Automatic estimation of very large residual statics

Daniel H. Rothman

ABSTRACT

Conventional approaches to residual statics estimation obtain solutions by performing linear inversion of observed traveltime deviations. A crucial component of these procedures is the picking of time delays; gross errors in these picks are known as "cycle-skips" or "leg-jumps," and are the bane of linear traveltime inversion schemes.

This paper is a sequel to an earlier work (Rothman, 1984), which demonstrated that the estimation of large statics in noise-contaminated data is posed better as a nonlinear, rather than as a linear, inverse problem. Cycle-skips then appear as local (secondary) minima of the resulting nonlinear optimization problem. In the earlier paper, a Monte Carlo technique that originated in statistical mechanics was adapted to perform global optimization, and the technique was applied to synthetic data. This paper presents an application of a similar Monte Carlo method to field data from the Wyoming Overthrust belt. Key changes, however, have led to a more efficient and practical algorithm. The new technique performs explicit crosscorrelation of traces. Instead of picking the peaks of these crosscorrelation functions, the method transforms the crosscorrelation functions to probability distributions, and then draws random numbers from these distributions. Estimates of statics are iteratively updated by this procedure until convergence to the optimal stack is achieved.

This paper also derives several theoretical properties of the algorithm. The method is expressed as a Markov chain, in which the equilibrium (steady-state) distribution is the Gibbs distribution of statistical mechanics.

INTRODUCTION

The near-surface of the Earth is often highly irregular. Typical inhomogeneities include changes in surface elevation, uneven consolidation of soil, and extreme lateral variations in seismic velocity. These irregularities can twice degrade the quality of a seismic signal: first as the signal travels downward from the source, and again when the signal returns upward to the receivers. Among the many effects caused by the near-surface, the most obvious and most important are often delays in recorded traveltimes. Because these timing delays distort apparent structure and cause misalignment of signals, the analysis of seismic data acquired on land routinely includes the use of timing adjustments called statics corrections. In their most elementary form, statics corrections can be derived deterministically from measurements of elevations, uphole times, and first-arrivals—these corrections are called "field statics." Statics corrections that cannot be derived from these field measurements are called "residual statics." Residual statics are usually estimated statistically from the seismic data.

When near-surface velocity anomalies are severe, residual statics can be 100 ms or greater, which can be several multiples of the dominant period of the data. Because noise significantly contaminates most seismic data, the automatic identification of these large static shifts is difficult. If the true peaks of crosscorrelation functions are obscured, then the so-called "cycle-skipping" or "leg-jumping" problem results.

A recent paper (Rothman, 1984) demonstrated that the estimation of large statics in noise-contaminated data is posed better as a nonlinear, rather than as a linear, inverse problem. Cycle-skips then appear as local (secondary) minima of the resulting nonlinear optimization problem. To perform global optimization, Rothman (1984) adapted a Monte Carlo technique that originated in statistical mechanics and applied this technique to synthetic data. This paper presents an application of a similar Monte Carlo method to field data from the Wyoming Overthrust belt. Although the optimization technique introduced here is fundamentally the same as that previous one, several key changes have led to a more efficient, more practical, and more readily explicable method. This paper also derives several theoretical properties of the new statics algorithm.

The paper begins by introducing the field data example that is the subject of later discussion. This example clarifies the meaning of the term "large statics," and illustrates the nature of the data-processing and interpretation problems that large statics cause. A brief review of many of the ideas developed in Rothman

(1984) is then followed by an explication of the new Monte Carlo technique, which is based on the transformation of crosscorrelation functions to probability distributions. Theoretical properties of the algorithm are discussed; in particular, the formal connections with the theory of Markov chains are emphasized. Finally, the application of the technique to data from the Wyoming Overthrust belt is illustrated. Several characteristics of the algorithm emerge from the discussion of this example; the most interesting features include the the nature of the algorithm's approach to the final solution and its behavior due to non-uniqueness.

VERY LARGE STATICS

Before discussing the technical details of the statics algorithm, I will first illustrate the type of statics problem being addressed. Looking ahead to the field data example in Figure 2a, we see a 24-fold common-midpoint stack from the Wyoming Overthrust belt; this stack was produced without statics corrections. These data have been previously analyzed by Johnson et al. (1983). According to their interpretation, the survey traverses an area in which anomalous, low-velocity fill is as deep as 500 m at both ends of the line. Indeed, lateral variations in the near surface are evident from the roughly continuous reflector that appears on the left at about 600 ms, arches upward to 100 ms at the center of the line, and then dips slowly downward to about 600 ms at the far right.

These data will be the subject of much discussion later in the paper. At this time, however, it is advantageous to peek ahead to Figure 9, which shows the data after statics corrections have been applied. As we shall see later, the combined (shot plus receiver) statics corrections that yielded this stack are as large as 200 ms. After such large statics corrections are applied, the appearance of the data changes significantly. Among the many features that have been revealed are the steeply dipping events in the first 3 seconds of the data, and the now continuous, relatively flat, deep reflector at 3.5 s.

No single quantity demarcates the "small" statics problem from the "large" statics problem. In principle, this distinction depends on a mixture of bandwidth, signal-to-noise ratio, and the magnitude of the timing delays. For our purposes here, a practical delineation is defined by the success or failure of a conventional statics algorithm. Looking ahead again, we see that Figure 11 presents one such failure; in particular, the deep reflector (which was used to estimate the statics) shows severe "cycle-skipping" or "leg-jumping." In this paper, a large statics

problem is defined to be a statics problem in which cycle-skipping is a potential pitfall, as it is for this set of data. The objective of the remainder of this paper is to present an algorithm that can accurately solve for large statics without falling prey to a cycle-skipping problem.

RESIDUAL STATICS ESTIMATION BY SIMULATED ANNEALING

In addressing the problem of estimating large statics (Rothman, 1984), I adapted the simulated annealing algorithm of Kirkpatrick et al. (1983), which, in turn, was based on a Monte Carlo technique due to Metropolis et al. (1953). Despite the encouraging success of the technique on synthetic data, I found my early implementation of the Metropolis algorithm to be ineffective when applied to the field data used in this paper. The overriding problem was computational speed: the Metropolis algorithm is simply too inefficient. Thus a new, more efficient, and more readily vectorizable algorithm was developed. The new technique uses the familiar tool of crosscorrelation, yet it maintains the same probabilistic properties of the earlier method. Thus the technique retains the crucial ability to escape local minima.

Before discussing the new developments, I will first briefly review the elements of the previous method that are necessary to clarify the ensuing discussion.

Review

Statics estimation is posed as the following problem. Let the unknown shot and receiver statics be represented by the vector-valued parameter $\mathbf{x}$. Following Ronen and Claerbout (1985), I assume that the quality of the statics solution can be quantified by the power in a common-midpoint stack, and seek the solution of the optimization problem

$$\min_{\mathbf{x}} E(\mathbf{x}) \quad , \tag{1}$$

where E is the negative power of the stack. This optimization problem is addressed here for cases in which statics are large; that is, when cycle-skipping is a problem. Because cycle-skips appear as local minima of E, optimization must be performed globally instead of locally, thus avoiding the local minima.

Optimization of (1) can be nontrivial: linearized techniques that descend to the nearest minimum often fail when statics are large. A previous paper (Rothman, 1984) demonstrated global optimization by simulated annealing (Kirkpatrick et al., 1983), which is a method based on an analogy between optimization

and crystallization. Briefly stated, the idea is to generate a sequence of parameter vectors in a way that mimics thermal equilibrium. By slowly lowering a control parameter analogous to temperature, a system that simulates equilibrium eventually settles into its global minimum, or, analogously, its ground state. Thermal equilibrium is simulated by use of the Metropolis algorithm (Metropolis et al., 1953). To estimate residual statics, the Metropolis algorithm sequentially "visits" each shot and receiver static and conditionally assigns a new, randomly chosen value to it. The change in stack power, ΔE, is then computed. If $\Delta E < 0$ (i.e., if stack power increases), then the new value for this shot or receiver static is always accepted. If $\Delta E \geq 0$, then the new value is accepted sometimes, with probability $\exp\{-\Delta E / T\}$, where T is "temperature." Each time a new value is accepted, the stack is accordingly updated. Note that the objective function, negative stack power, is occasionally allowed to increase, so that local minima can be escaped. Random moves according to these rules eventually simulate thermal equilibrium, and when T is small enough the global minimum can be approximately located.

Transforming correlation to probability: Random moves in one step

The Metropolis version of the statics algorithm sequentially cycles through a two-step scheme for each parameter: (1) a random guess for the static shift is made; and then (2) a decision is made to either accept or reject this new guess. For the reasons discussed below, the new technique consolidates the two-step Metropolis method into a single step in which a random guess is made and always retained.

When estimating statics, the Metropolis random guesses are uniformly likely between some predetermined maximum and minimum value for each static. In the field data example of this paper, statics are initially limited to the range ± 160 ms. Coarse discretization at 8 ms intervals would then yield 41 possible values from which one could draw a random guess if one were to use the Metropolis technique. If the Metropolis algorithm were to locate a region near a deep minimum at a low temperature, then only a very few of these 41 candidates would be acceptable guesses. Much effort can therefore be wasted in the evaluations of random guesses that have a high probability of rejection.

The new technique avoids low acceptance-to-rejection ratios by computing the relative probabilities of acceptance for each possible move before any random guesses are made. Instead of making random guesses that are then accepted or

rejected, the new method simply produces weighted guesses that are always accepted. In contrast to the two-step Metropolis method, this is a *one-step* technique. The one-step method sequentially "visits" each shot and receiver static X_m, and calculates stack power ϕ_m as a function of static shift s_j for each of N possible shifts. The algorithm then chooses a new value for X_m by drawing a random number from the probability distribution

$$P(X_m = s_j) = \frac{\exp\{\phi_m(s_j)/T\}}{\sum_{j=1}^{N} \exp\{\phi_m(s_j)/T\}} . \tag{2}$$

This new value for X_m is always retained and the stack is always updated, *regardless* of the change in stack power. As the power $\phi_m(s_j)$ increases, the probability that the algorithm chooses $X_m = s_j$ also increases. Choices that decrease power, however, are also possible; they are just less likely.

The discussion in the next section and the results in the Appendix show that the one-step technique is formally equivalent to the Metropolis method; thus both methods simulate thermal equilibrium. The similarity of the two techniques can be seen by considering the effects of each method on a single parameter. If one were to employ the two-step Metropolis rules many times to repeatedly update X_m, then the initial value of X_m would eventually be "forgotten," and the frequency of occurrence of each possible s_j would be proportional to the probability distribution (2), which is also the distribution from which random moves are drawn using the one-step technique. Repeated application of the Metropolis rules to X_m simulates the physical effects of a heat bath surrounding X_m. In this heat bath, the "states" of X_m (e.g., molecular positions) would be distributed with equilibrium probabilities proportional to $\exp\{-E(s_j)/T\}$, where E is energy and s_j denotes some physical parameter of interest. Thus, by sampling from the probability distribution (2), we are sampling from a *local equilibrium* distribution. Sampling from these local equilibrium distributions for each X_m significantly facilitates the approach to *global equilibrium,* which must be approximated at a low temperature if simulated annealing is to yield a reasonable result.

In the literature of Monte Carlo simulations in statistical physics, the one-step technique is known as the *heat-bath method* (Rebbi, 1984; Creutz, 1984). Recently, Geman and Geman (1984) adapted this method in their application of simulated annealing to image restoration.

Although the heat-bath method is fundamentally the same as the Metropolis algorithm, for circumstances in which the probabilities (2) are readily computable, the heat-bath method can be far more efficient. We can see this in residual statics estimation by considering the practical details of implementation. As shown by Ronen and Claerbout (1985), the crosscorrelation function of two traces differs from the stack power function by only a constant. Therefore one can compute $\phi_m(s_j)$ by crosscorrelation, and the missing constant becomes irrelevant because the probability distribution is normalized. By viewing the probability distribution in equation (2) from this more familiar perspective, we can now see that it is just a crosscorrelation function that is scaled by T, exponentiated, and then scaled again so that it has unit area.

The temperature, T, determines the relative probabilities of each possible static shift; this is illustrated by the sequence of functions in Figures 1a-c. Figure 1a is a normalized crosscorrelation function computed from the Wyoming data. Conventional statics algorithms would usually choose the lag (56 ms) that yields the maximum correlation for use in some estimation scheme. Because this lag might correspond to a skipped cycle, however, the one-step method considers all possible lags, and favors each lag by the weights given in equation (2). Figure 1b shows the transformation of the crosscorrelation function of Figure 1a into the probability distribution (2), with $T = .044$. We see that the modes (peaks) of the resulting probability distribution correspond to the peaks of the crosscorrelation function. Figure 1c depicts the effects of a lower value for T, now .013. The greatest peak is now strongly accentuated relative to the others. As T goes to zero, this peak becomes a spike of unit height. Thus the zero temperature, or "quenching" algorithm, always picks the greatest peak, and would always arrive at the nearest minimum.

The spikiness of the probability distribution (2) is controlled not only by T, but also by the quality of the stacked traces against which crosscorrelation is performed. Thus, in order that the algorithm can converge to an approximation of the best possible stack, it is important that T be chosen to be high enough to allow the stack to change significantly, yet low enough so that the probability distribution (2) is sufficiently spiky when the correct solution is nearly achieved. The greatest advantages of the one-step method are incurred when the distribution (2) is composed of only a few spikes and near-zero values elsewhere. The correct value for X_m can then be located after only a few iterations (one iteration includes one attempt to change the value of each parameter). With the two-step

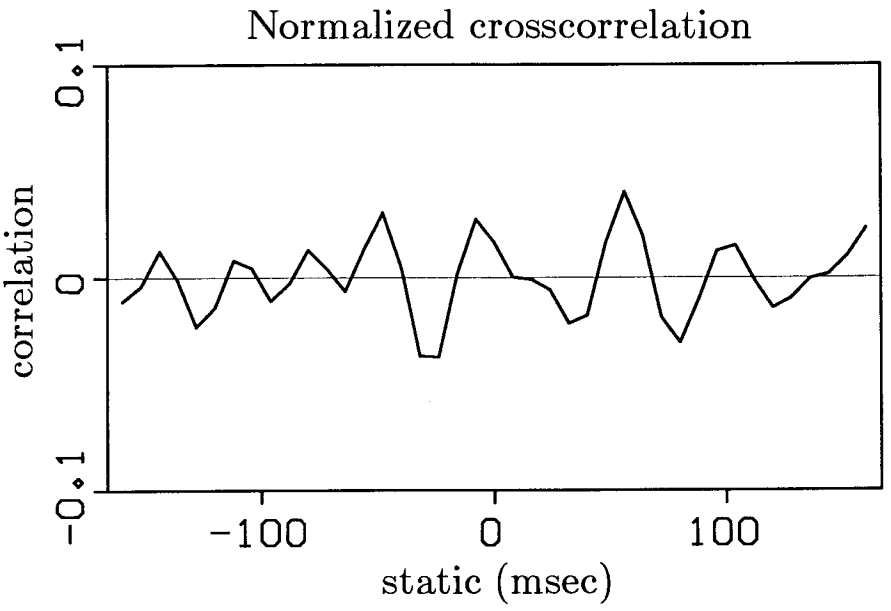

FIG. 1a. A normalized crosscorrelation function.

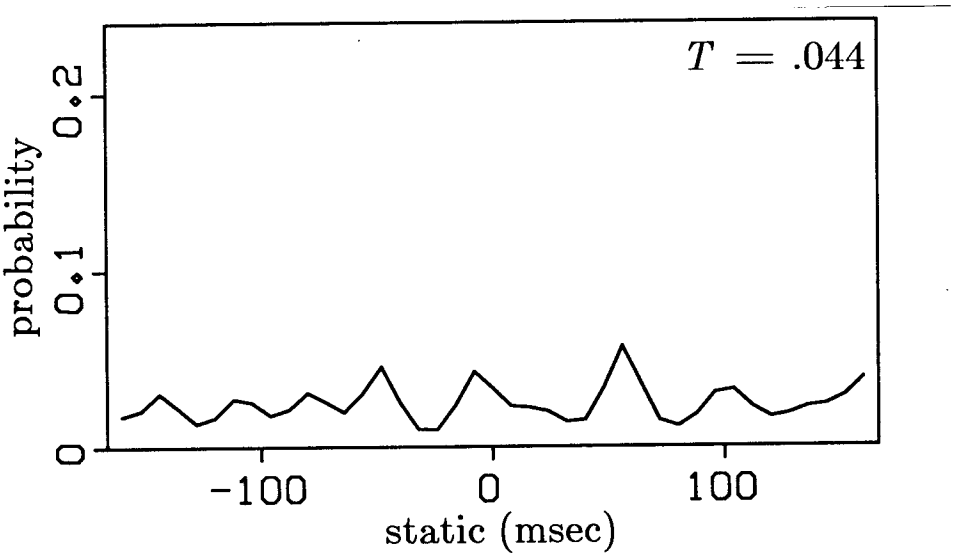

FIG. 1b. Transformation of the crosscorrelation function in Figure 1a into a probability distribution using equation (2), with ϕ_m taken to be crosscorrelation, and $T = .044$. Note the general correspondence of the peaks and troughs in Figures 1a and 1b.

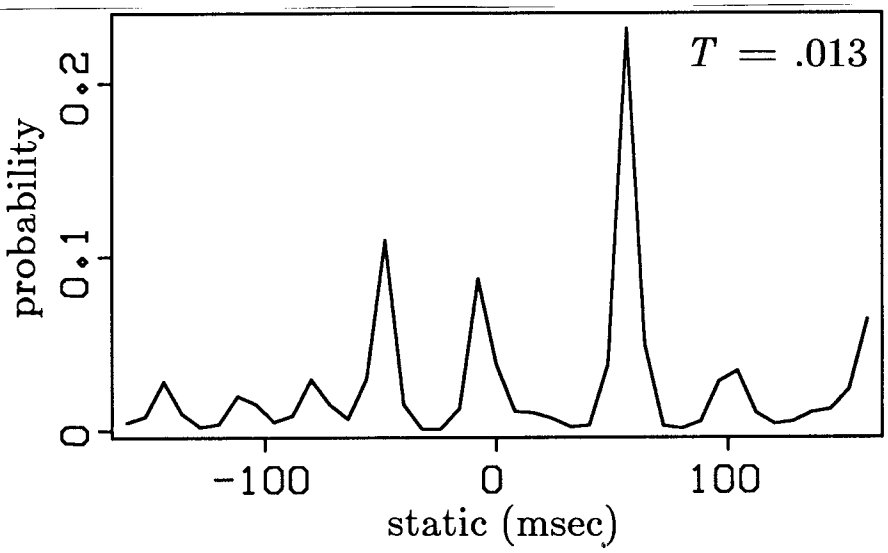

FIG. 1c. Transformation of the crosscorrelation function in Figure 1a into a probability distribution using equation (2), with ϕ_m taken to be crosscorrelation, and $T = .013$. Compared to Figure 1b, the tallest peak is now strongly accentuated relative to the other peaks; thus the lag (56 ms) corresponding to that peak is strongly favored when random samples are drawn from this distribution.

technique, however, guessing and accepting the correct value can take several hundred iterations or more.

Although it might appear that each calculation of the probability distribution (2) would require about N times the computation that an equivalent Metropolis move requires, modern array processors allow the crosscorrelation functions to be calculated rapidly at a cost that is a modest function of the number of lags. Thus the additional information that is contained in the crosscorrelation function more than compensates for the cost of obtaining it.

Before illustrating the results obtained using this method, I will first briefly introduce some of the theoretical properties of the technique.

THEORETICAL RESULTS

In the remainder of this paper, the one-step method is referred to as the Monte Carlo statics estimation algorithm. In this section, I present the mathematical theory underlying this technique. Similar theory has long been applied to problems in statistical physics, and the reader seeking general reviews

is referred to the two volumes by Binder (1979, 1984), the book by Hammersley and Handscomb (1964), and the article by Fosdick (1963).

Theoretical results are presented here for the Monte Carlo statics algorithm when it is run at constant temperature. Similar results for the Metropolis algorithm can be found in the original paper by Metropolis et al. (1953) and the references mentioned above.

The following notation is used. There are assumed to be M parameters $\mathbf{X} = \{X_1, X_2, \cdots, X_M\}$; these are the unknown statics for each shot and receiver. X_i is taken to be a random variable; x_i is its value. Each parameter may assume only one of N distinct values; thus there are N possible values for each x_i. A *state* $\mathbf{x}_i$ (note boldface) is any combination of values assumed by the entire set of parameters $\mathbf{X}$. Because there are M parameters that may each assume one of N values, there are a total of N^M distinct states of the system. N is typically on the order of 50, and M can be several hundred or more.

Markov chains

The basic objective of the Monte Carlo statics algorithm is to generate states $\mathbf{x}_j$ from the probability distribution

$$\Pi_j \equiv P(\mathbf{X}=\mathbf{x}_j) = \frac{\exp\{-E(\mathbf{x}_j)/T\}}{\sum_{j \in A} \exp\{-E(\mathbf{x}_j)/T\}}, \tag{3}$$

where E is the negative stack power, and $j \in A = \{1, 2, \ldots, N^M\}$. If $\mathbf{x}_j$ were the configuration of a physical system, E its energy, and T scaled by Boltzmann's constant, then equation (3) would be the Gibbs (or canonical) distribution of statistical mechanics. Justification of the use of the Gibbs distribution for residual statics estimation is given in Rothman (1984); here we restrict the discussion to an explication of technique.

The problem, then, is to generate states $\mathbf{x}_j$ with probability Π_j. This generation is called *importance sampling* (Fosdick, 1963; Hammersley and Handscomb, 1964). Note that the $\{\Pi_j\}$ cannot be explicitly computed because there are N^M terms in the denominator. If we can, however, generate states with the distribution (3) when $T << E(\mathbf{x})$, then states that yield the greatest stack power can be selectively favored due to the exponential weighting, and approximations of the best stack can be generated. The generation of random states from (3) is performed by construction of a *Markov chain.*

A Markov chain is a sequence of states in which the probability of each newly generated state depends only on the previous state, not on any others. The Monte Carlo statics algorithm is a Markov chain in which each state is a new set of timing delays for each shot and receiver. Consider these states to evolve over a sequence of iterations indexed by t. The notion of immediate dependence is formally expressed by

$$P(\mathbf{X}_t = \mathbf{x}_j \mid \mathbf{X}_{t-1} = \mathbf{x}_{i_{t-1}}, \cdots, \mathbf{X}_2 = \mathbf{x}_{i_2}, \mathbf{X}_1 = \mathbf{x}_{i_1}) = P(\mathbf{X}_t = \mathbf{x}_j \mid \mathbf{X}_{t-1} = \mathbf{x}_{i_{t-1}}) .$$

The probability of going from any state $\mathbf{x}_i$ to any state $\mathbf{x}_j$ in one iteration is stationary in time, and is given by the *transition probability*

$$p_{ij} = P(\mathbf{x}_i \rightarrow \mathbf{x}_j) = P(\mathbf{X}_t = \mathbf{x}_j \mid \mathbf{X}_{t-1} = \mathbf{x}_i) . \tag{4}$$

Most important developments in the theory of Markov chains concern the evolution of a chain over time. Generalizing equation (4), let $p_{ij}^{(n)}$ be the probability of transition from $\mathbf{x}_i$ to $\mathbf{x}_j$ in n iterations:

$$p_{ij}^{(n)} = P(\mathbf{X}_t = \mathbf{x}_j \mid \mathbf{X}_{t-n} = \mathbf{x}_i) .$$

The results in the Appendix, summarized below, show that as $n \rightarrow \infty$ the probability of the Monte Carlo statics algorithm going from $\mathbf{x}_i$ to $\mathbf{x}_j$ is independent of the initial state $\mathbf{x}_i$.

Results: The steady state

It is shown in the Appendix that the Monte Carlo statics algorithm exhibits the limit

$$\Pi_j = \lim_{n \rightarrow \infty} p_{ij}^{(n)} . \tag{5}$$

$\{\Pi_j\}$ is the Gibbs distribution (3); it is called the *equilibrium,* or *steady-state,* distribution of the Markov chain. The term equilibrium is used here as it is in physics: it describes the behavior of the system after the system has "forgotten" its initial state. Because the low-energy (high-stack-power) states are exponentially favored by the distribution $\{\Pi_j\}$, equilibrium at low T approximates the global minimum. Thus the algorithm is theoretically sound if it is run at low T for a sufficiently large number of iterations. Yet this procedure might be, of course, far from practical. How, then, does one reconcile the theory with practice?

Reconciliation begins by experimentation. In principle, it is easy to see why the algorithm should work if T is lowered sufficiently slowly so that equilibrium

is attained for every T—this is how Kirkpatrick et al. (1983) originally proposed the concept of simulated annealing. The experimental results illustrated in the next section, however, show that the choice of a sufficiently low constant temperature is indeed practical: we need only wait for equilibrium at this one temperature. The solution is obtained by dropping immediately from a high T to a low T, and then maintaining this low T for almost 2000 iterations. As described in the next section, the exact specification of the low T is determined by previous experimentation. Because equilibrium is attained at low T, the algorithm locates a deep (if not the deepest) minimum. The success of this approach depends, of course, on how rapidly equilibrium can be reached. In general, the approach to equilibrium slows down as the temperature decreases and/or the number of deep minima increases.

EXPERIMENTAL RESULTS

Conventional methods of estimating statics (e.g., Wiggins et al., 1976; Taner et al., 1974) produce favorable results when static shifts are small enough and signal quality is good enough so that cycle-skipping is not a problem. The field data example that was introduced briefly at the outset of this paper exhibits an unusually severe statics problem, however, and is thus an excellent test case for Monte Carlo statics estimation. This section details the results of that test.

The data

The seismic section in Figure 2a is a 24-fold "raw" or "brute" stack; this stack was produced without any statics corrections. Figure 2b shows two representative common-midpoint gathers. The data were collected in the Wyoming Overthrust belt; a 48-trace, split-spread cable was used. The source was Vibroseis, with an 8-55 Hz sweep. The data have undergone the following processing steps: (1) predictive deconvolution; (2) bandpass filtering, from 8 to 35 Hz; (3) normal-moveout (NMO) corrections; and (4) automatic gain control. Stacking velocities were laterally invariant. No field statics corrections were made. The cablelength extends over approximately 100 stacked traces (3350 m), which is about 60% of the section. Both ends of the line exhibit the usual roll-on and roll-off, so the first and last 24 stacked traces are less than 24-fold.

Although it may not be immediately obvious that these data suffer from a statics problem, supporting evidence exists. As discussed earlier, a strong, shallow reflection in the stack (Figure 2a) appears to arch downward from about 100

ms at the center of the section to about 600 ms at both sides. We shall see later that this is the reflection from the base of low-velocity fill. The gathers (Figure 2b) reveal the unusual character of the reflection at approximately 3.5 s. Although the same velocity function was used to correct for normal moveout in both gathers, CMP 34 shows the event dipping upward with offset, whereas CMP 64 (only one-third of the cablelength away) shows the event dipping downward. Furthermore, in the stack, this event exhibits gross discontinuities that mirror the shallow, arched reflector—this consistency with depth is a good indication of near-surface velocity variations.

One could construct a model of the presumed shot and receiver statics, and then, if necessary, input this model as the initial guess in a conventional, linearized algorithm. Johnson et al. (1983) were able to manually construct a model based solely on a geologic interpretation of these data; from the model they then produced a successful stack. A comparable result, obtained using only the Monte Carlo statics algorithm, is discussed below.

Processing parameters

Statics are estimated from normalized crosscorrelations of the data between 2.9 and 3.9 s, thus concentrating on the prominent reflector at approximately 3.5 s. Static shifts are constrained to fall within ± 160 ms, in 8 ms increments; each crosscorrelation is accordingly performed over 41 lags. Shot statics are estimated independently of receiver statics. There are 85 source locations and 90 receiver locations, resulting in a total of 175 free parameters. Because each parameter may assume any of 41 values, there are 41^{175} possible solutions.

Choices for the temperature parameter, T, inevitably depend on the data. In this example, the primary objective is to mend the discontinuities on the left side of the section between traces 20 and 80 (see Figure 2a). Because statics appear to have caused a very large (approximately 200 ms) break, we need more than just an incremental change in the apparent structure in the stack. Thus the initial choice of T must be high enough so that the statics are given the freedom to mend the deep reflector. From a mathematical point of view, this requirement is equivalent to saying that the input stack (all statics equal zero) is already located near a sub-optimal local minimum, and that the initial T must be high enough so that this minimum can be escaped easily. The input stack is recognized as being near a local minimum because the reflections already stack in well in isolated regions of the stack. In these zones, the input stack is *locally*

satisfactory. When one views the stack *globally,* however, one sees that considerable changes are necessary to make the deep reflector continuous.

Once the algorithm departs from the region near the initial minimum, T can then be quickly lowered to a minimum temperature, $T_{\min}$. $T_{\min}$ is chosen by previous experimentation: it must be high enough so that local minima are escaped, but low enough so that convergence can occur only in the deepest minimum, or in minima that are nearly as deep. It is unlikely that the algorithm will return to the initial location (the input stack); because the method generates solutions $\mathbf{x}$ with a frequency proportional to $\exp\{-E(\mathbf{x})/T_{\min}\}$, deep minima are more probable than the more shallow initial position.

Our objective, then, is to approach equilibrium as rapidly as we can at a temperature that is low enough to strongly favor the global solution, yet high enough to ignore the overwhelming number of shallow minima. This goal assumes a simple model in which the objective function contains a few very deep minima among a multitude of shallow minima. Thus far this appears to be a reasonable model for data with large statics and moderate noise contamination, which is the case in this example.

One might think that it is inefficient to begin with a high T: statics are chosen that significantly decrease the stack power, leading to the loss of all structure in the data. However, if one were to begin instead at $T_{\min}$, the approach to equilibrium would be far slower because much time would be expended attempting to climb out of the initial minimum. (In practice, this ascent may appear impossible.) By starting at a high temperature, the algorithm begins at one of the highest locations on the objective function. Although we still risk entrapment in a local minimum, the probability of entrapment is significantly lowered by our not beginning near a local minimum.

The temperature function chosen in the following example has the form

$$T_k = \begin{cases} \alpha^k T_0 , & \alpha^k T_0 > T_{\min} \\ T_{\min} , & \text{otherwise} \end{cases}$$

where T_k is the temperature for the k th iteration (one iteration includes one attempt to change the value of each parameter), $\alpha = .99$, $T_0 = .045$, and $T_{\min} = .0265$. For this choice of parameters, $T_k = T_{\min}$ after only 53 iterations. T_0 was chosen to insure that the structure in the input stack would be

destroyed quickly. In practice, the specification of T_0 requires just a few (if any) quick tests of only a few iterations each. The choice of α is fairly arbitrary. Choosing $T_{\min}$, however, is not so simple; a workable value is determined only after a fair amount of experimentation. Before the result discussed below was obtained, it was necessary to run 4 full-length test runs (of about 1000 iterations each) with different values for $T_{\min}$. If $T_{\min}$ is too high, then convergence will not occur within a reasonable number of iterations; if $T_{\min}$ is too low, then the algorithm is likely to converge to a horrendous and obvious local minimum. It appears thus far that the algorithm's sensitivity to $T_{\min}$ is substantial: differences of just a few per cent can lead to very dissimilar results. Fortunately, however, systematic experimentation with a range of values yields the correct $T_{\min}$ without undue difficulty.

Results

Figures 3a-e depict the progress of the statics estimation algorithm at different points in the iterative process. Each of these figures is a 24-fold stack that is performed after corrections are made with the current estimate of the statics. Figure 3a displays the stack after 5 iterations; we see that $T_0 = .045$ leads to immediate obliteration of all spatial coherence in the stack. By iteration 53, $T = T_{\min}$, but after 1000 iterations (Figure 3b) the stack still exhibits no improvement over the result in Figure 3a. By iteration 1125, however, convergence begins; this stage is illustrated in Figure 3c. The algorithm then rapidly descends into a minimum, as is evident in Figure 3d, the stack after 1250 iterations. The algorithm was run for 2000 iterations. The stack with the greatest power was achieved in iteration 1835, and is shown in Figure 3e. It should be compared with Figure 2a, the stack of the input data. We see now that the deep reflector has not only become continuous across most of the section, but also that the statics corrections have revealed steeply dipping events in the more shallow data. There are two apparent imperfections in Figure 3e, however. First, the dip of the deepest reflector appears to have reversed, and second, the far right side of the same reflector appears to be artificially discontinuous. Both of these shortcomings will be addressed soon.

Figure 4 summarizes the results of the 2000 iterations in one graph; stack power is plotted as a function of iteration. Stack power is computed only within the computation window (2.9-3.9 s), and the power of the input is normalized to one. Power quickly decreases to about .5, and does not begin to rise until after

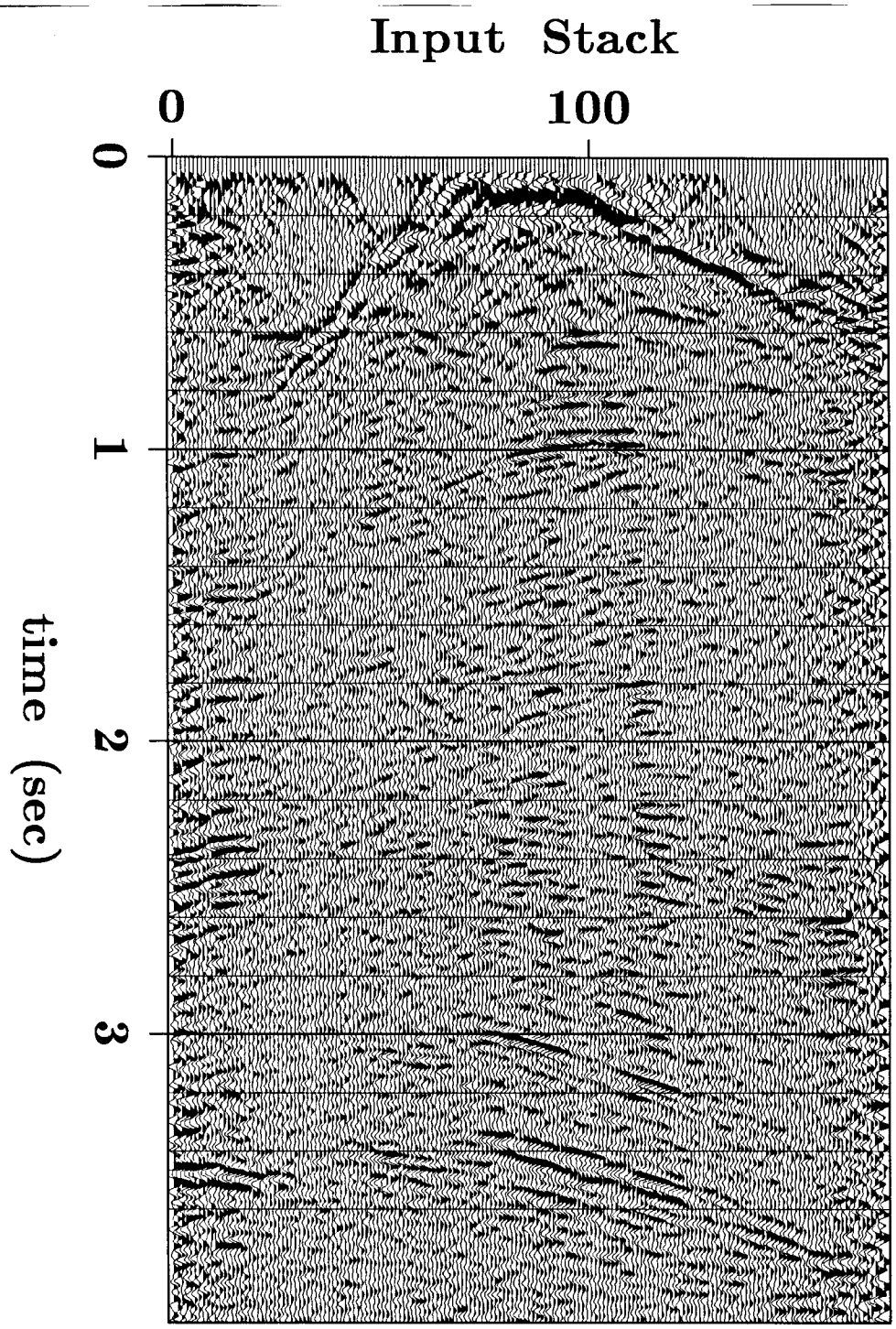

FIG. 2a. 24-fold stack of data from the Wyoming Overthrust belt; the stack is performed prior to statics estimation. One time-variable stacking velocity function was used for the entire line. The data used for residual statics estimation are between 2.9 and 3.9 s. The strong reflections at the near surface are roughly indicative of the near-surface velocity variations. The first and last 24 traces are underfold due to the usual roll-on and roll-off.

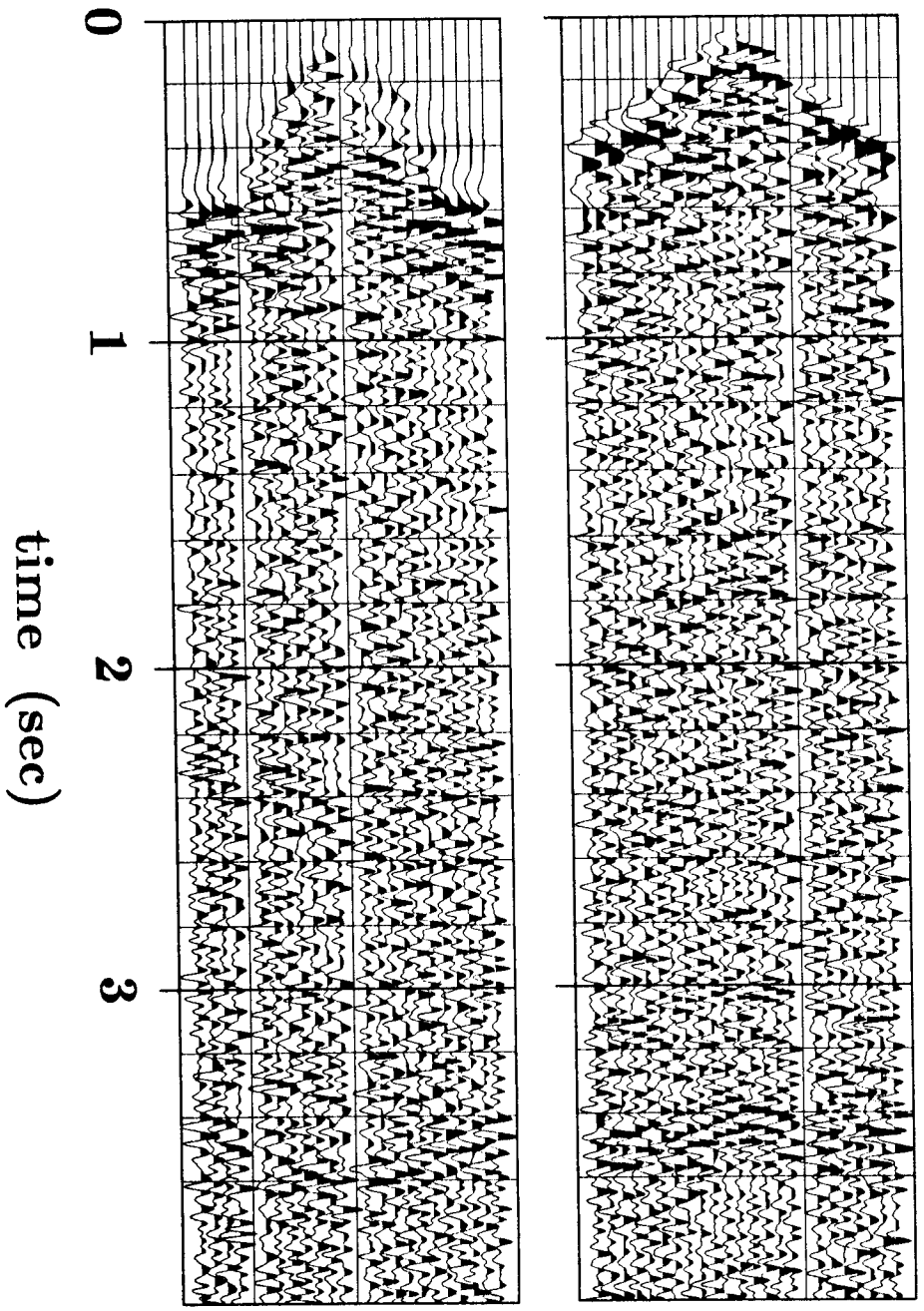

FIG. 2b. NMO-corrected common-midpoint gathers 34 and 64; the gathers are displayed without statics corrections. Offset increases in each plot from the center outward. Both gathers are corrected with the same velocity function. The near-surface velocity anomalies have produced dipping structure in events that should be flat; this is most evident in the data near 3.5 s. In gather 34, dip appears to bend upward with offset. In gather 64, just one-third of a cablelength down the line, dip now appears to bend downward with offset.

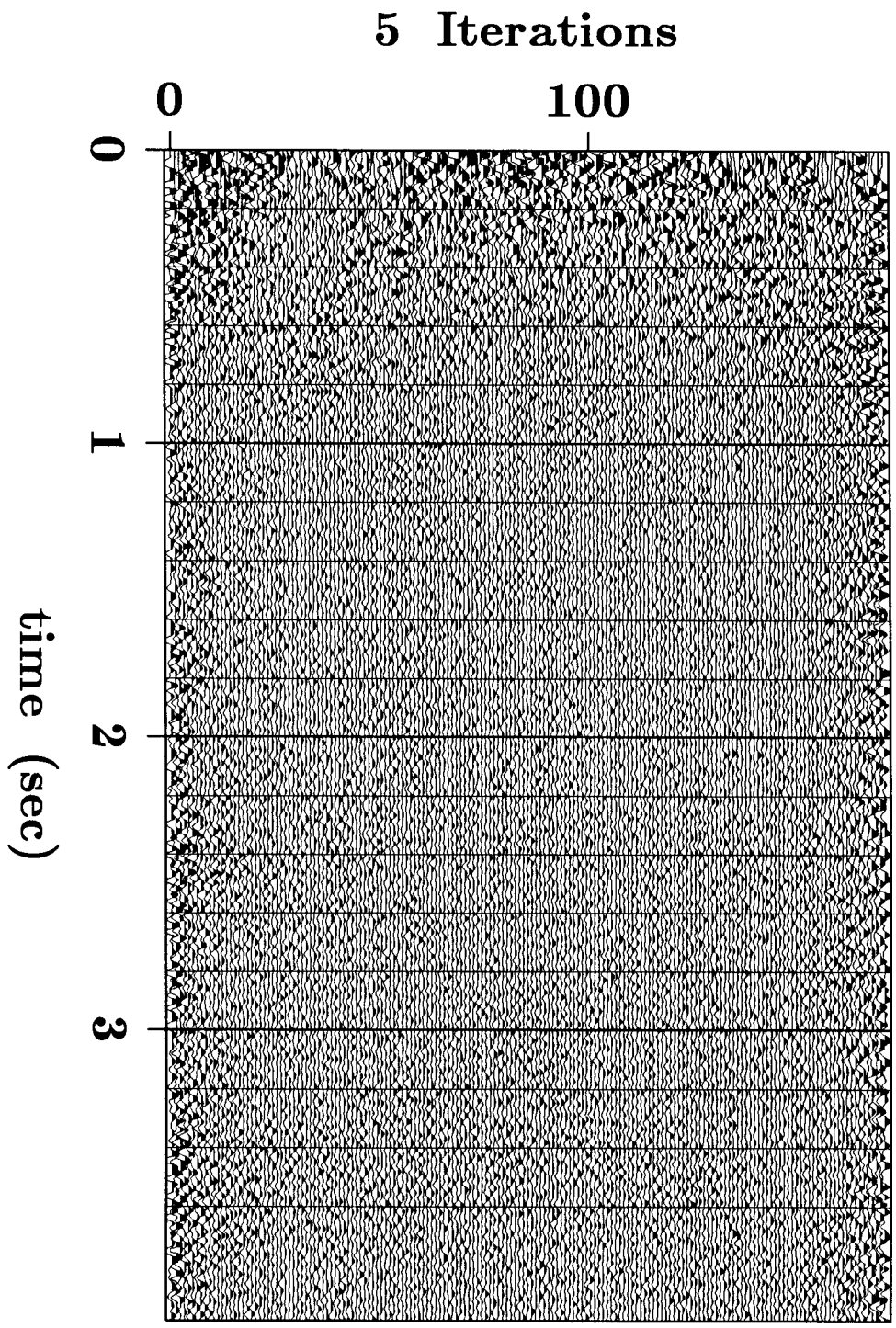

FIG. 3a. Stack after 5 iterations of the statics estimation algorithm. T_0 was chosen high enough so that all reflection events are now obliterated. This choice removes all tendencies for the algorithm to make only incremental improvements in the stack, as would be expected from a linearized technique. Amplitudes are greater at the sides because of the low fold at the ends of the line.

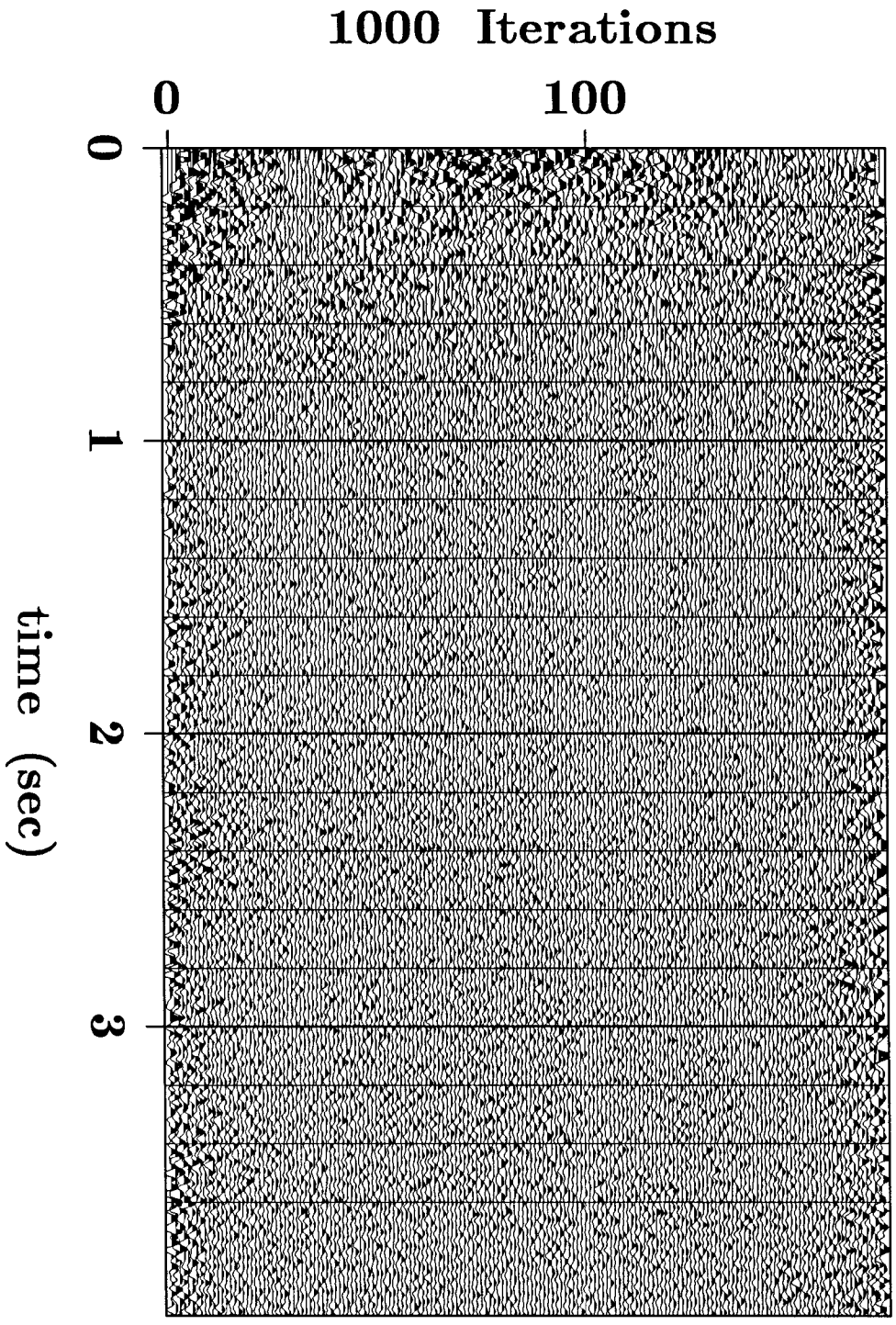

FIG. 3b. Stack after 1000 iterations of the statics estimation algorithm. $T = T_{\min}$ from iteration 53 onward. No appreciable differences are evident between this stack and the result after 5 iterations (Figure 3a).

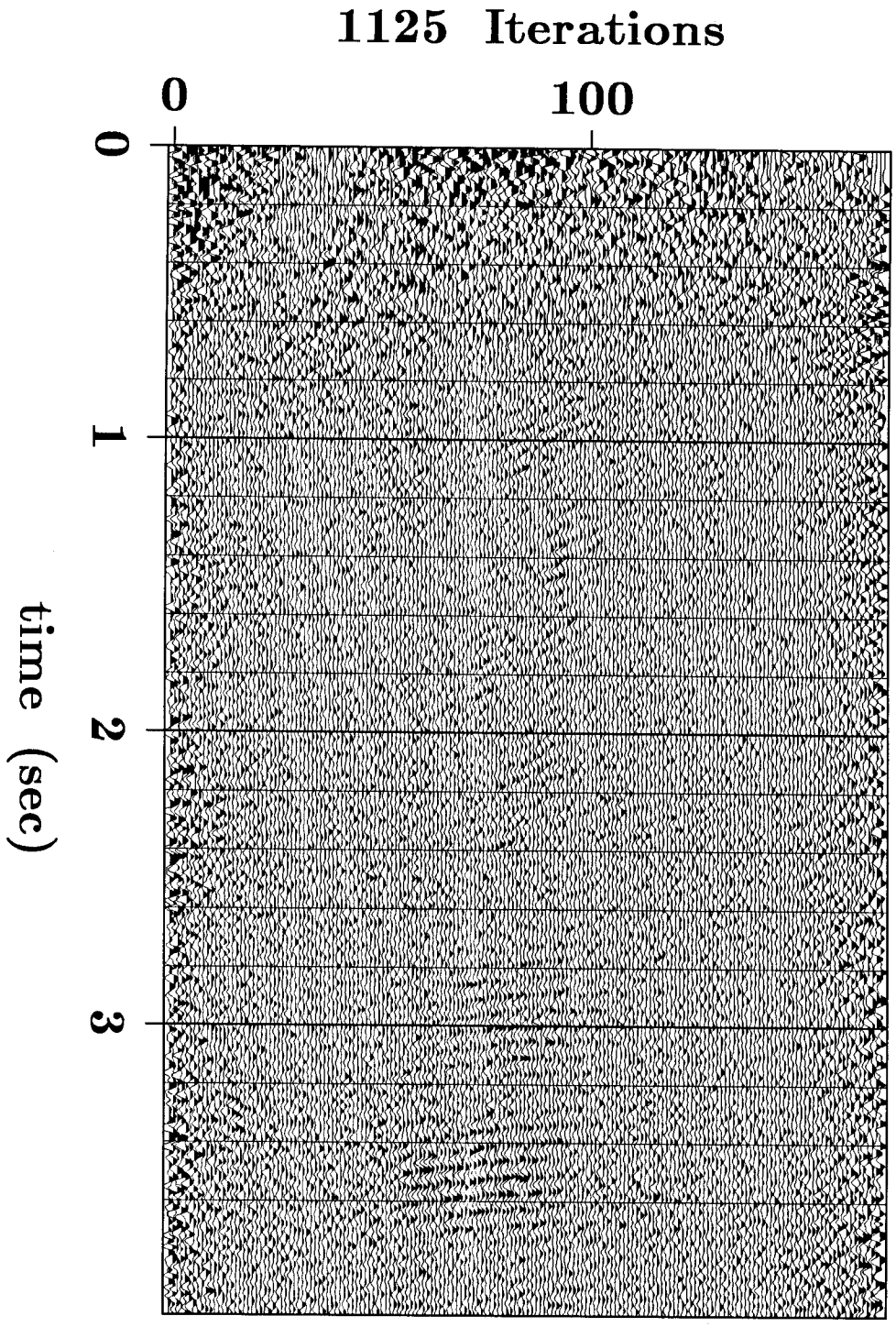

FIG. 3c. Stack after 1125 iterations of the statics estimation algorithm. The faint spatial coherence in the middle of the section shows that convergence is now beginning.

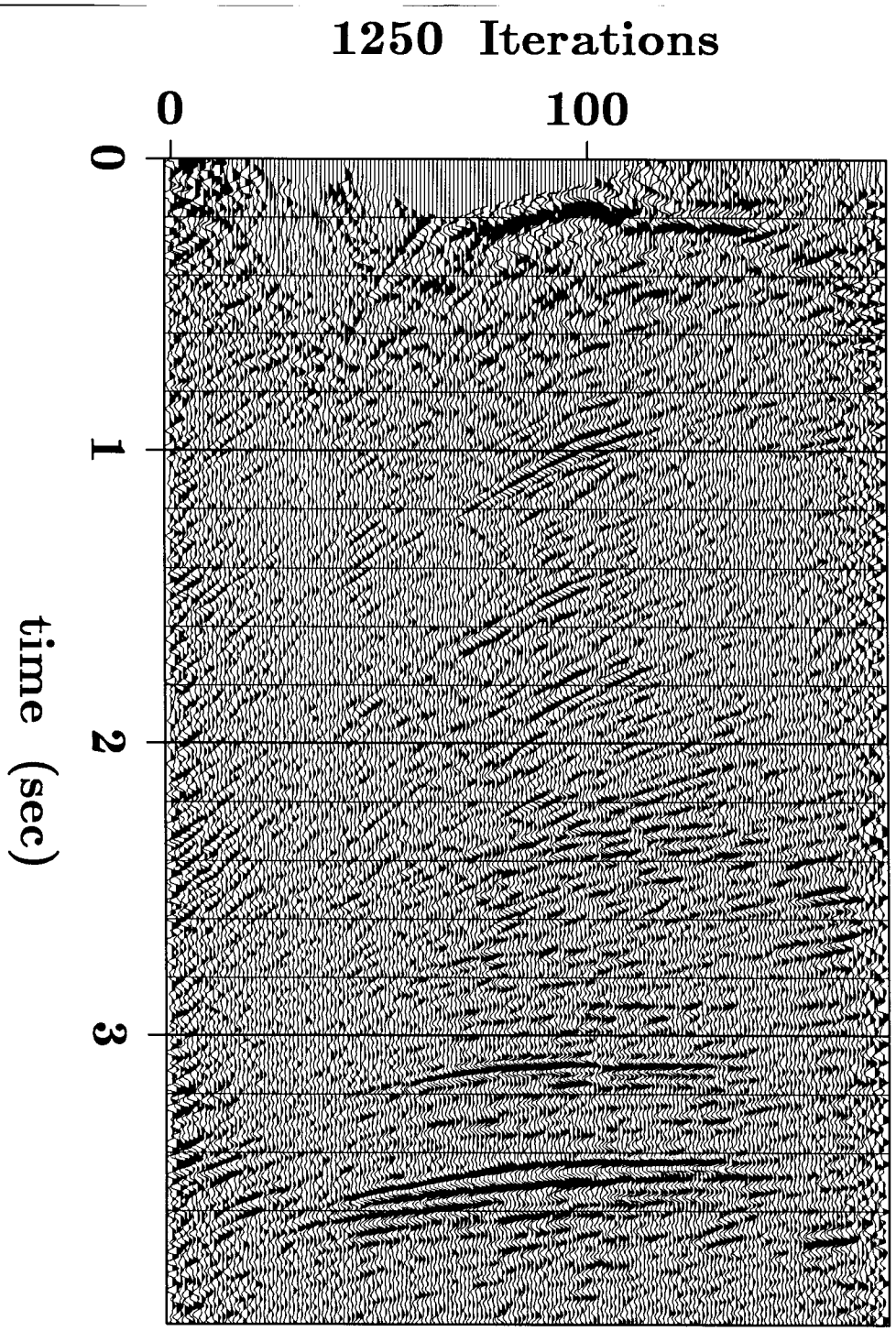

FIG. 3d. Stack after 1250 iterations of the statics estimation algorithm. Convergence is now almost complete; only the ends of the line have not converged.

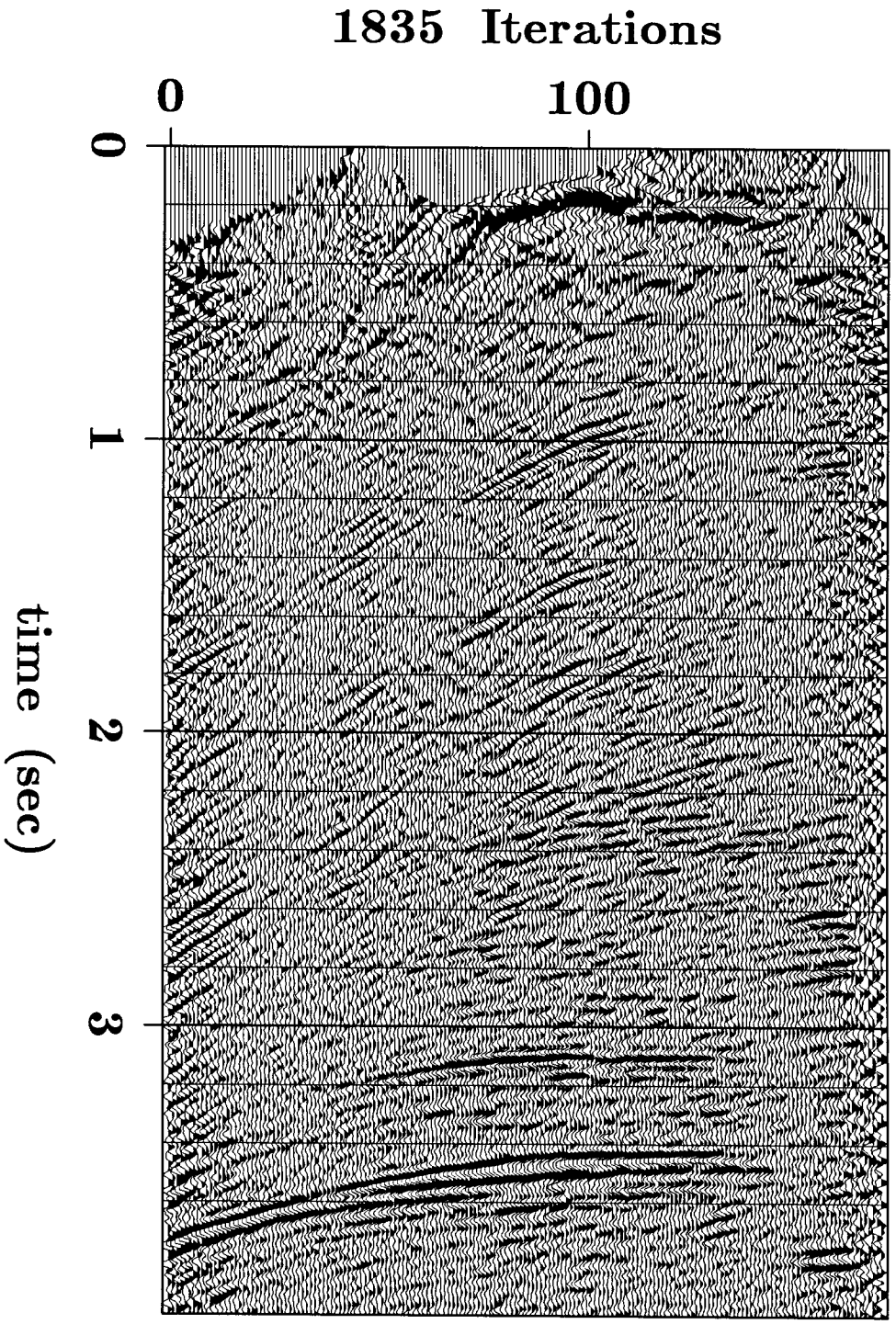

FIG. 3e. Stack after 1835 iterations of the statics estimation algorithm. Of the 2000 iterations, this stack yielded the greatest stack power. This stack should be compared with the stack of the input data in Figure 2a. The deep reflection is now continuous throughout much of the section. Moreover, this statics solution has uncovered steeply dipping structure in the first 3 seconds of the data. Two imperfections are apparent, however. First, the dip of the deep reflector appears to have reversed, and second, the extreme right end of the line shows artificially discontinuous reflections.

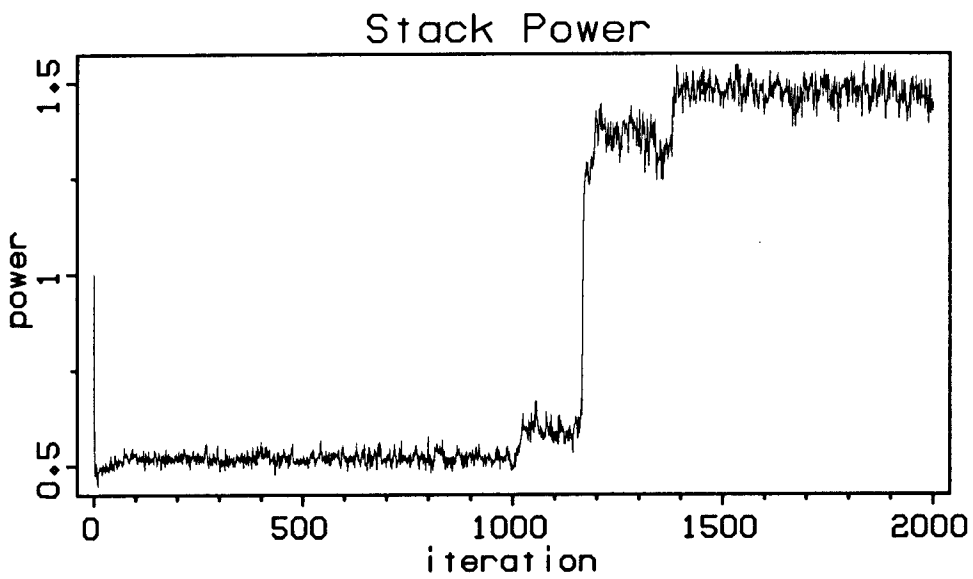

FIG. 4. Stack power as a function of iteration number, for the example illustrated in Figures 3a-e. The input stack power is normalized to 1. Power is computed within the estimation window, which extends from 2.9 to 3.9 s. Power initially decreases quickly to about .5. T decreases from .045 to .0265 during the first 53 iterations; thereafter it remains constant. Convergence begins at approximately iteration 1000. A sharp increase in power occurs near iteration 1150; this rapid change is analogous to crystallization. Global convergence occurs after about 1400 iterations. The remaining 600 iterations yield essentially similar stacks except for the behavior at the far right. The power of the stack displayed in Figure 3e is 1.553, which is the greatest power of the 2000 iterations.

1000 iterations. Beginning at approximately iteration 1150 there is a sharp increase in power; convergence finally occurs by about iteration 1400. The stacks produced by the remaining iterations were roughly equivalent except for the behavior at the far right, which never became stable. The maximum stack power occurred in iteration 1835 (Figure 3e), where the power is 1.553.

Figure 5 shows the same two gathers in Figure 2b; statics corrections from iteration 1835 have now been performed. In gather 34, the upward dips at far offsets have now been flattened. Gather 64 has had its downward dips also leveled.

Figure 6 shows the estimates of the shot and receiver statics; these estimates were produced by iteration 1835, the best of the 2000 iterations. The static shifts extend the full range within ± 160 ms, and most of the variations are smooth. Examination of the stack in Figure 3e in conjunction with the graphs of shot and

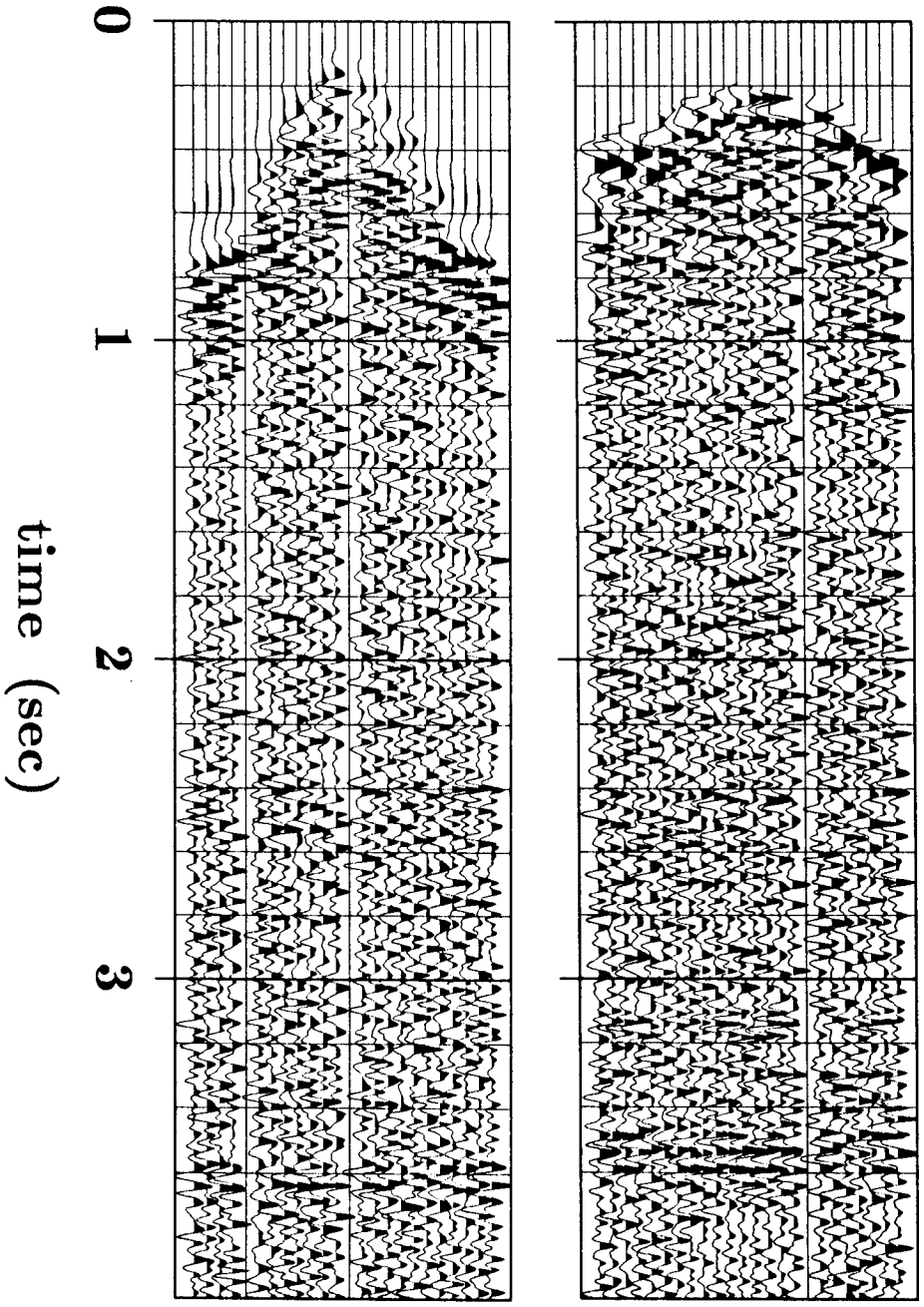

FIG. 5. The same two NMO-corrected common-midpoint gathers of Figure 2b, now shown after statics corrections from iteration 1835 have been made. Both show substantial alignment after application of the statics solution.

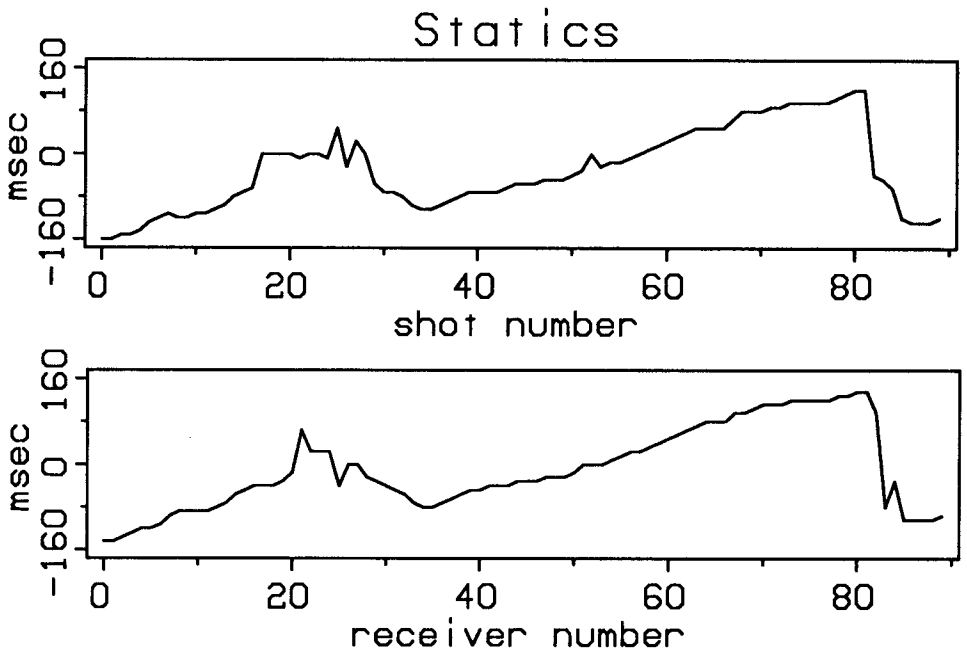

FIG. 6. The shot and receiver statics output by iteration 1835. Shots 17-20 and shot 24 were skipped; these are plotted as zeroes. At the right end, statics are as large as 160 ms, which was the upper bound. Examination of the stack in Figure 3e in conjunction with these graphs suggests that the "true" statics at the right end are greater than 160 ms, thus resulting in the artificial discontinuity in the deep reflector in Figure 3e.

receiver statics reveals the cause of the artificial discontinuity in the deep reflector at the right end. The "true" statics at the right end of the line are apparently greater than the upper bound, which was 160 ms. The algorithm thus found the best available solution, which unfortunately resulted in a large mismatch at the end of the line. The magnitude of this mistake is roughly indicative of the size of the statics being considered for the solution.

This error at the right side can, of course, be corrected. The Monte Carlo statics algorithm was run again; this time the statics of Figure 6 were the starting guess. To allow the estimation of larger statics, the upper bound was doubled to 320 ms, and the estimation window was also doubled to extend from 2 to 4 s. Temperature remained at $T = T_{\min} = .0265$ throughout. After 250 iterations, the stack in Figure 7 was produced. The deep reflector is now continuous throughout the section, and the more shallow reflections also show greater continuity on the right side. Figure 8 shows the statics that were used to produce the stack in Figure 7. Statics are now as great as 256 ms; this high level supports

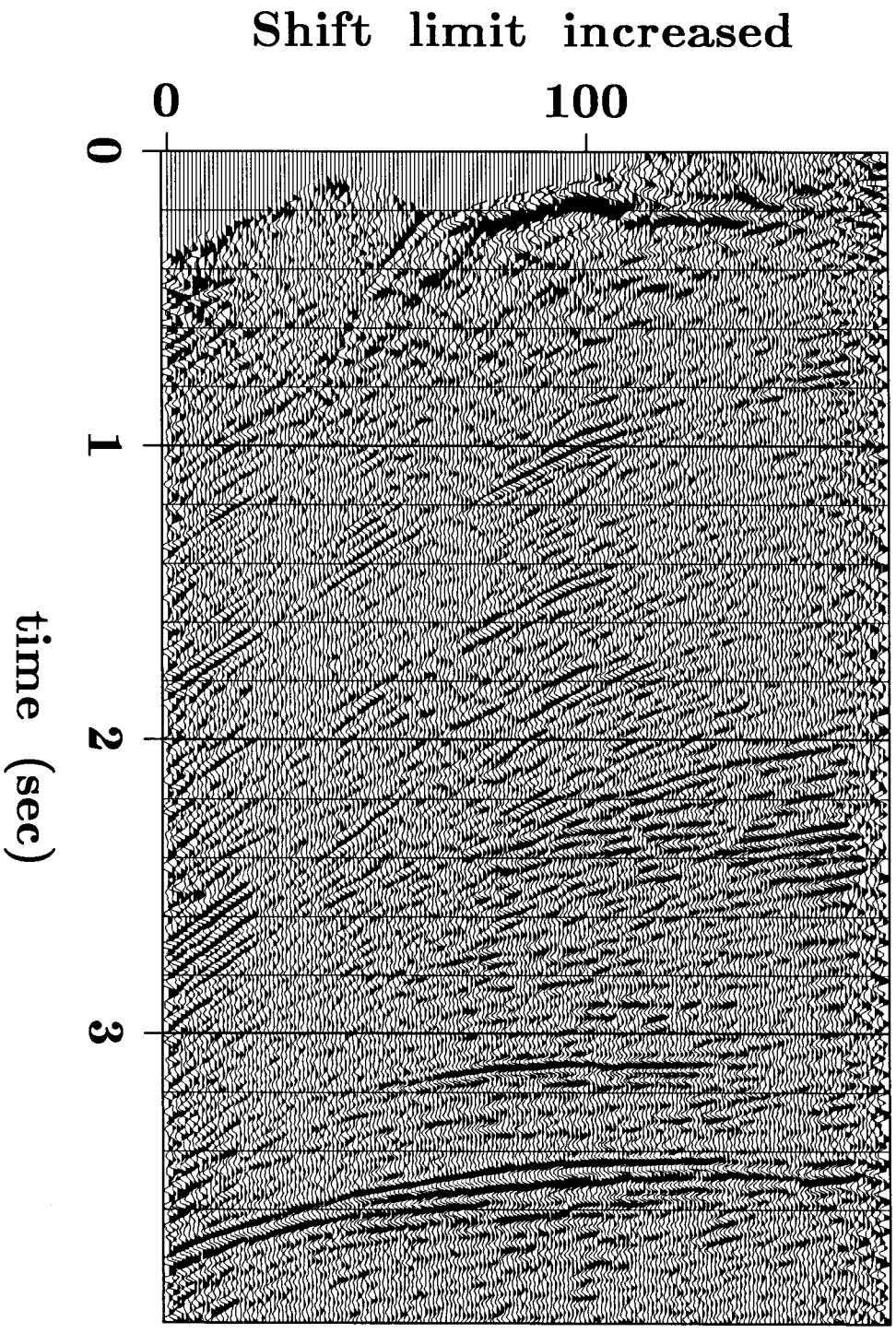

FIG. 7. Stack produced after the Monte Carlo statics estimation algorithm was run again, this time using the statics of Figure 6 as the starting guess. To allow the estimation of larger statics, the upper bound was doubled to 320 ms, and the estimation window was also doubled to extend from 2 to 4 s. This stack was produced after 250 iterations, in which $T = T_{\min} = .0265$ throughout. Compare with the stack in Figure 3e: the deep reflector is now continuous across the entire line, and the more shallow reflections are stronger, especially in the region on the right between 2.0 and 2.5 s. The stack power is now 1.687.

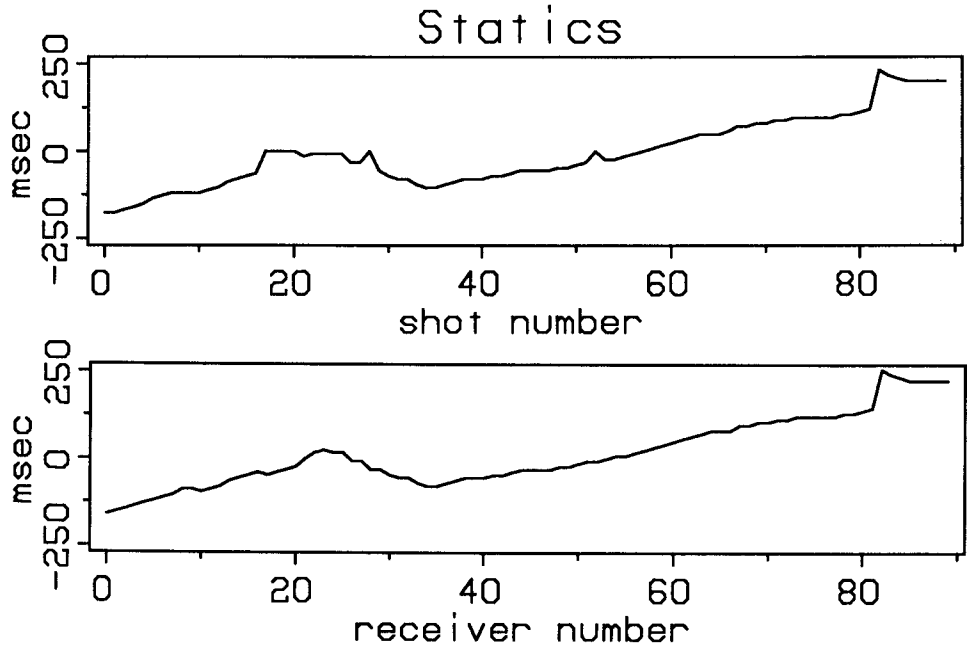

FIG. 8. The shot and receiver statics used to produce the stack in Figure 7. Statics are now as great as 256 ms; this high level supports the conclusion that the previous upper bound of 160 ms had been too small.

the conclusion that the upper bound had previously been too small. The stack power calculated for the estimation window is now 1.687.

The dip reversal on the deep reflector must be suspect. Statics solutions are highly *non-unique;* that is, different solutions can yield the same stack power. The most obvious non-unique component is a simple shift (up or down) of all shot and receiver statics by the same amount. This shift changes the timing of events, but the stack power remains the same. A linear trend in the shot and receiver statics is also not uniquely determined (see Taner et al., 1974; Wiggins et al., 1976; or the elegantly simple derivation by Ronen and Claerbout, 1985). Thus, from the viewpoint of statics estimation, the dip on a common-midpoint stack is completely unresolved by the data. The contribution from the linear trend can be removed, however, by fitting a regression line to the average of the shot and receiver statics and then subtracting it (Ronen and Claerbout, 1985). Figure 9 shows the stack produced by subtraction of the linear trend, and Figure 10 shows the corresponding shot and receiver statics. The prevailing dip of the deep reflector now matches the input (Figure 2a) well. It is interesting to note that the statics in Figure 10 now extend only up to 120 ms. Also, the shape of

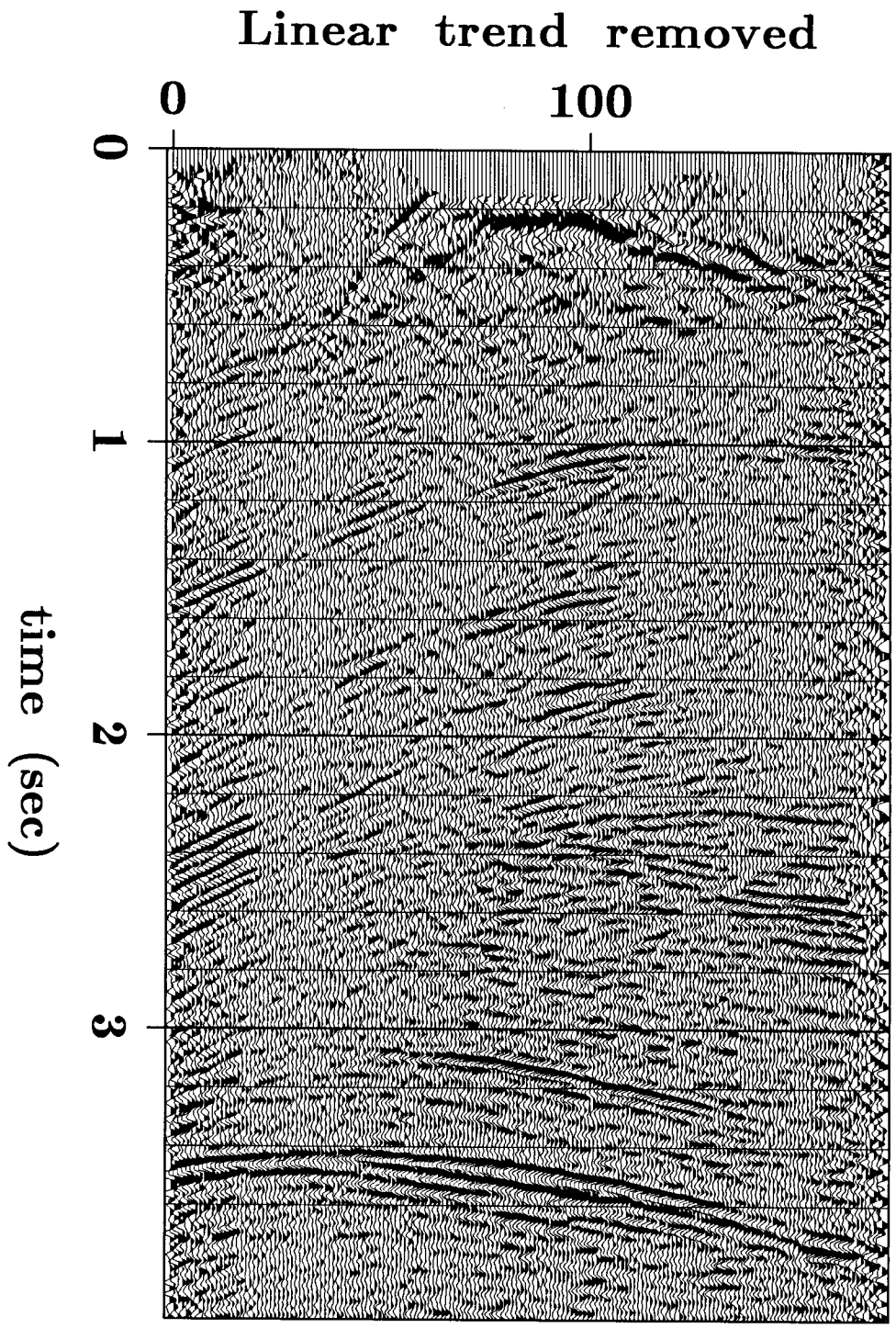

FIG. 9. Stack produced after the linear trend in the shot and receiver statics of Figure 8 was subtracted. Because the linear trend in the statics is fully unresolved by the data, *any* prevailing dip on the stack is equally as good as any other dip. Elimination of the linear trend thus discards this nonuniquely determined component of the solution. The prevailing dip on the stack now matches the dip on the input stack (Figure 2a) well. This is the final solution: note the considerable differences in quality between this stack and Figure 2a.

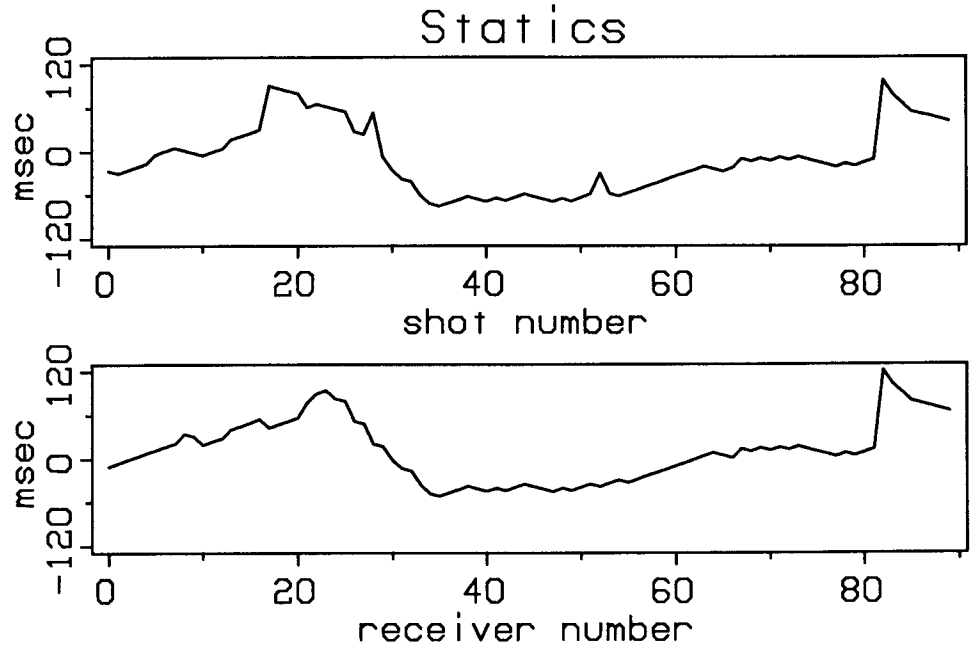

FIG. 10. The statics of Figure 8, now shown after subtraction of the linear trend. Note that the shape of both the shot and receiver statics is roughly opposite that of the near-surface reflector seen on the input; this shape indicates that this reflection occurred at the base of low-velocity fill.

the statics is roughly opposite that of the near-surface reflector on the input (Figure 2a); this shape indicates that this reflection occurred at the base of low-velocity fill.

If one expects that the objective function, the negative stack power, contains few or no local minima, then one can sequentially choose for each shot and receiver the static shifts that yield the greatest crosscorrelation coefficient. This process is optimization by *iterative improvement,* or, equivalently, Monte Carlo statics estimation with $T = 0$. A test of iterative improvement was run on the data of Figure 2a; the test used processing parameters (except for T) identical to those used to generate the stack in Figure 3e. The result is shown in Figure 11; convergence was attained after only 13 iterations. All reflections have been enhanced, but the poor stack in the region between traces 20 and 80 still remains. This "cycle-skipping" is most evident at about 3.5 s, where the reflections should be laterally continuous—compare with Figures 3e, 7, and 9. Because the statics for these data are so large, many local minima exist; thus iterative improvement fails because it finds only the nearest minimum. The stack power for this result is 1.395.

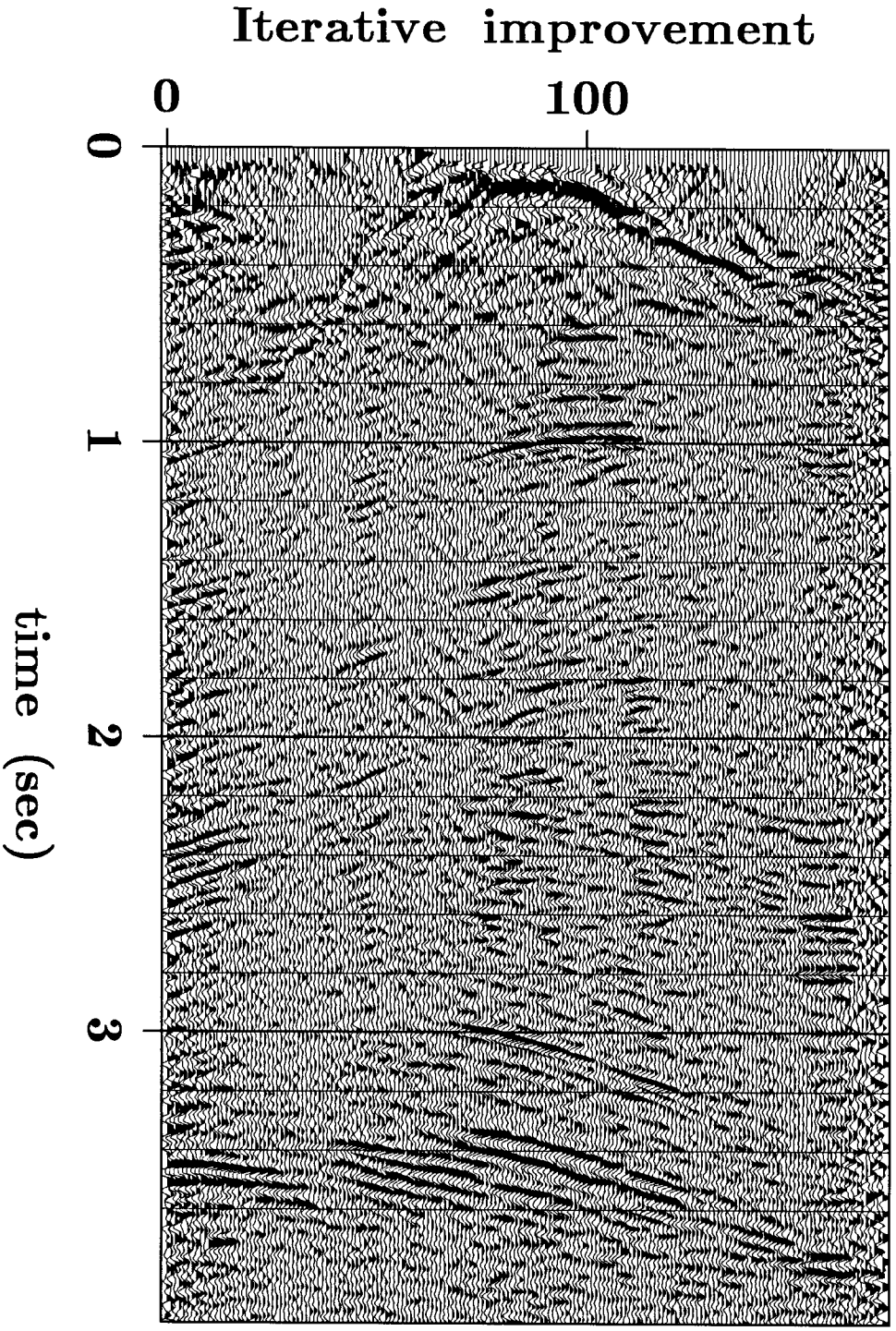

FIG. 11. Stack after application of a statics solution obtained by a process that iteratively chooses the best value for each shot and receiver static until stack power can no longer increase. This process is equivalent to Monte Carlo statics estimation with $T = 0$; i.e., quenching. All other processing parameters are identical to those used for the example in Figures 3a-e. Convergence occurred after only 13 iterations. This stack should be compared with the solution obtained by simulated annealing with non-zero T, shown in Figure 3e (and also Figures 7 and 9). Because this technique of iterative improvement converges to the nearest extremum, cycle-skipping can be a problem. Cycle-skipping is evident in the region near 3.5 s, which should exhibit a continuous reflector.

Discussion

Several interesting characteristics of the Monte Carlo statics algorithm emerge from the example illustrated in Figures 2-10. I detail a few of these features below.

Perhaps the most interesting result of this test is the rapid, almost discontinuous jump in power that occurs at approximately iteration 1150. Looking at the stacks in Figures 3a-e, we see that this jump corresponds to a period of rapid change in which the stack is transformed from almost total disorder to a very reasonable approximation of the final solution. This behavior is analogous to the sudden crystal groupings that occur when a liquid approaches its freezing point, so the behavior may be aptly labeled a *critical phenomenon.* In this example of statics estimation, however, there is no obvious passage through a *critical temperature* T_c (i.e., the freezing point). I instead choose a $T_{\min} \leq T_c$ and wait for the system to reach equilibrium. The system eventually falls into the equivalent of a potential well. This abrupt transition underscores the analogy to statistical mechanics and crystallization.

Carrying this analogy even further, one can foresee how the placement of a "seed" would help the "crystal" grow. This idea is indicated in Figure 2c. The rapid increase in power beyond this point (1125 iterations) was possible because the barest indication of a good stack had spontaneously formed. If there were enough prior knowledge of the correct solution so that just a few statics could be held constant during a run, then convergence to a high-stack-power solution should be significantly faster. In practice, it might often be possible to hold some statics constant when a zone of large statics is enclosed by data that are relatively free of statics (i.e., when there is a statics "bust"). The statics surrounding the poor region could then be held to zero while the algorithm searched for the optimal statics in the inner zone. The good stack in the surrounding region should propagate into the poor zone in a way that would be analogous to the nucleation of a crystal.

In a practical context, Monte Carlo statics estimation would be most efficiently used only to approximate the best stack, not necessarily to find it. In the Wyoming example, each static could be only an integral multiple of 8 ms; thus the true optimal (non-discretized) solution was surely missed. The best estimate from a Monte Carlo statics algorithm should be used as the initial guess for a conventional, linearized technique. Simulated annealing does not replace conventional statics algorithms, but this Monte Carlo method can be a useful tool

when conventional methods fail.

Monte Carlo statics estimation must be carefully used, however, because the solutions are non-unique. Simulated annealing can discriminate among minima of different depths, but it clearly cannot choose among minima of the *same* depth. Unless otherwise constrained, the algorithm will settle into the first deep minimum it finds, regardless of the magnitude of the non-unique contribution to the solution. Thus, as illustrated in the Wyoming example, dips can change considerably. One way to rectify this problem is to eliminate the poorly determined components of the solution; this technique is demonstrated in Figures 9 and 10. Another method would be to filter out the non-unique components after each iteration (Ronen, pers. comm.). As in any poorly determined inverse problem, it is wise to interpret the solution carefully.

The final solution can be surprising: statics estimation by simulated annealing is powerful enough to produce a virtually unpredictable change in the apparent structure in an input stack. In fact, Monte Carlo statics estimation can reveal structure where no previous hint exists. Although the input stack in Figure 2a does show clear reflection events, the solution was actually obtained after passage through complete disorder, as illustrated by Figures 3a and 3b. This firmly attests to the power of the technique.

Given, then, that the Monte Carlo statics algorithm can create something from nothing, one must question the accuracy of the solution that it produces. Other than non-uniqueness, the reliability of the solution can be measured by comparing the improvements inside the statics estimation window with the possible improvements outside the window. In this example the two improvements compare favorably: the new continuity in the deep reflector appears below steeply dipping events that were not apparent before. Thus the shallow data act as an independent confirmation of the result.

CONCLUSIONS

This paper establishes that Monte Carlo statics estimation can be used successfully in a practical setting. The Wyoming Overthrust data present severe statics that can produce the classic cycle-skipping problem, but the result illustrated here shows that cycle-skipping need no longer be the difficult, if not insurmountable, problem it has previously been for automatic statics algorithms.

The objective of this work is to obtain useful solutions that alternative automated techniques cannot find. Although the use of thousands of iterations might appear impractical, it might be a necessity. The number of possible solutions for the Wyoming example is immense—practically, infinity; thus the mere attainment of a reasonable solution is intriguing.

The obstacle of obtaining a solution in a reasonable length of time is not, however, trivial. Because the one-step (heat-bath) method is significantly more efficient than the Metropolis technique, it is regarded as an essential ingredient in the quest for practical capabilities. Its success stems from its focusing specifically on cycle-skips, rather than randomly choosing among all possible shifts. Yet the algorithm is, of course, slow. The nonlinear nature of the problem appears to preclude a truly fast method.

A more efficient technique might be possible, however, if the problem were posed differently. The Monte Carlo algorithm treats statics as a fully nonlinear problem, thereby extending our ability to estimate statics from a linear into a completely nonlinear realm. But perhaps the problem need not be fully nonlinear; for example, there should be a method that would more strongly discount the possibility of producing a highly disordered stack, at any stage of the iterative process. A more effective algorithm would merge the efficiency of linearized techniques with the Monte Carlo method's crucial ability to escape local minima. Random wandering would be limited, but not eliminated.

No systematic, empirical study of statics estimation by simulated annealing has yet been undertaken. Among the many unresolved issues is the algorithm's reliability: no two runs with identical parameters will exhibit identical behavior if different random numbers are used. Although global optimization with a given T may succeed, another attempt might fail. Preliminary tests indicate that the algorithm is workably reliable. The method's theoretic generality obviously gives in to practicality, but at exactly what point remains unknown. As deep minima become more numerous, global optimization becomes harder to achieve. In general, the number of deep minima increases as the magnitude of the statics increases, and as the signal-to-noise ratio and the bandwidth decreases.

The behavior of the algorithm can appear magical: from complete disorder, a relatively structured, ordered stack can appear. Order/disorder transitions in physics produce much the same effect. To the extent that both phenomena are understood, both can be explained in the same way: for a system in equilibrium, low energy states are more probable than high energy states. This statement of

probability does not fully discount the possibility of entrapment in a local minimum, but does strongly bias the search toward the deepest minima.

ACKNOWLEDGMENTS

I would like to thank Jim Johnson and Conoco Oil Co. for generously providing the Wyoming Overthrust data. Shuki Ronen suggested the subtraction of the linear trend in the estimates of statics; I benefited from many discussions on statics estimation both with him and Jon Claerbout. Stew Levin critically reviewed the first draft of the manuscript. The paper was then meticulously edited by Fannie Toldi, and Paul Fowler pointed out a few remaining flaws. I also learned much from my discussions with Peter Mora, Francis Muir, John Toldi, Ken Larner, Ron Chambers, and Steve Cole. This work was supported by the Amoco Foundation and the sponsors of the Stanford Exploration Project.

REFERENCES

Binder, K., Ed., 1984, Applications of the Monte Carlo method in statistical physics: Springer-Verlag.

Binder, K., Ed., 1979, Monte Carlo methods in statistical physics: Springer-Verlag.

Creutz, M., 1984, Quarks, gluons, and lattices: Cambridge University Press.

Feller, W., 1968, An introduction to probability theory and its applications: 3rd ed., v. 1, John Wiley and Sons.

Fosdick, L.D., 1963, Monte Carlo computations on the Ising lattice: Methods in Computational Physics, 1, 245-280.

Geman, S. and Geman, D., 1984, Stochastic relaxation, Gibbs distributions, and the Bayesian restoration of images: IEEE Transactions on Pattern Analysis and Machine Intelligence, PAMI-6, 721-741.

Hammersley, J.M. and Handscomb, D.C., 1964, Monte Carlo methods: Chapman and Hall.

Johnson, J.H., Yancey, M.S., and D'Onfro, P.S., 1983, Application of a statics solution, Wyoming Overthrust, *in* Bally, A.W., Ed., Seismic expression of structural styles: v. 3, The American Association of Petroleum Geologists, 3.4.1-35 - 3.4.1-40.

Kemeny, J. and Snell, J., 1960, Finite Markov chains: D. Van Nostrand and Co., Inc.

Kirkpatrick, S., Gelatt, C.D., Jr., and Vecchi, M.P., 1983, Optimization by simulated annealing: Science, 220, 671-680.

Metropolis, N., Rosenbluth, A., Rosenbluth, M., Teller, A., and Teller, E., 1953, Equation of state calculations by fast computing machines: Journal of Chemical Physics, 21, 1087-1092.

Rebbi, C., 1984, Monte Carlo calculations in lattice gauge theories, *in* Binder, K., Ed., Applications of the Monte Carlo method: Springer-Verlag, 277-298.

Ronen, J. and Claerbout, J., 1985, Residual statics estimation by stack-power maximization: Geophysics, 50, no. 12 (in press).

Rothman, D.H., 1984, Nonlinear inversion, simulated annealing, and residual statics estimation: Stanford Exploration Project Rep. 41, 297-325, [submitted to Geophysics (7 Nov 84)].

Taner, M.T., Koehler, F., and Alhilali, K.A., 1974, Estimation and correction of near-surface time anomalies: Geophysics, 39, 441-463.

Wiggins, R., Larner, K., and Wisecup, D., 1976, Residual statics analysis as a general linear inverse problem: Geophysics, 41, 922-938.

APPENDIX

This appendix shows that the Monte Carlo statics estimation algorithm produces a series of parameter vectors with the Gibbs probability distribution (3), when the algorithm is run at constant T for an infinite number of iterations. To obtain this result, I describe the algorithm as a Markov chain, and then show that the standard limit theorem (stated below) holds. The basic properties of Markov chains are sketched only briefly. Readers unfamiliar with the general theory should refer to Chapter 15 of Feller (1968), Kemeny and Snell (1960), or any other relevant text.

Preliminaries

In this appendix, I refer only to those Markov chains having a finite number of states. A chain is *irreducible* if every state can be reached (after an arbitrary number of iterations and with some positive probability) from every other state. Any state $\mathbf{x}_i$ is said to be *periodic* if the probabilities of recurrence $p_{ii}^{(n)}$ are non-zero only for some $n > 1$ and an integral multiple of n. Otherwise, $\mathbf{x}_i$ is termed *aperiodic*. The following theorem (see, e.g., Feller, 1968; or Kemeny and Snell, 1960) will be employed in the derivation of the main result:

If a Markov chain is irreducible and aperiodic, then the limits

$$U_j = \lim_{n \to \infty} p_{ij}^{(n)} \tag{A-1}$$

exist and are independent of the initial state $\mathbf{x}_i$. *The numbers* $\{U_j\}$ *uniquely satisfy*

$$U_j > 0, \quad \sum U_j = 1 \tag{A-2}$$

and

$$U_j = \sum_i U_i \, p_{ij} \ . \tag{A-3}$$

The *transition-probability matrix* $\mathbf{P}$ contains the transition probabilities $\{p_{ij}\}$. $\mathbf{P}$ is called a *stochastic matrix* because each $p_{ij} \geq 0$ and each row sums to unity. Each element of the *state-probability vector* $\mathbf{u}(n) = \{u_i(n)\}$ is the probability of being in state $\mathbf{x}_i$ at time n. Letting $\mathbf{u}$ be a row vector, the one-step transition from $\mathbf{u}(0)$ to $\mathbf{u}(1)$ can be represented by the equation

$$\mathbf{u}(1) = \mathbf{u}(0)\,\mathbf{P} \quad ,$$

where $\mathbf{u}(0)$ contains the initial state probabilities. The state probability vector after n steps is

$$\mathbf{u}(n) = \mathbf{u}(0)\,\mathbf{P}^n \quad .$$

For chains that satisfy the theorem above, we can define

$$\mathbf{U} \equiv \lim_{n \to \infty} \mathbf{u}(0)\,\mathbf{P}^n \quad .$$

$\mathbf{U}$ is the steady-state, or equilibrium vector of the Markov chain. Rewriting (A-3) shows explicitly that $\mathbf{U}$ is an eigenvector of $\mathbf{P}$, with eigenvalue 1:

$$\mathbf{U} = \mathbf{U}\,\mathbf{P} \quad .$$

This relationship will be frequently used below.

To show that the statics algorithm is a Markov chain with Gibbs equilibrium probabilities, I shall first specify the structure of the transition-probability matrix $\mathbf{P}$. It will then be shown that $\mathbf{P}$ is both irreducible and aperiodic, after which it will be proven that the steady-state distribution is Gibbs.

The transition-probability matrix

The statics estimation algorithm sequentially "visits" each parameter X_m (a shot or receiver static) and changes the parameter's value by choosing a random number from the probability distribution in equation (2). Only N distinct values for each parameter are allowed (this limitation is just an upper and lower bound, sometimes called a "shift limit"). One iteration is completed after each parameter has undergone a (possible) transition.

The transition-probability matrix $\mathbf{P}(m)$ directs each change in the value of X_m. Because there are M parameters that can each assume any of N values, there are N^M possible states of the system, and $\mathbf{P}(m)$ is an N^M by N^M matrix. Each row contains only N non-zero elements, because only N new states are directly accessible from any given state. There are M distinct transition-probability matrices $\mathbf{P}(1), \cdots, \mathbf{P}(M)$ for each of the M parameters,

respectively. The matrix of transition probabilities that directs the changes from one complete iteration to another is given by the product of the M matrices:

$$\mathbf{P} = \mathbf{P}(1)\,\mathbf{P}(2)\cdots\mathbf{P}(M) \ . \qquad \text{(A-4)}$$

To show that the Gibbs distribution $\{\Pi_j\}$ in equation (3) is the equilibrium vector for $\mathbf{P}$, it will first be shown that $\mathbf{P}$ represents an irreducible and aperiodic Markov chain. It will then be shown that $\{\Pi_j\}$ is an eigenvector with eigenvalue 1 for each $\mathbf{P}(m)$ and thus also for $\mathbf{P}$. Using the theorem stated above, we can then conclude that the steady state of the Markov chain is the Gibbs distribution of equation (3).

Irreducibility

A transition-probability matrix is irreducible if every state can be reached from every other state with some positive probability over some arbitrary time. A Markov chain must be irreducible if it is to have an equilibrium distribution; otherwise the system may fall into a state from which it can not enter some other states, and the system can no longer be independent of its initial configuration.

A set of states in which all members of the set are reachable (over time and with positive probability) from all other members of the set is called an *ergodic class*. Following the argument used by Fosdick (1963) for a two-dimensional (Ising) lattice, I now show that $\mathbf{P}$ is irreducible by showing that all possible states belong to the same ergodic class.

For each transition of X_m, only N possible values are allowed. This transition produces one of N new (or repeated) states. Prior to the transition of X_m, N^{M-1} different configurations of the other $M-1$ parameters are possible. When X_m changes, the other $M-1$ parameters remain constant, and the new state is one of the N possible states that contain the pre-existing configuration of the other $M-1$ parameters. These N states are all accessible to each other via the transition matrix $\mathbf{P}(m)$, but they are inaccessible from any other state using this transition matrix. Thus each $\mathbf{P}(m)$ partitions the N^M states into N^{M-1} ergodic classes, and each of these ergodic classes is a disjoint set of N states.

Now suppose that we are interested in the transition of X_m followed by the transition of X_{m+1}. Both $\mathbf{P}(m)$ and $\mathbf{P}(m+1)$ partition the states into N^{M-1} ergodic classes. Each ergodic class in $\mathbf{P}(m)$ is different from each ergodic class in $\mathbf{P}(m+1)$, but each state is contained in an ergodic class defined by both matrices. For example, $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$ may be the states of an ergodic class

in $\mathbf{P}(m)$. $\mathbf{P}(m+1)$ defines N ergodic classes (among others) that each contain one of these N states:

$$\{\mathbf{x}_1, \mathbf{x}_{N+1}, \cdots, \mathbf{x}_{2N}\}$$
$$\{\mathbf{x}_2, \mathbf{x}_{2N+1}, \cdots, \mathbf{x}_{3N}\}$$
$$\vdots$$
$$\{\mathbf{x}_N, \mathbf{x}_{N^2+1}, \cdots, \mathbf{x}_{N^2+N}\}.$$

When the product $\mathbf{P}(m)\,\mathbf{P}(m+1)$ is taken, each of these N ergodic classes are linked via the class $\{\mathbf{x}_1, \mathbf{x}_2, \cdots, \mathbf{x}_N\}$ in $\mathbf{P}(m)$; thus $\mathbf{P}(m)\,\mathbf{P}(m+1)$ partitions the N^M states into N^{M-2} ergodic classes. By induction, we see that the product (A-4) links all states into a single ergodic class of N^M states. Thus, because every state can be reached from every other state, $\mathbf{P}$ is an irreducible transition-probability matrix.

Aperiodicity

A state $\mathbf{x}_i$ is periodic if its probability of recurrence, $p_{ii}^{(n)}$, is non-zero only for some $n > 1$ and an integral multiple of n. Otherwise, the state is aperiodic. All states of a system must be aperiodic if there is to be a limiting equilibrium distribution; otherwise the system will not exhibit a distribution of states that is independent of time. To show that all N^M states of the system under consideration are aperiodic, I will demonstrate that $p_{ii} = p_{ii}^{(1)}$ is non-zero for every i.

Each row of each $\mathbf{P}(m)$ contains N, and only N, non-zero elements. One of these elements is always on the diagonal, because there is always some positive probability of retaining the current value of the m th parameter. For simplicity, consider a two-parameter system with transition matrix $\mathbf{Q} = \mathbf{Q}(1)\,\mathbf{Q}(2)$. Explicitly stated, the matrix product is

$$q_{ij} = \sum_k q_{ik}(1)\, q_{kj}(2)$$

and the diagonal element is

$$q_{ii} = \sum_k q_{ik}(1)\, q_{ki}(2) \ .$$

Because $q_{ii}(1) > 0$ and $q_{ii}(2) > 0$, and all the other elements are non-negative, then $q_{ii} > 0$. By induction, the same conclusion is true for the diagonal elements of the transition matrix for an M-parameter system. Thus, because each

diagonal element of $\mathbf{P}$ is positive, all N^M states are aperiodic.

The steady state

I now explicitly construct $\mathbf{P}(m)$ for the one-step (heat-bath) method. Unlike the two-step (Metropolis) approach, the random changes of a parameter's value that this method performs do *not* depend on the parameter's current value. The method can be characterized as a Markov chain, however, because the current values of the *neighboring* parameters (those shot and receiver statics within a cablelength) determine the conditional probability distribution for any particular parameter.

The transition probabilities that govern the m th parameter are given by

$$p_{ij}(m) = \begin{cases} \dfrac{\exp\{-E(\mathbf{x}_j)/T\}}{\sum\limits_{j \in A_i(m)} \exp\{-E(\mathbf{x}_j)/T\}} & j \in A_i(m) \\ \\ 0 & \text{otherwise} \end{cases} \tag{A-5}$$

where $A_i(m)$ is the set of N indices j such that $\mathbf{x}_j = \mathbf{x}_i$ everywhere except (possibly) at X_m. It will be shown that

$$\Pi_j = \sum_{i \in A} \Pi_i \; p_{ij}(m) \; , \tag{A-6}$$

where Π_j is given by the Gibbs probability distribution of equation (3) and $A = \{1, 2, \cdots, N^M\}$. This relationship will establish that $\{\Pi_j\}$ is an eigenvector with eigenvalue 1 for each $\mathbf{P}(m)$.

Equation (A-5) says that, for a given i, p_{ij} is non-zero only if $j \in A_i(m)$. Likewise, for a given j, p_{ij} is non-zero only if $i \in A_j(m)$. Thus we may write

$$\sum_{i \in A} \Pi_i \; p_{ij}(m) = \sum_{i \in A_j(m)} \Pi_i \; p_{ij}(m) \; .$$

Substituting equation (3) for Π_i and equation (A-5) for p_{ij} in the right-hand side above, we obtain

$$\sum_{i \in \mathbf{A}} \Pi_i \; p_{ij}(m) = \sum_{i \in A_j(m)} \frac{\exp\{-E(\mathbf{x}_i)/T\}}{\sum\limits_{i \in \mathbf{A}} \exp\{-E(\mathbf{x}_i)/T\}} \; \frac{\exp\{-E(\mathbf{x}_j)/T\}}{\sum\limits_{j \in A_i(m)} \exp\{-E(\mathbf{x}_j)/T\}} \; .$$

Reversing the order of the numerators and bringing the outside summation inside then yields

$$\sum_{i \in \mathbf{A}} \Pi_i \; p_{ij}(m) \;=\; \frac{\exp\{-E(\mathbf{x}_j)/T\}}{\sum_{i \in \mathbf{A}} \exp\{-E(\mathbf{x}_i)/T\}} \; \frac{\sum_{i \in A_j(m)} \exp\{-E(\mathbf{x}_i)/T\}}{\sum_{j \in A_i(m)} \exp\{-E(\mathbf{x}_j)/T\}} \;.$$

Now note that when $j \in A_i(m)$, $A_j(m) = A_i(m)$. Thus the second term on the right-hand side cancels to yield

$$\sum_{i \in \mathbf{A}} \Pi_i \; p_{ij}(m) \;=\; \frac{\exp\{-E(\mathbf{x}_j)/T\}}{\sum_{i \in \mathbf{A}} \exp\{-E(\mathbf{x}_i)/T\}} \;,$$

which is equal to Π_j. Thus $\{\Pi_j\}$ is an eigenvector with eigenvalue 1 for $\mathbf{P}(m)$.

Inspection of equation (A-4) shows readily that $\{\Pi_j\}$ is also an eigenvector with eigenvalue 1 for $\mathbf{P}$, and that $U_j = \Pi_j$ satisfies equations (A-2) and (A-3) of the limit theorem. Thus, because $\mathbf{P}$ is irreducible and aperiodic, $\{\Pi_j\}$ is the steady-state distribution of the Markov chain.

Residual statics estimation by simulated annealing: Another view

Daniel H. Rothman

INTRODUCTION

This paper proposes an attractive alternative to the earlier adaptation of simulated annealing for residual statics estimation (Rothman, 1984; Rothman, 1985). Simulated annealing is still used to solve a nonlinear optimization problem, but residual statics estimation is addressed now via traveltime decomposition, *à la* the classic approach of Taner et al. (1974) and Wiggins et al. (1976). The new method is potentially faster, and it can also handle residual normal moveout without undue complications.

MERGING LINEAR AND NONLINEAR METHODS

In the paper by Wiggins et al. (1976), residual statics estimation is explicitly posed as a linear inverse problem. Given observed time deviations t_{ij} for traces associated with the i th shot and j th receiver, Wiggins et al. used least-squares techniques to solve the system

$$t_{ij} \approx s_i + r_j \tag{1}$$

for the surface-consistent time delays s_i and r_j associated with the i th shot and j th receiver, respectively. (I've left out residual normal moveout and the so-called structure term; residual normal moveout will be considered at the end of the paper).

As discussed in Rothman (1984), statics present an inherently *nonlinear* inverse problem. It is the specification of the t_{ij}, however, that allows this problem to be linearized. When the observations $\{t_{ij}\}$ are accurate, linearization is not only valid, but it is also quite efficient. If the t_{ij} contain gross errors, however, we're no longer on firm ground—this results in the "cycle-skipping" problem.

My earlier papers (Rothman, 1984; Rothman, 1985) approached the cycle-skipping problem by treating statics as a fully nonlinear problem, in which it is necessary to globally minimize a functional that contains innumerable local minima. This method presupposes nothing about the optimal set of statics: there is no starting guess, nothing

akin to the $\{t_{ij}\}$ are ever picked, and the final solution can be wildly different from the input. The key issue is that neither the $\{t_{ij}\}$ nor the maxima of any other crosscorrelation functions play an explicit role in obtaining the solution; the algorithm simply tries to find the optimal statics, wherever they may be.

Because there are an an enormous number of possible solutions, simulated annealing is employed for optimization. Given the enormity of the solution space, however, it is no surprise that the solution is obtained only after thousands of iterations. But we must question whether this much effort is really necessary.

This effort is necessary if, *a priori,* we have no inkling of the optimal solution. In some cases, however, reflection events might be clearly apparent, albeit considerably discontinuous due to statics. Perhaps we can somehow use this partial clarity as an indication of the final stack, and hopefully narrow down the number of candidate solutions. Ideally, we would like to combine the efficiency of traveltime decomposition methods with the generality of simulated annealing. The new method, proposed below, is one such combination of linear and nonlinear techniques.

NONLINEAR TRAVELTIME DECOMPOSITION

When cycle-skipping is not a problem, the primary advantage of solving for statics by decomposing the observed traveltimes in equation (1) is that the size of the statics problem is greatly reduced. One begins by computing crosscorrelation functions for every trace. Then, instead of continuing to work directly with the seismic data or even the crosscorrelation functions themselves, one picks the peaks of the crosscorrelations; the corresponding lags are the $\{t_{ij}\}$. Thus the size of the problem is twice reduced: first, by going from seismic traces to much shorter crosscorrelation functions; and second, by reducing the information in each crosscorrelation function to a single point, t_{ij}. In contrast, the Monte Carlo method presented in my previous papers always uses the seismic traces for each calculation in every iteration. No attempt is ever made to reduce the data to a smaller size.

The method I propose here works primarily with crosscorrelation functions, instead of the seismic traces. Thus we can expect considerable computational savings.

Let each data trace be given by $d_{ij}(t)$, when indexing by the i th shot and j th receiver; or by $\overline{d}_{yh}(t)$, when indexing by midpoint y and offset h. The k th common-midpoint stacked trace is then given by

$$y_k(t) = \sum_h \overline{d}_{yh}(t) \ .$$

I also define *partially stacked* traces

$$y_k^{ij}(t) \;=\; y_k(t) - d_{ij}(t)\,, \qquad k \;=\; (i+j)/2 \;\;.$$

The trace $y_k^{ij}(t)$ is just the usual common-midpoint stacked trace minus one of the member traces in the CMP gather.

Step 1

Compute crosscorrelation functions $R_{ij}(\tau)$ by crosscorrelating $d_{ij}(t)$ against $y_k^{ij}(t)$; i.e., crosscorrelate each trace against a partially stacked trace:

$$R_{ij}(\tau) \;=\; \sum_t d_{ij}(t+\tau)\, y_k^{ij}(t) \;\;.$$

This results in one crosscorrelation function for each seismic trace.

Step 2

Transform the crosscorrelation functions to probability distributions, as described in Rothman (1985):

$$P_{ij}(t_{ij}) \;=\; \frac{\exp\{R_{ij}(t_{ij})/T\}}{\sum\limits_{t_{ij}} \exp\{R_{ij}(t_{ij})/T\}} \;\;. \tag{2}$$

$P_{ij}(t_{ij})$ is the probability that t_{ij} is the correct time delay for this trace. The scalar T regulates the spikiness of the distribution; T is chosen, as always, by experimentation.

Step 3

This is where the work gets done. We write

$$P_{ij}(t_{ij}) \;\approx\; P_{ij}(s_i + r_j) \;\;. \tag{3}$$

Assume that the $\{P_{ij}\}$ are all mutually independent. (This is clearly not true, but it will suffice as an approximation.) Let $\mathbf{s} = \{s_i\}$ and $\mathbf{r} = \{r_j\}$. Then we seek the $\mathbf{s}$ and $\mathbf{r}$ that maximize the product of the probabilities in equation (3):

$$\max_{\mathbf{s},\mathbf{r}} \prod_{ij} P_{ij}(s_i + r_j) \;\;.$$

Equivalently, we can pose the minimization problem

$$\min_{\mathbf{s},\mathbf{r}} \left\{ -\sum_{ij} \log[\, P_{ij}(s_i + r_j)\,] \right\} \;\;. \tag{4}$$

Because the function being minimized in equation (4) can be expected to contain innumerable local minima, simulated annealing is employed to perform global optimization. Each s_i and r_j is initially equated to zero. Then, for each s_i, we calculate the function

$$\phi_{s_i}(s_i) = \sum_{<j>_i} \log[\, P_{ij}(s_i + r_j)\,] \;, \tag{5}$$

where $<j>_i$ represents an index j that is paired with the given i; that is, the sum is taken over all receivers j that are turned on for shot i. We then form the probability distribution

$$Q_{s_i}(s_i) = \frac{\phi_{s_i}(s_i)}{\sum_{s_i} \phi_{s_i}(s_i)} \;. \tag{6}$$

Random guesses for s_i are drawn from this distribution; this guess is then the new value for s_i. In a similar manner, equations (5) and (6) are then set up for each r_j, and the $\{r_j\}$ are also updated. This sequence is then repeated until the minimization in equation (3) is achieved. Hopefully, T need only be chosen in the calculation of equation (2). It is possible that T may need to be gradually lowered to achieve the minimization.

Step 4

Restack the data, using the new estimates of the statics. Then go back to step one and repeat the whole cycle again. The procedure terminates when two consecutive iterations through steps 1 to 4 produce nearly equal results.

SIMULTANEOUS ESTIMATION OF VELOCITIES AND STATICS

The traveltime model given by equation (1) is predicated on the assumption that normal moveout has been accurately removed. Since stacking velocities are difficult to estimate before statics have been corrected, most traveltime models include *residual normal moveout* (RNMO). To include RNMO in equation (1), one writes

$$t_{ij} \approx s_i + r_j + m_k\, x_{ij}^2 \;, \tag{7}$$

where m_k is the *residual normal-moveout coefficient* for the k th common-midpoint gather and x_{ij} is the distance between shot i and receiver j. In reality, RNMO is different at every time; for simplicity, however, m_k represents the average RNMO within the entire computation window.

To solve the statics problem with RNMO included, we just need to incorporate the additional set of parameters $\mathbf{m} = \{m_k\}$. Steps 1 and 2 are the same as before. In Step 3, we now want to solve the optimization problem

$$\min_{\mathbf{s,r,m}} \left\{ -\sum_{ij} \log[\, P_{ij}(s_i + r_j + m_k x_{ij}^2)\,] \right\} , \tag{8}$$

where k is of course given by $(i+j)/2$. As before, we calculate

$$\phi_{s_i}(s_i) = \sum_{<jk>_i} \log[\, P_{ij}(s_i + r_j + m_k x_{ij}^2)\,] , \tag{9a}$$

where the sum is now taken over all receivers j and midpoints k associated with shot i. $\phi_{r_j}(r_j)$ is computed similarly. We also need to compute

$$\phi_{m_k}(m_k) = \sum_{<ij>_k} \log[\, P_{ij}(s_i + r_j + m_k x_{ij}^2)\,] , \tag{9b}$$

where m_k is allowed to vary over a predetermined range, and the sum is taken over all i,j pairs associated with a given midpoint k. The estimates of s_i, r_j, and m_k are drawn from the probability distributions Q. When the minimization in (8) is achieved, the data are then restacked as in Step 4. This time, however, RNMO corrections are also made prior to stacking.

The next level of complexity would be to compute RNMO as a function of time, thereby trying to estimate not only statics, but also corrections to the initial stacking velocities. Time dependence can be included in equation (7) by writing

$$t_{ijl} \approx s_i + r_j + m_{kl} x_{ij}^2 , \tag{10}$$

where t_{ijl} is the approximate time deviation for the i,j trace within time window l. Time windows could be placed around key reflectors, or they could be short overlapping segments. The optimization problem is now

$$\min_{\mathbf{s,r,m}} \left\{ -\sum_{ijl} \log[\, P_{ijl}(s_i + r_j + m_{kl} x_{ij}^2)\,] \right\} , \tag{11}$$

where the number of probability distributions P_{ijl} is given by the number of traces times the number of computation windows. The calculations of the functions ϕ are now

$$\phi_{s_i}(s_i) = \sum_{<jkl>_i} \log[\, P_{ijl}(s_i + r_j + m_k x_{ij}^2)\,] , \tag{12a}$$

$$\phi_{r_j}(r_j) = \sum_{<ikl>_j} \log[\, P_{ijl}(s_i + r_j + m_k x_{ij}^2)\,] , \tag{12b}$$

and

$$\phi_{m_{kl}}(m_{kl}) = \sum_{<ij>_{kl}} \log[\, P_{ijl}(s_i + r_j + m_k\, x_{ij}^2)\,] \quad . \tag{12c}$$

In equation (12a), the sum is taken over over all receivers j, midpoints k, and time windows l for s_i fixed; in (12b), the sum is over all shots i, midpoints, and windows for r_j fixed; and in (12c) the sum is taken over all i,j pairs, with midpoint and time window fixed. Estimates of the parameters are drawn from the probability distributions Q.

In many cases, RNMO is unlikely to vary rapidly in space. In practice, the $\{m_k\}$ could be smoothed after each iteration, or constraints could be incorporated into the $Q_{m_{kl}}(m_{kl})$.

DISCUSSION

Unlike my previous applications of simulated annealing, the method proposed here does not restack the data each time a parameter is updated. Instead, the crosscorrelation functions $R_{ij}(\tau)$ are assumed to be reasonably representative of reality, and we simply try to find the surface-consistent solution that best matches the crosscorrelations. We do not restrict ourselves to the peaks of the crosscorrelations, however, and thus hope to obtain a solution that is free of cycle-skips.

Importantly, estimates of RNMO are easily incorporated into this technique. This might make the simultaneous estimation of residual statics and stacking velocities a practical reality.

ACKNOWLEDGMENT

I thank Francis Muir for some helpful remarks concerning the manuscript.

REFERENCES

Rothman, D.H., 1984, Nonlinear inversion, simulated annealing, and residual statics estimation: Stanford Exploration Project Rep. 41, 297-325.

Rothman, D.H., 1985, Automatic estimation of very large residual statics: Stanford Exploration Project Rep. 42 (this report).

Taner, M.T., Koehler, F., and Alhilali, K.A., 1974, Estimation and correction of near-surface time anomalies: Geophysics, 39, 441-463.

Wiggins, R., Larner, K., and Wisecup, D., 1976, Residual statics analysis as a general linear inverse problem: Geophysics, 41, 922-938.