Chapter 2

One-Dimensional Velocity Analysis without Picking

2.1 OVERVIEW OF CHAPTER

By changing the interval-velocity model, the velocity-analysis algorithm proposed in this thesis searches for the stacking velocities that correspond to the maximum values of semblance. Because the search is model driven, a key component of the algorithm is the theory connecting interval velocities to stacking velocities, $v_s(v_{in})$. The simplest such theory has v_s equal to the rms average of the v_{in}. A more generally applicable theory would relate v_s and v_{in} through the traveltimes, that is as $v_s[t(v_{in})]$.

The algorithm can also be examined in a general way that emphasizes the search itself. Viewed this way, the theory connecting interval velocities and stacking velocities is just a detail; either of the two theories mentioned above could be used. This general view is the one taken in this chapter. The search algorithm is emphasized by the use of the simplest theory: stacking velocity equals the rms average of interval velocity. Furthermore, this chapter studies the algorithm for a simple dataset: a single CMP gather. The next chapter then extends the theory connecting interval and stacking velocities to allow for laterally varying velocities. Although the theory becomes more complicated than a simple rms calculation, the structure of the search algorithm is identical to the one used in this chapter.

Fundamental to any algorithm that searches for the best model, is a means of evaluating a model. The evaluation is embodied in what is known in optimization theory as an objective function—that is, the function to be maximized. The objective function used here evaluates a model by looking at the semblance values corresponding to $v_s(v_{in})$. Next, a new model must be selected for evaluation. How should this selection be made? This question is answered as follows: look along the direction of the gradient of the objective function, taken with respect to the model.

The basic algorithm is then enhanced by a more sophisticated, conjugate gradient search. An even more important enhancement explicitly incorporates a priori information about the interval velocity model into the algorithm. Finally, the algorithm is illustrated with several field-data examples.

2.2 BASIC SETUP; DEFINITIONS AND NOTATION

For this chapter, the model takes an especially simple form: a series of homogeneous layers. Although one could choose the model parameter for each layer to be interval velocity, I use instead interval slowness, where *slowness* $= 1$ / *velocity*. Likewise, in place of stacking velocity, is used stacking slowness. This choice of parameters will prove quite useful in the next chapter.

The model can be specified by a vector $\mathbf{m}$, in which each element is the interval slowness for one layer. Corresponding to this model $\mathbf{m}$, will be a stacking slowness vector $\mathbf{w}(\mathbf{m})$, in which each element w_i, is the stacking slowness at one particular zero-offset time, τ_i. Because the layers are chosen to be of equal traveltime thickness, the zero-offset times, τ_i, will be evenly spaced. Note that throughout this thesis bold characters represent vectors.

Figure 2.1a shows an interval slowness model $\mathbf{m}$ with 100 layers, each of thickness .04 seconds. Figure 2.1b shows the corresponding stacking slownesses, $\mathbf{w}(\mathbf{m})$. Because this chapter assumes that stacking velocity equals rms velocity, each value of $\mathbf{w}(\mathbf{m})$ is formed by means of the standard rms calculation, recast in terms of slowness. That is, the stacking slowness at τ_i, $w_i(\mathbf{m})$, is calculated from the interval slownesses in all layers shallower than τ_i as

$$w_i(\mathbf{m}) = \frac{1}{[v_{rms}(\mathbf{m})]_i} = \left[\frac{\tau_i}{\sum_{j=1}^{i} \frac{(\tau_j - \tau_{j-1})}{m_j{}^2}} \right]^{1/2} . \qquad (2.1)$$

2.3 OBJECTIVE FUNCTIONS

The fundamental question in velocity analysis is, how do we know if one model is better than another? To answer that question, this thesis proposes the following two-step procedure. First, calculate the stacking slownesses predicted by the interval-slowness model to be tested—that is, calculate $\mathbf{w}(\mathbf{m})$. This calculation gives a set of stacking slownesses $\mathbf{w}$, with one element w_i for each zero-offset time, τ_i. These stacking slownesses and zero-offset times can then be used to define hyperbolic summation trajectories over offset, with

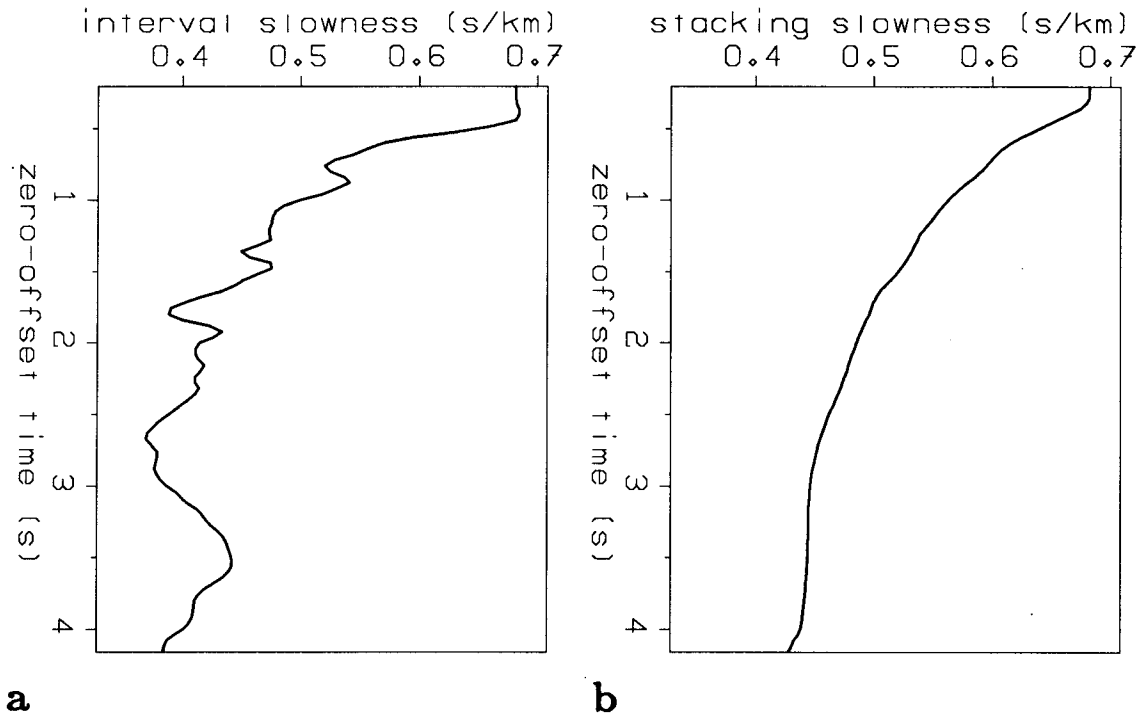

FIG. 2.1. a) Interval slowness model **m**. b) Corresponding to **m** are stacking slownesses **w**(**m**).

$$t^2 = \tau_i^{\,2} + w_i^{\,2}(\mathbf{m})\, x^2 . \tag{2.2}$$

Equation (2.2) is the NMO equation, rewritten in terms of stacking slowness. Thus, the second step consists of performing NMO according to **w**(**m**), and then stacking over offset. The larger the power in the stack due to **w**(**m**), the better the model **m**.

This measure of the quality of **m** can be written as

$$Q_1(\mathbf{m}) = \sum_i \left(\sum_x D\left[x\,,\, t = (\tau_i^{\,2} + w_i^{\,2}(\mathbf{m})\, x^2)^{\frac{1}{2}} \right] \right)^2 . \tag{2.3}$$

$D(x\,,\,t)$ are the data in a CMP gather. The outer sum of equation (2.3) derives the total power in the stack by adding up the contributions from each zero-offset time τ_i.

The criterion $Q_1(\mathbf{m})$ of equation (2.3) favors the large-amplitude parts of the data. Thus, instead of Q_1, a normalized measure is more appropriate; here I replace the stack power with a normalized sum, the semblance. The measure of quality becomes

$$Q(\mathbf{m}) = \sum_i S(w_i(\mathbf{m}), \tau_i) , \tag{2.4}$$

where $S(w_i(\mathbf{m}), \tau_i)$ is the semblance at zero-offset time τ_i and stacking slowness $w_i(\mathbf{m})$:

$$S(w_i(\mathbf{m}), \tau_i) = \frac{\left(\sum_x D\left[x, t = (\tau_i^2 + w_i^2(\mathbf{m})\, x^2)^{\frac{1}{2}} \right] \right)^2}{\sum_x \left(D\left[x, t = (\tau_i^2 + w_i^2(\mathbf{m})\, x^2)^{\frac{1}{2}} \right] \right)^2} . \qquad (2.5)$$

In practice, the numerator and denominator of equation (2.5) are summed over a time window large enough to include the seismic wavelet. The length of the time window can be different for the numerator and the denominator.

Thus far, the evaluation of the model $\mathbf{m}$ has taken the following form:

for each zero-offset time τ_i
{
calculate stacking slowness, $w_i(\mathbf{m})$ (equation (2.1))
calculate semblance, $S(w_i(\mathbf{m}), \tau_i)$ (equation (2.5))
}
add up the $S(w_i(\mathbf{m}), \tau_i)$ from all τ_i (equation (2.4))

The problem with this form of the algorithm is that every time a new model is evaluated, the semblance must be recomputed. This means that the sums over offset of equation (2.5) must be recalculated for each model. This problem can be overcome if the algorithm is reorganized to start with the same first step as the conventional velocity-analysis method: calculate semblance for a range of w and τ values. Provided that this range is sufficiently large, any $S(w_i(\mathbf{m}), \tau_i)$ required by equation (2.4) can then be derived through interpolation between these precalculated $S(w, \tau)$.

Thus, by using the same first step as the conventional method—the precalculation of a plane of semblance values $S(w, \tau)$— the algorithm of this thesis takes an attractive form:

calculate $S(w, \tau)$ for a suitable range of w and τ
for each zero-offset time τ_i
{
calculate stacking slownesses, $w_i(\mathbf{m})$ (equation (2.1))
calculate $S(w_i(\mathbf{m}), \tau_i)$ by interpolation between the $S(w, \tau)$
}
add up the $S(w_i(\mathbf{m}), \tau_i)$ from all τ_i (equation (2.4))

Because most of the cost is in the initial calculation of $S(w, \tau)$, this form allows for the

repeated evaluation of models at little extra cost.

This form of the algorithm also provides a clear geometric insight into the evaluation process. Figure 2.2 shows two different interval-slowness models and their corresponding stacking slownesses. The first model, $\hat{\mathbf{m}}$ of Figure 2.2a, has the same interval slowness at all depths. The second model, $\mathbf{m}$ of Figure 2.2b, has interval slownesses that vary with depth. Which model better describes the data in a particular CMP gather? The answer is provided by a sum through the semblance panel $S(w, \tau)$ for the CMP gather, along the curves $\mathbf{w}(\hat{\mathbf{m}})$ and $\mathbf{w}(\mathbf{m})$ of Figure 2.2c and 2.2d.

Figure 2.3 illustrates this evaluation process. It shows the stacking slowness curves of Figure 2.2c and 2.2d, overlaid on a contour plot of the semblance values $S(w, \tau)$ for the CMP gather. For this example,

$$S(w_i(\mathbf{m}), \tau_i) > S(w_i(\hat{\mathbf{m}}), \tau_i) \quad \text{for } \textit{all } i \tag{2.6}$$

Thus,

$$Q(\mathbf{m}) = \sum_i S(w_i(\mathbf{m}), \tau_i) > \sum_i S(w_i(\hat{\mathbf{m}}), \tau_i) = Q(\hat{\mathbf{m}})$$

which says that model $\mathbf{m}$ is better than model $\hat{\mathbf{m}}$. Thus, according to the objective function $Q(\mathbf{m})$ of equation (2.4), the closer that the curve $\mathbf{w}(\mathbf{m})$ is to the peaks of the semblance plane, the better the model $\mathbf{m}$.

Because the algorithm does try to find the peaks of the semblance plane, one might wonder how this differs from an algorithm that tries to fit picked peaks. The difference arises from the way that it interacts with a priori information about the model. As will be shown in a later part of this chapter, an extra term can be added to the basic objective function of equation (2.4), which makes the algorithm also try and satisfy a priori information about the model. For example, the extra term could make the algorithm try for a smooth model. Such a term can also be incorporated into an algorithm that tries to fit picked peaks.

As long as the semblance peaks are pulling the algorithm in a direction that is not too inconsistent with the a priori information, the difference between the algorithm of this thesis and one based on fitting picked peaks will be small. However, when spurious peaks are present, due to coherent noise such as multiple reflections, the algorithm of this thesis will be better able to ignore those peaks. The reason is that in the original semblance plane the spurious peaks only affect the solution over a limited range of slownesses; the corresponding picked peaks would influence the solution over an infinite range of slowness.

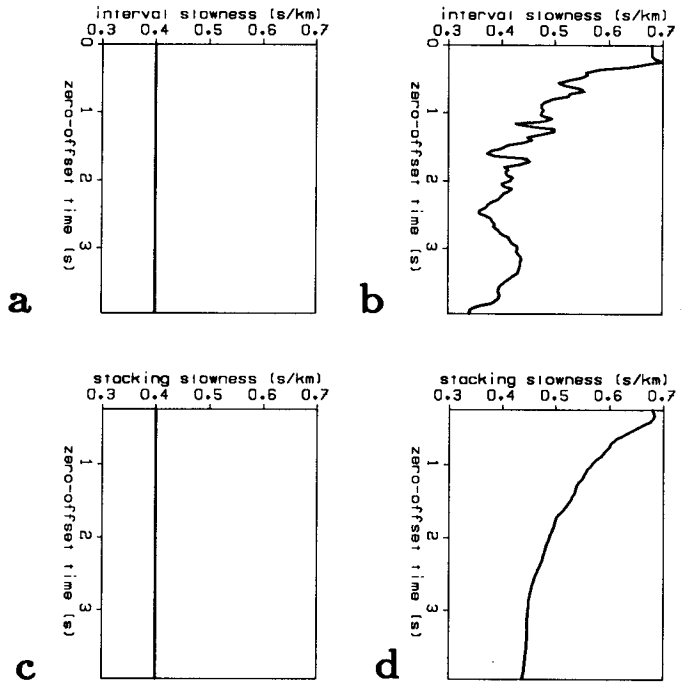

FIG. 2.2. Two interval slowness models and their corresponding stacking slownesses. a) A model, $\hat{\mathbf{m}}$, which has the same interval slowness at all depths. b) Another model, $\mathbf{m}$, which has interval slownesses that vary with depth. c) The stacking slownesses predicted by the model $\hat{\mathbf{m}}$. d) The stacking slownesses predicted by the model $\mathbf{m}$.

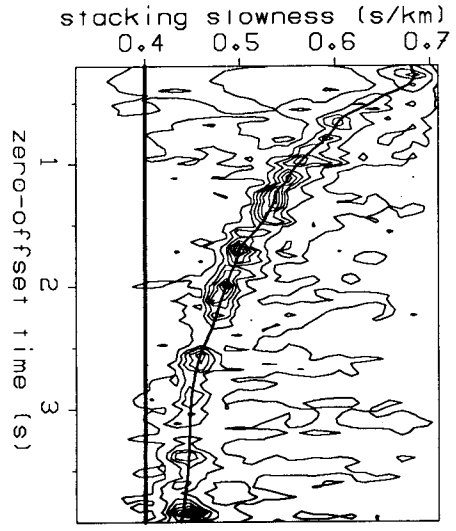

FIG. 2.3. The stacking slownesses curves of figure 2.2 overlaid on a contour plot of the semblance values for this CMP gather. These curves act as summation paths through the semblance values. The straight line is $\mathbf{w}(\hat{\mathbf{m}})$. The curved line is $\mathbf{w}(\mathbf{m})$. The contour interval is .05 (semblance ranges from 0 to 1).

The example shown in Figure 2.4 illustrates this point. The series of peaks on the right side of Figure 2.4a are regularly spaced in time and are all at water slowness (.667 s/km): they are water-bottom multiples. Superposed on the contour plot of semblance values is a stacking slowness curve corresponding to an interval slowness model. Because the stacking slowness curve is out of the range of the multiples, they will contribute nothing to the sum of equation (2.4).

On the other hand, consider how an algorithm based on picked peaks would handle the same data. Suppose that for each zero-offset time, τ_i, of Figure 2.4a, the stacking slowness at the peak of semblance, W_i, was picked. Then, a model $\mathbf{m}$ could be evaluated by the determination of how well its stacking slownesses $\mathbf{w(m)}$ fit the picked stacking slownesses, $\mathbf{W}$. A weighted least-squares criterion would then be,

$$Q_{LS}(\mathbf{m}) = \sum_i \left(A_i - \frac{1}{2} \frac{\left[W_i - w_i(\mathbf{m}) \right]^2}{\sigma_i^2} \right) \tag{2.7}$$

The constants σ_i and A_i are the width and height of the peak picked at time τ_i. The criterion is written in this fashion so that it is maximized when $\mathbf{w(m)}$ is close to $\mathbf{W}$.

At any zero-offset time, equation (2.7) says that the quality of the model $\mathbf{m}$ decreases with the square of the distance of the stacking slowness value $w_i(\mathbf{m})$ from the picked peak location W_i. For the first 1.5 seconds of the data in Figure 2.4a, the peak locations would surely be those of the multiples. Thus, equation (2.7) pulls the solution back towards the multiples, no matter how far away the solution gets.

Like equation (2.4), equation (2.7) evaluates the model $\mathbf{m}$ by summing over time, along the curve $\mathbf{w(m)}$. The difference is that instead of summing through a plane composed of the original semblance values $S(w, \tau)$, equation (2.7) sums through a plane composed of parabolas centered on the picked peaks. A comparison between Figures 2.4a and 2.4b illustrates this point, showing how the fit to picked peaks greatly increases the range of stacking slownesses affected by the multiples. Indeed for the least-squares criterion of equation (2.7), the further the solution is from the peak, the harder the pull. Although the strength of this pull could be lessened if the least-squares criterion were replaced with an absolute-value criterion, the infinite range of influence of the multiples would still hold.

The algorithm of this thesis, however, by using the original semblance values, retains the original range of influence of any peak. Once the solution has been pulled away from the multiples, it no longer feels them; indeed it may find some other, smaller peaks at the new location. The need to also satisfy the a priori information is what pulls

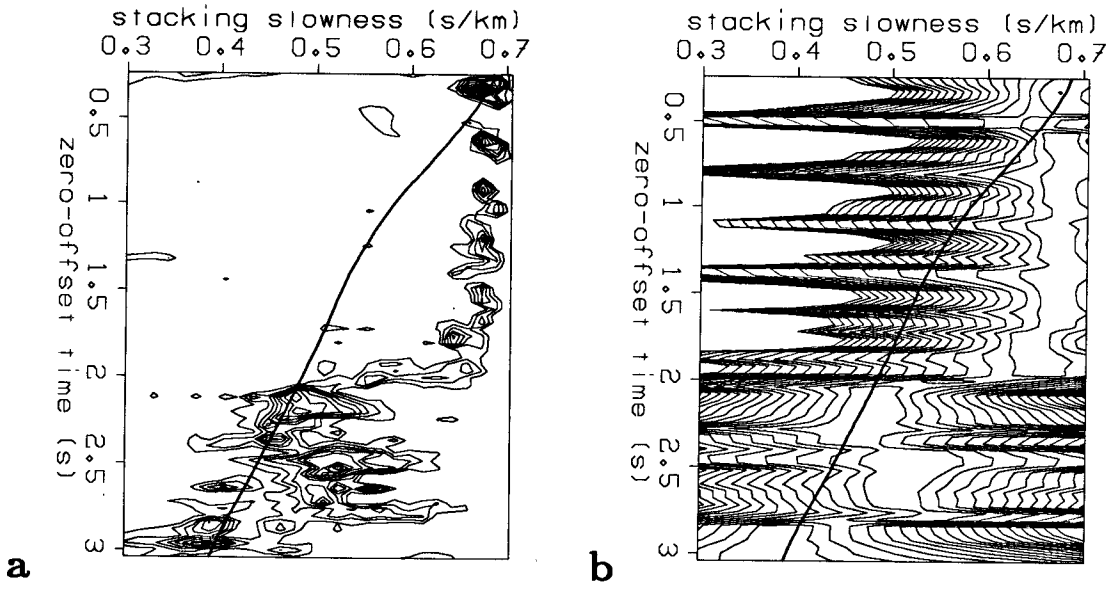

FIG. 2.4. a) The algorithm of this thesis evaluates a model **m**, by summing through the semblance plane along the stacking slowness curve **w**(**m**). Note the strong water-bottom multiples at the right side of the figure. b) The evaluation of a model **m** by an algorithm based on a least-squares fit to picked peaks. The same summation curve **w**(**m**) is used, but instead of summing through the original semblance plane of 2.4a, it sums through a plane composed of parabolas centered on the picked peaks. Using a fit to picked peaks greatly increases the range of stacking slownesses affected by the multiples.

the solution away from spurious peaks. Finally, when the model implied by the peaks is not too inconsistent with the a priori information, the algorithm will perform as well as an algorithm based on picked peaks.

2.4 CHOOSING THE NEXT MODEL

The first part of this chapter has attempted to answer the question: how can we know if one interval slowness model is better than another? The proposed answer is based on a look at the power in a CMP stack (specifically the function $Q(\mathbf{m})$ of equation (2.4)). But knowing how to compare two velocity models is only the beginning of the solution. How do we choose that second model?

This second question is directly addressed by optimization theory; indeed it classifies methods according to the type of information they use in choosing a second model (Gill, Murray and Wright, 1981). In general, the level of information used about the derivatives of the function $Q(\mathbf{m})$, the so-called objective function, is the key consideration in this classification. The simplest methods use no derivative information at

all; they just repeatedly evaluate the objective function.

The very simplest such method would try every possible model. For the velocity analysis problem such a method is clearly not practical. (If every one of the 100 parameters of a velocity model were allowed to try any of 50 values, 50^{100} models would need to be tested.) There are cleverer ways of searching through models than the simple exhaustive search. Rothman (1985) and Ronen and Claerbout (1985) have successfully applied different non-derivative methods to the residual statics problem.

The next level of methods use information about the first derivative of the objective function. In particular, the gradient of the objective function, taken with respect to the model parameters, indicates the direction of maximum increase of the objective function. This gradient is a natural choice of direction in which the next model point can be sought. By always changing the model in a direction that increases the value of the objective function, first-derivative methods can be more easily trapped in local extrema than can non-derivative methods. First-derivative methods thus depend more strongly on the starting position than do the non-derivative methods. They do, however, have a distinct advantage of speed.

The algorithm used in this thesis is an enhanced first-derivative method, the conjugate-gradient method (Luenberger, 1973). The essential elements of the algorithm can be demonstrated by a simple gradient method. Let $Q(\mathbf{m})$ be the objective function, evaluated for model $\mathbf{m}$. Furthermore, let $\nabla_{\mathbf{m}} Q$ be the gradient of the objective function with respect to the model parameters. A simple gradient-ascent (steepest-ascent) algorithm proceeds as follows:

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

Calculating the gradient, $\nabla_{\mathbf{m}} Q$

Velocity analysis calculates $Q(\mathbf{m})$ through the intermediary of stacking slowness. That is, $Q(\mathbf{m})$ is more properly $Q[\mathbf{w}(\mathbf{m})]$. (Recall that when $\mathbf{m}$ is evaluated, the curve $\mathbf{w}(\mathbf{m})$ is first calculated, then this curve is used for the summation through the semblance panel.) The j^{th} component of the gradient can thus be expressed, with the help of the chain rule of partial differentiation, as

$$(\nabla_{\hat{\mathbf{m}}} Q)_j = \left. \frac{\partial Q}{\partial m_j} \right|_{\mathbf{m} = \hat{\mathbf{m}}} \tag{2.8}$$

$$= \sum_{i=0}^{i=n_\tau} \left. \frac{\partial Q}{\partial w_i} \right|_{\mathbf{w} = \mathbf{w}(\hat{\mathbf{m}})} \left. \frac{\partial w_i}{\partial m_j} \right|_{\mathbf{m} = \hat{\mathbf{m}}} .$$

In equation (2.8), the index j refers to the interval slowness in the j^{th} layer, the index i to the stacking slowness at time τ_i. The sum in equation (2.8) includes all stacking slownesses that could be affected by a change in the j^{th} model parameter. All derivatives are evaluated at the current model point $\hat{\mathbf{m}}$.

The derivative $\partial w_i / \partial m_j$ in equation (2.8) is calculated from equation (2.1), the equation defining $\mathbf{w}(\mathbf{m})$:

$$\left. \frac{\partial w_i}{\partial m_j} \right|_{\mathbf{m} = \hat{\mathbf{m}}} = \frac{\tau_j - \tau_{j-1}}{\tau_i} \left[\frac{w_i(\hat{\mathbf{m}})}{m_j} \right]^3 \quad \text{for } j \leq i \ . \tag{2.9}$$

$$= 0 \qquad \text{for } j > i \ .$$

The derivative is zero for $j > i$, because the stacking slowness at time τ_i will not be affected by changes in the interval slownesses of deeper layers.

Each derivative $\partial w_i / \partial m_j$ in equation (2.9) can be thought of as a component of a derivative matrix $\hat{\mathbf{G}} = \left. \partial \mathbf{w} / \partial \mathbf{m} \right|_{\mathbf{m} = \hat{\mathbf{m}}}$. With the substitution of this definition, equation (2.8) can be rewritten as

$$(\nabla_{\hat{\mathbf{m}}} Q)_j = \sum_{i=0}^{i=n_\tau} \left. \frac{\partial Q}{\partial w_i} \right|_{\mathbf{w}(\mathbf{m}) = \mathbf{w}(\hat{\mathbf{m}})} \hat{\mathbf{G}}_{ij} \ , \tag{2.10}$$

$$= \sum_{i=0}^{i=n_\tau} \left(\nabla_{\hat{\mathbf{w}}} Q \right)_i \hat{\mathbf{G}}_{ij} \ .$$

The gradient with respect to the model parameters can now be written as

$$\nabla_{\hat{\mathbf{m}}} Q = \hat{\mathbf{G}}^T \nabla_{\hat{\mathbf{w}}} Q \ . \tag{2.11}$$

Note that equation (2.11) uses the transpose of $\mathbf{G}$ to connect the two gradients. This use of the transpose can be understood in the following manner. The derivative matrix $\mathbf{G}$ relates a perturbation in the interval-slowness model to the corresponding perturbation in the stacking slownesses. That is, in the linear approximation,

$$\mathbf{w}(\mathbf{m}) \approx \mathbf{w}(\hat{\mathbf{m}}) + \hat{\mathbf{G}} (\mathbf{m} - \hat{\mathbf{m}}) \ . \tag{2.12}$$

When viewed this way, each row of the matrix $\mathbf{G}$ is seen to compute the appropriate

stacking-slowness perturbation through a weighted sum of the interval slowness perturbations $(\mathbf{m} - \hat{\mathbf{m}})$. Conversely, the transpose of $\mathbf{G}$ takes a perturbation in the i^{th} stacking slowness, $\left(\nabla_{\mathbf{w}} Q\right)_i$, and computes the corresponding model perturbation by "back-projecting" $\left(\nabla_{\mathbf{w}} Q\right)_i$ onto the parts of the model that might have caused it.

The gradient $\nabla_{\mathbf{w}} Q$ contains the interaction with the data. If a weighted least squares objective function (i.e. Q_{LS} of equation (2.7)) were used, this gradient would just be the residual error. That is,

$$\left(\nabla_{\mathbf{w}} Q_{LS}\right)_i = -\frac{1}{\sigma_i^2}\left(W_i - w_i(\hat{\mathbf{m}})\right). \tag{2.13}$$

Now consider the objective function of this thesis, equation (2.4),

$$Q\left[(\mathbf{m})\right] = \sum_i S\Big[w_i(\mathbf{m}),\, \tau_i\Big].$$

There is not an analytic expression for the derivative as there was for the least squares problem; the derivative can, however, be determined numerically. In particular consider one component of the gradient, $\partial Q / \partial w_i$. This derivative expresses the change in the value of the objective function due to a change in the i^{th} component of the stacking slowness.

Figure 2.5 shows the semblance as a function of stacking slowness for one zero-offset time; the arrow points to the current value of $w_i(\mathbf{m})$. Thus, $\partial Q / \partial w_i$ can be measured directly from Figure 2.5. The simplest way is with a two-point finite-difference approximation:

$$(\nabla_{\mathbf{w}} Q)_i = \frac{\partial Q}{\partial w_i} \approx \frac{S\Big[w_i(\mathbf{m}) + \Delta w\,,\, \tau_i\Big] - S\Big[w_i(\mathbf{m}),\, \tau_i\Big]}{\Delta w} \tag{2.14}$$

Δw is a small change in stacking slowness.

If the derivative of a smooth function, fit locally to the points shown in Figure 2.5 is taken, a finite-difference approximation involving more than two points results. Note that if w_i is near the peak that would have been picked, a smooth function based on the three surrounding points would be exactly the quadratic function given in equation (2.7). Additional smoothing of the semblance plane along the slowness axis will allow the finite-difference derivatives to sense the peaks from a greater distance. This additional smoothing may be necessary in the early stages of the iterative algorithm; it can then be removed in the later stages, as the algorithm converges.

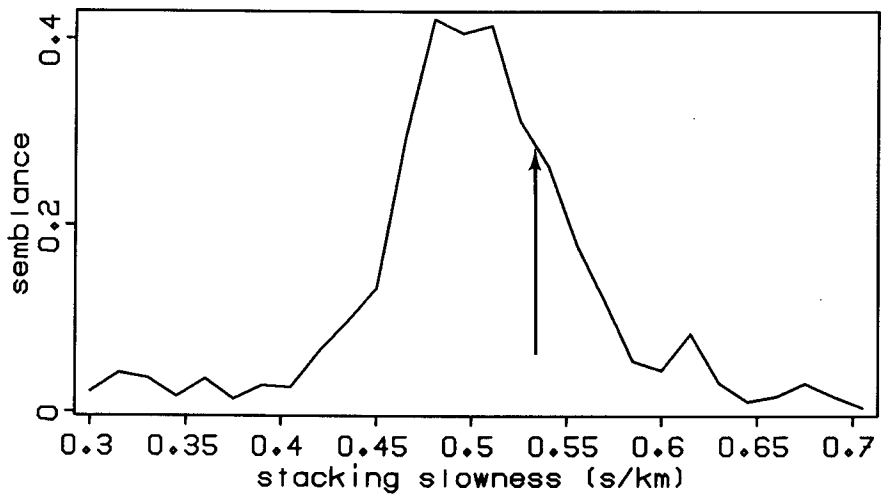

FIG. 2.5. Semblance as a function of stacking slowness for a particular zero-offset time τ_i. The value of $w_i(\mathbf{m})$ indicated by the arrow is the point at which the derivative is being evaluated.

The preceding discussion has shown how the derivative matrix $\mathbf{G}$ and the gradient with respect to stacking slowness $\nabla_{\mathbf{w}} Q$ are calculated. Their product, the gradient with respect to the model parameters $\nabla_{\mathbf{m}} Q$, defines the search direction. This search is of the following form: find α, such that $Q(\hat{\mathbf{m}} + \alpha \nabla_{\hat{\mathbf{m}}} Q)$ is a maximum. Because $Q(\mathbf{m})$ is actually $Q[\mathbf{w}(\mathbf{m})]$, the search is more properly: find α, such that $Q[\mathbf{w}(\hat{\mathbf{m}} + \alpha \nabla_{\hat{\mathbf{m}}} Q)]$ is a maximum. This search can be considerably simplified by use of the linear approximation to $\mathbf{w}(\mathbf{m})$ given by equation (2.12). That is,

$$Q[\mathbf{w}(\hat{\mathbf{m}} + \alpha \nabla_{\hat{\mathbf{m}}} Q)] \approx Q[\mathbf{w}(\hat{\mathbf{m}}) + \alpha \mathbf{G} \nabla_{\hat{\mathbf{m}}} Q)] . \tag{2.15}$$

Now the search consists of the evaluation of Q at a series of points that are linear combinations of two known vectors, $\mathbf{w}(\hat{\mathbf{m}})$ and $\mathbf{G} \nabla_{\hat{\mathbf{m}}} Q$.

A steepest-ascent algorithm

The previous discussion can be summarized by the following velocity-analysis algorithm:

Set $\mathbf{m}$ to starting model: $\mathbf{m} = \hat{\mathbf{m}}$

Set $\mathbf{w}$ to starting value: $\mathbf{w} = \mathbf{w}(\hat{\mathbf{m}})$

Calculate $\mathbf{G}$ by taking derivatives at $\hat{\mathbf{m}}$.

Begin loop on iterations

1. Form $\nabla_{\mathbf{m}} Q$ at current model point $\mathbf{m}$:

$$(\nabla_{\mathbf{w}} Q)_i = \frac{S(w_i(\mathbf{m}) + \Delta w, \tau_i) - S(w_i(\mathbf{m}), \tau_i)}{\Delta w}$$

$$\nabla_{\mathbf{m}} Q = \mathbf{G}^T \nabla_{\mathbf{w}} Q$$

2. Line search for α that maximizes $Q(\mathbf{m} + \alpha \nabla_{\mathbf{m}} Q)$

$$Q(\mathbf{m} + \alpha \nabla_{\mathbf{m}} Q) = Q[\mathbf{w}(\mathbf{m}) + \alpha \mathbf{G} \nabla_{\mathbf{m}} Q]$$

3. Update model

$$\mathbf{m} = \mathbf{m} + \alpha \nabla_{\mathbf{m}} Q$$

$$\mathbf{w} = \mathbf{w} + \alpha \mathbf{G} \nabla_{\mathbf{m}} Q$$

recalculate $\mathbf{G}$ by taking derivatives at new $\mathbf{m}$.

End loop on iterations

As written, the algorithm requires that the matrix $\mathbf{G}$ be recalculated after each search. In practice, $\mathbf{G}$ does not change too quickly; it can be recalculated every few iterations.

2.5 EXAMPLE OF BASIC ALGORITHM

The ascent algorithm proposed in the last section can be illustrated by use on field data. Figure 2.6a shows a CMP gather from the Gulf Coast in Texas; Figure 2.6b shows the corresponding panel of semblance as a function of zero-offset time and stacking slowness. The goal of the velocity analysis is to build an interval slowness model $\mathbf{m}$, such that the sum along the curve $\mathbf{w}(\mathbf{m})$ through Figure 2.6b will be maximized.

Figure 2.7a shows the starting model $\hat{\mathbf{m}}$, which has the velocity of water until the time of the first reflection (.2 sec); the model thereafter has interval velocities that increase linearly with time. The model has 100 layers, one for each zero-offset time of the semblance panel (the time increment is .04 s). Figure 2.7b shows the corresponding stacking slowness curve, $\mathbf{w}(\hat{\mathbf{m}})$, plotted over the semblance panel. This curve, $\mathbf{w}(\hat{\mathbf{m}})$, can now be used as a summation trajectory through the semblance values of Figure 2.7b; the resulting sum is the starting value of the objective function.

Now the ascent algorithm begins to search for the best model. Figure 2.8a shows the stacking-slowness curves generated at 20 successive iterations. Throughout most of

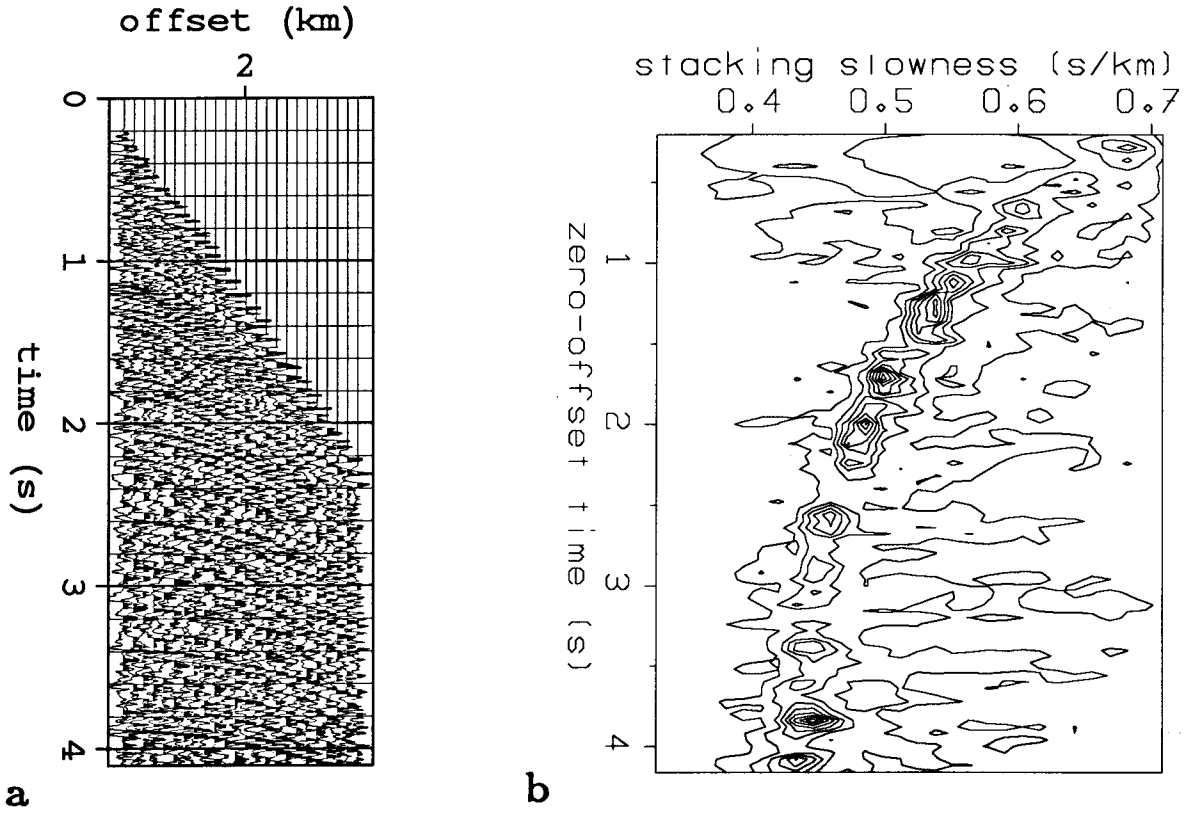

FIG. 2.6. a) A CMP gather from the Gulf Coast in Texas. b) Derived from the data in a), contour plot of semblance as a function of stacking slowness and zero-offset time.

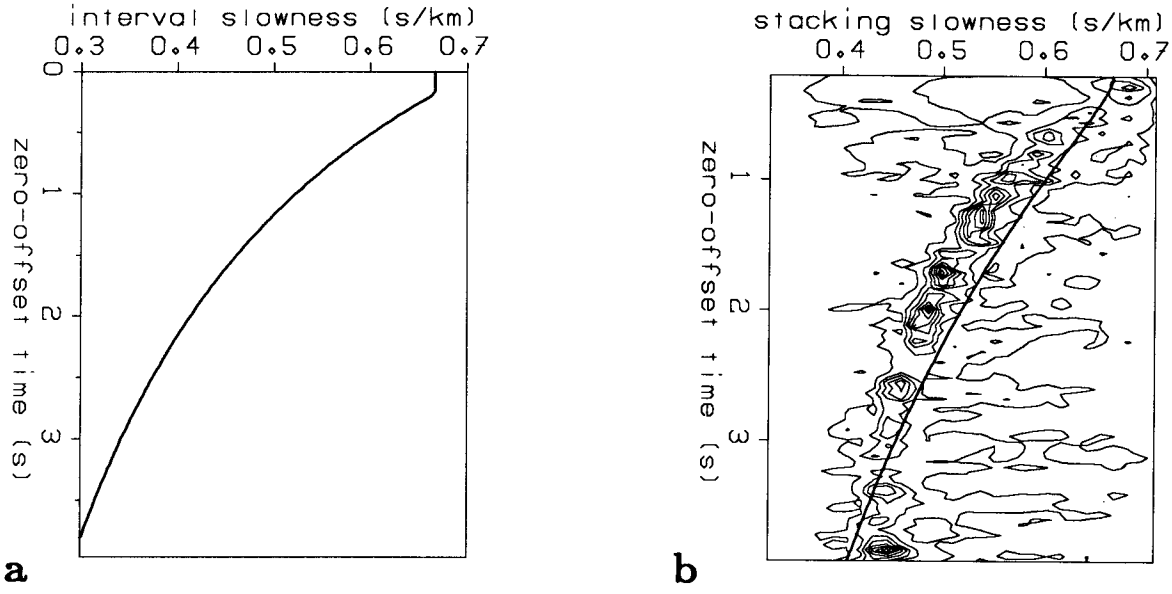

FIG. 2.7. a) Starting interval-slowness model $\hat{\mathbf{m}}$. It has interval velocities that increase linearly with time, which leads to the interval slowness curve shown. b) The corresponding stacking-slownesses curve $\mathbf{w}(\hat{\mathbf{m}})$, overlaid on a contour plot of the semblance values.

the semblance panel, the algorithm has quickly found the peaks of the semblance values. But the stacking slownesses are only an intermediate result: the iterative process is driven by the interval slownesses. The successive interval-slowness models are shown in Figure 2.8b.

The interval slownesses can be converted to the more familiar interval velocities. The interval velocities of the starting and ending models are shown in Figure 2.8c: the shallowest layer has the velocity of water, the deeper layers have typical velocities for the Gulf Coast. Note however, that the slownesses for the deepest layers have remained near the starting guess. As will be seen shortly, this result is due to incomplete convergence of the algorithm.

The rate of convergence can be seen in the value of the objective function at successive iterations; these values are shown in Figure 2.9. This figure shows that most of the change occurred in the first few iterations, during which the algorithm was finding most of the peaks (see also Figure 2.8a). A closer look at Figure 2.9 shows that the value of the objective function was still increasing when the iterations finished: the algorithm had not entirely converged after 20 iterations.

This incomplete convergence can also be seen in Figure 2.8a at approximately 3.8 seconds. There, the stacking slowness curve at the twentieth iteration was still creeping towards the peaks. This lack of convergence was directly responsible for the lack of change in the interval slownesses for late times of Figure 2.8b.

This slow convergence is a familiar problem with the steepest ascent algorithm (Luenberger, 1973). This problem is shown by the prototypical optimization problem of figure 2.10: the search for the maximum of a surface with elliptical contours. When there is a large eccentricity, the location of the maximum is better determined in one direction than in the other. The steepest-ascent path, shown in Figure 2.10 as a solid line, converges slowly in the poorly determined direction. Considerably faster convergence occurs if the steepest-ascent algorithm is replaced by a more sophisticated, conjugate-gradient algorithm; indeed when the two-dimensional problem of figure 2.10 is solved with the conjugate-gradient algorithm, convergence is achieved in just two steps (as shown by the dashed line of Figure 2.10).

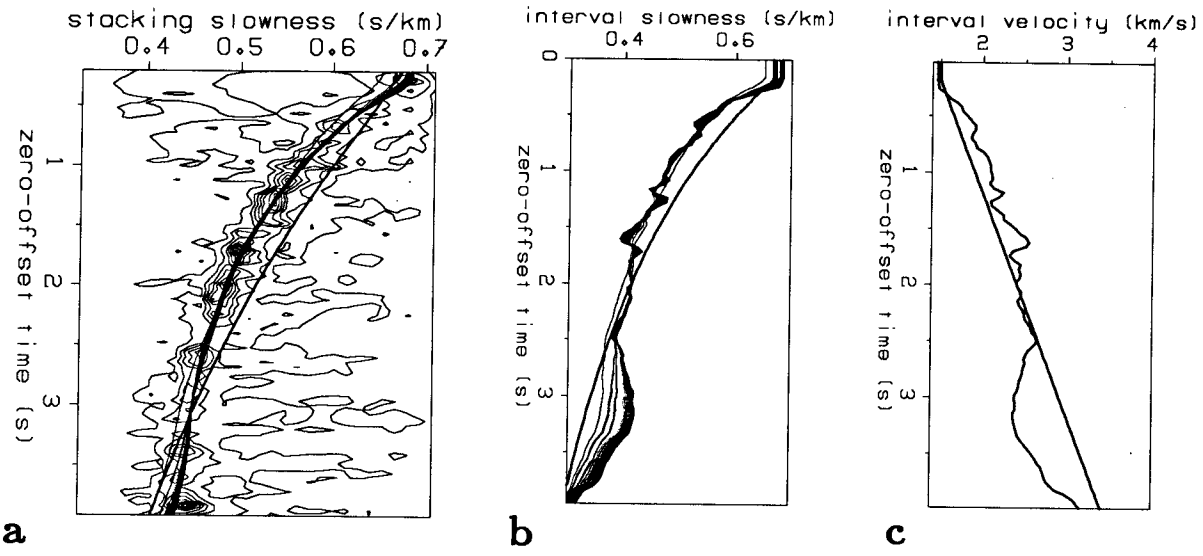

FIG. 2.8. a) The stacking slowness curves for 20 successive iterations, overlaid on a contour plot of the semblance values. The stacking slowness curve of the starting model is the curve that lies to the right of the peaks for early times (at higher slowness). b) The interval-slowness models for the 20 iterations. c) Starting and final models of b), converted to interval velocity. The shallowest layer has the velocity of water, the deeper layers have typical velocities for the Gulf Coast.

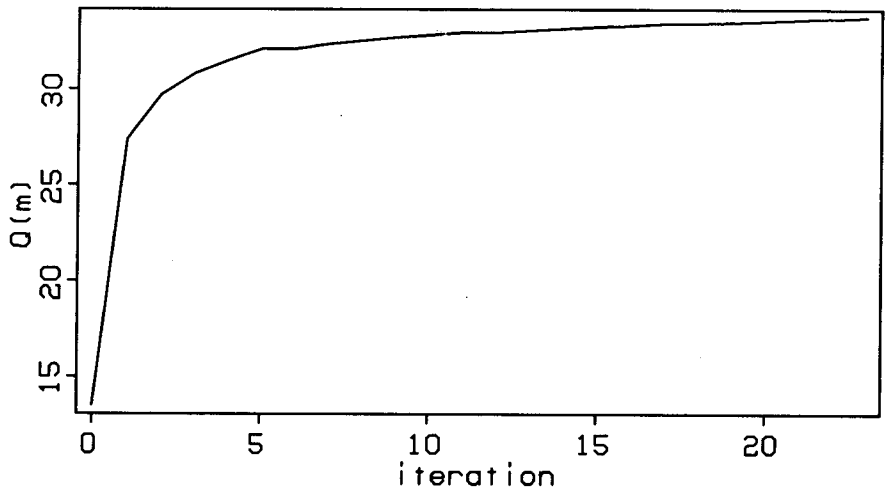

FIG. 2.9. Value of the objective function at successive iterations. Most of the change occurred in the first few iterations. The algorithm has not entirely converged after 20 iterations.

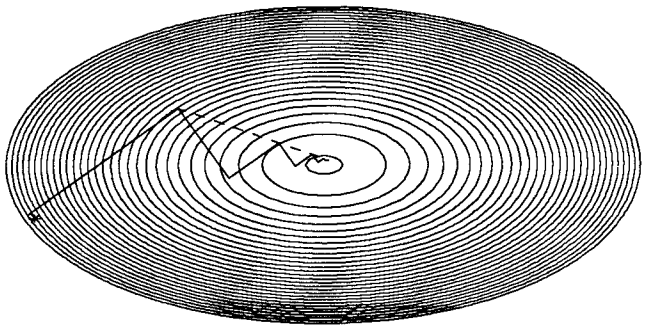

FIG. 2.10. Prototypical optimization problem: the search for the maximum of a surface with elliptical contours. The steepest-ascent path is shown in figure 2.10 as a solid line; convergence is slow. The much faster, conjugate-gradient path is shown as a dashed line.

2.6 ENHANCEMENTS TO THE BASIC ALGORITHM

A particular implementation of the conjugate-gradient algorithm, known as Partan (short for parallel tangents, see Luenberger (1973)), makes direct use of the geometric characteristics shown in Figure 2.10. Instead of always searching along the gradient direction, Partan combines the gradient with the previous ascent step, and thereby moves directly to the peak shown in Figure 2.10. The remainder of the examples of this thesis were generated using the Partan algorithm.

Except in its use of the Partan algorithm instead of the steepest-ascent algorithm, Figure 2.11 is directly analogous to Figure 2.8. Although the difference between Figures 2.8 and 2.11 is subtle, it is most visible in the deep part of the data and model. There the Partan results have moved farther away from the starting guess, than did the steepest-ascent results.

A priori information about the model

The results of Figure 2.11 indicate that a conjugate-gradient algorithm can converge faster than a steepest-ascent algorithm. But, if the conjugate gradient algorithm is allowed to keep running, it will simply converge to the largest peaks of the semblance plane. Because these peaks might or might not constitute a physically feasible set, little

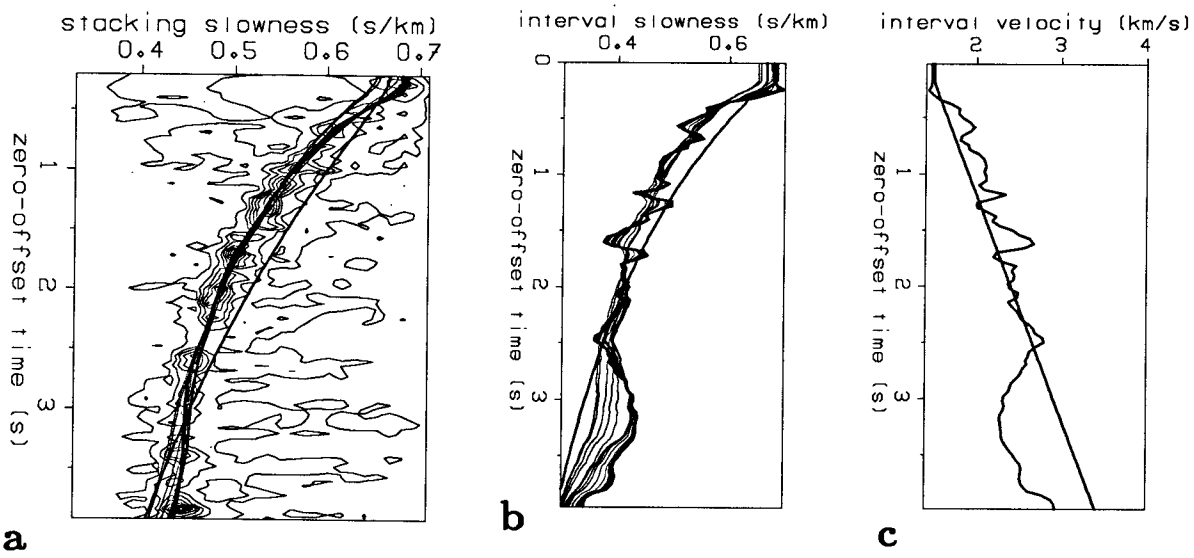

FIG. 2.11. Results generated with the accelerated, conjugate-gradient algorithm. a) The stacking-slowness curves for 20 successive iterations, overlaid on a contour plot of the semblance values. b) The interval slowness models for the 20 iterations. c) Starting and final models of b), converted to interval velocity. Unlike the simple gradient algorithm, the conjugate-gradient algorithm has converged on the peaks at 3.8 seconds.

would have been gained by the use of the model-driven, velocity-analysis algorithm of this thesis. Thus, this algorithm does more than simply find the largest peaks of the semblance plane only when additional a priori information about the model is explicitly included.

This information provides conditions that the algorithm should try to satisfy, in addition to finding the peaks of the semblance plane. Although this information applies to all components, it most strongly influences those components to which the data is least responsive. These poorly determined components are of two kinds. First are those components that are diminished by the matrix $\mathbf{G}$. The deep layers of the model affect only the stacking slownesses at the latest times, whereas the shallow layers affect stacking slownesses at all times. Thus, the deep layers have the least effect on the data. Indeed these deep layers were the cause of the slow convergence in Figure 2.8. The second type of poorly determined component is a model layer that is between times at which there are no peaks in the semblance panel.

For either of these kinds of poorly determined components, the ideal a priori information would be a set of velocity values that are expected for the dataset. This

information might be available from earlier work in the same area, or from general knowledge of the geology. It is more common though, that the a priori information must take the form of some relationship among the unknown model parameters. A good choice is a smoothness condition (Claerbout, 1976). This condition says that unless the data show otherwise, a model parameter should take the same value as that of the layers above and below.

The a priori information is easily incorporated into the objective function through an additional penalty term. That is, replace $Q(\mathbf{m})$, by

$$Q'(\mathbf{m}) = Q(\mathbf{m}) - \beta\, Q_p(\mathbf{m}), \tag{2.16}$$

where Q_p is large when the a priori condition is not satisfied. β in equation (2.16) determines the overall strength of the penalty. This means of the incorporation of a priori information is well known in inverse problems (Menke, 1984). This thesis uses a smoothness condition, based on the square of the derivative with depth of the model.

With this choice, equation (2.16) looks very much like a classical, damped-least squares objective function; the difference is in the first term, which contains the interaction with the data. That is,

$$Q'(\mathbf{m}) = \sum_i S(w_i(\mathbf{m}), \tau_i) - \beta\,(\mathbf{m}-\hat{\mathbf{m}})^T\, Tri(-1, 2, -1)\,(\mathbf{m}-\hat{\mathbf{m}}) \tag{2.17}$$

Here, $Tri(-1, 2, -1)$ is a tridiagonal matrix with the element 2 on the diagonal, and -1 on the off-diagonals. $\hat{\mathbf{m}}$ is the a priori model, and usually also acts as the starting guess for the algorithm. The new term in equation (2.17) makes the algorithm pay a penalty for models that differ from the starting model in the way they vary with depth.

The additional term also modifies the calculation of the gradient. This modification is straightforward:

$$\nabla_{\mathbf{m}} Q' = \nabla_{\mathbf{m}} Q - \beta\, Tri(-1, 2, -1)\,(\mathbf{m}-\hat{\mathbf{m}}). \tag{2.18}$$

Thus, the modified velocity-analysis algorithm looks almost like the original algorithm, with one term added on to the gradient of equation (2.11).

Smoothing

The basic algorithm is further enhanced if the early iterations work on a version of the semblance panel that is smoothed over stacking slowness. This smoothing allows the algorithm to initially sense peaks that are distant from the starting values of stacking slowness. Then, after the solution has initially found some peaks, the smoothing can be removed. Thus, the results of the early iterations will be similar to the results of a

pick-based process: large peaks of semblance influence the objective function for a large range of stacking-slowness values. The difference arises when the smoothing is removed, and the peaks' range of influence goes back to the original values.

2.7 FURTHER EXAMPLES

The enhancements of the last section lead to a velocity analysis algorithm that converges rapidly and is stable. The enhanced algorithm is first applied to the Gulf Coast dataset used in Figures 2.8 and 2.11. This example illustrates the relative unimportance of the starting guess for good data. Furthermore, the example shows how the smoothness condition on the model keeps the solution away from a peak that would otherwise lead to unreasonable interval slownesses.

Figure 2.12a shows the semblance panel along with the initial stacking-slowness curve. This example starts with an interval-slowness model that is constant with depth; this model produces a stacking-slowness curve that is also constant with depth. Because this starting curve is so far from any peaks, a long smoother over stacking slowness is appropriate for the first few iterations. Thus, Figure 2.12b shows the smoothed semblance panel, overlaid with the stacking-slowness curves for the first 5 iterations. Clearly, the algorithm has had no trouble moving to the peaks of the smoothed semblance panel.

Next, the model corresponding to the final iteration for the smoothed semblance values is used as a starting model back on the unsmoothed semblance panel. In practice the algorithm just keeps around a copy of the original smoothness panel, and switches back over to that panel after the smoothed iterations are finished. Because this starting model is so different from the constant-with-depth model that started the entire process, the derivative matrix must be recalculated. Figure 2.13 shows the stacking slownesses and interval slownesses for all 20 iterations. Note that the stacking slowness have completely avoided the small secondary peak at 2.6 seconds. This peak is due to a fault-plane reflection (Hale, 1984); it stands out from the other peaks because of the dip-induced cosine correction on the velocity. Because the smoothness condition does not allow the interval slowness to make the drastic, sudden change that would have been required for the stacking slownesses to go through this peak, the solution has been kept along the main ridge of peaks.

The next dataset, from the Grand Banks of offshore Newfoundland, is more demanding than the first one. This example was used to introduce the basic ideas of this velocity-analysis algorithm (see Figure 2.4); the shallow times are thoroughly dominated by a series of water-bottom multiples. Figure 2.14a shows the semblance panel,

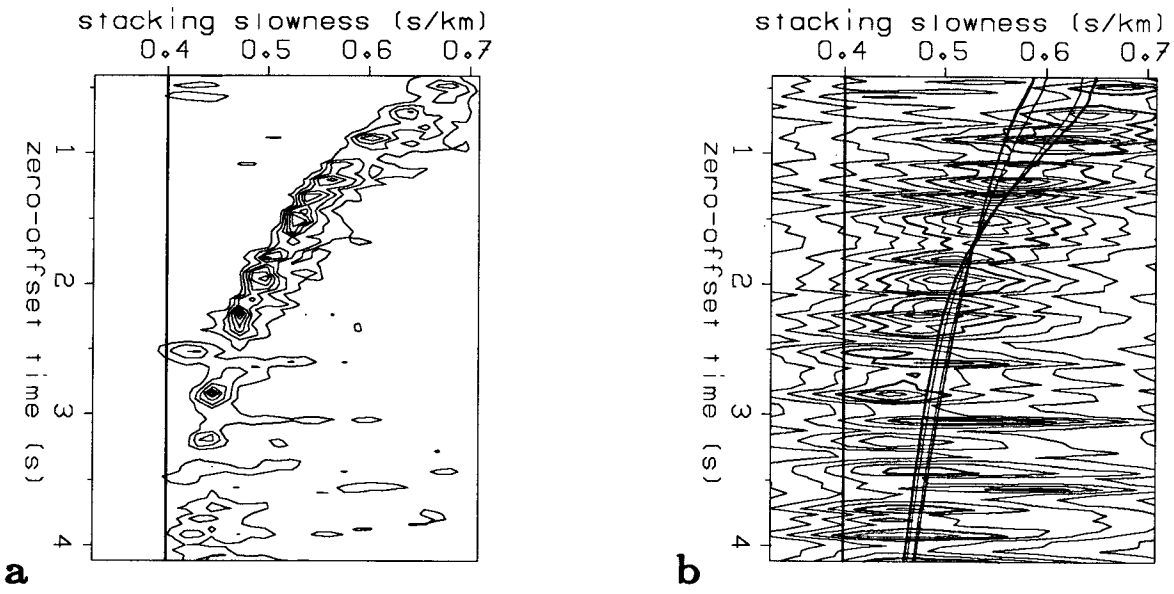

FIG. 2.12. a) Contour plot of semblance values with initial stacking slowness curve that is constant with depth. b) Contour plot of smoothed semblance values, with stacking slowness curves from the first 5 iterations. The initial stacking slowness curve is the line with constant slowness with depth.

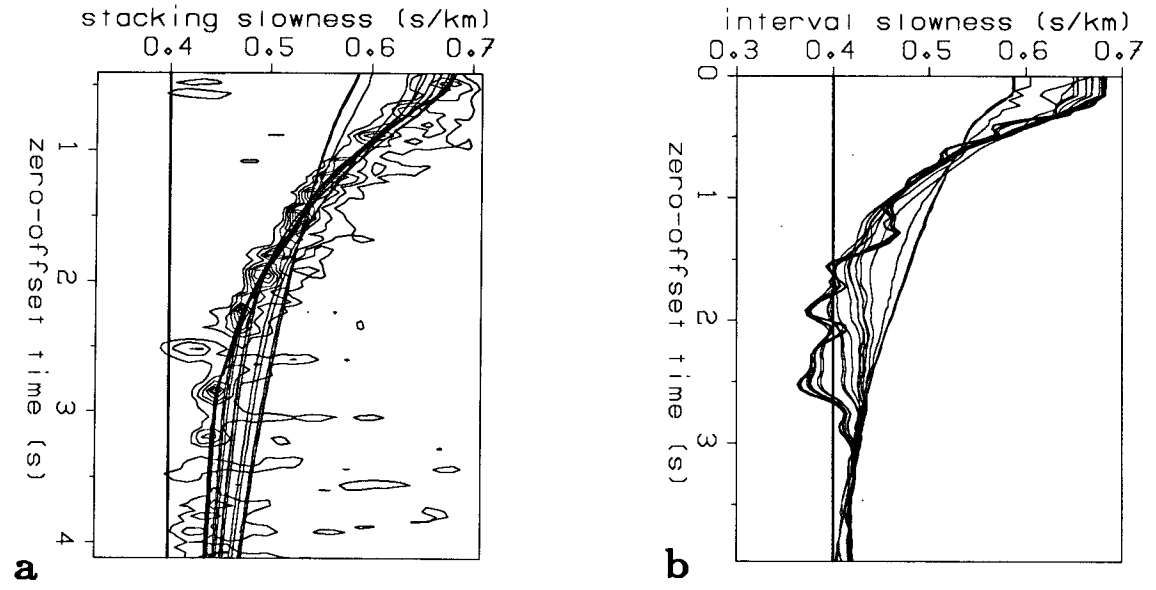

FIG. 2.13. a) The stacking slowness curves for 20 successive iterations, overlaid on a contour plot of the (unsmoothed) semblance values. b) The interval slowness models for the 20 iterations; the vertical line is the starting model.

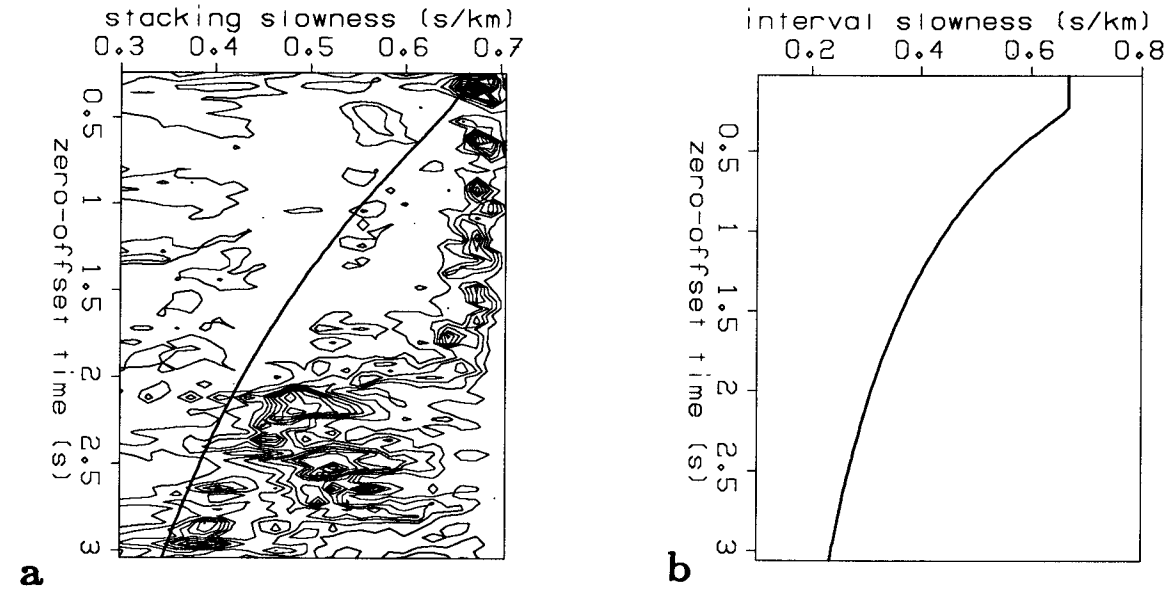

FIG. 2.14. a) Contour plot of semblance values with initial stacking-slowness curve. Note the strong multiples from 0-2 seconds. b) Initial interval-slowness model. This model has velocities that increase linearly with time, from water velocity to three times water velocity.

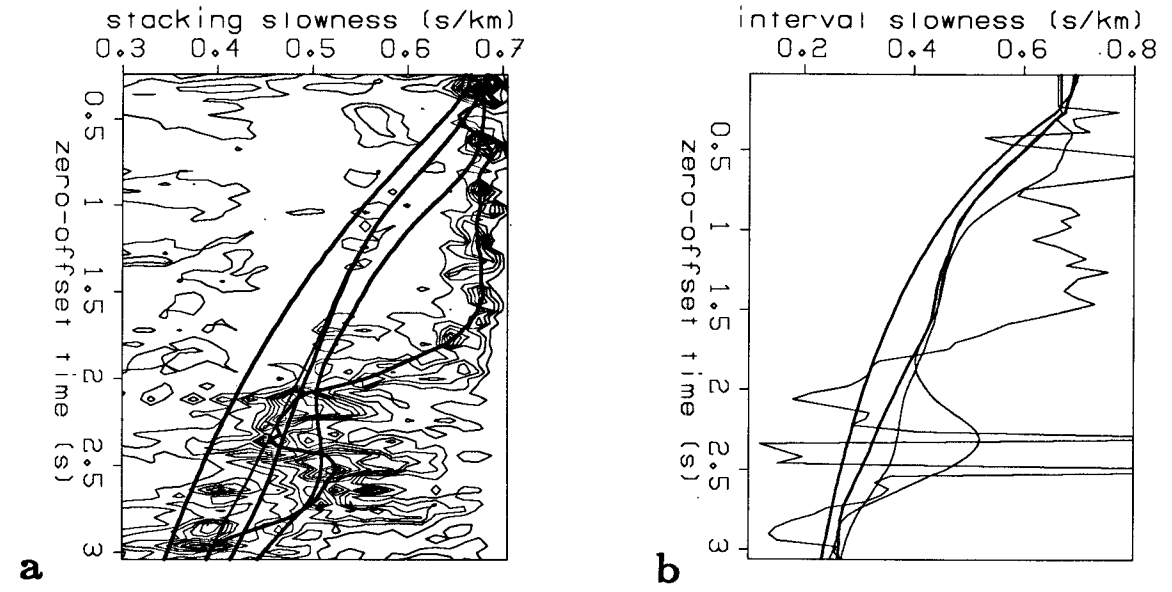

FIG. 2.15. a) Contour plot of semblance values with the initial stacking-slowness curve (the curve at the lowest values of slowness), and the final stacking slowness curves for varying strength of constraints. b) Initial interval slowness model, and final models for varying strength of constraints.

along with the initial stacking-slowness curve. Figure 2.14b shows the initial interval-slowness model. This model has the velocity of water until the first reflector (.4 seconds), then has velocities that increase linearly with time, from water velocity to three times water velocity.

Because of the strong multiples, the a priori information on the model plays a very important role for this example. This role is shown in figure 2.15a, which shows only the final stacking-slowness curves derived with different strengths of the smoothness condition. Figure 2.15b shows the analogous interval slowness models. The case without any smoothness condition has simply found a peak for every zero-offset time; this includes, of course, the multiples and the pegleg reflection around 2.6 seconds. It is also responsible for the wildly fluctuating interval slowness curve of figure 2.15b.

As the strength of the smoothness weight increases, the interval-slowness models naturally become smoother. Furthermore, the stacking-slowness curves pull away from the multiples. The semblance panel does have two strong primary reflections at fairly late times, at 2.2 and 3. seconds. At these times, the influence of the smoothness condition will thus be small; indeed it is at these times that the smooth models of figure 2.15b are farthest from the a-priori values. The choice of starting guess, and the weight given the smoothness condition played important roles in this example. Because the one-dimensional algorithm of this chapter is so fast, the best choice of these parameters can be determined by experimentation.

2.8 CONCLUSIONS

This chapter has described the basic principles of the automatic velocity-analysis method of this thesis. The fundamental premise is that stacking slownesses are always considered from the point of view of interval slownesses. This allows the algorithm to directly incorporate a priori information; use of this information can be quite effective, as was shown by the last example.

This chapter dealt with the velocity-analysis algorithm in its simplest form: for one CMP gather, under the assumption that the Dix formula is valid. Because velocity analysis is often performed in precisely this form, the algorithm developed for this simple case should be useful in its own right. Furthermore, as shall be shown in the next chapter, the basic principles of the algorithm are valid for the more general case, that is, when velocities vary laterally.