Velocity analysis without picking

John Toldi

ABSTRACT

Velocity analysis is posed in this paper as an optimization problem. The paper begins by defining an objective function, which is the measure of how well a velocity model explains the recorded data. The objective function I chose is the power in a common-midpoint stack. This stack is formed by a summation along offset-dependent trajectories, determined by the velocity model. I propose two forms of this objective function: one with trajectories that honor the calculated traveltimes, the other with hyperbolic trajectories. Having defined an objective function, I then look more carefully at its calculation.

This calculation is divided into two steps. The first relates a velocity model to the corresponding traveltimes and stacking slownesses. This step can be easily linearized, so it can be rapidly calculated. The second step uses these traveltimes or stacking slownesses to define a summation trajectory through the data. In this step the traveltime and stacking slowness approaches diverge: the stacking slownesses lead to much simpler and faster algorithms than do the traveltimes.

Then, I develop a steepest-descent algorithm, based on stacking slownesses, that does not require any velocity picking. I apply this algorithm to a field dataset that has a large, near-surface, low-velocity anomaly.

INTRODUCTION

The goal of most velocity analysis methods is a velocity model that can explain either the observed traveltimes or some function that depends on the traveltimes. The conventional method builds this velocity model in a two-step process. The first step forms sums along trajectories that are hyperbolic in time and offset; the "velocity" that maximizes the power in the sum is called the stacking velocity. These stacking velocities are clearly dependent on the traveltimes. The second step transforms these stacking velocities into an earth model, which in this paper is called an interval velocity model.

The conventional method uses the Dix equation to perform this second step. This use is justified by the assumption that the stacking velocity is the root mean square (RMS) velocity; this assumption is valid only for laterally invariant velocities. An alternative to the Dix equation has been described: a linear scheme that relates the laterally varying components of interval velocity to those of stacking velocity (Rocca and Toldi, 1982, Loinger, 1983). The second step of the velocity analysis then consists of using the inverse of the linear relation to determine the interval velocity model that is consistent with the measured stacking velocities.

Methods that work directly with the traveltimes, rather than indirectly through a function of traveltime (such as stacking velocity), present an alternative to the conventional stacking-velocity method. They are similar to the conventional approach, in that they construct a velocity model through a two-stage process. The first stage measures the traveltimes of certain reflectors, usually by cross-correlating time windows of data (those centered on the reflectors of interest). The location of the peak of the cross-correlation is interpreted as the traveltime difference between the two rays that arrive in the correlated time windows. During the second stage, an interval velocity model corresponding to these traveltimes is determined through the use of an inversion procedure, for example tomography.

Both conventional and traveltime-based methods of velocity analysis share one common feature: both are two-step processes, in which some kind of "picking" or interpretation comes between the two steps. This picking may be done automatically, as is usually done for picking the peaks of cross correlations, or manually, as for a conventional velocity analysis.

When the data are contaminated by noise, picking can become difficult; one can easily pick the wrong peak. These picking errors do not obey the gaussian distribution on which the least-squares inversion to the velocity model is based (Rothman, 1984). I propose a method that dispenses with this picking procedure, by posing velocity estimation as an optimization problem. This method is based on the same principle of stack optimization that Shuki Ronen (1984) and Dan Rothman (1984) have used in their work with residual statics.

The first part of this paper discusses the objective function, which is the measure of how well a velocity model explains the data. The objective function I choose is the power in a common-midpoint stack. Most of this first part of the paper discusses the calculation of the objective function for an arbitrary velocity model. Particular attention is paid to methods of simplifying this calculation.

Throughout the first part of the paper, I discuss two forms of the objective function: one based strictly on traveltimes, the other on stacking slownesses. Indeed, within the framework of this paper, the two approaches are very similar; they differ only in the details of how the objective function is calculated. These differences do, however, have important implications for practical velocity analysis algorithms. In particular, the stacking velocity formulation can be simplified in a way that the traveltime formulation cannot.

Thus, the second part of this paper shows the development of a computationally efficient algorithm, based on stacking velocities, that finds an optimum velocity model by repeatedly searching along a gradient direction. I apply this algorithm to a field dataset that has a large near-surface low-velocity anomaly.

THE OBJECTIVE FUNCTION

Consider a parameterized velocity model $\mathbf{m}$. (Throughout this paper, bold characters represent vectors). The parameters might be velocities at a set of gridpoints, or expansion coefficients for some more global set of velocity basis functions. How can we decide if this model is better than some other one?

Ray-based velocity analysis methods provide an answer to this question: the better the match of the raytrace-derived traveltime surfaces to the true traveltime surfaces of the data, the better will be the velocity model. Different methods of determining what constitutes a better fit lead to different objective functions.

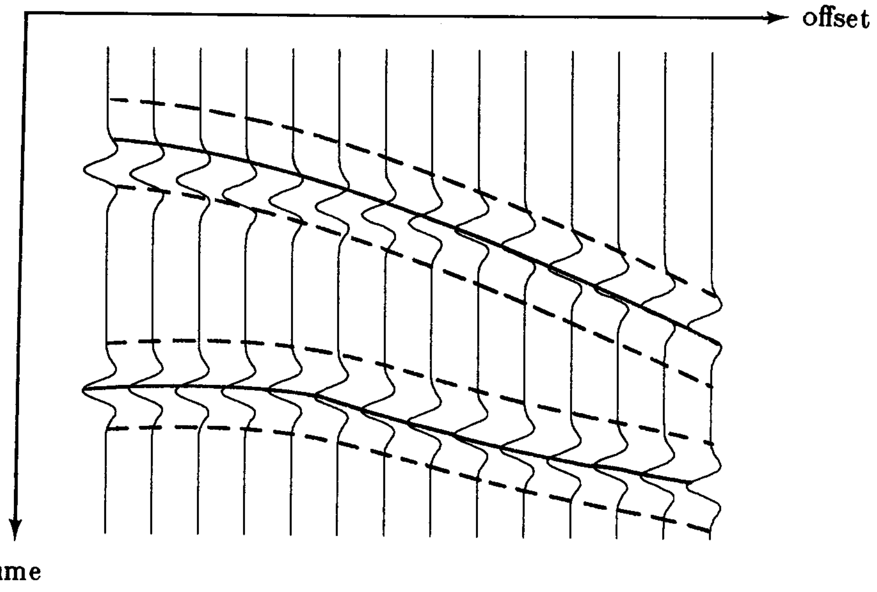

FIG. 1. Common midpoint gather with two reflections. The solid lines are the summation trajectories predicted by the velocity model. The dashed lines indicate the boundaries of the summation's time windows.

Power in the stack

The power in the stack, formed by summing along the calculated traveltime surfaces, is a simple and effective measure of the quality of the fit. This idea is illustrated in Figure 1, which shows a common-midpoint gather that contains two reflections. Raytracing through an assumed velocity model presents traveltime as a function of offset for each of the reflectors in this midpoint gather (the traveltime curves are shown as solid lines in Figure 1). These traveltimes can now be used to define a summation trajectory through the midpoint gather. Because a summation trajectory is valid for a specific reflector, it applies to a time window (Δt_{win} in Figure 1) large enough to include the entire wavelet. The boundaries of the time windows are indicated in dashed lines in Figure 1.

The closer the match of the predicted traveltimes to the true traveltimes, the more the contributions from the different offsets will add in phase. Squaring the sum gives a purely positive function. Thus, the sum over midpoint and reflector of all of the individual squared sums will increase as the velocity model approaches the true one. This total sum is simply the power in the stack.

This sum is - mathematically presented, with $D(y,x,t)$, the data at midpoint y, offset x, and time t, and with $P \equiv$ *power in stack*, as

$$P = \sum_{y} \sum_{t_0} \sum_{\Delta t = -\Delta t_{win}}^{\Delta t = \Delta t_{win}} \left(\sum_{x} D\left[y,x,t(y,x,t_0;\mathrm{m}) + \Delta t\right] \right)^2 \qquad (1)$$

In equation (1), the zero-offset time t_0 can be thought of as a reflector identifier; the sum over t_0 is the sum over reflectors of interest. $t(y, x, t_0; \mathbf{m})$ fully describes the traveltimes predicted by the model **m**.

Sums along hyperbolic trajectories

A simple form of the objective function can be derived by replacing the detailed traveltime trajectories with ones that are hyperbolic over time and offset. Suppose that corresponding to each reflector in a midpoint gather, one finds the hyperbola over offset and time that presents the best least-squares fit to the the calculated traveltimes. This hyperbola is parameterized by the stacking slowness $w = \frac{1}{stacking\ velocity}$, in the well-known NMO equation:

$$t^2 = {t_0}^2 + x^2 w^2 \tag{2}$$

where x is the offset, t_0 the zero-offset time. Now sum over offset along this hyperbolic trajectory and take the square of the sum. Add together these squared sums for all midpoints and reflectors, exactly as is done with the sums from the traveltime trajectories. This total power indicates how well the velocity model explains the traveltimes.

Because the moveouts in figure 1 are clearly non-hyperbolic, the hyperbolic summation trajectories will never pass exactly through all of the datapoints. The key assumption of this method is that the same velocity model leads to maximum power for the hyperbolic trajectories and the traveltime trajectories. That is, the sum along the hyperbola that comes as close as possible to the best set of traveltimes should lead to greater power than a sum along the hyperbola that comes as close as possible to a lesser set of traveltimes.

Unfortunately, there might be sequences of traveltimes that the hyperbolic sums will simply not detect. This is the classic problem of a model null-space: two different models might produce the same value for the objective function. As I have discussed in another paper, much of this problem is avoided if the effects of a specific anomalous zone are viewed from several reflectors (Toldi, 1983). A component of the velocity model, although unable to produce any detectable effect for some reflectors, might have no such problem with the other reflectors.

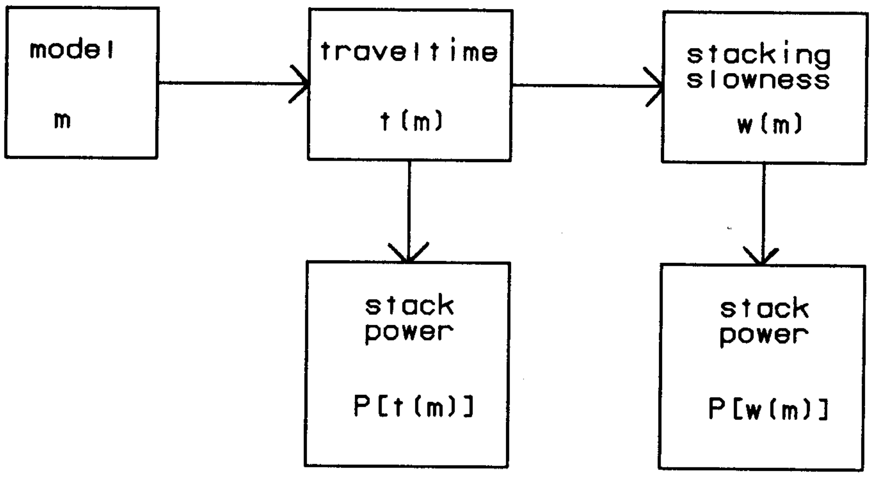

FIG. 2. A velocity model leads to traveltimes and stacking slownesses. These lead to summation trajectories, which provide a test of the quality of the velocity model.

Figure 2 summarizes the discussion so far. A velocity model produces a set of theoretical traveltimes or stacking slownesses. For each reflector and midpoint, these traveltimes or stacking slownesses determine a summation trajectory over offset. The total power, which is formed by the squaring and the adding together of the individual sums for the entire dataset, is a measure of the quality of the velocity model.

SIMPLIFYING THE CALCULATION OF $P(\mathbf{m})$

A velocity analysis algorithm will necessarily involve the repeated evaluation of the objective function discussed in the last section. As presented thus far, the algorithm would be quite impractical. Each of the two steps of Figure 2--the calculation of traveltimes or stacking slownesses for a velocity model, and the calculation of the power in the appropriate stack--can, however, be simplified.

Simplifying step one - linear theory

The theory discussed thus far assumes that there is a non-linear relationship between the model $\mathbf{m}$ and the intermediate data $\mathbf{d}$ (which may be either traveltime $\mathbf{t}$ or stacking slowness $\mathbf{w}$). Thus, for each change in the model one would have to redo the raytracing. Furthermore, for the stacking slowness method, this raytracing would be followed by a new least-squares fit through the relevant traveltimes. If the change in the model is fairly small, a linear approximation to the relationship between $\mathbf{d}$ and $\mathbf{m}$ is valid.

Changing the model from $\mathbf{m}_1$ to $\mathbf{m}_2$ leads to a new traveltime $\mathbf{t}$,

$$\mathbf{t}(\mathbf{m}_2) \approx \mathbf{t}(\mathbf{m}_1) + \frac{\partial \mathbf{t}}{\partial \mathbf{m}} (\mathbf{m}_2 - \mathbf{m}_1) \tag{3}$$

As is well known in seismic tomography (Aki and Richards, 1980, Fawcett, 1983), the partial derivative matrix $\frac{\partial \mathbf{t}}{\partial \mathbf{m}}$ is easily calculated with the help of Fermat's principle. The change in the j^{th} component of the traveltime, (j refers to a specific reflector, offset and midpoint) due to a small perturbation in the slowness model is simply:

$$\Delta \mathbf{t}_j \;=\; \int_{S_j} \Delta w(y,z)\; dS_j$$

where S_j is the unperturbed raypath for the ray indexed by j, and Δw the perturbation in slowness.

The stacking slowness method has an additional complication: the non-linearity of the relationship between traveltime and stacking slowness. This relationship can also be linearized (Loinger, 1983, Toldi and Rocca, 1982). Specifically, one considers how the stacking slowness (the slope of the best fit line in $t^2 - x^2$ space), would change if one of the traveltimes within the midpoint gather were changed slightly. This second linearization provides the matrix $\frac{\partial \mathbf{w}}{\partial \mathbf{t}}$. Thus,

$$\mathbf{w}(\mathbf{m}_2) \approx \mathbf{w}(\mathbf{m}_1) + \frac{\partial \mathbf{w}}{\partial \mathbf{m}} (\mathbf{m}_2 - \mathbf{m}_1) \tag{4}$$

$$= \mathbf{w}(\mathbf{m}_1) + \frac{\partial \mathbf{w}}{\partial \mathbf{t}} \frac{\partial \mathbf{t}}{\partial \mathbf{m}} (\mathbf{m}_2 - \mathbf{m}_1) \tag{5}$$

$\frac{\partial \mathbf{t}}{\partial \mathbf{m}}$ is the same matrix as that in the traveltime problem. Thus, with $\Delta \mathbf{w} \equiv \mathbf{w}(\mathbf{m}_2) - \mathbf{w}(\mathbf{m}_1)$ and $\Delta \mathbf{t} \equiv \mathbf{t}(\mathbf{m}_2) - \mathbf{t}(\mathbf{m}_1)$, combining equations (3) and (5) gives:

$$\Delta \mathbf{w} \;=\; \frac{\partial \mathbf{w}}{\partial \mathbf{t}} \Delta \mathbf{t}$$

That is, the stacking slowness perturbations are just a filtered version of the traveltime perturbations. Note that the multiplication of the two partial derivative matrices in equation (5) collapses the offset dimension. A detailed derivation of $\frac{\partial \mathbf{w}}{\partial \mathbf{m}}$ is given in Rocca and Toldi, 1982.

Simplifying step two - stacking slowness method

The linearization presented in the last section can greatly simplify the calculations of traveltimes and stacking slownesses. This section examines how the second step--stacking according to a calculated trajectory--can be simplified.

A conventional velocity analysis program provides semblance as a function of midpoint y, zero-offset time t_0, and stacking velocity v_s. I prefer to use stacking slowness, $w = \frac{1}{v_s}$, in place of stacking velocity. Note that semblance then takes the place of power in the stack.

Semblance is similar to power in the stack; its calculation includes an additional normalization (the individual squared sums over offset are normalized by the sum of the squares over offset). The real goal is to measure alignment along some curve, and semblance provides an effective measure. This cube of data, *semblance* $= S(y, t_0, w)$, is ideal for use with a velocity analysis program based on an overall optimization principle.

Consider again the velocity analysis algorithm of the last section. By some as yet unspecified means, a model $\mathbf{m}$ is chosen. Corresponding to this model are a set of theoretical stacking slownesses, $\mathbf{w}$, such that,

$$\left[\mathbf{w}(\mathbf{m})\right]_j = w(y, t_0; \mathbf{m})$$

The index j ranges over all combinations of midpoint y and zero-offset time t_0. The total power P corresponding to the model $\mathbf{m}$ is

$$P(\mathbf{m}) = \sum_{y, t_0} S(y, t_0, w(y, t_0; \mathbf{m})) \tag{6}$$

The key point is that $S(y, t_0, w)$ has already been calculated for all y, t_0, and w in the conventional semblance analysis. Thus, evaluating the objective function for a new model simply means recalculating $\mathbf{w}(\mathbf{m})$, then summing through the semblance cube along these new $\mathbf{w}$ curves. Of course, because the $\mathbf{w}(\mathbf{m})$ values may not fall exactly on computed semblance values, an interpolation in the slowness direction is necessary.

Of the four sums found in equation (1), (over y, t_0, Δt_{win}, and x) only two--the sums over y and t_0--remain in equation (6). The sums over offset x and time window Δt_{win} are computed in the initial semblance analysis. The absence of these sums from the active velocity-analysis loop significantly simplifies the repeated evaluation of the objective function. At the cost of the initial overhead of the semblance analysis, one achieves a process that can much more readily be made interactive.

Simplifying step two - traveltime method

It is natural to ask if such a simplification is possible for a traveltime based method. The answer is a qualified maybe. The analog to the stacking-slowness scan would be a scan over traveltime for each of the trace segments in a common-midpoint gather. The basic difficulty with traveltimes is that the stack depends on the traveltimes predicted for all of the offsets. Thus, the scan over traveltime for one trace would correspond to all of the other offsets having their traveltime fixed.

To form the full set of scans, one would need to look at all combinations of shifts for the traces in the gather. The number of stacks formed in this manner would be enormous: allowing each of the n_x traces in the midpoint gather to scan through n_{lag} different traveltimes would lead to $(n_x)^{n_{lag}}$ different stacks. Thus, the velocity-scan method that proved to be so useful in the stacking-slowness method has no simple analog in the traveltime method.

Instead of adding the time segment to a stack trace formed with the current estimates of traveltime, one can add it to a model trace that remains constant throughout the process. Such a method, which is similar to the pilot-trace schemes commonly found in residual statics, allows the problem to be reduced to a more reasonable size. A traveltime scan used in that scheme would produce only $nx * nlag$ stack traces for each event in a midpoint gather.

Flat reflectors - a practical detail

For a flat reflector, the zero-offset time will vary only moderately as a function of midpoint. By taking the semblance analysis within a time window of some average zero-offset time and summing over time, one can collapse the time axis of the semblance cube into a few points, one time point for each reflector. The end result is a semblance cube that consists of a few planes, one for each reflector, with axes of slowness and midpoint.

Figure 3 shows one such plane for a shallow reflector on a dataset with a near-surface low-velocity anomaly. This is essentially a horizon velocity analysis: stacking slowness versus midpoint for a specific reflector. Such an analysis can likewise be done for a sloping or undulating reflector. The flat reflector has a horizontal time window; the time window for a curved reflector follows along the zero-offset times.

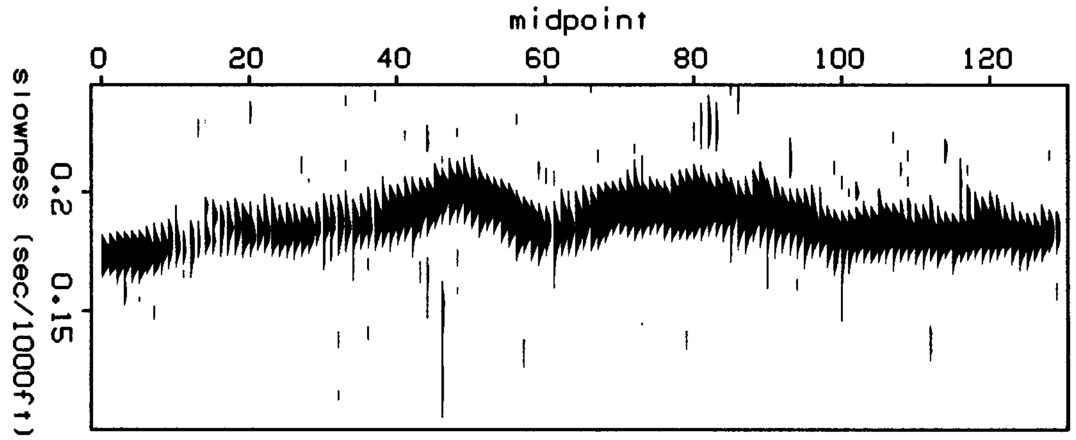

FIG. 3. Semblance as a function of midpoint and stacking slowness for a flat reflector.

OPTIMIZATION METHODS

The first part of this paper has attempted to answer the question: how can we know if one velocity model is better than another? The answer is to define an objective function, here the power in a common midpoint stack. But knowing how to compare two velocity models is only the beginning of the solution. How do we choose that second velocity model?

Methods using no derivative information

Optimization methods can be classified according to how they choose the second model point. In general, the key consideration in the classification is the level of information used about the derivatives of the objective function. Thus, the simplest methods use no derivative information at all; they simply evaluate the objective function repeatedly. Non-derivative methods are important when the derivatives are either difficult to calculate or unreliable.

The very simplest such method would simply try every possible model. For the velocity analysis problem such a method is clearly out of the question. (If every one of the 100 parameters of a velocity model were allowed to try any of 50 values, 100^{50} models would be tested.)

There are cleverer ways to search through the models than the simple exhaustive search. Dan Rothman (1984) and Shuki Ronen (1984) have applied different non-derivative methods to the residual statics problem, with considerable success. Residual statics is a special case of traveltime-based velocity analysis; it has a very simple linear relation between the model parameters--the shot and geophone statics--and the traveltimes.

A velocity analysis method based on traveltimes could likewise be designed as a non-derivative search method. Such an algorithm would be time consuming: the relationship between the model and the traveltimes is considerably more complicated for a full velocity model than for a statics model. A very practical non-derivative method of velocity analysis could result from using stacking slownesses. The simplified evaluation of the objective function that the use of stacking slownesses permits could prove extremely useful.

The next class of optimization methods use first-derivative information. The remainder of this paper develops a velocity analysis method of this sort.

Gradient methods

The gradient of the objective function, taken with respect to the model parameters, indicates the direction in which there is the maximum increase in the value of the objective function. A natural choice for the next model point is somewhere along this gradient direction. Let $P(\mathbf{m})$ be the objective function, evaluated for model $\mathbf{m}$. Furthermore, let $\nabla_{\mathbf{m}} P$ be the gradient of the objective function with respect to the model parameters. A simple gradient descent algorithm (steepest descent) proceeds as follows:

1. At a given model point $\mathbf{m}_1$, form $\nabla_{\mathbf{m}} P$
2. Search for α that maximizes $P(\mathbf{m}_1 + \alpha \nabla_{\mathbf{m}} P)$
3. Update $\mathbf{m}_1$ by setting $\mathbf{m}_1 = \mathbf{m}_1 + \alpha \nabla_{\mathbf{m}} P$
4. If the algorithm has not converged, go to 1.

In velocity analysis, $P(\mathbf{m})$ is calculated through an intermediary, for example stacking slowness. That is, $P(\mathbf{m})$ is more properly $P[\mathbf{w}(\mathbf{m})]$. The gradient can then be expressed, with the help of the chain rule of partial differentiation, as

$$(\nabla_{\mathbf{m}} P)_i = \frac{\partial P}{\partial m_i} \tag{7}$$

$$= \sum_{j=0}^{j=n_y * n_{t_0}} \frac{\partial P}{\partial w_j} \frac{\partial w_j}{\partial m_i} \tag{8}$$

All derivatives are evaluated at the current model point $\mathbf{m}$; n_y = number of midpoints, and n_{t_0} = number of reflectors.

The derivative matrix $\mathbf{G}$ can be defined, with elements,

$$(\mathbf{G})_{kl} = \frac{\partial w_k}{\partial m_l} \qquad (9)$$

With the substitution of this definition, equation (8) becomes

$$(\nabla_{\mathbf{m}} P)_i = \sum_{j=0}^{j=n_y * n_{t_0}} \frac{\partial P}{\partial w_j} (\mathbf{G})_{ji} \qquad (10)$$

or, in matrix notation,

$$\nabla_{\mathbf{m}} P = \mathbf{G}^T \nabla_{\mathbf{w}} P \qquad (11)$$

The earlier discussion on linearization showed how to calculate the derivative matrix $\mathbf{G}$. I calculate $\nabla_{\mathbf{w}} P$ by using a finite-difference method. Because the velocity scans have provided $P(\mathbf{w})$ for all $\mathbf{w}$, this calculation is easily performed.

Before proceeding to the discussion of the finite-difference derivative, I find it helpful to compare the algorithm presented here with a method based on picked peaks. Using the conventional formulation, one would look for the model that provides the best fit (in a least-squares sense) to the picked stacking slownesses. Because of the varying quality of the data, a weighted-least-squares method is appropriate.

One can approximate the objective function by a quadratic function about $\mathbf{w}_{\mathrm{p}}$, the picked stacking slownesses. The weights, $\mathbf{C}_{\mathrm{d}}^{-1/2}$, are related to the width and height of the peaks, and thus contain the information about how accurately the peak could be picked. That is,

$$P(\mathbf{w}) \approx -(\mathbf{w} - \mathbf{w}_{\mathrm{p}})^T \, \mathbf{C}_{\mathrm{d}}^{-1} (\mathbf{w} - \mathbf{w}_{\mathrm{p}}) \qquad (12)$$

Using the conventional approach, one would have

$$\nabla_{\mathbf{w}} P = -2\mathbf{C}_{\mathrm{d}}^{-1} (\mathbf{w} - \mathbf{w}_{\mathrm{p}}) \qquad (13)$$

for $\nabla_{\mathbf{w}} P$ in equation (11). Then $\mathbf{w}$ could be calculated with the help of the linear theory. Thus, the difference between a conventional gradient-based velocity analysis and the method proposed here is only the way in which $\nabla_{\mathbf{w}} P$ is calculated.

The finite-difference derivative is easily calculated. Recall from equation (6) that

$$P(\mathbf{w}(\mathbf{m})) = \sum_{y,t_0} S(y, t_0, w(y, t_0; \mathbf{m})) = \sum_j S_j(w_j(\mathbf{m})) \qquad (14)$$

The index j in equation (14) is used to identify y and t_0. Thus, $S_j(w)$ is the semblance at y and t_0, as a function of slowness w. With this notation,

$$(\nabla_{\mathbf{w}} P)_j = \frac{\partial P}{\partial w_j} \approx \frac{S_j\left[w_j(\mathbf{m}) + \Delta w\right] - S_j\left[w_j(\mathbf{m})\right]}{\Delta w} \tag{15}$$

Δw is a unit change in stacking slowness.

Because the finite-difference method deals directly with the data, it incorporates the weights automatically. Specifically, the higher and sharper the peaks are, the larger the gradient (equation (15)) will be. Thus, the parts of the data with high and sharp peaks will dominate the solution. A certain amount of smoothing of the semblance cube along the slowness axis is desirable; the finite-difference derivative should be made insensitive to the jitter of small-scale noise.

When the model places the stacking slowness in the vicinity of a strong peak, the quadratic approximation will be valid: both the finite difference and the picked-peak methods should provide similar results. Sufficient smoothing of the semblance cube along the slowness axis will allow the finite-difference derivatives to sense the peaks from a greater distance. Ideally, the descent algorithm would start with a long smoother, then gradually shorten it as the algorithm converged.

Used with either the picked or finite-difference methods, the weights insure that the high-quality parts of the data dominate the early stages of the descent algorithm. The model formed at these early stages predicts stacking slownesses for all of the data; the approximate location of the stacking slowness curves will be controlled by the good parts of the data. Note that this resolves the issue of which peak to pick in zones of poor quality data: the finite-difference algorithm will be working on the peak that is most consistent with the model determined by the good quality parts of the data. The conventional approach would still be trying to fit the peak that was originally picked.

The automatic weighting device of the finite-difference method also proves to be important for reflectors that are not continuous across an entire section. When a reflector within a time window dies out, the smoothed objective function will be almost completely flat. Thus, the relevant midpoints of that time window will contribute almost nothing to the descent algorithm.

The linear approximation to the relationship between $\mathbf{w}$ and $\mathbf{m}$ further simplifies the algorithm,

$$\mathbf{w}(\mathbf{m}) \approx \mathbf{w}(\mathbf{m}_0) + \frac{\partial \mathbf{w}}{\partial \mathbf{m}}(\mathbf{m} - \mathbf{m}_0) \tag{16}$$

$$\approx \mathbf{w}(\mathbf{m}_0) + \mathbf{G}\,(\mathbf{m} - \mathbf{m}_0) \tag{17}$$

The partial-derivative matrix $\mathbf{G}$ in equation (17) is evaluated at the model point $\mathbf{m}_0$.

As the descent algorithm moves between points, $\mathbf{G}$ should be recalculated. On the other hand, the linearization will typically be valid for a fairly wide range of model points $\mathbf{m}$. The real benefits of linearizing the relationship between $\mathbf{m}$ and $\mathbf{w}$ follow from this second observation: $\mathbf{G}$ need only rarely be recalculated. Thus the raytracing that goes into the calculation of $\mathbf{G}$ will be done infrequently.

I assume in this paper that the starting value of $\mathbf{G}$ is valid for the entire process. Because my starting model is laterally invariant, the forward modeling (i.e. calculating $\mathbf{w}$ given $\mathbf{m}$) is particularly efficient.

VELOCITY ANALYSIS WITHOUT PICKING

Theory

The discussion of the previous section can now be summarized in the following velocity analysis algorithm:

```
Set m to starting model: $\mathbf{m} = \mathbf{m}_0$
Set w to starting value: $\mathbf{w} = \mathbf{w}(\mathbf{m}_0) = \mathbf{w}_0$

Begin loop on iterations
    Form $\nabla_{\mathbf{m}} P$ at current model point m:
        $(\nabla_{\mathbf{w}} P)_j = \frac{S_j(w_j + \Delta w) - S_j(w_j)}{\Delta w}$
        $\nabla_{\mathbf{m}} P = \mathbf{G}^T \nabla_{\mathbf{w}} P$
    Line search for $\alpha$ that maximizes $P(\mathbf{m} + \alpha \nabla_{\mathbf{m}} P)$
        $P(\mathbf{m} + \alpha \nabla_{\mathbf{m}} P) = P[\mathbf{w}(\mathbf{m} + \alpha \nabla_{\mathbf{m}} P)]$
                          $= P[\mathbf{w}_0 + \mathbf{G}(\mathbf{m} - \mathbf{m}_0) + \alpha \mathbf{G} \nabla_{\mathbf{m}} P]$
    Update model
        $\mathbf{m} = \mathbf{m} + \alpha \nabla_{\mathbf{m}} P$
        $\mathbf{w} = \mathbf{w} + \alpha \mathbf{G} \nabla_{\mathbf{m}} P$
End loop on iterations
```

Note that the algorithm searches for the α that maximizes $P[\mathbf{w}_0 + \mathbf{G}(\mathbf{m} - \mathbf{m}_0) + \alpha \mathbf{G}\nabla_{\mathbf{m}} P]$. This is a simple search of the form: find α to maximize $P(\mathbf{a} + \alpha \mathbf{b})$, where $\mathbf{a}$ and $\mathbf{b}$ are known vectors. The linear theory is, of course, responsible for this simple result. Using the fully non-linear relationship between $\mathbf{w}$ and $\mathbf{m}$ requires that $\mathbf{w}(\mathbf{m}) = \mathbf{w}(\mathbf{m}_0 + \alpha \nabla_{\mathbf{m}} P)$ be recalculated for each value of α. Once again, when using an iterative linearization scheme one would want to retain the form of **G** for as wide a range of **m** as possible.

Application to field dataset

The descent algorithm can best be shown by an example. Figure 4 is a stacked section of a field dataset. The time sag and the lack of coherence beneath midpoint 125, indicate a near-surface low-velocity zone. I calculated interval slownesses for this dataset, in an accompanying paper in this report, using the form of the algorithm based on picked stacking slownesses. At the end of this section is presented a comparison between the velocity model derived here and that derived from the picked stacking slownesses.

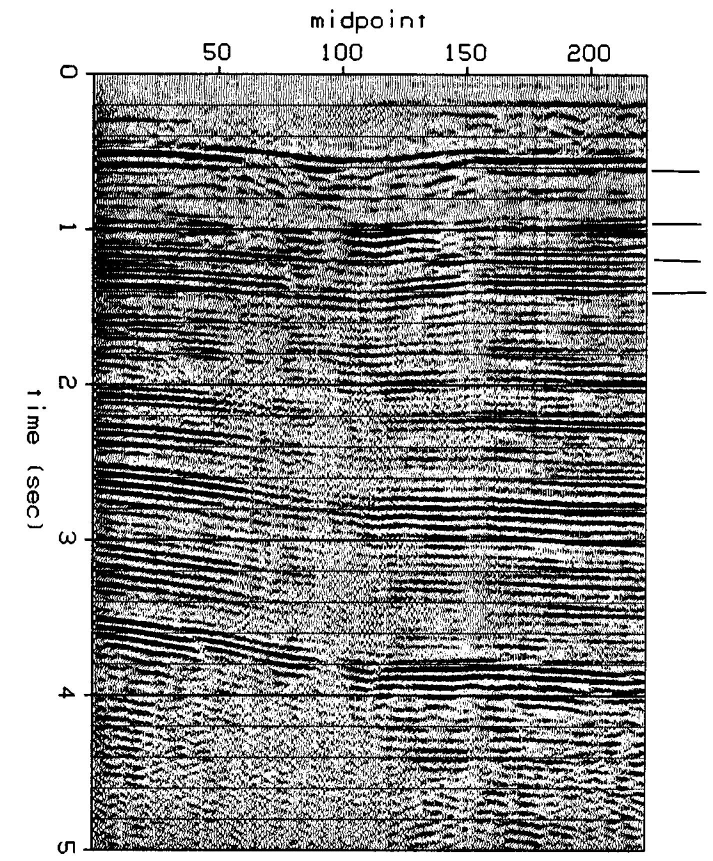

FIG. 4. CMP stack of field data from the Central Valley of California. The time sag and the lack of coherence beneath midpoint 125, indicate a near-surface, low-velocity zone. The tick marks on the right show the location of the reflectors that I used in the inversion.

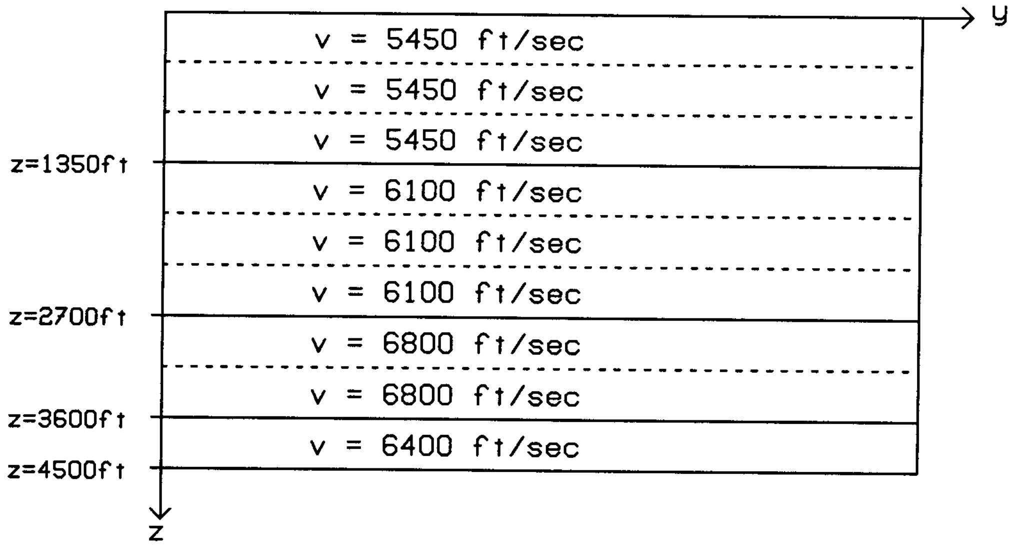

FIG. 5. The starting model for the descent algorithm. The model has nine layers. The solid lines show the locations of the reflectors that supplied the stacking slownesses.

The starting velocity model for the descent algorithm is shown in Figure 5. The model consists of nine layers; three are above the shallowest reflector. I derived the laterally-invariant interval velocities shown in Figure 5, by doing a conventional velocity analysis on a midpoint gather some distance from the anomaly, then using the Dix equation to relate stacking velocity to interval velocity.

Figure 6 shows contour plots of semblance as a function of stacking slowness and midpoint, for the time planes that I used in the velocity analysis. The reflectors are at times .5, .94, 1.16, and 1.34 sec (see also Figure 4). The solid line through each of the planes is the stacking slowness corresponding to my starting model. The goal of the velocity analysis will be to construct a model that warps these stacking slowness curves towards the peaks of Figure 6. By so doing, the analysis will be maximizing the power in the stack. Previous velocity analysis algorithms (e.g., those in my other paper in this report) required that I pick that maximum by hand; here the solution will be driven toward the peaks as the iterations progress.

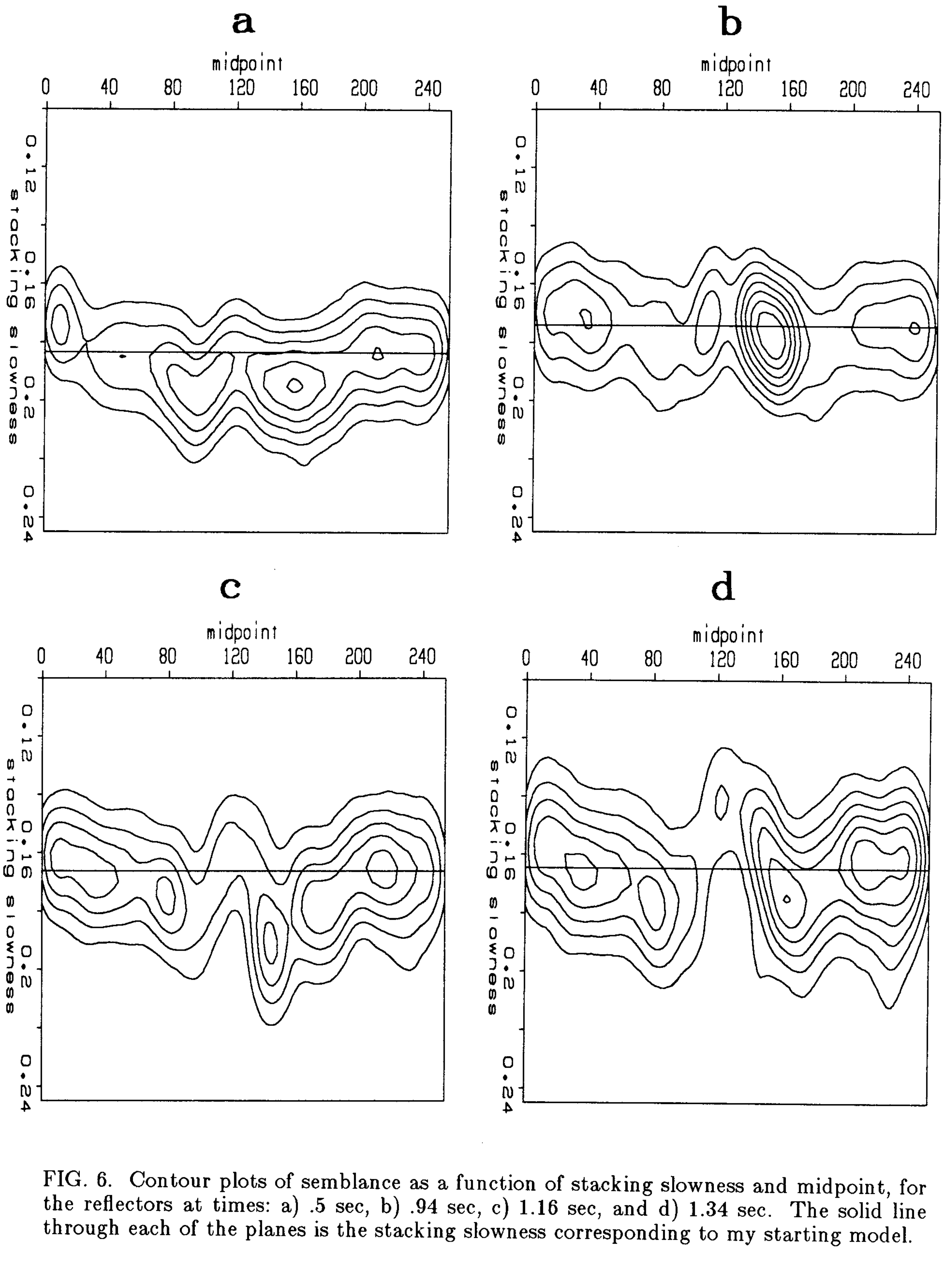

FIG. 6. Contour plots of semblance as a function of stacking slowness and midpoint, for the reflectors at times: a) .5 sec, b) .94 sec, c) 1.16 sec, and d) 1.34 sec. The solid line through each of the planes is the stacking slowness corresponding to my starting model.

Figure 7 shows the same contour plots as were in figure 6, only now the stacking slownesses corresponding to the successive models in the iterative process are drawn on as solid lines. The final stacking slowness is drawn as a thick line. After 20 iterations, the stack power was changing imperceptibly.

The two strongest reflectors, those at .5 and 1.34 sec, have the stacking slowness curves that come closest to the peaks. This is exactly what one would have sought to accomplish by introducing weights into the system; here it has happened automatically.

The interval velocity model derived by the iterative method is shown in Figure 8. It actually shows the change in the interval velocity from the starting model to the final model. A large shallow anomaly is clearly visible; Figure 8 shows it to be a positive slowness (low velocity) anomaly.

Figure 9 shows the velocity anomaly that I determined by inverting picked stacking slownesses. This result is from my other paper in this report (Toldi, 1984). Clearly, the iterative algorithm has produced a result that is very similar to that derived from the picked stacking slownesses.

I examine the velocity model of Figure 9 in great detail in my other paper in this report. I will only summarize the results from the other paper, because of the similarity of the two derived models, The analysis shows that after the effects of the laterally variable part of the calculated interval velocities have been removed, the traveltimes and velocities are almost laterally invariant. Thus, the models shown in Figures 8 and 9 are quite successful at explaining the laterally variable velocities in the field dataset.

FURTHER DISCUSSION

Accelerated methods

It is well known that steepest descent (or ascent) algorithms can converge quite slowly (Luenberger, 1973). Various optimization methods are available that converge more rapidly: among them, the conjugate gradient algorithms are particularly attractive. I am currently doing some preliminary tests with a conjugate gradient algorithm.

A priori information

The algorithm presented here does not allow a-priori information to be incorporated. This can be easily remedied by the addition of an extra term to the objective function. One can define a new objective function P' $(\mathbf{m})$

$$P'(\mathbf{m}) = P(\mathbf{m}) - (\mathbf{m} - \mathbf{m}_0)^T \mathbf{C_m}^{-1} (\mathbf{m} - \mathbf{m}_0) \qquad (18)$$

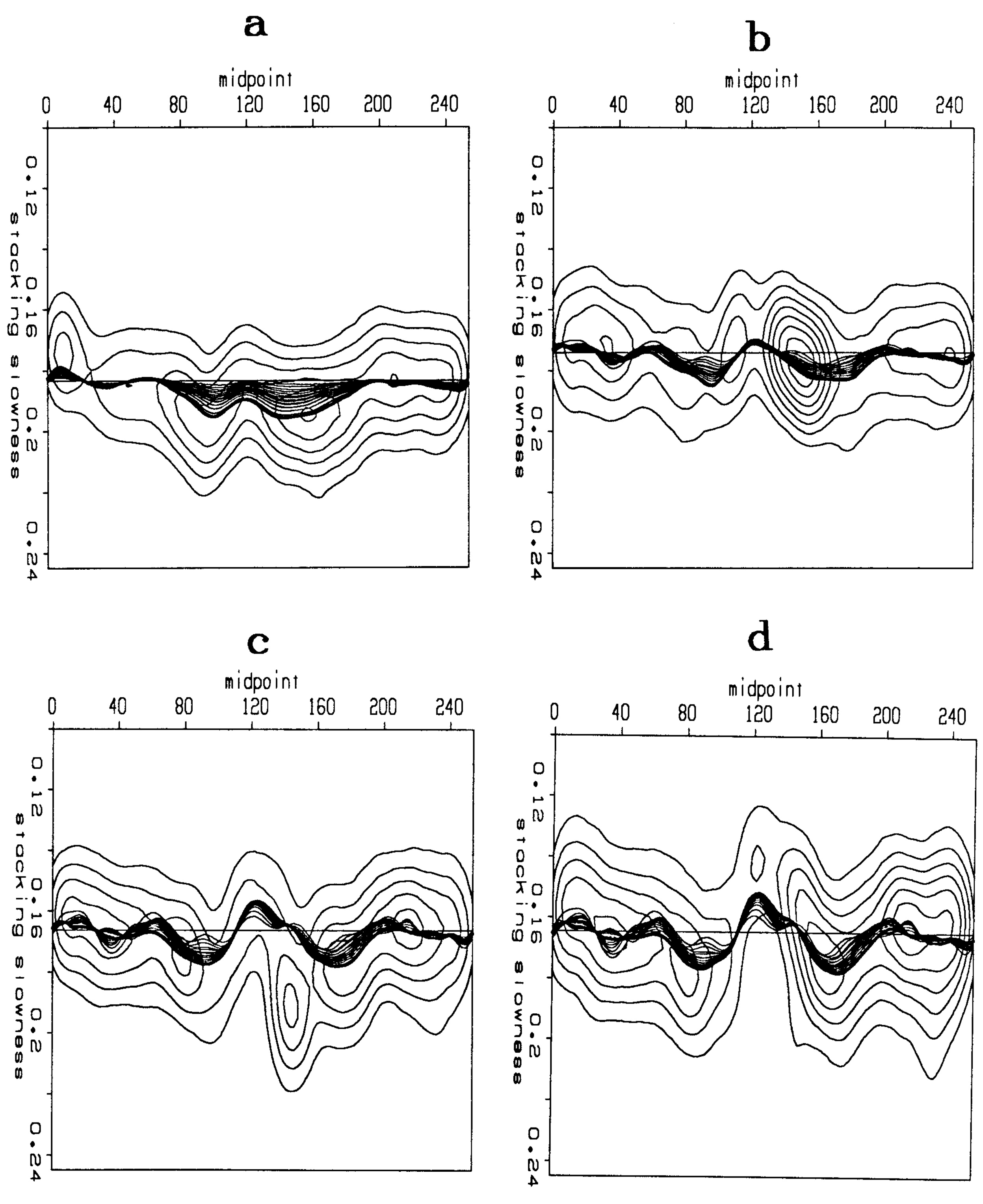

FIG. 7. Contour plots of semblance as a function of stacking slowness and midpoint for the reflectors at times: a) .5 seconds, b) .94 seconds, c) 1.16 seconds, and d) 1.34 seconds. The solid lines through each of the planes are the stacking slownesses corresponding to the successive models in the iterative process.

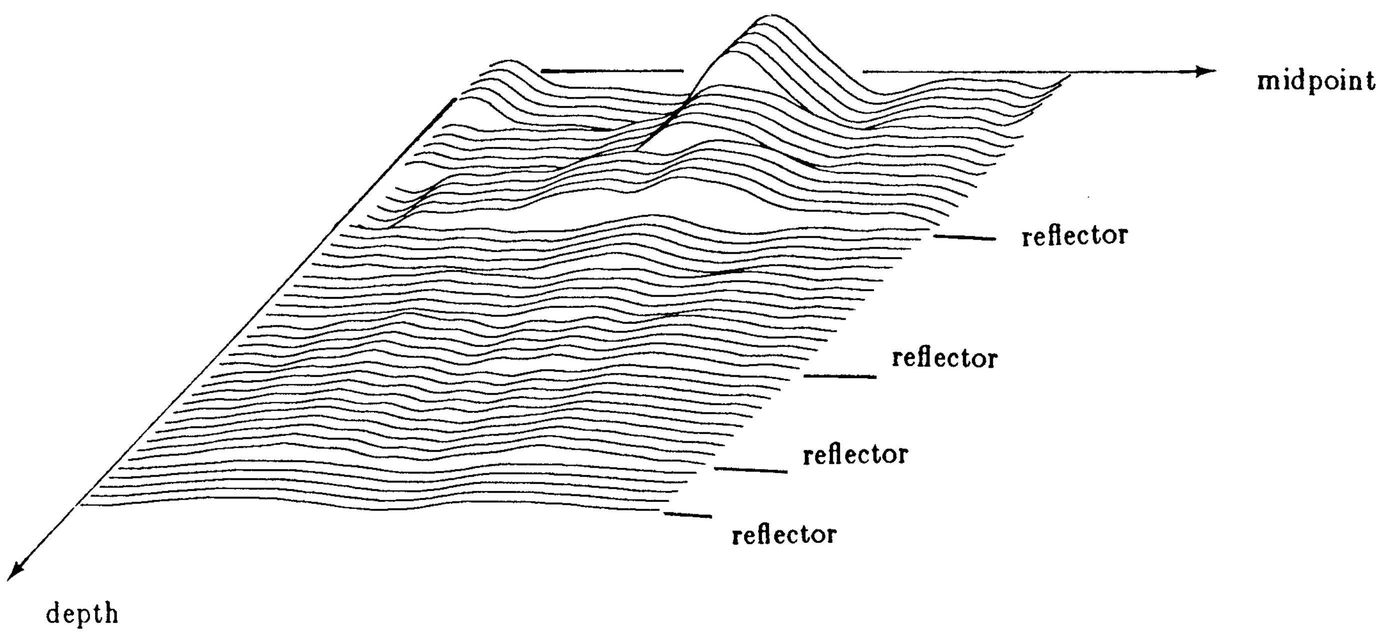

FIG. 8. Slowness model derived through the iterative process. This figure shows the change from the starting model to the final model.

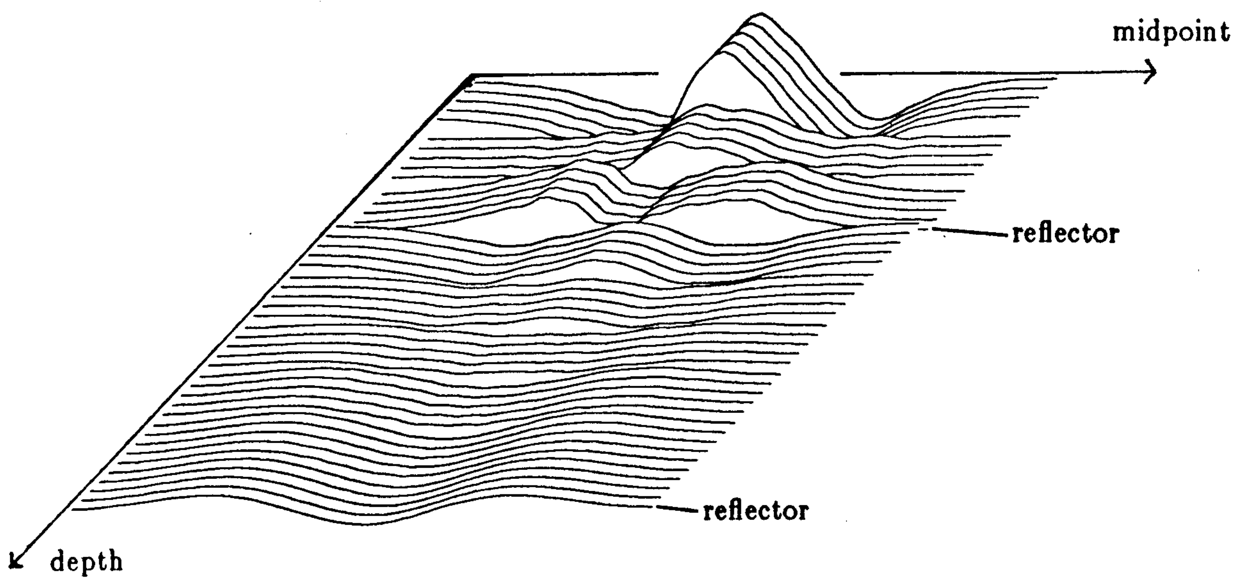

FIG. 9. Slowness model derived through the pick-based process. This figure shows the difference between the starting (i.e. background) model and the final model.

where $\mathbf{m}_0$ is the a-priori model. $\mathbf{C}_m$ is the a-priori model covariance, taken here to be diagonal. Its elements determine how strongly the algorithm should try to hold to the initial starting model. With this new objective function, the algorithms can be developed exactly as before. Adding a-priori information stabilizes an inversion procedure (Jackson, 1979), This damping should prove quite important in the accelerated ascent methods.

Traveltimes revisited

The success of the gradient algorithm shown in the last section suggests that a similar algorithm might be appropriate for the traveltime problem. Because of the formal similarity between the traveltime and stacking slowness methods, most of the equations of the velocity analysis algorithm can be directly transferred when the simple substitution of $\mathbf{t}$ for $\mathbf{w}$ is made. Thus,

$$\nabla_{\mathbf{m}} P \;=\; \mathbf{G}^T \; \nabla_{\mathbf{t}} P$$

where now:

$$\mathbf{G} \;=\; \frac{\partial \mathbf{t}}{\partial \mathbf{m}}$$

The gradient, $\nabla_{\mathbf{t}} P$, can once again be calculated with a finite-difference method.

For the traveltime problem this calculation consists of determining how the stack power changes if a traveltime is changed. That is, for the j^{th} traveltime (the traveltime to a particular reflector, for a particular midpoint and offset)

$$\begin{aligned} (\nabla_{\mathbf{t}} P)_j \;&\approx\; \left\{ \frac{\Delta P}{\Delta t} \right\}_j \\ &\equiv\; \frac{P_j(t_j + \Delta t) - P_j(t_j)}{\Delta t} \end{aligned} \tag{19}$$

P_j is the power in the part of the stack that is affected by the j^{th} traveltime.

CONCLUSIONS

The formulation of velocity analysis as an optimization problem has led to an efficient algorithm that does not require picking. This method has been tested on a field dataset that shows strong lateral variation in the velocities. The resulting velocity model quite successfully explains the observed traveltimes.

ACKNOWLEDGMENTS

I thank Bill Ostrander of Chevron USA for providing the field dataset. I would also like to thank Peter Mora, Shuki Ronen and Dan Rothman for helpful discussions.

REFERENCES

Aki, K., and Richards, P., Quantitative seismology: San Francisco, W.H Freeman, 1980.

Fawcett, J., 1983, Curved ray tomography with applications to seismology, PhD Thesis, California Institute of Technology.

Jackson, D., 1979, The use of a priori data to resolve non-uniqueness in linear inversion, Geophys. J. R. Astron. Soc.

Loinger, E., 1983, A linear model for velocity anomalies, Geophysical Prospecting, v.31, p.98-118.

Luenberger, D., Introduction to linear and non-linear programming: Reading, Addison Wesley, 1965.

Rocca, F. and Toldi, J., 1982, Lateral velocity anomalies: SEP-32, p.1-13.

Rothman, D., 1984, Non-linear inversion by stochastic relaxation with applications to residual statics: SEP-38, p.1-26.

Ronen, J., 1984, Surface-consistent residual statics by stack optimization: SEP-38, p.27-38.

Toldi, J., 1983, The linear stacking slowness method for a field dataset: SEP-37, p.17-26.

Toldi, J., 1984, Estimation of a near-surface velocity anomaly from stacking velocities: SEP-37, p.17-26.