# Seismic Modeling and Inversion On The NCUBE

J. Sochacki[1], P. O'Leary[2], C. Bennett[3], R. E. Ewing[2,3], and R. C. Sharpley[3]

[1]Department of Mathematics and Computer Science, James Madison University
[2]Institute for Scientific Computation, University of Wyoming
[3]Department of Mathematics, University of South Carolina

## Introduction

The uncovering of the earth's interior encompasses two important procedures. The first step is to make a prediction of the earth's interior based on scientific data collected from geophones and a knowledge of the geology of the area in which the data is being collected. The second step is to improve this prediction by numerical simulation of the wave equations. The former step is referred to as the inverse problem and the latter as the forward problem. Both problems are difficult and interconnected. Once there is scientific confidence that the structure of the interior of the earth is adequately known, this structure is encoded and analyzed under various stresses, strains, and pressures. Using wave equations to simulate these types of problems involves an immense number of calculations, overburdening the largest available vector and parallel computers. In this paper, we discuss these aspects and indicate how the distributed memory computations can be used to solve them in an efficient manner.

The inverse problem in which we are interested is to determine not only the depth, but also the shape, of complex structures below the surface of the earth. That is, given a known disturbance of the earth and the record of the geophones caused by this disturbance, we wish to accurately describe the structure. The forward problem then takes this structure and the disturbances as its data and produces the wave field over time, testing for comparison with surface waves and geophone readings. In an iterative manner, the inverse solver utilizes these results to improve the initial guess of the underlying structure of the earth. This process is repeated until convergence to the true structure is obtained to within a specified degree of accuracy. Once this accurate structure is obtained, the forward solver is used to test how this section of the earth reacts to various pressures, stresses, and strains.

Many of the inherent problems encountered in numerically approximating the wave equations to simulate the propagation of sound waves in the earth's interior have been well documented and analyzed in the literature. In this paper, we only discuss a few of the major difficulties. One of the most commonly mentioned problems is related to the vastness of the earth's interior. The numerical solution of the wave equation requires the placing of grid blocks over a finite region and therefore requires boundary conditions for the computational domain. However, the earth's interior (for the problem of interest) has no subsurface boundaries. Therefore, the boundary conditions imposed must model a 'void' boundary resulting in what are known in the literature as *absorbing* or *radiating boundary conditions*. We incorporate absorbing boundary conditions and load-balancing in the distributed memory setting of computation. Since memory is limited, the sizes for the grid blocks are bounded from below. The hyperbolic nature of the model equations requires that a correspondingly small time step be used to avoid dispersion and numerical instabilities. The limit on the grid size also restricts how accurately the interfaces of the complex structures can be represented. In this paper, we consider all these problems in the forward model using finite differences and distributed memory computing, and strongly argue that the improper treatment of any of the above-mentioned topics can lead to gross misinterpretations in the inverse problem.

The equations that we model for the pressure distribution are the acoustic wave equations

$$P_t + c^2 \vec{\nabla} \cdot (\rho \vec{v}) = 0$$
$$\vec{v}_t + \rho^{-1} \vec{\nabla} P = F(\vec{x}, t),$$

where $\vec{x} = (x_1, x_2, x_3)$, $P$ is the pressure distribution, $\vec{v} = (v_1, v_2, v_3)$ is the velocity, $\rho$ is the density, and $c$ is the speed of sound in the medium. However, we actually solve the equivalent potential equation

$$U_{tt} + A(\vec{x})\, U_t - c^2\, \vec{\nabla} \cdot (\vec{\nabla} U) = g(\vec{x}, t),$$

where $g = c^2\vec{\nabla} \cdot (\rho G)$ and $G_t = F$, noting that $P = -U_t$ (cf. Sochacki et al., 1990). The parameters $\rho$ and $c$ are determined by the structure of the earth's interior and are thus obtained from the inverse problem.

For measuring the displacement at the earth's surface, we use the elastic wave equations

$$\rho U_{tt} + A(\vec{x})U_t - \vec{\nabla} \cdot \vec{S}_1 = F_1(\vec{x}, t)$$
$$\rho V_{tt} + A(\vec{x})V_t - \vec{\nabla} \cdot \vec{S}_2 = F_2(\vec{x}, t)$$
$$\rho W_{tt} + A(\vec{x})W_t - \vec{\nabla} \cdot \vec{S}_3 = F_3(\vec{x}, t),$$

where $\vec{S}_i = (S_{i1}, S_{i2}, S_{i3})$ and $S_{ij} = \delta_{ij} + 2me_{ij}$, and $\delta_{ij}$ is the Kronecker function. Moreover, $e_{ij} = \frac{1}{2}(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i})$ and $u_1 = U$, $u_2 = V$, $u_3 = W$, $p = \sqrt{\frac{l+2m}{\rho}}$ is the $p$-wave velocity and $\sigma = \sqrt{m/\rho}$ is the $s$-wave velocity. (cf. Ewing, Jardetzky, and Press(1957)). The parameters $p$, $\sigma$, and $\rho$ are also determined by the earth's structure.

The term $A(\vec{x})$ is used for the absorbing boundary conditions rather than dissipation. This term is equal to zero in the interior, since we are modeling nondissipative waves, and is assigned values on the boundary of the model so that waves are sufficiently decayed to reduce the amplitude of the spurious reflected waves off of the boundary (cf. Sochacki et al., 1987). The source terms are localized disturbances occuring either in the interior or at the surface.

Although the problems discussed above occur in both two- and three-dimensional wave propagation, we address two dimensions in this paper because of the simplicity of visualization. Also, since the problems discussed above are similar in acoustic and elastic wave propagation, we only consider two-dimensional acoustic wave equations. However, we also discuss the problems specific to three-dimensional wave propagation and elastic waves as they arise. Also, a single complicated interface can illustrate all the problems that occur with a region containing many complicated interfaces; thus the model we analyze deals with a single interface. The technique used to handle the numerical calculations required at an interface is taken from Sochacki et al. (1990). The distributed memory computation allows two different programming strategies to be considered when solving the finite difference equations. We discuss the pros and cons of these two strategies and present timings for each.

Of course, all the considerations discussed above must be displayed visually in order to allow proper analysis and interpretation. The graphics can be done on the parallel processing machine or the data sets for the graphics can be transferred to a graphics workstation that has greater visualization capabilities.

The graphics produced by the NCUBE/ten are a result of information on the nodes being dumped to an 8-bit or 24-bit graphics board. The viewpoint of these snapshots would be fixed. Hence, one can only analyze a given data set until this data is replaced by an updated set. The interactive capabilities would be acheived only through a sequence of runs. On the other hand, if data sets are passed through the Sun4 to a high performance graphics workstation via efficient data compression techniques, the process of interactive analysis is enhanced.

## The Interface Problems

The model considered is 1600 meters by 1600 meters and contains a complicated interface at an average depth of 800 meters (see Figure 1). The interface is actually at a depth of 800 meters at each horizontal 8 meter interval. Between these points, the interface has a complicated and random shape. The purpose of this choice of interface configuration is to show that extremely different surface seismograms are generated by using different grid sizes. The simulations (Table 1) are run for constant size square grids varying from $h$=8 meters down to 2 meter. The $p$-wave velocity above the interface is 2000 $m/s$ and the density is 3200 $kg/m^3$. Below the interface, the $p$-wave velocity is 6000 $m/s$ with a density of 2600 $kg/m^3$. In all cases, the source is located at a depth of 400 meters and a horizontal distance of 800 meters. The source is the derivative of the Gaussian and has the form

$$f(t) = A\,(t - t_0)\,e^{-s(t-t_0)^2}.$$

For stability, the time step $\Delta t$ must be chosen to satisfy the CFL condition: $\Delta t \leq \frac{h}{c\sqrt{2}}$, where $c$ is the maximum $p$-wave velocity. To minimize dispersion, the source should act for $t = 2t_0$ seconds and $h$ should be no larger than $\frac{at}{10}$, where $a$ is the minimum $p$-wave velocity, and $\sigma$ should be no smaller than $20\ln(10)/t^2$. In Table 1, we give the parameters used in the model runs to show the discrepancies of surface seismograms. The time shown is the number of time steps it takes for the wave to reflect off the interface and create a reasonable surface seismogram; this is approximately .5 seconds.

Of course, the memory needed and the number of calculations done are directly proportional to the number of grids and time steps. For $h \leq 4$ it is clearly seen that a supercomputer is needed to carry out the calculations. Therefore, the parallel computer is an excellent machine to display the importance of representing an interface accurately for the inverse problem.

In the field, seismologists use source frequencies from 2 Hz to 100 Hz. In these models, a derivative of a Gaussian source provides a frequency equal to $\frac{2}{t}$. As illustrated in Table 1, a higher frequency is possible as smaller values of $h$ are allowed. A parallel distributed memory architecture provides the memory and computational power needed to decrease $h$ to realistic physical values. In addition, with the decrease in $h$, $\sigma$ is used to maintain continuity in time and not for dispersion.

All the above models use a single interior source which requires that the node whose grid points contain the source information does substantially more computations on startup than the other nodes, resulting in load imbalance. This increased load is insignificant, however, compared with the total number of computations that must be done for large problems, i.e. small $h$. Also, to highlight the differences in the seismograms and snapshots, the data is compressed to be outputted every 8 meters so that the sizes remain the same in all three runs. Currently, the damping (or ABC) term $A(x)$ is nonzero only for the 30 outer grid points of the bottom and edges. Hence, in the interior of the model, there is a memory drain in our algorithms. One could play off memory versus performance on this aspect, but we do not address this in the current investigation.

In both the seismograms and the snapshot for the entire wavefield it is easily seen that completely different information is given by using the finer grid sizes. It is also worth noting that the reflections off the random interfaces shows up clearly in the seismograms (Figure 2) and on the interface in the snapshots (Figure 1) . Initially, however, it appears that the seismograms are similar. Therefore, the importance of having accurate forward solvers to test for interfaces in the inverse problem is highlighted in the seismograms. The memory capabilities and computational speed of the NCUBE/ten were necessary for carrying out this numerical experiment.

## Computing Strategies

There are two basic methods for locating the interfaces using the finite difference scheme presented in Sochacki et al. (1990). The strategies depend on how one describes the $p$-wave velocity and density at each grid point. The two different strategies arise when attempting to pass these parameters to the equations being calculated in the most efficient manner for the distributed memory. One method (Method A) is to create a matrix that contains an integer for each grid point indicating its region. Each $p$-wave velocity and density is assigned the corresponding integer. This means that we need a matrix equal in size to the number of grid points and two vectors equal in size to the number of structures. This strategy leads to a load balancing problem at the interfaces, since for this scheme the number of calculations away from an interface is much smaller than at the interface. For models that are not too complicated, this problem can be alleviated by strategic assignment of the nodes to the interfaces. Also, each grid point must be tested to see if it lies on an interface at each time step. This, however, does not cause load balancing problems; it just increases the number of operations.

The second method (Method B) is to create two matrices both of which are equal in size to the number of grid points. One matrix contains the $p$-wave velocity at each grid point while the other contains the density at each grid point. This strategy eliminates load balancing problems, because at each grid point the same calculations are being performed. However, this scheme increases the amount of memory needed and the total number of calculations done significantly. However, if there is a large number of interfaces the number of calculations is balanced by the testing of each grid point in the former scheme.

We have presented timings for both of these schemes applied to the model with the single interface for three grid sizes. In Table 2 we see that Method B is much more efficient for smaller matrices. This is due to the fact that the extra number of calculations in this method for smaller problems does not overtake the grid point checking of Method A and that the memory requirements are still minimal. However, we see that as the model size increases the calculation times for Method A and Method B approach the same magnitude. In addition, we present timings for both schemes applied to a model containing seven structures with relatively complicated interfaces to test for the trade-off in this situation between performing conditional and computational instructions (see Figure 3). The data for the various $p$-wave speeds $(m/s)$ and corresponding densities $(kg/m^3)$ of the structures are provided in Table 3.

The time increment $\Delta t$ is .00047 and a source is used which has a duration of $t = .126$ and a spread $\sigma = 3143$. We have presented timings for this medium comparing both Methods A and B in Tables 4-5. Table 4 gives the timings which include

the initial startup computations, including the source calculations, while Table 5 gives the timings for later time only.

These two tables provide a test for the trade-off between performing conditional and computational instructions in this situation. Here again, we see that the calculation times are of the same magnitude. However, in all the cases Method B is faster than Method A, and this suggests that a clever method for assigning the nodes to the interface calculations is appropriate.

We also note that in three dimensions, the sizes of the matrices for these strategies are increased in size by a factor of the number of grid points in the extra dimension; additionally, for elastic wave simulation, a matrix for the $s$-wave velocity is needed in the latter strategy.

## Conclusions

The locating of interior structures in the earth's interior is one of the important challenges of geophysics. One method of attacking this problem is using the acoustic and elastic wave equations in two and three dimensions on distributed memory machines. In this paper we have presented two methods for accomplishing this and have presented data from these two methods performed on an NCUBE/ten. There are many more tests that can be run on the two strategies presented here, and these need to be carried out; however, the groundwork has been layed.

The data we have presented are for 2D acoustic wave analysis, but the ideas can be carried to 2D elastic wave analysis and 3D in a similar manner. The main problem in 3D is that the structures become more complicated and the number of calculations is greatly increased. However, the work done here is currently being extended to 3D, and the major differences are in visualization and the fact that all the nodes of the NCUBE/ten must be used.

### Acknowledgments

The authors would like to thank Chuck Baldwin, Bill Nestlerode, and Mark Oliver for all their help in porting the codes to the NCUBE and in producing the figures and graphs. This work was supported in part by Westinghouse, by Office of Naval Research Contract No. 0014–88–K–0370, by the Pittsburgh and Minnesota Supercomputer Centers, and by the Institute for Scientific Computation through NSF Grant No. RII–8610680.

### References

[1 ] Ewing W.M., Jardetzky W.S., and Press F. (1957), *Elastic Waves in Layered Media*, McGraw Hill.

[2 ] Sochacki J., Kubichek R., George J., Fletcher R.W., and Smithson S. (1987),*Absorbing Boundary Conditions and Surface Waves*, Geophysics, **52**, 60–71.

[3 ] Sochacki J., George J., Ewing R., and Smithson S. (1990), *Interface Conditions for Acoustic and Elastic Wave Propagation*, Geophysics, in press.

**Table 1. Media Parameters**

| h | $\Delta$ t | number of grid points | t | $\sigma$ | source frequency | total time steps |
|---|---|---|---|---|---|---|
| 8 | .00094 | 200 × 200 | .252 | 3,131 | 7.95 Hz. | 725 |
| 4 | .00047 | 400 × 400 | .126 | 12,415 | 15.87 Hz. | 1700 |
| 2 | .00024 | 800 × 800 | .063 | 48,500 | 31.74 Hz. | 2400 |

**Table 2. Single Processor Timings**

| # grid pts. | Method A | Method B |
|---|---|---|
| 10× 10 | .018 s. | .0038 s. |
| 25× 25 | .377 s. | .020 s. |
| 50× 50 | 1.472 s. | 1.033 s. |

**Table 3. Subregion parameters**

| structure: | 1 | 2&4 | 3&7 | 5&6 |
|---|---|---|---|---|
| $p$-wave velocity: | 1000 | 2000 | 3000 | 6000 |
| density: | 1000 | 2800 | 2500 | 5800 |

**Table 4. Timings (including startup time)**

| | Method A | Method B |
|---|---|---|
| Sequential | 94.214 | 66.1071 |
| Concurrent | 3.0331 | 2.3214 |
| Speedup | 31.0678 | 28.4772 |
| Percent Speedup | 48.5% | 44.5% |

**Table 5. Timings (excluding startup time)**

| | Method A | Method B |
|---|---|---|
| Sequential | 94.214 | 66.1071 |
| Concurrent | 2.7386 | 2.3021 |
| Speedup | 34.4084 | 28.7160 |
| Percent Speedup | 53.8% | 44.9% |

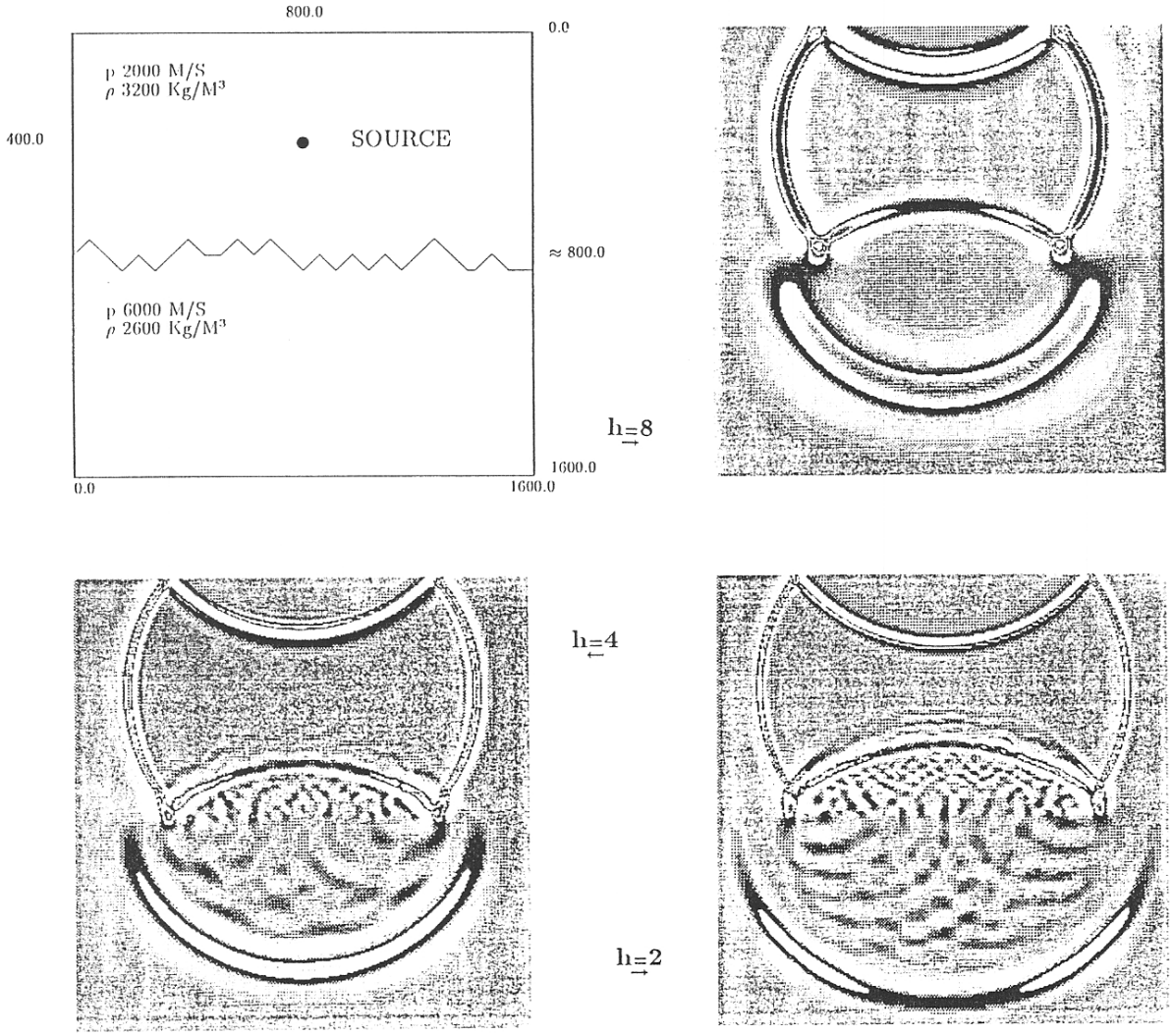


*Figure 1.*

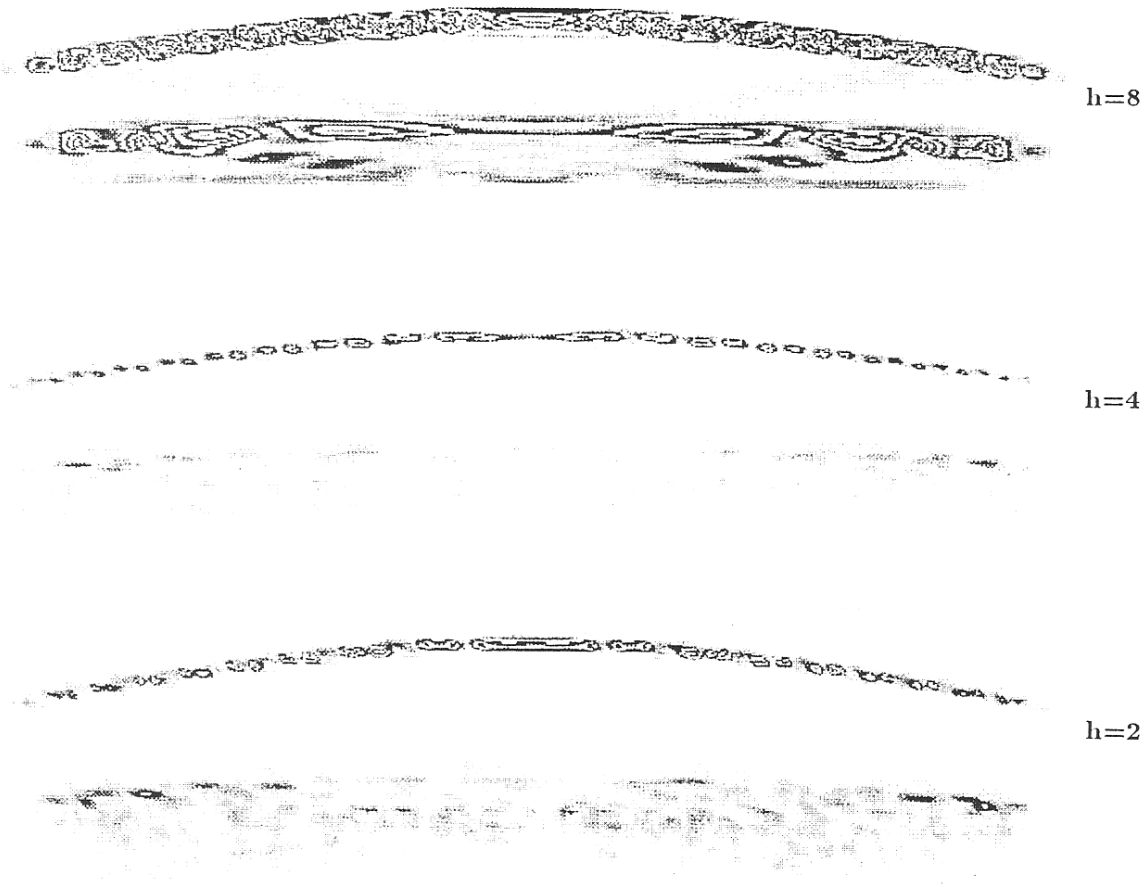


*Figure 2.*

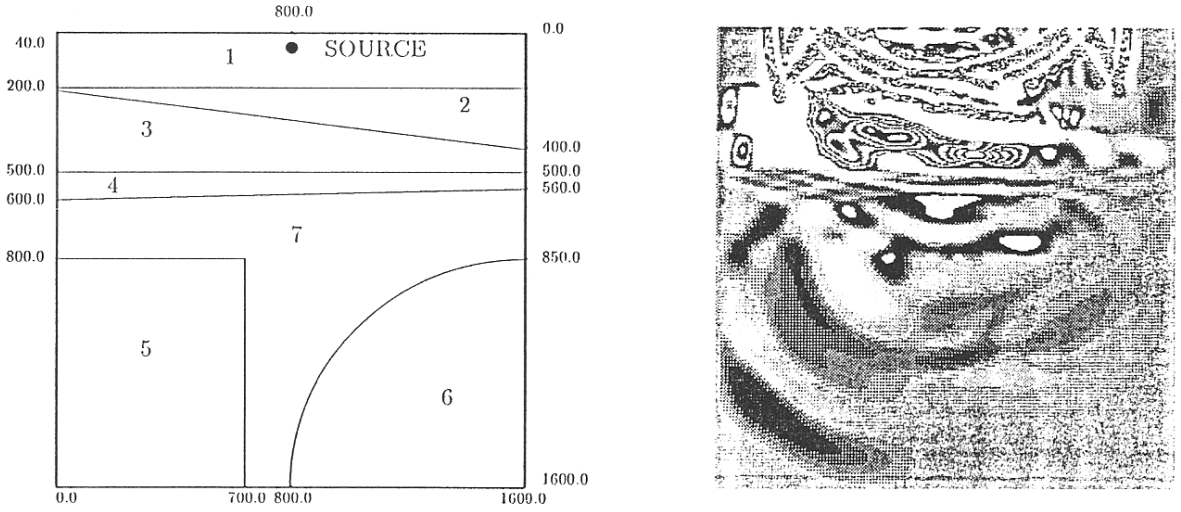


*Figure 3.*