Constrained Dix Inversion

William S. Harlan

Dec. 1999

Overview

Dix inversion estimates interval velocities from picked stacking velocities, usually as a function of vertical two-way time. The stacking velocities are assumed to be explained by a root-mean-square (RMS) averaging of the interval velocities. A conventional method [3] uses an explicit solution that inverts the RMS integral. This explicit solution easily produces wildly unrealistic interval velocities from small variations in stacking velocities.

Constrained inversion fits stacking velocities with a smooth, bounded interval velocity function. This method is slower but almost always preferable to the fast explicit solution. Damped least-squares minimizes errors in picked velocities and also minimizes unnecessary complexities in interval velocities.

Constrained inversion distributes errors uniformly when fitting the squared reciprocal of stacking velocity. This distribution corresponds to uniform errors in residual normal moveouts.

Interval velocities are constructed as a sum of overlapping bell curves extending in all spatial directions. Coefficients of these curves are damped to avoid unnecessary sharpness in the estimated interval velocities. Rough changes in interval velocity are allowed only if strongly required by the input data. Finally, interval velocities are not allowed to exceed specified minimum and maximum values.

An explicit Dix solution inverts one vertical function at a time, whereas least-squares finds a global solution. Each estimated coefficient must explain stacking velocities over a range of spatial positions on the map. Redundancy greatly improves, so a single bad stacking function does not easily corrupt the solution. A few bad data points are largely ignored when contradicted by many neighboring values.

Many geophysical programmers familiar with damped least-squares have developed similar methods [5, 2, 1].

Smooth interval velocities

We assume interval velocities to be smooth in all physical directions. This assumption is most appropriate for "soft" rocks, where fluid pressure dominates seismic velocities. In "hard" rocks, velocities tend to be homogeneous in intervals, with abrupt discontinuities at changes in lithology. Soft constraints can still accurately describe the time/depth conversion of hard media.

A smoothing operator with unit area (DC) does not introduce any bias into smoothed values. On average, values are no larger or smaller than before. If interval velocities are sampled as a function of vertical traveltime, then depth is just the integral of velocity over time. If smoothing does not bias the interval velocity, then it also does not bias depth conversions. Away from the immediate vicinity of a large discontinuity, smoothing has no effect on time/depth conversions.

Our convolutional smoothing operator is a bell-shaped curve described by a third-order polynomial. The curve has unit area to preserve magnitudes. The convolution is renormalized at boundaries to preserve unit area when the convolution is truncated. A smoothing width is the "half-width" of the curve, the span over which the curve drops to half the peak value. The total width of the curve is twice the smoothing interval. Over this interval, the third-order curve is $b(r) = r^2(2r - 3) + 1$, where $r \leq 1$ is the distance from the peak divided by the smoothing distance. The curve has zero slope at the peak and endpoints. (The half-width in the Fourier domain is approximately the reciprocal of the half-width in the untransformed domain.)

Squared stacking slownesses

A stacking "velocity" is a parameter for the hyperbolic curve that best fits the moveout of reflection times over source-receiver offset. Stacking velocities are estimated from prestack seismic data by scanning ranges of acceptable values and examining weighted sums of the data over offset. Resolution depends on the width of seismic wavelets at the largest recorded offsets. Regular sampling of stacking velocities does not correspond to regular sampling of wavelets.

However, the squared reciprocal of stacking velocity, which we call squared stacking slowness (or "sloth"), does regularly sample wavelets at the farthest offset. We prefer to minimize errors in squared slowness as the best way to minimize errors in corresponding reflection times.

Interpreters tend to pick stacking velocities at locations where moveouts change the most. The locations of picks are not necessarily more significant or reliable than others. Interpreters also examine moveout adjustments at locations well away from the picks. If the interpolated behavior is acceptable, then no new picks are added. For this reason, we give interpolated stacking velocities the same significance as picked values.

We treat an interpolated regular grid of squared stacking slownesses as our hard data to be inverted. Usually input stacking velocities are interpolated linearly between picked times, with constant values off the ends. Functions are then triangulated and interpolated linearly over spatial directions. A regular grid of values needs enough resolution to represent all useful information in the original functions.

Root-mean-square equations

For this inversion, we assume a stacking "velocity" to be equivalent to the root-mean-square (RMS) average of interval velocities. This equivalence holds exactly only for infinitesimal offsets in a horizontally stratified medium.

Let a single sampled function of squared stacking slownesses be represented by the one-dimensional vector $\mathbf{s}$, and interval velocities by $\mathbf{v}$. Vector indices mark samples of vertical traveltime. Index zero corresponds to zero time. We write the RMS average of $\mathbf{v}$ in discrete form as

$$1/s_j = \frac{1}{j+1} \sum_{k=0}^{j} v_k^2. \tag{1}$$

A fast, explicit inverse does exist for the RMS equation (1):

$$v_k = \left(\frac{k}{s_k} - \frac{k-1}{s_{k-1}} \right)^{\frac{1}{2}}. \tag{2}$$

This equation (2) is typically referred to as the Dix equation, although the original reference [3] preferred more accurate variations. This explicit solution can easily fail when required to take the square root of negative numbers. Worse, statistically meaningless variations in stacking velocities can cause interval velocities to vary wildly.

For a constrained inversion, we also find it useful to write the linearization of this equation. A small perturbation Δv_k of interval velocity results in the following perturbation Δs_j of squared stacking slowness:

$$\Delta s_j = [-2\, s_j^2/(j+1)] \sum_{k=0}^{j} v_k \Delta v_k. \tag{3}$$

Unperturbed variables retain their reference values.

Finally, the adjoint linearized equation gives the perturbation of interval velocity required to explain a small perturbation of squared stacking slowness:

$$\Delta v_k = v_k \sum_{j=k}^{\infty} [-2\, s_j^2/(j+1)] \Delta s_j. \tag{4}$$

Gradient optimization methods like conjugate-gradients usually require the adjoint.

Damped least-squares

Damped least-squares attempts to balance data errors with minimal complexity in the model.

Let $\underset{\sim}{\mathbf{B}}$ be a linear smoothing operator with unit area. Define a smooth interval velocity with the convolution

$$v_k \equiv \sum_i b_{k-i}\, w_i \equiv (\underset{\sim}{\mathbf{B}} \cdot \mathbf{w})_k. \tag{5}$$

where the vector $\mathbf{w}$ contains the coefficients of smooth, shifted basis functions. Implicitly, this smoothing operator also convolves over all spatial indices, which we suppress in our equations.

The best coefficients $\mathbf{w}$ should minimize the following objective function:

$$\left\{ s_j - \left[\frac{1}{j+1} \sum_{k=0}^{j} (\underset{\sim}{\mathbf{B}} \cdot \mathbf{w})_k^2 \right]^{-1} \right\}^2 + \epsilon \sum_k w_k^2. \tag{6}$$

The small damping factor ϵ is the ratio of the variance of data errors to the variance of interval velocities. A large range of plausible values will give similar results. Damping ensures that small variations in squared stacking slowness will not cause extreme variations in interval velocity. For a purely quadratic objective function, the damping is equivalent to pre-whitening, which adds a small constant to the diagonal of the least-squares "normal" equations.

Optimization

Once we have written the objective function (6), we have unambiguously specified a solution, although only implicitly. Much has been written on the optimization of objective functions, so we will not cover the details here. See Luenberger [4] for more information on the Gauss-Newton method and conjugate-gradients.

The objective function (6) is not a perfectly quadratic function of the interval velocities $\mathbf{v}$ but behaves similarly to a quadratic. The objective function has a clear global minimum and is convex far away from that minimum. In the vicinity of the minimum, the objective function is indistinguishable from a quadratic.

If a suboptimum set of coefficients $\mathbf{w}$ produce a particular set of squared stacking slownesses $\mathbf{s}$, then the actual picked slownesses may differ by an error $\Delta\mathbf{s}$. With linearization (3), we can say that the best perturbation of coefficients $\Delta\mathbf{w}$ should minimize the following objective function:

$$\left[\Delta s_j - \frac{2 s_j^2}{j+1} \sum_{k=0}^{j} (\underset{\sim}{\mathbf{B}} \cdot \mathbf{w})_k (\underset{\sim}{\mathbf{B}} \cdot \Delta\mathbf{w})_k \right]^2 + \epsilon \sum_k (w_k + \Delta w_k)^2. \tag{7}$$

This approximate objective function (7) is perfectly quadratic. The optimum solution $\Delta\mathbf{w}$ is a linear function of the data error $\Delta\mathbf{s}$. Quadratic objective functions are easily optimized by the conjugate-gradient algorithm.

In our implementation, an outer Gauss-Newton loop iteratively replaces the objective function by the quadratic approximation (7). Each Gauss-Newton iteration begins with the best interval velocity function so far. The first iteration uses a constant interval velocity function far from the correct solution. An inner conjugate-gradient loop minimizes the objective function that has been approximated as a quadratic to find a perturbation to the reference interval velocity. A non-linear line-search finds the best factor to scale this perturbation before adding to the reference interval velocity function. (The line-search algorithm uses a combination of a parabolic Newton method for speed and a golden-section search for robustness.) Finally, the Gauss-Newton loop begins again with a new approximation of the objective function. Typically, some four to eight iterations are necessary for the Gauss-Newton and conjugate-gradient loops.

We apply hard constraints (minimum and maximum values) to interval velocities immediately after updating with a perturbation. These constraints are honored during the non-linear line-search, but not during the temporary linearization for conjugate-gradients.

As a final optimization, early iterations begin with a large smoothing operator, and thus few degrees of freedom. After full optimization with an oversimplified interval velocity, the smoothing is reduced. Finer details are allowed into the velocity model only when the background velocity is known to be near the final correct solution. Because of damping, rough details are introduced only when justified to fit a sufficiently large error in the picked data.

References

[1] Jon Claerbout. *Geophysical Estimation by Example.* http://sepwww.stanford.edu/sep/prof/toc_html/toc_html/gee/toc_html/, 1999.

[2] R. Clapp, P. Sava, and J.F. Claerbout. Interval velocity estimation with a null space. *Stanford Exploration Project Report, http://sepwww.stanford.edu/research/reports/*, 97:147–156, 1998.

[3] C. Hewett Dix. *Seismic Prospecting for Oil.* Harper and Brothers, 1952.

[4] David G. Luenberger. *Introduction to Linear and Nonlinear Programming.* Addison Wesley, 1973.

[5] J. L. Toldi. *Velocity analysis without picking.* PhD thesis, Stanford University, 1985.