) V(IPIXEL)=VAVTOP
       END IF
  725 CONTINUE
      DO 745 ICBOT=1,NLIBOT
      VAVBOT=0.
      NPX=0
      DO 735 IPIXH=1,NPIXH
      IPIXEL = IPIXH + (NPIXV-ICBOT)*NPIXH
      IF(SMFPIX(IPIXEL).LT..00001) GO TO 735
      NPX=NPX+1
      VAVBOT=VAVBOT+V(IPIXEL)
  735 CONTINUE
       IF(NPX.GT.0) THEN
       VAVBOT=VAVBOT/NPX
       DO 740 IPIXH=1,NPIXH
       IPIXEL = IPIXH + (NPIXV-ICBOT)*NPIXH
  740  IF(LGLCOR(IPIXEL)) V(IPIXEL)=VAVBOT
       END IF
  745 CONTINUE

      IF(MSMTH.NE.1) GO TO 776
C*********SMOOTHING*****************
  750 DO 770 IPIXH = 1, NPIXH
      DO 770 IPIXV = 1, NPIXV
      IPIXEL = IPIXH + (IPIXV-1)*NPIXH
      IDHMIN = -1
      IDHMAX = 1
      IDVMIN = -1
      IDVMAX = 1
      IF(IPIXH.EQ.1) IDHMIN = 0
      IF(IPIXH.EQ.NPIXH) IDHMAX = 0
      IF(IPIXV.EQ.1) IDVMIN = 0
      IF(IPIXV.EQ.NPIXV) IDVMAX = 0
      VSMTH(IPIXEL) = 0.
      NPXSM = 0
      DO 765 IDH = IDHMIN,IDHMAX
      DO 765 IDV = IDVMIN,IDVMAX
      IPXSM = IPIXEL + IDH + IDV*NPIXH
      MLWT=1
      IF(IDH.EQ.0 .OR. IDV.EQ.0) MLWT=4
      IF(IDH.EQ.0.AND.IDV.EQ.0) THEN
      MLWT=20
      GO TO 764
      END IF
      IF(SMFPIX(IPXSM).LT..00001) GO TO 765
   764 VSMTH(IPIXEL)=VSMTH(IPIXEL) + MLWT*V(IPXSM)
```

```
      NPXSM = NPXSM+MLWT
  765 CONTINUE
  770 VSMTH(IPIXEL) = VSMTH(IPIXEL)/NPXSM
      DO 775 IPIXEL = 1, NPXTOT
  775 IF(LGLCOR(IPIXEL)) V(IPIXEL) = VSMTH(IPIXEL)
C**********CONSTRAINING VELOCITY WITHIN LOWER AND UPPER LIMITS*********
  776 IF(VLOCAL.NE.-99.) THEN
      DO 777 IPIXEL = 1, NPXTOT
  777 IF(V(IPIXEL).LT.VLOCAL) V(IPIXEL)=VLOCAL
      END IF
      IF(VHICAL.NE.-99.) THEN
      DO 778 IPIXEL = 1,NPXTOT
  778 IF(V(IPIXEL).GT.VHICAL) V(IPIXEL)=VHICAL
      END IF
  780 IF(IFLGPT.NE.1) GO TO 430

C********SAVE ITERATION RESULTS IN DATA FILE**********************************
  800 OPEN(4,FILE=DATFIL,ERR=128,IOSTAT=IOS)
      IF(ITRMAX.EQ.-1) THEN
      ITRFLG=-1
      ITRMAX=5
      ITRSAV=0
      END IF
  805 WRITE(4,806) HEADER
  806 FORMAT(1X,A79)
      WRITE(4,808) MDATA,MWT,MSTOR,MCOR,MBORE,MSMTH,MDIF
  808 FORMAT(8I10)
      WRITE(4,810) ITRMAX,ITRSAV,TOLABS,TOLINC,RELAX
  810 FORMAT(I10,I10,1X,F11.7,1X,F11.7,1X,F11.6)
      WRITE(4,812) IPDAT,IPIVG,IPFRC,IPERR,IPSEG,IPGRA,ISGRA
  812 FORMAT(7I10)
      WRITE(4,814) POSP1X,POSP1Y,POSP1Z,POSPNX,POSPNY,POSPNZ
  814 FORMAT(6(1X,F11.4))
      WRITE(4,816) NPIXH,NPIXV,NLITOP,NLIBOT,NFXLFT,NFXRHT
  816 FORMAT(6I10)
      IF(MDATA.NE.1) THEN
      WRITE(4,830) NPST,NPSR,ADJDIS
  830 FORMAT(2I10,F11.5)
      DO 832 IPST=1,NPST
  832 WRITE(4,834) IDPST(IPST),POSTX(IPST),POSTY(IPST),POSTZ(IPST)
  834 FORMAT(I10,3(1X,F11.4))
      DO 836 IPSR=1,NPSR
  836 WRITE(4,834) IDPSR(IPSR),POSRX(IPSR),POSRY(IPSR),POSRZ(IPSR)
      WRITE(4,854) NEXPTT,ADJTT
      IF(MWT.NE.1) THEN
      DO 838 IEXPTT=1,NEXPTT
  838 WRITE(4,852) IDEXTT(IEXPTT),IDEXTR(IEXPTT),EXPTT(IEXPTT)
        ELSE
      DO 850 IEXPTT=1,NEXPTT
  850 WRITE(4,852) IDEXTT(IEXPTT),IDEXTR(IEXPTT),EXPTT(IEXPTT),
     1  WEIGHT(IEXPTT)
  852 FORMAT(2I10,F15.6,F5.2)
```

```
       END IF
      ELSE
      WRITE(4,854) NEXPTT,ADJDIS,ADJTT
  854  FORMAT(I10,2F11.5)
       IF(MWT.NE.1) THEN
      DO 860 IEXPTT=1,NEXPTT
  860  WRITE(4,862)IDPATH(IEXPTT),POSTX(IEXPTT),POSTY(IEXPTT),
     1 POSTZ(IEXPTT),POSRX(IEXPTT),POSRY(IEXPTT),POSRZ(IEXPTT),
     1 EXPTT(IEXPTT)
  862 FORMAT(I8,6(1X,F9.3),F11.6)
       ELSE
      DO 864 IEXPTT=1,NEXPTT
  864  WRITE(4,866)IDPATH(IEXPTT),POSTX(IEXPTT),POSTY(IEXPTT),
     1 POSTZ(IEXPTT),POSRX(IEXPTT),POSRY(IEXPTT),POSRZ(IEXPTT),
     1 EXPTT(IEXPTT),WEIGHT(IEXPTT)
  866 FORMAT(I8,6(1X,F9.3),F11.6,F5.2)
       END IF
      END IF
      WRITE(4,880)SMFPLL,VLOCAL,VHICAL,VLODSP,VHIDSP,VLODIF,VHIDIF
  880 FORMAT(1X,7F10.3,'   3   ')
       DO 882 IPIXV=1,NPIXV
  882  WRITE(4,884)IPIXV,(V(IPIXH+(IPIXV-1)*NPIXH),IPIXH=1,NPIXH)
  884 FORMAT(I5,61F10.6)
       CLOSE(4)
      IF(ICRFLG.EQ.-99) GO TO 120
      IF(ISFLG.EQ.-99) GO TO 957
  885 IF(ITRFLG.EQ.-1) THEN
       WRITE(*,886) DATFIL
  886  FORMAT(' SYNTHETIC DATA SAVED IN ',A12)
       GO TO 900
      END IF
      WRITE(*,888) DATFIL,ITRSAV,AVERN,DELERN
  888 FORMAT(1X,A12,' ITER',I5,'  AVE ERROR=',G10.3,'  CHANGE/ITER IN AV
     1E ERR=',G9.2)
       ITRPRN=0
  890  IF(IFLGPT.NE.1) GO TO 680
C
C***********PRINT RESULTS*************************************************
  900 ISFLG = 0
      IF(IPFRC.EQ.1) THEN
      WRITE(2,902)
  902 FORMAT(/,' SUM OF FRACTIONS OF PATH SEGMENTS IN EACH PIXEL')
      DO 904 IPIXV=1,NPIXV
  904 WRITE(2,906) (SMFPIX(IPIXH+(IPIXV-1)*NPIXH),IPIXH=1,NPIXH)
  906 FORMAT(1X,61F5.2)
      END IF
      IF(IPERR.EQ.1) THEN
      WRITE(2,910)
  910 FORMAT(//,' ADJUSTED EXPERIMENTAL MINUS CALCULATED TRAVEL TIMES')
      IF(MDATA.NE.1) THEN
      WRITE(2,*)(' TRANS IDENT  REC IDENT    EXPERIMENTAL     EXP-CALC')
      DO 912 IEXPTT=1,NEXPTT
```

```
912 WRITE(2,914) IDEXTT(IEXPTT),IDEXTR(IEXPTT),EXTADJ(IEXPTT),
   1 DELT(IEXPTT)
914 FORMAT(I11,I11,F18.5,F13.5)
    ELSE
    WRITE(2,*)(' PATH IDENT    EXPERIMENTAL     EXP-CALC')
    DO 916 IEXPTT=1,NEXPTT
916 WRITE(2,918) IDPATH(IEXPTT),EXTADJ(IEXPTT),DELT(IEXPTT)
918 FORMAT(I10,F18.5,F13.5)
    END IF
    END IF
920 IF(ITRFLG.EQ.-1) THEN
     ITRFLG=-100
     GO TO 328
     END IF
930 WRITE(2,932) IPMAX
932 FORMAT(//,' DIMENSION NEEDED FOR D(ISG) IS ISG =',I4)
    IF(IARSG.EQ.0) IARSG=NTOTSG
    WRITE(2,934) IARSG,NEXPTT
934 FORMAT(' IF MSTOR = 1, DIMENSION NEEDED FOR DP(IARSG), IDPIX(IARSG
   1),',/,' IPP(NEXPTT), AND DPPATH(NEXPTT) IS IARSG =',I6,
   2'   NEXPTT =',I5)
    IF(ITRFLG.EQ.-100) THEN
    WRITE(2,935)
935 FORMAT(/,' VELOCITIES IN EACH PIXEL')
    GO TO 942
    END IF
936 RMS=0.
    DO 937 IEXPTT=1,NEXPTT
937 RMS=RMS+DELT(IEXPTT)*DELT(IEXPTT)
    RMS=SQRT(RMS/NEXPTT)
    IF(ITRNEW.EQ.1)  WRITE(2,938) ITRSAV,AVERN
    IF(ITRNEW.GT.1)  WRITE(2,939) ITRSAV,AVERN,DELERN
938 FORMAT(/,' NUMBER OF ITERATIONS= ',I5,17X,'   AVE ERROR=',G10.3)
939 FORMAT(/,' NUMBER OF ITERATIONS= ',I5,17X,'   AVE ERROR=',G10.3,
   1 /,17X,       '                CHANGE/ITER IN AVE ERROR=',G10.3)
    WRITE(2,940) RMS
940 FORMAT(' SQUARE ROOT OF (SUM OF SQUARES OF ERROR/NUMBER OF PATHS)=
   1',G10.3)
    WRITE(2,941)
941 FORMAT(//,'  CALCULATED VELOCITIES FOR EACH PIXEL')
942 WRITE(2,944) SMFPLL
944 FORMAT(' *** INDICATES SUM OF PATH FRACTIONS LESS THAN OR EQUAL ',
   1 G9.3)
946 DO 956 IPIXV=1,NPIXV
    DO 948 IPIXH=1,NPIXH
    IPIXEL=IPIXH+(IPIXV-1)*NPIXH
     IF(SMFPIX(IPIXEL).GT.SMFPLL) THEN
     WRITE(VALPHA(IPIXH),950) V(IPIXEL)
     ELSE
     WRITE(VALPHA(IPIXH),952)
     END IF
948 CONTINUE
```

```
      WRITE(2,954) (VALPHA(IPIXH),IPIXH=1,NPIXH)
  950 FORMAT(F5.2)
  952 FORMAT('  ###')
  954 FORMAT(1X,61A5)
  956 CONTINUE
  957 IF(VLODSP.EQ.-99.) THEN
        VMIN=10000.
        DO 958 IPIXEL=1,NPXTOT
        IF(SMFPIX(IPIXEL).LE.SMFPLL) GO TO 958
        VMIN=AMIN1(VMIN,V(IPIXEL))
  958   CONTINUE
      ELSE
        VMIN=VLODSP
      END IF
      IF(VHIDSP.EQ.-99.) THEN
        VMAX=-10000.
        DO 960 IPIXEL=1,NPXTOT
        IF(SMFPIX(IPIXEL).LE.SMFPLL) GO TO 960
        VMAX=AMAX1(VMAX,V(IPIXEL))
  960   CONTINUE
      ELSE
        VMAX=VHIDSP
      END IF
      VDIFF=VMAX-VMIN
      DIV1=VMIN+0.2*VDIFF
      DIV2=VMIN+0.4*VDIFF
      DIV3=VMIN+0.6*VDIFF
      DIV4=VMIN+0.8*VDIFF
      IF(ISFLG.EQ.-99) GO TO 983
      IF(MDFLG.NE.-99) WRITE(2,962) VMIN,VMAX
  962 FORMAT(/,'  VELOCITIES SCALED TO RANGE 0 TO 9',/,
     1 ' VELOCITY CORRESPONDING TO 0 = ',G9.3,/,
     2 ' VELOCITY CORRESPONDING TO 9 = ',G9.3)
      IF(MDFLG.EQ.-99) WRITE(2,964) VMIN,VMAX
  964 FORMAT(/,'  VELOCITY DIFFERENCES SCALED TO RANGE 0 TO 9',/,
     1 ' VELOCITY DIFFERENCE CORRESPONDING TO 0 = ',G9.3,/
     2 ' VELOCITY DIFFERENCE CORRESPONDING TO 9 = ',G9.3)
      WRITE(2,966) SMFPLL
  966 FORMAT(' # INDICATES SUM OF PATH FRACTIONS LESS THAN OR EQUAL ',
     1 F7.4,/)
      IF(VDIFF.LT..00001) THEN
       DO 970 IPIXEL=1,NPXTOT
  970   IVSCAL(IPIXEL)=5
      ELSE
       SCALE=9./(VMAX-VMIN)
       DO 972 IPIXEL=1,NPXTOT
  972   IVSCAL(IPIXEL)=NINT((V(IPIXEL)-VMIN)*SCALE)
      END IF
      DO 980 IPIXV=1,NPIXV
      DO 974 IPIXH=1,NPIXH
      IPIXEL=IPIXH+(IPIXV-1)*NPIXH
       IF(SMFPIX(IPIXEL).GT.SMFPLL) THEN
```

```
      WRITE(VSCALE(IPIXH),975) IVSCAL(IPIXEL)
      ELSE
      WRITE(VSCALE(IPIXH),976)
      END IF
974 CONTINUE
    IF(NPIXH.LT.40) WRITE(2,978) (VSCALE(IPIXH),IPIXH=1,NPIXH)
    IF(NPIXH.GE.40) WRITE(2,979) (VSCALE(IPIXH),IPIXH=1,NPIXH)
975 FORMAT(I1)
976 FORMAT('*')
978 FORMAT(1X,39(1X,A1))
979 FORMAT(1X,79A1)
980 CONTINUE
    IF(IPGRA.EQ.1) THEN
    WRITE(2,981)
981 FORMAT(//)
    IF(MDFLG.NE.-99) WRITE(2,*)('  GRAPH OF VELOCITIES')
    IF(MDFLG.EQ.-99) WRITE(2,*)('  GRAPH OF VELOCITY DIFFERENCES')
    WRITE(2,*)(' VELOCITY LIMITS FOR A PATTERN ARE ON EACH SIDE OF IT.
   1')
    END IF
    IF(IPGRA.EQ.1)WRITE(2,982) VMIN,CHAR(32),CHAR(32),DIV1,CHAR(176),
   1 CHAR(176),DIV2,CHAR(177),CHAR(177),DIV3,CHAR(178),CHAR(178),DIV4,
   2 CHAR(219),CHAR(219),VMAX
982 FORMAT(/,'   BELOW <   ',F5.2,1X,2A1,F5.2,1X,2A1,F5.2,1X,2A1,
   1 F5.2,1X,2A1,F5.2,1X,2A1,F5.2,'   > ABOVE',/)
    IF(IPGRA.EQ.1) WRITE(2,966) SMFPLL
983 DO 988 IPIXV=1,NPIXV
    DO 984 IPIXH=1,NPIXH
    IPIXEL=IPIXH+(IPIXV-1)*NPIXH
     IF(SMFPIX(IPIXEL).GT.SMFPLL) THEN
     WRITE(VSCALE(IPIXH),975) IVSCAL(IPIXEL)
      IF(V(IPIXEL).LT.VMIN) IASCI=60
      IF(V(IPIXEL).GE.VMIN .AND. V(IPIXEL).LT.DIV1) IASCI=32
      IF(V(IPIXEL).GE.DIV1 .AND. V(IPIXEL).LT.DIV2) IASCI=176
      IF(V(IPIXEL).GE.DIV2 .AND. V(IPIXEL).LE.DIV3) IASCI=177
      IF(V(IPIXEL).GT.DIV3 .AND. V(IPIXEL).LE.DIV4) IASCI=178
      IF(V(IPIXEL).GT.DIV4 .AND. V(IPIXEL).LE.VMAX) IASCI=219
      IF(V(IPIXEL).GT.VMAX) IASCI=62
       IF(IPIXV.EQ.1 .AND. IASCI.EQ.32) IASCI=196
       IF(IPIXV.EQ.NPIXV .AND. IASCI.EQ.32) IASCI=95
      WRITE(VSCALE(IPIXH),985) CHAR(IASCI)
     ELSE
     WRITE(VSCALE(IPIXH),976)
     END IF
984 CONTINUE
    IF(IPGRA.EQ.1 .AND. ISFLG.NE.-99) THEN
    IF(NPIXH.LT.40) WRITE(2,987) (VSCALE(IPIXH),VSCALE(IPIXH),
   1 IPIXH=1,NPIXH)
    IF(NPIXH.GE.40) WRITE(2,987) (VSCALE(IPIXH),IPIXH=1,NPIXH)
    END IF
    IF(ISFLG.EQ.-99 .OR. ISFLG.EQ.-98) THEN
    IF(NPIXH.LT.40) WRITE(*,987) (VSCALE(IPIXH),VSCALE(IPIXH),
```

```
     1 IPIXH=1,NPIXH)
       IF(NPIXH.GE.40) WRITE(*,987) (VSCALE(IPIXH),IPIXH=1,NPIXH)
       END IF
  985 FORMAT(A1)
  986 FORMAT('CHAR(176)')
  987 FORMAT(1X,79A1)
  988 CONTINUE
       IF(ISFLG.EQ.-99 .OR. ISFLG.EQ.-98) WRITE(*,990) VMIN,CHAR(32),
     1 CHAR(32),DIV1,CHAR(176),CHAR(176),DIV2,CHAR(177),CHAR(177),
     2 DIV3,CHAR(178),CHAR(178),DIV4,CHAR(219),CHAR(219),VMAX
  990 FORMAT('  BELOW <     ',F5.2,1X,2A1,F5.2,1X,2A1,F5.2,1X,2A1,
     1 F5.2,1X,2A1,F5.2,1X,2A1,F5.2,'    > ABOVE')
       IF(ISFLG.EQ.-99) GO TO 885
C*****IF VELOCITIES WILL BE COMPARED, STORE VELOCITIES IN COR*****
       IF(MDIF.NE.1 .AND. MDFLG.EQ.0) GO TO 999
       IF(MDFLG.EQ.-99) GO TO 996
       DO 991 IPIXEL=1,NPXTOT
  991 COR(IPIXEL)=V(IPIXEL)
       OPEN(3,FILE=OLDFIL,STATUS='OLD',ERR=128)
       MDFLG=-999
       GO TO 130
  992 DO 993 IPIXEL=1,NPXTOT
       V(IPIXEL)=COR(IPIXEL)-V(IPIXEL)
  993 CONTINUE
       WRITE(2,994) OLDFIL,SMFPLL
  994 FORMAT(//,'  VELOCITIES OF THIS RUN MINUS VELOCITIES FROM ',A12,
     1 /,' ### INDICATES SUM OF PATH FRACTIONS LESS THAN OR EQUAL ',
     2F7.4)
       VLODSP=VLODFP
       VHIDSP=VHIDFP
       MDFLG=-99
       ISFLG=-98
       GO TO 946
  996 RMS=0
       ABDIF=0
       NPX=0
       DO 997 IPIXEL=1,NPXTOT
       IF(SMFPIX(IPIXEL).LE.SMFPLL) GO TO 997
       ABDIF=ABDIF+ABS(V(IPIXEL))
       RMS=RMS+V(IPIXEL)*V(IPIXEL)
       NPX=NPX+1
  997 CONTINUE
       ABDIF=ABDIF/NPX
       RMS=SQRT(RMS/NPX)
       WRITE(2,998) SMFPLL,ABDIF,RMS
  998 FORMAT(/,'  FOR PIXELS WITH SUM OF FRACTIONS OF PATHS GREATER THAN
     1 ', G9.3,/,' THE AVERAGE ABSOLUTE VALUE OF VELOCITY DIFFERENCES =
     2',G9.3,/,' SQUARE ROOT (SUM OF DIFFERENCES SQUARED/NUMBER OF PIXEL
     3S) = ',G9.3)
  999 PRINT *,CHAR(7)
       PRINT *,CHAR(7)
       END
```

APPENDIX B.--INPUT FOR BOMTOM

```
MARCH 6, 1987. SYNTHETIC TRAVEL TIMES FOR 11 COLUMNS, 16 ROWS.
     0        0        1        1        1        0        0
   200        0   0.0000000   0.0000000   1.000000
     0        1        0        0        0        1        1
0.0000   0.0000   0.0000  15.0000   0.0000  15.0000
    11       16        0        0        0        0
    16       16      0.0
     1   0.0000   0.0000   0.0000
     2   0.0000   0.0000   1.0000
     3   0.0000   0.0000   2.0000
     4   0.0000   0.0000   3.0000
     5   0.0000   0.0000   4.0000
     6   0.0000   0.0000   5.0000
     7   0.0000   0.0000   6.0000
     8   0.0000   0.0000   7.0000
     9   0.0000   0.0000   8.0000
    10   0.0000   0.0000   9.0000
    11   0.0000   0.0000  10.0000
    12   0.0000   0.0000  11.0000
    13   0.0000   0.0000  12.0000
    14   0.0000   0.0000  13.0000
    15   0.0000   0.0000  14.0000
    16   0.0000   0.0000  15.0000
     1  15.0000   0.0000   0.0000
     2  15.0000   0.0000   1.0000
     3  15.0000   0.0000   2.0000
     4  15.0000   0.0000   3.0000
     5  15.0000   0.0000   4.0000
     6  15.0000   0.0000   5.0000
     7  15.0000   0.0000   6.0000
     8  15.0000   0.0000   7.0000
     9  15.0000   0.0000   8.0000
    10  15.0000   0.0000   9.0000
    11  15.0000   0.0000  10.0000
    12  15.0000   0.0000  11.0000
    13  15.0000   0.0000  12.0000
    14  15.0000   0.0000  13.0000
    15  15.0000   0.0000  14.0000
    16  15.0000   0.0000  15.0000
   256   0.0
     1     1  3.750000
     1     2  3.758325
     1     3  3.783186
     1     4  3.824264
     1     5  3.836941
     1     6  3.898944
     1     7  3.947081
     1     8  4.044185
     1     9  4.129261
     1    10  4.234066
```

1	11	4.363536
1	12	4.477325
1	13	4.605884
1	14	4.759353
1	15	4.919724
1	16	5.086348
2	1	3.758325
2	2	3.750000
2	3	3.758325
2	4	3.783186
2	5	3.772115
2	6	3.828121
2	7	3.863009
2	8	3.947081
2	9	4.006565
2	10	4.114772
2	11	4.218605
2	12	4.322564
2	13	4.460031
2	14	4.605884
2	15	4.759353
2	16	4.913063
3	1	3.783186
3	2	3.758325
3	3	3.750000
3	4	3.758325
3	5	3.731597
3	6	3.766321
3	7	3.792838
3	8	3.845042
3	9	3.910364
3	10	4.003878
3	11	4.080966
3	12	4.194310
3	13	4.322564
3	14	4.460031
3	15	4.602245
3	16	4.736796
4	1	3.824264
4	2	3.783186
4	3	3.758325
4	4	3.750000
4	5	3.707074
4	6	3.697205
4	7	3.737349
4	8	3.757556
4	9	3.827075
4	10	3.885886
4	11	3.968944
4	12	4.076136
4	13	4.194310
4	14	4.322564

4	15	4.438893
4	16	4.565864
5	1	3.881044
5	2	3.824264
5	3	3.783186
5	4	3.758325
5	5	3.698864
5	6	3.672908
5	7	3.697205
5	8	3.702583
5	9	3.748735
5	10	3.791140
5	11	3.873647
5	12	3.968944
5	13	4.076136
5	14	4.174431
5	15	4.281591
5	16	4.402382
6	1	3.952847
6	2	3.881044
6	3	3.824264
6	4	3.783186
6	5	3.707074
6	6	3.664773
6	7	3.638741
6	8	3.662812
6	9	3.667817
6	10	3.713453
6	11	3.773172
6	12	3.849169
6	13	3.907139
6	14	3.979545
6	15	4.083875
6	16	4.199647
7	1	4.008276
7	2	3.916912
7	3	3.836941
7	4	3.772115
7	5	3.680009
7	6	3.621658
7	7	3.579545
7	8	3.553324
7	9	3.576831
7	10	3.586696
7	11	3.634068
7	12	3.683335
7	13	3.745137
7	14	3.837273
7	15	3.926420
7	16	4.026449
8	1	4.081806
8	2	3.983798

8	3	3.898944
8	4	3.819300
8	5	3.685200
8	6	3.576831
8	7	3.553324
8	8	3.511363
8	9	3.519158
8	10	3.542438
8	11	3.563519
8	12	3.598786
8	13	3.647400
8	14	3.708420
8	15	3.799653
8	16	3.902272
9	1	4.177557
9	2	4.057621
9	3	3.947081
9	4	3.845042
9	5	3.704633
9	6	3.615668
9	7	3.542438
9	8	3.519158
9	9	3.511363
9	10	3.502074
9	11	3.473653
9	12	3.511370
9	13	3.563504
9	14	3.629432
9	15	3.708420
9	16	3.799653
10	1	4.273823
10	2	4.153409
10	3	4.044186
10	4	3.922603
10	5	3.755205
10	6	3.642889
10	7	3.580902
10	8	3.542438
10	9	3.467908
10	10	3.443182
10	11	3.450824
10	12	3.473653
10	13	3.511370
10	14	3.563504
10	15	3.629432
10	16	3.696181
11	1	4.404509
11	2	4.273823
11	3	4.153409
11	4	4.001191
11	5	3.806332
11	6	3.683335

11	7	3.598786
11	8	3.528753
11	9	3.473653
11	10	3.450824
11	11	3.443182
11	12	3.450824
11	13	3.473653
11	14	3.511370
11	15	3.537042
11	16	3.593497
12	1	4.544581
12	2	4.404509
12	3	4.262779
12	4	4.051989
12	5	3.839960
12	6	3.745137
12	7	3.647400
12	8	3.563504
12	9	3.511370
12	10	3.473653
12	11	3.450824
12	12	3.443182
12	13	3.450824
12	14	3.439260
12	15	3.476604
12	16	3.528222
13	1	4.675008
13	2	4.515757
13	3	4.363536
13	4	4.110380
13	5	3.940909
13	6	3.818463
13	7	3.708420
13	8	3.629432
13	9	3.563504
13	10	3.511370
13	11	3.473653
13	12	3.416658
13	13	3.409091
13	14	3.416658
13	15	3.439260
13	16	3.476604
14	1	4.806200
14	2	4.649541
14	3	4.456187
14	4	4.199647
14	5	4.035284
14	6	3.902272
14	7	3.799653
14	8	3.708420
14	9	3.629432
14	10	3.554683

```
        14          11   3.476604
        14          12   3.439260
        14          13   3.416658
        14          14   3.409091
        14          15   3.416658
        14          16   3.439260
        15           1   4.966358
        15           2   4.790585
        15           3   4.558588
        15           4   4.310146
        15           5   4.138189
        15           6   4.015406
        15           7   3.902272
        15           8   3.796965
        15           9   3.683942
        15          10   3.593497
        15          11   3.528222
        15          12   3.476604
        15          13   3.439260
        15          14   3.416658
        15          15   3.409091
        15          16   3.416658
        16           1   5.126524
        16           2   4.919725
        16           3   4.658717
        16           4   4.431253
        16           5   4.269792
        16           6   4.138189
        16           7   3.999945
        16           8   3.868465
        16           9   3.762032
        16          10   3.671703
        16          11   3.593497
        16          12   3.528222
        16          13   3.476604
        16          14   3.439260
        16          15   3.416658
        16          16   3.409091
          0.001     -99.     4.4      3.8       4.6     -0.5    0.5     3
    1  4.000000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.000000
    2  4.000000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.000000
    3  4.000000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.000000
    4  4.000000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.000000
    5  4.000000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.400000
    6  4.000000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.400000
    7  4.400000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
```

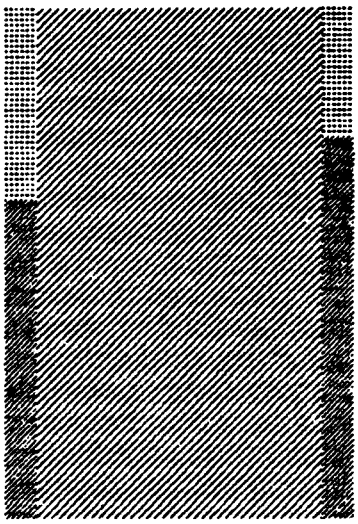
```
4.200000  4.200000  4.200000  4.400000
    8  4.400000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.400000
    9  4.400000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.400000
   10  4.400000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.400000
   11  4.400000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.400000
   12  4.400000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.400000
   13  4.400000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.400000
   14  4.400000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.400000
   15  4.400000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.400000
   16  4.400000  4.200000  4.200000  4.200000  4.200000  4.200000  4.200000
4.200000  4.200000  4.200000  4.400000
```

APPENDIX C.--OUTPUT FROM BOMTOM

```
1
 MAY 7, 1987. FIXED VELOCITY IN BOREHOLES.
 TOMOGRAPHIC RECONSTRUCTION FROM BOMTOM

 INPUT FILE IS BT1116S.IN
 THIS OUTPUT FILE IS BT1116S.OUT
 OUTPUT DATA FILE IS BT1116S.DAT
 MDATA = 0, POSITIONS AND TIMES ARE INPUT SEPARATELY
 MWT = 0, NO WEIGHTING FACTORS FOR ACCURACY OF TRAVEL TIMES
 MSTOR = 1, STORE AND RECALL PATH SEGMENTS
 MCOR = 1, ADDITIVE CORRECTION FACTORS
 MBORE = 1, VELOCITIES AT TRANS AND REC POS ARE FIXED
 MSMTH = 0, NO SMOOTHING
 MDIF = 0, WILL NOT BE COMPARED WITH PRIOR RUN
 IPDAT,IPIVG,IPFRC,IPERR,IPSEG,IPGRA,ISGRA =  0, 1, 0, 0, 0, 1, 1

 NUMBER OF EXPERIMENTAL TRAVEL TIMES = 256
 CENTER OF PIXEL 1 IS AT X =     0.0000  Y =     0.0000  Z =     0.0000
 CENTER OF LAST PIXEL IS AT X =   15.0000  Y =     0.0000  Z =   15.0000
 NUMBER OF COLUMNS, ROWS OF PIXELS =  11, 16
 PIXEL WIDTH, HEIGHT =     1.5000,    1.0000
 NUMBER OF LATERALLY INVARIANT ROWS AT TOP, BOTTOM =  0, 0
 NUMBER OF FIXED VELOCITY COLUMNS AT LEFT, RIGHT =  0, 0
 ABSOLUTE ERROR TOLERANCE FOR CONVERGENCE =  0.000000
 INCREMENTAL ERROR TOLERANCE FOR CONVERGENCE =  0.000000
 RELAXATION PARAMETER =  1.00000
 DISTANCE ADJUSTMENT ADDED BETWEEN BOREHOLES =     0.0000
 TIME ADJUSTMENT ADDED TO ALL TRAVEL TIMES =     0.0000
 HIGHEST ALLOWED VELOCITY =   4.400

 INITIAL GUESSES FOR VELOCITIES FOR EACH PIXEL
  4.00 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.00
  4.00 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.00
  4.00 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.00
  4.00 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.00
  4.00 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
  4.00 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
  4.40 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
  4.40 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
  4.40 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
  4.40 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
  4.40 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
  4.40 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
  4.40 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
  4.40 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
  4.40 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
  4.40 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.20 4.40
```

GRAPH OF INITIAL VELOCITY GUESSES
VELOCITY LIMITS FOR A PATTERN ARE ON EACH SIDE OF IT.

BELOW < 3.80 3.96 4.12 4.28 4.44 4.60 > ABOVE



DIMENSION NEEDED FOR D(ISG) IS ISG = 26
IF MSTOR = 1, DIMENSION NEEDED FOR DP(IARSG), IDPIX(IARSG),
IPP(NEXPTT), AND DPPATH(NEXPTT) IS IARSG = 4176 NEXPTT = 256

NUMBER OF ITERATIONS= 200 AVE ERROR= 0.126E-02
CHANGE/ITER IN AVE ERROR= 0.437E-05
SQUARE ROOT OF (SUM OF SQUARES OF ERROR/NUMBER OF PATHS)= 0.174E-02

CALCULATED VELOCITIES FOR EACH PIXEL
INDICATES SUM OF PATH FRACTIONS LESS THAN OR EQUAL 0.100E-02

4.00	3.94	4.00	4.02	4.02	4.03	4.04	4.04	4.00	3.91	4.00
4.00	3.96	3.97	4.02	4.02	4.04	4.06	4.03	3.99	3.90	4.00
4.00	3.95	3.97	4.01	4.04	4.05	4.04	4.03	3.97	3.93	4.00
4.00	3.94	3.98	4.02	4.05	4.06	4.04	4.00	3.96	3.95	4.00
4.00	3.93	3.98	4.03	4.06	4.05	4.04	3.97	3.96	4.37	4.40
4.00	3.92	3.99	4.04	4.07	4.05	4.02	3.96	4.35	4.39	4.40
4.40	4.30	3.98	4.06	4.06	4.08	3.97	4.36	4.36	4.40	4.40
4.40	4.32	4.34	4.08	4.06	4.09	4.31	4.36	4.38	4.40	4.40
4.40	4.35	4.32	4.05	4.12	4.06	4.30	4.37	4.40	4.40	4.40
4.40	4.38	4.32	4.38	4.22	4.38	4.32	4.37	4.40	4.40	4.40
4.40	4.40	4.35	4.29	4.27	4.31	4.36	4.38	4.40	4.40	4.40
4.40	4.40	4.40	4.30	4.19	4.35	4.39	4.38	4.40	4.40	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.38	4.40	4.39	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.39	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.39	4.40
4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40	4.40

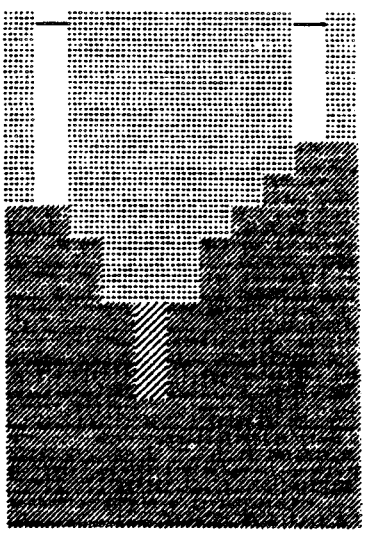
VELOCITIES SCALED TO RANGE 0 TO 9
VELOCITY CORRESPONDING TO 0 = 3.80
VELOCITY CORRESPONDING TO 9 = 4.60
# INDICATES SUM OF PATH FRACTIONS LESS THAN OR EQUAL 0.0010

```
 2 2 2 2 2 3 3 3 2 1 2
 2 2 2 2 3 3 3 3 2 1 2
 2 2 2 2 3 3 3 3 2 2 2
 2 2 2 2 3 3 3 3 2 2 2
 2 1 2 3 3 3 3 2 2 6 7
 2 1 2 3 3 3 2 2 6 7 7
 7 6 2 3 3 3 2 6 6 7 7
 7 6 6 3 3 3 6 6 7 7 7
 7 6 6 3 4 3 6 6 7 7 7
 7 7 6 6 5 7 6 6 7 7 7
 7 7 6 6 5 6 6 7 7 7 7
 7 7 7 6 4 6 7 7 7 7 7
 7 7 7 7 7 7 7 7 7 7 7
 7 7 7 7 7 7 7 7 7 7 7
 7 7 7 7 7 7 7 7 7 7 7
 7 7 7 7 7 7 7 7 7 7 7
```

GRAPH OF VELOCITIES
VELOCITY LIMITS FOR A PATTERN ARE ON EACH SIDE OF IT.

BELOW < 3.80 3.96 4.12 4.28 4.44 4.60 > ABOVE

# INDICATES SUM OF PATH FRACTIONS LESS THAN OR EQUAL 0.0010



U.S. GOVERNMENT PRINTING OFFICE: 1988 — 547-000/80,020

INT.-BU.OF MINES,PGH.,PA. 28667