CHAPTER 4

Residual statics estimation by simulated annealing: results

This chapter contains results of residual statics estimation by simulated annealing for synthetic and field data. Both sets of data contain challenging problems that would normally cause gross errors in statics estimation. In both cases, however, the Monte Carlo statics algorithm is able to locate solutions that are nearly globally optimal. Any errors that do occur are due to poorly resolved spectral components of the statics solution, not the algorithm's inability to locate a deep minimum of the objective function.

The synthetic data results were generated using the adaptation of the Metropolis algorithm described in §3.3.2. Because the field data pose a more difficult problem for statics estimation (due to larger statics and more shots, receivers, and offsets), the Metropolis algorithm is too slow to be effective within a practical length of time. Thus the more powerful, one-step heat-bath method (§3.3.3) was applied to the field data. Despite the differences between the two techniques and the dissimilarities between the field and the synthetic data, both methods produced remarkably similar results.

As discussed in the previous chapter, there is no general practical scheme to determine the rate at which the temperature T is reduced. The results illustrated here were obtained by experimentation. Although the temperature-reduction schemes differed for the two examples, a single concept emerges from the two tests: a carefully chosen constant temperature can suffice for convergence. Because the choice of T appears to be data-dependent, I defer further remarks on this topic until the problem-specific discussions in §4.1.2 and §4.2.2.

4.1 SYNTHETIC DATA EXAMPLE

4.1.1 The data

The synthetic data simulate the results of a survey conducted with a 12-trace cable, off-end shooting with a two-receiver group gap, and evenly spaced shots and receivers. There are 100 6-fold common midpoint gathers. The sampling rate is 4 ms and the data contain frequencies from 5 to 60 Hz. The data, prior to the introduction of static shifts, are shown in Figures 4.1a and 4.1b. Figure 4.1a shows four representative "moveout-corrected" CMP gathers, and Figure 4.1b is the common midpoint stack. The cablelength extends over 24 stacked traces. The signal-to-noise ratio (the total power of the signal divided by the total power of the noise) after stack is approximately 2.0. The entire dataset is scaled to an rms amplitude of 100. For all traces the signal is identical, except for the bulk time shift simulating a fault. (Real faults would not exhibit such a severe discontinuity before migration.) These data represent the desired solution for the test illustrated in the following figures.

Random shot and receiver statics are displayed in Figure 4.2. These statics vary between ± 40 ms, in 4 ms increments. Figure 4.3a shows the same common midpoint gathers shown in Figure 4.1a, but now the traces have been shifted in accordance with the statics model in Figure 4.2. Figure 4.3b is the common midpoint stack after the model statics were applied. Because of the severity of the statics, almost no indications of reflection events can now be observed. The data in Figures 4.3a and 4.3b are the input to the statics estimation algorithm.

4.1.2 Processing parameters and method

Statics were estimated using the adaptation of the Metropolis algorithm described in §3.3.2. The power computations in equations (3.5) and (3.6) were performed over the entire length of the traces (280 ms) depicted in the stacked sections. Random guesses for shot and receiver statics were constrained to fall within ± 40 ms, in 4 ms (one-sample) increments.

For this example,

$$T_k \;=\; T_0 \log k_0 \,/\, \log(k_0 + 2k) \;\;, \tag{4.1}$$

with $T_0 = 4500$, $k_0 = 5000$, and k equal to the number of iterations. (One iteration includes one attempted perturbation of each shot and receiver static.) The choice of this rate of temperature reduction was motivated by a theoretical result of Geman and Geman (1984), who proved that convergence occurs when $T_k = T_0/\log k$, with the

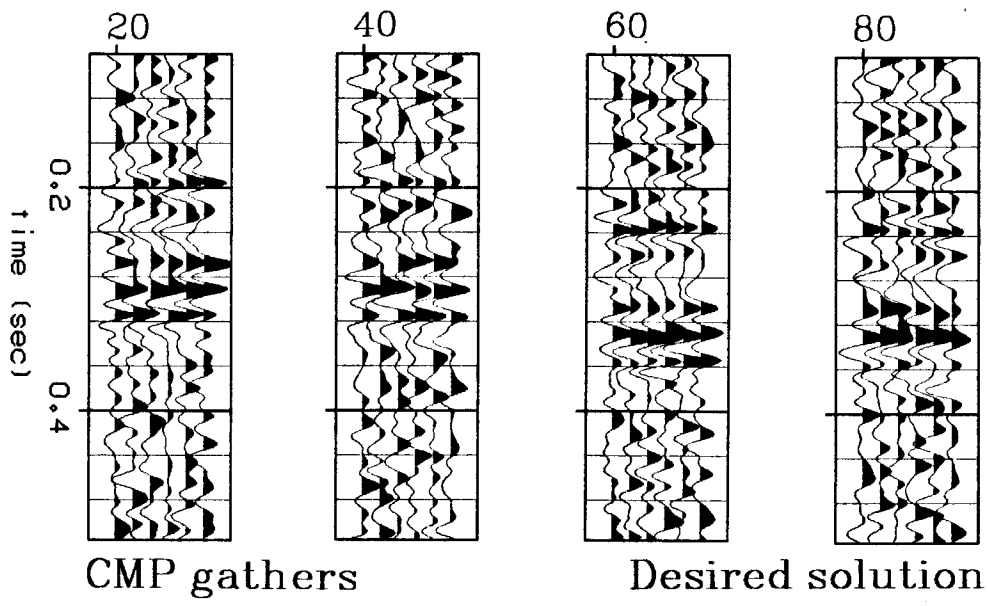

FIG. 4.1a. Four "moveout-corrected" CMP gathers. The gathers are shown without static shifts; there are 6 offsets in each gather. This correct alignment of traces is the desired solution for pre-stack data.

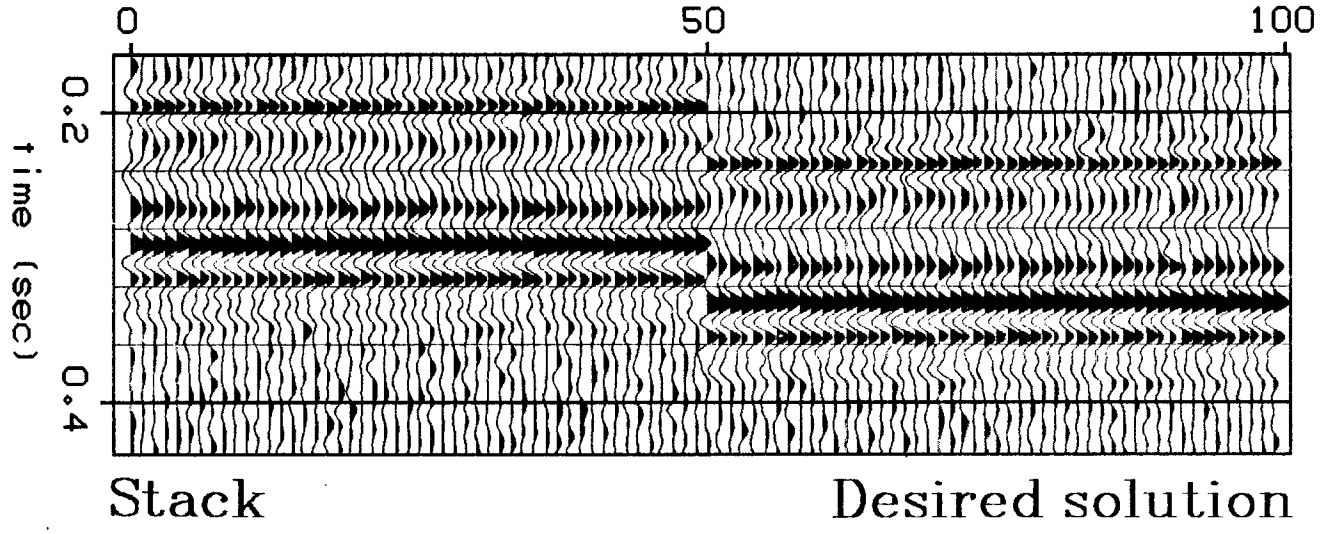

FIG. 4.1b. CMP stack prior to the introduction of static shifts. The cablelength extends over 24 midpoints; there are 100 midpoints in total. An artificial fault has been placed at the center of the section. The signal-to-noise ratio is approximately 2. This is the desired solution for stacked data.

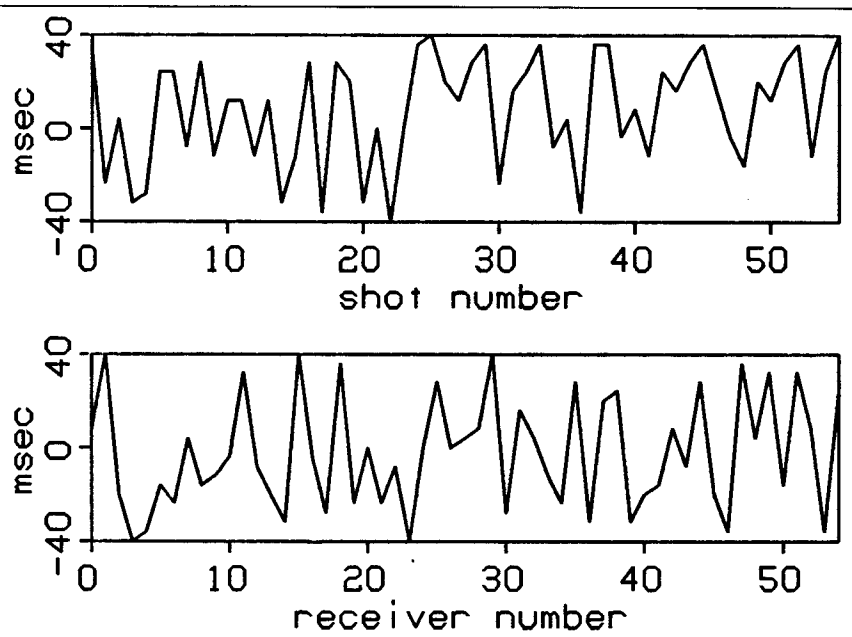

FIG. 4.2. Random shot statics (above) and receiver statics used to generate the test data shown in Figures 4.3a and 4.3b. Statics range between ±40 ms, in 4 ms increments, for both shots and receivers.

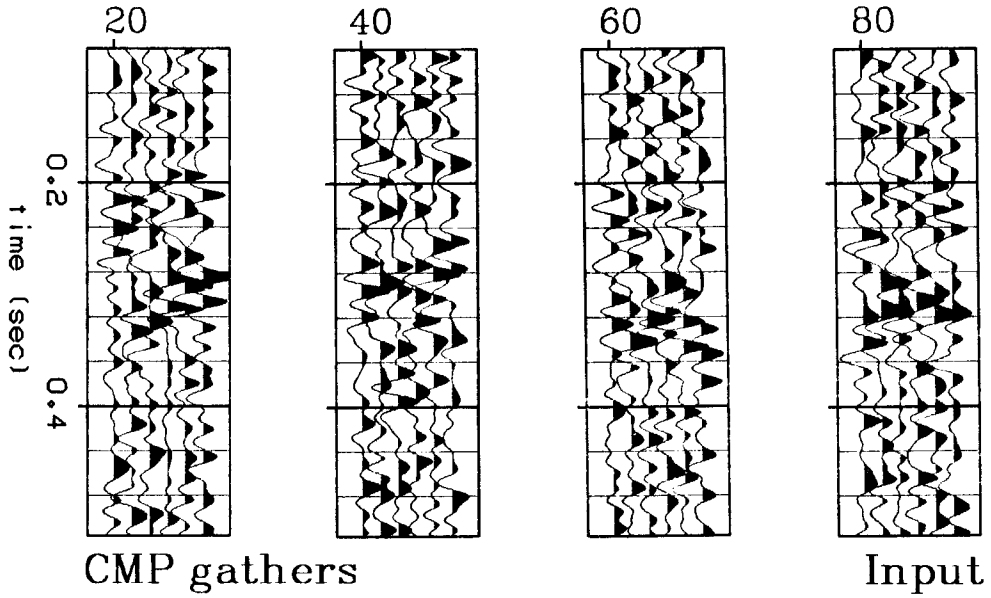

FIG. 4.3a. The CMP gathers of Figure 4.1a after the application of the static shifts in Figure 4.2. Note how the application of the statics has degraded the appearance of the data.

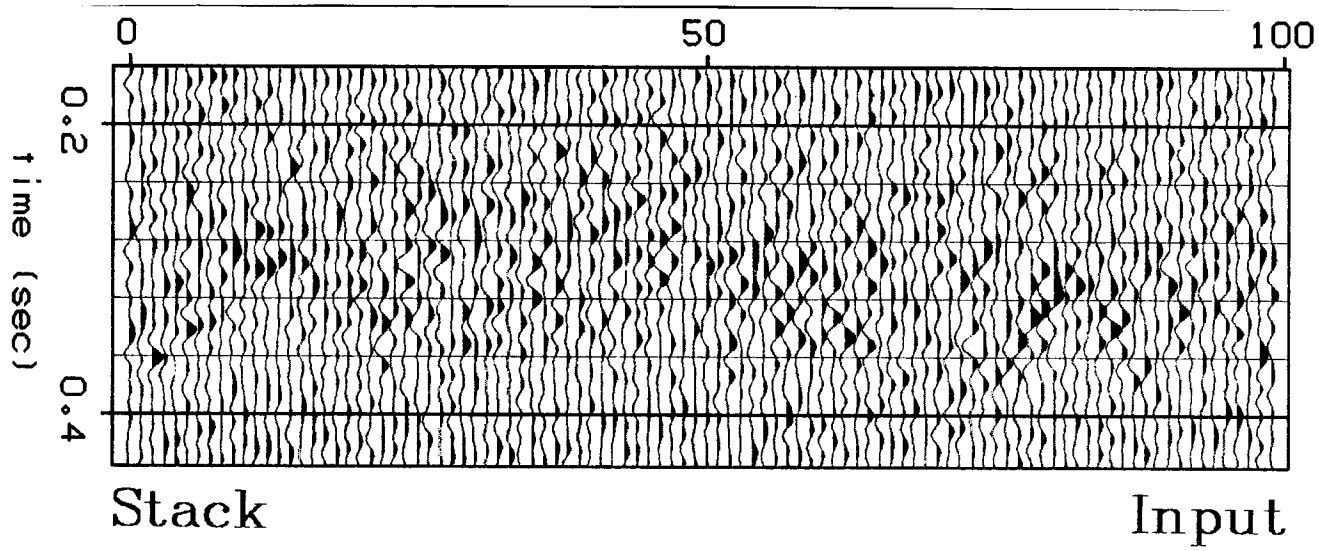

FIG. 4.3b. CMP stack after application of the statics in Figure 4.2. Because the shifts are as much as 160 ms apart, almost no indication of the reflection events seen in Figure 4.1b can now be observed. The data in Figures 4.3a and 4.3b are the input to the statics estimation algorithm.

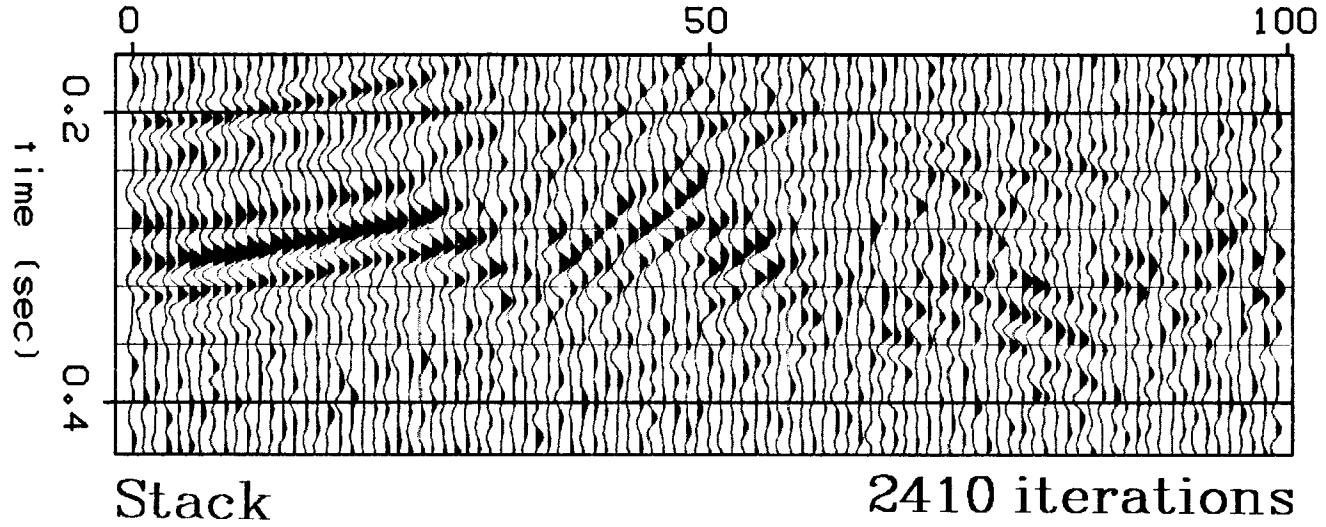

FIG. 4.4a. CMP stack after 2410 iterations of the statics estimation algorithm. Good convergence already appears on the left, though the remainder of the section exhibits the effects of misaligned traces.

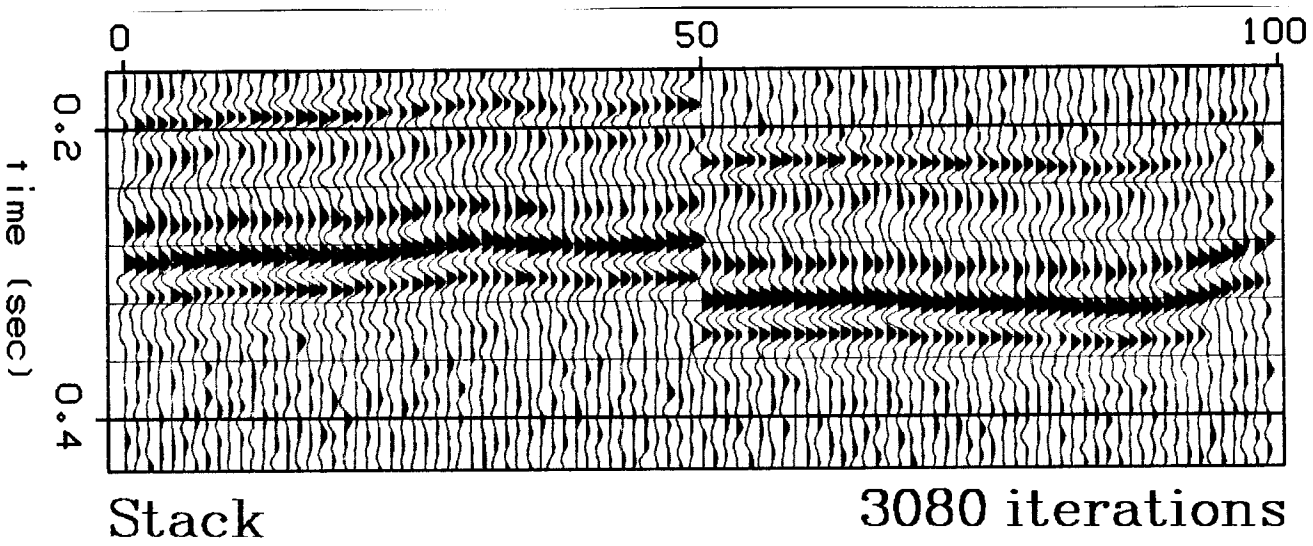

FIG. 4.4b. CMP stack after 3080 iterations. Although long-wavelength statics remain to be resolved, the bulk of the algorithm's work is completed. Note that, despite the ambiguity between structure and long-wavelength statics, the artificial fault at trace 50 is properly resolved.

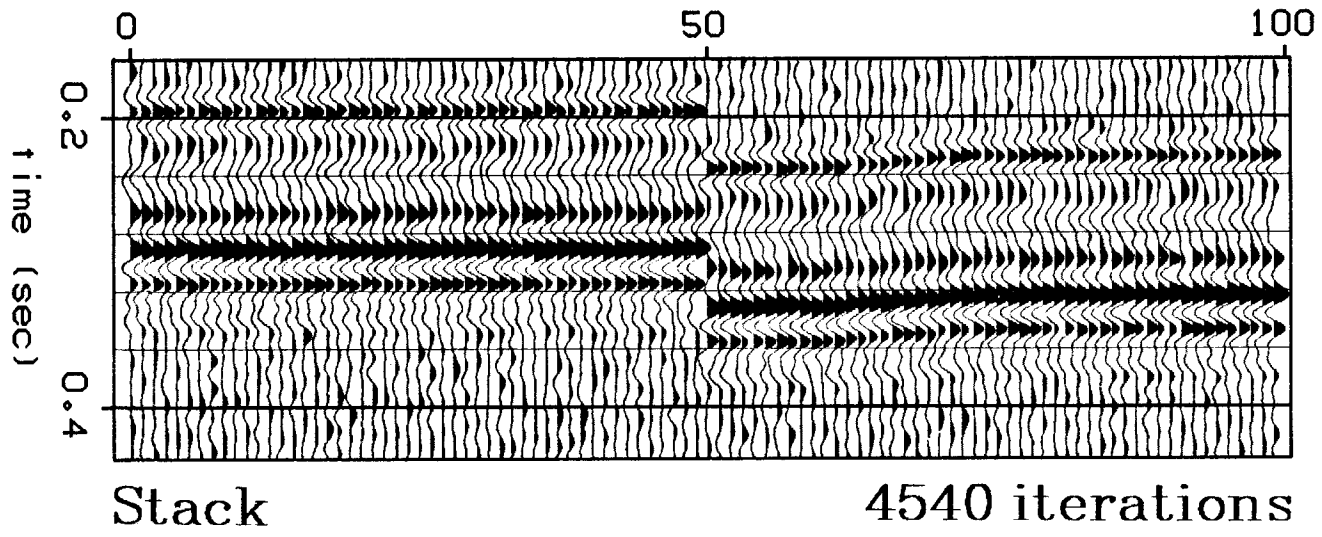

FIG. 4.4c. CMP stack after 4540 iterations. This is the final solution, and should be compared with the input (Figure 4.3b) and the known, desired solution (Figure 4.1b). The 8 ms rise in the right half of the section is a result of poorly resolved long-wavelength statics, due mostly to the noise contamination in the data.

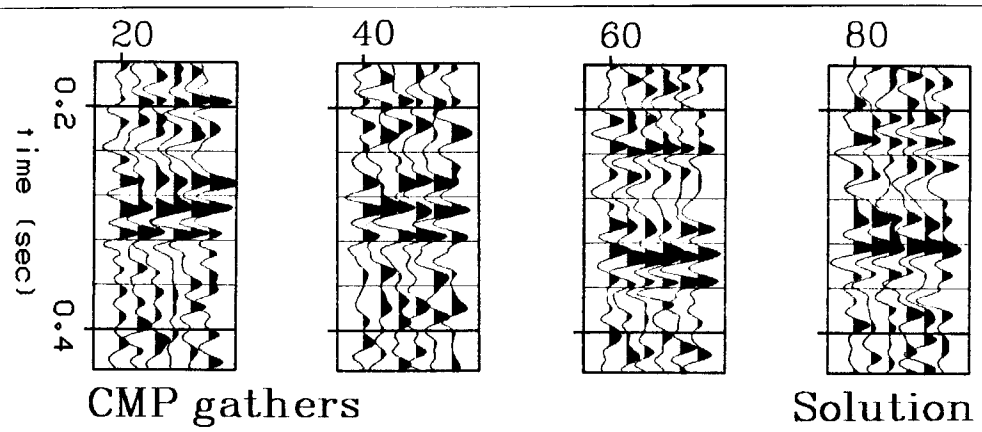

FIG. 4.4d. CMP gathers after the statics solution has been applied. This should be compared to the input (Figure 4.3a) and to the desired solution (Figure 4.1a). CMP 60 exhibits a slight error due to the poorly resolved long wavelength. The time axis is shorter now because the application of statics creates zeroes at early and late times.

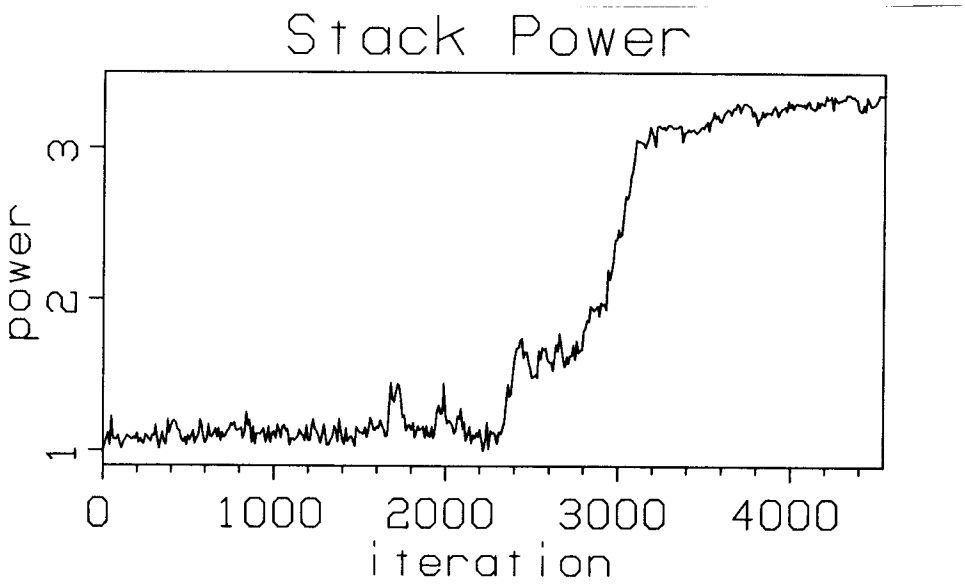

FIG. 4.4e. Stack power versus iteration number for the test leading to the result shown in Figure 4.4c. Stack power is computed every 10 iterations; the input stack power is normalized to 1. The final solution yields a stack power of 3.354, which is short of the true solution by 1.3%. Note the sudden increase in power after 3000 iterations. This abrupt change is analogous to rapid crystallization. Temperature decreases by less than 11% between the first and last iteration.

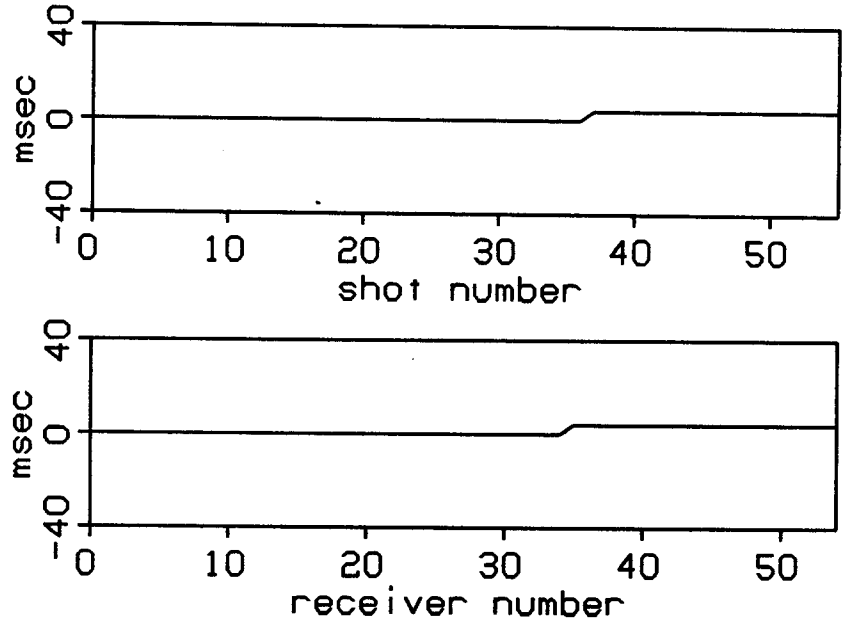

FIG. 4.5. Difference between the estimated statics and the true statics for the result in Figure 4.4c. The 8 ms rise in the right half of Figure 4.4c is the result of the constant 4 ms error for approximately the last 20 shot and receiver statics. The allowable values for statics fell within ±40 ms, in 4 ms (one sample) increments. The noise contamination for this test was too strong for the long-wavelength residual to be resolved.

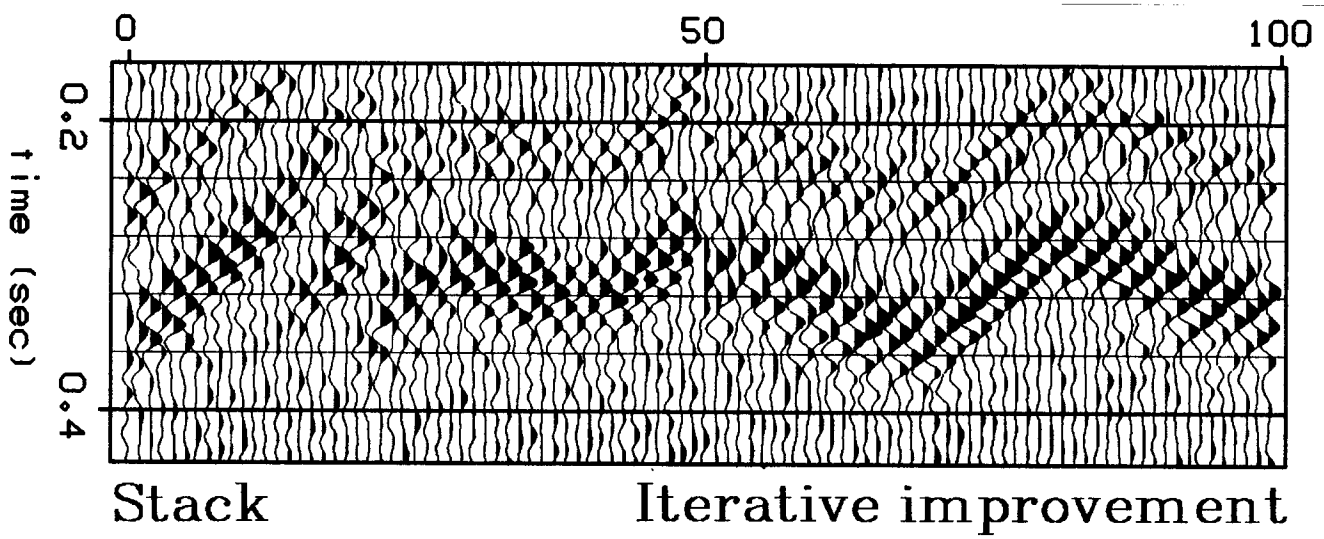

FIG. 4.6. The best result of 100 trials of residual statics estimation, made by iteratively choosing the best value for each shot and receiver static until a (local) power maximum was attained. Each trial was performed with a different random initialization of the shot and receiver statics. The stack power here is 29% short of the stack power of the result obtained by simulated annealing, shown in Figure 4.4c. The diagonal appearance of the stack is due to a severe cycle-skipping problem.

choice of T_0 dependent on the difference between the greatest and least possible energies. In performing my early tests I observed that choosing high initial values of T was unimportant: convergence consistently occurred near the same low T every time. In using equation (4.1), I elected to retain the same gentle logarithmic decay as Geman and Geman had used, but I additionally chose the constant k_0 such that T_k decreased very slowly from the first iteration onward. The result was a mild rate of cooling in which T changed by less than 11% from the first to the last iteration. Thus T was held nearly constant; this result suggested the constant-temperature approach used in the field data example.

4.1.3 Results

Figures 4.4a-e illustrate the results obtained with the statics algorithm. Three stages of the algorithm's execution are depicted: the stack after 2410 iterations (4.4a); after 3080 iterations (4.4b); and the final solution, after 4540 iterations (4.4c), which closely resembles the desired solution in Figure 4.1b. Figure 4.4d shows the four common midpoint gathers from Figure 4.3a; the statics solution has now been applied to them. Figure 4.4e is a graph of stack power versus iteration. Note that there is very little change in power until after approximately 2300 iterations. After 3000 iterations, the power sharply increases. By the time iteration 3080 was reached, the statics algorithm had completed its most important work: solving for the shorter-wavelength statics, leaving only long-wavelength residuals. The longer wavelengths are the most poorly resolved components of the solution; this is as true for other techniques (Taner et al., 1974; Wiggins et al., 1976; Ronen and Claerbout, 1985) as it is here. By iteration 4540 (the final solution), only a slight long-wavelength residual remains. Although the fundamental ambiguity of long-wavelength statics and structure is observed here as elsewhere, it is important to note that the structural discontinuity implied by the artificial fault does not influence the solution.

The quality of the solution is measured by the objective function, stack power. For comparison with results, the power of the input stack in Figure 4.3b is normalized to 1. The final stack power for the solution in Figure 4.4c is 3.354. The known, desired solution has a stack power of 3.399, so the computed solution is in error by approximately 1.3%. The difference between the estimated statics and the true statics is graphed in Figure 4.5. Note that, for both shots and receivers, the basic error occurs as a slight kink about two-thirds along the line. The noise contamination for this test was strong enough so that this long wavelength residual could not be resolved; other tests (not shown) with higher signal-to-noise ratios more successfully resolved the long

wavelengths.

The opposite of annealing is "quenching"; i.e., setting $T = 0$, so that only random perturbations that increase power are accepted. Efficient quenching can be accomplished by iteratively scanning all possible values for each shot and receiver static, and always choosing the shift that yields the greatest local stack power. This technique is optimization by iterative improvement. Because iterative improvement always finds the nearest local minimum, it is customary to perform several runs with different starting positions (i.e., different initial values for the shot and receiver statics) and to save the best result. I ran 100 such tests of iterative improvement on these synthetic data; the best of these 100 results is shown in Figure 4.6. Computations for these 100 runs consumed approximately the same time as that needed to obtain the annealing result of Figure 4.4c. The stack power for the solution in Figure 4.6 is 2.378, which is almost 30% less than the result obtained by simulating annealing. (The worst of the 100 trials was short by almost 50%.) The diagonal appearance of this result is due to a severe cycle-skipping problem, and is the visual manifestation of convergence to a local minimum. Local minima such as the one illustrated here would also be the expected result if simulated annealing were run with too low a value for temperature. Local minima are thus not only suboptimal solutions, but they are also usually geologically implausible solutions.

4.2 FIELD DATA EXAMPLE

4.2.1 The data

The seismic section in Figure 4.7a (p. 52) is a 24-fold "raw" or "brute" stack; this stack was produced without any statics corrections. Figure 4.7b shows two representative CMP gathers. The data were collected with a 48-trace, split-spread cable in the Wyoming Overthrust belt. 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) 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 width 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 obvious that these data suffer from a statics problem, supporting evidence exists. Note that the strong, shallow reflection in the stack (Figure 4.7a) appears to arch downward from about 100 ms at the center of the section to

about 600 ms at both sides. It will be evident later that this is the reflection from the base of low-velocity fill. The gathers (Figure 4.7b) 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 such 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 in §4.2.3. Before illustrating that result, however, I discuss the choice of T and other processing parameters for these data.

4.2.2 Processing parameters and method

Statics estimation for this set of data was performed with the one-step heat-bath method described in §3.3.3. Statics were estimated from normalized crosscorrelations of the data between 2.9 and 3.9 s, the zone dominated by the prominent reflector at approximately 3.5 s. Static shifts were constrained to fall within ± 160 ms, in 8 ms (one-sample) increments; thus the crosscorrelations in equation (3.9) were performed over 41 lags. Shot statics were estimated independently of receiver statics. The 85 source locations and 90 receiver locations yield a total of 175 free parameters. Because each parameter may assume any of 41 values, there are 41^{175} possible solutions.

As I have noted previously, choices for T appear to be data-dependent. 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 4.7a). Because statics appear to have caused a very large (approximately 200 ms) break, more than just an incremental change in the apparent structure in the stack is needed. 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 a statement 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

regions, 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 causes solutions to depart 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 in the deepest minimum, or in minima that are nearly as deep. It is unlikely that the algorithm will return solutions 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.

The objective, then, is to approach equilibrium as rapidly as possible 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. Based on my experimentation, this appears to be a reasonable model for data with large statics and moderate noise contamination; these conditions exist both in this example and in the synthetic data.

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 the algorithm would spend much time 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 entrapment in a local minimum is still possible, the probability of entrapment is significantly lowered by not beginning near a local minimum.

Regardless of how T is chosen, the solution remains independent of the starting position as long as the algorithm is run for enough iterations so that equilibrium is attained. The algorithm would be most efficiently used, however, if one could begin near the global minimum; if this were possible, the initial temperature could indeed be low. This scheme would be analogous to the placement of a "seed" from which a "crystal" could quickly grow. Further comments on the use of a seed are in the discussion in §4.3.

The temperature function chosen for this 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 considerable 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 (or at all); if $T_{\min}$ is too low, then the algorithm is likely to converge to an obviously unacceptable local minimum similar to the result in Figure 4.6. 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. In the example illustrated here, however, systematic experimentation with a range of values yielded the correct $T_{\min}$ without undue difficulty.

4.2.3 Results

Figures 4.8a-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 performed after corrections are made with the current estimate of the statics. Figure 4.8a displays the stack after 5 iterations; one sees 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 4.8b) the stack still exhibits no improvement over the result in Figure 4.8a. By iteration 1125, however, convergence begins; this stage is illustrated in Figure 4.8c. The algorithm then rapidly descends into a minimum, as is evident in Figure 4.8d, 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 4.8e. It should be compared with Figure 4.7a, the stack of the input data. In Figure 4.8e, not only has the deep reflector become continuous across most of the section, but also the statics corrections have revealed steeply dipping events in the more shallow data. (Gaps in these dipping events are the result of inaccurate stacking velocities.) There are two apparent imperfections in Figure 4.8e, 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 later in this section.

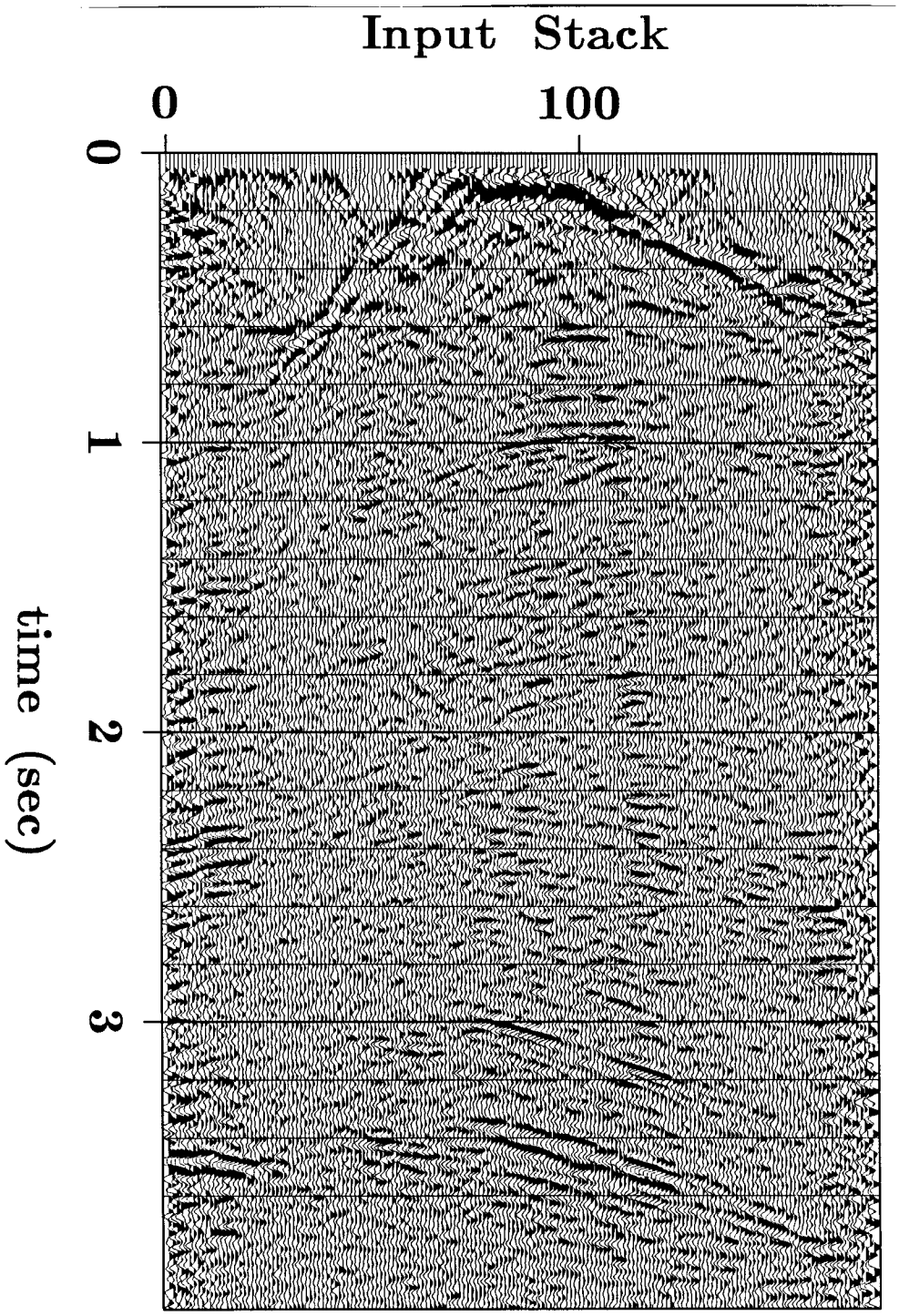

FIG. 4.7a. 24-fold stack of data from the Wyoming Overthrust belt; the stack is performed prior to statics estimation. One time-variable stacking-velocity function was used for the entire line. The data used for residual statics estimation are between 2.9 and 3.9 s. The strong reflections at the near 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.

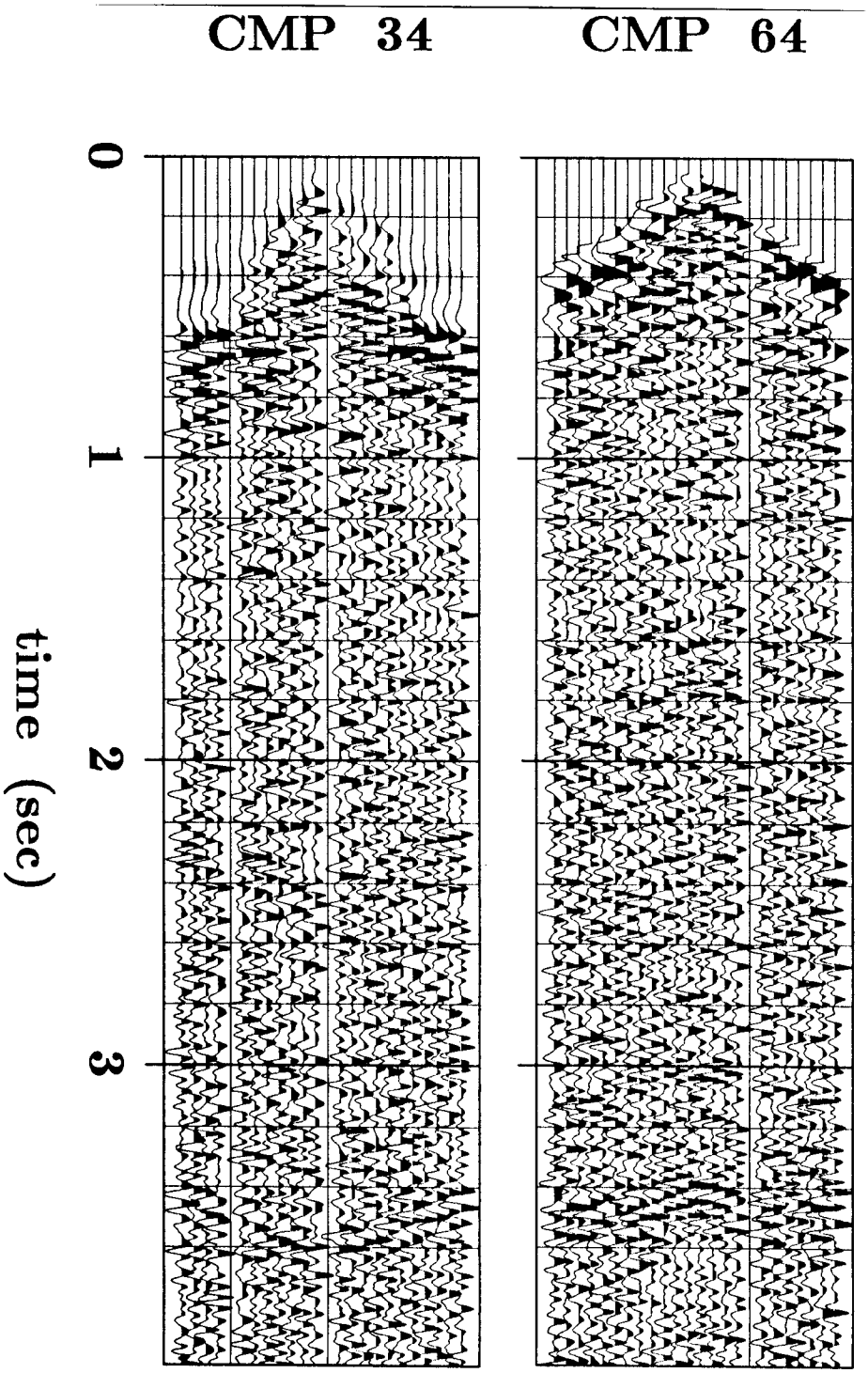

FIG. 4.7b. NMO-corrected common-midpoint gathers 34 and 64; the gathers are displayed without statics corrections. Offset increases in each plot from the center outward. Both gathers are corrected with the same velocity function. The near-surface velocity anomalies have produced dipping structure in events that should be flat; this effect 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.

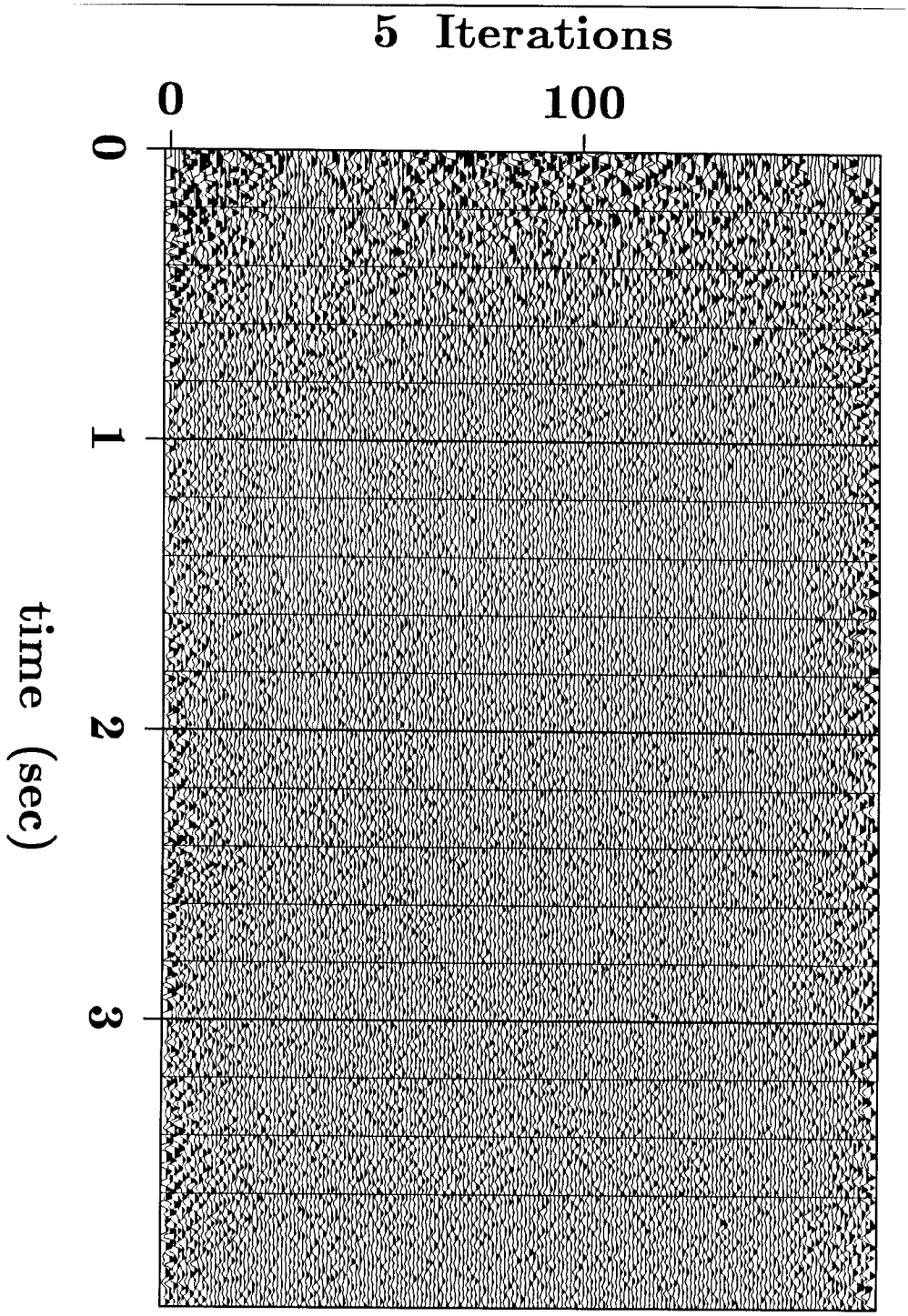

FIG. 4.8a. Stack after 5 iterations of the statics estimation algorithm. T_0 was chosen high enough so that all reflection events are now obliterated. This choice removes all tendencies for the algorithm to make only incremental improvements in the stack, as would be expected from a linearized technique. Amplitudes are greater at the sides because of the low fold at the ends of the line.

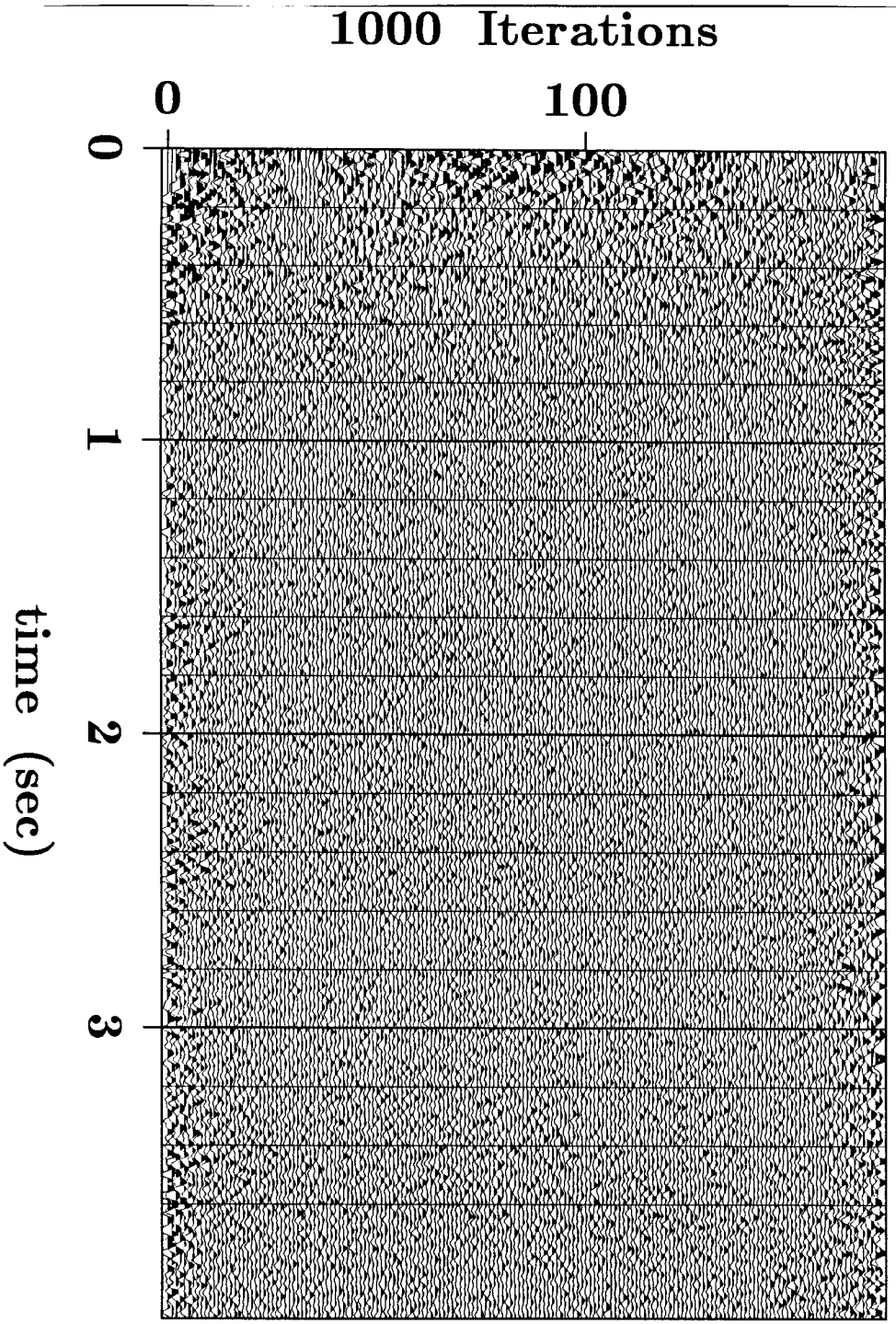

FIG. 4.8b. Stack after 1000 iterations of the statics estimation algorithm. $T = T_{\min}$ from iteration 53 onward. No appreciable differences between this stack and the result after 5 iterations (Figure 4.8a) are evident.

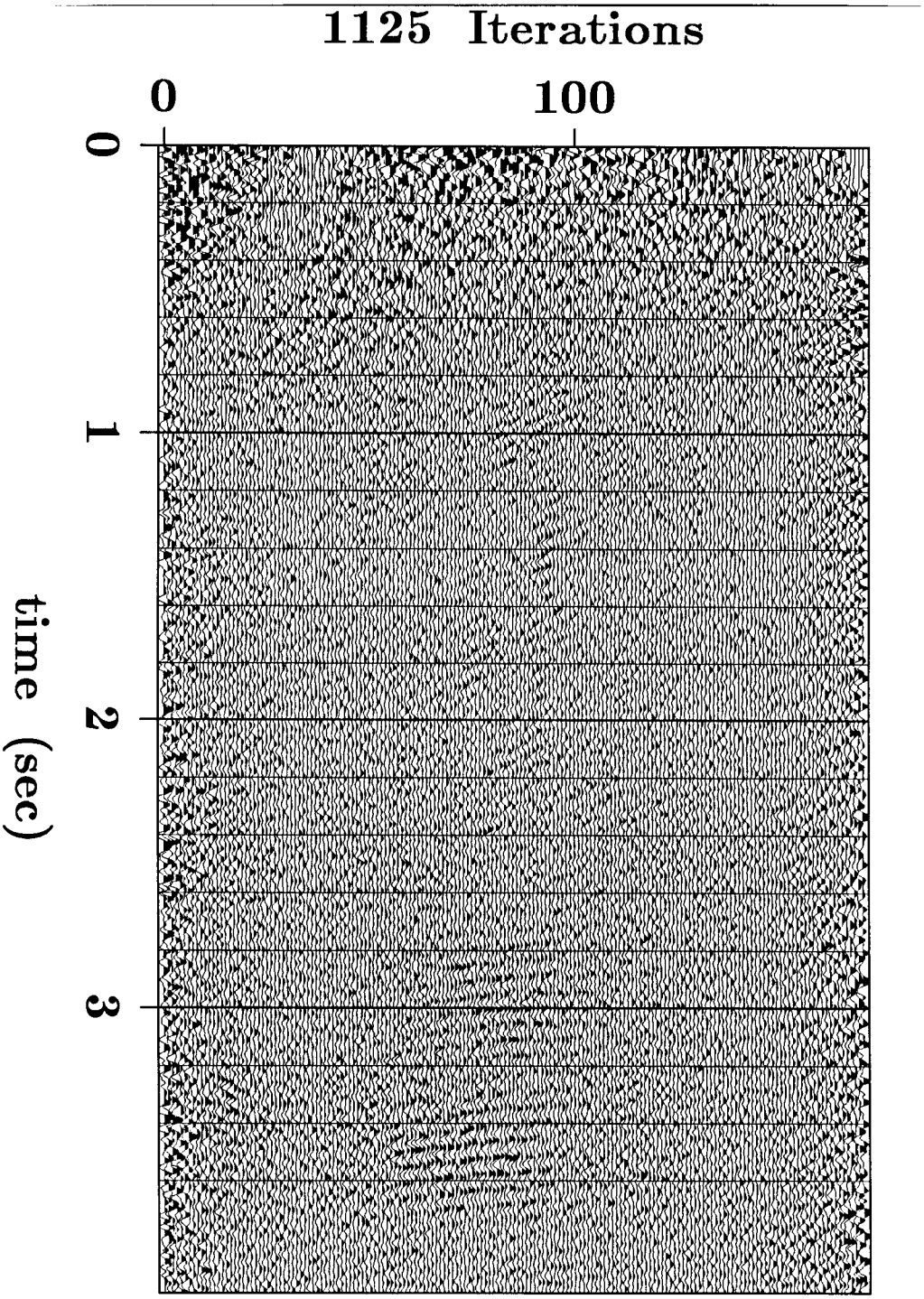

FIG. 4.8c. Stack after 1125 iterations of the statics estimation algorithm. The faint spatial coherence in the middle of the section shows that convergence is now beginning.

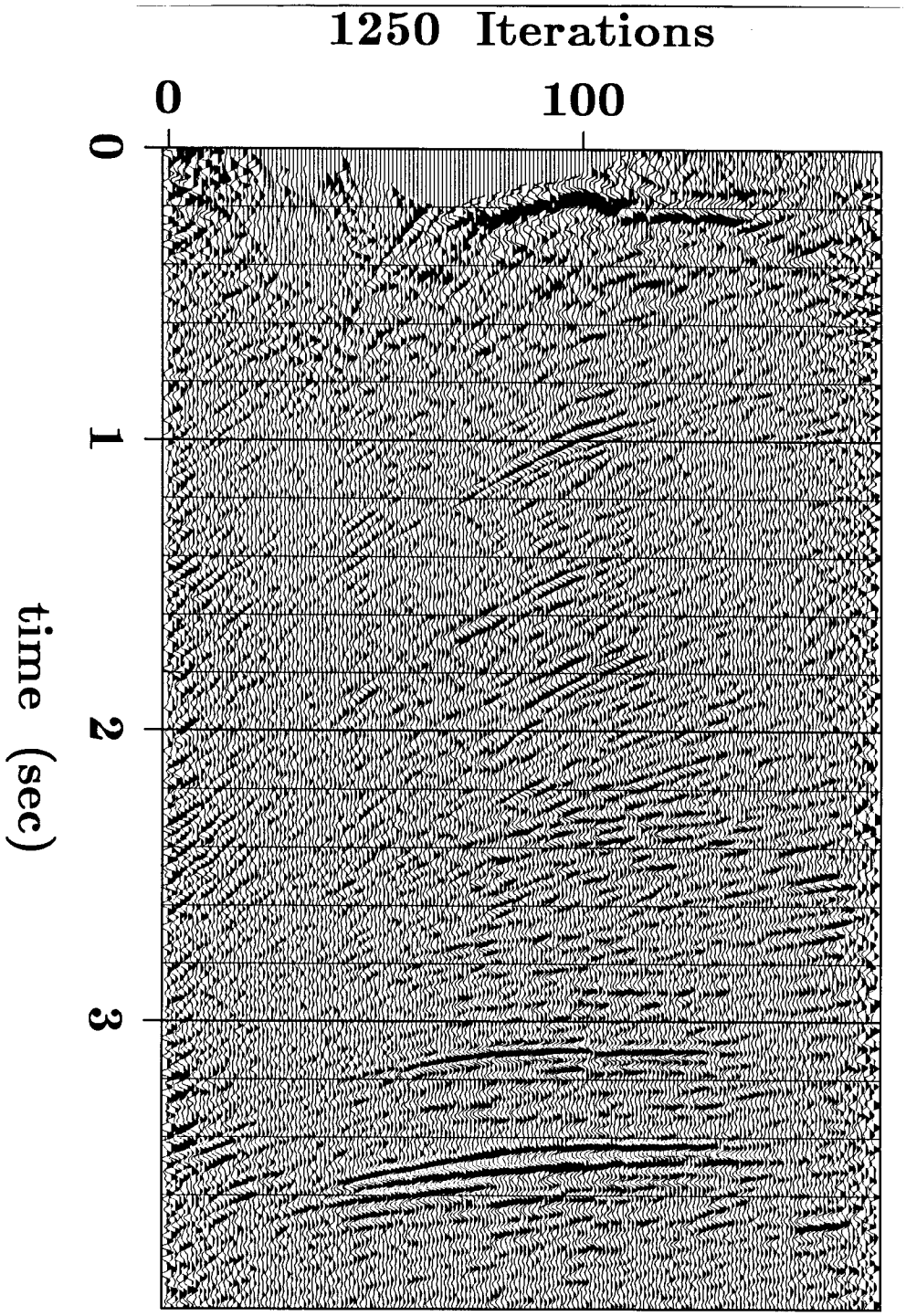

FIG. 4.8d. Stack after 1250 iterations of the statics estimation algorithm. Convergence is now almost complete; only at the ends of the line has the solution not converged.

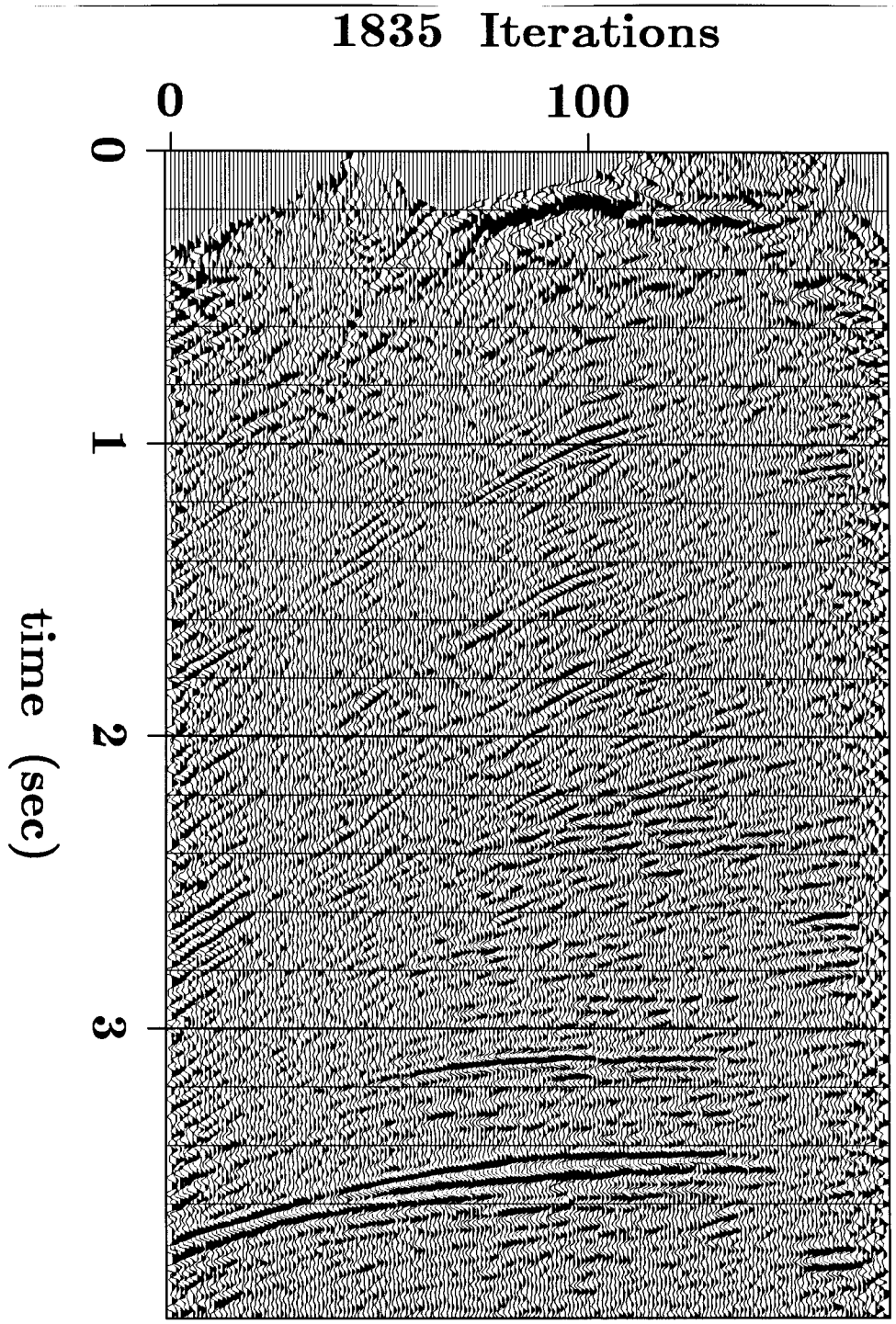

FIG. 4.8e. Stack after 1835 iterations of the statics estimation algorithm. Of the 2000 iterations, this stack yielded the greatest stack power. This stack should be compared with the stack of the input data in Figure 4.7a. The deep reflection is now continuous throughout much of the section. Moreover, this statics solution has uncovered steeply dipping structure in the first 3 seconds of the data (outside the window used for computing the crosscorrelation functions). 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.

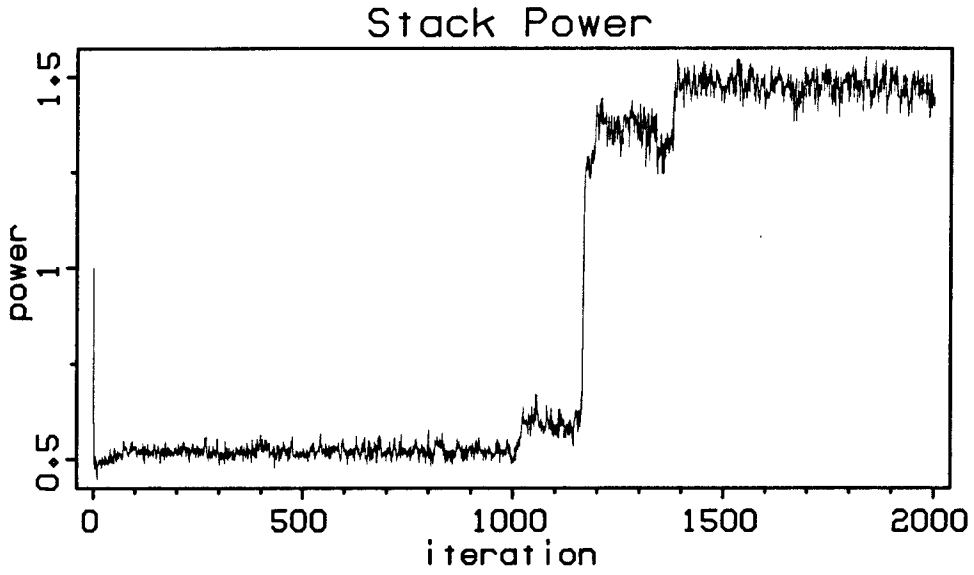

FIG. 4.9. Normalized stack power as a function of iteration number, for the example illustrated in Figures 4.8a-e. Normalization is based on setting the input stack power to 1. Power is computed within the estimation window, which extends from 2.9 to 3.9 s. Power initially decreases quickly to about .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 abrupt changes in a system's energy upon its crystallization. Global convergence occurs after about 1400 iterations. The remaining 600 iterations yield essentially similar stacks except for the behavior at the far right of the seismic sections. The power of the stack displayed in Figure 4.8 is 1.553, the greatest power attained in the 2000 iterations.

Figure 4.9 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 1000 iterations. Beginning at approximately iteration 1150, power increases sharply; 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 side of the section, which never became stable. The maximum stack power occurred in iteration 1835 (Figure 4.8e), where the power is 1.553.

Figure 4.10 shows the two gathers shown in Figure 4.7b; statics corrections from iteration 1835 have now been performed. In gather 34, the upward dips at far offsets have now been flattened. Also, gather 64 has had its downward dips leveled.

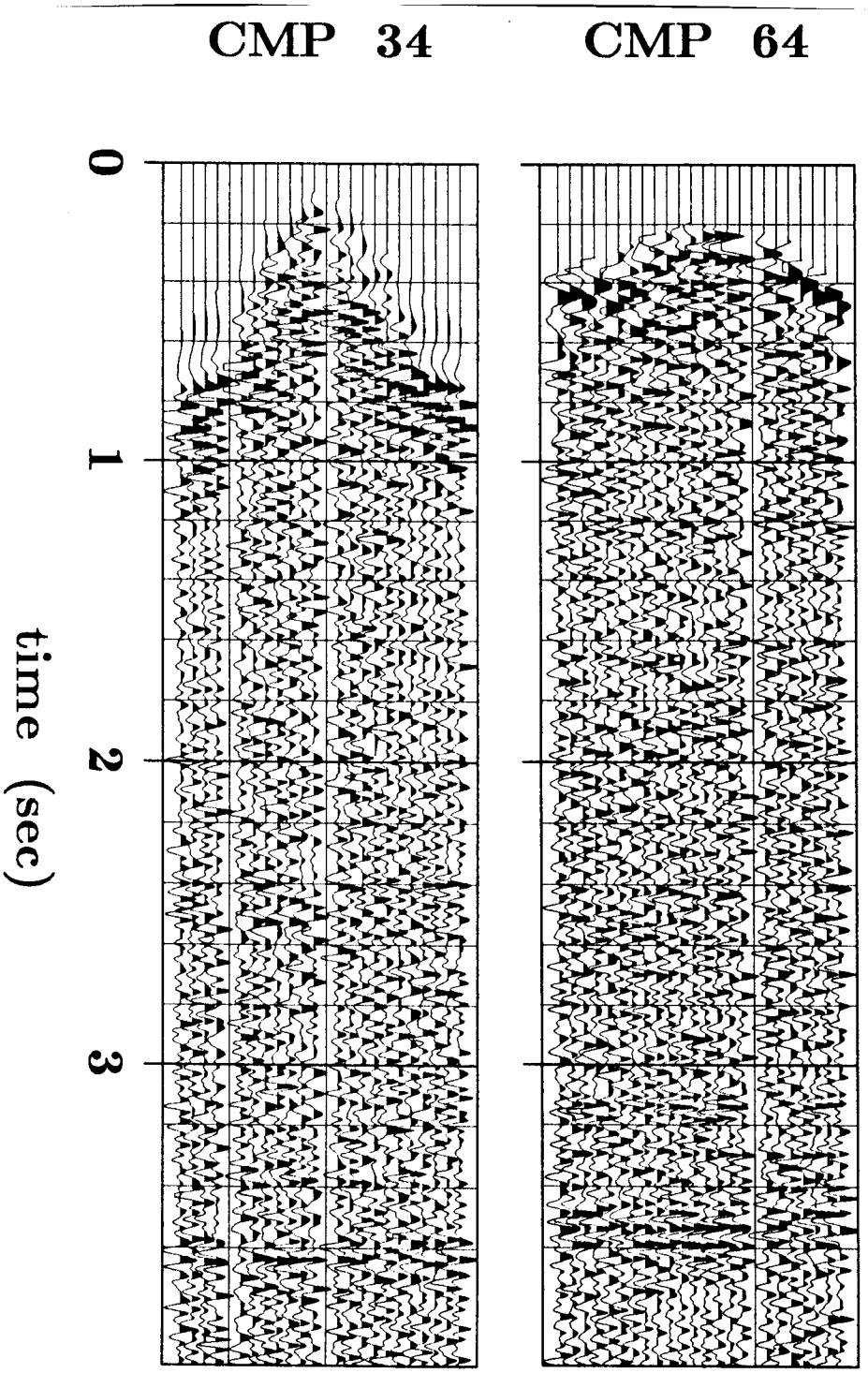

FIG. 4.10. The same two NMO-corrected common-midpoint gathers of Figure 4.7b, now shown after statics corrections from iteration 1835 have been made. Both show substantial alignment after application of the statics solution.

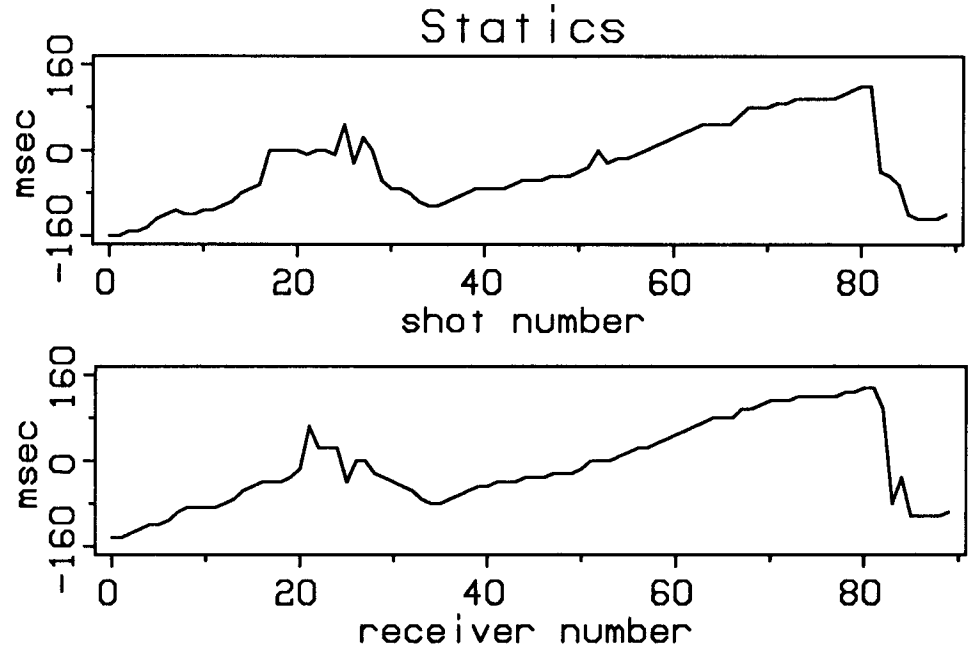

FIG. 4.11. The shot and receiver statics output by iteration 1835. Shots 17-20 and shot 24 were skipped; these are plotted as zeroes. At the right end, statics are as large as 160 ms, which was the upper bound. Examination of the stack in Figure 4.8e in conjunction with these graphs suggests that the "true" statics at the right end are greater than 160 ms; this condition resulted in the artificial discontinuity in the deep reflector in Figure 4.8e.

Figure 4.11 shows the estimates of the shot and receiver statics produced in iteration 1835, the best of the 2000 iterations. The static shifts span the full allowed range of ± 160 ms, and most of the variations are smooth. Examination of the stack in Figure 4.8e in conjunction with the graphs of shot and 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 could find only the best available solution; this constraint 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 be easily corrected. The Monte Carlo statics algorithm was run again; this time the statics of Figure 4.11 were used as the starting guess, and the temperature remained at $T = T_{\min} = .0265$ throughout. 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. After 250 iterations, the stack in Figure 4.12 was produced. The deep reflector is now continuous throughout the

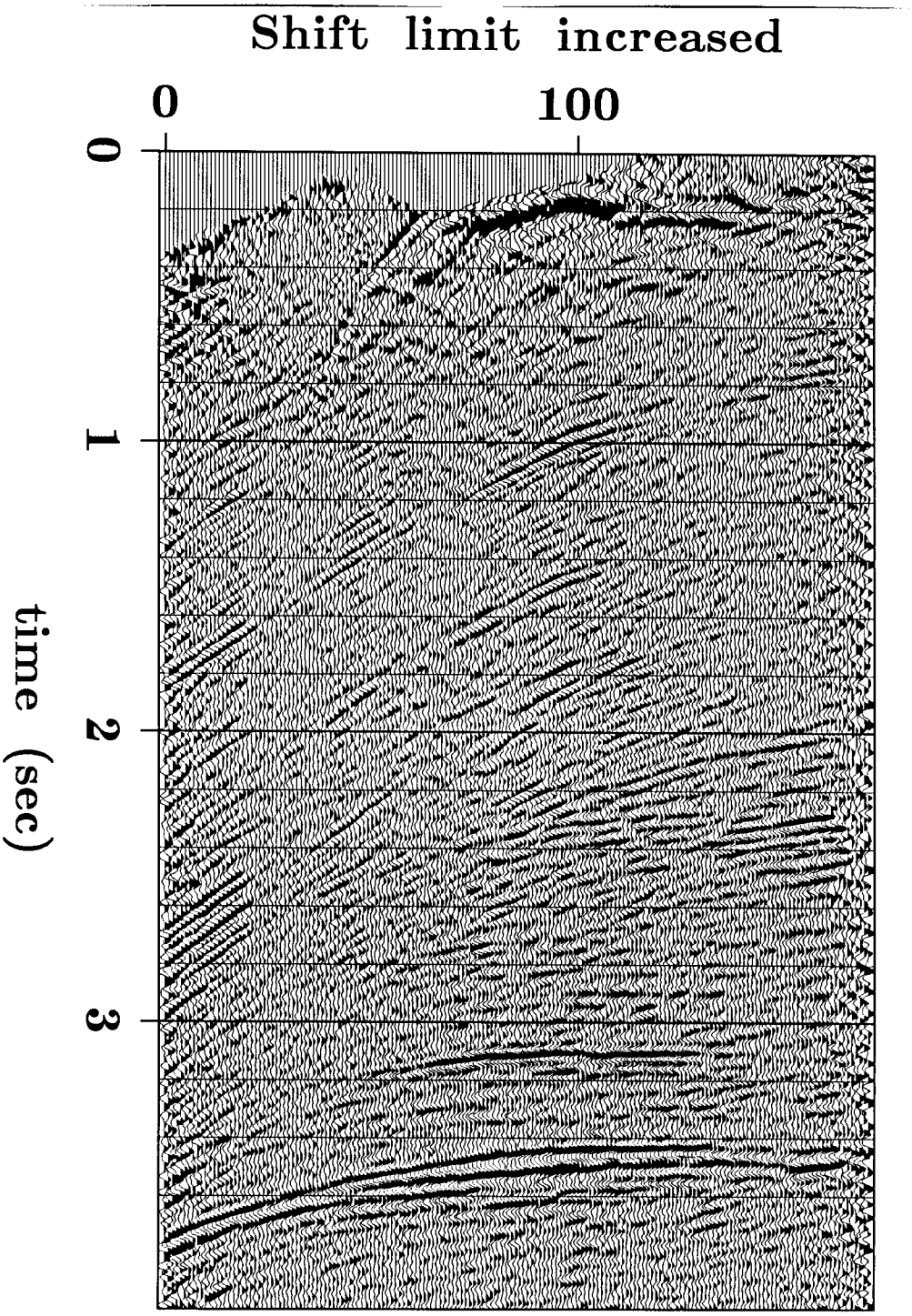

FIG. 4.12. Stack produced after the Monte Carlo statics estimation algorithm was run again, this time using the statics of Figure 4.11 as the starting guess. To allow the estimation of larger statics, the upper bound was doubled to 320 ms, and the estimation window was also doubled to extend from 2 to 4 s. This stack was produced after 250 iterations, in which $T = T_{\min} = .0265$ throughout. Compare this result with the stack in Figure 4.8e: 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.

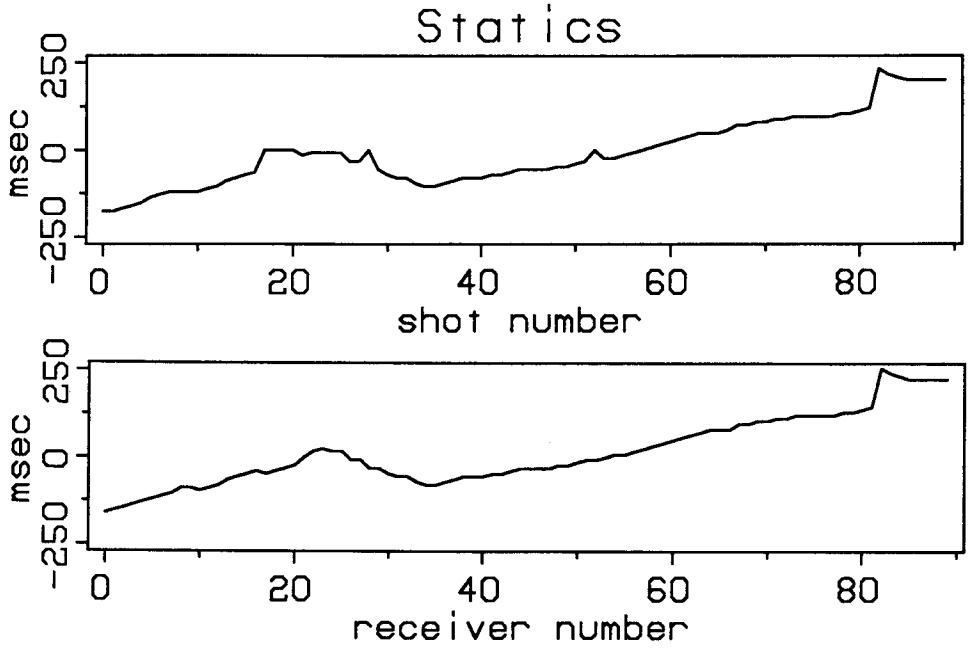

FIG. 4.13. The shot and receiver statics used to produce the stack in Figure 4.12. Statics are now as great as 256 ms; this high level supports the conclusion that the previous upper bound of 160 ms had been too small.

section, and the more shallow reflections also show improved continuity on the right side. Figure 4.13 shows the statics that were used to produce the stack in Figure 4.12. Statics are now as large as 256 ms; this high level supports the conclusion that the upper bound had previously been too small.

The dip reversal of the deep reflector must also be suspect. As discussed earlier with regard to the synthetic data example, long-wavelength components of statics solutions are *nonunique*; that is, different solutions can yield the same stack power. The most obvious nonunique 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 also cannot be 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 CMP 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 4.14 shows the stack produced by subtraction of the linear trend, and Figure 4.15 shows the corresponding shot and receiver statics. The prevailing dip of the deep reflector now matches the input (Figure 4.7a) well. Note that the

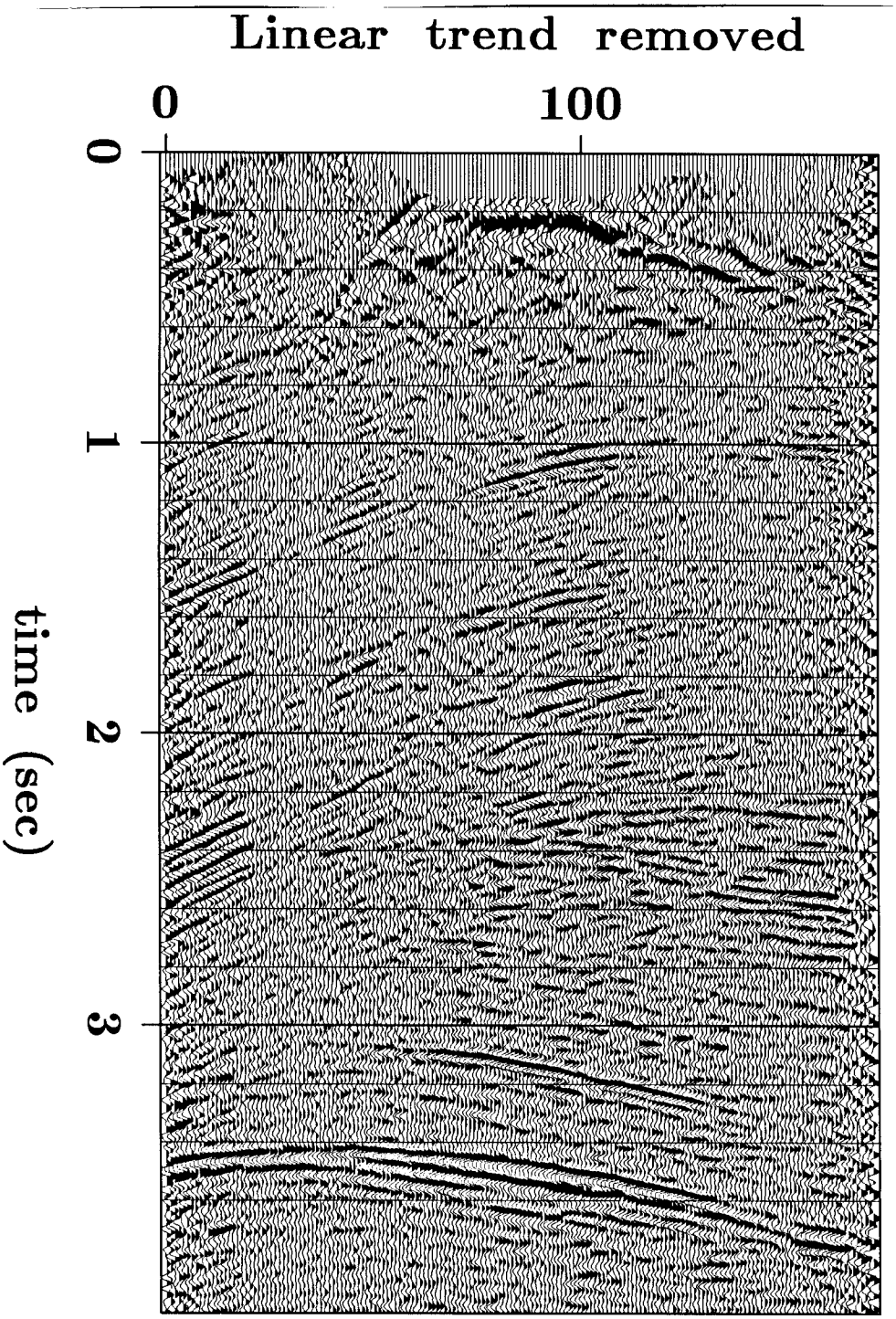

FIG. 4.14. Stack produced after the linear trend in the shot and receiver statics of Figure 4.13 was subtracted. Because the linear trend in the statics is fully unresolved by the data, *any* prevailing dip on the stack is equally as good as any other dip. Elimination of the linear trend thus discards this nonuniquely determined component of the solution. The prevailing dip on the stack now matches the dip on the input stack (Figure 4.7a) well. This result is taken as final solution: note the considerable differences in quality between this stack and Figure 4.7a.

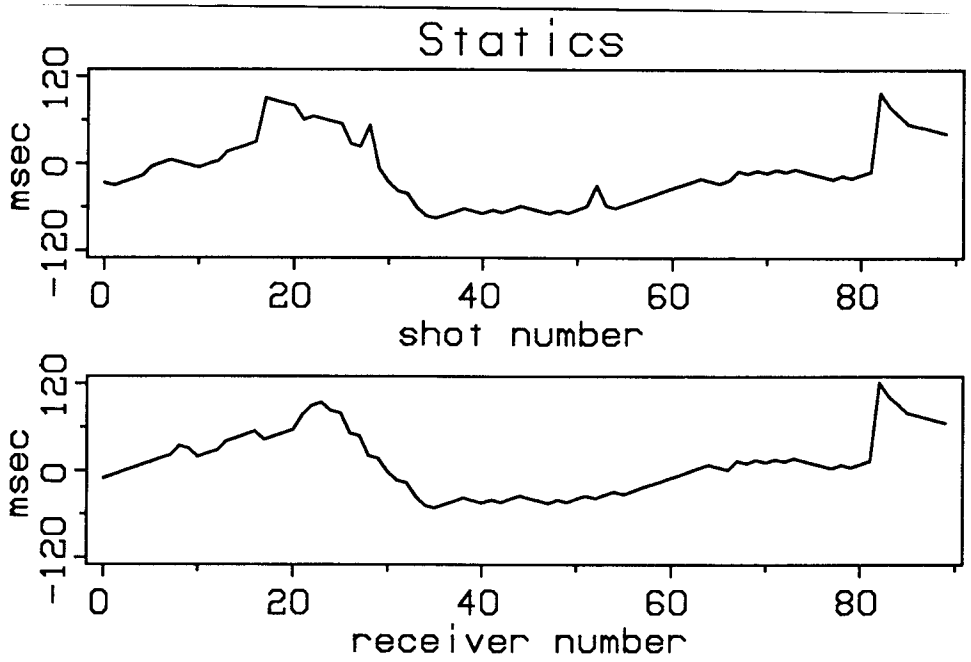

FIG. 4.15. The statics of Figure 4.13, now shown after subtraction of the linear trend. Note that the shape of both the shot and receiver statics is roughly opposite that of the near-surface reflector seen on the input; this shape indicates that this reflection occurred at the base of low-velocity fill.

statics in Figure 4.15 now extend only up to 120 ms. Also, the shape of the statics is roughly opposite that of the near-surface reflector on the input (Figure 4.7a); this shape suggests that this reflection occurred at the base of low-velocity fill.

For comparison with the results of the Monte Carlo algorithm, a test of statics estimation by iterative improvement was run on the data of Figure 4.7a. The test used processing parameters (except for T) identical to those used to generate the stack in Figure 4.8e. The result is shown in Figure 4.16; 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 this result with Figures 4.8e, 4.12, and 4.14. Because the statics for these data are so large, many local minima exist; thus iterative improvement fails because it finds only a nearby local minimum. The stack power for this result is 1.395.

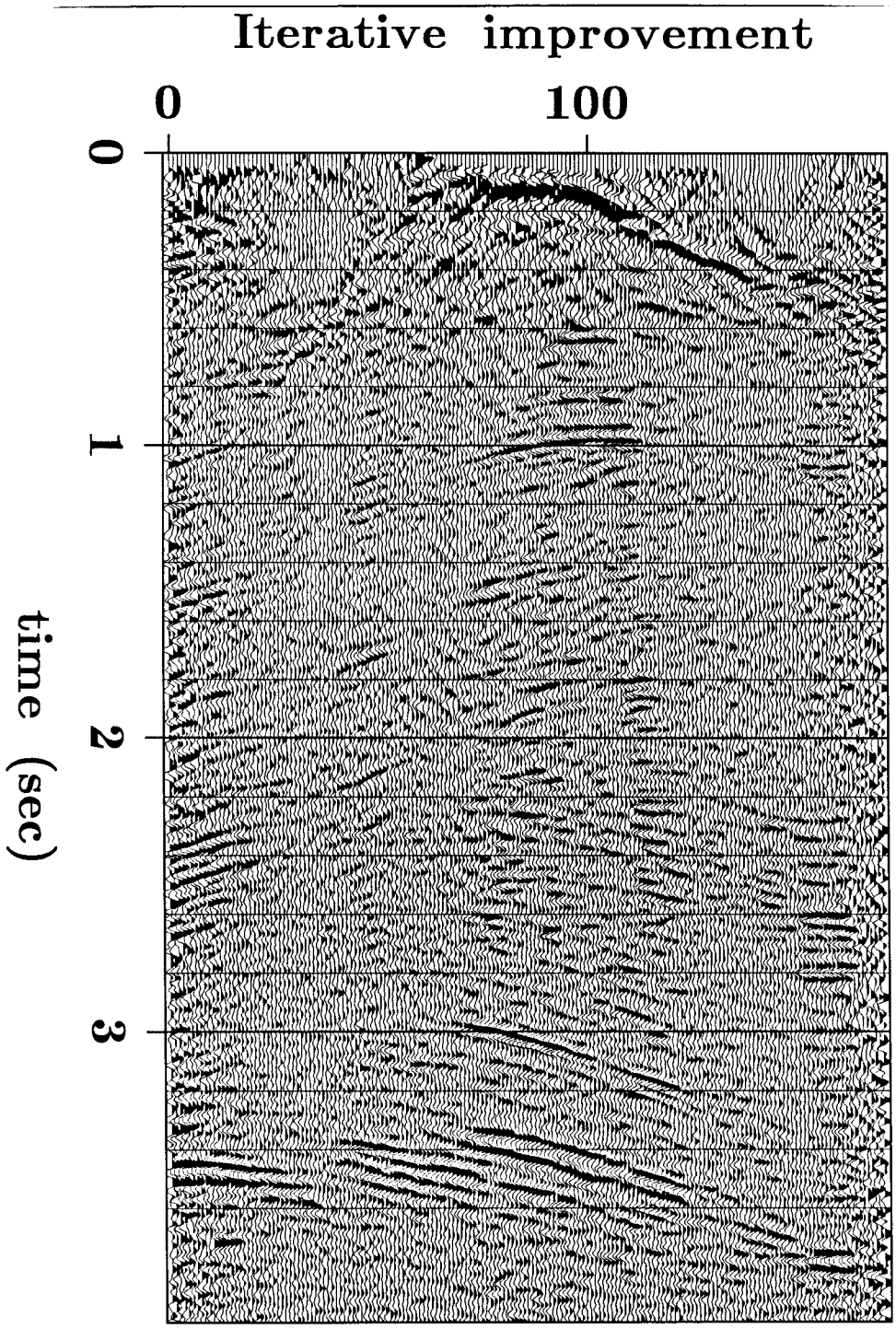

FIG. 4.16. Stack after application of a statics estimated by iterative improvement. All processing parameters (except T) were identical to those used for the example in Figures 4.8a-e. Convergence occurred after only 13 iterations. This stack should be compared with the solution obtained by simulated annealing with non-zero T, shown in Figure 4.8e (and also Figures 4.12 and 4.14). Because iterative improvement converges to a nearby extremum, cycle-skipping can be a problem. Cycle-skipping is evident in the region near 3.5 s, which should exhibit a continuous reflector.

4.3 DISCUSSION

Several interesting characteristics of the Monte Carlo statics algorithm emerge from the examples illustrated in the previous two sections. I detail a few of these features below.

Perhaps the most interesting result of these tests is the rapid, almost discontinuous jumps in power that occurred at approximately iteration 1150 in the field data example and iteration 3000 in the synthetic data example. Looking at the field data stacks in Figures 4.8a-e, one sees that this jump corresponds to a period of rapid change in which the stack is transformed from almost total disorder to a 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 the field data example, however, there was no obvious passage through a *critical temperature* T_c (i.e., the freezing point). I instead chose a $T_{\min} < T_c$ and waited for the system to reach equilibrium. The system eventually fell 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, indicated earlier in §4.2.2, is graphically depicted in Figure 4.8c. 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 (i.e., a statics "bust") is enclosed by data that are relatively free of statics problems. 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 practice, Monte Carlo statics estimation is most efficiently used only to approximate the best stack, not necessarily to find it. In both examples, each static was allowed to be only an integral multiple of the sampling interval; 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 (or, alternatively, T could be set to zero once convergence is obtained). Simulated annealing does not replace conventional statics algorithms when they provide

acceptable solutions, but this Monte Carlo method can be a useful tool when conventional methods fail.

Because statics solutions are nonunique and the Monte Carlo approach typically destroys the first guess, the final solution obtained by this method must be interpreted carefully. Although simulated annealing can discriminate among minima of different depths, it clearly cannot choose among minima of the *same* depth. More particularly, the algorithm cannot resolve parameters that have little or no effect on the final result. Unless otherwise constrained, the algorithm will settle into the first deep minimum it finds, regardless of the magnitude of the nonunique contribution to the solution. Thus, as illustrated by the field data example, dips can change considerably. One way to rectify this problem is to force the poorly determined components of the solution to match some prior notion of what the solution should be; this technique is demonstrated in Figures 4.14 and 4.15, where the linear trend was removed. Another method would be to filter out the nonunique components after each iteration (Ronen and Claerbout, 1985).

Solutions resulting from Monte Carlo statics estimation can be surprising: the method is powerful enough to produce a virtually unpredictable change in the apparent structure in an input stack. In fact, the Monte Carlo method can reveal structure where no previous hint existed. Although the input stack in Figure 4.7a does show clear reflection events, a much improved stack was actually obtained after passage through complete disorder, as illustrated by Figures 4.8a and 4.8b. This improvement attests to the power of the technique.

Aside from the problem of nonuniqueness, the reliability of the statics solution can be measured by comparing the improvements inside the statics estimation window with the possible improvements outside the window. In the field data 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.