

# Parabolic and linear 2-D $\tau$ - $p$ transforms using the generalized radon transform

*John E. Anderson*

## ABSTRACT

The parabolic and linear  $\tau$ - $p$  transforms can be cast as a least-squares problem in the frequency-space domain and solved using a complex form of Levinson recursion.

This paper briefly documents the algorithms coded in routines `suradon` and `suiradon`.

## INTRODUCTION

There is a rich lore of papers invoking slant stack and velocity stack methods (Thorson, 1984). The mathematical foundation for the discrete Radon transform appears in Beylkin (1987). Dan Hampson (1986) won the Canadian SEG Best Paper award based on his work using the parabolic Radon transform for multiple elimination. Hampson's method removes multiples from the near-offset traces on a common midpoint gather more effectively and with fewer edge-effect artifacts than does the traditional F-K filtering approach. Traditional F-K filtering works well on intermediate-offset and long-offset traces where Hampson's method may encounter aliasing problems. Kostov (1989) presented his work as part of the Stanford Exploration Project demonstrating that the Radon transform inversions could be computed using a complex form of Levinson recursion. Several companies had discovered this independently (Anderson, 1988; Bednar, 1988) but delayed publishing their results. In 1990, many papers were presented at the SEG meeting covering variations on this topic as corporations released their internal work in anticipation of Kostov's paper. Those authors include Darche (1990), Foster and Mosher (1990), Gulunay(1990), Johnston (1990), Kostov (1990), and Sullivan, Schneider, and Shurtleff (1990).

Traditional 2-D  $\tau$ - $p$  methods discretize equations derived using continuous function theory (Chapman, 1981). Finite aperture, finite sampling, and aliasing distort the numerical solution obtained. The discrete Radon transform approach works effectively for transforms based on temporally shift-invariant operators. Operators that fit the discrete Radon transform theory well include those for the parabolic and linear  $\tau$ - $p$  transforms since the shapes of their characteristic curves do not change with time.

NMO operators do not fit the theory well since the shapes of NMO curves do vary with time. In the first step for a discrete Radon transform, data are transformed to the frequency domain. Radon transforms are defined in terms of forward and inverse matrix operators to be applied to the data for each frequency component. Least-squares inverses are used when an exact inverse does not exist. The least-squares inverse is often different from the discretized version of the continuous inverse function. The least-squares approach takes note of the aperture and sampling of the data and does a least-squares correction to minimize their influence. Aliasing remains a problem that shows up in the form of instabilities in the least-squares inverse.

The approach taken here is to first define a transform that takes data from the "slowness" domain for linear  $\tau$ - $p$  back to the  $x$  domain. The transform of the original data to the slowness domain will be accomplished using the least-squares inverse to that transform. This allows one to model the data as a sum of slowness components. One can use this high-resolution slowness model for any purpose requiring a linear 2-D  $\tau$ - $p$  transform. In particular, based on this model, one can interpolate the original data or subtract slowness components identified as noise. The least-squares algorithm ensures that amplitudes estimated for the model are properly balanced relative to the original data. The algorithm for the parabolic transform will be defined and used in an analogous manner except that a new parameter defining parabola shape should be substituted for slowness. Typically, multiple removal is accomplished by subtracting from normal-move-out corrected common-midpoint gathers any model parabolas identified by a seismic processor as multiples.

In this paper, I derive the least-squares equations for the discrete Radon transform and show that under certain fairly general assumptions they form a Hermitian Toeplitz system solvable by a complex form of Levinson recursion. Levinson recursion requires a computational effort proportional to  $n^2$  and thus is far more computationally efficient than is standard matrix inversion which requires an effort proportional to  $n^3$ . The derivation will apply to either the parabolic transform, as described by Hampson, or to 2-D linear  $\tau$ - $p$  methods. This derivation adds no new knowledge on the topic beyond that in the literature. It does provide a concise overview and an introduction to the algorithm in the routine `suradon`.

## DEFINING THE TRANSFORM IN TERMS OF THE LEAST-SQUARES EQUATIONS

Consider a transform of the form

$$f(x, \omega) \equiv \sum_{k=0}^{n_p-1} F(p_k, \omega) e^{i\omega p_k g(x)}.$$

This can be rewritten in the matrix form  $BF = f$ , where the transformation matrix  $B$  is defined as  $B_{kl} = e^{i\omega p_k g(x_l)}$ . The vector  $f(x, \omega)$  is provided from the input data. The goal is to find a least-squares representation of  $F(p, \omega)$ .

The general least-squares solution can be written in the form of the normal equations as  $(B^\dagger B)^{-1}(B^\dagger B)F = (B^\dagger B)^{-1}B^\dagger f$  which reduces to  $F = (B^\dagger B)^{-1}B^\dagger f$ . Here the  $\dagger$  symbol is used to denote the adjoint or complex conjugate transpose.

Now, restrict the generalized slowness parameter  $p$  to be of the form

$$p = p_{\min} + k \cdot \delta p$$

Under that restriction, the matrix  $B^\dagger B$  is Hermitian Toeplitz. To see this, first note that for any matrix  $B$ ,  $B^\dagger B$  is Hermitian. That is,  $(B^\dagger B)^\dagger = B^\dagger B$ . Now, study the matrix  $C = B^\dagger B$  for our choice of  $B$ . The components of  $C$  are

$$C_{mk} = \sum_{l=0}^{nx-1} B_{lm}^\dagger B_{lk}$$

$$C_{mk} = \sum_{l=0}^{nx-1} e^{-i\omega p_m g(x_l)} e^{i\omega p_k g(x_l)} = \sum_{l=0}^{nx-1} e^{i\omega(p_k - p_m)g(x_l)}$$

Note that the quantity  $p_k - p_m = (k - m) \cdot \delta p$ . Therefore,

$$C_{mk} = \sum_{l=0}^{nx-1} e^{i\omega(k-m)\delta p g(x_l)}$$

Note also that  $C_{mk} = R_{k-m}$ , so  $C$  is Toeplitz in form, with the same elements going down each diagonal. This fact allows us to use Levinson recursion to solve the systems  $CF = B^\dagger f$  for each frequency component. This derivation did not require regular spacing in  $x$  or even a particular form for  $g(x)$ . We set  $g(x)$  equal to  $x$  for 2-D linear  $\tau$ - $p$  transforms or equal to  $x^2$  for parabolic transforms. Foster and Mosher (1992) set  $g(x)$  equal to  $\sqrt{x^2 + z_{\text{ref}}^2} - z_{\text{ref}}$  in attempt to more closely fit multiple events to hyperbolas rather than to parabolas.

## STABILITY

A good overview of the stability issues is given by Gulunay (1990). For some frequency components such as 0 Hz, or at frequencies where aliasing artifacts occur, the matrix inversion cannot successfully determine  $F$ , as the problem is clearly underdetermined. At 0 Hz, all values of the Toeplitz matrix have the same amplitude so the matrix is clearly not diagonally dominant or invertible. A similar matrix form can occur at higher frequencies when there is a spatial aliasing problem. For these frequencies, a large white-noise parameter is required to stabilize the result. The limiting case, where an infinitely large white-noise parameter is used, completely diagonalizes the matrix  $C$  and provides a solution identical to that obtained with a classical  $\tau$ - $p$  inversion based only on the conjugate operator  $B^\dagger$ . More accurate inverses are computed using small values of white noise. Ideally, the white-noise parameter would vary frequency-by-frequency and be set at the smallest level required

to ensure stability. **Suradon** applies a single white-noise factor to all frequencies. As the white-noise level is increased, the advantage of the matrix inversion is lost and the solution moves toward that of traditional conjugate operator methods. Conversely, as the white-noise level is decreased, a point is reached beyond which results become unstable.

Gulunay also points out that in the limit as the aperture becomes large and the trace spacing becomes small, the solution approaches that obtained for continuous functions and once again the matrix  $C$  becomes diagonal. The goal of the matrix inversion is to minimize the errors due to finite aperture and finite sampling.

### SUMMARY OF THE ALGORITHM IN SURADON

Loop over gathers:

Loop over  $x$  within each gather:

- get an input trace,
- extract  $x$  information for computing  $g(x)$  from trace header,
- do forward temporal FFT (sign in transform is +1),
- store result in memory.

Loop over  $\omega$  from  $\omega_{\min}$  to  $\omega_{\max}$ :

- gather all  $x$  components for a given  $\omega$ ,
- solve system  $BF = f$  for  $F$  using a complex form of Levinson recursion.

Loop over  $p$ :

- gather all  $\omega$  for a given  $p$ ,
- do inverse temporal FFT (sign in transform is -1),
- put  $p$  information in trace header,
- output trace in  $\tau$ - $p$  domain.

**Suiradon** has a similar 3 loop structure with the following changes. The loop bringing in the  $\tau$ - $p$  data and doing the forward temporal Fourier transform is over  $p$  instead of  $x$ . The information for  $p$  is extracted from the trace headers. The information for  $g(x)$  must be supplied by the user. In the second loop, the system  $BF = f$  is solved for  $f$  by straightforward matrix multiplication. The final loop doing the inverse temporal FFT would be over  $x$  instead of  $p$ . The output data is in the (time,space) domain.

### ADDITIONAL CONSIDERATIONS

The least-squares formalism derived here assumes that  $n_p \leq n_x$ . Ideally, when  $n_p \geq n_x$ , the least-squares inverse should be computed as  $F = B^\dagger(BB^\dagger)^{-1}f$ . For

this formulation, the option to use Levinson recursion is lost. However, one can use standard matrix techniques to do the inversion. Once the decision is made to go to the cost of using standard matrix inversion methods, it is possible to expand the theory, for either  $n_p \leq n_x$  or  $n_p \geq n_x$ , to include operators with spatial weights. For example, spatial aperture weights could be included in a more complicated definition of the transform. Spatially variant anti-alias filters could be used. The theory also expands to cover other transforms such as Fourier-Bessel transforms. Anderson (1988) studied this broader class of transforms and compared inversions computed by Gauss-Jordan elimination, singular value decomposition, and, for applicable Toeplitz systems, Levinson recursion with Simpson's sideways recursion. Singular value decomposition provided the most accurate and consistently stable result, but at the greatest expense. Because, the inverse matrices are independent of the data, they can be computed once, stored, and reused, whenever a regular geometry can be assumed. However, the disk space required to store an inverse matrix for each frequency can be large, and the corresponding input/output effort is significant. The regular-geometry assumption can be very limiting because even when the acquisition geometry is regular, the editing of noisy traces and variations in the mute zone can make the geometry appear irregular to this algorithm. It far preferable to use irregular geometry than to fit artificially dead data when computing the model data in the transform domain.

In practice, one can use the Levinson recursion approach with  $n_p \geq n_x$  and prewhiten the inversion step to ensure stability. It is generally better to ensure that the basis functions of the model in transform space span all of the events in the data, with minimal aliasing, than to work with too small a value for  $n_p$ .

## CONCLUSION

For regular increments in generalized slowness  $p$  (i.e.,  $p = p_{\min} + k \cdot \delta p$ ), transforms of the form  $f(x, \omega) \equiv \sum_{k=0}^{n_p-1} F(p_k, \omega) e^{i\omega p_k g(x)}$  can be solved using a complex form of Levinson recursion. This general form allows the function  $g(x)$  to be  $x$  for 2-D linear  $\tau$ - $p$  transforms or  $x^2$  for parabolic transforms. Without giving up the Hermitian Toeplitz form required for rapid solution,  $g(x)$  could be defined as a more general function of spatial position if a geophysical application demanded it.

## ACKNOWLEDGMENTS

To a large extent, this paper represents more an exercise in learning the SU system,  $\text{\LaTeX}$ , and how to become compatible with CWP than anything related to new or original research. Craig Artley and Jack Cohen provided indispensable help. Ken Larner and Andreas Rueger proofread the paper and provided useful comments. The author was employed by British Petroleum during the 1987-88 time period when he first derived the algorithms presented here. R.G. Keys, S.T. Hildebrand, and D.E. Johnston provided useful discussions. As the Mobil Visiting Scientist at the Colorado School of Mines, the author is currently working with the Center for Wave Phenom-

ena, the Reservoir Characterization Project, the Center for Geoscience Computing, and Mobil.

## REFERENCES

- Anderson, J. E., 1988, The general discrete Radon transform— recent advances which enhance its utility for geophysical applications: BP internal research report.
- Bednar, J.B., 1988, personal communication...
- Beylkin, G., 1987, The discrete Radon transform: IEEE Transactions of Acoustics, Speech, and Signal Processing, **35**, 162–172.
- Chapman, C.H., 1981, Generalized Radon transforms and slant stacks: Geophysical Journal of the Royal Astronomical Society, **66**, 445–453.
- Claerbout, J.F., 1985, Fundamentals of geophysical data processing: Blackwell Scientific Publications.
- Darche, G., 1990, Spatial interpolation using a fast parabolic transform: SEG Expanded Abstracts 1990, 1647–1650.
- Foster, D.J. and Mosher, C.C., 1990, Multiple suppression using curvilinear Radon transforms: SEG Expanded Abstracts 1990, 1759.
- Foster, D.J. and Mosher, C.C., 1992, Suppression of multiple reflections using the Radon transform: Geophysics, **57**, NO. 3, (March 1992), 386–395.
- Gulunay, N., 1990, F-X domain least-squares Tau-P and Tau-Q: SEG Expanded Abstracts 1990, 1607–1610.
- Hampson, D., 1986, Inverse velocity stacking for multiple elimination: J. Can. Soc. Expl. Geophys., **22**, 44–55.
- Hampson, D., 1987, The discrete Radon transform: a new tool for image enhancement and noise suppression: SEG Expanded Abstracts 1987, 141–143.
- Johnston, D.E., 1990, Which multiple removal method should I use? SEG Expanded Abstracts 1990, 1750–1752.
- Kostov, C., 1989, Finite-aperture slant-stack transforms: SEP 61, 261.
- Kostov, C., 1990, Toeplitz structure in slant-stack inversion: SEG Expanded Abstracts 1990, 1618–1621.
- Sullivan, M., Schneider, W.A., and Shurtleff, R.N., 1990, Issues and applications of the Radon transform for multiple elimination: SEG Expanded Abstracts 1990, 1755–1758.
- Thorson, J.R., 1984, Velocity stack and slant stack inversion methods: PhD Thesis, Stanford University, May, 1984.

## APPENDIX A: LEVINSON RECURSION WITH HERMITIAN TOEPLITZ SYSTEMS

Solve the system  $RA = B$  where  $R$  is a Hermitian Toeplitz matrix,  $B$  is a known complex vector, and  $A$  is the complex solution vector to be determined.

As a part of the solution effort, we will find the solution to  $RF = S$ , where  $F$  is a complex spiking filter and  $S$  is a vector with a unit spike in location 0 and zeros everywhere else.

Levinson recursion for Toeplitz systems is discussed by Claerbout (1985). The critical differences worth noting in this derivation are which terms become complex conjugates for a Hermitian Toeplitz system.

Iteration 0:

Initialize  $v$ :  $v_0 = 1$

Solve for the spiking filter:  $f_0 = v_0/r_0$

Solve for the shaping filter:  $a_0 = b_0/r_0$

Higher Iterations:

Each successive iteration is built up using the solution from the previous iteration. For example, here we will build the 3 x 3 spiking filter for iteration 2 assuming that the 2 x 2 results from iteration 1 are available as listed below.

$$\begin{bmatrix} r_0 & r_1 \\ r_1^* & r_0 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \end{bmatrix} = \begin{bmatrix} v_1 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} r_0 & r_1 \\ r_1^* & r_0 \end{bmatrix} \begin{bmatrix} a_0 \\ a_1 \end{bmatrix} = \begin{bmatrix} b_0 \\ b_1 \end{bmatrix}$$

Note that subscript on  $v$  corresponds to the iteration number. Solve for  $c$  such that

$$\begin{bmatrix} r_0 & r_1 & r_2 \\ r_1^* & r_0 & r_1 \\ r_2^* & r_1^* & r_0 \end{bmatrix} \left\{ \begin{bmatrix} f_0 \\ f_1 \\ 0 \end{bmatrix} - c \begin{bmatrix} 0 \\ f_1^* \\ f_0^* \end{bmatrix} \right\} = \begin{bmatrix} v_2 \\ 0 \\ 0 \end{bmatrix}$$

Because we are working with a basis made up of spiking filters, we obtain a simple form after performing the matrix multiplication by  $R$ . The only terms that need to be computed are those for the top row and the bottom row.

$$\left\{ \begin{bmatrix} v_1 \\ 0 \\ e \end{bmatrix} - c \begin{bmatrix} e^* \\ 0 \\ v_1^* \end{bmatrix} \right\} = \begin{bmatrix} v_2 \\ 0 \\ 0 \end{bmatrix}$$

where

$$e = \sum_{k=0}^{N_{\text{iter}}-1} r_{N_{\text{iter}}-k}^* f_k$$

For this computation, the output value of  $v_2$  is not important as long as it does not equal 0. If  $v_2$  equals 0, then our system has become unstable and the recursion should stop. We only need to concentrate on the bottom row to estimate a value for  $c$ . Based on the computed value of  $c$ , one can compute new values for the spiking filter  $f$  and  $v_2$ .

$$e - cv_{N_{\text{iter}}-1}^* = 0$$

$$c = \frac{e}{v_{N_{\text{iter}}-1}^*}$$

$$(f_k)_{N_{\text{iter}}} = (f_k - cf_{N_{\text{iter}}-k}^*)_{N_{\text{iter}}-1}$$

$$v_{N_{\text{iter}}} = v_{N_{\text{iter}}-1} - ce^*$$

At the end of this iteration we have the 3 x 3 spiking filter solution for

$$\begin{bmatrix} r_0 & r_1 & r_2 \\ r_1^* & r_0 & r_1 \\ r_2^* & r_1^* & r_0 \end{bmatrix} \begin{bmatrix} f_0 \\ f_1 \\ f_2 \end{bmatrix} = \begin{bmatrix} v_2 \\ 0 \\ 0 \end{bmatrix}$$

Now, update the shaping filter  $A$  for this iteration by solving

$$\begin{bmatrix} r_0 & r_1 & r_2 \\ r_1^* & r_0 & r_1 \\ r_2^* & r_1^* & r_0 \end{bmatrix} \left\{ \begin{bmatrix} a_0 \\ a_1 \\ 0 \end{bmatrix} - c \begin{bmatrix} f_2^* \\ f_1^* \\ f_0^* \end{bmatrix} \right\} = \begin{bmatrix} b_0 \\ b_1 \\ b_2 \end{bmatrix}$$

Once again, we are taking advantage of the fact that  $F$  is a spiking filter so that we only need to pay attention to the bottom row of  $R$  for this computation.

$$\sum_{k=0}^{N_{\text{iter}}-1} r_{N_{\text{iter}}-k}^* a_k - cv_{N_{\text{iter}}}^* = b_{N_{\text{iter}}}$$

After computing  $c$ , we can update  $A$  and move on to the next iteration.

$$(a_k)_{N_{\text{iter}}} = (a_k)_{N_{\text{iter}}-1} - cf_{N_{\text{iter}}-k}^*$$

After the last iteration, the vector  $F$  should be normalized based on the current value of  $v$ . This ensures that  $RF = S$ , if  $F$  is a desired output from this computation.

## APPENDIX B: A SIMPLE PARABOLIC $\tau$ - $P$ TRANSFORM EXAMPLE

A very simple example of the parabolic  $\tau$ - $p$  transform demonstrates the utility of the least-squares approach. Figure B-1 displays a CMP gather that has a small-amplitude primary event properly flattened by NMO. A larger-amplitude multiple event still contaminates the data. An ideal parabolic transform would reveal that these data are the sum of only two parabolic events. Figure B-2 displays the parabolic transform obtained by using the conjugate operator. Note that the events are smeared together. Figure B-3 shows the discrete Radon transform result obtained using **suradon**. The least-squares approach does a better job of separating the two events in transform space.

A production-oriented processing code for parabolic filtering would operate somewhat differently than using **suradon**, followed by a parabolic  $\tau$ - $p$  mute, and **suiradon**. In practice, the desired result is a section with no multiples, created by an algorithm that is easy to use. A seismic processor would probably prefer to supply the maximum and minimum move-out error in ms at a reference offset rather than the parabolic shape factors used in **suradon**. The move-out timing error of a parabola at a reference offset can be readily picked from CMP gather displays. A parabolic filtering module would take care to not fit dead traces or muted regions during design of the model parabolas. Such care typically requires separate transforms for multiple time windows within the mute zone. **Suradon** operates on all of the data in only one time window, does not check for dead traces, and does not preserve all of the trace header information. A production-oriented module would operate in reject-filter mode, doing the forward transform, inverse transforming the estimated multiples, and subtracting the estimated multiples from the input data. Generally, it is preferable to apply a reject filter in order to preserve as much of the original data character as possible than to depend on the quality of a forward and inverse parabolic  $\tau$ - $p$  transform for every aspect of the data. The one-module, reject-filter approach also makes it easy to avoid stability problems by using only a limited range of frequencies for the reject filter without explicitly band-pass filtering the data. Because the data are input and output in the same domain, with the same number of traces per gather, it is easy to preserve all of the trace header information. **Suradon** and **suiradon** serve as general-purpose tutorial codes illustrating the method.

## APPENDIX B: A SIMPLE PARABOLIC $\tau$ - $P$ TRANSFORM EXAMPLE

A very simple example of the parabolic  $\tau$ - $p$  transform demonstrates the utility of the least-squares approach. Figure B-1 displays a CMP gather that has a small-amplitude primary event properly flattened by NMO. A larger-amplitude multiple event still contaminates the data. An ideal parabolic transform would reveal that these data are the sum of only two parabolic events. Figure B-2 displays the parabolic transform obtained by using the conjugate operator. Note that the events are smeared together. Figure B-3 shows the discrete Radon transform result obtained using `suradon`. The least-squares approach does a better job of separating the two events in transform space.

A production-oriented processing code for parabolic filtering would operate somewhat differently than using `suradon`, followed by a parabolic  $\tau$ - $p$  mute, and `suiradon`. In practice, the desired result is a section with no multiples, created by an algorithm that is easy to use. A seismic processor would probably prefer to supply the maximum and minimum move-out error in ms at a reference offset rather than the parabolic shape factors used in `suradon`. The move-out timing error of a parabola at a reference offset can be readily picked from CMP gather displays. A parabolic filtering module would take care to not fit dead traces or muted regions during design of the model parabolas. Such care typically requires separate transforms for multiple time windows within the mute zone. `Suradon` operates on all of the data in only one time window, does not check for dead traces, and does not preserve all of the trace header information. A production-oriented module would operate in reject-filter mode, doing the forward transform, inverse transforming the estimated multiples, and subtracting the estimated multiples from the input data. Generally, it is preferable to apply a reject filter in order to preserve as much of the original data character as possible than to depend on the quality of a forward and inverse parabolic  $\tau$ - $p$  transform for every aspect of the data. The one-module, reject-filter approach also makes it easy to avoid stability problems by using only a limited range of frequencies for the reject filter without explicitly band-pass filtering the data. Because the data are input and output in the same domain, with the same number of traces per gather, it is easy to preserve all of the trace header information. `Suradon` and `suiradon` serve as general-purpose tutorial codes illustrating the method.

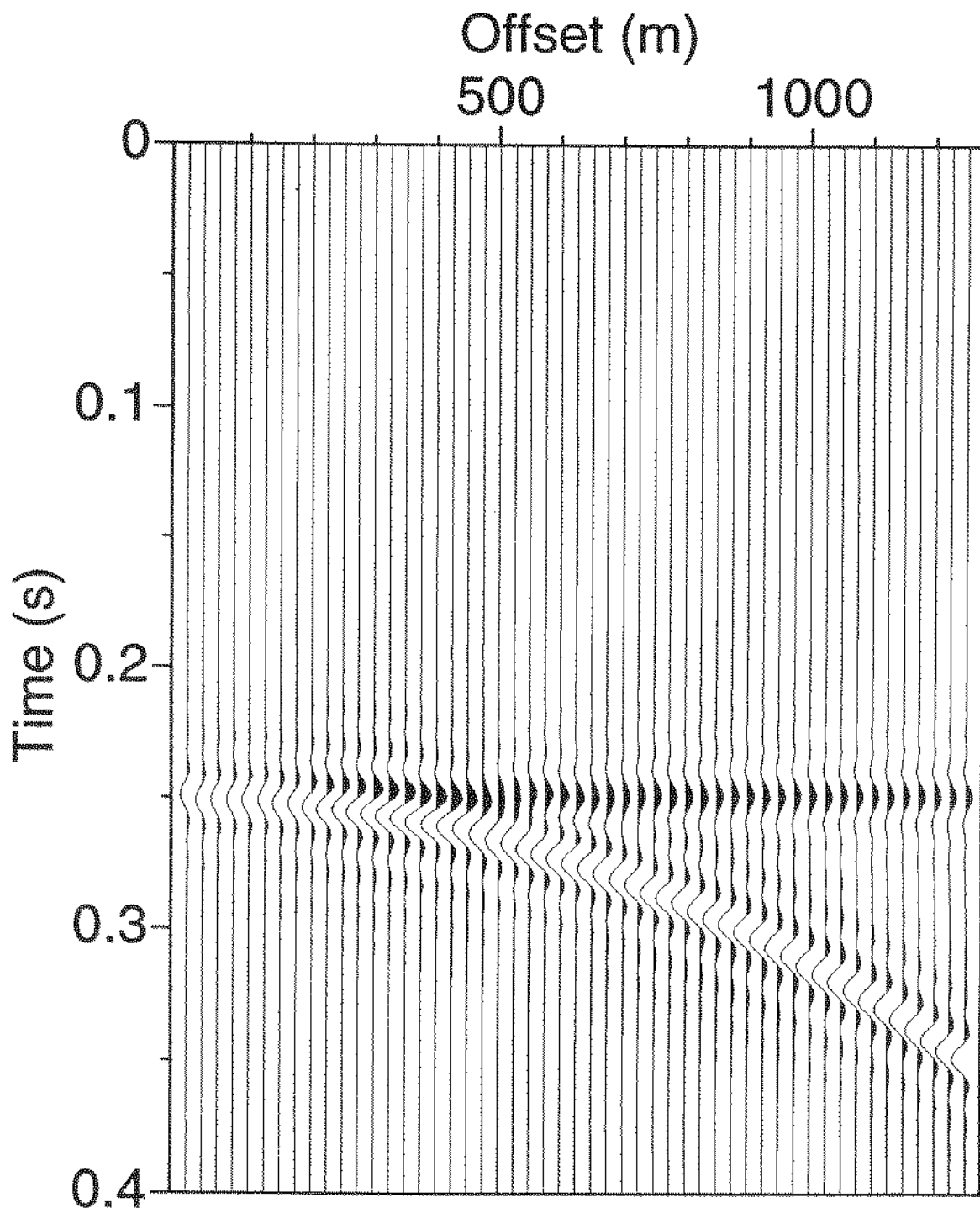


FIG. B-1. CMP gather with an aligned primary event (amplitude 1, parabola shape 0) and a multiple event (amplitude  $-2$ , parabola shape  $64 \cdot 10^{-9} \text{ (s/m)}^2$ ) both at the same 0 offset intercept time of 0.25 s. A zero-phase band-pass filter 5/10 80/100 Hz has been applied to the data.

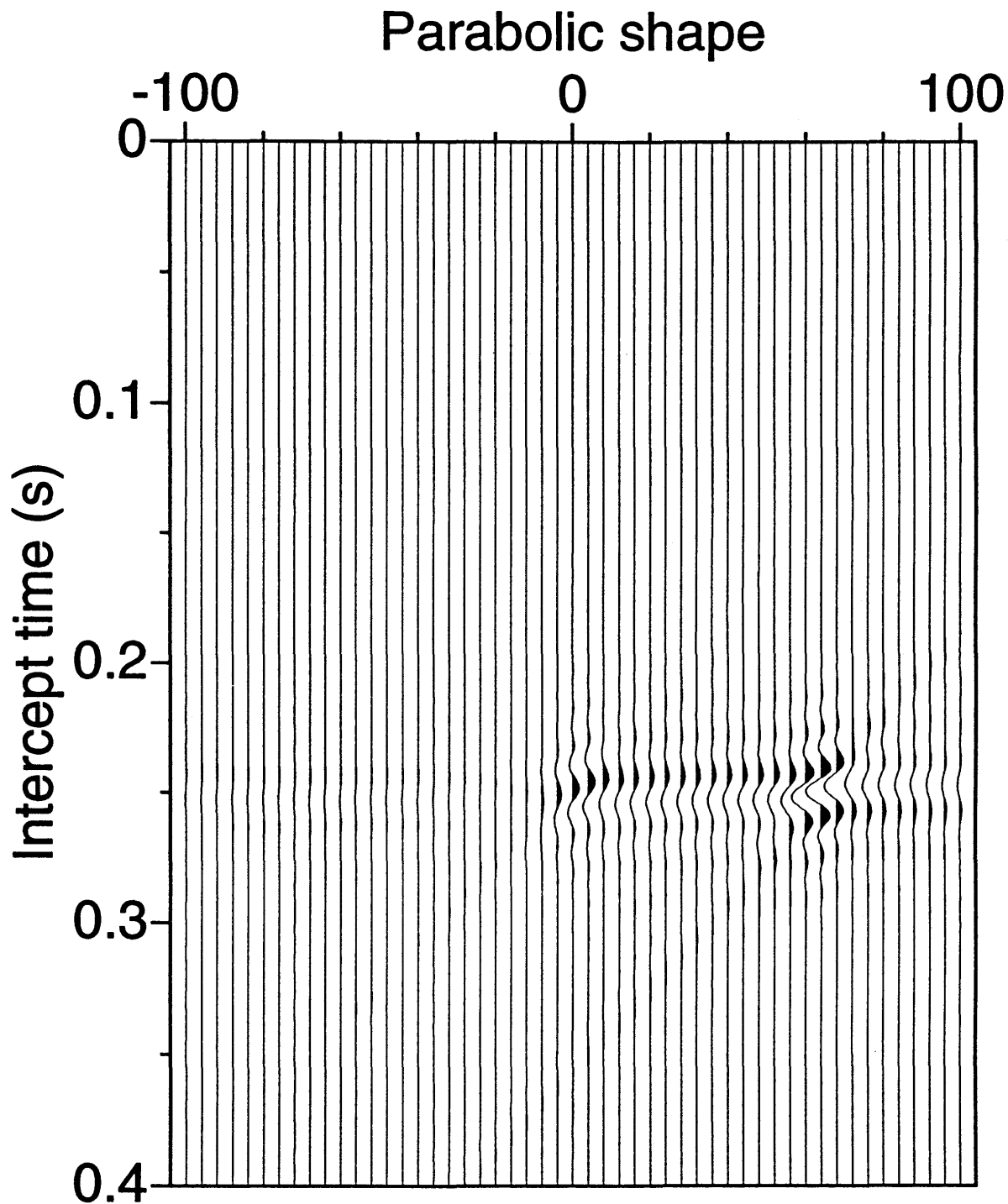


FIG. B-2. The conjugate method for computing the parabolic  $\tau$ - $p$  transform would compute  $F = |\omega|B^\dagger f$ . The scaling of the display covers up the fact that the overall gain of this result is incorrect. Compare the resolution of this result with that of the least-squares method. The horizontal axis labeled "parabolic shape" displays the parabolic  $p$  values, scaled by  $10^9$ , in units of  $(\text{s}/\text{m})^2$ .

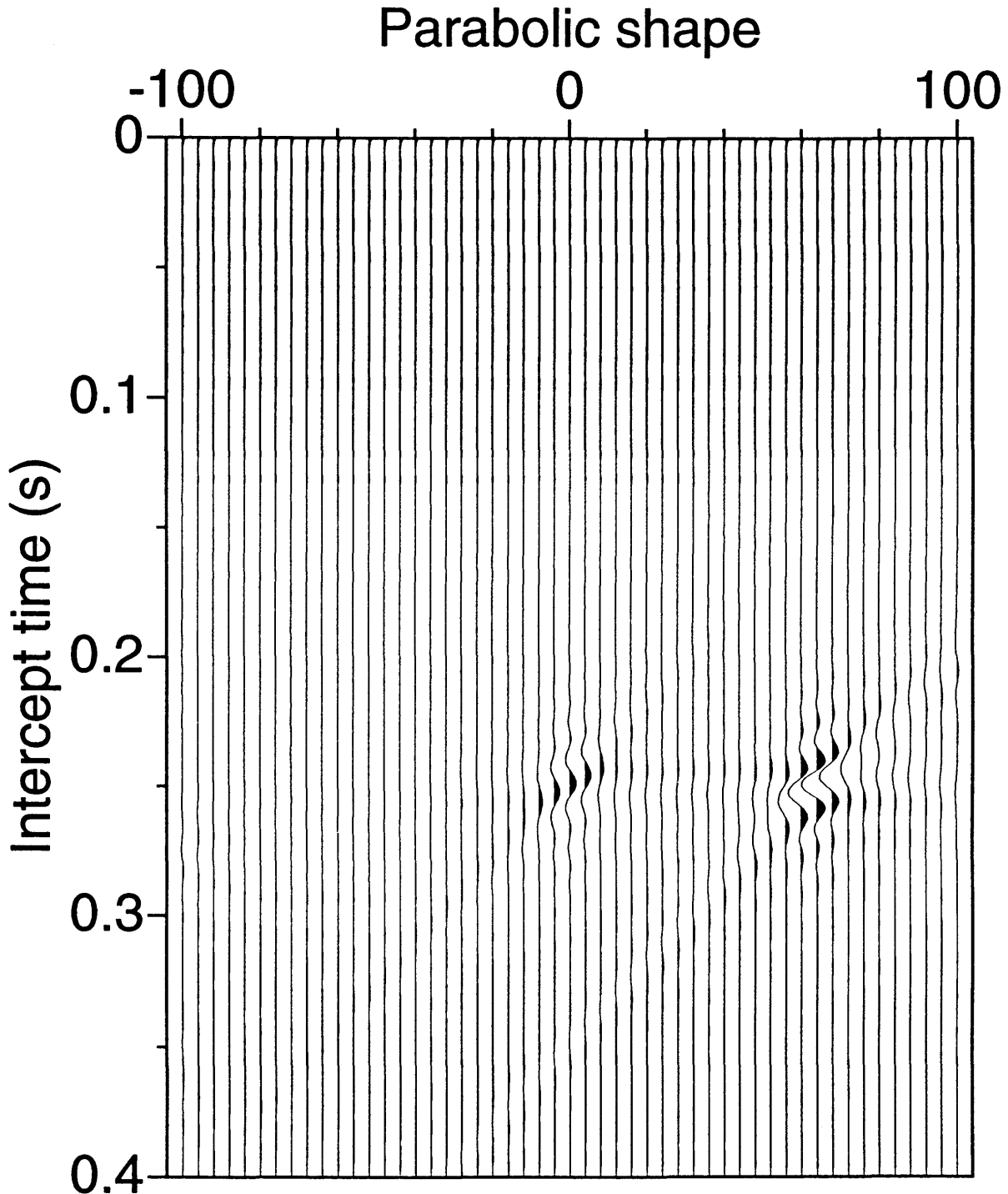


FIG. B-3. Using the least-squares approach for computing the transform provides better separation between the events in the parabolic  $\tau$ - $p$  domain. A perfect result would have a filtered amplitude of  $-2$  at parabola shape  $64 \cdot 10^{-9} \text{ (s/m)}^2$ , a filtered amplitude of  $1$  at parabola shape  $0$  for intercept time  $0.25 \text{ s}$ , and zero values everywhere else. The horizontal axis labeled "parabolic shape", displays the parabolic  $p$  values, scaled by  $10^9$ , in units of  $\text{(s/m)}^2$ .