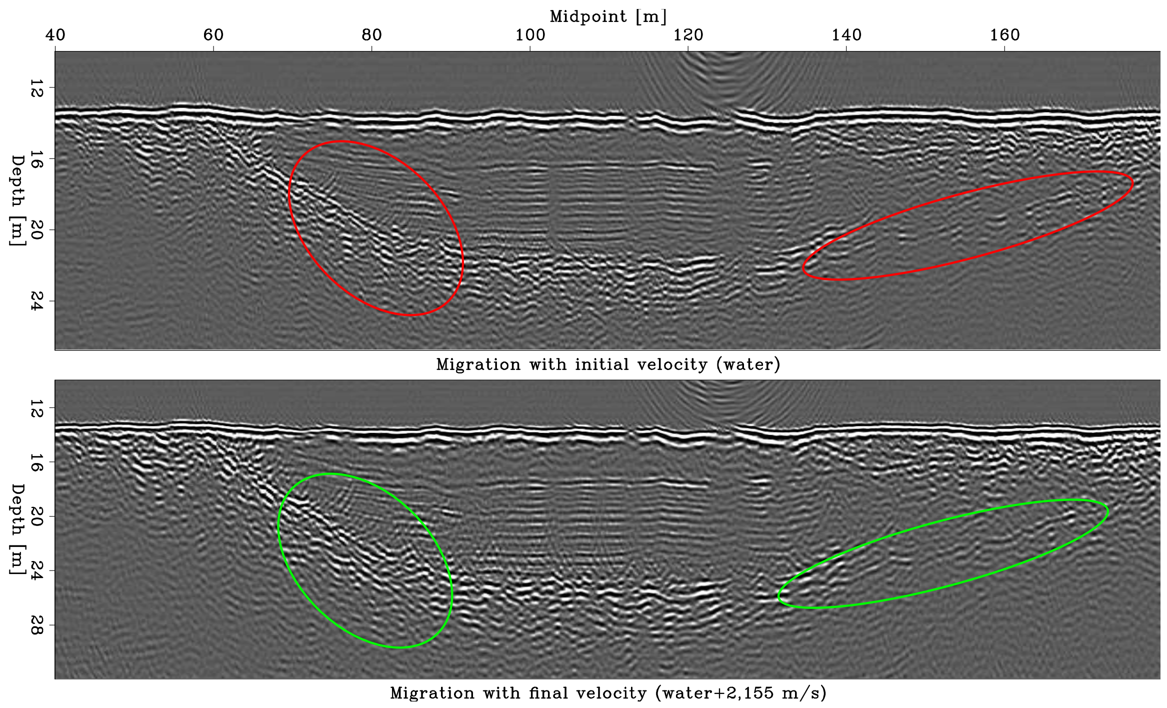
STANFORD EXPLORATION PROJECT

James Berryman, Biondo Biondi, Jon Claerbout, Robert Clapp, Haohuan Fu, Claudio Guerra, Antoine Guitton, Yunyue Li, Olav Lindtjorn, Mohammad Maysami, Nader Moussa, Shuki Ronen, and Mandy Wong

Report Number 139, October 2009



Preface

The electronic version of this report[1] makes the included programs and applications available to the reader. The markings [ER], [CR], and [NR] are promises by the author about the reproducibility of each figure result. Reproducibility is a way of organizing computational research that allows both the author and the reader of a publication to verify the reported results. Reproducibility facilitates the transfer of knowledge within SEP and between SEP and its sponsors.

ER denotes Easily Reproducible and are the results of processing described in the paper. The author claims that you can reproduce such a figure from the programs, parameters, and makefiles included in the electronic document. The data must either be included in the electronic distribution, be easily available to all researchers (e.g., SEG-EAGE data sets), or be available in the SEP data library[2]. We assume you have a UNIX workstation with Fortran, Fortran90, C, X-Windows system and the software downloadable from our website (SEP makerules, SEPlib, and the SEP latex package), or other free software such as SU. Before the publication of the electronic document, someone other than the author tests the author's claim by destroying and rebuilding all ER figures. Some ER figures may not be reproducible by outsiders because they depend on data sets that are too large to distribute, or data that we do not have permission to redistribute but are in the SEP data library.

CR denotes Conditional Reproducibility. The author certifies that the commands are in place to reproduce the figure if certain resources are available. The primary reasons for the CR designation is that the processing requires 20 minutes or more, or commercial packages such as Matlab or Mathematica.

NR denotes Non-Reproducible figures. SEP discourages authors from flagging their figures as NR except for figures that are used solely for motivation, comparison, or illustration of the theory, such as: artist drawings, scannings, or figures taken from SEP reports not by the authors or from non-SEP publications.

Our testing is currently limited to LINUX 2.6 (using the Intel Fortran90 compiler), but the code should be portable to other architectures. Reader's suggestions are welcome. More information on reproducing SEP's electronic documents is available online[3].

[1]http://sepwww.stanford.edu/private/docs/sep139
[2]http://sepwww.stanford.edu/public/docs/sepdatalib/toc_html
[3]http://sepwww.stanford.edu/research/redoc/

SEP139 — TABLE OF CONTENTS

Blocky models and Sparsity-promoting solvers

Jon Claerbout, Blocky models via the L1/L2 hybrid norm 1

Mohammad Maysami and Nader Moussa, Generalized-norm conjugate direction solver 11

Yunyue (Elita) Li and Mohammad Maysami, Dix inversion constrained by L1-norm optimization.................................. 23

Mandy Wong and Nader Moussa and Mohammad Maysami, Applications of the generalized norm solver.............................. 37

Nader Moussa, Alternatives to conjugate direction optimization for sparse solutions to geophysical problems 49

Velocity model building

Biondo Biondi, Measuring velocity from zero-offset data by image focusing analysis.. 57

Claudio Guerra, Gradient of image-space wave-equation tomography by the adjoint-state method 69

Mohammad Maysami, Geophysical data integration and its application to seismic tomography 83

Reverse Time Migration

Robert G. Clapp and Haohuan Fu and Olav Lindtjorn, Selecting the right hardware for Reverse Time Migration 91

Mandy Wong and Biondo L. Biondi and Shuki Ronen, Reverse Time Migration of up and down going signal for ocean bottom data.. 107

Miscellaneous

Antoine Guitton and Jon Claerbout, Theory and practice of interpolation in the pyramid domain.................................... 119

James G. Berryman, Schoenberg's angle on fractures and anisotropy: A study in orthotropy 137

James G. Berryman, Poroelastic measurements resulting in complete data sets for granular and other anisotropic porous media 145

SEP phone directory 167

Research personnel 169
('SEP article published or in press, 2009',) 173

Stanford Exploration Project, Report SEP139, October 19, 2009

Blocky models via the L1/L2 hybrid norm

Jon Claerbout

ABSTRACT

This paper seeks to define robust, efficient solvers of regressions of $L1$ nature with two goals: (1) straightforward parameterization, and (2) "blocky" solutions. It uses an $L1/L2$ hybrid norm characterized by a residual R_d of transition between $L1$ and $L2$ for data fitting and another R_m for model styling. Both the steepest descent and conjugate direction methods are included. The 1-D blind deconvolution problem is formulated in a manner intended to lead to both a blocky impedance function and a source waveform. No results are given.

INTRODUCTION

I've seen many applications improved when least-squares ($L2$) model fitting was changed to least absolute values ($L1$). I've never seen the reverse. Never-the-less we always return to $L2$ because the solving method is easier and faster. It does not require us to specify parameters of numerical analysis that are unclear how to specify.

Another reason to re-investigate $L1$ is its natural ability to estimate blocky models. Sedimentary sections tend to fluctuate randomly, but sometimes there is a homogeneous material continuing for some distance. A function with such homogeneous regions is called "blocky". The derivative of such a function is called "sparse". $L2$ gives huge penalties to large values and minuscule penalties to small ones, hence it never really produces sparse functions and their integrals are never really blocky. If we had an easy, reliable $L1$ solver, we could expect to see many more realistic solutions.

There are reasons to abandon strict $L1$ and revert to an $L1/L2$ hybrid solver. A hybrid solver has a parameter, a threshold, at which $L2$ behavior transits to $L1$. We have good reasons to use two different hybrid solvers, one for the data fitting, the other for the model styling (prior knowledge or regularization). Each requires a threshold of residual, let us call it R_d for the data fitting, and R_m for the model styling. Processes that require parameters are detestible when we have a poor idea of the meaning of the parameters (especially if they relate to numerical analysis), however the meaning of the thresholds R_d and R_m is quite clear. When we look at a shot gather and see about 30% of the area is covered with ground roll, it is clear we would like to choose R_d to be at about the 70th percentile of the fitting residual. As for the model styling, if we'd like to see blocks about 20 points long, we'd like our spikes to average about 20 points apart, so we would like R_m about the 95th percentile allowing 5% of the spikes to be of unlimited size, while the others small.

I was first attracted to strict $L1$ by its potential for blocky models. But then I realized for each nonspike (zero) on the time axis, theory says I would need a "basis equation". That implies an immense number of iterations, so it is unacceptable in imaging applications. With the hybrid solvers, instead of exact zeros we have a large region driven down by the L2 norm and a small L1 region where large spikes are welcomed.

MODEL DERIVATIVES

Here is the usual definition of residual r_i of theoretical data $\sum_j F_{i,j} m_j$ from observed data d_i

$$r_i = (\sum_j F_{i,j} m_j) - d_i \qquad \text{or} \qquad \mathbf{r} = \mathbf{F}\mathbf{m} - \mathbf{d}. \tag{1}$$

Let $C()$ be a convex function ($C'' \geq 0$) of a scalar. The penalty function (or norm of residuals is expressed by

$$N(\mathbf{m}) = \sum_i C(r_i) \tag{2}$$

We denote a column vector $\mathbf{g}$ with components g_i by $\mathbf{g} = \mathrm{vec(g_i)}$. We soon require the derivative of $C(r)$ at each residual r_i:

$$\mathbf{g} \quad = \quad \mathrm{vec}\left[\frac{\partial \mathrm{C(r_i)}}{\partial \mathrm{r_i}}\right] \tag{3}$$

We often update models in the direction of the gradient of the norm of the residual.

$$\Delta\mathbf{m} \quad = \quad \frac{\partial N}{\partial m_k} \quad = \quad \sum_i \frac{\partial C(r_i)}{\partial r_i}\frac{\partial r_i}{\partial m_k} \quad = \quad \sum_i g(r_i) F_{i,k} \quad = \quad \mathbf{F}'\mathbf{g} \tag{4}$$

Define a model update direction by $\Delta\mathbf{m} = \mathbf{F}'\mathbf{g}$. Since $\mathbf{r} = \mathbf{F}\mathbf{m} - \mathbf{d}$, we see the residual update direction will be $\Delta\mathbf{r} = \mathbf{F}\Delta\mathbf{m}$. To find the distance α to move in those directions

$$\mathbf{m} \quad \leftarrow \quad \mathbf{m} + \alpha\Delta\mathbf{m} \tag{5}$$

$$\mathbf{r} \quad \leftarrow \quad \mathbf{r} + \alpha\Delta\mathbf{r} \tag{6}$$

we choose the scalar α to minimize

$$N(\alpha) = \sum_i C(r_i + \alpha\Delta r_i) \tag{7}$$

The sum in equation (7) is a sum of “dishes”, shapes between L2 parabolas and L1 V’s. The i-th dish is centered on $\alpha = -r_i/\Delta r_i$. It is steep and narrow if Δr_i is large, and low and flat where Δr_i is small. The positive sum of convex functions is convex. There are no local minima. We can get to the bottom by following the gradient. Next we consider some choices for convex functions. We’ll need them, their first and second derivatives.

Some convex functions and their derivatives

LEAST SQUARES:

$$
\begin{aligned}
C &= r^2/2 && (8)\\
C' &= r && (9)\\
C'' &= 1 \qquad \geq\ 0 && (10)
\end{aligned}
$$

L1 NORM:

$$
\begin{aligned}
C &= |r| && (11)\\
C' &= \operatorname{sgn}(r) && (12)\\
C'' &= 0 \text{ or } \infty \qquad \geq\ 0 && (13)
\end{aligned}
$$

HYBRID:

$$
\begin{aligned}
C &= R^2(\sqrt{1+r^2/R^2}-1) && (14)\\
C' &= \frac{r}{\sqrt{1+r^2/R^2}} && (15)\\
C'' &= \frac{1}{(1+r^2/R^2)^{3/2}} \qquad \geq\ 0 && (16)
\end{aligned}
$$

HUBER:

$$
\begin{aligned}
C &= \left\{ \begin{array}{ll} |r|-R/2 & \text{if } |r| \geq R \\ r^2/2R & \text{otherwise} \end{array} \right\} && (17)\\
C' &= \left\{ \begin{array}{ll} \operatorname{sgn}(r) & \text{if } |r| \geq R \\ r/R & \text{otherwise} \end{array} \right\} && (18)\\
C'' &= \left\{ \begin{array}{ll} 0 \text{ or } \infty & \text{if } |r| \geq R \\ 1/R & \text{otherwise} \end{array} \right\} \qquad \geq\ 0 && (19)
\end{aligned}
$$

I have scaled Hybrid so it naturally approaches the least squares limit as $R \to \infty$. As $R \to 0$, it tends to $C = R|r|$, scaled $L1$.

Because of the erratic behavior of C'' for $L1$ and Huber, and our planned use of second order Taylor series, we will not be using $L1$ and Huber norms here. Also, we should prepare ourselves for danger as HYBRID approaches the $L1$ limit. (I thank Mandy for reminding me of the infinite second derivative and I thank Mohammad and Nader for demonstrating numerical erratic behavior.)

PLANE SEARCH

The most universally used method of solving immense linear regressions such as imaging problems is the Conjugate Gradient (CG) method. It has the remarkable property that in the presence of exact arithmetic, the exact solution is found in a finite number of iterations.

A simpler method with the same property is the Conjugate Direction method. It is debatable which has the better numerical roundoff properties, so we generally use the Conjugate Direction method as it is simpler to comprehend. It says not to move along the gradient direction line, but somewhere in the plane of the gradient and the previous step. The best move in that plane requires us to find two scalars, one α to scale the gradient, the other β to scale the previous step. That is all for $L2$ optimization. We proceed here in the same way with other norms and hope for the best.

So here we are, embedded in a giant multivariate regression where we have a bivariate regression (two unknowns). From the multivarate regression we are given three vectors in data space. $\bar{r}_i$, g_i and s_i. You will recognize these as the current residual, the gradient (Δr_i), and the previous step. (The gradient and previous step appearing here have previously been transformed to data space (the conjugate space) by the operator $\mathbf{F}$.) Our next residual will be a perturbation of the old one.

$$r_i \quad = \quad \bar{r}_i \ + \ \alpha g_i \ + \ \beta s_i \tag{20}$$

We seek to minimize by variation of (α, β)

$$N(\alpha,\beta) = \sum_i \ C(\bar{r}_i + \alpha g_i + \beta s_i) \tag{21}$$

Let the coefficients (C_i, C_i', C_i'') refer to a Taylor expansion of $C(r)$ about r_i.

$$N(\alpha,\beta) = \sum_i \ C_i \ + \ (\alpha g_i + \beta s_i) C_i' \ + \ (\alpha g_i + \beta s_i)^2 C_i''/2 \tag{22}$$

We have two unknowns, (α, β) in a quadratic form. We set to zero the α derivative of the quadratic form, likewise the β derivative getting

$$\begin{bmatrix} 0 \\ 0 \end{bmatrix} \quad = \quad \sum_i \ C_i' \begin{bmatrix} g_i \\ s_i \end{bmatrix} \ + \ C_i'' \left\{ \begin{bmatrix} \frac{\partial}{\partial \alpha} \\ \frac{\partial}{\partial \beta} \end{bmatrix} (\alpha g_i \ + \ \beta s_i) \right\} (\alpha g_i \ + \ \beta s_i) \tag{23}$$

resulting in a 2×2 set of equations to solve for α and β.

$$\left\{ \sum_i C_i'' \left[\begin{pmatrix} g_i \\ s_i \end{pmatrix} \begin{pmatrix} g_i & s_i \end{pmatrix} \right] \right\} \begin{bmatrix} \alpha \\ \beta \end{bmatrix} \quad = \quad - \sum_i C_i' \begin{bmatrix} g_i \\ s_i \end{bmatrix} \tag{24}$$

The solution of any 2×2 set of simultaneous equations is generally trivial. The only difficulties arise when the determinant vanishes which here is easy (luckily) to understand. Generally the gradient should not point in the direction of the previous step if the previous move went the proper distance. Hence the determinant should not vanish. Practice shows that the determinant will vanish when all the inputs are zero, and it may vanish if you do so many iterations that you should have stopped already, in other words when the gradient and previous step are both tending to zero.

Using the newly found (α, β), update the residual $\bar{r}_i$ at each location (and update the model). Then go back to re-evaluate C_i' and C_i'' at the new r_i locations. Iterate.

In what way do we hope/expect this new bivariate solver embedded in a conjugate direction solver to perform better than old IRLS solvers? After paying the inevitable price, a substantial price, of computing $\mathbf{F}'\mathbf{r}$ and $\mathbf{F}\ \Delta\mathbf{m}$ the iteration above does some serious thinking, not a simple linearization, before paying the price again.

If the convex function $C(r)$ were least squares, subsequent iterations would do nothing. Although the Taylor series of the second iteration would expand about different residuals r_i than the first iteration, the new second order Taylor series are the exact representation of the least squares penalty function, i.e. the same as the first – so the next iteration goes nowhere.

Will this computational method work (converge fast enough) in the $L1$ limit? I don't know. Perhaps we'll do better to approach that limit (if we actually want that limit) via gradually decreasing the threshold R.

BLOCKY LOGS: BOTH FITTING AND REGULARIZATION

Here we set out to find blocky functions, such as well logs. We will do data fitting with a somewhat $L2$-like convex penalty function while doing model styling with a more $L1$-like function. We might define the composite norm threshold residual R_d for the data fitting at the 60th percentile, and that for regularization seeking spiky models (with blocky integrals) as R_m at the 5th percentile.

The data fitting goal and the model regularization goal at each z is independent from that at all other z values. The fitting goal says the reflectivity $m(z)$ should be equal to its measurement $d(z)$ (the seismogram). The model styling goal says the reflectivity $m(z)$ should vanish.

$$0 \quad \approx \quad r_d(z) \ = \ m(z) - d(z) \tag{25}$$
$$0 \quad \approx \quad r_m(z) \ = \ \epsilon\ m(z) \tag{26}$$

These two goals are in direct contradiction to each other. With the L2 norm the answer would be simply $m = d/(1+\epsilon^2)$. With the L1 norm, the answer would be either $m = d$ or $m = 0$ depending on the numerical choice of ϵ. Let us denote the convex function and its derivatives for data space at the residual as (B, B', B'') and for model space as (C, C', C''). Remember, m and d, while normally vectors, are here scalars (independently for each z).

```
loop over all time points {
    $m = d/(1+\epsilon^2)$        # These are scalars!
    loop over non-linear iterations {
        $r_d = m - d$
        $r_m = m$
        Get derivatives of hybrid norm $B'(r_d)$ and $B''(r_d)$ for data goal.
        Get derivatives of hybrid norm $C'(r_m)$ and $C''(r_m)$ for model goal.
        # Plan to find $\alpha$ to update $m = m + \alpha$
        # Taylor series for data penalty $N(r_d) = B + B'\alpha + B''\alpha^2/2$
        # Taylor series for model penalty $N(r_m) = C + C'\alpha + C''\alpha^2/2$
        # $0 = \frac{\partial}{\partial\alpha}(N(r_d) + \epsilon N(r_m))$
        $\alpha = -(B' + \epsilon C')/(B'' + \epsilon C'')$
        $m = m + \alpha$
        } end of loop over non-linear iterations
    } end of loop over all time points
```

To help us understand the choice of parameters R_d, R_m, and ϵ, We examine the theoretical relation between m and d implied by the above code as a function of ϵ and R_m at $R_d \to \infty$, in other words, when the data has normal behavior and we are mostly interested in the role of the regularization drawing weak signals down towards zero. The data fitting penalty is $B = (m - d)^2/2$ and its derivative $B' = m - d$. The derivative of the model penalty (from equation (15)) is $C' = m/\sqrt{1 + m^2/R_m^2}$. Setting the sum of the derivatives to zero we have

$$0 \quad = \quad B' + \epsilon C' \quad = \quad m \ - \ d \ + \frac{\epsilon m}{\sqrt{1 + m^2/R_m^2}} \tag{27}$$

This says m is mostly a little smaller than d, but it gets more interesting near $(m, d) \approx 0$. There the slope $m/d = 1/(1 + \epsilon)$ which says an $\epsilon = 4$ will damp the signal (where small) by a factor of 5. Moving away from $m = 0$ we see the damping power of ϵ diminishes uniformly as m exceeds R_m.

UNKNOWN SHOT WAVEFORM

A one-dimensional seismogram $d(t)$ is unknown reflectivity $c(t)$ convolved with unknown source waveform $s(t)$. The number of data points ND≈NC is less than the number of unknowns NC+NS. Clearly we need a ”smart” regularization. Let us see how this problem can be set up so reflectivity $c(t)$ comes out with sparse spikes so the integral of $c(t)$ is blocky.

This is a nonlinear problem because the convolution of the unknowns is made of their product. Nonlinear problems elicit well-warranted fear of multiple solutions leading to us getting stuck in the wrong one. The key to avoiding this pitfall is starting “close enough” to the correct solution. The way to get close enough (besides luck and a good starting guess) is to define a linear problem that takes us to the neighborhood where a nonlinear solver can be trusted. We will do that first.

Block cyclic solver

In the Block Cyclic solver (I hope this is the correct term.), we have two half cycles. In the first half we take one of the variables known and the other unknown. We solve for the unknown. In the next half we switch the known for the unknown. The beauty of this approach is that each half cycle is a linear problem so its solution is independent of the starting location. Hooray! Even better, repeating the cycles enough times should converge to the correct solution. Hooray again! The convergence may be slow, however, so at some stage (maybe just one or two cycles) you can safely switch over to the nonlinear method which converges faster because it deals directly with the interactions of the two variables.

We could begin from the assumption that the shot waveform is an impulse and the reflectivity is the data. Then either half cycle can be the starting point. Suppose we assume we know the reflectivity, say **c**, and solve for the shot waveform **s**. We use the reflectivity **c** to make a convolution matrix **C**. The regression pair for finding **s** is

$$\begin{aligned} \mathbf{0} &\approx \mathbf{C s} - \mathbf{d} \qquad (28) \\ \mathbf{0} &\approx \epsilon_s \mathbf{I}\, \mathbf{s} \qquad (29) \end{aligned}$$

These would be solved for **s** by familiar least squares methods. It's a very easy problem because **s** has many fewer components than **c**. Now with our source estimate **s** we can define the operator **S** that convolves it on reflectivity **c**.

The second half of the cycle is to solve for the reflectivity **c**. This is a little trickier. The data fitting may still be done by an $L2$ type method, but we need something like an $L1$ method for the regularization to pull the small values closer to zero to yield a more spiky $c(t)$.

$$\begin{aligned} \mathbf{0} \approx_{L2} \mathbf{r}_d &= \mathbf{S c} - \mathbf{d} \qquad (30) \\ \mathbf{0} \approx_{L1} \mathbf{r}_m &= \mathbf{I}\, \mathbf{c} \qquad (31) \end{aligned}$$

Normally we expect an ϵ_c in equation (31) but now it comes in later. (It might seem that the regularization (29) is not necessary, but without it, **c** might get smaller and smaller while **s** gets larger and larger. We should be able to neglect regression (29) if we simply rescale appropriately at each iteration.) We can take the usual L2 norm to define a gradient vector for model perturbation $\Delta\mathbf{c} = \mathbf{S}'\mathbf{r}_d$. From it we get the residual perturbation $\Delta\mathbf{r}_d = \mathbf{S}\Delta\mathbf{c}$. We need to find an unknown distance α to move in those directions. We take the norm of the data fitting residual, add to it a bit ϵ of the model styling residual, and set the derivative to zero.

$$\mathbf{0} \quad = \quad \frac{\partial}{\partial \alpha} \left[N_d(\mathbf{r}_d + \alpha\, \Delta\mathbf{r}) \; + \; \epsilon N_m(\mathbf{c} + \alpha \Delta\mathbf{c}) \right] \qquad (32)$$

We need derivatives of each norm at each residual. We base these on the convex function $C(r)$ of the Hybrid norm. Let us call these A_i for the data fitting, and B_i for the model styling.

$$\begin{aligned} A_i &= C(R_d, r_i) \qquad (33) \\ B_i &= C(R_m, c_i) \qquad (34) \end{aligned}$$

(Actually, we don't need A_i (because for Least Squares, $A_i' = r_i$ and $A_i'' = 1$), but I include it here in case we wish to deal with noise bursts in the data.) As earlier, expanding the norms in Taylor series, equation (32) becomes

$$0 \quad = \quad \sum_i A_i'\, \Delta r_i \; + \; \alpha \sum_i A_i''\, \Delta r_i^2 \; + \; \epsilon \left(\sum_i B_i'\, \Delta c_i \; + \; \alpha \sum_i B_i''\, \Delta c_i^2 \right) \qquad (35)$$

which gives the α we need to update the model **c** and the residual $\mathbf{r}_d$.

$$\alpha \quad = \quad - \frac{\sum_i A_i'\, \Delta r_i \; + \; \epsilon \sum_i B_i'\, \Delta c_i}{\sum_i A_i''\, \Delta r_i^2 \; + \; \epsilon \sum_i B_i''\, \Delta c_i^2} \qquad (36)$$

This is the steepest descent method. For the conjugate directions method there is a 2×2 equation like equation (24).

Non-linear solver

The non-linear approach is a little more complicated but it explicitly deals with the interaction between **s** and **c** so it converges faster. We represent everything as a "known" part plus a perturbation part which we will find and add into the known part. This is most easily expressed in the Fourier domain.

$$0 \quad \approx \quad (S+\Delta S)(C+\Delta C) \ - \ D \tag{37}$$

Linearize by dropping $\Delta S \Delta C$.

$$0 \quad \approx \quad S\ \Delta C \ + \ C\ \Delta S \ + \ (CS-D) \tag{38}$$

Let us change to the time domain with a matrix notation. Put the unknowns ΔC and ΔS in vectors $\tilde{\mathbf{c}}$ and $\tilde{\mathbf{s}}$. Put the knowns C and S in convolution matrices $\mathbf{C}$ and $\mathbf{S}$. Express $CS-D$ as a column vector $\bar{\mathbf{d}}$. Its time domain coefficients are $d_0 = c_0 s_0 - d_0$ and $d_1 = c_0 s_1 + c_1 s_0 - d_1$, etc. The data fitting regression is now

$$\mathbf{0} \quad \approx \quad \mathbf{S}\tilde{\mathbf{c}} + \mathbf{C}\tilde{\mathbf{s}} + \bar{\mathbf{d}} \tag{39}$$

This regression is expressed more explicitly below.

$$\mathbf{0} \quad \approx \quad \left[\begin{array}{ccccccc|ccc} s_0 & . & . & . & . & . & . & c_0 & . & . \\ s_1 & s_0 & . & . & . & . & . & c_1 & c_0 & . \\ s_2 & s_1 & s_0 & . & . & . & . & c_2 & c_1 & c_0 \\ . & s_2 & s_1 & s_0 & . & . & . & c_3 & c_2 & c_1 \\ . & . & s_2 & s_1 & s_0 & . & . & c_4 & c_3 & c_2 \\ . & . & . & s_2 & s_1 & s_0 & . & c_5 & c_4 & c_3 \\ . & . & . & . & s_2 & s_1 & s_0 & c_6 & c_5 & c_4 \\ . & . & . & . & . & s_2 & s_1 & . & c_6 & c_5 \\ . & . & . & . & . & . & s_2 & . & . & c_6 \end{array} \right] \left[\begin{array}{c} \tilde{c}_0 \\ \tilde{c}_1 \\ \tilde{c}_2 \\ \tilde{c}_3 \\ \tilde{c}_4 \\ \tilde{c}_5 \\ \tilde{c}_6 \\ \hline \tilde{s}_0 \\ \tilde{s}_1 \\ \tilde{s}_2 \end{array} \right] \quad + \quad \left[\begin{array}{c} d_0 \\ d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \\ d_7 \\ d_8 \end{array} \right] \tag{40}$$

The model styling regression is simply $0 \approx C + \Delta C$, which in familiar matrix form is

$$\mathbf{0} \quad \approx \quad \mathbf{I}\ \tilde{\mathbf{c}} + \bar{\mathbf{c}} \tag{41}$$

It is this regression, along with a composite norm and its associated threshold that makes $c(t)$ come out sparse. Now we have the danger that $\mathbf{c} \rightarrow \mathbf{0}$ while $\mathbf{s} \rightarrow \infty$ so we need one more regression

$$\mathbf{0} \quad \approx \quad \mathbf{I}\ \tilde{\mathbf{s}} + \bar{\mathbf{s}} \tag{42}$$

We can use ordinary least squares on the data fitting regression and the shot waveform regression. Thus

$$\mathbf{0} \quad \approx \quad \left[\begin{array}{c} \mathbf{r_d} \\ \mathbf{r_s} \end{array} \right] \quad = \quad \left[\begin{array}{cc} \bar{\mathbf{S}} & \bar{\mathbf{C}} \\ \mathbf{0} & \bar{\mathbf{I}} \end{array} \right] \left[\begin{array}{c} \tilde{\mathbf{c}} \\ \tilde{\mathbf{s}} \end{array} \right] \quad + \quad \left[\begin{array}{c} \bar{\mathbf{d}} \\ \bar{\mathbf{s}} \end{array} \right] \tag{43}$$

$$\mathbf{0} \quad \approx \quad \mathbf{r} \quad = \quad \mathbf{Fm} \ + \ \mathbf{d} \tag{44}$$

The model styling regression is where we seek spiky behavior.

$$\mathbf{0} \quad \approx \quad \mathbf{r_c} \ = \ \mathbf{I}\ \tilde{\mathbf{c}} \ + \ \bar{\mathbf{c}} \tag{45}$$

The big picture is that we minimize the sum

$$\min_{\tilde{\mathbf{c}},\tilde{\mathbf{s}}} \quad N_d(\mathbf{r}) \ + \ \epsilon \mathbf{N_m}(\mathbf{r_c}) \tag{46}$$

Inside the big picture we have updating steps

$$\Delta\mathbf{m} \quad = \quad \mathbf{F}'\mathbf{r} \tag{47}$$

$$\mathbf{g_d} = \mathbf{\Delta r} \quad = \quad \mathbf{F\Delta m} \tag{48}$$

We also have a gradient for changing $\tilde{\mathbf{c}}$, namely $\mathbf{g_c} = -\bar{\mathbf{c}}$. (I need to be sure $\mathbf{g_d}$ and $\mathbf{g_c}$ are commensurate. Maybe need an ϵ here.) One update step is to choose a line search for α

$$\min_{\alpha} \quad N_d(\mathbf{r} + \alpha\mathbf{g_d}) \ + \ \epsilon\mathbf{N_m}(\bar{\mathbf{c}} + \alpha\mathbf{g_c}) \tag{49}$$

That was steepest descent. The extension to conjugate direction is straightforward.

As with all nonlinear problems there is the danger of bizarre behavior and multiple minima. To avoid frustration, while learning you should spend about half of your effort directed toward finding a good starting solution. This normally amounts to defining and solving one or two linear problems. In this application we might get our starting solution for $s(t)$ and $c(t)$ from conventional deconvolution analysis, or we might get it from the block cyclic solver.

Stanford Exploration Project, Report SEP139, October 19, 2009

Generalized-norm conjugate direction solver

Mohammad Maysami and Nader Moussa

ABSTRACT

In optimization problems, the L_1 norm outperforms the L_2 norm in presence of noise and when a blocky or sparse solution is appropriate. These applications call for a solver that can redefine the optimum criteria for a particular problem. We have implemented a generalized norm solver that is useful for a wide range of problems. Our solver modularizes the norm function so that it can easily be interchanged to experiment with different schemes on any particular geophysical problem. We implement L_1, L_2, and two additional norms: Huber and Hybrid L_1/L_2. These are useful for problems that seek the benefits of both the L_1 and L_2 norms.

INTRODUCTION

Strict L_1 norm optimization has numerous applications in geophysical inversion problems. Examples include solving for sparse functions, such as those that describe blocky, layered geology. However, we find that approximations to L_1, or modified L_1/L_2 systems, are more computationally feasible than pure L_1 solutions. These non-strict L_1/L_2 norms – the Huber (Huber, 1973; Guitton, 2000) and Hybrid norms – still satisfy our desire to find sparse-functions, and are suitable for most of our geophysical needs.

For many years, SEP has relied on a single incarnation of the conjugate-direction descent solver. This code is based on work by Claerbout (2008). For some time in the 1990s, SEP also used a competing least squares approach, *LSQR*, developed by Paige and Saunders (1982). Because numerical optimization is such an embedded part of many geophysical algorithms, it is extremely desirable to have a backward-compatible program framework that can continue to work within these decades-old codes. For this reason, we chose to implement our solver with the same interface as the conjugate-direction L_2 solver (Claerbout, 2008). Our new solver implements a generalized definition of the norm used for minimization.

It should be noted that the term “norm” is used for convenience; whereas in fact, some of the numeric measures under consideration do not strictly satisfy the necessary mathematical criteria to be a proper norm. Notably, the Huber and Hybrid norms exhibit non-linearity with regard to a scalar multiplication (Bube and Langan, 1997). We will use this terminology for simplicity, but the cautious reader should note that these functions do not satisfy all necessary properties of a norm.

We considered many solver options in our original quest for a numeric solution to the strict L_1 optimization problem. We investigated a weighted-median algorithm to descend to the L_1-sense minimum. Early development showed that, despite the theoretical promise of the weighted-median methodology, it did not suitably achieve the desired solution on non-trivial test problems. This work led to development of a new solver, discussed below, based on theory developed in Claerbout (2009). This solver is based on a generalized plane-search using Taylor series expansion of the norm.

In this paper, we will describe our implementation of this solver framework. It is based on SEP's conjugate-direction L_2 solver, but it also allows us to substitute different norms, including L_1 and our non-strict L_1/L_2 norms. First we will review the available norms which can be used as the optimization criteria. Then we will discuss the mechanics of our new solver framework and describe a technique for iterative plane-search, potentially enabling faster convergence for certain classes of problems. Finally, we will reference applications which are described in other SEP reports (Li and Maysami, 2009; Wong et al., 2009).

NORM OPTIONS

Our solver framework allows easy interchangeability between several norms. Although the code allows easy switching of the optimization measure, we recommend a thorough understanding of the theoretical and numerical caveats that result from the application of each solver criterion.

Norm	Description
L2	Conventional L_2 norm, utilizing the new solver framework
L1	L_1 norm with discontinuous 1^{st} and 2^{nd} order derivatives
Huber	Huber L_1/L_2 norm with with 1^{st} order derivative continuity
Hybrid	L_1/L_2 hybrid norm with 1^{st} and 2^{nd} order derivative continuity

The equations below summarize the analytical formulation for the listed norms above. $C,\ C', and\ C''$ represent the norm function, its first-order derivative and its second-order derivative, respectively. Figure 1 shows these norm functions along with their derivatives. Note discontinuous derivatives and zero-valued curvatures in some cases. These analytical forms are used in the Taylor series expansion for the adapted conjugate-direction plane-search.

L2 (Least Squares):

$$\begin{aligned} C(r) &= r^2/2 \\ C'(r) &= r \\ C''(r) &= 1 \end{aligned} \tag{1}$$

L1:

$$\begin{aligned} C(r) &= |r| \\ C'(r) &= \mathrm{sgn}(r) \\ C''(r) &= 0 \quad \text{or} \quad \infty \end{aligned} \tag{2}$$

Huber:

$$\begin{aligned}
C(r) &= \begin{cases} |r| - |r_t|/2 & |r/r_t| \geq 1 \\ r^2/2r_t & |r/r_t| < 1 \end{cases} \\
C'(r) &= \begin{cases} \operatorname{sgn}(r/r_t) & |r/r_t| \geq 1 \\ r/r_t & |r/r_t| < 1 \end{cases} \\
C''(r) &= \begin{cases} 0 & |r/r_t| \geq 1 \\ 1/r_t & |r/r_t| < 1 \end{cases}
\end{aligned} \tag{3}$$

Hybrid:

$$\begin{aligned}
C(r) &= r_t^2 \left(\sqrt{1 + r^2/r_t^2} - 1 \right) \\
C'(r) &= \frac{r}{\sqrt{1 + r^2/r_t^2}} \\
C''(r) &= \frac{1}{(1 + r^2/r_t^2)^{\frac{3}{2}}}
\end{aligned} \tag{4}$$

The choice of norm is specified as an input argument to our solver. A further benefit of this implementation is that other norms can be added with minimal modification to the overall solver framework. To add a new norm, all that is necessary is adding the appropriate definition of the norm and its derivatives in the code.

SOLVER INTERNAL MECHANISMS

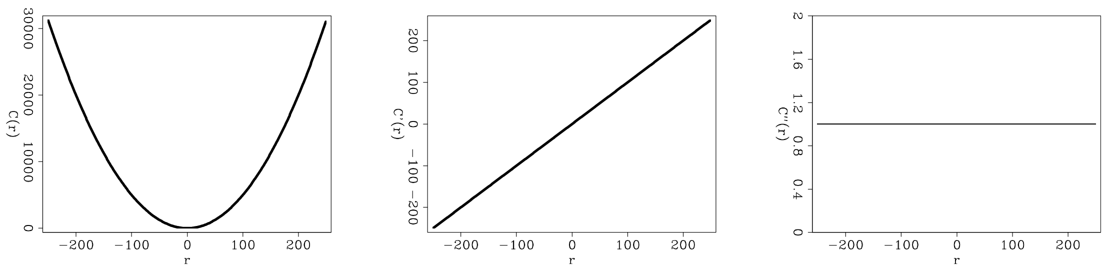
In any optimization scheme, we always attempt to minimize some measure of a data or model residual, $\mathbf{r}$. This measure, $C(\mathbf{r})$, is usually a convex function (commonly the L_2 norm, for least-squares fitting). For our solver, $C(\mathbf{r})$ can be any of the norms listed in previous section. As proposed by Claerbout (2009), the numerical value of a norm at an updated residual value, $C(\mathbf{r}_2)$ can be estimated based on a second-order Taylor series decomposition at point $\mathbf{r}_1$:

$$C(\mathbf{r}_2) \approx \; C(\mathbf{r}_1) + \frac{(\mathbf{r}_2 - \mathbf{r}_1)}{1!} \; C'(\mathbf{r}_1) + \frac{(\mathbf{r}_2 - \mathbf{r}_1)^2}{2!} \; C''(\mathbf{r}_1) \tag{5}$$

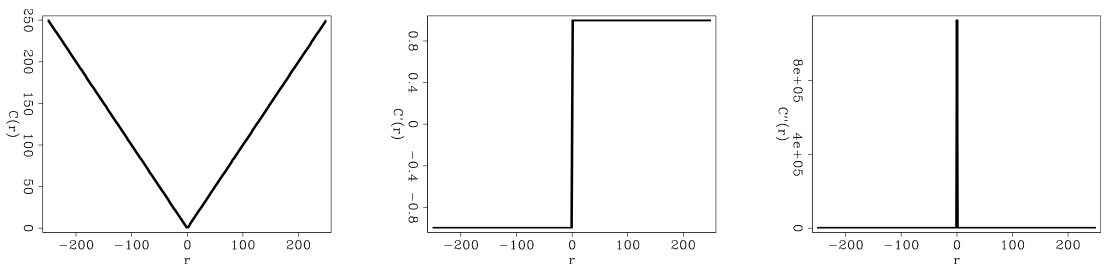
where r_2 is the point in a close neighborhood of r_1. With this generalization, we can conduct an iterative plane-search at any point r_2, without re-evaluating the forward operator. For operators with a high-op count per sample this is a less costly by finding a more optimal update to the solution.

Iterated Plane-Search

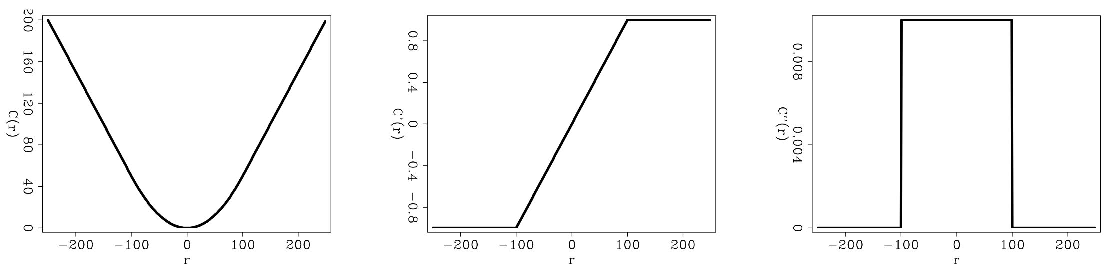
In the conjugate-direction descent (Claerbout, 2008), the update in the model and residual is defined by a linear combination of the current gradient of the objective function and the direction of the previous update step. However, this update can be iterated inside an inner



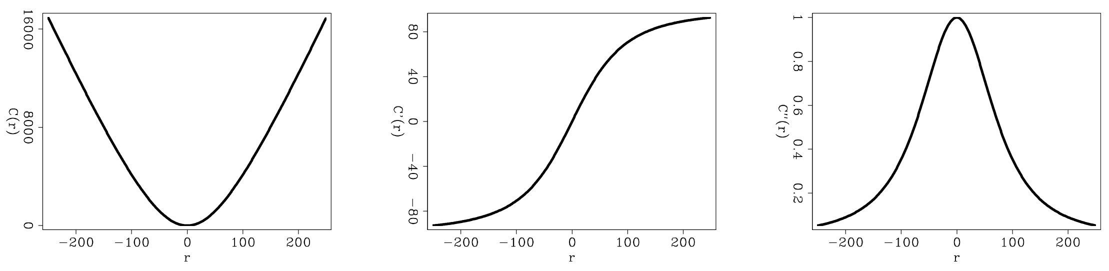
(a)



(b)



(c)



(d)

Figure 1: Norm functions and their first and second derivatives plotted for $r = -250, 250$ interval, with $r_t = 100$ where applicable. Rows from top to bottom are representing **(a)**L_2, **(b)** L_1, **(c)** Huber, and **(d)** Hybrid. Columns from left to right are representing norm function, first-order derivative, second-order derivative. **[ER]** mohammad1/. l2-norm,l1-norm,huber-norm,hybrid-norm

loop, assuming a linearization of the gradient around the current residual. Our algorithm can perform an iterated search in the plane defined by the current gradient and the previous update step. This search can locate the optimal update without a full re-evaluation of the gradient operator. In practice, this is equivalent to finding the solution of a 2×2 system of equations, where the unknowns are the step lengths in the direction of the gradient $\mathbf{g}^{(m)}$ and the previous step $\mathbf{s}^{(m)}$.

Taylor series expansion of the objective function around the initial residual value, assumes a small local neighborhood for the plane-search. We re-calculate this approximation at every inner iteration; this is an essential element of the plane-search, to ensure a reasonable degree of accuracy without a full re-computation of the forward operator in the main body of solver.

Algorithm Pseudo-code

The steps of actual solver have a structure very similar steps as the L_2 solver. The chief difference lies in the generalization to allow for different norms in the gradient calculation. The generalized form of the gradient for a given norm $C(\cdot)$ is given by

$$\mathbf{g}^{(m)} \;=\; \Delta\mathbf{m} \;=\; \mathbf{F}^T \, C'(\mathbf{r}) \;, \tag{6}$$

where F is the forward operator, $\mathbf{m}$ is the model value and $\mathbf{g}^{(m)}$ is the gradient. In addition, our code allows for an iterative plane-search to update $\mathbf{r}$, the intermediate estimated residual. The pseudo-code in Algorithm 1 summarizes the implementation for a simple solver (see Claerbout (2009) for more details). The benefit of our framework is that it can be easily modified to allow regularization and preconditioning without extensively changing the main solver algorithm. Such modifications primarily affect the stepper code. Fortran implementations of plane-search stepper function for both conjugate-direction and our generalized norm stepper are given in the appendices. Note that we approximate the data-space value of the gradient throughout the entire plane-search by a constant, to avoid re-evaluating the forward operator every iteration.

$$\mathbf{F}\mathbf{g}_i^{(m)} \approx \mathbf{F}\mathbf{g}_0^{(m)} \tag{7}$$

This assumption forces our plane-search to remain in a local neighborhood around $\mathbf{r}_1$, which is the desired outcome for this local approximation anyway.

NEW EXTERNAL PARAMETERS

Setting threshold with percentile

The benefit of a mixed L_1/L_2 norm is that small residuals can be optimized in an L_2 sense while large residuals are treated by L_1. This requires a definition of "small residual"; how small is "small"?

Our implementation addresses this issue with a numerical parameter, r_t. This is the threshold for transition between L_1 and L_2 in Huber and Hybrid norms. According to the analytic definition of each norm, r_t adjusts the crossover point. Needless to say, pure L_1 and L_2 have no such transition. To reduce the problem-specific dependency of r_t, we

Algorithm 1 Generalized Norm Solver Algorithm

```
$\mathbf{m} = \mathbf{m}_0$
$\mathbf{s}^{(m)}, \mathbf{s}^{(d)} = \mathbf{0}$

$\mathbf{\Delta m} = \mathbf{0}$
$\mathbf{\Delta r} = \mathbf{0}$
$\alpha = 0,\ \beta = 0$

!! Main iteration loop
for $iter = 1, niter$ do
  $\mathbf{r} = \mathbf{Fm} - \mathbf{d}$

  $\mathbf{g}^{(m)} = \mathbf{\Delta m} = \mathbf{F}'\ C'(\mathbf{r})$    !! Gradient in model space
  $\mathbf{g}^{(d)} = \mathbf{\Delta r} = \mathbf{F}\ \mathbf{g}^{(m)}$    !! Gradient in data space

  !! Plane-search loop
  for $i = 1, psiter$ do

    !! Solve for $\alpha$, $\beta$
    $\sum C_i''(\mathbf{r}) \left[ \begin{pmatrix} g_i^{(d)} \\ s_i^{(d)} \end{pmatrix} \begin{pmatrix} g_i^{(d)} & s_i^{(d)} \end{pmatrix} \right] \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = -\sum C_i'(\mathbf{r}) \begin{pmatrix} g_i^{(d)} \\ s_i^{(d)} \end{pmatrix}$

    $\mathbf{s}^{(m)} = \mathbf{\Delta m} = \alpha\ \mathbf{g}^{(m)} + \beta\ \mathbf{s}^{(m)}$    !! update solution step
    $\mathbf{s}^{(d)} = \mathbf{\Delta r} = \alpha\ \mathbf{g}^{(d)} + \beta\ \mathbf{s}^{(d)}$    !! update residual step

    $\mathbf{m} = \mathbf{m} + \mathbf{s}^{(m)}$    !! update solution
    $\mathbf{r} = \mathbf{r} + \mathbf{s}^{(d)}$    !! update residual

    if $\sum \|s_i^{(m)}/m_i\| < 10^{-6}$ then

        Quit plane-search iteration loop
    end if
  end for

end for
```

have configured our solver to compute this threshold based on a user-defined percentile. By switching to percentile, we retain a physical meaning for this user-specified parameter. It is possible to use separate thresholds, and even different norms, for the model- and data-fitting goals of a general regularized or preconditioned optimization problem.

Plane-search iteration count

Our iterative plane-search adds additional flexibility if the user so desires. The conventional conjugate-direction descent stepper (Claerbout, 2008) lacks this feature since it only searches the plane of the gradient and the previous step once. To utilize our plane-search the user may specify another value for this parameter, allowing the solver to scan the local neighborhood around the internally-computed residual values. When the plane-search iteration is greater than one, the solver will estimate that many model updates between each full function evaluation. This is particularly useful if the forward-operator is computationally expensive.

SUMMARY

We have provided this implementation of the generalized norm solver to SEP for benchmarking and testing on a variety of sample problems. These experiments have been documented in separate SEP reports which focuses on the geophysical ramifications of the L_1 and modified L_1 optimization criteria. In conclusion, our solver has been designed to be interchangeable with existing codes, requiring minimal code modification. We hope this will encourage other researchers inside and outside SEP to experiment with these available optimization objectives.

APPENDIX A: ANALYTICAL DERIVATION OF PLANE-SEARCH STEP SIZES

This appendix shows the details on generalization of plane-search algorithm for a general norm (or measure) $C(\cdot)$. As discussed previously and by Claerbout (2009), we use Taylor series expansion to find analytical forms for the step sizes in the plane-search algorithm. We form the updates in the residual value $\mathbf{r}$ by a linear combination of the gradient $\mathbf{g}^{(d)}$ and the previous step update of the residual $\mathbf{s}^{(d)}$, i.e. $\mathbf{r} = \mathbf{r} + \alpha\mathbf{g}^{(d)} + \beta\mathbf{s}^{(d)}$. Then the misfit objective function $E(\mathbf{r})$ is given by

$$\begin{aligned} E(\mathbf{r}) &= \sum_i C(r_i + \alpha g_i^{(d)} + \beta s_i^{(d)}) \\ &= \sum_i C(r_i) + \frac{\alpha\ g_i^{(d)} + \beta\ s_i^{(d)}}{1!} C'(r_i) + \frac{\left(\alpha\ g_i^{(d)} + \beta\ s_i^{(d)}\right)^2}{2!} C''(r_i) \end{aligned} \tag{8}$$

where r_i is the residual from the current iteration. The Taylor series expansion in Equation 8 lets us find analytical derivatives of the misfit function $E(\mathbf{r})$ with respect to both α and

β as follows:

$$\frac{\partial E}{\partial \alpha} = \sum_i g_i^{(d)} C'(r_i) + (\alpha\ g_i^{(d)} + \beta\ s_i^{(d)}) g_i^{(d)} C''(r_i) = 0, \tag{9}$$

$$\frac{\partial E}{\partial \beta} = \sum_i s_i^{(d)} C'(r_i) + (\alpha\ g_i^{(d)} + \beta\ s_i^{(d)}) s_i^{(d)} C''(r_i) = 0. \tag{10}$$

By setting these derivatives to zero and solving the 2×2 system of equations we find an optimal step size in both directions $\mathbf{g}^{(d)}$ and $\mathbf{s}^{(d)}$. The equation below shows the solutions α and β for this system of equations in a simplified notation.

$$\sum C_i''(r) \left[\begin{pmatrix} g_i^{(d)} \\ s_i^{(d)} \end{pmatrix} \begin{pmatrix} g_i^{(d)} & s_i^{(d)} \end{pmatrix} \right] \begin{bmatrix} \alpha \\ \beta \end{bmatrix} = -\sum C_i'(r) \begin{pmatrix} g_i^{(d)} \\ s_i^{(d)} \end{pmatrix} \tag{11}$$

APPENDIX B: FORTRAN CODES FOR CGSTEP AND GENERALIZED CGNORM PLANE-SEARCH STEPPER

This appendix includes the fortran codes for `cgstep.f90` and `cgnorm.f90`. Note that type of norm and threshold are set inside an initialization subroutine which is called prior to stepper call.

cgstep.f90

```
integer function cgstep(forget, x, g, rr, gg)
  real, dimension (:)    :: x, g, rr, gg
  logical                :: forget
  double precision       :: alfa, beta, determ
  double precision       :: sds, gdg, gds, gdr, sdr
  if ( .not. allocated (s)) then
    forget = .true.
    allocate ( s (size ( x)))
    allocate (ss (size (rr)))
  end if
  if ( forget) then
    s = 0.
    ss = 0.
    beta = 0.d0     ! steepest descent
    if ( dot_product(gg, gg) .eq. 0 ) then
      call erexit('cgstep: grad vanishes identically')
    end if
    alfa = - sum( dprod( gg, rr)) / sum( dprod(gg, gg))
  else
    gdg = sum( dprod( gg, gg))
    ! search plane by solving 2-by-2
    sds = sum( dprod( ss, ss))
    !  G . (R - G*alfa - S*beta) = 0
    gds = sum( dprod( gg, ss))
    !  S . (R - G*alfa - S*beta) = 0
    if ( gdg.eq.0. .or. sds.eq.0.) then
      cgstep = 1
      return
    end if
    determ = gdg*sds * max(1.d0-(gds/gdg)*(gds/sds),1.d-12)
    gdr    = - sum( dprod( gg, rr))
    sdr    = - sum( dprod( ss, rr))
    alfa   = ( sds * gdr - gds * sdr ) / determ
    beta   = (-gds * gdr + gdg * sdr ) / determ
  end if
  s  = alfa *  g + beta *  s          ! update solution step
  ss = alfa * gg + beta * ss          ! update residual step
  x  =  x +  s                        ! update solution
  rr = rr + ss                        ! update residual
  forget = .false.
  cgstep = 0
end function
```

cgnorm.f90

```
 subroutine cgnorm_init(rthr,norm,psiter)
    ! Initialize CGNORM Stepper with r_thr, norm type and psiter_in
    !
    ! Arguments IN:
    !    rthr      : RBAR or R_THR for norm
    !    norm_in   : Norm ype to be used (Hybrid or Huber)
    !    psiter_in : Number of Plane-search iterations on Alpha,Beta
    !
    real           :: rthr
    integer        :: psiter
    character(10)  :: norm
    optional       :: psiter,norm

   ps_norm="huber"
    if (present(norm)) then
       ps_norm=norm
    end if

    ps_iter=1
    if (present(psiter)) then
       ps_iter=psiter
    end if

   rt = abs(rthr)      ! Rt is positive
    if (ps_norm(1:5)=="huber" .or. ps_norm(1:6)=="hybrid") then
       if (rt==0.0)  call erexit('r_thr = 0. is not a valid choice.')
    endif

  end subroutine

  ! --------------------------------------------
  !              CGNORM function
  ! --------------------------------------------
  integer function cgnorm( forget, x, g, rr, gg)
    ! Iterative Plane-Search Stepper function
    !    for different norms (Hybrid, Huber, etc.)

    ! Arguments:
    ! x, g   : x, dx
    ! rr,gg  : R, dR
    ! forget : set steepest-decsend if true and CG-direction if false
```

```
! Output Flag (cgnorm):
!   2 : Exit
!   1 : Exit with vanishing gradient status
!   0 : Done without any exit or error
!
double precision, dimension(:), allocatable     :: sds, gdg, gds
double precision, dimension(:), allocatable     :: c0,c1,c2
real, dimension(:), allocatable     :: check
real, dimension (:)   :: x, g, rr, gg
logical               :: forget
double precision      :: alfa, beta, determ
double precision      :: c2dgg,c2dss,c2dgs,c1dg,c1ds
integer               :: i,ps,stat

cgnorm = 2
allocate (c0(size(rr)),c1(size(rr)),c2(size(rr)))
allocate (gdg(size(rr)),sds(size(rr)),gds(size(rr)))
allocate ( check(size( x)))

if( .not. allocated(s)) then
   forget = .true.
   allocate ( s(size( x)))
   allocate (ss(size(rr)))
end if

! Loop over alpha , Beta
do ps=1,ps_iter

   ! === Compute Norm and its derivatives
   if      (ps_norm(1:2)=="l2")      then
      stat = l2      (rr, c0,c1,c2)
   else if (ps_norm(1:2)=="l1")      then
      stat = l1      (rr, c0,c1,c2)
   else if (ps_norm(1:6)=="hybrid")  then
      stat = hybrid  (rr, c0,c1,c2)
   else
      stat = huber   (rr, c0,c1,c2)
   end if

  ! === Pick Line Search or Plane Search
   if( forget) then
      ! Steepest Descent
      s    = 0.
      ss   = 0.
      beta = 0.d0

      gdg   = dprod(gg,gg)
      c2dgg = sum(dprod(c2,gdg))

      !!! Discard C" if C"= 0  (OPTIONAL)
      ! if (sum(abs(c2))==0.0) c2dgg=sum(gdg)

      if( c2dgg == 0.0 )  then
         cgnorm = 1
         call erexit('NORM_CGSTEP: Gradient vanishes: C"*gg*gg=0.')
      end if
      alfa = - sum( dprod(c1,gg)) / c2dgg
      else
         ! Plane Search for alpha & beta
         c1dg = - sum(dprod(c1,gg))  ! c1dg = -C'*G
         c1ds = - sum(dprod(c1,ss))  ! c1ds = -C'*S

         gdg = dprod(gg,gg)       ! search plane by solving 2-by-2
         sds = dprod(ss,ss)       ! C"*G. (G*alfa + S*beta) = -C'*G
         gds = dprod(gg,ss)       ! C"*S. (G*alfa + S*beta) = -C'*S


         c2dgg = sum(dprod(c2,gdg))  ! c2dgg = C"*G*G
         c2dss = sum(dprod(c2,sds))  ! c2dss = C"*S*S
         c2dgs = sum(dprod(c2,gds))  ! c2dgs = C"*G*S

         !if (sum(abs(c2))==0.0) then
         ! !!! Discard C" if C"= 0 (OPTIONAL)
         ! c2dgg=sum(gdg)
         ! c2dss=sum(sds)
         !end if

         if ((c2dgg==0.d0) .OR. (c2dss==0.d0)) then
            cgnorm = 1;
            write (0,*) 'Plane Search : Gradient vanishes:
            C"*gg*gg=0 OR C"*ss*ss=0.'
            return
         end if
         determ = c2dgg * c2dss * max(1.d0 - (c2dgs/c2dgg)*(c2dgs/c2dss),1.
             d-12)
         alfa = ( c2dss * c1dg - c2dgs * c1ds ) / determ
         beta = (-c2dgs * c1dg + c2dgg * c1ds ) / determ
      end if

      ! Updates on model and residual
      s  = alfa *  g + beta *  s          ! update solution step
      ss = alfa * gg + beta * ss          ! update residual step

      check=0.
      do i=1,size(x)
          check=check+(s(i)/x(i))**2
      end do
      if  (check< 1.e-6) exit

      x  =  x +  s                        ! update solution
      rr = rr + ss                        ! update residual
      forget = .false.;

      !write (0,*) "Alpha, Beta::" ,alfa,beta
      !write (0,*) "Determ::" ,determ
   end do


   deallocate (c0,c1,c2)
   deallocate (gdg,sds,gds)
   cgnorm = 0
end function
```

REFERENCES

Bube, K. P. and R. T. Langan, 1997, Hybrid l^1/l^2 minimization with applications to tomography: Geophysics, **62**, 1183–1195.

Claerbout, J. F., 2008, Image estimation by example.

———, 2009, Blocky models via the $l1/l2$ hybrid norm: SEP-Report, **139**, 1–10.

Guitton, A., 2000, Implementation of a nonlinear solver for minimizing the huber norm: SEP-Report, **103**, 281–289.

Huber, P. J., 1973, Robust regression: Asymptotics, conjectures, and monte carlo: Annals of Statistics, **1**, 799–821.

Li, Y. E. and M. Maysami, 2009, Dix inversion constrained by l1-norm optimization: SEP-Report, **139**, 23–36.

Paige, C. C. and M. A. Saunders, 1982, Algorithm 583 lsqr: spare linear equations and sparse least squares problems: ACM Transaction on Mathematical Software, **8**, 195–209.

Wong, M., N. W. Moussa, and M. Maysami, 2009, Applications of generalized norm solver: SEP-Report, **139**, 37–48.

Stanford Exploration Project, Report SEP139, October 19, 2009

Dix inversion constrained by L1-norm optimization

Yunyue (Elita) Li and Mohammad Maysami

ABSTRACT

To accurately invert for velocity in a model with a blocky interval velocity inversion using Dix inversion, we set up our optimization objective function using $L1$ criterion. In this study, we analyze and test an improved version of the Iterative Reweighted Least Squares (IRLS) solver, a hybrid $L1/L2$ solver and a conjugate direction $L1$ solver. We use a 1-D synthetic velocity data set and a 1-D field RMS velocity data set as test cases. The results of the inversion are promising for applications on realistic geophysical problems.

INTRODUCTION

Dix formula (Dix, 1952) estimates interval velocities from picked stacking velocities. The conventional result of constrained least-squares Dix inversion (Koren and Ravve, 2006; Harlan, 1999; Clapp, 2001) is always a smooth velocity model, because the regularization is imposed in the $L2$ sense. However, to represent a geological environment with sharp velocity contrasts, e.g., carbonate layers, salt bodies and strong faulting, we may need a blocky velocity model rather than a smooth velocity model. Valenciano et al. (2003) proposed to use edge-preserving regularization with Dix inversion in order to get sharp edges in interval velocity. However, one of the solvers they used, IRLS (Iterative Re-weighted Least Squares) is cumbersome to use because users must specify numerical parameters with unclear physical meaning.

$L1$-norm optimization is known to be a robust estimator to yield sparse models. Many works (Claerbout and Muir, 1973; Darche, 1989; Nichols, 1994; Guitton, 2005) has shown that $L1$-norm is not sensitive to outliers, while it penalizes the small residuals down to zero. In theory, when the model space is sparse and the data are noisy, regressions produced by $L1$ optimization always outperform those produced by $L2$ norms.

In this study, we analyze, improve and test different methods on a simple synthetic problem as well as a field-data problem. We aim to develop robust and efficient solvers to perform regressions of an $L1$ nature. We initially improve the traditional IRLS method, explore the conjugate direction $L1$ method, and finally test an $L1/L2$ hybrid method. The inversion results of a 1-D, synthetic, 2-step, interval-velocity model and a 1-D field data example are given at the end of the paper.

DIX INVERSION AS AN $L1$-OPTIMIZATION PROBLEM

The linear relationship between the RMS velocity and the square of the interval velocity is given by Dix Equation:

$$v_k^2 \;=\; kV_k^2 - (k-1)V_{k-1}^2, \tag{1}$$

where v is the interval velocity, V is the stacking velocity or RMS velocity, and k is the sample number. Both velocities run down the traveltime depth axis. If we define $u_k = v_k^2$ and $d_k = kV_k^2$, we can set up the Dix inversion problem in an $L1$ sense as follows:

$$||\boldsymbol{W_d}(\boldsymbol{C}\mathbf{u} - \mathbf{d})||_1 \approx 0, \tag{2}$$

where $\mathbf{u}$ is the unknown model we are inverting for, $\mathbf{d}$ is the known data from velocity scan, $\mathbf{C}$ is the causal integration operator, $\mathbf{W}_d$ is a data residual weighting function, which is proportional to our confidence in the RMS velocity.

Fitting goal (2) itself cannot fully constrain the inversion problem, because the integration operator has a large null space at high frequencies. Therefore, Clapp et al. (1998) supplement this system with a regularization term to take the advantage of the prior geological information, of which smoothness and blockiness are two typical examples. For the case we are interested in, we use blockiness as regularization. In a mathematical form, it can be written as follows:

$$||\epsilon \boldsymbol{D_z}\mathbf{u}||_1 \approx 0, \tag{3}$$

where $\mathbf{D}_z$ is the vertical derivative of the velocity model and ϵ is the weight controlling the strength of the regularization.

DIX INVERSION BY IRLS METHOD

Bube and Langan (1997) and Tang (2006) show that the nonlinear objective functions, such as the regression equation (3), can be solved by the IRLS algorithm. Many authors have demonstrated successful applications of IRLS as a robust estimator to yield sparse models. To take advantage of the well-established $L2$ norm regression, we can transform the problem by introducing a diagonal weighting function $\boldsymbol{W_p}$. Then the fitting goal 2 and the regularization 3 become:

$$||\boldsymbol{W_d}(\boldsymbol{C}\mathbf{u} - d)||_2 \approx 0, \tag{4}$$

$$||\epsilon \boldsymbol{W_p} \boldsymbol{D_z}\mathbf{u}||_2 \approx 0, \tag{5}$$

where $\boldsymbol{W_p}$ is a diagonal weighting function on the model residual. To use the gradient-based method, we have to recompute the weight $\boldsymbol{W_p}$ at each iteration, and the algorithm can be summarized as follows:

1. Set the initial weighting matrix $\boldsymbol{W_p^{(0)}}$ to equal the identity matrix:

$$\boldsymbol{W_p^{(0)}} \; = \; I \tag{6}$$

2. At the $k - th$ iteration, the $i - th$ element of the weighting matrix is recomputed as

$$W_{pi}^{(k-1)} \; = \; (|p_i^{k-1}|)^{-1/2} \tag{7}$$

where $\mathbf{p} = \boldsymbol{D_z}\mathbf{u}$ is the model residual. However, applying equation (7) to compute the weighting matrix is dangerous when p_i is zero or too small. One way to avoid this issue is to bound the weights at a certain cutoff σ:

$$\mathbf{W}_{pi}^{(k-1)} = \begin{cases} (|p_i^{k-1}|)^{-1/2} & \text{if } |p_i^{k-1}| \geq \sigma \\ \sigma^{-1/2} & \text{otherwise.} \end{cases} \tag{8}$$

One of the most important disadvantages of IRLS algorithm arises in equation 8: how should we choose the cutoff number σ? We would like to derive this number automatically according to its physical meaning, instead of cumbersome numerical experiments.

Further examining the weighting function, we notice that when applying the truncated weights, we end up treating small residuals in the $L2$ norm, and at the turning point ($p = \sigma$) we have a sharp transition to the $L1$ norm. Thus, σ is the cutoff between the $L1$ region and the $L2$ region, determining the tolerance to the large residuals. Therefore, we can choose σ according to the desired blockiness of the model space. For the synthetic example, which is a 40-point-long interval velocity model with three layers, we expect only two spikes out of those 40 points in the derivative. Therefore we would like σ to be around the 95^{th} percentile of the derivative, allowing 5% of the spikes to be of unlimited size, while the others are small.

DIX INVERSION BY AN $L1/L2$ HYBRID METHOD

Extending the discussion in last section, many authors (Bube and Langan, 1997; Claerbout, 2009) generalize the optimization problem of Dix inversion. Claerbout (2009) points out that any arbitrary norm P can be used as a penalty function. Then the optimization problem can be written as:

$$0 \approx \sum_i P(\sum_j C_{i,j} u_j - d_i), \tag{9}$$

where $P()$ is a convex function of a scalar, $C_{i,j}$, u_j, and d_i are elements in the causal integration operator C, the model u and the known data d.

We have special interests in an $L1/L2$ hybrid norm, because instead of a sharp transition, this norm provides a smooth transition between $L1$ and $L2$. This can be shown in the formulation of the hybrid norm:

$$P \;=\; \sigma^2(\sqrt{1 + r^2/\sigma^2} - 1) \tag{10}$$

where r is the data residual and σ is the same threshold as in last section, and thus can be chosen according to the same physical explanation. The hybrid norm approaches the least squares limit as $\sigma \rightarrow \infty$ and approaches the $L1$ limit as $\sigma \rightarrow 0$.

The first and the second derivative of this hybrid norm with respect to the residuals are given in equation 11 and equation 12. We can see the penalty function transits from $L2$ to $L1$ smoothly at $r = \sigma$ since the first derivative is continuous.

$$P' \;=\; \frac{r}{\sqrt{1 + r^2/\sigma^2}} \tag{11}$$

$$P'' \;=\; \frac{1}{(1 + r^2/\sigma^2)^{3/2}} \tag{12}$$

Claerbout (2009) also proposed a new method based on Taylor's series to search the plane spanned by the gradient and the previous step. He embedded the new iterative bivariate solver in a conjugate direction solver, hoping for significant savings by expending more effort to find a better next step. In this experiment, we use the solver coded by Maysami and Mussa (2009), who adapt Claerbout's theory. For more information, refer to these two papers.

DIX INVERSION BY CONJUGATE DIRECTION $L1$ METHOD

Choosing proper parameters for hybrid $l1/l2$ method and IRLS is still quite empirical, even when we understand their physical meanings. Instead, conjugate direction methods do not need setting parameters. Thus, we develop a similar conjugate direction method in $L1$ sense. The pseudo code of this method is given in Table 1.

Table 1: Pseudo Code - Conjugate direction $L1$ solver using Weighted Median

```
Initialization :
    $\mathbf{m} = \mathbf{m}_{init}$
    $\mathbf{r}^{(d)} = \boldsymbol{F}\mathbf{m} - \mathbf{d}$
    $\mathbf{s} = \mathbf{0}$
    $\mathbf{s}^{(d)} = \mathbf{0}$
Iteration i :
    $\mathbf{r}^{(t)} = sgn(\mathbf{r}^{(t)})$
    $\mathbf{g} = \boldsymbol{F'}\mathbf{r}^{(t)}$                    *
    $\mathbf{g}^{(d)} = \boldsymbol{F}\mathbf{g}$                    *

    $(\alpha,\beta) = \underset{(\alpha,\beta)}{\text{argmin}}||\mathbf{r}^{(t)} + \alpha\mathbf{g}^{(d)} + \beta\mathbf{s}^{(d)}||_1$
    $\mathbf{s}^{(d)} = \alpha\mathbf{g}^{(d)} + \beta\mathbf{s}^{(d)}$
    $\mathbf{r}^{(d)} = \mathbf{r}^{(d)} + \mathbf{s}^{(d)}$
    $\mathbf{s} = \alpha\mathbf{g} + \beta\mathbf{s}$
    $\mathbf{m} = \mathbf{m} + \mathbf{s}$
```

The structure of the conjugate-direction $L1$ method is similar to the $L2$ conjugate-direction solver given by Claerbout (2008). The main difference arises in the part of plane search.

Given the gradient $\mathbf{g}^{(d)}$, previous step $\mathbf{s}^{(d)}$, and the current residual $\mathbf{r}^{(d)}$, we construct the $2 \times N$ matrix $\mathbf{B} = [\mathbf{g}^{(d)}\ \mathbf{s}^{(d)}]$ and the column vector $[\alpha\ \beta]'$. We seek to find $[\alpha\ \beta]'$ that minimizes $\mathbf{0} \approx \mathbf{r}^{(d)} + \mathbf{B}[\alpha\ \beta]'$ in the L1-norm sense. This bivariate regression embedded in the plane search is solved in an iterate manner.

At the ultimate solution of the bivariate regression there will be two basis equations that are exactly satisfied. The first one is found by steepest descent. After the first iteration, we do plane searches using the weighted median solver to choose the best equation to be exactly satisfied. "Best equation" is the one that decreases the residual the most, while satisfying the equation chosen by the previous iteration exactly as well.

To do this, suppose the previous equation is $g_k^{(d)}\alpha + s_k^{(d)}\beta + r_k^{(d)} = 0$. We seek a $(\Delta\alpha, \Delta\beta)$ that still satisfies the equation k. This requirement gives a solution to $(\Delta\alpha, \Delta\beta)$ as $\gamma(s_k^{(d)}, -g_k^{(d)})$, where γ is a scalar. Then the plane search becomes $\mathbf{0} \approx \gamma\mathbf{B}[s_k^{(d)}, -g_k^{(d)}]' + r^{(d)}$, which is a weighted median problem. Thus, using the weighted median solver, we can solve for γ and get a new equation, e.g., equation j. Then we can update (α, β) and $\mathbf{s}^{(d)}$ accordingly, drop the old equation k, and keep the new equation j. We iterate on this process until the inner loop keeps tracking the same equation. The final results of the inner loop are passed out to update the model, residual and the gradient.

The value of expending more effort to find the best step direction will be supported by the real geophysical applications, because the most computationally expensive part of these iterative methods is applying the forward and adjoint operators (steps starred in Table 1). By doing the sophisticate plane search, we hope to decrease the number of outer-loop iterations required for convergence.

However, conjugate direction L1 regression theory is not perfect for a practical problem. The problem of a flat bottom in $L1$ minimization will cause trouble in geophysical practice. Sometimes even where the bottom is not exactly as flat as the median of an even number of points, the slope of the gradient can be so small that we can never reach convergence in a finite number of iterations.

SYNTHETIC AND FIELD DATA EXAMPLE

Figure 1 shows the input synthetic RMS velocities with and without random noise, and the true blocky interval velocity we try to invert for. For the synthetic problem, we experimented on solvers with and without regularization to learn the nature of the solver itself and the nature of its regularization.

Figure 2 shows the inversion results when clean data (free of noise) are fed into different simple solvers without any regularization. The result of $L2$ regression is comparable to the IRLS and hybrid norm. However, the simple conjugate direction $L1$ solver failed to give a satisfactory result (Figure 2(c)). As we have discussed in the previous section, this might be due to the flat bottom caused by the data configuration.

Figure 3 shows the inversion results when clean data are fed into different regularized solvers. As expected, the smoothing effect of $L2$ regularization on the derivative of model produces the round corners at the turning point. In contrast, IRLS and hybrid solvers give perfect exact solutions, which benefit from their $L1$ nature in regularization. We do not fully understand the behavior of conjugate-direction $L1$ solver, but the change in the result can be explained by the change of the data configuration when the regularization term is added.

Figure 4 shows the inversion results when noisy data are fed into different simple solvers without any regularization. When creating the synthetic noisy RMS velocity data, uniform distributed random noise is added. However, $L2$ norm assumes Gaussian noise, and $L1$ minimization is derived under the assumption of exponential distribution. Therefore, the simple $L2$ solver, the IRLS solver or hybrid solver all fail to recognize the noise and attenuate it. Surprisingly, conjugate-direction $L1$ solver successfully eliminates the high-frequency noise and keeps the low-frequency trend of the interval velocity function. It shows great potential for finding the exact solution when the problem is slightly more complicated.

Figure 5 shows the inversion results when noisy data are fed into different regularized solvers. By adding this regularization term to further constrain the problem, we expect better results out of each solver. Comparing the results in Figure 5, IRLS result has the most blocky transition between layers and is almost flat within the layers. However, the big jump at shallower depths is apparently due to its tolerance of the large residuals. The hybrid solver gives result comparable to the IRLS, but it oscillates at deeper depths. The results from $L2$ and conjugate direction $L1$ solver are similar, but the smaller steps in the result of conjugate direction $L1$ (Figure 5(c)) are promising.

Figure 6 shows the 1-D field RMS velocity from the velocity scan. This field data has 1,000 sample points, and the number of blocks in the model space is unknown. In real life, we can never fully constrain a inversion problem without regularization: that is when Dix inversion becomes unstable. Therefore, only regularized solvers are tested.

Figure 7 shows the inversion results when field data in Figure 6 are fed into different regularized solvers. The results from the IRLS and the conjugate direction $L1$ solver have more blocky nature than the other two. The result from IRLS is more flat within layers, which is a nice property in well-log matching and many other geophysical applications. The hybrid and $L2$ solvers give comparable results, although we have chosen a very small σ for hybrid norm to force it towards the $L1$ norm.

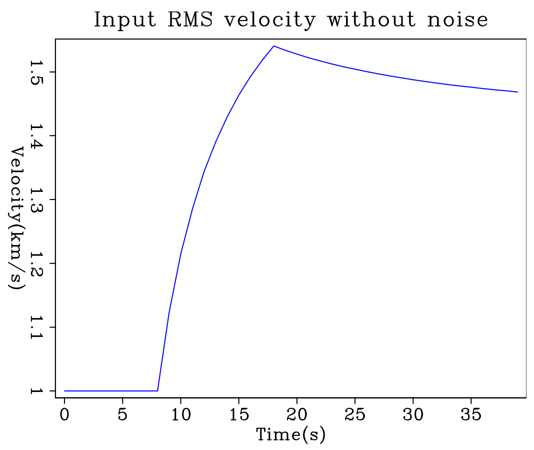
CONCLUSIONS AND DISCUSSIONS

We explore three methods to constrain Dix inversion in an $L1$ nature: an improved IRLS method, a hybrid $L1/L2$ method and conjugate direction $L1$ method. The IRLS and hybrid methods are implemented in a non-linear least-squares scheme by adding a diagonal weighting function. Conjugate direction $L1$ method is realized by a weighted median solver.

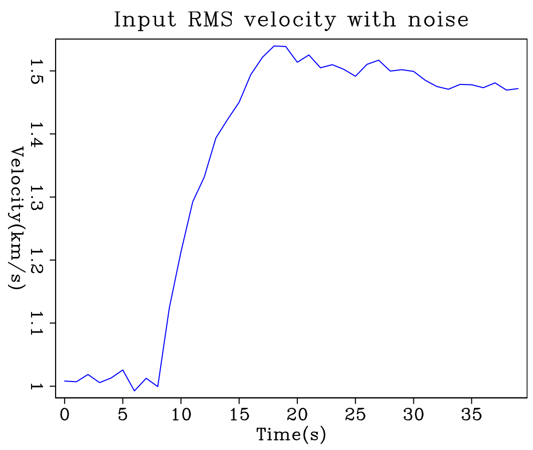
The IRLS method is improved by the physical explanation of the cutoff number σ, allowing this numerical parameter to be determined automatically. The hybrid method has a novel plane search scheme based on Taylor's series at each residual. Conjugate direction $L1$ method has an iterative plane search scheme using a weighted median solver. Both of hybrid and the conjugate direction $L1$ are designed to reduce major computational cost by expending more effort in finding a better next step.

In the numerical experiment, we find that the conjugate direction $L1$ method decreases the iteration number for the outer loop significantly. Hence, the value of spending more to find a better next step is proved. The same concept can be applied to the hybrid method as well. We can expect better inversion results and faster convergence by adding iterations to plane search, which has not been demonstrated before.

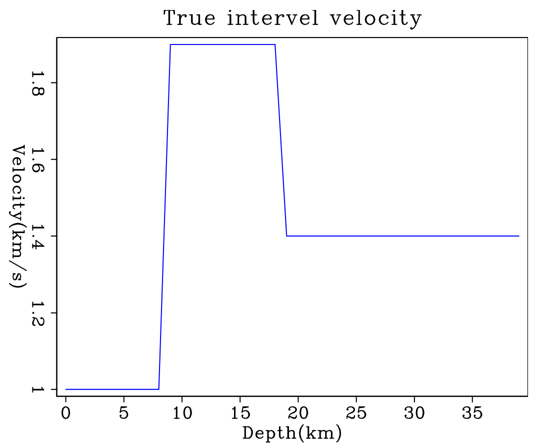
In the current study of the conjugate direction $L1$ method, we keep only two equations exactly satisfied in the whole system when searching the plane. In future research, we can add as many equations as needed by Gram Schmidt process. Hopefully, this process can lead us to an even better next step.



(a)

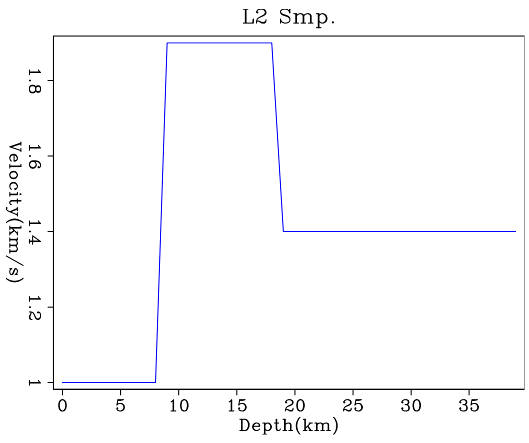


(b)

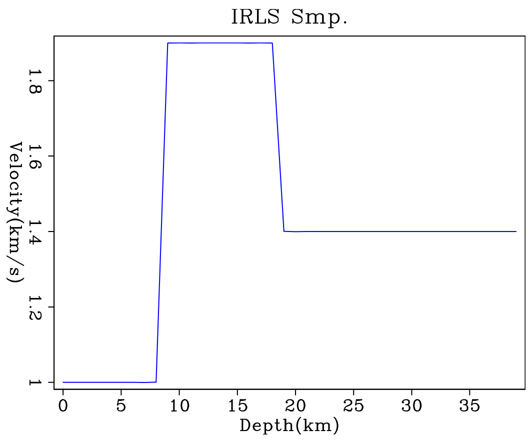


(c)

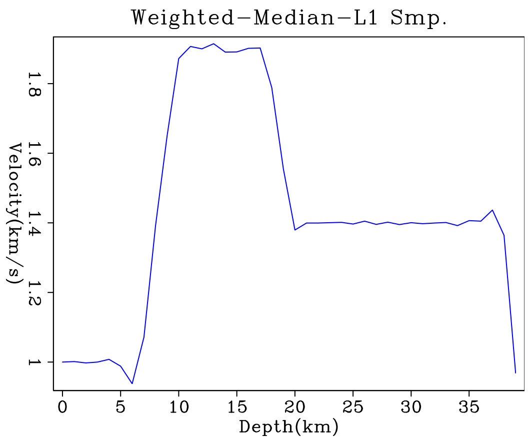
Figure 1: Input synthetic RMS velocity and true interval velocity. The two plots on the top row are the input RMS velocities **(a)** without noise and **(b)** with random noise, respectively. The plot on the bottom is **(c)** the true interval velocity which is true model in the estimation problem. [**ER**] elita1/. rmsvel,rms-noise,intervel



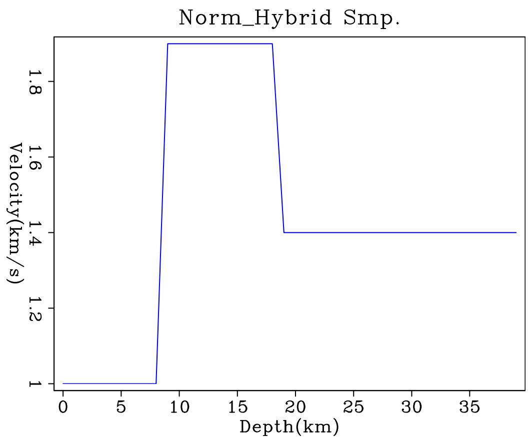
(a)



(b)

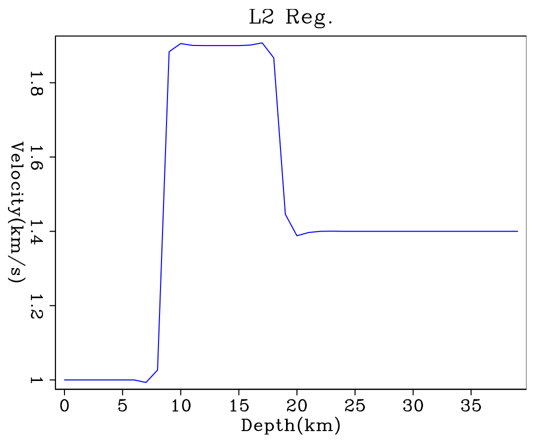


(c)

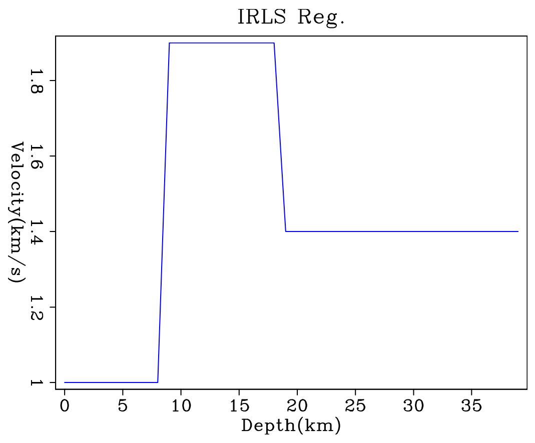


(d)

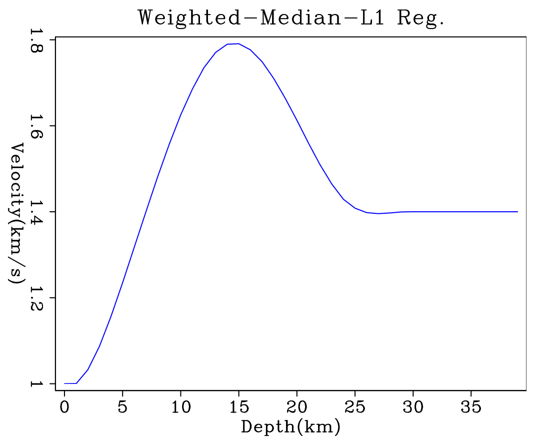
Figure 2: Inversion results of simple **(a)** $L2$ solver; **(b)** IRLS solver; **(c)** conjugate direction $L1$ solver and **(d)** Hybrid solver when clean data are fed in. All the regressions are without regularization. [**ER**] **elita1/. l21,irls1,wmed1,nrm3**



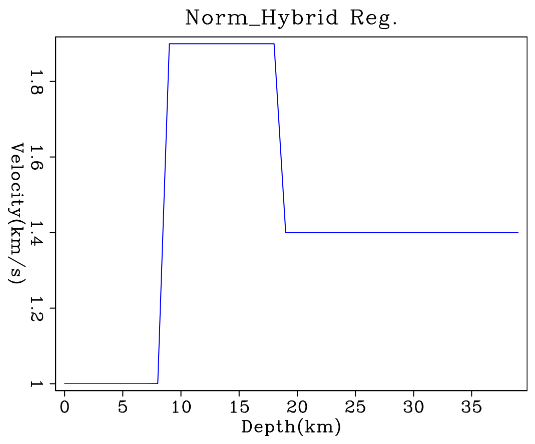
(a)



(b)

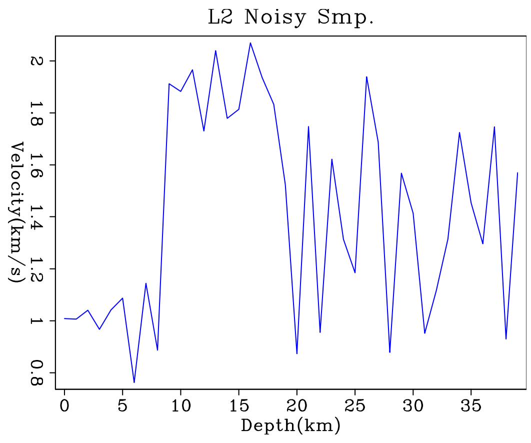


(c)

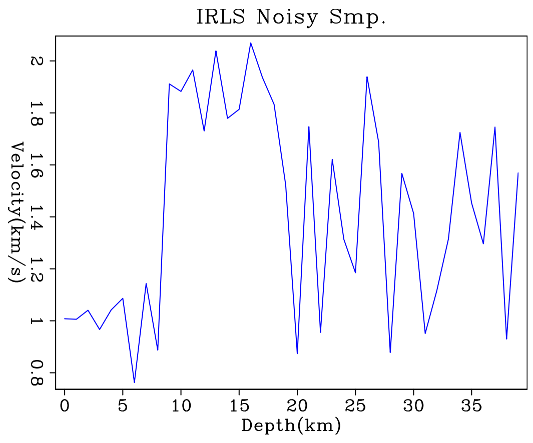


(d)

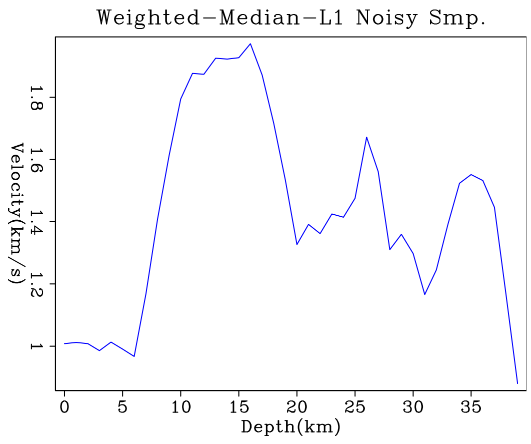
Figure 3: Inversion results of **(a)** $L2$ solver; **(b)** IRLS solver; **(c)** conjugate direction $L1$ solver and **(d)** Hybrid solver when clean data are fed in. All the regressions are regularized.[**ER**] elita1/. l22,irls2,wmed2,nrm2



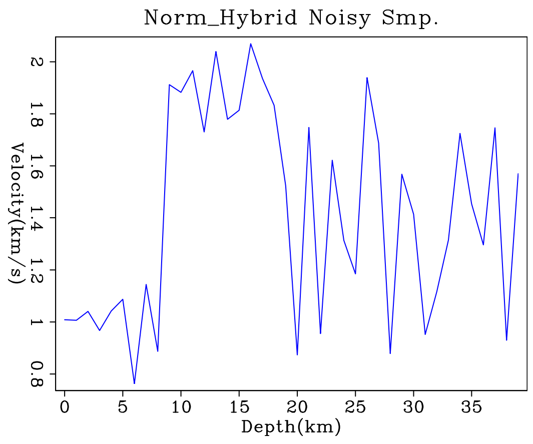
(a)



(b)

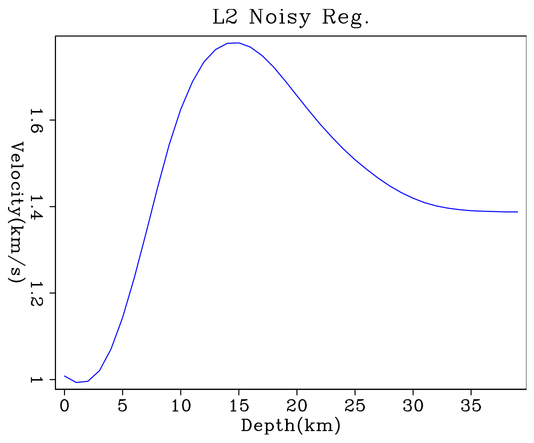


(c)

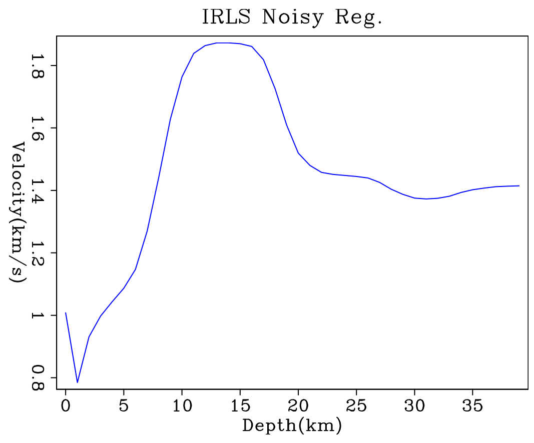


(d)

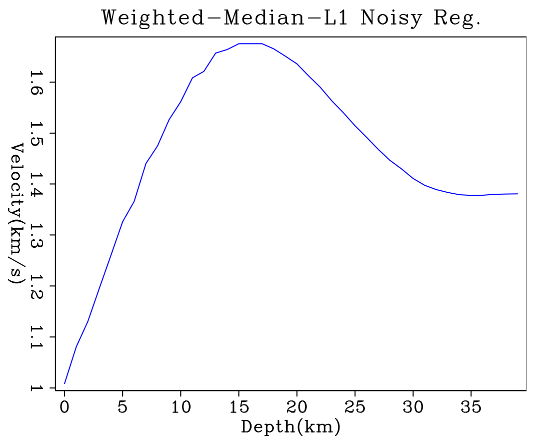
Figure 4: Inversion results of simple **(a)** $L2$ solver; **(b)** IRLS solver; **(c)** conjugate direction $L1$ solver and **(d)** Hybrid solver when noisy data are fed in. All the regressions are without regularization.[**ER**] $\boxed{\text{elita1/. l23,irls3,wmed4,nrm5}}$



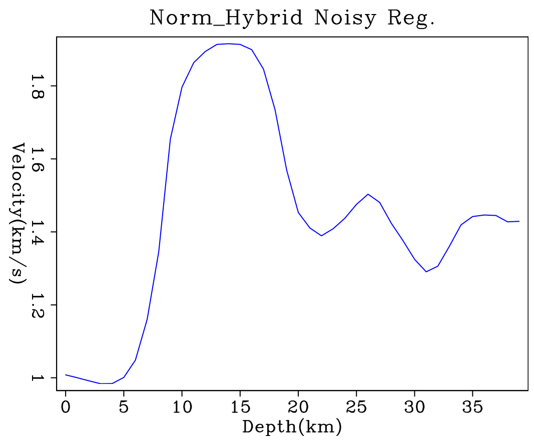
(a)



(b)



(c)



(d)

Figure 5: Inversion results of **(a)** $L2$ solver; **(b)** IRLS solver; **(c)** conjugate direction $L1$ solver and **(d)** Hybrid solver when noisy data are fed in. All the regressions are regularized. **[ER]** elita1/. l24,irls4,wmed5,nrm6

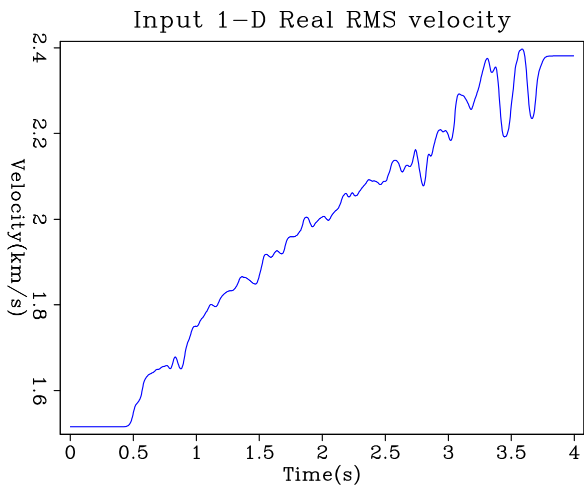


Figure 6: Input 1-D field RMS velocity data from velocity scan. [**ER**] elita1/. realrms

ACKNOWLEDGMENTS

I would like to thank Jon Claerbout for the idea of expending the effort in the inner loop to reduce the necessary iteration numbers of the outer loop. I would also like to acknowledge Robert Clapp for the code of conjugate direction $L1$ solver.

REFERENCES

Bube, K. P. and R. T. Langan, 1997, Hybrid l1/l2 minimization with applications to tomography: Geophysics, **62**, 1183–1195.

Claerbout, J. F., 2008, Image estimation by example.

———, 2009, Blocky models via the l1/l2 hybrid norm: SEP-Report, **139**, 1–10.

Claerbout, J. F. and F. Muir, 1973, Robust modeling with erratic data: Geophysics, **18**, 826–844.

Clapp, R. G., 2001, Geologically constrained migration velocity analysis: PhD thesis, Stanford University.

Darche, G., 1989, Iterative l_1 deconvolution: SEP-Report, **61**, 281–302.

Dix, C. H., 1952, Seismic prospecting for oil.

Guitton, A., 2005, Multidimensional seismic noise attenuation: PhD thesis, Stanford University.

Harlan, W. S., 1999, Constrained dix inversion.

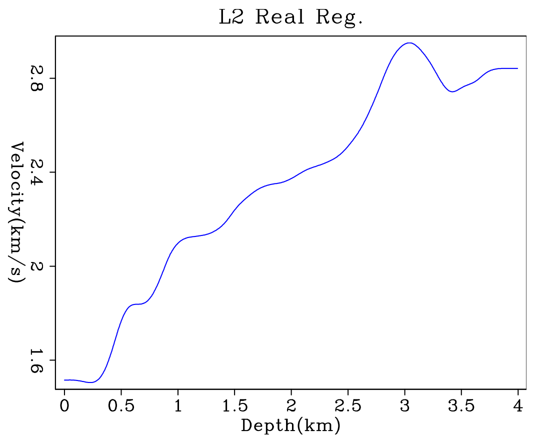
Koren, Z. and I. Ravve, 2006, Constrained dix inversion: Geophysics, **71**.

Maysami, M. and N. Mussa, 2009, Generalized-norm conjugate direction solver: SEP-Report, **139**, 11–22.

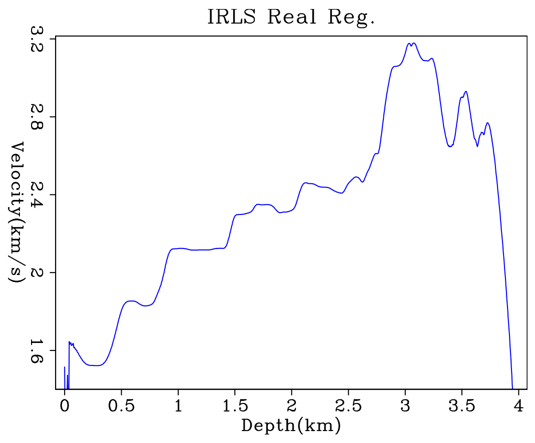
Nichols, D., 1994, Velocity-stack inversion using l^p norms: SEP-Report, **94**, 1–16.

Tang, Y., 2006, Least-squares migration of incomplete data sets with regularization in the subsurface-offset domain: SEP-Report, **125**.

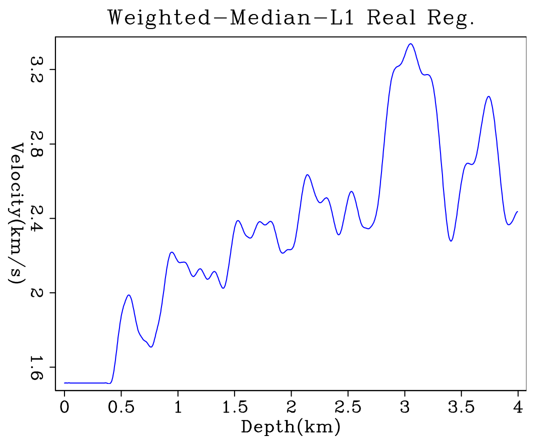
Valenciano, A. A., M. Brown, M. D. Sacchi, and A. Guitton, 2003, Interval velocity estimation using edge-preserving regularization: SEP-Report, **114**, 136–150.



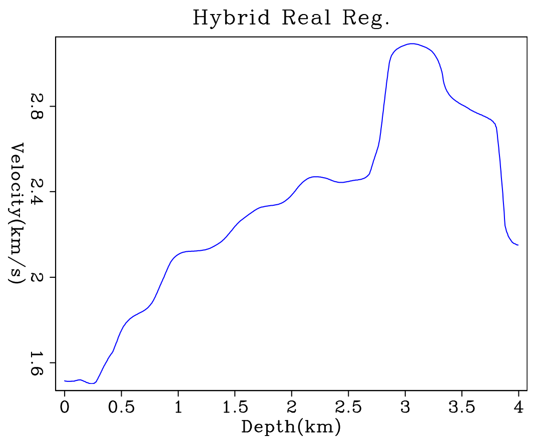
(a)



(b)



(c)



(d)

Figure 7: Inversion results of **(a)** $L2$ solver; **(b)** IRLS solver; **(c)** conjugate direction $L1$ solver and **(d)** Hybrid solver when 1-D field RMS velocity data are fed in. All the regressions are regularized. [**ER**] elita1/. real2,real1,real4,real6

Stanford Exploration Project, Report SEP139, October 19, 2009

Applications of the generalized norm solver

Mandy Wong, Nader Moussa, and Mohammad Maysami

ABSTRACT

The application of a L1/L2 regression solver, termed the generalized norm solver, to two test cases, shows that it is potentially an efficient method for L1 inversion and is easy to parameterize. The generalized norm solver iterates with conjugate direction. Our first test case, the line fitting problem, shows that the generalized solver is capable of removing outliers in data. Our second test case, the 1D Galilee problem, shows that the generalized solver can produce a satisfactory "blocky" solution. In terms of parameters, a low threshold value, if giving convergent solution, gives the best result. Experience shows the optimal number of inner loop iterations is one.

INTRODUCTION

Currently, many geophysics problems are solved with least-squares (L2) model fitting because of its fast convergence, simple parametrization, and easy to understand numerical analysis. However, L2 minimization places disproportionate emphasis on large residual values. Therefore, an L1-type norm inversion technique is more appropiate for solving geophysical problems that have a "blocky" model space. One example is to do adaptive subtraction of multiples Guitton (2005) using IRLS (iterative re-weighted least squares). Other possible applications are tomography (Bube and Langan (1997)) and deconvolution of noisy data (Chapman and Barrodale 1983).

In geophysics, a popular way to run L1-type inversions is with IRLS (Gersztenkorn et al. (1986)) . Running with IRLS often improves the results. A drawback is that its computational time is considerably higher because of repeated application of the costly forward and adjoint data-fitting operator within two iterative loops. To overcome this computational deficiency while retaining the benefit of L1-type inversion, we came up with another solver that does a better job than IRLS. Claerbout (2009) developed an algorithm of that we call the **generalized norm solver**, which steps with conjugate-direction and using the Taylor series expansion. Such a solver allows us to perform inversion using the L2 or the mixed L1/L2 norm Maysami and Moussa (2009). For convenience, we will use the term 'norm' to refers to all kind of measures and norms. It is understood that a norm has a strict definition in mathematics. For the theoretical description of the solver, please refer to the report by Claerbout (2009).

Maysami and Moussa (2009) have implemented such a solver, which allows us to test its robustness in this paper. Three norms will be used in our study: the least-squares (L2), Hybrid, and Huber norms. It is worth mentioning that the theory for using the least-squares norm with our solver is exactly the same as the theory for solving the least-squares problem with conventional conjugate-direction algorithm. For the rest of this paper, we will refer the generalized norm solver with the Hybrid norm as the **hybrid solver** and the generalized norm solver with the Huber norm as the **Huber solver**.

We have applied the generalized norm solver to two test cases. The first test case recovers the equation of a straight line given data that are corrupted with Gaussian noise and spikes. We find that the generalized norm solver recovers the equation of a straight line when using either the hybrid solver or the Huber solver. The second test case is called the 1-D Galilee problem. This problem is a simplified version of a real depth sounding experiment of the Sea of Galilee and a standard test problem in SEP textbooks, (Claerbout (2008); Claerbout and Fomel (2008)).The synthetic data for this case are measurements between the water surface and the bottom of the lake. The first part of the test simply aims to recover the true depth of a 1-D lake given that the data contain occasional large spikes. The second part of the test has data corresponding to a water level that change in a step-like manner. We will refer to these kind of jumps as "drift." We aim to recover the true depth and sudden drift in data. We find that the hybrid solver always gives the best result as compared to the Huber solver and the least-squares solver.

FIRST TEST CASE: LINEAR FITTING PROBLEM

Basic formulation for the linear fitting problem

To verify the L_1-norm solver and establish its utility across several geophysical optimization problems, we tested on a series of one-dimensional fitting problems. We began with the simplest fitting problem, a line-fit estimation that would be suitably solved by least-squares line fitting in most cases.

To show the advantages of an L_1-based methodology, we injected both Gaussian and non-Gaussian noise. In general, while L_2 fitters are well suited to wide-spectrum noise, they are particularly prone to misleading or unphysical results if the noise has many spike- or burst-like. We experimented with a linear fit of a set of data plus several spikes.

The problem setup of the line-fit is similar, but simpler, than the other examples. It serves as a good explanatory case-study of the set-up of a solver. We begin with a data space, $\mathbf{d}$, representing the noisy sampling of a straight line. We intend to model this line with a very simple, two-element line model – namely, slope and intercept according to the conventional $y = \alpha x + \beta$ formula.

We construct a forward operator, $\mathbf{L}$, which implements the mapping of this model space onto our recorded data. I n the presence of interfering noise, there will be a deviation between predicted data ($\mathbf{Lm}$) and recorded data ($\mathbf{d}$). We have tested our solver to minimize this deviation according to L_2, Huber, and hybrid L_1/L_2 definitions. This simplified example shows the important ability of the L_1-style norms to reject large data outliers.

In this case, we formally define the model space,

$$\mathbf{m} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}, \tag{1}$$

where α represents thes slope and β represents the y-intercept of a line. And the line operator,

$$\mathbf{L} = \begin{bmatrix} x & 1 \end{bmatrix}, \tag{2}$$

We attempt to minimize the objective functions for each defined norm:

$$\begin{aligned} r_{L_2} &= ||\mathbf{Lm} - \mathbf{d}||_{L_2}, \\ r_{L_1} &= ||\mathbf{Lm} - \mathbf{d}||_{L_1}, \\ r_{Huber} &= ||\mathbf{Lm} - \mathbf{d}||_{Huber}, \\ r_{Hybrid} &= ||\mathbf{Lm} - \mathbf{d}||_{Hybrid}, \end{aligned}$$

Because each norm represents a different method for computing the residual, the corresponding minimization has different behaviors with regard to the optimal modeled data. As we will show in the following sections, this impacts the ability of each optimization criterion to produce geophysically useful results in the presence of different types of interference, and for the different characteriestics of the desired function (e.g. sparseness or block-like intervals).

Results of the line-fitting problem

The results of solving this problem with the least-squares, hybrid, and Huber norms are shown in Figure 1. We can see that the fitted line in the L2 norm deviates from the true line due to the presence of spiked data, whereas for the Huber solver and the hybrid solver, the fitted line correctly overlaps the true line. We conclude that our trivial line-fitting example functions properly when using the L_1-type hybrid and Huber norms.

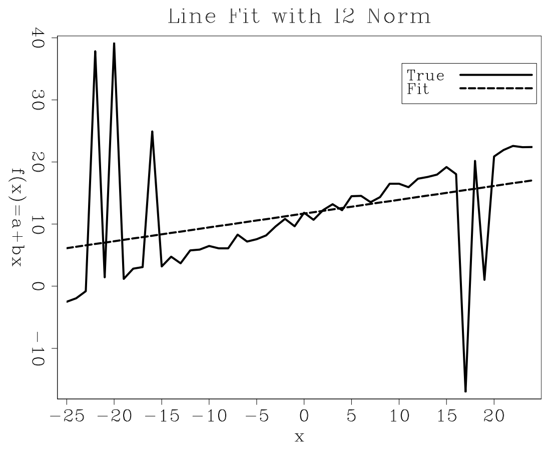
TEST CASE TWO: THE ONE DIMENSIONAL GALILEE PROBLEM

Formulation of the 1D Galilee Problem

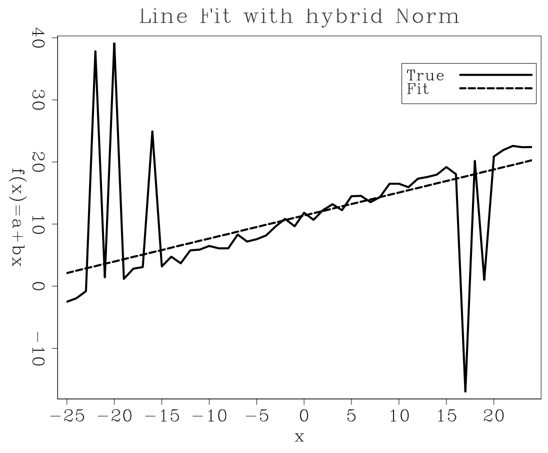
Our next test case is the removal of spikes and drift from a 1-dimensional representation of the Galilee depth data. We constructed this experiment to be more rigorous than the earlier line-fitting and noise-removal problem. The 1D Galilee problem is a synthetic problem that originates from a depth sounding experiment on the Sea of Galilee. Imagine performing a depth-sounding experiment along a fixed track in the lake. The lake has a sinusoidal depth with blocky drift and large spikes in time. As a boat goes back and forth across this 1D lake, it is measuring the sum of the true lake depth plus the drift in time. Figure 2 shows the true depth of our 1D Galilee lake and figure 3 shows the recorded data, which is the sum of the drift function and the true depth. We have added 2 spikes as outliers in the data. These 2 spikes can be viewed as equipment failure. Note that the data covers the lake back and forth roughly 6.25 times.

To formulate the problem for inversion, we have set our unknown model space to be the lake depth, $\mathbf{m}$, and the drift function, $\mathbf{u}$. Data space $\mathbf{d}$ is the recorded depth as shown in Figure 3. Our data fitting goal can be defined as

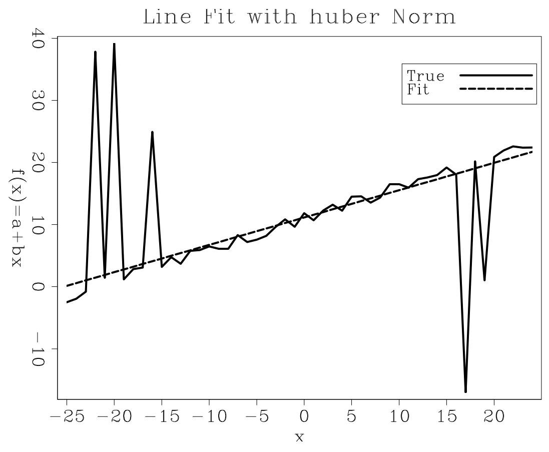
$$0 \approx \mathbf{Lm} + \mathbf{u} - \mathbf{d}, \tag{3}$$



(a)



(b)



(c)

Figure 1: Line fitting using the generalized norm solver: **(a)**L_2 fitting, **(b)** hybrid norm fitting, **(c)** Huber norm fitting. Notice that the L_2 fit-line does not match the actual data trend – this illustrates the susceptibility of least-squares minimization to strong outliers (spikes), while the other norms are totally unaffected by these data points. **[ER]** mandy1/. fit-l2,fit-hybrid,fit-huber

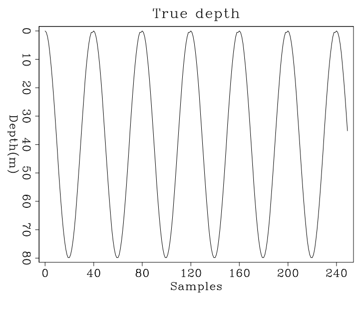
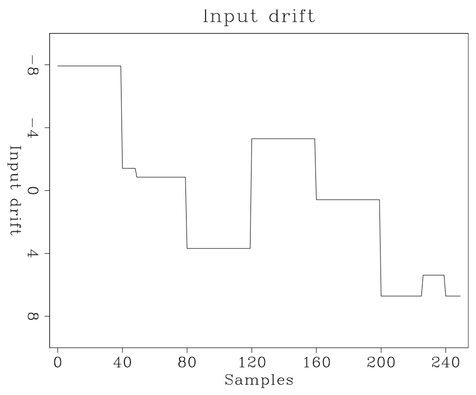
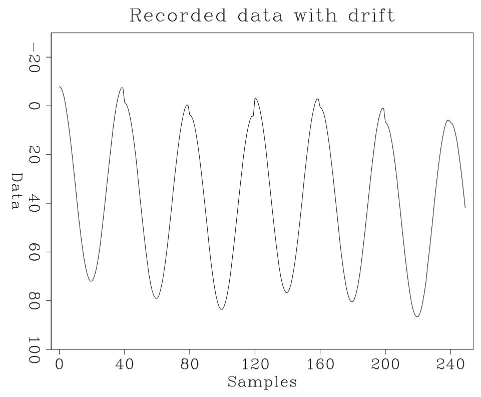


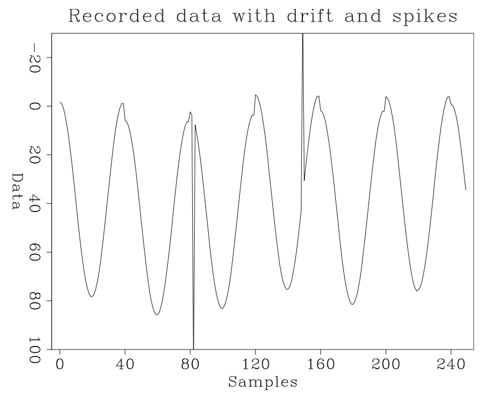
Figure 2: The true depth of the Sea of Galilee along a fixed track. **[ER]** mandy1/. truedepth



(a)



(b)



(c)

Figure 3: **(a)** The drift as a function of aquisition time. **(b)** The recorded data, which is the sum of the true lake depth and drift. **(c)** The recorded data with two outliers. The two spikes are added to the data to account for equipment failure.[**ER**] $\boxed{\text{mandy1/. data-drift,data-aq1,data-aq2}}$

where $\mathbf{L}$ is the binning operator that matches the data acqusition in time to its corresponding location in space. For a lake with 4 grid points and 6 data points, equation 3 would look like this:

$$0 \approx \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} m_1 \\ m_2 \\ m_3 \\ m_4 \end{bmatrix} + \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \\ u_6 \end{bmatrix} - \begin{bmatrix} d_1 \\ d_2 \\ d_3 \\ d_4 \\ d_5 \\ d_6 \end{bmatrix}.$$

Equation 3 by itself is an under-determined problem, because there are more unknowns than the recorded data points. If there are n_d data points and the lake has n_m grid points, then the model space has a dimension of $n_m + n_d$, because we are solving for both lake depth and drift in time. The data space has a dimension of n_d. To introduce more constraints, we can add a regularization by requiring the drift function $\mathbf{u}$ to be smooth,

$$0 \approx \frac{d}{dt}\mathbf{u}. \tag{4}$$

It is worth pointing we only expect good results when we run this regularization with $L1$ or $L1$-type norms, as $L2$ smoothing will wipe out the "blockiness" in the drift function, which is part of the model space. To illustrate the limitation of least-squares fitting, I will first show the result of applying inversion to the 1D Galilee problem.

Result of least-squares inversion of the 1D Galilee Problem

Least-squares inversion looks a for solution in the model space that minimizes the square of the residual. I first run un-regularized inversion on spike-free but drifted data. That means using the data-fitting goal in equation 3 to fit the data shown from Figure 3 (b). After L2 fitting, the estimated depth is shown in figure 4 (a), and the estimated drift is shown in figure 4 (b). An interesting observation from the un-regularized L2 inversion of the non-spike data is that it gives very good estimate of the lake depth and drift. Although the problem is still under-determined, the amount of data collected is sufficient enough to determine the relative jumps from each pass in the lake. As mentioned before, the data cover the lake back and forth roughly 6.25 times.

At this point, further attempts to solve this problem seem redundant, as we are getting a nearly perfect result. However, the result is different when I re-run the un-regularized inversion (equation 3) on the spiked and drifted data (Figure 3 (c)). After L2 fitting, the estimated depth is shown in figure 4 (c),and while the estimated drift is shown in Figure 4 (d). This time, the fitted depth deviates from the true depth, with segments that are clearly affected by spikes. The fitted drift function shows erratic jumps. This is because by minimizing the square of the residual, L2 inversion emphasizes large spikes in data. In a situation like this, L1 or L1-type inversion like the hybrid and Huber solvers should give better results. This is because minimizing the absolute value of the residual puts less emphasis on large spikes in data. This assertion will be verified when I apply the hybrid and the Huber solvers in the next section.

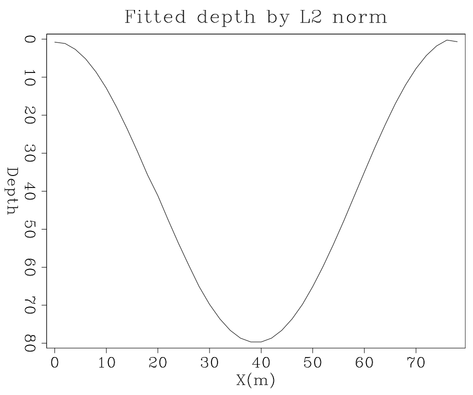
In addition to the two un-regularized least-squares examples above, I also ran a regularized inversion on the spike-free but drifted data. That means using the fitting goals in equation 3 and equation 4 on the data shown in Figure 3 (b). After L2 fitting, the estimated depth is shown in Figure 4 (e), and the estimated drift is shown in Figure 4 (f). The regularized drift in this case, shown in Figure 4 (f) , is a smoothed version of the un-regularized drift as shown in Figure 4 (b). The smoothing is as expected because of the type of regularization used. One conclusion is that when the data is spike-free, the L1-type solver is unnecessary, because Figure 4 (b) shows that we are obtaining a satisfactory result with L2 unregularized fitting. However, Figure 4 (d) demonstrates that L1-type inversion is needed for data containing large spikes. The next step is to see how the generalized L1 solver handle this problem using the hybrid and the Huber norms.

Result of the 1D Galilee Problem using the Generalized norm solver

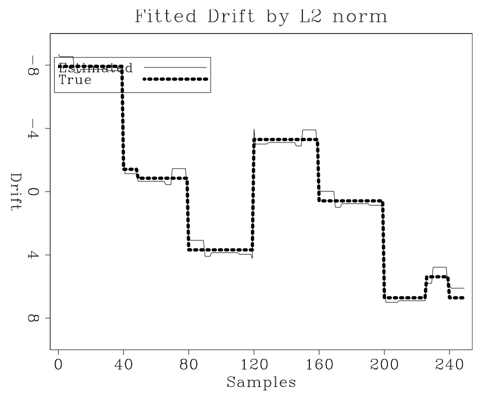
We begin with the simplest test for any L1-type solver, which is the ability to remove outliers in the data. When there is no water-level drift in the Galilee sounding experiment, the data is affected only by random spikes (non-Gaussian noise). The ideal fitted output would be to recover the true depth. The result is shown in figure 5. When we apply the generalized solver to the problem, L2 gives the worst result, as expected, because it cannot isolate outliers from the overall fitting goal. The Huber criteria gives an intermediate result, while the hybrid minimization criteria gives the best result, almost completely recovering the true depth of the lake without distortion.

Next, I apply the solver to the full 1-D Galilee problem, which is the inversion using fitting goals from equation 3 and 4 of data that including drifts and spikes, as shown in Figure 3 (b). The hybrid norm gives the most satisfying result, as summarized in Figure 6. There are two parameters that can be adjusted in this problem: `epsilon`, which describes the level of regularization, and `percentile`, which describes the transition point between the $L2$ and the $L1$ measure (per the defining equations of each norm). Please refer to Claerbout (2009) and Maysami and Moussa (2009) for the definition of each norm. In general, a small `epsilon` means less weight is placed on the regularization goal, meaning a less smooth result. The `percentile` parameter allows us to configure the degree of confidence that any randomly-selected data element is a statistical outlier, placing it within either the $L1$ or the $L2$ fitting goal. If a low percentile is set, it indicates that we believe most of the data should be fitted with L_1. For this problem, we have set the percentile to be as low as possible without having a divergent solution. For the Huber solver, the limit is $percentile \approx 0.3$. We found that the hybrid solver has a better tolerance, and the limit for it is $percentile \approx 0.1$.

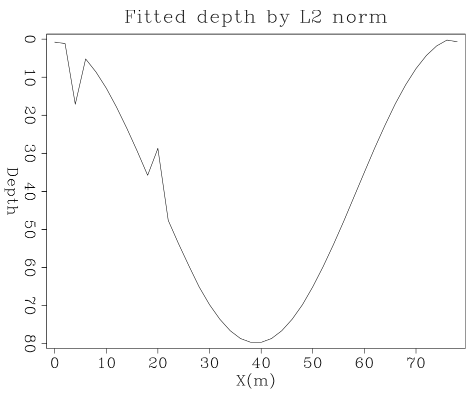
In terms of convergence, there are two iterative parameter. The first one is `niter`, which controls then number of applications of the costly forward and adjoint operators. Another one is `psiter` which corresponds to the number of iterations of the inner loop that determines the step sizes, α and β, in our conjugate direction scheme. We found that when $psiter = 1$, we obtain the best result in the full 1-D Galilee problem. For the outer loop parameter, `niter`, the $L2$ solver converges in just 16 iterations, the Huber solver converges in 36 iterations, and the hybrid solver converges in 66 iterations. Repeated experience



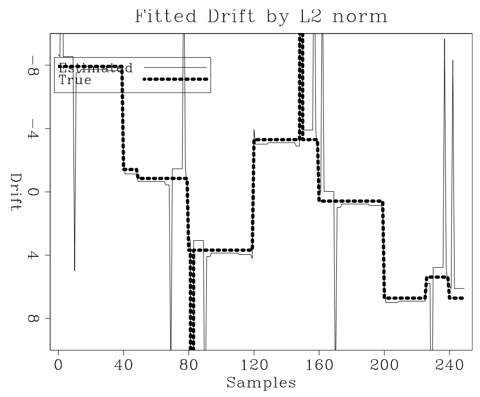
(a)



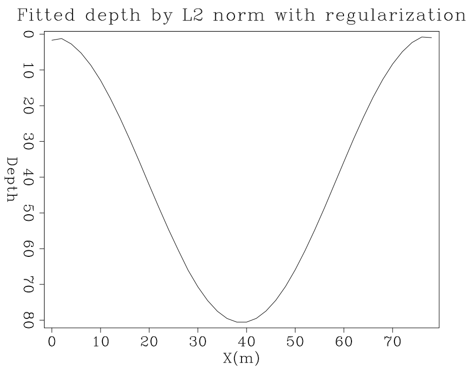
(b)



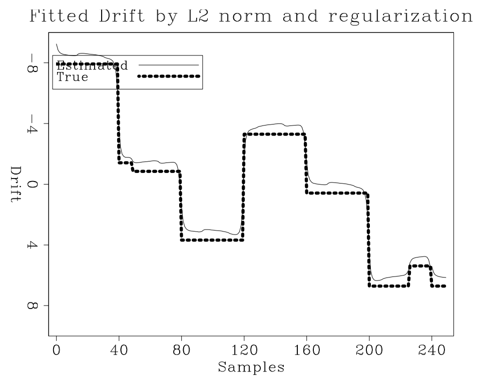
(c)



(d)

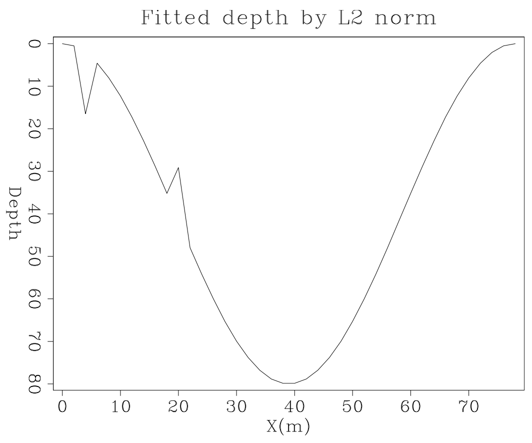


(e)

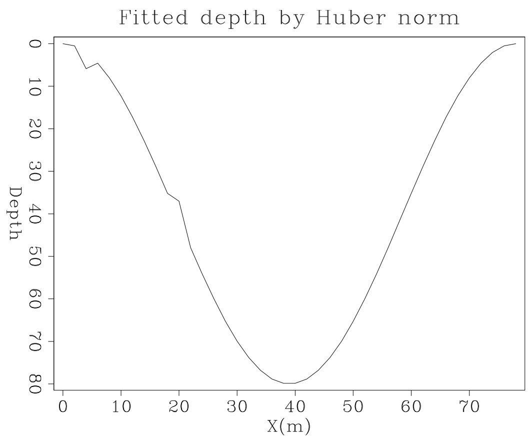


(f)

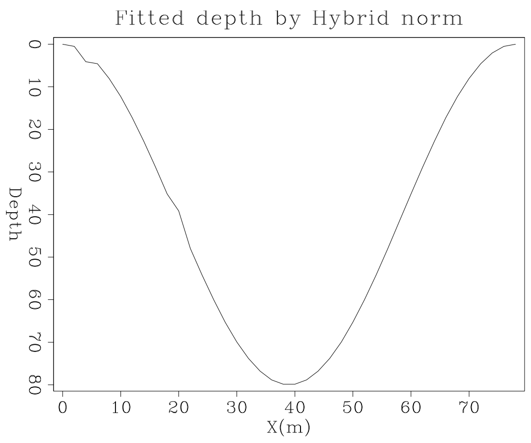
Figure 4: **(a,b)**: L2 inversion without regularizaton using the spike-free data. **(c,d)**: L2 inversion without regularization using the spiked data. **(e,f)**: L2 inversion with regularization using the spike-free data. **[ER]** mandy1/. l2dpt1,l2drf1,l2dpt0,l2drf0,l2dpt2,l2drf2



(a)



(b)



(c)

Figure 5: Fitting depth for measurements with spiky noise with the generalized norm solver using **(a)** L2 norm. **(b)** Huber norm with $eps = 0.1$ and $percentile = 0.375$ **(c)** Hybrid norm with $eps = 0.1$ and $percentile = 0.275$. **[ER]** $\boxed{\text{mandy1/. spike-l2,spike-huber,spike-hybrid}}$

indicates that we can always set *psiter* to one when using the generalized norm solver.

Comments on the result

From observating the fitted drift between figure 6 (b) and 6 (d) , the Huber solver is not doing significantly better than the least-squares solver. One possible explanation is that the underlying Taylor series assumption failed while trying to solve for the stepping coeficient α and β. Recall that the formulas for the Huber norm are

$$\begin{aligned} C(r) &= \begin{cases} |r| - |r_t|/2 & |r/r_t| \geq 1 \\ r^2/2r_t & |r/r_t| < 1 \end{cases} \\ C'(r) &= \begin{cases} \text{sgn}(r/r_t) & |r/r_t| \geq 1 \\ r/r_t & |r/r_t| < 1 \end{cases} \\ C''(r) &= \begin{cases} 0 & |r/r_t| \geq 1 \\ 1/r_t & |r/r_t| < 1 \end{cases} \end{aligned} \tag{5}$$

Notice that the second derivative vanishes if the residual falls to the threshold value r_t. This could lead to failure of the Huber solver, because the second derivatives are used in the denonimator when solving for the step size α and β in the conjugate-direction scheme (Claerbout (2009)).

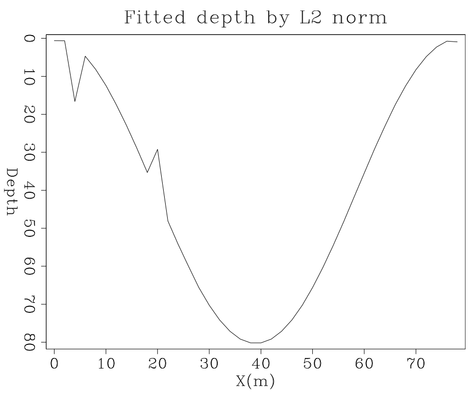
We are delighted to see that hybrid solver gives a resonable result for the full Galilee problem as shown in figure 5(c), we can hardly describe the model solution as "blocky." This might be because we have used a small threshold value for the model-fitting goal. For example, a threshold value of 0.30 percentile means we would like to see blocks about 3 to 4 points long. A higher threshold value for the model-fitting goal (equation 4) can increase blockiness; however our present solver has restricted us to use the same threshold value for both the model-fitting and the data-fitting goals (equation 3). One possible improvement for the future is to separate the thresholds for these goals.

CONCLUSION

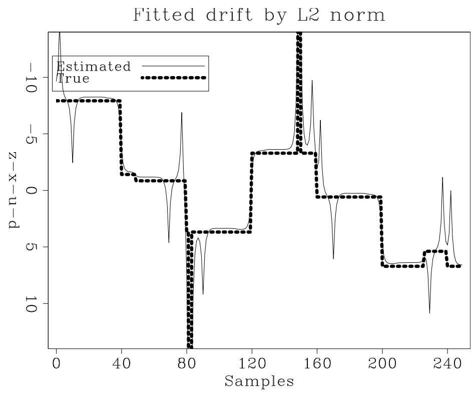
The applications of the generalized norm solver show promising results in our two sample problems. The line-fitting problem shows that our solver can correctly remove spikes and noise added to the data. The 1-D Galilee problem shows that the solver can properly produce a blocky model space while removing outliers. In terms of convergence, the hybrid solver takes longer to converge than the Huber solver. While only the one-dimensional problem is examined with this solver, we plan to further explore this solver with 2-D field data problem and directly compare the result with the IRLS algorithm.

ACKNOWLEDGMENTS

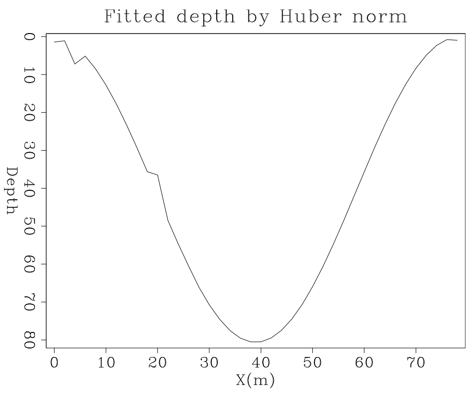
We thank Bob Clapp for providing guidance, ideas, and technical support in this project.



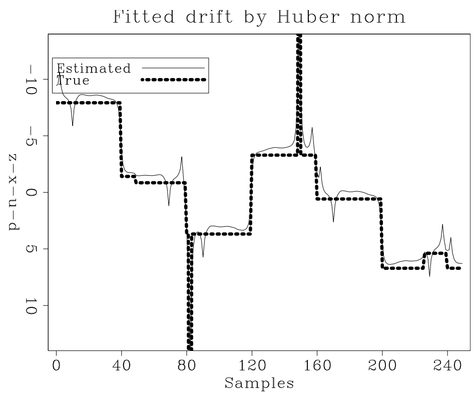
(a)



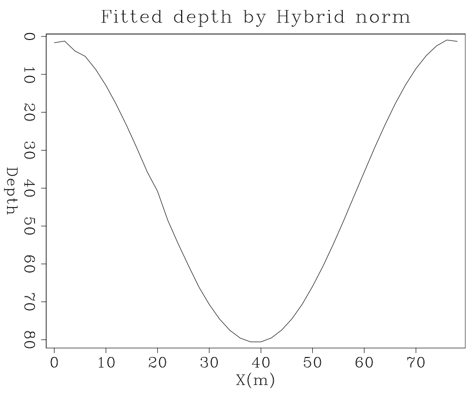
(b)



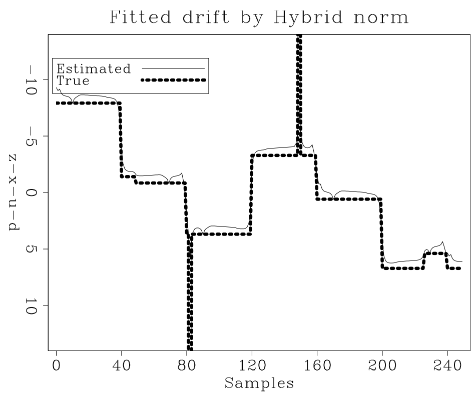
(c)



(d)



(e)



(f)

Figure 6: Fitting depth and drift for measurements with both drift and spiky noise with the generalized norm solver (regularized system) using the L2 norm **(a,b)**; the Huber norm with *eps* = 0.1 and *percentile* = 0.07 **(c,d)**; and the hybrid norm with *eps* = 0.1 and *percentile* = 0.32 **(e,f)**. **[ER]**
mandy1/. norm-dpt0,norm-drf0,norm-dpt1,norm-drf1,norm-dpt2,norm-drf2

REFERENCES

Bube, K. P. and R. T. Langan, 1997, Hybrid lambda1/lambda2 minimization with application to tomography: Geophysics, **62**, 1183–1195.

Claerbout, J., 2009, Blocky model via the l1/l2 hybrid norm: SEP report, **139**, 1–10.

Claerbout, J. F., ed., 2008, Image estimation by example: Geophysical sounding image construction.

Claerbout, J. F. and S. Fomel, eds., 2008, Basic earth imaging, 3.2 ed.

Gersztenkorn, A., J. B. Bednar, and L. R. Lines, 1986, Robust iterative inversion for the one-dimensional acoustic wave equation: Geophysics, **51**.

Guitton, A., 2005, Multidimensional seismic noise attenuation: PhD thesis, Stanford University.

Maysami, M. and N. Moussa, 2009, Generalized-norm conjugate direction solver: SEP report, **139**, 11–22.

Alternatives to conjugate direction optimization for sparse solutions to geophysical problems

Nader Moussa

ABSTRACT

Throughout much of this summer, we experimented with extensions to the conjugate direction method to find optimal solutions to sparse geophysical problems. However, this category of techniques is not unique in its ability to optimize L1-styled fitting goals. We also investigated a variety of other techniques, including a pure L1 solution via the weighted median; a steepest-descent algorithm using the signum-function as a gradient of the true L1 norm; and a totally different approach using the Simplex Algorithm, by mapping our objective function into a linear programming form. Categorically, the approaches that relied on the true L1 method failed due to what we believe is a theoretical shortcoming of the direct application of the pure L1 norm to geophysical optimization problems. The use of linear programming turned out to be quite successful. This could be an interesting option for future research in geophysical optimization.

INTRODUCTION

Part of our objective in this summer's study of the L_1 optimization criteria was motivated by new theoretical ideas for the conjugate direction solver (Claerbout, 2009), and its corresponding implementation (Maysami and Moussa, 2009). In addition to this new technique, we also extensively investigated the basic theory of convex optimization, motivated by our ultimate desire to find the most generally applicable toolkit for geophysical inversion on sparse or "blocky" models. Optimization theory has been subject to much research at Stanford across many fields. Prior art that is directly applicable to L_1 minimization spans the departments of Geophysics (Claerbout, 2008; Guitton, 2000), Computer Science (Golub and Van Loan, 1996), Operations Research, Management Science & Engineering (Paige and Saunders, 1982), and Electrical Engineering (Boyd and Vandenberghe, 2009). The enormous wealth of prior research across so many different disciplines has produced numerous algorithms and mathematical techniques which superficially bear no resemblence to each other – but all share the same final goal, which is the minimization of a generalized convex objective function. For the case of conventional geophysical inversion, this objective is some measure of the error between modeled- and recorded- data.

This broad-based investigation brought attention to techniques, such as linear programming, which are well-developed and widely used in other fields. However, these tools were rarely utilized by SEP (and presumably in the geophysical inversion community outside Stanford). Our efforts have developed a formulation of these techniques to convert a standard form geophysical data-fitting and inversion problem ($\mathbf{L}\ \mathbf{m} = \mathbf{d}$) into a linear programming problem.

I demonstrate in the following sections the efforts to construct a pure L_1 solver, and its associated numerical difficulties. Next is our foray into the realm of linear programming – a well-developed toolset that has seen little application in geophysical inversion.

PURE L1 SOLUTIONS

Early work focused on a pure conjugate-gradient or steepest-descent method using the true L_1 norm. We implemented a modified version of `cg_step`, implementing a line search first as a weighted median search, and also with a full, explicit calculation of the Frechet derivative. It was conclusively shown that the gradient of a pure L_1 solution resulted in introduction of local minima, resulting in a failure to converge. This has been noted in prior work (Bube and Langan, 1997) – the pure L_1 norm is not strictly convex. This can be shown with the trivial example of finding the L_1 minimum for a median problem with even number of elements:

$$\begin{bmatrix} 1 \\ 1 \end{bmatrix} \begin{bmatrix} m_1 \end{bmatrix} = \begin{bmatrix} 1 \\ 2 \end{bmatrix} \tag{1}$$

The residual in the L_1 case is the sum $|m - 1| + |m - 2|$, which has nonunique global minima for all $m \in \left[1, 2\right]$. The gradient, in this entire region, is exactly zero. In a larger, non-trivial data-fitting problem, these zero-value gradients can occur at locations other than the global optimum - and thus the gradient-based methods do not function properly.

LINEAR PROGRAMMING

Theory

The goal of Linear Programming, like optimization in general, is to maximize an objective function, subject to a set of constraints. In this case, the objective function is a linear equation in $\mathbf{x}$.

Linear programming developed out of the Operations Research community following World War II (Dantzig, 1963). It developed from earlier planning algorithms and optimization methods, finally emerging as one of the simplest tools that could optimally satisfy a set of competing equations. It is commonly used in business analytics, supply-chain management, and path planning. Variations of the concept have been applied to geophysical problems, including tomography inversion (Berryman, 1989); but in general, the geophysical inversion community has not shown strong adoption of this body of techniques. This is unfortunate, because the methodology of constructing and navigating within a feasible region of solutions enables both numerical optimization to find a global minimum, as well as heuristic "picking" interpretations of other valid, non-minimum solutions that satisfy the problem constraints.

We have successfully mapped general-purpose geophysical optimization problems into this numerical framework, particularly emphasizing data-fitting with linear operators. Due to its widespread use in other fields, a variety of software tools exist to find solutions to linear programming problems. A variety of numerical programming environments provide linear programming solvers natively; we explored several of these environments and evaluated their solver implementations. The most promising environment of the ones we considered is the GNU Scientific Library and the GNU Linear Programming Kit (GLPK), with language bindings for C, Python, and FORTRAN. Tools and libraries are also available for GNU Octave and its commercial equivalent, MATLAB.

The setup for linear programming revolves around a generalized parameter space, $\mathbf{x}$, which we seek to modify until an optimum is found with respect to some objective function $\mathbf{c}$ subject to constraints $\mathbf{b}$.

We have:

$$\begin{aligned} x_i &= \text{the parameter space} && (2)\\ c_i &= \text{the objective definition} && (3)\\ a_{ij} &= \text{the definition of the constraints} && (4)\\ b_j &= \text{the constraints vector} && (5) \end{aligned}$$

where i ranges from 1 to the number of parameters; and j ranges from 1 to the number of constraints.

Additional physical constraints are represented numerically by introducing *auxiliary variables*, denoted $\mathbf{x_s}$, and adding an objective coefficient c_i for each introduced constraint. Minimization of $|\mathbf{x_s}|$ is equivalent to optimal satisfaction of these physical constraints; this objective competes with the minimization of $|\mathbf{x}|$. This relationship between model- and data-fitting constraints is the basis of the optimzation problem.

General solutions to this optimization can be found according to the *Simplex Algorithm* (Dantzig, 1963), design of which is rooted in topological graph theory. To find an optimal solution, the system is represented in augmented matrix form. A scalar value Z is constructed, representing the objective function minus the constraints weightings, for any given intermediate solution. At each iteration, the simplex solver evaluates the maximal gradient of Z, with respect to a current set of basis equations, $\left[x|x_s\right]^T$.

This gradient provides a descent direction; the model x is updated accordingly, traversing the linear system until a vertex of the solution graph is found. At each vertex, a new basis set is calculated by matrix transvection (or equivalent, but slower, Gaussian elimination). This allows the objective scalar Z to be expressed in terms of the new basis $\left[x|x_s\right]^T$ – in other words, by rotating the state-matrix according to a transvection operator. (Transvection simply involves adding a scalar multiple of one row to another row). By choosing the correct transvections, the next descent direction can be directly read as the coefficients of the matrix representation. It can be shown that an optimum solution, if one exists, must lie on either a vertex or as a set of equally-optimum elements on a single edge of the solution space. This lies within the *feasible region* defined by the constraining equations.

Mapping Optimization Problems to Linear Programming

This maps to our conventional model-fitting treatment using optimization theory according to the following:

$$x_i \xrightarrow[\text{parameter space}]{} \text{model space} \tag{6}$$
$$c_i \xrightarrow[\text{objective function}]{} \text{regularization} \tag{7}$$
$$a_{ij} \xrightarrow[\text{constraint functions}]{} \text{forward operator} \tag{8}$$
$$b_j \xrightarrow[\text{constraint vector}]{} \text{data space} \tag{9}$$

with $i = 1..(\text{model size})$ and $j = 1..(\text{data size})$ Conveniently, this allows a direct representation of geophysical data fitting, including regularization on the model space. This linear programming setup can be represented in matrix form, as follows:

$$\mathbf{c}^T\ \mathbf{x} = Z \tag{10}$$
$$\mathbf{A}\mathbf{x} \leq \mathbf{b} \tag{11}$$

I have introduced the scalar, Z, which is the value of the objective. This will be either minimized or maximized depending on the particular geophysical problem. Effectively, this means finding a value for $\mathbf{x}$ that optimally aligns with the model-fitting and data-fitting goals.

Correspondingly, for a simple 2×3 example:

$$\begin{bmatrix} c_1 & c_2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = Z \tag{12}$$

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} \leq \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \tag{13}$$

To solve according to the L_1 norm, we need to take account of the absolute value in the definition of the L_1 error criteria, noting that the i^{th} element of the residual is defined as an absolute value of the error term:

$$r_i = |\mathbf{L}\ \mathbf{m} - \mathbf{d}| \tag{14}$$

To account for this, we extend the linear programming matrix representation to *augmented form.* This requires an extension of the $\mathbf{x}$ vector. The approach is not entirely dissimilar to the placement of a regularization in the model vector in a conventional setup, in that it is reformulating the equations to provide us with a fit in compliance with our *a priori* geophyiscal knowledge.

The augmented representation is written in matrix form as

$$\begin{bmatrix} \mathbf{1} & -\mathbf{c}^T & \mathbf{0} \\ \mathbf{0} & \mathbf{A} & \mathbf{I} \end{bmatrix} \begin{bmatrix} Z \\ \mathbf{x} \\ \mathbf{x_s} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \mathbf{b} \end{bmatrix} \tag{15}$$

The first row of the augmented form results in the optimization criteria:

$$Z - \mathbf{c}^T\mathbf{x} = 0 \tag{16}$$

A large value of Z maximizes the original objective function, subject to the constraints in the **c** matrix. The goal is to put as much "energy" in the Z (objective function) with as little energy in all other rows of the augmented matrix; this is accomplished with the *Simplex Algorithm*, simultaneously satisfying the minimization of the pure L_1 norm criteria.

L_1 Formulation

To satisfy the L_1 optimization criteria for a data fitting problem, with data **d** of length N_D, and model **m** of size N_M, modeled by $|\mathbf{Lm} - \mathbf{d}| = r$, we set up the following to specify the terms in (15) and (16):

$$\mathbf{A} = \begin{bmatrix} \mathbf{I}_{2N_D} & -\mathbf{L} \\ \mathbf{I}_{2N_D} & -\mathbf{L} \end{bmatrix} \tag{17}$$

$$\mathbf{b} = \begin{bmatrix} \mathbf{d} \\ -\mathbf{d} \end{bmatrix} \tag{18}$$

$$\mathbf{c}^T = \begin{bmatrix} 0 & \mathbf{1}_{2N_M} \end{bmatrix} \tag{19}$$

The initial model can be stored in **x**, which will be updated. The final value of auxiliary variables $\mathbf{x_s}$ represents the status of the model regularization constraints (which can also be assigned an initial value).

$$\mathbf{x} = \mathbf{m} \tag{20}$$

$$\mathbf{x_s} = \mathbf{m}_{regularization} \tag{21}$$

In this format, the matrices can be fed directly into the GLPK function call in C or Octave.

RESULTS

The linear programming formulation was among our most robust techniques for estimating and tracking water-level drift in our synthetic 1-D Sea of Galilee. Below are results plotted from a sample experiment. In this case, every single data point was corrupted by a water-level drift of unknown magnitude, and water-level was allowed to vary at any time during synthetic data collection. After ten sweeps, the linear programming solver is able to nearly perfectly reconstruct the drift profile and correctly estimates the true lake depth profile (except for minor artifacts).

This problem is an example of an underconstrained inversion problem. The source data is corrupted by an unknown data drift error function, which we seek to estimate based on our approximation that it should be defined by a sparse derivative. This model approximates a boat that is sampling lake-depth while floating on an unknown water level, illustrated in Figure 1. The problem is under-constrained, because we do not know the nature of the drift function; but by assuming that its derivative is sparse, an L_1 or linear programming optimization problem is set up. In Figure 2, the result of the linear programming drift estimation is shown.

I conclude that linear programming yields a fairly deterministic result, even in the case of an underconstrained inversion problem. The method is stable and straightforward. The most serious drawback is that most implementations of the Simplex Algorithm require an explicit definition of the forward operator (in matrix form) – so for very large geophysical problems, this can severely limit its applicability.

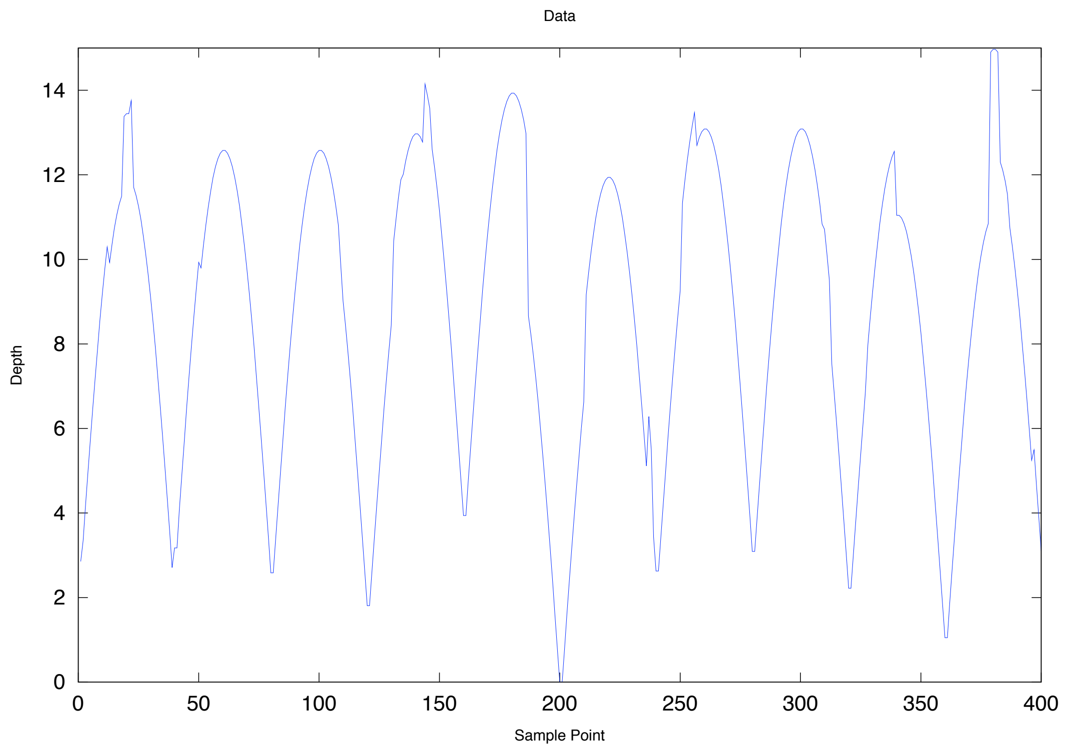


Figure 1: The input data for the linear programming Galilee estimation. **[CR]** nwmoussa1/. lprog-data

ACKNOWLEDGEMENTS

This work was spurned by Jon Claerbout's theoretical appreciation for alternate methods of optimization. Mohammad Maysami experimented with a Python linear programming

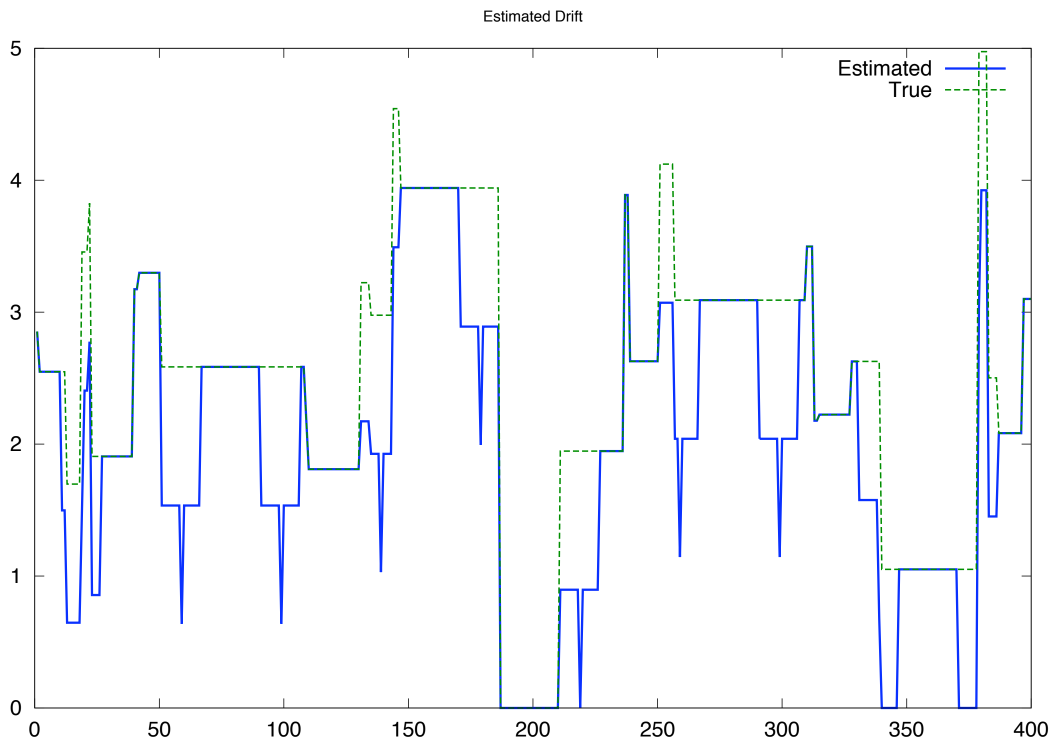


Figure 2: The estimated lake drift. **[CR]** $\boxed{\text{nwmoussa1/. lprog-estimated-drift}}$

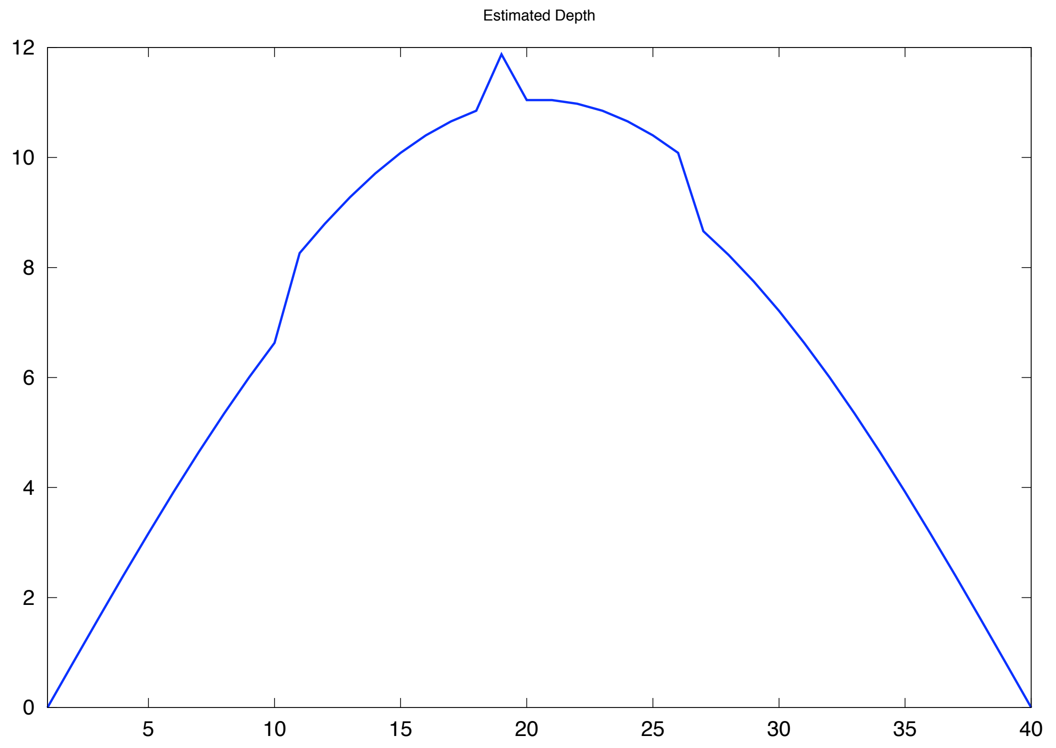


Figure 3: The estimated lake depth. **[CR]** $\boxed{\text{nwmoussa1/. lprog-estimated-depth}}$

code which successfully solved many test cases. The matrix mapping between Linear Programming and more conventional SEP-style data-fitting was largely developed with Elita Li. Bob Clapp's programming and technical assistance also expedited our progress.

REFERENCES

Berryman, J. G., 1989, Seismic crosshole tomography and nonlinear constrained optimization: Proceedings of the SIAM Geophysical Inversion Workshop, 396–414.

Boyd, S. and L. Vandenberghe, 2009, Convex optimization, seventh printing ed.

Bube, K. P. and R. T. Langan, 1997, Hybrid l^1/l^2 minimization with applications to tomography: Geophysics, **62**, 1183–1195.

Claerbout, J. F., 2008, Image estimation by example.

———, 2009, Blocky models via the $l1/l2$ hybrid norm: SEP-Report, **139**, 1–10.

Dantzig, G. B., 1963, Linear programming and extensions.

Golub, G. H. and C. F. Van Loan, 1996, Matrix computations, 3rd ed.: The Johns Hopkins University Press.

Guitton, A., 2000, Huber solver versus IRLS algorithm for quasi L1 inversion: SEP-Report, **103**, 255–270.

Maysami, M. and N. Moussa, 2009, Generalized-norm conjugate direction solver: SEP-Report, **139**, 11–22.

Paige, C. C. and M. A. Saunders, 1982, Algorithm 583 LSQR: spare linear equations and sparse least squares problems: ACM Transaction on Mathematical Software, **8**, 195–209.

Measuring velocity from zero-offset data by image focusing analysis

Biondo Biondi

ABSTRACT

Migration velocity can be estimated from zero-offset data by analyzing focusing and defocusing of residual-migrated images. The accuracy of these velocity estimates is limited by the inherent ambiguity between velocity and reflector curvature. However, velocity resolution improves when reflectors with different curvatures are present, as demonstrated by simple synthetic examples. The application of the proposed method to zero-offset field data recorded in the New York harbor yields a velocity function that is consistent with available geologic information and clearly improves the focusing of the reflectors.

INTRODUCTION

In Biondi (2009) I presented a method to extract quantitative velocity information by analyzing the focusing and defocusing of seismic images. This method is based on the *image-focusing* semblance functional that simultaneously measures image coherency along the structural-dip axes and the aperture-angle axes. I also discussed the ambiguity between reflector curvature and velocity, and how reflector curvature may bias velocity estimates from image focusing. The method I presented has two important characteristics. First, it explicitly takes into account the relation between reflector curvature and velocity. Second, it provides velocity information from image focusing that is consistent with the velocity information that we routinely extract from migrated images by analyzing their coherency along the data offset or the reflection-aperture angle axes.

In this paper, I apply the new method to velocity estimation from zero-offset data. Zero-offset data represent the extreme case where there is no velocity information coming from data redundancy with offset. In this case, velocity-estimation methods can only rely on velocity information contained in the focusing of the image. Therefore, the ambiguity between reflector curvature and migration velocity can severely limit velocity resolution.

Tests on simple synthetic data sets illustrate the curvature-velocity ambiguity, but also demonstrate that velocity resolution increases as the range of reflector curvature broadens. These results are corroborated by the application of the method to zero-offset data acquired by a shallow-seismic survey in the New York harbor. I iteratively update interval velocity in a sedimentary layer just below the water bottom. At each iteration, I estimate the value of residual-migration parameter that corresponds to the best focused image by evaluating the image-focusing semblance and picking its maximum. This value is then used to perform a conventional vertical interval-velocity update. The process converged after two iterations to an estimate of the sediment velocity that is consistent with available geologic information and improves the focusing of the migrated image.

IMAGE-FOCUSING VELOCITY ESTIMATION

In this section I briefly summarize the procedure I used to produce the results shown in the following two sections. The procedure is a simplification to zero-offset data of the method I presented in Biondi (2009).

The process starts from a partially-focused migrated image $\mathbf{R}(\mathbf{x})$, which is function of spatial coordinate vector $\mathbf{x} = \{z, x\}$, and continues with the following steps:

1. Perform residual migration on the initial image to produce an ensemble of residual-migrated images $\mathbf{R}(\mathbf{x}, \rho)$, where the parameter ρ is the ratio between the new migration velocity and the migration velocity used for the initial migration.

2. Estimate the local apparent structural dips in the residual-migrated images.

3. Dip-decompose the residual-migrated images, $\mathbf{R}(\mathbf{x}, \rho)$, to obtain the dip-decomposed images $\mathbf{R}(\mathbf{x}, \alpha, \rho)$, where α is the structural dip.

4. Perform the curvature correction (equation 2 in Biondi (2009)) of the dip-decomposed residual-migrated images, $\mathbf{R}(\mathbf{x}, \alpha, \rho)$, using the local-dip information extracted at step 2. The results of this process are the curvature-corrected images $\mathbf{R}_{\text{Curv}}(\mathbf{x}, \alpha, \rho, R)$, where R is the radius of curvature.

5. Compute image-focusing semblance as a function of ρ and R by applying the following equation,
$$S_\alpha(\mathbf{x}, \rho, R) = \frac{\left[\sum_\alpha \mathbf{R}_{\text{Curv}}(\mathbf{x}, \alpha, \rho, R)\right]^2}{N_\alpha \sum_\alpha \mathbf{R}_{\text{Curv}}(\mathbf{x}, \alpha, \rho, R)^2}, \tag{1}$$
where N_α is the number of dips included in the semblance computation.

6. Average the semblance computed using equation 1 over a spatial analysis window, after clipping out the smallest values of the semblance to remove noise and artifacts.

To perform the residual migration listed in step 1 of the procedure outlined above I used the linearized residual migration described in the Appendix of Biondi (2008). Other residual migration methods could be used, such as the one presented in Sava (2003). To simplify the analysis, I remapped the residual-migrated sections to pseudo-depth; that is, I remapped the depth axis of residual-migrated images according to the relationship $\tilde{z} = z/\rho$, where $\tilde{z}$ is pseudo-depth (Sava, 2004).

To estimate the local structural dips required by step 2, I used the Seplib program *Sdip* that implements a variant of the algorithms described by Fomel (2002). Any other local-dips estimator would be suitable. When performing the curvature correction at step 4, I define the curvature to be positive if the reflector frowns down (e.g. anticline) and negative if the reflector smiles up (e.g. syncline). The parameter N_α required for evaluating the focusing semblance at step 5 can be spatially varying according to the actual dip spectrum in the image. I kept it constant for my tests.

ZERO-OFFSET SYNTHETIC-DATA EXAMPLES

In this section, I present the application of the method outlined in the previous section to three zero-offset synthetic data sets. These data sets were modeled assuming reflectivity models of increasing complexity. The range of reflector curvature progressively increases from the first model to the third model.

Figure 1 shows the reflectors geometry used to model the three synthetic data sets. I modeled the first data set assuming a "cloud"of 46 point diffractors (panel a). For the second data set, I added eight curved reflectors with positive radius of curvature of approximately 55 meters (panel b). Finally, for the third data set, I added eight additional curved reflectors with a negative radius of curvature of approximately 55 meters (panel c). I set the maximum amplitude of the curved reflectors to be about 40% of the maximum amplitude of the point diffractors to maintain a balance between the velocity information provided by the point diffractors and that provided by the curved reflectors.

All the figures in this section follow the same pattern established in Figure 1. The left panels correspond to the reflectivity model shown in Figure 1a, the middle panels correspond to the reflectivity model shown in Figure 1b, and the right panels correspond to the reflectivity model shown in Figure 1c.

Figure 2 shows the three data sets modeled from the reflectivity models shown in Figure 1 assuming constant velocity equal to 2 km/s. The data increases in complexity and the texture changes as the curved reflectors are added to the reflectivity model.

Figure 3 shows the migrated sections obtained with the initial (too low) velocity of 1.951 km/s. The crossing of events in these images clearly indicates undermigration. The events corresponding to the reflectors with negative curvature are sufficiently undermigrated that they appear as reflectors with high positive curvature.

Figure 4 shows the results of the focusing analysis on the residual migrated ensembles obtained from the undermigrated images shown in Figure 3. All three panels show the image-focusing semblance spatially averaged in analysis windows defined by the following inequalities along the depth axis: 1.8 km $\leq$z$\leq$ 2.1 km, and by the following inequalities along the midpoint axis: 4.85 km $\leq$x$\leq$ 5.15 km. These analysis windows are represented in Figure 3 by the inner squares delimited by the grid superimposed onto the images. The panels show the average semblance as a function of the velocity parameter ρ and the radius of curvature R.

The semblance panels show diagonal trends for all cases because of the velocity/curvature ambiguity. When only point diffractors are present, only one trend is visible and the pattern is symmetric around ρ=1.025; that is, the correct value of the parameter. The addition of the positive-curvature reflectors adds another trend to the semblance panel (Figure 4b) and breaks downs the symmetry. When reflectors with both negative and positive curvature are present (Figure 4c), the semblance maxima occur around the correct value (ρ=1.025) for all the trends in the panel.

In poorly focused images corresponding to ρ values both lower and higher than the correct one, the increase in range of reflector curvatures causes additional crossing events. These crossing events interfere with the local dip estimation (step 2) and consequently with the curvature correction (step 4) Biondi (2009). As a result, the image-focusing semblance

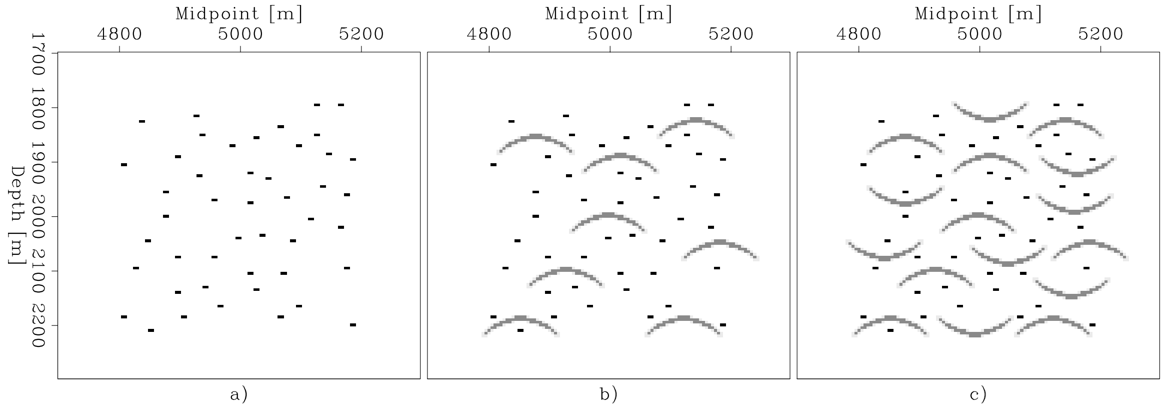


Figure 1: Reflectors geometry used to model the three zero-offset synthetic data sets I used to test the proposed image-focusing velocity-estimation method: (a) a "cloud" of point diffractors, (b) point diffractors and curved reflectors with positive curvature, (c) point diffractors, curved reflectors with positive curvature, and curved reflectors with negative curvature. [**ER**] biondo1/. Refl-all-overn

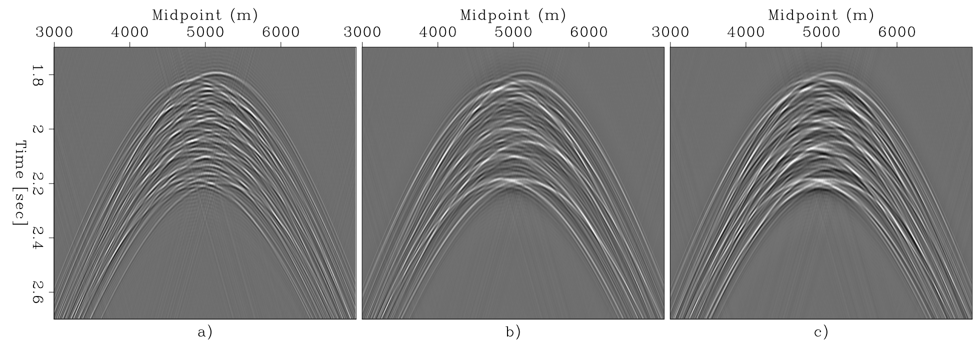


Figure 2: Zero-offset data modeled from the reflectivity functions shown in Figure 1. [**ER**] biondo1/. Data-all-overn

that measures dip coherency after the curvature correction is strongly attenuated for poorly focused images.

NEW YORK HARBOR DATA EXAMPLE

I tested the method for extracting velocity information from zero-offset data on a shallow seismic data set acquired in New York harbor using a 512i sub-bottom profiler with a 1-10 kHz pulse (Schock et al., 1989, 1994). Compared with conventional exploration data, these data contain much higher frequencies (useful signal is up to about 2,000 Hz) and have been acquired with correspondingly dense, though quite irregular, spatial sampling. For the sake of simplicity, I assumed that the acquisition grid was regular with spatial sampling of 0.225 meters; that is, the average of the actual sampling for the subset I analyzed. The actual standard deviation of the spatial sampling is about 0.018 meters; that is, about 8%.

Figure 5 shows the subset of the data that I worked with. The water bottom primary

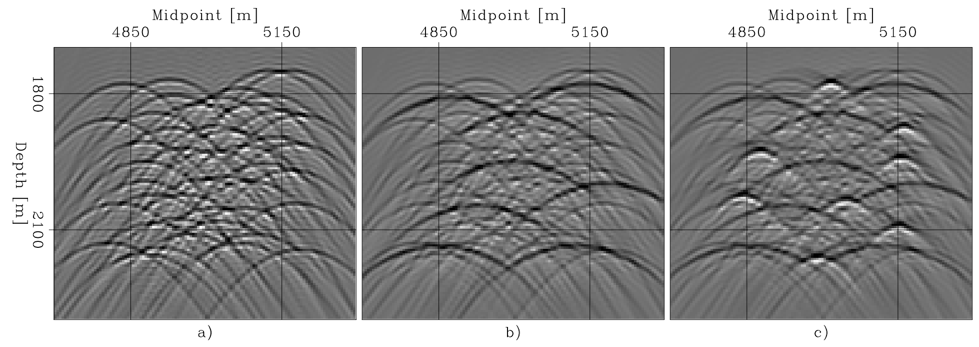


Figure 3: Migrated sections obtained by migrating the data shown in 2 with the initial (too low) velocity of 1.950 km/s. The inner squares delimited by the grid superimposed onto the images show the analysis windows, where the semblance is spatially averaged to produce the results shown in Figure 4. [**ER**] biondo1/. Mig-all-overn

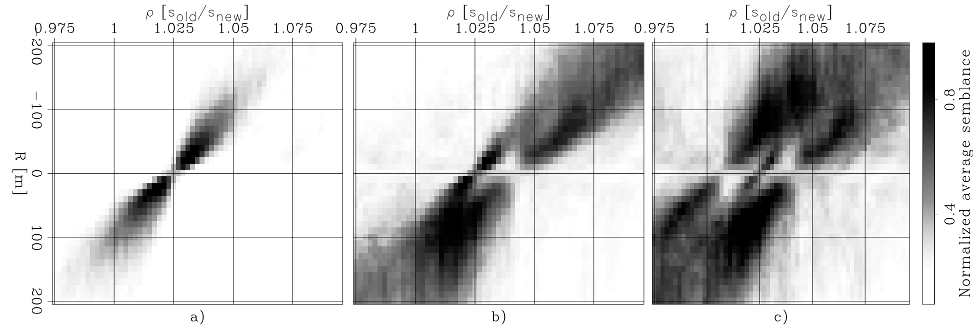


Figure 4: The image-focusing semblance spatially averaged over the analysis windows shown as a function of velocity parameter ρ and the radius of curvature R. Panel (a) shows the result corresponding to the point diffractors, panel (b) shows the result corresponding to the point diffractors and curved reflectors with positive curvature, and panel (c) shows the result corresponding to point diffractors, curved reflectors with positive curvature, and curved reflectors with negative curvature. [**CR**] biondo1/. Sembl-all-overn

reflection is recorded at about 18 milliseconds; strong multiple reflections are visible below the primaries. From nearby well-boring, the layered body in the middle is thought to be composed by Holocene sediments, mostly sands, with velocity of approximately 2.150 km/s. The sediments are surrounded by serpentinite that has much higher seismic velocity. The serpentinite velocity ranges from 2.500 km/s where the rock is fractured (on the left of the sediments) to 4.300 km/s, where the rock is intact (below the sediments).

I first migrated the data assuming a constant velocity equal to the velocity of water; that is, 1.500 km/s. Figure 6 shows the depth-migrated section. The diffraction-like events visible in the data just after the water-bottom reflection are properly focused in the migrated section. The deeper events show some sign of undermigration, but it is difficult to judge with certainty.

I performed the focusing analysis described earlier in the paper on a small analysis window; this window was centered on the events at flattish sediment-serpentinite interface at the bottom of the sediment layers. The depth of the analysis window ranged from 21 to 25 meters, for the first iteration. I adjusted the depth for the following iterations to ensure that the analysis windows consistently included the same reflectors across iterations. To avoid artifacts caused by the noisy traces clearly visible in the data around the midpoint location of 126 meters, I further limited the analysis window horizontally, to span only the midpoint interval between 87 to 126 meters.

Figure 7 shows the image-focusing semblance averaged in the analysis window as a function of the velocity parameter ρ and the radius of curvature R. There are two distinguishable trends in this semblance plot. The strongest trend corresponds to reflectors with positive radius of curvature between 2 and 5 meters. A much weaker trend corresponds to reflectors with curvature of similar magnitude, but negative. The semblance global maximum can be found at ρ=1.225 and R=4 meters. It is thus reasonable to assume that the majority of reflectors have positive curvature and a small fraction have negative curvature. Consequently, I used the value of ρ corresponding to the semblance global maximum to update the interval velocity for the sediments. Notice that if the curved reflectors from the strongest trend were assumed to be diffractors (R=0), the corresponding estimated residual-migration parameter ρ would be approximately 1.3, which is clearly too high.

To update the velocity in the sediments from the picked ρ value I followed conventional migration velocity analysis procedure for vertical velocity updating (Biondi, 2006). I assumed constant velocity in the sediment layer and estimated the updated velocity in the sediments $\widehat{V_s}$ by applying the following equation:

$$\widehat{V_s} = \sqrt{\frac{\left(\rho^2-1\right) V_w^2 \Delta t_w + \rho^2 V_s^2 \Delta t_s}{\Delta t_s}}, \tag{2}$$

where V_w is water velocity, V_s is the current estimate of sediment velocity, Δt_w is two-way traveltime in the water layer, and Δt_s is two-way traveltime in the sediments. At the first iteration V_s is set to water velocity. I picked the values of the traveltime intervals from the section displayed in Figure 5 and set Δt_w=.018 s and Δt_s=.012 s. Entering these values and ρ=1.225 into equation 2 results into $\widehat{V_s}$=2.250 km/s.

This velocity is slightly higher than the final one because of the limitation of the linearized residual migration I used. For large velocity errors, this residual migration undercorrects the image because it does not take into account ray bending. The error encountered in this case is larger than 20%. Consequently, the residual-migration parameter ρ is overestimated. However, another iteration of the velocity updating is sufficient to converge. My conjecture about the cause of the velocity-correction overshooting is supported by Figure 12, as discussed below.

The two panels in Figure 8 show respectively the result of the focusing analysis after migrating the data with the intermediate velocity function, V_s=2.250 km/s (panel a), and the final velocity function, V_s=2.155 km/s (panel b). This final velocity for the sediments was estimated using equation 2 with ρ=0.975; that is, the ρ value that corresponds to the maximum in Figure 8a. The semblance peak in Figure 8b occurs at ρ=1, indicating that the process has converged. The estimated value of 2.155 km/s for the sediment velocity is consistent with the geologic information available from well drilled near the location where the data were acquired.

Figure 9 summarizes the iterative velocity-estimation process by showing the velocities

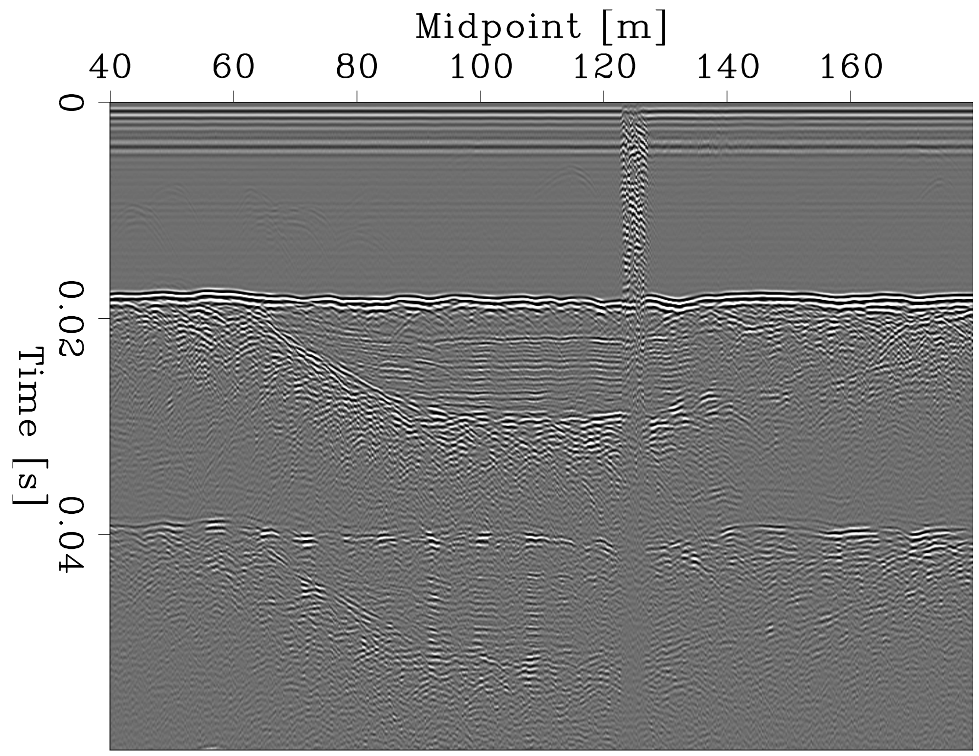


Figure 5: Zero-offset data recorded in New York harbor using a 512i sub-bottom profiler with a 1-10 kHz pulse. [**CR**] biondo1/. Data3-overn

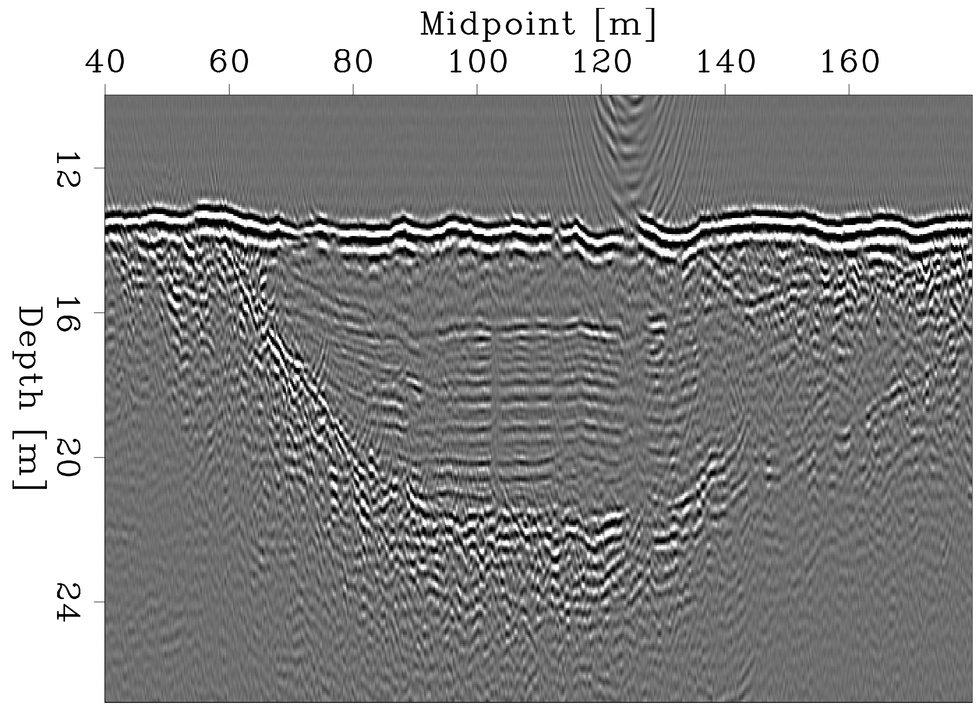


Figure 6: Depth-migrated section obtained assuming a constant velocity equal to the velocity of water; that is, 1.500 km/s. [**CR**] biondo1/. Mig-water-overn

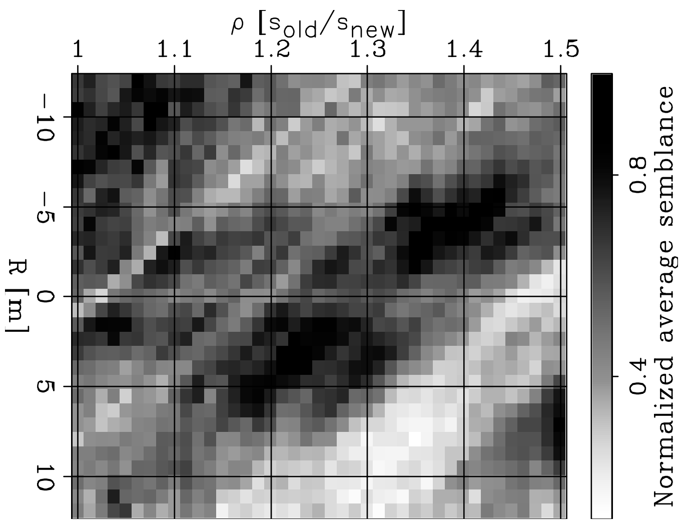


Figure 7: Image-focusing semblance computed from the initial migrated section and spatially averaged over the analysis window. Figure 6. [**CR**] biondo1/. Sembl3-overn

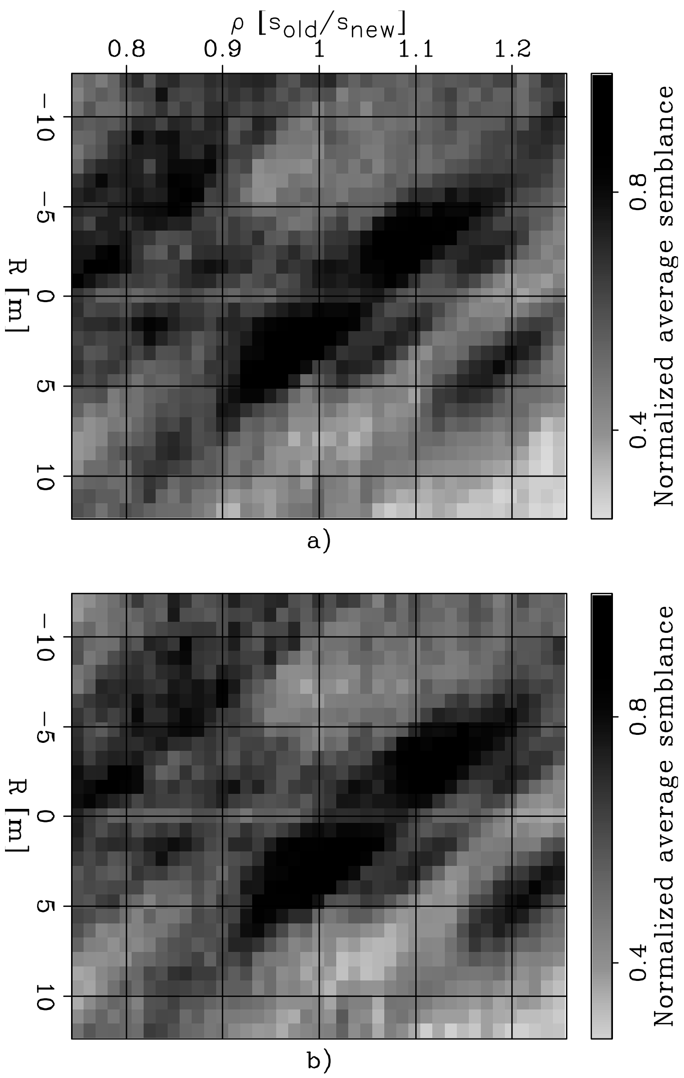


Figure 8: Image-focusing semblance computed from: (a) the intermediate migrated section, and (b) the final migrated section. [**CR**] biondo1/. Sembl-iter12-overn

as a function of depth at each iteration. The solid line (labeled as "Initial") represents the starting velocity, constant and equal to water velocity. The dashed line (labeled as "Intermediate") represents the result of the first correction, which slightly overshoots. The dotted line (labeled as "Final") represents the final velocity function.

Figure 10 compares the images obtained with the three velocities in the analysis window. Figure 10a shows a zoom into the image shown in Figure 6. Figure 10b shows a zoom into the migrated image obtained using the overcorrected velocity function, and Figure 10c shows the final image. Compared with the other two images, the final image shows substantial improvement in the continuity of the reflectors and a reduction in the amount of crossing events.

Figure 11 compares the entire image obtained with the initial velocity (top panel) with the entire image obtained with the final velocity (bottom panel). The improvement in the image are substantial also outside of the analysis window. The image for the dipping interface between the sediments and serpentinite at the left of the analysis window (between 60 and 90 meters along the midpoint axis) is more continuous in the final than in the initial image. The reflections just below this interface are also better focused in Figure 11b than in Figure 11a. Similarly, the reflector at the bottom of the sediment layer on the right of the analysis window (between 140 and 160 meters along the midpoint axis) is more continuous in the final image than in the initial one.

Finally, Figure 12 supports my previous conjecture that the overestimation of the velocity correction at the first iteration is caused by accuracy limitations of the linearized residual migration I employed when correcting for large velocity errors. This figure compares the analysis window for the final migrated image (top panel) with the analysis window taken from the residual migration of the first migrated image that corresponds to the picked value of ρ; that is, with $\rho = 1.225$ (bottom panel). These two images are very similar, demonstrating that the first ρ estimate is consistent with the final, and "optimal", velocity estimate. It leads to overshooting the interval-velocity correction because the residual migration is inaccurate. Notice that the differences in the vertical axis between the two sections are caused by the remapping to pseudo-depth of the residual-migrated section shown in Figure 12b.

CONCLUSIONS

Velocity information can be successfully extracted from zero-offset data by analyzing focusing and defocusing of residual-migrated images. The results shown in this paper demonstrate that the estimation method I presented in Biondi (2009) provides quantitative information on velocity without relying on subjective interpretative criteria.

The synthetic examples illustrate how the resolution and reliability of the velocity estimates increase when the target contains reflectors with a broad range of curvatures. This insight is useful to guide the application of the suggested method to practical problems.

The New York harbor example shows that the proposed method is sufficiently robust to provide reliable and quantitative velocity information from field data. The estimated interval velocity function is consistent with available geologic information and improves the focusing of the image both in the analysis window and outside the analysis window.

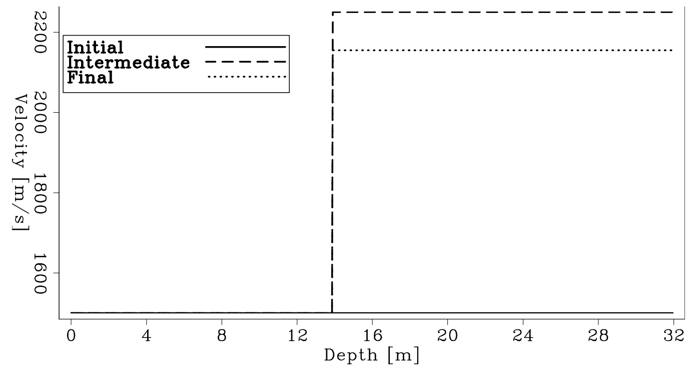


Figure 9: Velocity functions used for the three iterations of velocity-estimation process. [**ER**] biondo1/. Vel-overn

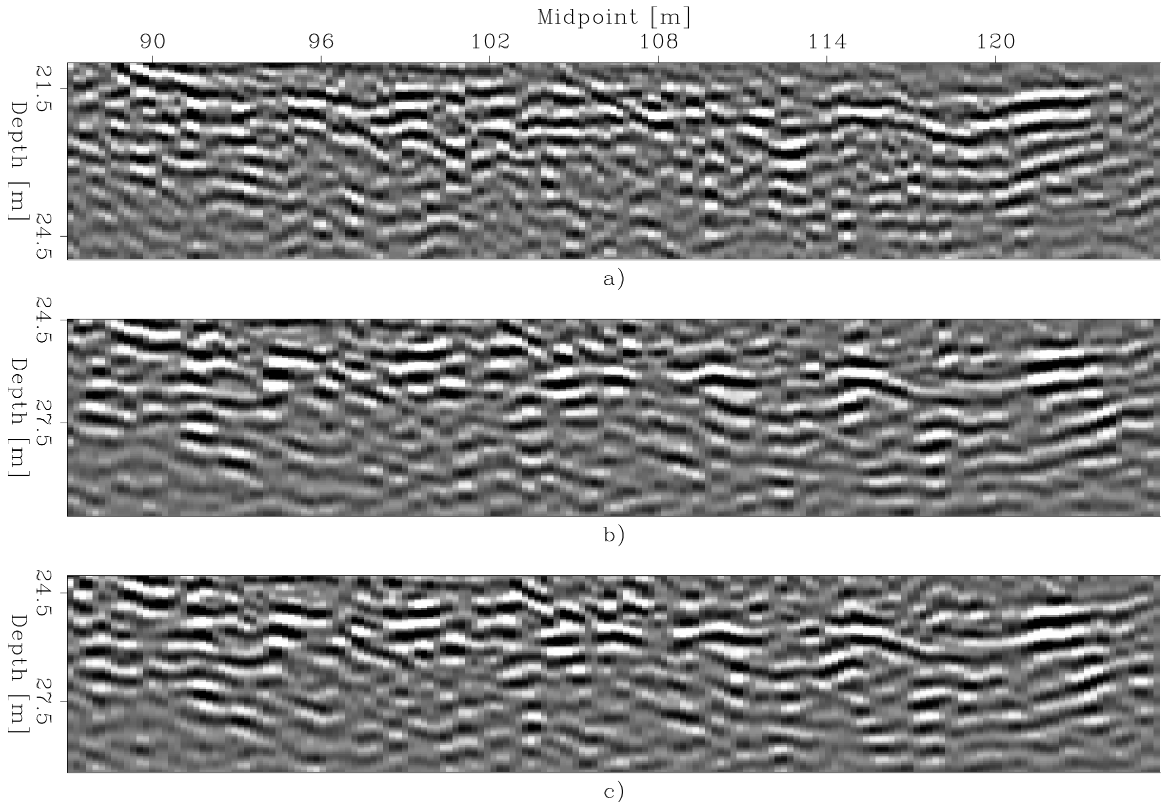


Figure 10: Migrated images in the analysis window for the three iterations: (a) initial velocity, (b) intermediate velocity, and (c) final velocity. [**CR**] biondo1/. Mig-win-overn

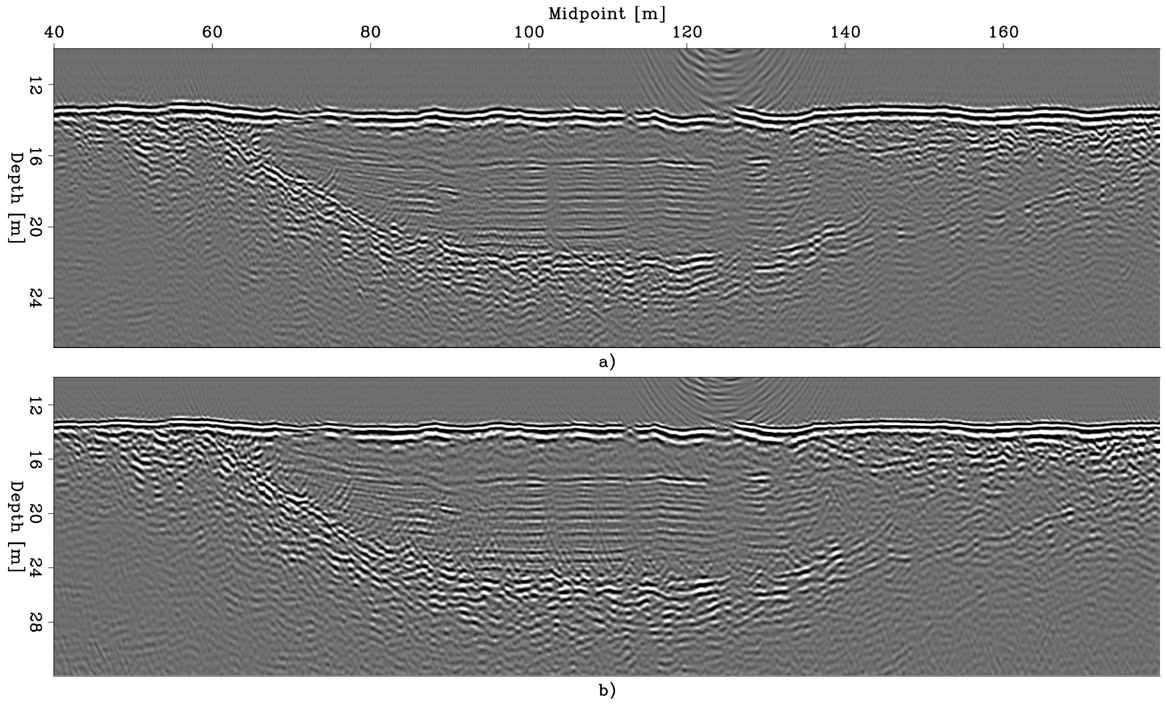


Figure 11: Comparison of the entire images obtained with: (a) initial velocity, and (b) final velocity. [**CR**] biondo1/. Mig-overn

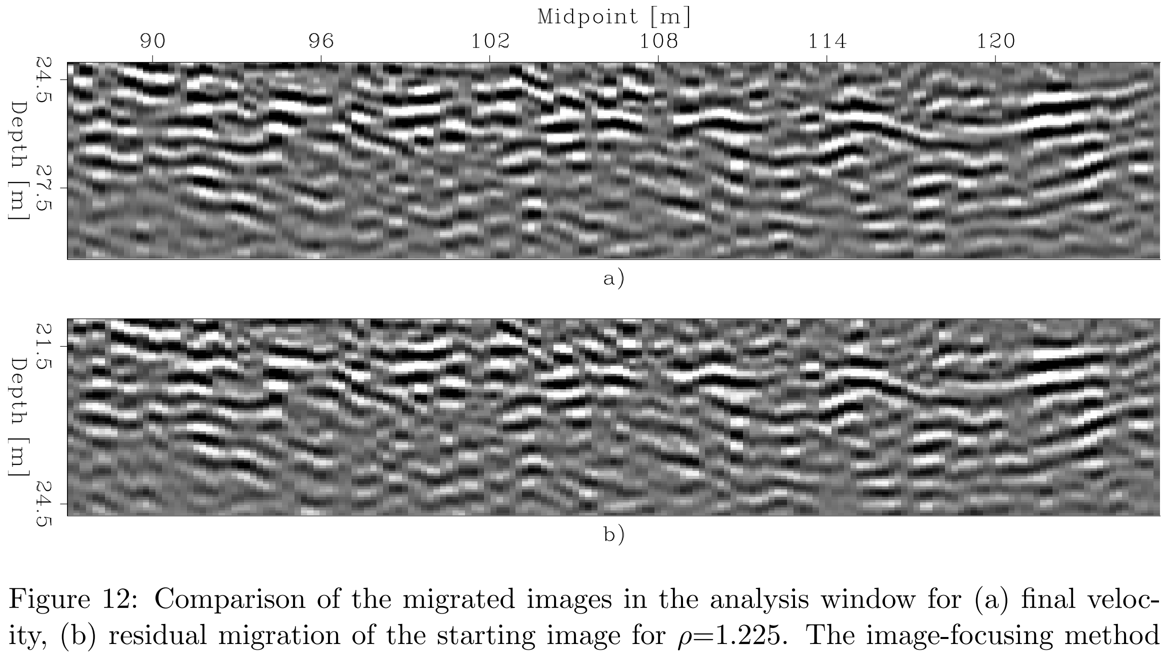


Figure 12: Comparison of the migrated images in the analysis window for (a) final velocity, (b) residual migration of the starting image for ρ=1.225. The image-focusing method selected the bottom image to be the best focused image among the ensemble of residual migrated images obtained from the initial migration. [**CR**] biondo1/. ResMig-win-overn

ACKNOWLEDGMENTS

I would like to thank the United States Army Corps of Engineers (USACE) New York District and Earthworks under contract number W912DS-06-D-0001 for providing the field data. I would also to acknowledge Daniel Rosales with Earthworks for his enthusiastic help in providing the data to SEP.

REFERENCES

Biondi, B., 2006, 3D Seismic Imaging: Society of Exploration Geophysicists.

———, 2008, An image-focusing semblance functional for velocity analysis: SEP-Report, **136**, 43–54.

———, 2009, Measuring image focusing for velocity analysis: SEP-Report, **138**, 59–79.

Fomel, S., 2002, Applications of plane-wave destruction filters: Geophysics, **67**, 1946–1960.

Sava, P., 2004, Migration and velocity analysis by wavefield extrapolation: PhD thesis, Stanford University.

Sava, P. C., 2003, Prestack residual migration in frequency domain: Geophysics, **68**, 634–640.

Schock, S. G., L. R. LeBlanc, and L. A. Mayer, 1989, Chirp subbottom profiler for quantitative sediment analysis: Geophysics, **54**, 445–450.

Schock, S. G., L. R. LeBlanc, and S. A. Panda, 1994, Spatial and temporal pulse design considerations for a marine sediment classification sonar: IEEE Journal of Oceanic Engineering, **19**, 406–415.

Stanford Exploration Project, Report SEP139, October 19, 2009

Gradient of image-space wave-equation tomography by the adjoint-state method

Claudio Guerra

ABSTRACT

Optimization with gradient-descent techniques requires computing the gradient of the objective function. The gradient can be determined by using the Frechét derivatives, but, for practical problems, this can be very expensive. The gradient can be more efficiently computed by the adjoint-state method, which does not require the use of the Frechét derivatives. Here, I derive the gradient of the image-space wave-equation tomography using the adjoint-state method. I also show its application with a numerical example using image-space phase-encoded gathers.

INTRODUCTION

Wave-equation tomography aims to solve for earth models that explain observed seismograms under some norm. There are two main categories, depending on the domain in which the objective function is computed. In one category, known as waveform inversion (Lines and Treitel, 1984; Tarantola, 1987; Woodward, 1992), the objective function is defined in the data space, where the modeled data are compared with the recorded seismograms. In the other category, here called image-space wave-equation tomography (ISWET), the objective function is minimized in the image space. Two variants of ISWET are wave-equation migration velocity analysis (WEMVA) (Sava and Biondi, 2004a,b) and differential semblance velocity analysis (DSVA) (Shen, 2004; Shen and Symes, 2008).

Like waveform inversion, ISWET is a computationally demanding process. This computational cost is commonly decreased by using generalized sources (Shen and Symes, 2008; Tang et al., 2008). Because wavefield propagation is a linear process, generalized sources are computed by linearly combining source wavefields and receiver wavefields, using phase-encoding techniques (Whitmore, 1995; Romero et al., 2000). In particular, Guerra et al. (2009) used phase-encoding modeling in the image space to synthesize generalized source functions that drastically decrease the cost of DSVA.

WEMVA and DVSA seek the optimal slowness by driving an image perturbation to a minimum. However, they differ in the way the image perturbation is computed and, consequently, in the numerical optimization scheme applied. As Biondi (2008) points out, WEMVA is not easily automated. The image perturbation is computed by the linearized residual prestack-depth migration (Sava and Biondi, 2004a), which uses a manually picked residual-moveout parameter. Because the perturbed image computed with the linearized residual prestack-depth migration is consistent with the application of the forward wave-equation tomographic operator, the forward and adjoint ISWET operators can be used in conjugate-gradient methods to invert for the slowness perturbation. In DSVA, the perturbed image is computed by applying the fully automated differential-semblance operator (DSO) to the subsurface-offset gathers (ODCIGs) or angle gathers (ADCIGs). When applied to ODCIGs, DSVA minimizes the energy not focused at zero-offset. When applied to ADCIGs, DSVA minimizes energy departing from flatness of the reflectors. Although DSVA easily automates ISWET, it produces perturbed images that do not present the depth phase-shift introduced by the forward one-way ISWET operator, neither the DVSA amplitude behavior can be modeled by this operator. Therefore, the objective function computed with DSO is tipically minimized by quasi-Newton algorithms, which require computation of the gradient of the objective function.

The gradient of the objective function can be computed with the Frechét derivatives. However, even for 2D applications of ISWET this computation can be very expensive. An efficient way to compute the gradient without using Frechét derivatives is the adjoint-state method (Chavent and Jacewitz, 1995; Plessix, 2006). Plessix (2006) describes two methodologies for computing the gradient of the objective function using the adjoint-state method. One methodology uses the perturbation theory, which states that, at first order, a perturbation of the model parameters causes a perturbation of the objective function. The other uses the augmented Lagrangian. The augmented Lagrangian is formed by the objective function and the scalar product of the adjoint-state variables with general solutions of the forward modeling equations. The adjoint-state variables are, in turn, solutions of the adjoint-state equations. The adjoint-state equations are defined by equating to zero the derivatives of the augmented Lagrangian with respect to the state variables. For the linear case, the adjoint of the modeling operator applied to the adjoint-state variables gives the gradient of the objective function.

Although previous studies have computed the gradient of the ISWET objective function by the adjoint-state method (for example, Shen et al. (2003)), they have not provided a detailed derivation. Here, I show a detailed derivation the gradient of the ISWET objective function using the augmented Lagrangian methodology. The derivation is valid whether ISWET uses areal-shot migration or shot-profile migration. A complete description of the forward and adjoint of ISWET operators is given in Tang et al. (2008). A numerical example of slowness optimization on the Marmousi model illustrates the use of the gradient by a quasi-Newton algorithm.

Because I use image-space phase-encoded gathers in the numerical example, for completeness, I first briefly describe how to compute these phase-encoded gathers. A detailed treatment of the image-space phase-encoded gathers can be found in Biondi (2006, 2007) and Guerra and Biondi (2008a). Then I derive the ISWET gradient using the adjoint-state method, and show the numerical example .

IMAGE-SPACE PHASE-ENCODED GATHERS

Biondi (2006, 2007) introduced the concept of the prestack exploding-reflector as a generalization of the exploding-reflector method (Loewenthal et al., 1976). The prestack exploding-reflector modeling synthesizes areal data and the corresponding areal source function, having as an initial condition a prestack image computed with wave-equation migration. If the slowness is accurate and the energy is focused at zero-subsurface offset, the prestack exploding-reflector modeling reduces to the conventional exploding-reflector method. Basically, the prestack exploding-reflector method models one single reflection event from one single ODCIG by recursive upward continuation with the following one-way wave equations:

$$\left\{ \begin{array}{l} \left(\frac{\partial}{\partial z} - i\sqrt{\omega^2 \hat{s}^2(\mathbf{x}) - |\mathbf{k}|^2}\right) d(\mathbf{x}, \omega; x_m, y_m) = r_D(\mathbf{x}, \mathbf{h}; x_m, y_m) \\ d(x, y, z = z_{\max}, \omega; x_m, y_m) = 0 \end{array} \right. , \tag{1}$$

and

$$\left\{ \begin{array}{l} \left(\frac{\partial}{\partial z} + i\sqrt{\omega^2 \hat{s}^2(\mathbf{x}) - |\mathbf{k}|^2}\right) u(\mathbf{x}, \omega; x_m, y_m) = r_U(\mathbf{x}, \mathbf{h}; x_m, y_m) \\ u(x, y, z = z_{\max}, \omega; x_m, y_m) = 0 \end{array} \right. , \tag{2}$$

where $r_D(\mathbf{x}, \mathbf{h}; x_m, y_m)$ and $r_U(\mathbf{x}, \mathbf{h}; x_m, y_m)$ are a single ODCIG at the horizontal location (x_m, y_m) with a single reflector, and are suitable initial conditions for modeling the source and receiver wavefields, respectively. They are obtained by rotating the original unfocused ODCIGs according to the apparent geological dip of the reflector. This operation maintains the velocity information needed for migration velocity analysis, especially for dipping reflectors (Biondi, 2007). Note that $d(x, y, z = z_c, \omega; x_m, y_m)$ is the areal source data and $u(x, y, z = z_c, \omega; x_m, y_m)$ is the areal receiver data for a single reflector and a single ODCIG located at (x_m, y_m); $z = z_c$ denotes that the wavefields can be collected at any depth level, z_c. This characteristic is important for accelerating ISWET, especially if z_c separates regions of sufficiently accurate slowness above and inaccurate slowness below. Therefore, the synthesized gathers are naturally datumized, and the wavefield propagations during ISWET can be restricted to the region where the slowness model must be updated. This feature allows the application of ISWET to be target-oriented.

As initially formulated, if one models a single reflection from one ODCIG at a time, the prestack exploding-reflector method generates a dataset that can be orders of magnitude bigger than the original dataset. As discussed by Biondi (2006) and Guerra and Biondi (2008b,a), modeling several reflectors and several ODCIGs simultaneously and using random phase encoding generates a much smaller dataset that, when migrated, do not produce crosstalk. The randomly encoded areal source and areal receiver wavefields can be computed as follows:

$$\left\{ \begin{array}{l} \left(\frac{\partial}{\partial z} - i\sqrt{\omega^2 \hat{s}^2(\mathbf{x}) - |\mathbf{k}|^2}\right) \widetilde{d}(\mathbf{x}, \mathbf{p}_m, \omega) = \widetilde{r}_D(\mathbf{x}, \mathbf{h}, \mathbf{p}_m, \omega) \\ \widetilde{d}(x, y, z = z_{\max}, \mathbf{p}_m, \omega) = 0 \end{array} \right. , \tag{3}$$

and

$$\left\{ \begin{array}{l} \left(\frac{\partial}{\partial z} + i\sqrt{\omega^2 \hat{s}^2(\mathbf{x}) - |\mathbf{k}|^2}\right) \widetilde{u}(\mathbf{x}, \mathbf{p}_m, \omega) = \widetilde{r}_U(\mathbf{x}, \mathbf{h}, \mathbf{p}_m, \omega) \\ \widetilde{u}(x, y, z = z_{\max}, \mathbf{p}_m, \omega) = 0 \end{array} \right. , \tag{4}$$

where $\widetilde{r}_D(\mathbf{x}, \mathbf{h}, \mathbf{p}_m, \omega)$ and $\widetilde{r}_U(\mathbf{x}, \mathbf{h}, \mathbf{p}_m, \omega)$ are the encoded ODCIGs after rotations. They are defined as follows:

$$\widetilde{r}_D(\mathbf{x}, \mathbf{h}, \mathbf{p}_m, \omega) = \sum_{x_m} \sum_{y_m} r_D(\mathbf{x}, \mathbf{h}, x_m, y_m) \beta(\mathbf{x}, x_m, y_m, \mathbf{p}_m, \omega), \tag{5}$$

$$\widetilde{r}_U(\mathbf{x}, \mathbf{h}, \mathbf{p}_m, \omega) = \sum_{x_m} \sum_{y_m} r_U(\mathbf{x}, \mathbf{h}, x_m, y_m) \beta(\mathbf{x}, x_m, y_m, \mathbf{p}_m, \omega), \tag{6}$$

where $\beta(\mathbf{x}, x_m, y_m, \mathbf{p}_m, \omega) = e^{i\gamma(\mathbf{x}, x_m, y_m, \mathbf{p}_m, \omega)}$ is a pseudo-random phase-encoding function, with $\gamma(\mathbf{x}, x_m, y_m, \mathbf{p}_m, \omega)$ being a uniformly distributed random sequence in $\mathbf{x}$, x_m, y_m and ω; the variable $\mathbf{p}_m$ is the index of different realizations of the random sequence. The recursive solution of equations 3 and 4 gives the encoded areal source data $\widetilde{d}(x, y, z = z_c, \mathbf{p}_m, \omega)$ and areal receiver data $\widetilde{u}(x, y, z = z_c, \mathbf{p}_m, \omega)$, at the depth level, z_c.

GRADIENT OF ISWET BY THE ADJOINT-STATE METHOD

Image-space wave-equation tomography aims to iteratively solve for the slowness model, $s = s(\mathbf{x})$, that minimizes the nonlinear objective function:

$$J(s) = \frac{1}{2}\left|\left|\Delta r(s)\right|\right|^2 = \frac{1}{2}\left|\left|r(s) - \mathbf{M}r(s)\right|\right|^2, \tag{7}$$

where $\Delta r = \Delta r(\mathbf{x},\mathbf{h})$ is the image perturbation that measures the accuracy of the slowness model. To compute Δr, a differential residual-focusing operator $\mathbf{M}$ is applied to the image $r = r(\mathbf{x},\mathbf{h})$ obtained with the current slowness (Biondi, 2008), using either differential residual prestack migration (Sava and Biondi, 2004a,b) or differential-semblance optimization (DSO) operators (Shen and Symes, 2008). In this paper, operators are represented by bold capital letters.

If the differential residual-focusing operator $\mathbf{M}$ is independent of the slowness, the gradient of this objective function evaluated at the current slowness $\widehat{s} = \widehat{s}(\mathbf{x})$ is

$$\nabla J(s) = \left(\frac{\partial r}{\partial s}\right)'\Bigg|_{s=\widehat{s}} \left(\mathbf{I} - \mathbf{M}'\right)\Delta\widehat{r}, \tag{8}$$

where the prime denotes the adjoint, $\mathbf{I}$ is the identity operator, and $\Delta\widehat{r} = \Delta\widehat{r}(\mathbf{x},\mathbf{h})$ is the perturbed image obtained with the current slowness model. The linear operator $\frac{\partial r}{\partial s}$ defines the mapping $\frac{\partial r}{\partial s}\Delta s = \Delta r$ between the slowness perturbation Δs and the image perturbation Δr; it is called the image-space wave-equation tomographic operator.

Because the image-space wave-equation tomographic operator is composed of different operators, it is difficult to envision from equation 8 which operations are performed to compute the gradient. Therefore, for a clear explanation of the operators involved, I use the adjoint-state method to derive the gradient of the objective function (equation 7).

In migration with generalized sources or shot-profile migration, the source and receiver wavefields are propagated independently, and the image $r_z = r_z(\mathbf{x},\mathbf{h})$ at a depth level z, is computed by the crosscorrelation

$$r_z(\mathbf{x},\mathbf{h}) = \sum_{\omega} d_z^*(\mathbf{x}-\mathbf{h},\omega)u_z(\mathbf{x}+\mathbf{h},\omega), \tag{9}$$

where $d_z(\mathbf{x},\omega)$ is the source wavefield for a single frequency ω at horizontal coordinates $\mathbf{x} = (x,y)$; $u_z(\mathbf{x},\omega)$ is the receiver wavefield, $\mathbf{h} = (h_x,h_y)$ is the subsurface half-offset, and "*" stands for the complex-conjugate. An additional summation over shots is required when migrating more than one shot. Hereafter, letters d and u stand for source and receiver wavefields, respectively, irrespective of the migration scheme.

In a more compact notation, not explicitly writing the dependencies on $\mathbf{x}$ and $\mathbf{h}$, equation 9 can be re-written as follows:

$$r_z = \mathbf{S}\mathbf{D}_z'(\omega)u_z(\omega) = \mathbf{S}\mathbf{U}_z(\omega)d_z^*(\omega), \tag{10}$$

where $\mathbf{D}$ and $\mathbf{U}$ are convolutional operators composed of (h_x,h_y)-shifted versions of $d_z(\mathbf{x},\omega)$ and $u_z(\mathbf{x},\omega)$, respectively. Operator $\mathbf{S}$ corresponds to the summation over frequency.

For subsequent depth levels, $d(\mathbf{x},\omega)$ is computed by means of the recursive downward propagation

$$\begin{cases} d_{z+1}(\omega) = \mathbf{T}_z^{\downarrow}(\omega, s) d_z(\omega) \\ d_1(\omega) = q(\omega), \end{cases} \tag{11}$$

where $\mathbf{T}_z^{\downarrow}$ is the downward continuation operator, which is a function of the slowness s, and $q(\omega)$ is the source wavefield used as a boundary condition. In the case of conventional shot-profile migration, $q(\omega) = f_s(\omega)\delta(\mathbf{x} - \mathbf{x_s})$ is the source signature located at $\mathbf{x}_s = (x_s, y_s, 0)$. If using the generalized sources in the image space, $q(\omega)$ represents the image-space phase-encoded source wavefield of equation 3.

The downward continuation of the receiver wavefield is performed by

$$\begin{cases} u_{z+1}(\omega) = \mathbf{T}_z^{\downarrow}(\omega, s) u_z(\omega) \\ u_1(\omega) = w(\omega), \end{cases} \tag{12}$$

where $w(\omega)$ is the recorded data at the surface for shot-profile migration. If using generalized sources in the image space, $w(\omega)$ is the phase-encoded areal receiver wavefield of equation 4. In equations 11 and 12, I omitted the dependencies of the wavefield with respect to $\mathbf{x}$. The subscript *1* in equations 11 and 12 represents the surface for the shot-profile migration and the "collection" depth level, z_c, for the image-space phase-encoded wavefields.

In the image-space wave-equation tomography problem, the perturbed source and receiver wavefields, and the image perturbation are used to compute the slowness perturbation that updates the current slowness model. From the perturbation theory, we have $d = \widehat{d} + \Delta d$, $u = \widehat{u} + \Delta u$, and, consequently, $r = \widehat{r} + \Delta r$ are physical realizations with $s = \widehat{s} + \Delta s$, where the *hat* refers to fields obtained with the background slowness. To the first order (Born approximation), these perturbed fields are given by

$$\Delta d_{z+1}(\omega) = \mathbf{T}_z^{\downarrow}(\omega, \widehat{s}) \Delta d_z(\omega) + \widetilde{\widetilde{\mathbf{D}}}_z(\omega) \Delta s_z \tag{13}$$

and

$$\Delta u_{z+1}(\omega) = \mathbf{T}_z^{\downarrow}(\omega, \widehat{s}) \Delta u_z(\omega) + \widetilde{\widetilde{\mathbf{U}}}_z(\omega) \Delta s_z. \tag{14}$$

The diagonal operators $\widetilde{\widetilde{\mathbf{D}}}_z$ and $\widetilde{\widetilde{\mathbf{U}}}_z$ have in the diagonal entries the scattered source and receiver wavefields, respectively. These wavefields are given by the action of the scattering operator $\Delta \mathbf{T}_z^{\downarrow}$ on the background wavefields:

$$\widetilde{\widetilde{\mathbf{D}}}_z(\omega) = \Delta \mathbf{T}_z^{\downarrow}(\omega, \widehat{s}) \widehat{d}_z(\omega) = i \frac{\omega^2 \widehat{s}}{\sqrt{\omega^2 \widehat{s}^2 - |\mathbf{k}|^2}} dz \ \widehat{d}_z(\omega) \tag{15}$$

and

$$\widetilde{\widetilde{\mathbf{U}}}_z(\omega) = \Delta \mathbf{T}_z^{\downarrow}(\omega, \widehat{s}) \widehat{u}_z(\omega) = -i \frac{\omega^2 \widehat{s}}{\sqrt{\omega^2 \widehat{s}^2 - |\mathbf{k}|^2}} dz \ \widehat{u}_z(\omega). \tag{16}$$

The perturbed image is given by

$$\Delta r_z = \mathbf{S} \left(\widehat{\mathbf{U}}_z(\omega) \Delta d_z^*(\omega) + \widehat{\mathbf{D}}_z'(\omega) \Delta u_z(\omega) \right). \tag{17}$$

The matrix representations of equations 13, 14, 17 are

$$\Delta\underline{\mathbf{d}} = \mathbf{T}^{\downarrow}\Delta\underline{\mathbf{d}} + \widetilde{\widetilde{\mathbf{P}}}\mathbf{S}'\Delta\mathbf{s}, \tag{18}$$

$$\Delta\underline{\mathbf{u}} = \mathbf{T}^{\downarrow}\Delta\underline{\mathbf{u}} + \widetilde{\widetilde{\mathbf{U}}}\mathbf{S}'\Delta\mathbf{s}, \tag{19}$$

and

$$\Delta\underline{\mathbf{r}} = \mathbf{S}\left(\widehat{\mathbf{U}}\Delta\underline{\mathbf{d}}^{*} + \widehat{\mathbf{D}}'\Delta\underline{\mathbf{u}}\right), \tag{20}$$

where $\mathbf{S}'$ is a spreading operator that replicates the slowness perturbation for every frequency.

Equations 18, 19, and 20 are the forward modeling equations of the image-space wave-equation tomography problem using the generalized sources or shot-profile schemes. They depend on the state variables $\Delta\underline{\mathbf{d}}$, $\Delta\underline{\mathbf{u}}$, and $\Delta\underline{\mathbf{r}}$. Plessix (2006) describes how to compute the adjoint states using the augmented functional methodology. By introducing the adjoint-state variables $\underline{\lambda}_d$, $\underline{\lambda}_u$, and $\underline{\lambda}_r$, the augmented Lagrangian reads

$$\begin{aligned}\mathcal{L}(\Delta\underline{\mathbf{d}},\Delta\underline{\mathbf{u}},\Delta\underline{\mathbf{r}},\underline{\lambda}_d,\underline{\lambda}_u,\underline{\lambda}_r;\Delta\mathbf{s}) = \mathcal{R}\Big[\tfrac{1}{2}\,\|\Delta\underline{\mathbf{r}}\|^2 - &\\ \left\langle \underline{\lambda}_d, \left(\mathbf{I}-\mathbf{T}^{\downarrow}\right)\Delta\underline{\mathbf{d}} - \widetilde{\widetilde{\mathbf{D}}}\mathbf{S}'\Delta\mathbf{s}\right\rangle - &\\ \left\langle \underline{\lambda}_u, \left(\mathbf{I}-\mathbf{T}^{\downarrow}\right)\Delta\underline{\mathbf{u}} - \widetilde{\widetilde{\mathbf{U}}}\mathbf{S}'\Delta\mathbf{s}\right\rangle - &\\ \left\langle \underline{\lambda}_r, \Delta\underline{\mathbf{r}} - \mathbf{S}\left(\widehat{\mathbf{U}}\Delta\underline{\mathbf{d}}^{*} + \widehat{\mathbf{D}}'\Delta\underline{\mathbf{u}}\right)\right\rangle. &\end{aligned}$$

The adjoint-state variables are computed by taking the derivative of $\mathcal{L}$ with respect to the state variables and equating to zero, which gives

$$\left(\mathbf{I}-\mathbf{T}^{\downarrow}\right)'\underline{\lambda}_d = \widehat{\mathbf{U}}\underline{\lambda}_r, \tag{21a}$$

$$\left(\mathbf{I}-\mathbf{T}^{\downarrow}\right)'\underline{\lambda}_u = \widehat{\mathbf{D}}\underline{\lambda}_r, \tag{21b}$$

$$\underline{\lambda}_r = \Delta\underline{\mathbf{r}}. \tag{21c}$$

Notice that

$$\left(\mathbf{I}-\mathbf{T}^{\downarrow}\right)' = \left(\mathbf{I}-\mathbf{T}^{\downarrow'}\right) = \left(\mathbf{I}-\mathbf{T}^{\uparrow}\right) \tag{22}$$

corresponds to the recursive upward propagation operator. Therefore, equations 21a and 21b can be written as

$$\underline{\lambda}_p = \mathbf{T}^{\uparrow}\underline{\lambda}_d + \widehat{\mathbf{U}}\underline{\lambda}_r, \tag{23a}$$

$$\underline{\lambda}_u = \mathbf{T}^{\uparrow}\underline{\lambda}_u + \widehat{\mathbf{D}}\underline{\lambda}_r, \tag{23b}$$

which correspond to the recursive upward propagation of the perturbed wavefields resulting from the convolution of the wavefields computed with the current slowness and the perturbed image.

Finally, the gradient of J is

$$\nabla_s J(\mathbf{s}) = \mathbf{S}\left(\widetilde{\widetilde{\mathbf{D}}}'\underline{\lambda}_d + \widetilde{\widetilde{\mathbf{U}}}'\underline{\lambda}_u\right). \tag{24}$$

To compute the gradient, the adjoint-state wavefields, $\underline{\lambda}_d$ and $\underline{\lambda}_u$, are upward propagated and cross-correlated in time with the scattered wavefields.

NUMERICAL EXAMPLE

The gradient of the ISWET objective function computed in the previous section can be used in a quasi-Newton optimization scheme. I use the L-BFGS-B bound constrained optimization algorithm (Nocedal and Wright, 2000) to invert for slowness using the Marmousi model. A B-spline smoothing, with nodes spaced at 240 m in x and 16 m in z, is applied to the gradient to prevent problems caused by poor illumination and noise. I model 375 two-way shots with 24 m spacing using the original Marmousi model of Figure 1a. The maximum offset is 6600 m.

The one-way shot profile image with the correct slowness model can be seen in Figure 1b. The modeled data were also migrated using the background slowness of Figure 2a to compute the background image of Figure 2b. The background slowness differs from the true slowness in a limited region, as can be seen in Figure 3, which shows the ratio between true and background slowness. When comparing the two images, there is a clear pull-up effect at the center-bottom of the background image caused by migrating with a velocity which is too slow.

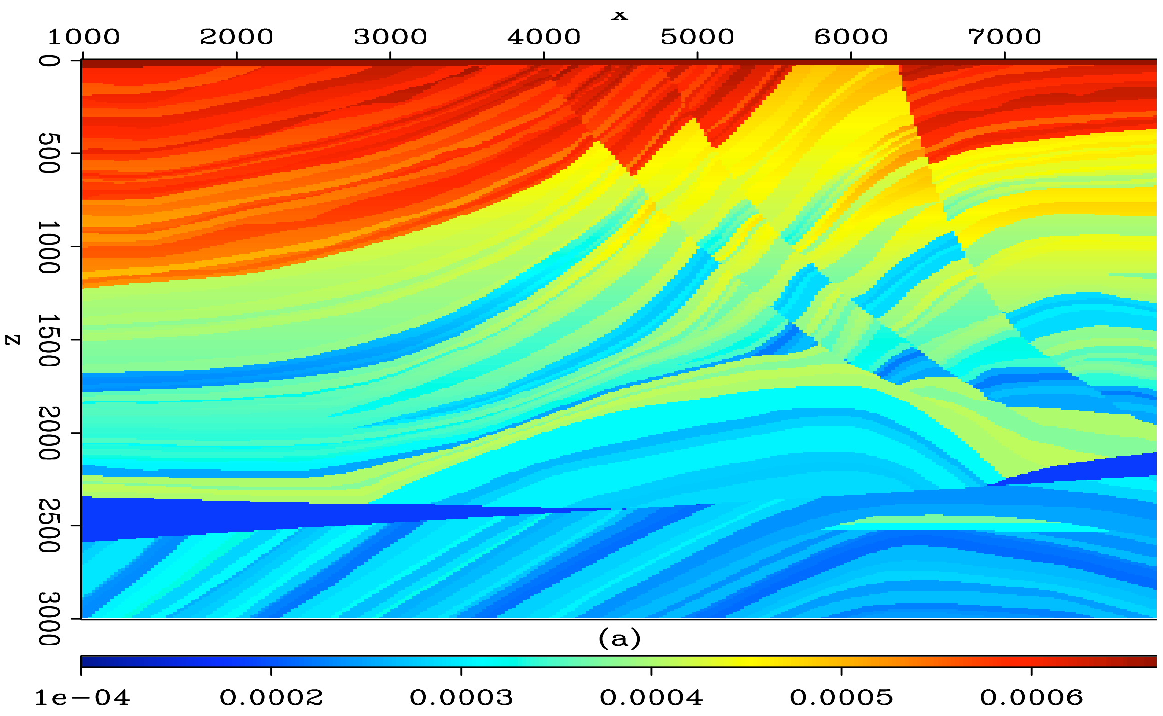
Following Guerra et al. (2009), representative reflectors were selected from the background image (Figure 4) to synthesize eleven image-space phase-encoded source and receiver wavefields using the background slowness. Prior to modeling, the selected reflectors were subjected to rotation according to the apparent geological dip as previously mentioned. I estimate that, by using only eleven areal shots, the computational cost decreased in 95% considering if the slowness optimization was carried out with the original 376 shot-profiles.

The optimization stopped after 4 iterations comprising 41 function and gradient evaluations. The optimized slowness model can be seen in Figure 5a. The optimized slowness decreased as expected, but did not recover the details of the true slowness. The smoothness of the optimized slowness is partially because of the B-spline filtering of the gradient but also because image-space wave-equation tomography cannot solve for the short wavelengths of the slowness model, as pointed out by Symes (2008).

The accuracy of the optimized slowness model can be evaluated by comparing the computed images with true, background, and optimized slowness. The shot-profile migration of the original shots using the optimized slowness is shown in Figure 5b. Notice that the pull-up effect, which occurs in the background image, is greatly mitigated. In addition, reflectors are more focused in the image computed with the optimized slowness than in the image computed with the background slowness. Another slowness accuracy indicator is flatness of the reflectors in the ADCIGs. The ADCIGs resulting from shot-profile migration of the original shots using, from top to bottom, the true slowness, the background slowness and the optimized slowness are shown in Figure 6. The reflectors computed with the optimized slowness are flatter than those with the background slowness, demonstrating the greater accuracy of the optimized slowness.

CONCLUSION

I showed a detailed derivation of the gradient of the objective function for the image-space wave-equation tomography by means of the adjoint-state method. I used the augmented La-



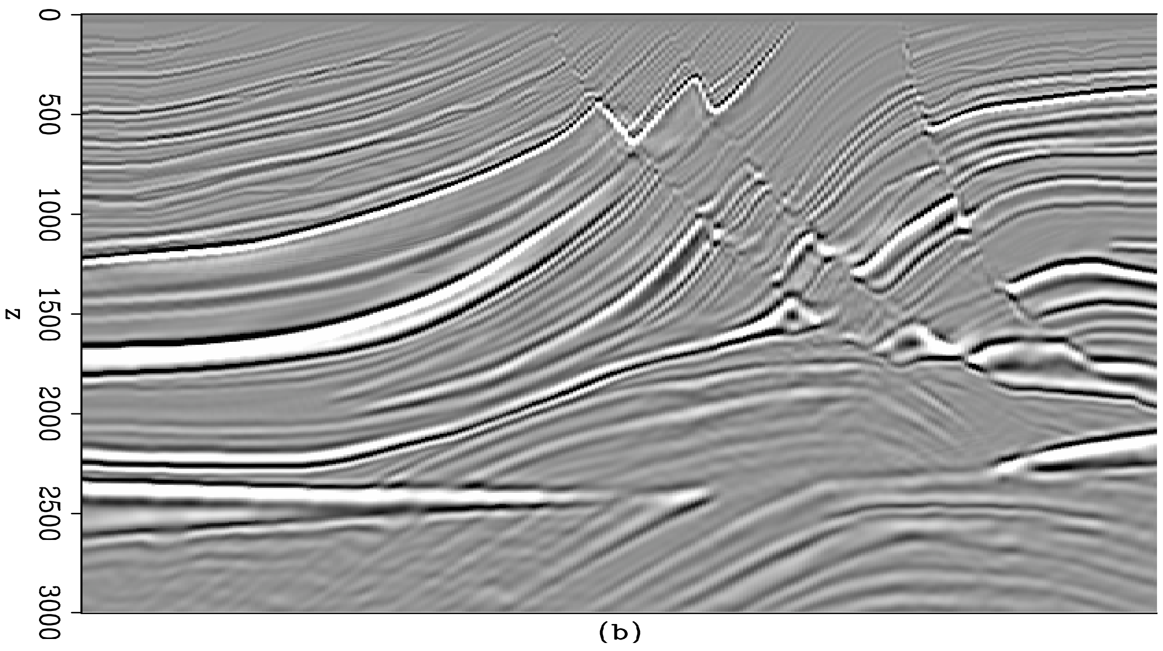


Figure 1: (a) True slowness of the Marmousi model. (b) Zero-subsurface-offset section of the one-way shot-profile migration with the true slowness model.[**CR**] claudio1/. true

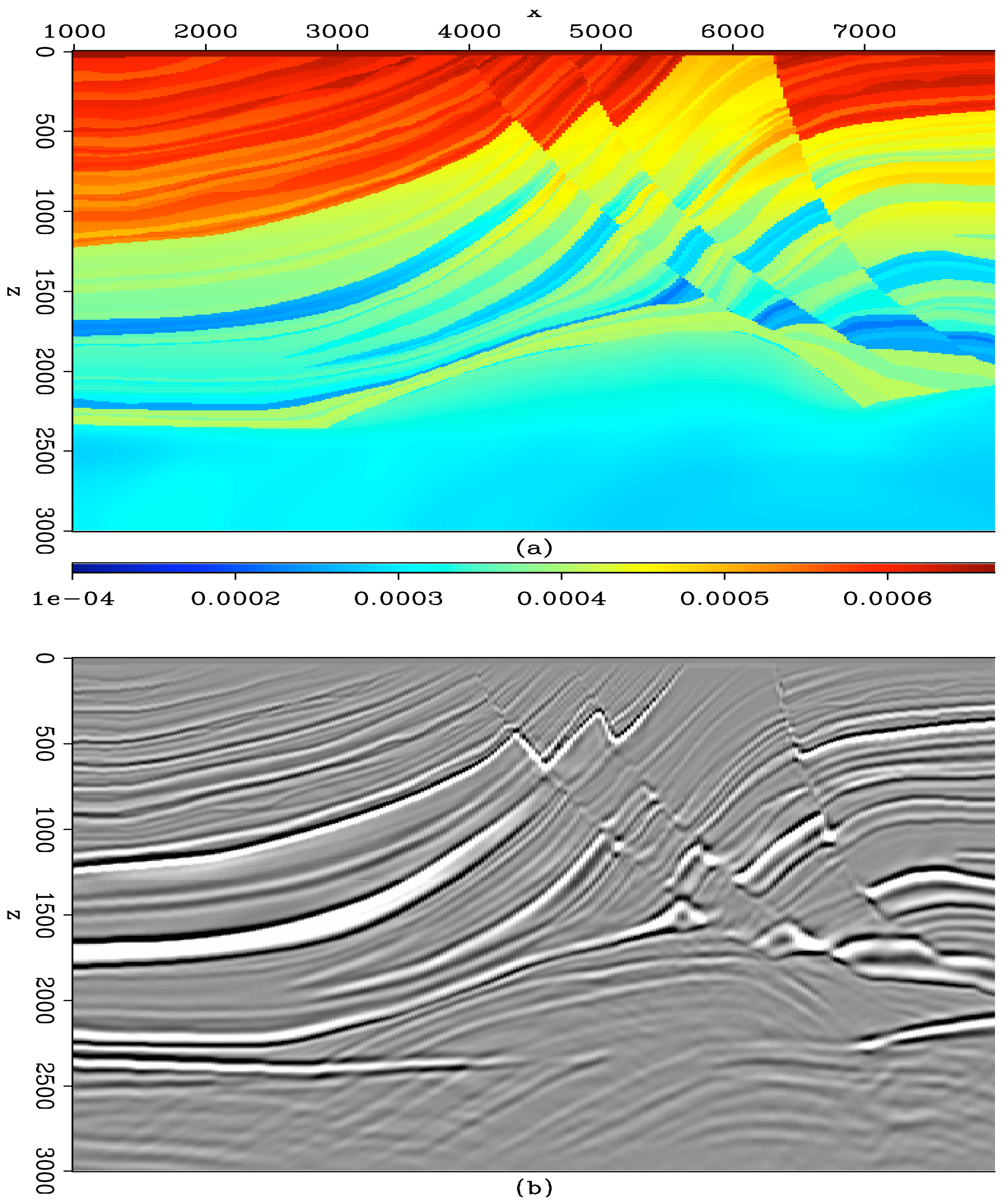


Figure 2: (a) Background slowness. (b) Zero-subsurface-offset section of the one-way shot-profile migration with the background slowness model.[**CR**] claudio1/. bckg

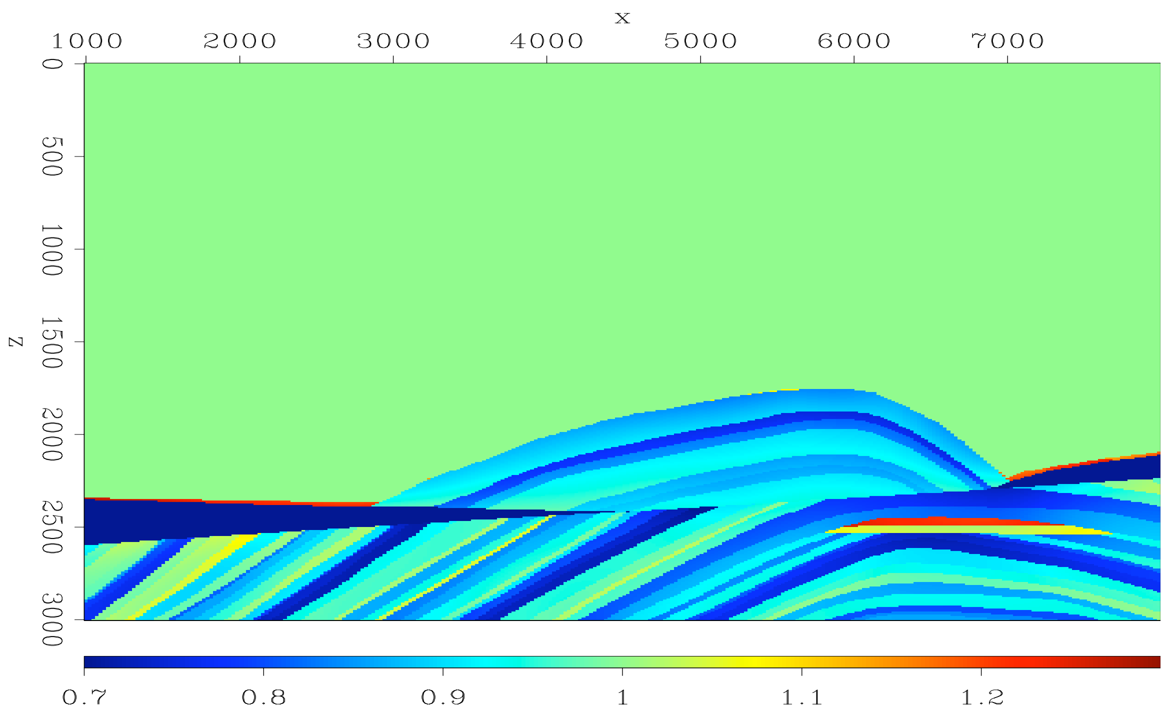


Figure 3: Ratio between the true and the background slowness.[**ER**] claudio1/. dslow

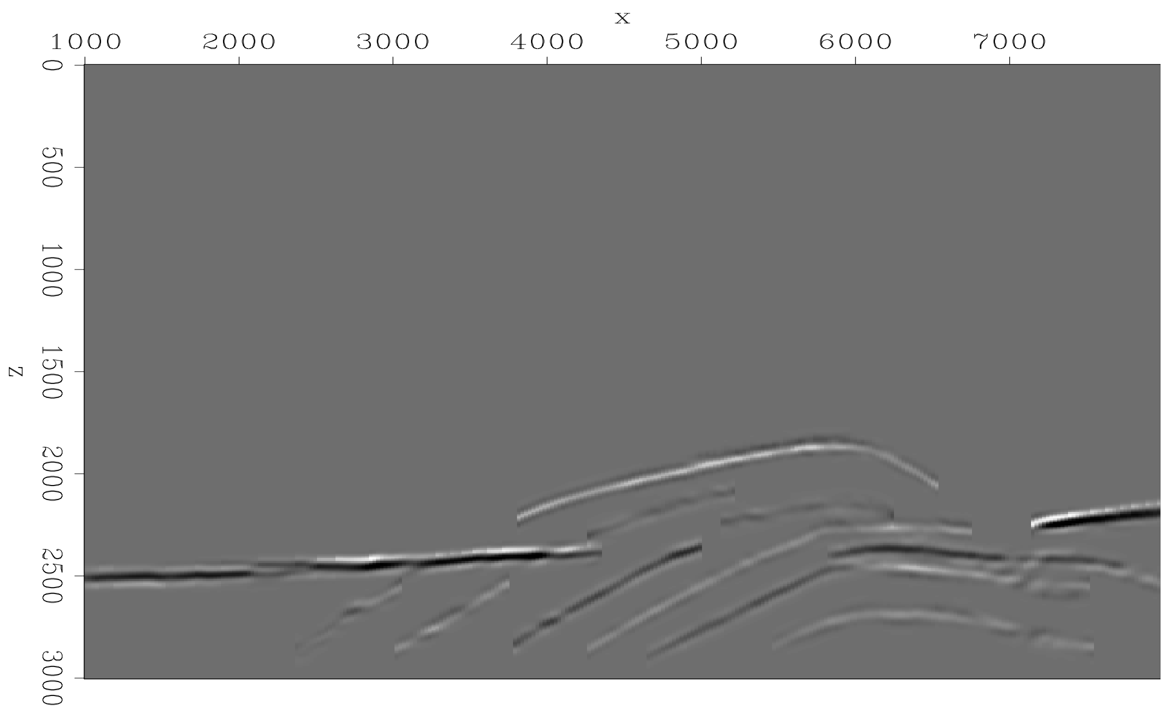


Figure 4: Zero-subsurface-offset section showing the reflectors selected on the one-way shot-profile migration with the background slowness model.[**CR**] claudio1/. bwkim

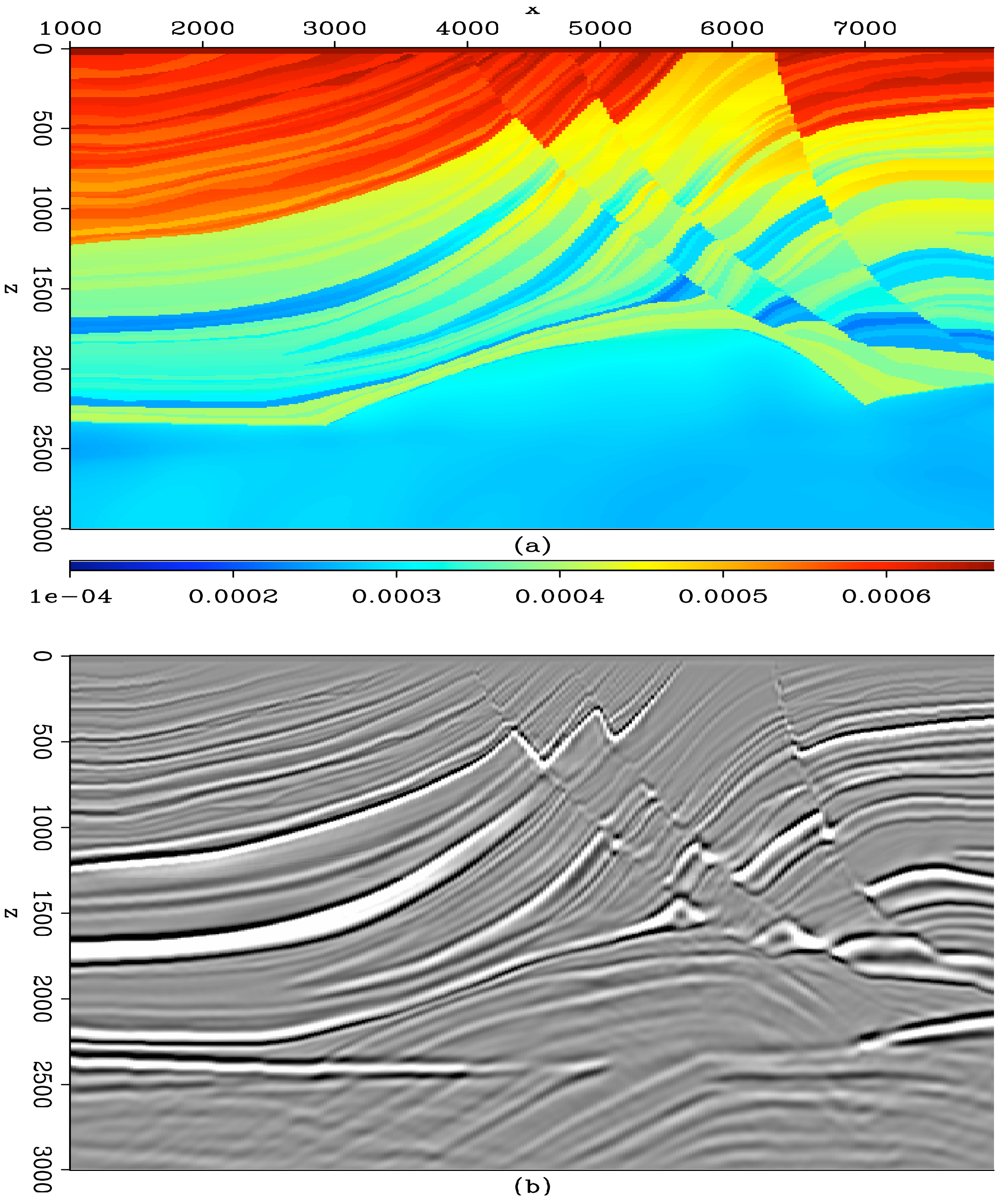


Figure 5: (a) Optimized slowness after 4 iterations with 41 function and gradient evaluations. (b) Zero-subsurface offset section of the one-way shot-profile migration with the optimized slowness model.[**CR**] claudio1/. opti

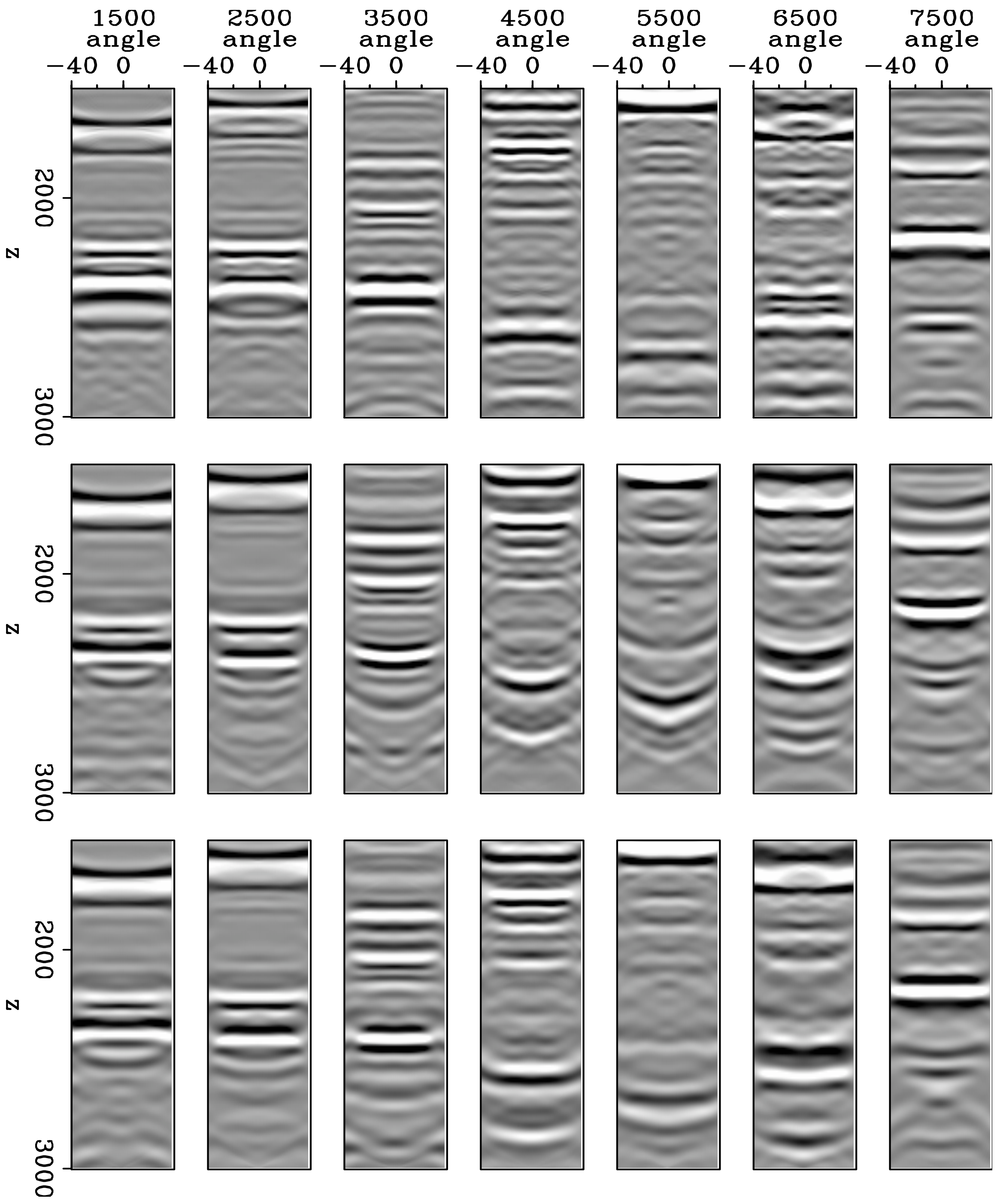


Figure 6: ADCIGs obtained with true slowness (top), background slowness (center), and optimized slowness (bottom).[**CR**] claudio1/. fang1

grangian methodology, which clarifies the action of the different operators that compose the image-space wave-equation tomography operator. The derivation is valid whether one considers migration of shot-profiles or generalized sources. Image-space phase-encoded gathers significantly accelerate image-space wave-equation tomography. In addition to decreasing the computational cost, the numerical example shows that image-space phase-encoded gathers can be used to generate sufficiently accurate optimized slowness.

REFERENCES

Biondi, B., 2006, Prestack exploding-reflectors modeling for migration velocity analysis: 76th Ann. Internat. Mtg., Expanded Abstracts, 3056–3060, Soc. of Expl. Geophys.

——, 2007, Prestack modeling of image events for migration velocity analysis: **SEP-131**, 101–118.

——, 2008, Automatic wave-equation migration velocity analysis: **SEP-134**, 65–78.

Chavent, G. and C. A. Jacewitz, 1995, Determination of background velocities by multiple migration fitting: Geophysics, **60**, 476–490.

Guerra, C. and B. Biondi, 2008a, Phase-encoding with Gold codes for wave-equation migration: **SEP-136**.

——, 2008b, Prestack exploding reflector modeling: The crosstalk problem: **SEP-134**, 79–92.

Guerra, C., Y. Tang, and B. Biondi, 2009, Wave-equation tomography using image-space phase-encoded data: **SEP-138**, 95–116.

Lines, L. and S. Treitel, 1984, A review of least-squares inversion and its application to geophysical problems: Geophys. Prospecting, **32**, 159–186.

Loewenthal, D., L. Lu, R. Roberson, and J. Sherwood, 1976, The wave equation applied to migration: Geophys.Prosp., **24**, 380–399.

Nocedal, J. and S. Wright, 2000, Numerical optimization: Springer Verlag,New York.

Plessix, R.-E., 2006, A review of the adjoint-state method for computing the gradient of a functional with geophysical applications: Geophys. J. Int., **167**, 495–503.

Romero, L. A., D. C. Ghiglia, C. C. Ober, and S. A. Morton, 2000, Phase encoding of shot records in prestack migration: Geophysics, **65**, 426–436.

Sava, P. and B. Biondi, 2004a, Wave-equation migration velocity analysis-I: Theory: Geophysical Prospecting, **52**, 593–606.

——, 2004b, Wave-equation migration velocity analysis-II: Examples: Geophysical Prospecting, **52**, 607–623.

Shen, P., 2004, Wave-equation Migration Velocity Analysis by Differential Semblance Optimization: PhD thesis, Rice University.

Shen, P. and W. W. Symes, 2008, Automatic velocity analysis via shot profile migration: Geophysics, **73**, VE49–VE59.

Shen, P., W. W. Symes, and C. C. Stolk, 2003, Differential semblance velocity analysis by wave-equation migration: SEG Technical Program Expanded Abstracts, **22**, 2132–2135.

Symes, B., 2008, Migration velocity analysis and waveform inversion: Geophysical Prospecting, **56**, 765–790.

Tang, Y., C. Guerra, and B. Biondi, 2008, Image-space wave-equation tomography in the generalized source domain: **SEP-136**, 1–22.

Tarantola, A., 1987, Inverse problem theory: Methods for data fitting and model parameter estimation: Elsevier.

Whitmore, N. D., 1995, An Imaging Hierarchy for Common Angle Plane Wave Seismogram: PhD thesis, University of Tulsa.

Woodward, M. J., 1992, Wave-equation tomography: Geophysics, **57**, 15–26.

Geophysical data integration and its application to seismic tomography

Mohammad Maysami

ABSTRACT

For oil exploration and reservoir monitoring purposes, we probe the earth's subsurface with a variety of geophysical methods, generating data with different natures, scales and frequency content. This diversity represents a large problem when trying to integrate all the gathered information. The concept of a shared earth by all these geophysical surveys suggests the presence of structural similarities in different data sets. For that purpose, it is necessary to work with geophysical properties that are scale-independent and not physical properties in individual layers. In this paper, I overview two methods for extracting structural information from data and using it as a constraint to the seismic tomography problem to compare different techniques and their effectiveness.

INTRODUCTION

In earth sciences, subsurface structures are studied by collecting large and various types of geophysical data. Different geophysical attributes of the subsurface are measured by a variety of geophysical measuring techniques, including but not limited to seismic, magnetic, and well-logs. For many years, different types of data have been used for specific stages of oil exploration and production. However, in recent years, many authors have considered using diverse data in geophysical inversion can reduce uncertainty (de Nardis et al., 2005; Bosch et al., 2005; Colombo and De Stefano, 2007). One of the main challenges of data integration is the difference in physical nature, scale, and frequency contents. All of the collected data, however, while measuring different properties, sample of the same geophysical structures. We can extract mathematical/geophysical properties from a data set that provides structural information. This structural information can be used in geophysical problems as auxiliary data to improve model estimation results by constraining the optimization problem. Geophysical inversion problems can benefit from this method in the form of a regularization misfit term that imposes structural similarity between the main and the auxiliary data fields. It is also applicable to both joint inversion and inversion using the auxiliary data.

STRUCTURAL SIMILARITY MEASURES

Properties of the subsurface can have very different dynamic ranges of values, the frequency band and scale in which they are measured. For example, magnetotelluric data has a much lower frequency than seismic data, while its dynamic range of amplitudes is broader. In this section, I review two attributes that provide some measure of structural similarity.

Cross-gradient function

Gallardo and Meju (2004) introduce the cross-gradient as follows:

$$\mathbf{g} = \nabla \mathbf{u} \times \nabla \mathbf{v}, \tag{1}$$

where **u** and **v** are two measured fields. By using normalized values of each data field, the cross-gradient function vanishes where the two fields are similar in structure or either one of them is very smooth. Maysami (2008) and Maysami and Clapp (2009) note that cross-gradient functions can be used as a measure of similarity. The cross-gradient function can be a suitable choice for the regularization (model-shaping) term of an optimization problem that imposes the structure of the auxiliary field on the estimated model. The differentiating nature of these functions, however, raises a sensitivity issue where there is a large gap between the frequency bands of different types of data. It is also expected that cross-gradient function is not a very effective choice for low-frequency data, since the smooth behavior causes the gradient to vanish.

Dip residual

In practice, the frequency spectrum of different types of geophysical data can change very widely, and there might be cases with no overlap in frequency band. To address these types of data integration problems, we need to choose properties that are frequency-independent. Local dip is one such property, which can be estimated by solving a regularized optimization (see Fomel (2000) for more details). Given a 2-D field $u(x, z)$, one can estimate the dip values where the data misfit function is given by

$$\begin{cases} \arg\min_{p_u} \qquad \mathbf{C}(p_u)\mathbf{u} \quad \text{subject to} \\ \\ \qquad\qquad \epsilon \mathbf{D} \Delta p_u \approx 0 \end{cases}, \tag{2}$$

where $\mathbf{C}(p_u)$ is a convolution operator with a 2-D filter based on the local dip, and $\mathbf{D}$ represents an appropriate roughening operator. The estimated local dip values are frequency independent, making dip a candidate for carrying geological information from one data field to another. Again, the estimated dips can be used in a regularization term of an optimization problem to impose the structure of the auxiliary data on the estimated model. This model misfit function can be given as follows:

$$r_m(x, z) = (\frac{\partial}{\partial x} + p_u \ \frac{\partial}{\partial z})v(x, z) \approx 0. \tag{3}$$

Note that the structural similarity measures do not include any physical link that might potentially exist between two fields. In other words, these functions only help the model-shaping part of optimization, and the physics of estimation lies in the data-misfit term with the mapping operator.

Figures 1 and 2 show examples where the reflectivity of the Marmousi synthetic model is used as auxiliary data to impose the geophysical structures on a random noise field and a smooth velocity field. The optimization problem used to generate the estimated model shown by Figures 1 and 2 is given by

$$\begin{cases} \arg\min_m \quad \left\| \mathbf{d} - \mathbf{I}\ \mathbf{m} \right\|_2^2 \quad \text{subject to} \\ \\ \qquad \epsilon\ \mathbf{A}(\mathbf{u})\ \mathbf{m} \approx 0 \end{cases}, \tag{4}$$

where $\mathbf{I}$ is identity matrix; $\mathbf{A}$ is either the cross-gradient or dip residual operator; $\mathbf{u}$ and $\mathbf{d}$ represent the auxiliary reflectivity field and the data (noise or smooth velocity field), respectively; and $\mathbf{m}$ is the estimated model which incorporates some of the structural information provided by $\mathbf{u}$. Note that a large ϵ is needed to emphasize on model-shaping term and impose structure. Figures 1(a) and 2(b) show the data $\mathbf{d}$ and the auxiliary field $\mathbf{u}$. Figure 1(d) shows a better reconstruction of the Marmousi structure with dip residual technique than the cross-gradient function (Figure 1(c)). This is clearly visible by comparing the continuity of reconstructed amplitude along the geological dips in Figure 1(d)).

Figure 2 shows a similar problem where we start with a smooth velocity of the Marmousi model instead of noise. Similarly, Figure 2(d) suggests that the dip residual technique leads to a less noisy partial reconstruction of the Marmousi structure; but not all the details are included. However, some of the horizontal parts of structures are reconstructed by the cross-gradient function (Figure 2(c)), but not with the dip residual (Figure 2(d)). Note that frequency ranges in smooth velocity and reflectivity are low and high, respectively.

APPLICATION TO SEISMIC TOMOGRAPHY

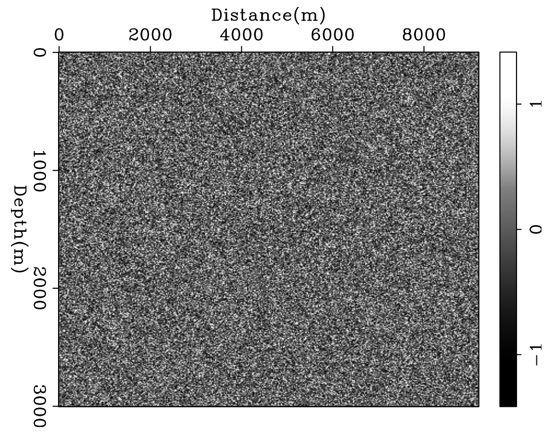
Following the earlier work by Maysami (2008) and Maysami and Clapp (2009), I apply the methods discussed above to regularize the seismic velocity tomography problem. In this optimization problem, the data misfit term includes the physics and kinematics of the tomography problem, while the regularization (model-shaping) term imposes the structure present in the auxiliary field, which is chosen to be either the reflectivity or the resistivity field (see Figure 3). This problem can be stated as

$$\begin{cases} \arg\min_{\Delta\mathbf{s}} \quad \left\| \ \Delta\mathbf{t} - \mathbf{T}_L\Delta\mathbf{s} \right\|_2^2 \quad \text{subject to} \\ \\ \qquad \epsilon \left\| \ \mathbf{A}(\mathbf{r})(\mathbf{s}_0 + \Delta\mathbf{s}) \ \right\|_2^2 \ \approx \ 0 \end{cases}, \tag{5}$$

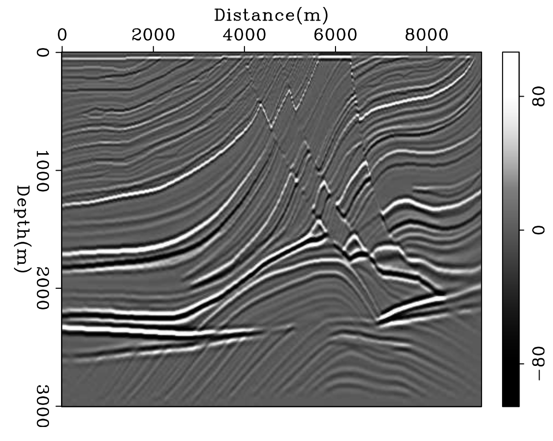
where $\mathbf{s}$ and $\mathbf{r}$ are the slowness and auxiliary fields, respectively; $\Delta\mathbf{t}$ and $\mathbf{T}_L$ represents traveltime updates and the linearized tomography mapping operator; and $\mathbf{A}$ is the model-shaping operator, which is picked as either the cross-gradient operator or the dip residual operator. In the latter case, the regularization operator is given by $\mathbf{A}(\mathbf{r}) = \frac{\partial}{\partial x} + p_r\ \frac{\partial}{\partial z}$.

Although the reflectivity field is in practice a function of velocity, we assume that it is an accurate representation. One may consider a more general case, where the regularization operator is also a function of model, which needs to be linearized around the current estimation.

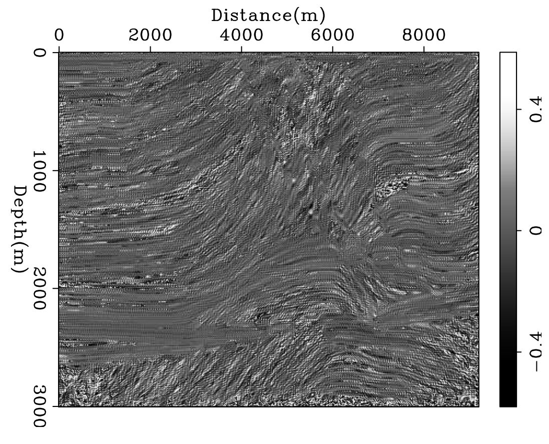
Figure 4 shows the results of solving the tomography problem (Equation 5) for an optimal update in velocity. I start with a smooth velocity as the initial guess. I use either



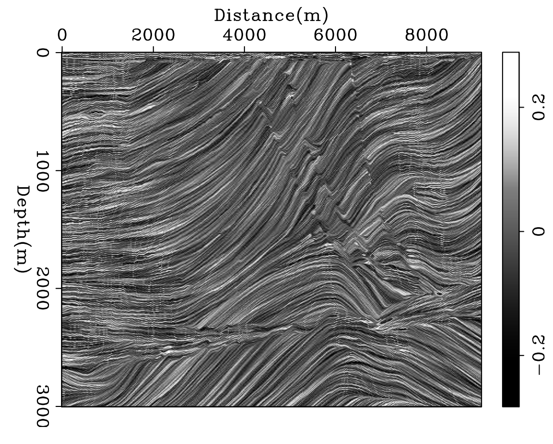
(a)



(b)

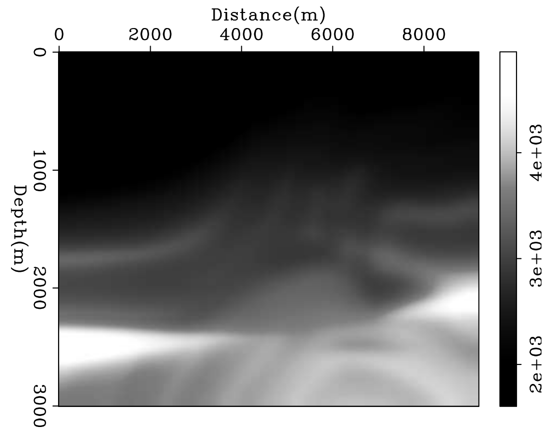


(c)

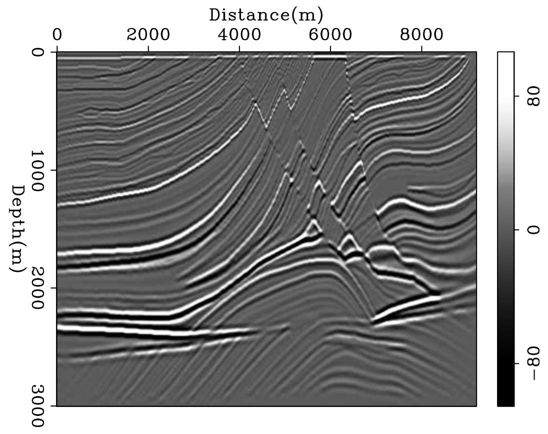


(d)

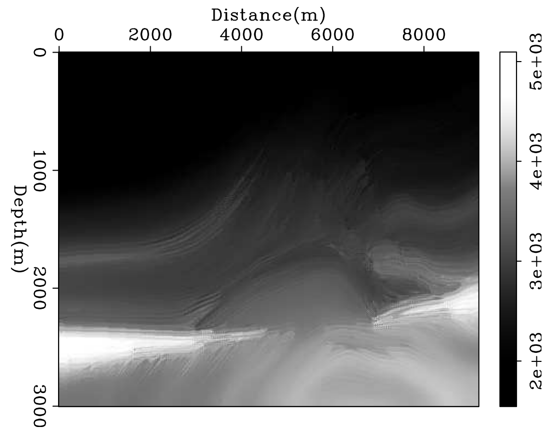
Figure 1: Data integration problem 1: The starting model is random noise **(a)** and the auxiliary data is the reflectivity of the Marmousi model **(b)**. Panel **(c)** shows the estimated model with the cross-gradient function and panel **(d)** shows the estimated model with the dip residual. Note the reconstruction of the structure in the estimated models and how each method provides different quality of reconstruction. [**ER**] mohammad2/. noise,ref,nxrx,nxrp



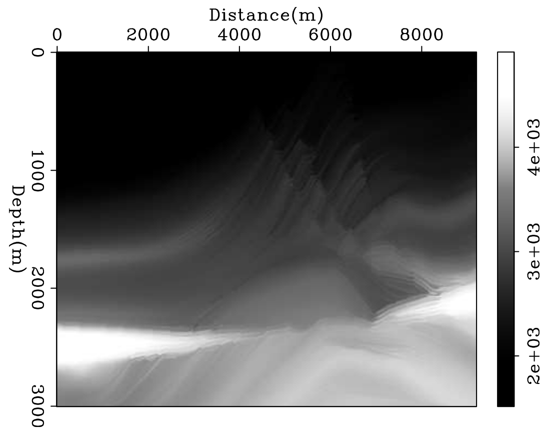
(a)



(b)



(c)



(d)

Figure 2: Data integration problem 2: The starting model is now changed to a smooth version of velocity **(a)** and the auxiliary data is the reflectivity of the Marmousi model **(b)** . Next panels show the estimated model with **(c)** the cross-gradient function and **(d)** dip residual. Note that we only expect to reconstruct the structure of the model, and the formulation of our optimization problems does not include the physics behind the wave propagation. [**ER**] $\boxed{\text{mohammad2/. vs,ref,mxrx,mxrp}}$

reflectivity or resistivity as auxiliary data and I choose either the cross-gradient function or the dip residual for the regularization operator. This leads to four different updates based on the choice of auxiliary data and data integration method. Note the different frequency contents of the two types of auxiliary data in Figure 3. Updated velocities with cross-gradient functions (Figure 4(a) and 4(b)) seem to have more contribution of structure (model-shaping term) than the physics (data misfit term) of the problem, as the structure of model is clearly visible in the results. The dip residual method, however, shows a better portion of physics in the results and less of the actual structure. This suggests that we may be able to benefit from combining these methods in a specific fashion to obtain a better velocity estimation.

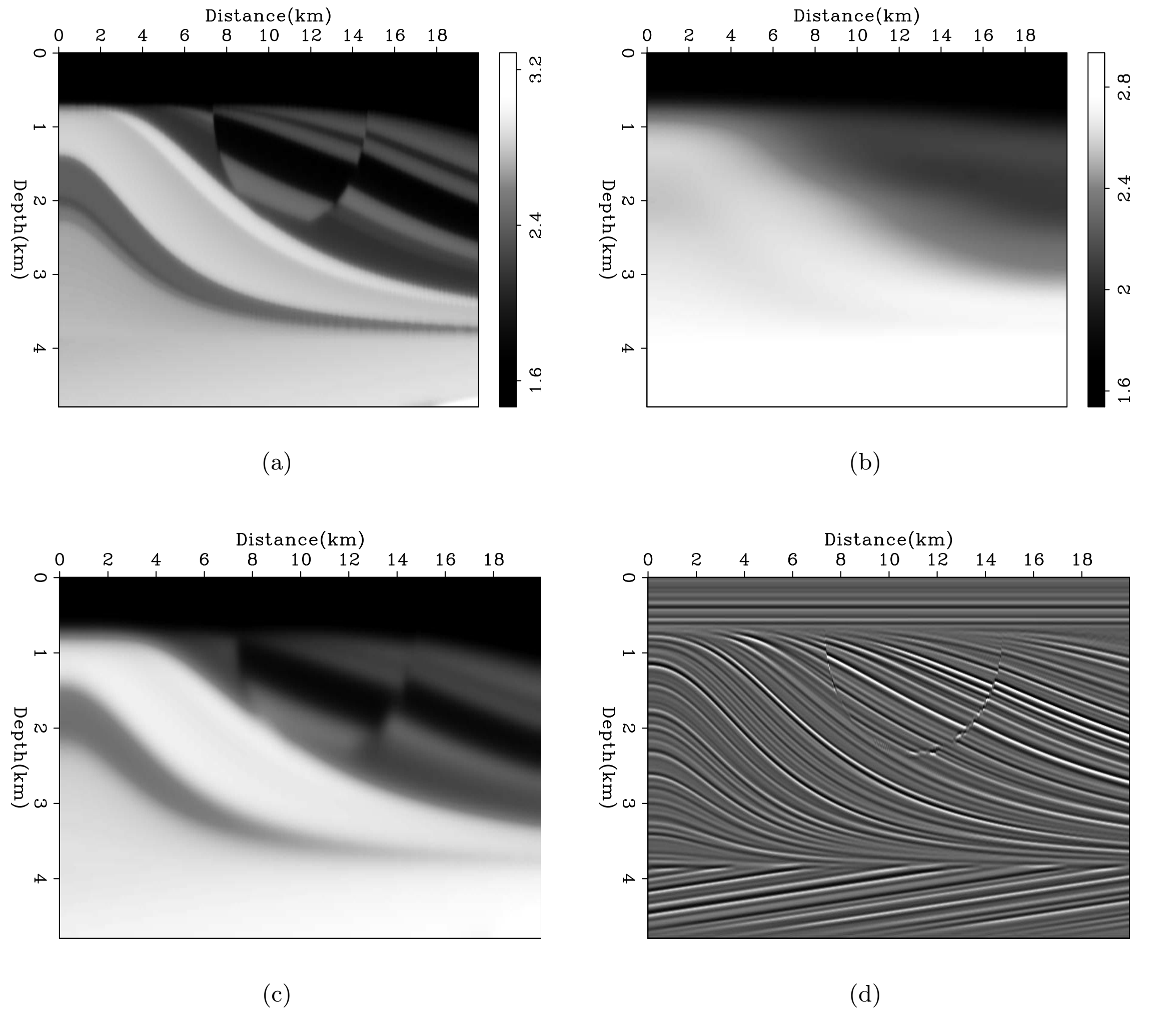
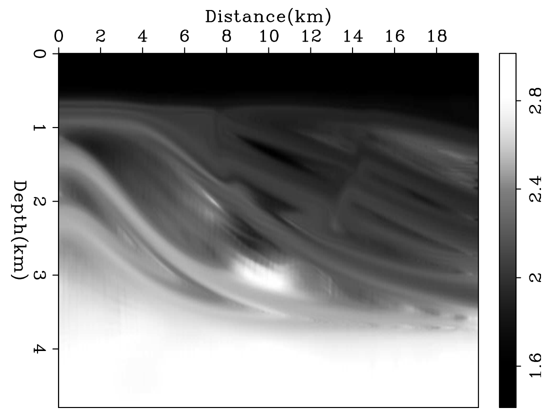
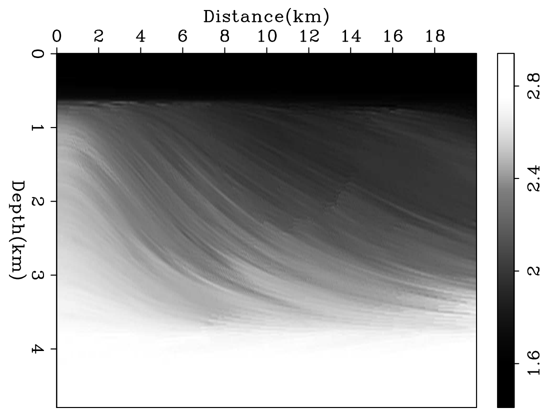


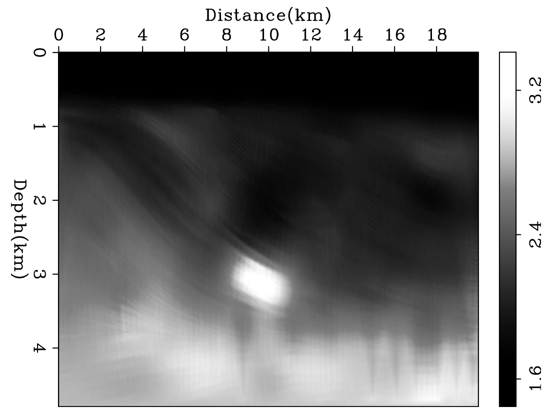
Figure 3: Synthetic 2-D model used as an example for comparison of different data integration methods: **(a)** True velocity, **(b)** initial guess for velocity, **(c)** approximation of resistivity, and **(d)** true reflectivity. **[ER]** mohammad2/. vel-t,vel-0,softdata1-0,softdata2-0



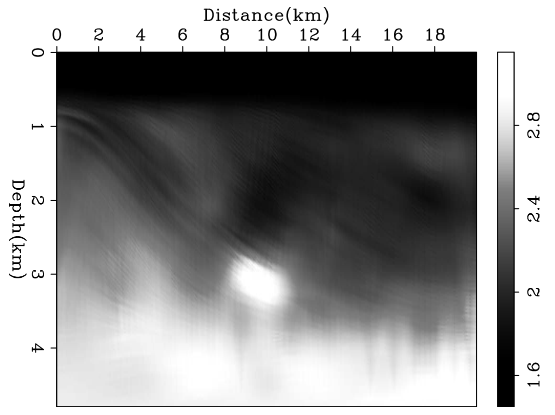
(a)



(b)



(c)



(d)

Figure 4: Updated velocities for the model shown in Figure 3 obtained by solving the tomography problem using **(a)** resistivity and the cross-gradient function, **(b)** reflectivity and the cross-gradient function **(c)**, resistivity and the dip residual, and **(d)** reflectivity and dip residual. [**CR**] mohammad2/. velx1-ds0,velx2-ds0,velp1-ds0,velp2-ds0

CONCLUSIONS

As mentioned above, geophysical data integration can greatly affect the quality of estimation of attributes by reducing the uncertainty in the model. In this paper, I reviewed the structural similarity and some measurement techniques. The efficiency of these techniques was compared with a simple example, where the physics were not taken into account to emphasize the effect of data-integration techniques. The results suggest that the dip-residual method provides better continuity in estimated structures than does the cross-gradient function. Model estimation results using cross-gradient function shows higher sensitivity, which can be explained by its differentiating nature.

I also compared these methods on a velocity estimation problem, where the physics are also included in the problem statement. I used two different types of auxiliary data with different frequency content to compare the efficiency of the techniques reviewed above. In this case, the cross-gradient function shows a stronger structure-imposing effect. However, the cross-gradient functions are not guaranteed to produce improved results when the frequency contents of the main data field and the auxiliary data are very different. The dip-residual method integrates less structural information by missing some of the anomalies. More examples are suggested to validate the comparison results.

REFERENCES

Bosch, M., P. Barton, S. C. Singh, and I. Trinks, 2005, Inversion of traveltime data under a statistical model for seismic velocities and layer interface: Geophysics, **70**, R33 – R43.

Colombo, D. and M. De Stefano, 2007, Geophysical modeling via simultaneous joint inversion of seismic, gravity, and electromagnetic data: Application to prestack depth imaging: The Leading Edge, **26**, 326–331.

de Nardis, R., E. Carderelli, and M. Dobroka, 2005, Quasi-2d hybrid joint inversion of seismic and geoelectric data: Geophysical Prospecting, **53**, 705–716.

Fomel, S., 2000, Applications of plane-wave destructor filters: SEP-Report, **105**, 1–26.

Gallardo, L. A. and M. A. Meju, 2004, Joint two-dimensional DC resistivity and seismic travel time inversion with cross-gradients constraints: J. Geophys. Res., **109**.

Maysami, M., 2008, Migration velocity analysis with cross-gradient constraint: SEP-Report, **136**, 55–64.

Maysami, M. and R. G. Clapp, 2009, Seismic tomography with co-located soft data: SEP-Report, **138**, 55–60.

Selecting the right hardware for Reverse Time Migration

Robert G. Clapp, Haohuan Fu, and Olav Lindtjorn

ABSTRACT

The optimal computational platform for Reverse Time Migration (RTM) has recently become a topic of significant debate, with proponents of the Central Processing Unit (CPU), General Purpose Graphics Processing Unit (GPGPU), and Field Programmable Gate Arrays (FPGA) all claiming superiority. The difficulty of comparing these three platforms for RTM performance is that the underlying architecture leads to significantly different algorithmic approaches. The flexibility of the CPU allows for significant algorithmic changes, which can lead to more than an order of magnitude improvement in performance. The GPGPU's large number of computational threads and overall memory bandwidth provide a significant uplift but require a simpler algorithmic approach, requiring more computation for the same size problem. The FPGA's streaming programming model results in an attractive but different cost metric. The current lack of a standardized high-level language is problematic.

INTRODUCTION

Over the last couple of years, it has become more difficult to assess which hardware can deliver the most cost-optimal solution for demanding imaging tasks. The days of faster and faster CPUs are over. A clear choice of hardware has been replaced with many-core technologies, and a proliferation of alternatives. In particular, accelerators like GP-GPU and Field Programmable Gate Arrays (FPGAs) have emerged as strong contenders for the title of hardware platform of choice. With the radical difference in hardware architectures, it has also become more and more difficult to evaluate which platform is optimal for the application in question. An apples-to-apples comparison is no longer possible. Through the example of RTM, we demonstrate that only through a careful optimization for each platform, with the involvement of Hardware, Computer-Science and algorithmic scientists, can we come up with a reasonable assessment of the alternatives available today.

At the core of the RTM algorithm is a modeling kernel. The simplicity of the modeling kernel has led to high-performance implementation on Field Programmable Gate Arrays (FPGA) (Nemeth et al., 2008), General Processing Graphics Processing Units (GPGPU) (Micikevicius, 2008), and conventional processors. There has been significant debate on which platform produces the most efficient code, both in terms of runtime and when code maintainability is factored in. One of the largest difficulties in comparing these three platforms is that while a naive, *unoptimized* RTM algorithm could be implemented on each platform and directly compared, an *optimized* version of the RTM algorithm varies significantly due to the strengths and weaknesses of each architecture.

In this paper, we discuss some of the algorithmic and optimization decisions used to implement RTM on each platform. In addition, we show how these choices are often directly tied to the characteristics of the underlying architecture. We begin by describing the basic RTM algorithm. We then discuss some of the strengths and weaknesses of each platform for RTM and the various algorithmic approaches and optimization techniques for implementing RTM on each platform. We conclude with a discussion about what generalities can be extracted from three different implementation approaches and at what level the implementations are architecture-specific.

RTM

The concept behind RTM is relatively simple. We start with a known earth model. This earth model might be simply acoustic velocity but can be anisotropic, elastic, or even visco-elastic. Two different modeling experiments are conducted simultaneously through the earth model. Both attempt to simulate the seismic experiment conducted in the field, one from the perspective of the source and one from the perspective of the receivers. The source experiment involves injecting our estimated source wavelet into the earth and propagating it from $t = 0$ to our maximum recording time $t_{\max}$, creating a 4-D source field $s(x, y, z, t)$. At the same time, we conduct the receiver experiment. We inject and propagate our recorded data starting from $t_{\max}$ to t_0, creating a similar four dimensional volume $g(x, y, z, t)$. The most common approach is to propagate these fields using an explicit time marching scheme (Dablain, 1986). We start from the acoustic wave equation

$$\frac{\partial^2 u}{\partial t^2} = \mathbf{v^2}\left(\frac{\partial^2}{\partial \mathbf{x^2}} + \frac{\partial^2}{\partial \mathbf{y^2}} + \frac{\partial^2}{\partial \mathbf{z^2}}\right), \tag{1}$$

where u is pressure and $\mathbf{v}$ is velocity. We can use a Taylor expansion to approximate these derivatives. Algorithm 2 shows the pseudo-code for how to forward propagate by nt steps a source function w, with a dt interval on a regular mesh whose size is nx, ny, nz indexed by ix, iy, iz, it, using a second-order approximation of the time and space derivatives.

Algorithm 2 Second-order acoustic modeling

```
for it=3...nt do
  for ix=2...nx-1 do
    for iy=2...ny-1 do
      for iz=2...nz-1 do
        s(ix,iy,iz,it)= 2*s(ix,iy,iz,it-1)-s(ix,iy,iz,it-2)+
        v(ix,iy,iz)*dt*dt*(s(ix,iy,iz,it-1)*8+
        -(s(ix-1,iy,iz,it-1)+s(ix+1,iy,iz,it-1))/dx/dx
        -(s(ix,iy-1,iz,it-1)+s(ix,iy+1,iz,it-1))/dy/dy
        -(s(ix,iy,iz-1,it-1)+s(ix,iy,iz+1,it-1))/dy/dz)+w(ix,iy,iz,it)
      end for
    end for
  end for
end for
```

We have a reflection where the energy propagated from the source and the receiver are located at the same position at the same time. The final image i is the summation of correlating the source and receiver wavefield at every time and every shot,

$$i(x, y, z) = \sum_{\text{shots}} \sum_{t=0}^{t_{\max}} s(g, x, y, z, t) g(x, y, z, t). \tag{2}$$

Bottlenecks

There are four potential bottlenecks in this process. The first two involve the cost of propagating the wavefield from $t = 0..t_{\max}$. The explicit time marching scheme described above has an operation count proportional to the number time steps nt times the size of the domain $nx * ny * nz$. In order to obtain a stable and non-dispersive solution, we are constrained in choosing our sampling in time and space. If we use too large of time steps or sample our medium too coarsely for a given velocity or frequency in our data, we can run into problems with stability, dispersion, and/or accuracy. Our sampling is controlled by the minimum and maximum velocity in the media, the maximum frequency we want propagated, and the order of accuracy of the derivative approximations. By using higher approximations to the space and time derivatives, we can get away with coarser sampling in time and space with the trade-off of a larger number of operations per sample. This is generally a beneficial trade-off, particularly for the spatial derivatives, because we trade off linear growth in our derivative approximation with cubic growth in the number of samples.

An unmentioned aspect of the time marching scheme is that our physical experiment takes place in a half-space while our computational experiment happens in a more limited domain. As a result the computational experiment has artificial reflections from the boundaries of the domain. The reduction of these effects is usually handled through one or more of the following techniques: introducing a damping region around the computation domain (Cerjan et al., 1985); introducing a boundary condition that attempts to kill energy propagating at some limited range of angles perpendicular to the boundary (Clayton and Engquist, 1980); or, most effective, costly, and complex, using Perfectly Matched Boundary (PML) (Berenger, 1996) which simulates propagating a complex wave whose energy decays as it approaches the boundary.

A third problem is that it is impractical to store the 4-D volumes s and g in memory. These volumes are often multiple terabytes in size. As a result, for a second-order time approximation, we normally keep in memory only the previous two time steps needed to update the wavefield. This causes a problem with our imaging condition where the source field must propagated from $t = 0.$ to $t =$ maxtime while the receiver wavefield must be propagated from $t =$ maxtime to $t = 0$ yet the fields must be correlated at equivalent time positions. To solve this problem, one propagation must be stored either completely or in a check-pointed manner to disk. Symes (2007) and Dussaud et al. (2008) discuss checkpointing methods to handle the different propagation directions.

A fourth potential problem is the very large size of the image domain required when constructing subsurface offset gathers. This is a particular problem for some accelerators with limited memory.

Modeling costs

From the above list of bottlenecks, one could conclude that optimization and algorithmic techniques are concentrated on reducing the total number of operations and figuring out how to minimize the need to go to disk. This is an incomplete and, particularly for reducing operation count, a misleading description of the problem. Reducing the total number of operations is important but in most cases the problem is the ‘cost’ of transferring data from where it exists on some memory device to the processing unit. The cost of transferring the data has two parts; the latency, which is the amount of time it takes to send the first byte; and the bandwidth, which is how much data can be sent in a given time. The definition of a memory device has many potential meanings. On the latest Intel CPU (Nehalem) these include L1, L2, and L3 cache; the Random Access Memory (RAM) on the system; the disk, solid state or conventional; and transferring from another processor, or node. Accelerators such as the GPU or FPGA have additional levels of memory devices. The core of many optimizations and algorithmic techniques is to get as much of the data as possible to the least costly memory device before the processor needs it.

To understand how latency and bandwidth can play an important role in seismic modeling, let's take a closer look at the general structure of algorithm 2 with a simplistic view of a conventional CPU, where vectorization, prefetching, and pipelining don't exist. To compute $s(2,2,2,3)$ we need one element in each of the 3-d volumes $s(:,:,:,1), w$ and v; and seven elements in $s(:,:,:,2)$. Let's ignore the source term w, as it is non-zero at very few values. So, to compute the very first element of the array, we need to read 9 values (36 bytes) and store 4 bytes. In reality, we are going to read significantly more than 36 bytes. CPUs don't transfer individual bytes from main memory but instead transfer cache lines which are composed of between 8-256 contiguous bytes (we will use 64 bytes for this example) to L1 cache. The individual value we want is then transferred to a register for our given calculation. So, for this idealized case, we will have the cost of transferring the following eight cache lines to the L1 cache.

Cache line	Usage
$v(1:16,2,2)$	Velocity
$s(1:16,2,2,1)$	Used for time derivative
$s(1:16,2,2,2)$	Used for time and z derivative
$s(1:16,1,2,2)$	Used for y derivative
$s(1:16,3,2,2)$	Used for y derivative
$s(1:16,2,3,2)$	Used for x derivative
$s(1:16,2,1,2)$	Used for x derivative
$s(1:16,2,2,3)$	Output

The cost will be dominated by the latency associated with transferring eight cache lines from main memory. Everything we need to compute the next 15 elements, $s(3,2,2,3)$, is now sitting on our L1 cache, which tends to have a latency an order of magnitude less than transferring from main memory. When we hit $s(17,2,1,3)$ we will have to transfer additional cache lines from main memory to the L1 cache. We will continue in this pattern along the iz axis. At some stage, the L1 cache becomes full an cache lines will instead be sent to the L2 cache. When the L2 cache is full, cache lines will be transferred to the L3 cache, then main memory. Things get more interesting when we begin the $iy = 3$ loop. We will definitely have to read in the new cache lines $v(1:16,3,2)$, $s(1:16,3,2,1)$, $s(1:16,4,2,2)$, $s(1:16,3,1,2)$ and $s(1:16,3,3,2)$ but the remaining three cache lines needed for the calculation have been read before and may exist in L1, L2, L3, or main memory depending on the size of our domain. The larger our domain, the more likely that the cache lines will have migrated to main memory, costing us the order of magnitude additional penalty. When we begin the $ix = 3$ loop only three of the cache lines needed ($v(1:16,2,3)$, $s(1:16,2,3,1)$, and $s(1:16,2,4,2)$) haven't been previously read. For higher order spatial derivatives, the problem is significantly worse. For example, in the case of a 14th order in space stencil, we need to read 46 different cache lines. The odds of reuse go down significantly, increasing our cache misses and straining the total bandwidth needed between our caches and main memory.

CPU

Of the three potential architectures described in this paper, the CPU is by far the most flexible and offers the most potential optimization and algorithmic manipulation paths.

Following the wavefield

The computational domain used for RTM is usually controlled by the desired migration aperture. The source function w from algorithm 2 is non-zero in a very small portion of the computational domain. When propagating a source, we might have as few as 9 non-zero elements in a mesh that is a billion points in size. As a result, at early times, algorithm 2 would spend over 99% of its efforts propagating zeros. As time progresses, the wavefield will be non-zero at a larger percentage of the volume. As Figure 1 shows, we can slowly expand the range of ix, iy and iz, reducing the total number of operations.

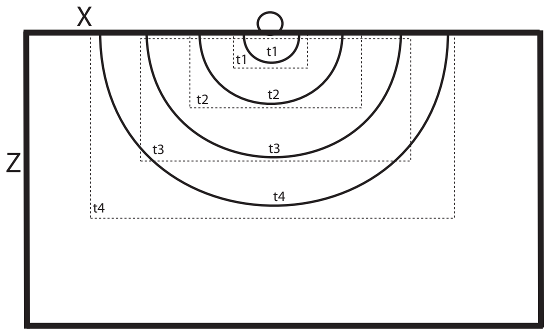


Figure 1: An illustion of how we can restrict the size of computational domain at early times. The circle represents a source at the surface and the wavefronts are drawn different time locations (t1,t2,t3). The domain where we update our wavefield can safely be restricted to the corresponding boxes.[**NR**] $\boxed{\text{bob1/. range}}$

Domain decomposition and resampling

Domain decomposition involves splitting our computation domain into several sub-domains. Figure 2 shows how each sub-domain is then marched forward in time (one or more steps) separately. Domain decomposition can be done at the core, processor, or node level. Different computational units are assigned different portions of the domain. They are responsible for computing an updated solution in their region.

The primary advantage of this approach is that each sub-domain is smaller, increasing the likelihood that the cache line will be close (in L1 or L2 cache) rather flushed to main memory. A more subtle advantage of domain decomposition is that sub-domains can represent significantly different velocity ranges, and therefore different spatial sampling requirements to obtain the same level of accuracy, stability, and dispersion. These modifications can potentially allow significant performance gains.

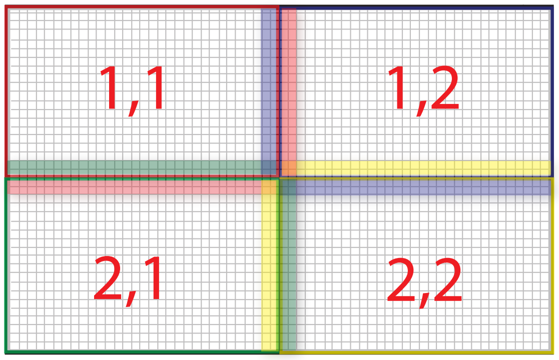


Figure 2: An example domain decomposition. The domain is broken into four portions reducing the amount of memory needed and/or reducing cache misses improving performance. The domains can not be treated completely indepently because the rely on retrieving information from the neighbors as the time marching scheme progresses.[**NR**] bob1/. decomposition

Naturally, there are also disadvantages. First, notice the shaded squares in Figure 2 that demonstrate that a sub-domain must access some of its neighbors' elements in order to calculate the spatial derivative along its own edge. As a result, there is a limit on how independently each sub-domain can operate and how small the sub-domains can become before the cost of rereading or passing the boundary region becomes the dominant cost. In addition, transferring this boundary region from one computational unit to another can become a bottleneck. Calculating derivatives at the boundaries and potentially handling load balancing can make the code considerably more complex.

Property compression

Usually we store our velocity, density, and anisotropic parameters as 4 byte floats. This is far more precision than is warranted by our knowledge of velocity, let alone our anisotropic density parameters. All of our property descriptors can be compressed, increasing the number of output points that can be computed per cache line read and the amount of space the storing property description takes up on the L1, L2, and L3 cache. For example, storing velocity as a short (2 bytes) rather than float would double the number of output points that could be calculated per cache line read.

Cache-oblivious approaches

Cache optimization techniques, such as finding the optimal sub-domain for domain decomposition, are very CPU-specific. Cache-oblivious approaches attempt to find a solution that is independent of cache size, resulting in a minimum number of cache misses. These approaches generally use a divide and conquer scheme. Imagine the 1-D version of algorithm 2 and say we want to calculate the 10th position at x at the 5th time sample $s(10,5)$. Ignoring velocity for a minute, we need $s(9:11,4)$ and $s(10,3)$. To obtain $s(9:11,4)$, we need $s(8:12,3)$ and $s(9:11,2)$. To obtain $s(8:12,3)$ we need $s(7:13,2)$ and $s(8:12,1)$. This concept is demonstrated in Figure 3. In addition, for all of these calculations, we will need $v(8:12)$. If we then do the progression of calculating $s(8:12,3)$, the L1 cache will have everything we need to update $s(9:11,4)$ and $s(10,5)$ without suffering a single cache miss.

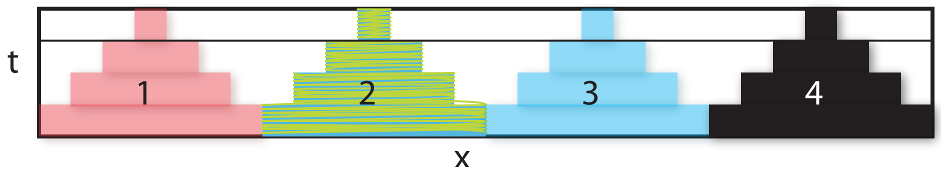


Figure 3: The concept behind oblivious cache. By reading all of the values at the bottom at the bottom of each pyramid the solution can be updated at all the location at the second level of the pyramid. The second level provides all that is needed to update the third level. Finally all the elements on the third level provide everything that is need to create the element at the top of the pyramid. This technique minimizes cache misses by reformulating the algorithm to maximize data reuse.[**NR**] bob1/. oblivious

GPGPU

GPGPUs are a rather new high performance computing platform. Current high-end GPGPUs have hundreds of processing units and total memory bandwidth that is several times greater than Intel and AMD's current architectures. The hardware roadmap for GPGPUs versus CPUs show this disparity increasing significantly in the near future. GPGPUs connect to the motherboard through either the PCI-X slot on the motherboard or directly into the CPU socket on the motherboard. The PCI-X approach has the advantage of being motherboard agnostic. Connecting directly into the CPU socket allows easier, and often quicker, access to system memory, but takes away from the system's total CPU computing power. The most common programming language for GPGPUs is Nvidia's CUDA. An open standard language, OpenCL, will soon be available on both Nvidia and ATI hardware.

GPGPUs are often referred to as "streaming processors". In this context, a streaming paradigm means that the same operation, or set of operations, is performed on a stream of data, a form of Single Instruction Multiple Data (SIMD). A GPGPU is broken into a grid of thread blocks. Ideally, the number of thread blocks is equal to greater than the number of processing units on the GPGPU. Multiple threads, up to 512 on the latest Nvidia GPGPU, are started on each grid block. The threads help hide memory latency. While one thread is waiting for data, another thread can be executing. Each thread block has its own set of registers (fastest memory) and each grid block has a pool of shared memory, which can be as fast to access as register. In addition, there is a large pool of global memory (4GB on the latest Nvidia GPGPU). This global memory has latency two orders of magnitude larger than accessing registers or shared memory. Figure 4 shows this two level parallelism.

On the GPGPU, the basic acoustic algorithm shown in algorithm 2 is broken into two parts. Replacing the unrealistic 4-D source volume with the previous `prev`, current `pcur`, and updated `pnext` wavefield, the CPU portion of the code has a form similar to algorithm 3.

The GPGPU update kernel has nearly as simple a form.

GPGPUs are often considered difficult to program. This is partially due to unfamiliarity with the parallelism model and partially due to the need to explicitly handle transfers to and from the GPGPU and at different levels of GPGPU memory in order to achieve maximum

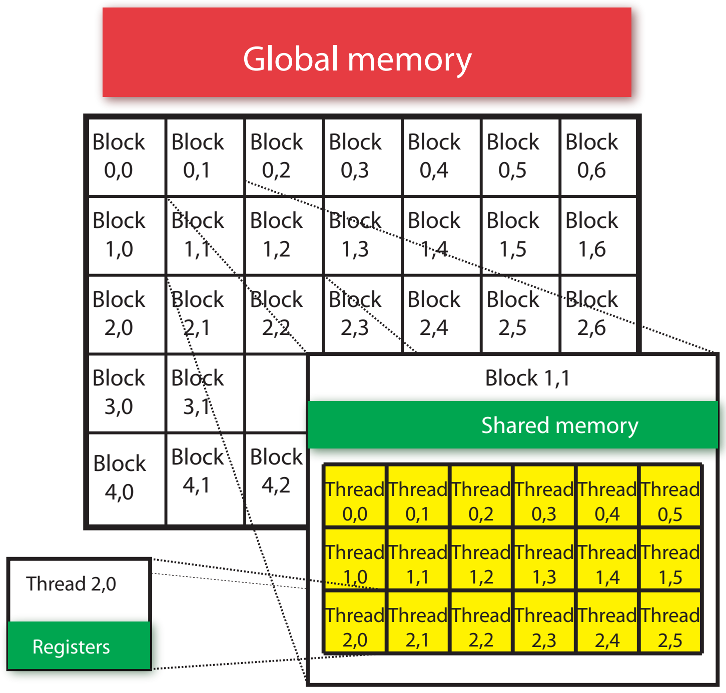


Figure 4: The memory hierarchy of typical GPGPU. The processing elements are broken into a 2-D grid block. Each 2-D grid block is broken into a grid of thread blocks. Each thread has its own set of registers. Each grid block can access the same shared memory and grid blocks can access the globabl memory.[**NR**] bob1/. gpgpu

Algorithm 3 Second-order acoustic modeling

```
Allocate and transfer velocity to GPU
Allocate and zero prev, pcur, pnext
Allocate and transfer source function locations and data
Create GPGPU grid blocks
Create GPGPU thread blocks
for it=3...nt do
  Call GPGPU kernel to inject wavefield into pcur
  Call GPGPU kernel to update wavefield
  Switch GPGPU pointers on prev, pcur, and pnext
end for
```

Algorithm 4 Second-order acoustic modeling

```
Figure out location based on grid and thread number
for ix=2,nx-2 do
  Transfer pcur values to shared memory
  Apply space and time derivatives to form pnew
end for
```

performance. For an unoptimized code, this perspective has validity. An optimized GPGPU code most likely contains less lines and less complexity. Once the relative simplicity of the GPGPU kernel parallelism and memory hierarchy is understood, one could argue that the GPGPU offers a quicker development cycle.

Simply looking at the total memory bandwidth available and the raw compute power offered, GPUs would seem to be the obvious best compute device for RTM. Unfortunately, some of the limitations on the programming model and memory hierarchy makes it less clear.

Programming model limitations

To achieve maximum GPGPU performance, we need to maximize the number of threads per thread block and have at least as many grid blocks as the number of processing units. On the high end GPGPU, this means we would like to have at least 100,000 identical tasks. Unfortunately, this eliminates some of the optimization opportunities available on the CPU. Oblivious cache is impractical and we are limited in our ability to compress model parameters. Both of these result in significantly more strains on global memory. A potentially greater problem is introduced with the boundary condition. The SIMD parallelism required for performance on the GPU makes complex boundary conditions like PML and probably the zero slope boundary condition impractical. As a result, we are limited to using damping schemes that require us to expand our computational domain to achieve damping results that are sub-optimal compared to PML schemes.

Memory limitations

As mentioned above, in the GPU hierarchy, registers and shared memory provide two orders of magnitude faster access than the global memory and function in the role of cache in CPU architectures. The size limits of registers and shared memory per block are both 16 KB. While the shared memory is shared among all the threads in the block, the threads keep their own copies of the registers. As a result, when we increase the number of threads in a block to achieve better performance, the size of registers usually becomes the bottleneck. In the CUDA framework, the supported maximum number of threads per block is 512. If the size of the stencil increases from 9 points to 33 points, the number of registers needed for each thread increases from 28 to 55. To stay under the maximum number of threads per block (512), the stencil sizes must be smaller than 17 points. For stencil sizes larger than that, we need to reduce the number of threads per block, thus sacrificing some of the computation performance.

Although global memory access is much slower than registers and shared memory, the bandwidth of global memory (80 GB/s) is much higher than the bandwidth to the host memory through PCI-Express (4 GB/s). However, the size of global memory is usually not enough for the large dimension sizes of seismic problems. Typically, we need to store four arrays (`prev, cur, next, v`), allowing a domain size of only 250 million points, much smaller than is often needed. For larger domains, we must split over multiple GPUs, creating a bottleneck either over the PCI-X link or the network.

The correlation step is also constrained by memory bandwidth. The fact that the source and receiver wavefields are propagated in different directions is also problematic. Assuming a check-pointing scheme is used, we must continually transfer snapshots of the source wavefield across the PCI-X bus when propagating the wavefield forward and transfer them back while propagating the receiver wavefield. Wavefield buffering that is usually used to minimize the number of checkpoints and the resulting IO load must be spaced much closer together given the relatively small amount of memory on the GPGPUs. Constructing subsurface offset gathers is also hindered by the limited memory. The GPGPU can only create offset gathers at a small percentage of imaging locations before we again become memory limited.

FPGAS

Similar to GPGPUs, FPGAs are usually classified as streaming computation devices. However, the actual 'streaming' mechanism differs significantly between GPGPUs and FPGAs. While a GPGPU generally issues loads of identical threads to handle different data items (Single Instruction Multiple Data), an FPGA pushes one data item through loads of different operations (Single Data Multiple Instruction). In a fully-pipelined FPGA design, you may even have different data items processed for different operations at different pipe stages (Multiple Data Multiple Instructions).

Before diving into the technical details of FPGA implementations, this section provides a general picture of the streaming architecture of FPGA computation compared to the classical Von Neumann architecture.

Figure 5 shows a classical Von Neumann architecture containing two major parts, the processing unit and the memory. The memory stores both the data and instructions. To execute each instruction, the processing unit loads the first instruction in the sequence and the related data from the memory, performs the corresponding arithmetic or logic operations, and stores the data back into the memory.

In contrast to the Von Neumann, FPGAs take a streaming approach to perform computation.

Figure 6 shows how as the data items are pushed in and out as sequential streams, the instructions are mapped into programmable circuit units along the path from the input ports to output ports. Therefore, instead of fetching instructions and data back and forth from the memory, the computation gets performed as the data streams flow through the circuit units in one pass.

The performance of an FPGA streaming design relates to two key factors: one is how fast we can clock the pipeline, i.e. the throughput of the FPGA circuit; the other one is how fast we can push the data in and out, i.e. the bandwidth between the FPGA and the outside data.

To achieve a high computation throughput, the FPGA designs are generally fully-pipelined, with registers storing intermediate computation results. With pipelining, the throughput of the entire design is no longer determined by the latency of entire pass. Instead, the throughput is determined by the largest latency among circuit parts separated by registers. Therefore, for current commercial FPGAs, as long as the FPGA has enough registers to break the large latency into small separated parts, the computation throughput of the design can always be optimized to a level around the FPGA frequency. The

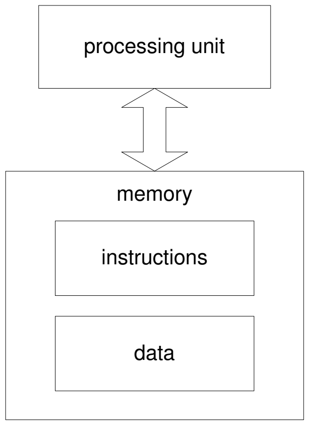


Figure 5: The classical Von Neumann architecture.[**NR**] bob1/. von-neumann

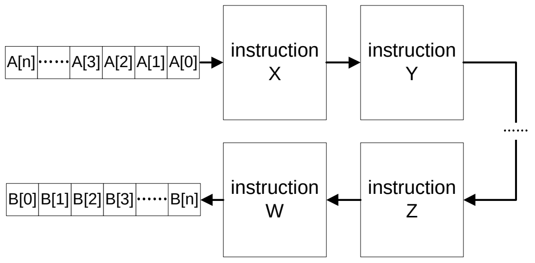

A[n] ······ A[3] A[2] A[1] A[0]
instruction X
instruction Y
instruction Z
instruction W
B[0] B[1] B[2] B[3] ······ B[n]

Figure 6: FPGA streaming architecture.[**NR**] bob1/. stream

pipelining in the design makes the FPGA's performance behavior quite different from conventional processors. In conventional processors, more mathematical operations consume more computation time. In FPGAs, more mathematical operations consume more hardware resources. While the latency from input to output port is increased, the throughput of the design remains the same. For applications like RTM that process a huge amount of data items, the long stream of input and output values hide the increase in latency.

Current commercial FPGAs contain three major categories of resources: reconfigurable logic; arithmetic units that can perform 18-bit by 25-bit multiplications; and Block RAMs that serve as distributed local storage units. On a current high end FPGA from Xilinx we can implement up to 1000 single-precision floating-point multipliers or 500 adders. Note that,if Moore's Law holds, these resources on the commercial FPGAs can get doubled every eighteen months.

Implementing RTM on FPGAs

The core computation of RTM is finite-difference-based 3D convolutions, which normally perform multiplications and additions on a number of adjacent points. While the points are neighbors to each other in a 3D geometric perspective, they are often stored relatively far apart in memory. For example, in the 7-point 2D convolution performed on a 2D array shown in Figure 7, data items $(0, 3)$ and $(1, 3)$ are neighbors in the y direction. However, suppose the array uses a row-major storage and has a row size of 512, the storage locations of $(0, 3)$ and $(1, 3)$ will be 512 items (one line) away. For a 3D array of the size $512 \times 512 \times 512$, the neighbors in the z direction will be 512×512 items (one slice) away. In software implementations, this memory storage pattern can incur a lot of cache misses when the domain gets larger, and decreases the efficiency of the computation.

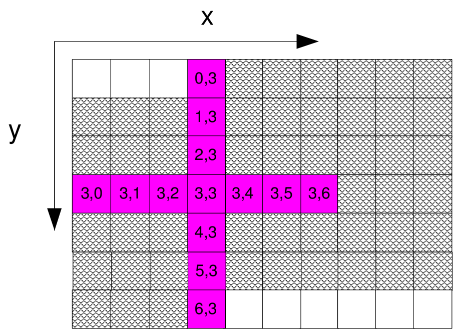


Figure 7: A streaming example of 2D convolution.[**NR**] $\boxed{\text{bob1/. convolution-2D}}$

In an FPGA implementation, we use BRAMs to store a window of the input stream and enable the circuit to compute one result per cycle. Figure 7 demonstrates this concept. Suppose we are applying the stencil on the data item $(3, 3)$, meaning the circuit requires 13 different values (shown in red). As the data items are streamed in one by one, in order to make the values from $(0, 3)$ to $(6, 3)$ available to the circuit, we use BRAMs to buffer the window of all the six lines of input values from $(0, 3)$ to $(6, 3)$ (shown in grey grid). In the next cycle, the first value $(0, 3)$ pops out of the window, as $(0, 3)$ is no longer needed for the following computation. Meanwhile, the next value $(6, 4)$ is pushed into the window buffer, as $(6, 4)$ is needed for the next stencil operation.

Considering the BRAMs as the 'cache' of an FPGA design, the above window buffer mechanism provides a perfect 'cache' behavior: firstly, the data item gets streamed into the window buffer at exactly the cycle that the data item is needed for the computation for the first time; secondly, the data item only resides in the window buffer for the period of time that the data item is needed for the computation. Current BRAMs can hold 6MB of data, or 1.5 million elements, when the data is stored as float. The window size that needs to be buffered is proportional to $n1*n2*no$, where no is the spatial derivative approximation. For large data volumes and high order stencils, this exceeds current BRAM storage, requiring domain decomposition. The size of BRAM tends to follow Moore's Law, doubling every 18 months.

Processing Multiple Time Steps

As mentioned above, current FPGAs have the resources to implement hundreds of floating-point arithmetic units. Therefore, one stencil operator design only consumes a small portion of the total resources of an FPGA. To make a full utilization of all the resources on an FPGA, one straightforward way is to fill multiple stencil operators into one FPGA.

However, increasing the number of stencil operators does not always improve the overall performance due to the constraint of the memory bandwidth between the FPGA and the onboard memories. When the input streams for the multiple stencil operators approach the saturation point of the memory bandwidth, increasing the number of stencil operators may not improve the performance any more.

Instead of implementing multiple stencil operators, another strategy is to extend the streaming pass to process multiple time steps. In RTM, the *dt* of modelling is a fraction of the sampling of the recorded data. These wavefields are intermediate results unneeded in the imaging process. The streaming architecture of FPGA provides an ideal platform for convolving multiple time steps in one pass.

As mentioned in the previous sections, as long as the FPGA has enough resources, we can add more computation units without reducing the throughput of the design. Therefore, instead of sending the results of the stencil operator to the output port, we can push the results into another stencil operator and perform the computation for the next time step.

Figure 8 shows the general structure for processing multiple time steps in one pass. As mentioned in previous section, at the input port, a window buffer is used to store all the data items needed for the stencil operator. Similarly, we can put another window buffer to store the convolution results of time step 0 and feed the corresponding values into the stencil operator for time step 1. Therefore, we can multiply the computation performance without consuming multiple times of memory bandwidth.

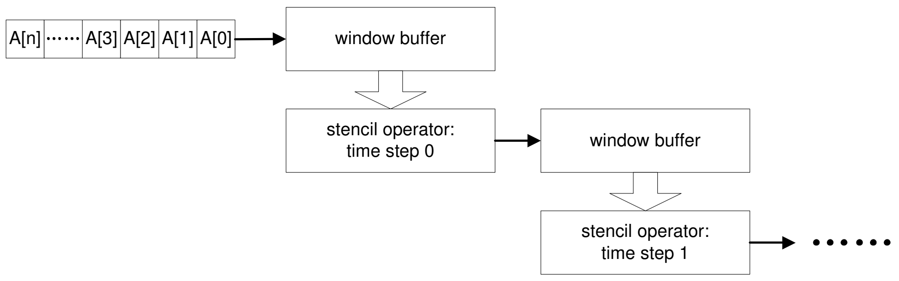


Figure 8: General structure for processing multiple time steps in one pass.[**NR**] bob1/. multiple-time-step

Bit-width Minimization

Conventional seismic processing is performed on a CPU with 32 or 64-bit precision for all operations. In certain cases, using a reduced precision produces equivalent results within acceptable tolerances. However, as CPUs do not support configurable bit-widths, reducing precision brings no benefits to performance. In contrast, FPGAs enable application-specific number representations. With hardware support for reconfigurable number format and bit-width, reduced precision can greatly decrease the area cost and I/O bandwidth of the design, thus multiplying the performance with concurrent processing cores on an FPGA.

In our previous work on exploring different bit widths in complex exponential operations Fu et al. (2008), we show that in certain parts of the calculation, fixed-point numbers with 12 bits can be applied to achieve a similar accuracy as 32-bit floating-point numbers. Similarly, for the 3D convolution in RTM algorithm, we can achieve acceptable accuracy using 24-bit fixed-point numbers instead of 32-bit floating-point numbers. The reduction in representation precision leads to significant reduction in the area cost of arithmetic operations and the bandwidth requirement between the FPGA and the outside data memory. As a result, the performance of the design gets significantly improved.

Data Compression

In contrast to CPUs, implementing decompression and compression on streaming architectures only consumes extra hardware resources. The computation performance does not get affected, as the fully-pipelined circuit would provide the same throughput. Moreover, as the performance of the design is in most cases constrained by how fast we can push the data in and out, i.e. the memory bandwidth, adding data decompression and compressing stages enables us to push more data through the processing circuits and improve the performance of the design.

For typical seismic applications, by utilizing the similar value range of neighboring data items, both the wavefield and velocity can be compressed by a significant ratio. For wavefield and the velocity data represented in 32-bit single-precision floating-point numbers, two times compression is usually achievable by converting the values into 16-bit fixed-point representation, and the performance of the design can be doubled for the cases in which memory bandwidth is the bottleneck.

Limitations

Streaming is not suited for all applications or even all portions of an application. Not all of the algorithmic tweaks that make RTM run efficiently on a CPU are feasible. Some algorithmic complexities, such as the PML boundary condition and variable grids, eat up too many FGPA resources to be viable. The number of arithmetic units is finite. Approaches such as multiple time steps per pass can oversaturate these resources. In addition, while wavefield precision and compression can significantly improve performance, their viability when using the RTM kernel for waveform inversion needs to be investigated.

Programming Difficulty

As mentioned in the above sections, the streaming architecture of FPGAs show great potential for achieving high performance for RTM. However, the difficulty in programming FPGAs has long been considered a disadvantage that prevents FPGA from becoming a general computation solution.

Firstly, the mindset for programming an FPGA is much different from the mindset of software programming. In contrast to writing a sequence of logic and mathematic operations in software programming, programming an FPGA is more like writing a description of a hardware design. The programmer needs to specify the input and output variables of each unit as well as the timing schedule between different signals. The different mindsets bring significant difficulties for common users, who are usually more into the software programming mindset, to map their designs into the FPGA's streaming architecture.

Secondly, the FPGA programming process itself is also much more complicated. In early days, FPGA programmers used to write their design using VHDL or Verilog, which are very low level hardware description languages. In recent years, people have made various efforts to develop high-level compilers for FPGA programming. Some of the emerging high-level compilers manage to hide most of the low-level details of hardware design, and enable the users to describe the hardware design using a high-level language, such as C++/Java.

```
void stencil_operator(input, coeff, output){

   output = coeff[0] * input +
            coeff[1] * (prev(input, 1) + after(input, 1) +
                        prev(input, nx) + after(input, nx) +
                        prev(input, nx*ny) + after(input, nx*ny)) +
            coeff[2] * (prev(input, 2) + after(input, 2) +
                        prev(input, 2*nx) + after(input, 2*nx) +
                        prev(input, 2*nx*ny) + after(input, 2*nx*ny)) +
            ......
}
```

The above shows an example of a high-level description for a 3D stencil operator. In this high-level description, the arithmetic operations are described using straightforward arithmetic expressions. The complicated window buffer mechanism (as described in previous sections) is described using the offset functions 'prev' and 'after'. In this piece of pseudo code, 'prev' and 'after' functions simply return the value of the variable in previous or afterwards cycles. By specifying the offset as a certain number of data items, data rows or even data slices, the compiler can automatically generate a buffer that covers all the data items needed for the current stencil operator. Therefore, the programming difficulty is significantly reduced.

DISCUSSION

Writing a code that uses the same data access pattern for all three architectures would lead to a far-from-optimal code. As a result, an unbiased comparison is difficult. Any one of the three can be declared the best or worst depending on what constraints you choose to put on your test. If you have a very large domain or a large stencil, the GPGPU isn't competitive. If you don't have a large domain requirement and want the simplest to understand and maintain optimized code, the GPGPU is probably the winner. The CPU has the advantage of being omni-present, and all but cache-specific code portions are virtually guaranteed to perform in the future, making it an attractive alternative. On the other hand, the raw compute advantage of GPGPUs and FPGAs looks to be growing over the next few years. FPGAs have the raw compute power to rival the GPGPUs and don't suffer from the size limitations of GPGPUs, but use a significantly different programming model whose abstraction level is still in flux. An abstraction layer that leaves the data movement and storage mechanism to a precompiler while limiting itself describing the mathematical update procedure, similar to the FPGA example shown above, holds some potential.

CONCLUSIONS

The variations in approaches for implementing an optimized version of RTM on CPU, GPGPU, and FPGA archictectures make direct comparison infeasible. Finding the platform that provides the optimal RTM performance involves balancing price, compute and memory potential, programmability, and the set of algorithmic approaches that are feasible on a given platform. Understanding the variation in underlying architecture and the resulting optimal programming approach can lead to new insights on how to produce optimal code.

ACKNOWLEDGEMENTS

We would like to thank the Center for Computational Earth and Environmental Science and the Stanford Exploration Project for funding this research. In addition, we would like to thank Nvidia and Maxeler for their hardware donations and technical support.

REFERENCES

Berenger, J. P., 1996, Perfectly matched layer for the fdtd solution of wave-structure interaction problems: Antennas and Propagation, IEEE Transactions on, **44**, 110–117.

Cerjan, C., D. Kosloff, R. Kosloff, and M. Reshef, 1985, A nonreflecting boundary condition for discrete acoustic and elastic wave equations: Geophysics, **50**, 705–708.

Clayton, R. W. and B. Engquist, 1980, Absorbing boundary conditions for wave equations migration: Geophysics, **45**, 895–904.

Dablain, M. A., 1986, Modeling and imaging with the generalized screen algorithm: Geophysics, **51**, 54–66.

Dussaud, E., W. W. Symes, L. Lemaistre, P. Singer, B. Denel, and A. Cherrett, 2008, Computational strategies for reverse-time migration: 78th Annual Internat. Mtg., Soc. Expl. Geophys., Expanded Abstracts, SPMI 3.3.

Fu, H., W. Osborne, R. Clapp, and O. Pell, 2008, Accelerating seismic computations on fpgas: From the perspective of number representations: Presented at the .

Micikevicius, P., 2008, 3d finite difference computation on gpus using cuda: 2nd Workshop on General Purpose Processing on Graphics Processing Units, Expanded Abstracts, 79–84.

Nemeth, T., J. Stefani, W. Liu, R. Dimond, O. Pell, , and R. Ergas, 2008, An implementation of the acoustic wave equation on fpgas: 78th Ann. Internat. Meeting, Expanded Abstracts, 2874–2877, Soc. Expl. Geophys.

Symes, W. M., 2007, Reverse time migration with optimal checkpointing: Geophysics, **72**, SM213–SM221.

Stanford Exploration Project, Report SEP139, October 19, 2009

Reverse Time Migration of up and down going signal for ocean bottom data

Mandy Wong, Biondo L. Biondi, and Shuki Ronen

ABSTRACT

We present the results of reverse time migration (RTM) on ocean-bottom data as a precursor to applying reverse time migration and inversion of multi-component ocean-bottom data using the two-way acoustic wave-equation. We propose a joint-inversion scheme that constructively combines up- and down-going migration results and removes spurious artifacts in the final image. Reverse time migration of up-going data gives stronger reflector amplitude and weaker artifacts than migration of down-going data; however, due to mirror-imaging, the area of sub-surface illumination is narrower when using up-going energy. In addition, we observe that a new class of artifacts is present due to RTM injection of receiver wavefields with the ocean bottom geometry.

INTRODUCTION

Reverse time migration (RTM) is a method of choice for imaging complex structure due to its ability to image steep reflectors (Baysal et al., 1983; Whitmore, 1983; Etgen, 1986). Despite its computational cost and sensitivity to the background velocity model, RTM has evolved from poststack migration to prestack migration using single component data acquired with the traditional streamer geometry. However, conventional streamers acquisition has significant limitations. One limitation is that production and drilling rigs interfere with the survey geometry. Such obstructions cause large gaps in the coverage of streamer surveys. Undershooting such obstructions with a separate source vessel is an imperfect solution because of the varying offset and azimuth distribution of such undershoots. These limitations foster a growing demand for ocean bottom seismometers (OBS). OBS data acquisition is an alternate approach in which seismometers are placed on the seafloor and shots are fired at the ocean surface. Therefore, OBS surveys are less sensitive to structural obstacles. OBS have other advantages including: full azimuth coverage, multi-component data, and potentially improved repeatability. Full azimuth coverage improves imaging and demultiple processing under complex overburden. Multi-component data enables better demultiple processing, or even imaging with multiples. Moreover, recording of shear waves contains additional information about the lithology and fractures and are less sensitive than P waves to gas clouds. These advantages often justify the additional cost of acquiring OBS data.

The ultimate goal of our research is to develop a better way to image OBS data and to refine the art of processing OBS data. The most significant difference is that OBS data are multi-components: pressure P, and the three components, X, Y, Z, of the particle velocity or acceleration. Figure 1 shows a common receiver gather of a single shot line in a typical ocean bottom acquisition. From Figure 1, we can see that both pressure (P) and shear (S) waves are recorded. A natural approach is to use elastic modeling in the migration process.

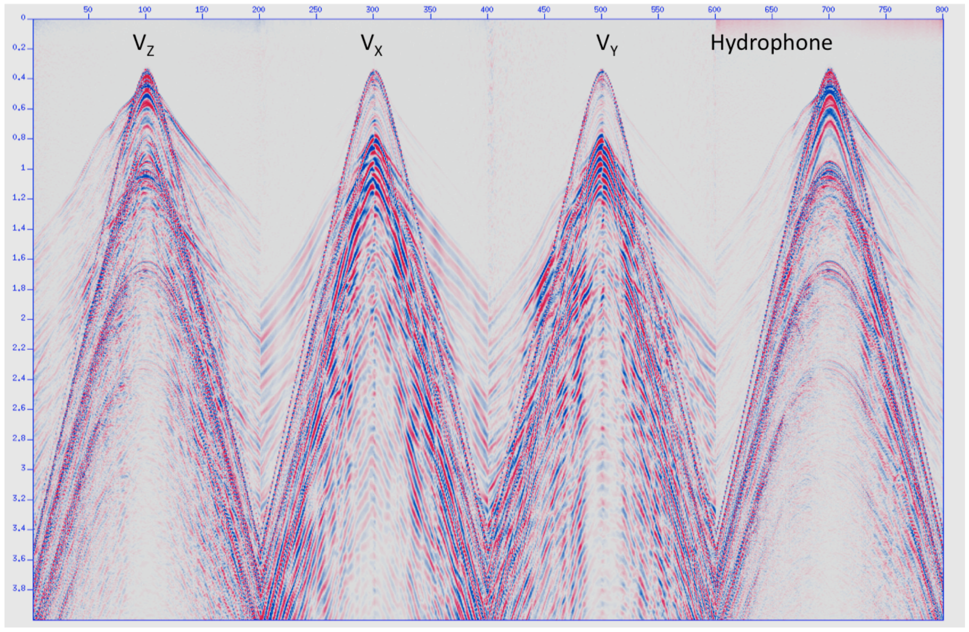


Figure 1: Four component (4C) common receiver gather for a single shot line. **[NR]** mandy2/. sufig1

Indeed, Yan and Sava (2008) has proposed a way to perform isotropic elastic reverse time migration in the angle-domain. Such a scheme, although feasible, is still computationally expensive and the dependency on both P and S velocity model can be challenging.

At this stage, we focus on P wave imaging. The hydrophone records both up- and down-going pressure waves. The additional information from the OBS geophone's Z velocity measurement includes a polarity flip between up- and down-going waves. This enables separation of the up- and the down-going waves. Accounting for instrument gain, coupling to the seabed, and the fact that pressure is the acoustic impedance times the particle velocity, we can write:

$$\begin{aligned} P &= \text{Up} + \text{Down}, \\ Z &= \text{Up} - \text{Down}. \end{aligned} \tag{1}$$

Therefore, up- and down-going data at the receiver can be obtained as shown,

$$\begin{aligned} \text{Up} &= \frac{P+Z}{2}, \\ \text{Down} &= \frac{P-Z}{2}. \end{aligned} \tag{2}$$

Equation 2 is often refered as *PZ summation* (White, 1965; Barr and Sanders, 1989; Amundsen, 1993). The advantage of turning P and Z data into up- and down-going data is that primary reflection events are contained solely in the up-going data. Therefore, a "conventional" migration can be performed on the cleaner up-going data. Recent work by Dash et al. (2009) suggested that the down-going waves should also be processed. Imaging the down-going waves is known as mirror imaging (Ebrom et al., 2000). We seek to incorporate the information in migrating with up-going data and migrating with down-going data by using a joint-inversion scheme as illustrated in Figure 2 and Figure 3.

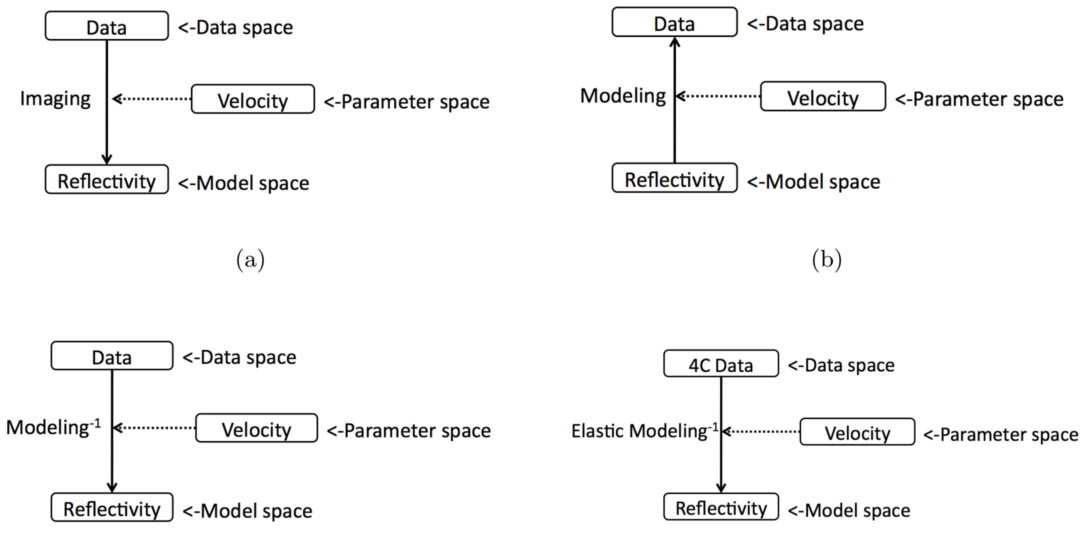
After velocity analysis, migration is the process that estimates the earth reflectivity for given data and velocity model. With the velocity as the parameter space, migration is a linear operator (Figure 2(a)). Similarly, with the same parameter and model space, modeling is a linear operator (Figure 2(b)) and can be inverted. Figure 2(c) describes the concept of linear inversion. Migration and stacking is the adjoint of modeling. Sometimes the adjoint is a very good approximation to the inverse. Ideally, a good adjoint (migration) operator should be as close to the inverse of modeling as possible.

For single-component (acoustic) data, imaging with migration or with inversion are common. For multi-component ocean bottom data, applying the corresponding concept would require the use of elastic wave-equation either in migration or in inversion (Figure 2(d)). Elastic modeling is costly and the dependency on both P and S velocity models is problematic. On the other hand acoustic modeling is less costly and does not depend on shear velocity. We limit our ambition to acoustic waves and use hydrophone and vertical component of geophone measurements to image with both the primaries and the multiples signal. We illustrate this method in Figure 3. P and Z data can be converted into up- and down-going data with PZ summation, Equation 2. Up- and down-going data can be converted into over/under data, which are pressure waves sampled at two depth levels. Effectively, we can transform from P and Z data to up- and down-going data, and finally to over/under data. This allows us to perform inversion in the acoustic regime.

It is worth mentioning that our joint inversion scheme can be applied to the processing of dual-sensor streamer data. The P and Z measurements of dual-sensor data can be conveniently combined to produce up- and down-going data using PZ summation and subsequently be used for joint-inversion imaging. The two advantages with this procedure for dual sensor data are (1) the repairing of attenuated signal at the notch frequencies and (2) the improved image from the additional information provided by the mirror signal (receiver ghost).

This is a progress report to apply joint inversion of up- and down-signal described in Wong et al. (2009). We apply reverse time migration to synthetic ocean bottom data to compare the migration result using either the up-going or the down-going signal. While OBS measure the pressure and shear waves, we limit our synthetic study to acoustic data only. To avoid using elastic modeling to synthesize multi-component ocean bottom data, we generate acoustic up- and down-going ocean bottom recording in two steps. First, two-way acoustic wave-equation is used to generate data at two different depths level, called the over/under data, near the seabed. Then an up-down separation operator, $\mathbf{S}$, is applied to produce our desired up- and down-going ocean bottom data.

We found that while migrating the up-going data gives strong reflector amplitudes and weak artifacts, its area of sub-surface illumination is narrower than migrating with down-going signal. We also found that there are additional RTM artifacts because of the recorded waves injection at the ocean bottom. This new class of artifacts does not exist with the traditional



(a) (b)

(c) (d)

Figure 2: (a) The processing flow of imaging with reflectivity, also known as migration. (b) The flow of modelling, which creating synthetic data from reflecitivity and velocity. (c) The flow of inversion, which is linear if the model space is reflectivity and the velocity is in the parameter space. (d) The flow of elastic inversion. [**NR**] mandy2/. sufig8a,sufig8b,sufig8c,sufig8d

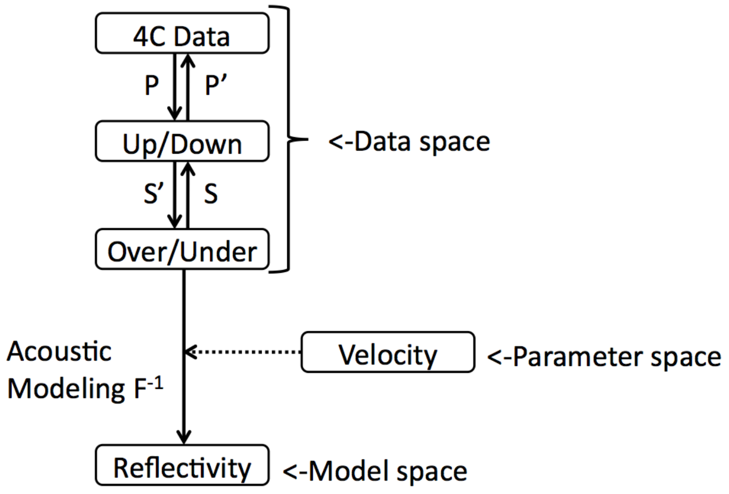


Figure 3: Acoustic inversion of multi-component data using only the P waves. Preprocessing of the 4C data to over/under enables us to use acoustic- instead of elastic-modeling. [**NR**] mandy2/. sufig9

streamer geometry.

The rest of the paper is organized as follows. We first explain our method of constructing the RTM operator and the up-down separation operator. Next, we present the results of migrating with the up-going wave only, down-going wave only, and the total signal. A discussion on the types of artifacts follows. Finally, we propose a way to eliminate ocean bottom RTM artifacts, as well as combining up- and down-going migration results using joint inversion, which will be the focus of future research.

THE RTM AND THE UP-DOWN SEPARATION OPERATORS

The RTM operator

Since there are many different ways to implement reverse time migration, we must specify how we construct our RTM operator. Our model space is the reflectivity of the subsurface. It is computed by cross-correlating the source wavefield with the receiver wavefield injected backward in time,

$$m(\mathbf{x}) = \sum_{\mathbf{s}} \sum_{t} u_s(\mathbf{x}, t, \mathbf{s}) u_r(\mathbf{x}, t, \mathbf{s}), \tag{3}$$

where $\mathbf{s}$ marks the location of the source, t is the time, $\mathbf{x}$ represents a point in the sub-surface, $m(\mathbf{x})$ is the reflectivity of the sub-surface, and $u_s(\mathbf{x}, t, \mathbf{s})$ and $u_r(\mathbf{x}, t, \mathbf{s})$ are the source and receiver wavefields at location $\mathbf{x}$, time t, and shot location $\mathbf{s}$. Our reverse time migration is set up so that $u_s(\mathbf{x}, t, \mathbf{s})$ and $u_r(\mathbf{x}, t, \mathbf{s})$ are computed using a background velocity model, $V_o(\mathbf{x})$ and the constant-density acoustic wave equation:

$$\frac{1}{V_o^2} \frac{d^2 u}{dt^2} = \nabla^2 u. \tag{4}$$

In our finite differencing scheme, we approximate the Laplacian to fourth order in space and the time derivative to second order in time. Note that in this study, we only migrate with the correct velocity. The correct velocity is smoothed to avoid spurious cross-correlation artifacts.

For ocean-bottom data, the receiver ghost reflection is very valuable, because when applied with mirror imaging, it produces better sub-surface illumination than the primary event. This claim will be apparent from later results. To create overlapping of the receiver ghost for ocean bottom RTM, we must have a reflecting top boundary.

The up-down separation operator

The derivation for decomposing over/under pressure waves into up-going and down-going signals is best done in the Fourier domain. For a thorough review of this method, please refer to Sonneland et al. (1986). Denote $P_1(\omega, k_x)$ and $P_2(\omega, k_x)$ to be the Fourier-transformed measurements of compressional waves at depths z_1 (over) and z_2 (under). Theoretically,

$P_1(\omega, k_x)$ is a sum of the up-going $U_1(\omega, k_x)$ and down-going $D_1(\omega, k_x)$ components. Likewise for $P_2(\omega, k_x)$:

$$\begin{aligned} P_1(\omega, k_x) &= U_1(\omega, k_x) + D_1(\omega, k_x), \\ P_2(\omega, k_x) &= U_2(\omega, k_x) + D_2(\omega, k_x). \end{aligned} \tag{5}$$

Down-going waves arrive at the under array (D_2) before the over (D_1) array. Therefore, shifting D_2 forward in time would match the signal D_1. Similarly, up-going waves visit the over array first. Therefore, shifting U_1 forward in time would match the signal U_2. This relationship is equivalent to a phase-shift in the Fourier domain:

$$\begin{aligned} e^{ik_z \Delta z} D_2 &= D_1, \\ U_2 &= e^{ik_z \Delta z} U_1, \end{aligned} \tag{6}$$

where $\Delta z = z_2 - z_1$, and k_z is the usual dispersion relation. Finally, substituting equation 6 into equation 5 yields the formula for the up-going and down-going waves at the receivers:

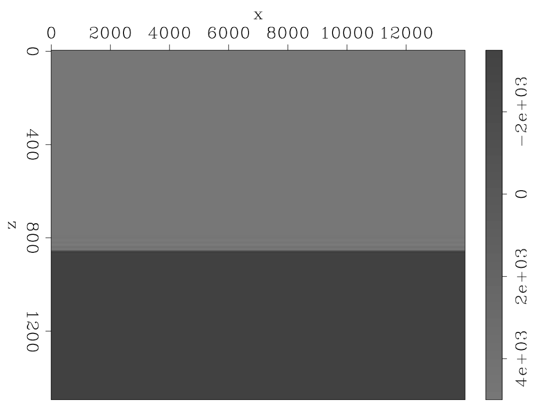
$$\begin{aligned} U_2 &= \frac{P_2 - e^{ik_z \Delta z} P_1}{1 - e^{2ik_z \Delta z}}, \\ D_2 &= \frac{e^{ik_z \Delta z} P_1 - e^{2ik_z \Delta z} P_2}{1 - e^{2ik_z \Delta z}}. \end{aligned} \tag{7}$$

Over/under acquisition is used to eliminate receiver ghosting and water reverberation. Although over/under arrays are rarely placed on the sea floor in real seismic surveys, this technique allows easy generation of up-going and down-going waves at the sea bottom for synthetic examples or modeling. For the remaining of this paper, we will denote the operation that separates over/under data into up-down data in equation 7 as $\mathbf{S}$.

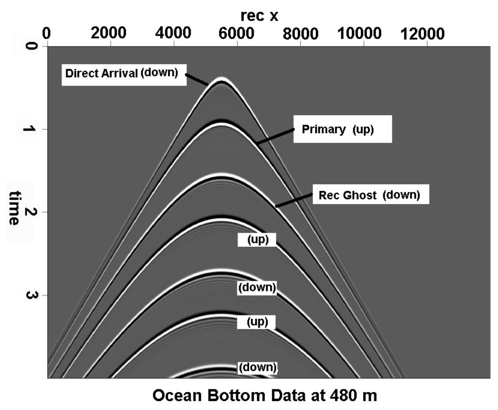
SYNTHETIC OCEAN-BOTTOM DATA

Originally, field ocean-bottom data consists of two measurements, the pressure and the shear waves. However, we limit our synthetic study by using the acoustic wave equation instead of the elastic wave equation. Instead of converting the P and Z measurements into up- and down-going data, we use the constant density, two-way acoustic wave equation to first generate over/under data, which are measured at two depth levels near the ocean bottom. Next, we convert the over/under data into up-down data using the separation operator $\mathbf{S}$ described in the last section. We use a very simple one-reflector problem. Figure 4(a) shows the velocity model used to generate the synthetic over-under data.

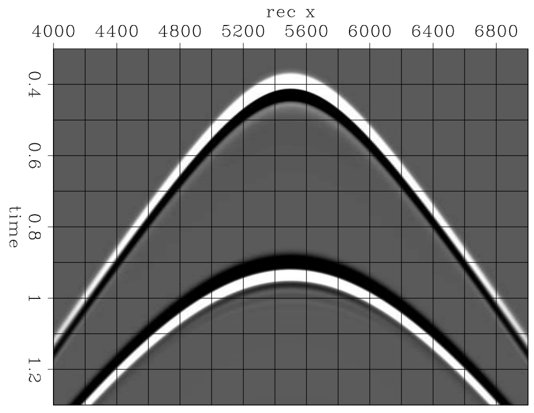
One shot is fired at zero depth and $x = 5500$ m. Two lines of receivers are placed at 480 m and 500 m depth and span from $x = 0$ m to $x = 14000$ m with a 10 m spacing. The receiver signals are shown in Figure 4(b). The first signal that arrives in time is the direct arrival, which propagates from the source directly downward to the receivers. The second



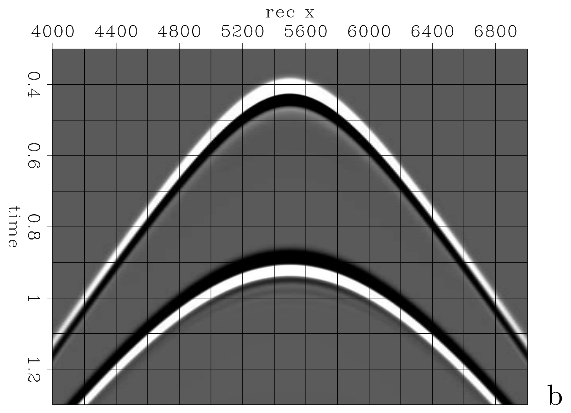
(a)



(b)



(c)



(d)

Figure 4: (a) Our simple one reflector velocity model. (b) The data collected at 480 m depth using the two-way acoustic wave equation. (c) The direct arrival and primary reflection of the over data. (d) The direct arrival and the primary reflection of the under data. **[CR]** mandy2/. velmodel,data1,over-data,under-data

signal is our primary reflection that bounces off the reflector at 800 m depth and returns to the receiver going upward. The third signal is the receiver ghost. Figure 4(c) and Figure 4(d) show a close up of the direct arrival and primary reflection of the over and the under data collected. Notice that the direct wave arrives sooner in figure 4(c) than in figure 4(d), because the direct arrival is down-going. Similarly, the up-going primary reflection arrives sooner in Figure 4(d) than in Figure 4(c).

RTM ON OCEAN-BOTTOM DATA

In our RTM scheme, we perform migration by injecting over-under data into the finite difference grid. Figure 5(a) shows the resulting reflectivity after injecting the over signal at 480m depth and under signal at 500 m depth. As expected from the original velocity model displayed in Figure 4(a), there is a reflector below the 800 m depth. In addition, there are also two weaker coherent artifacts at 1200 m and 1300 m depth.

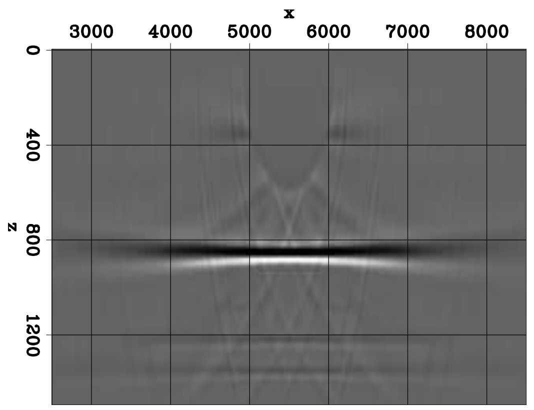
We also migrate with only the up-going and only the down-going signal. This can be accomplished using the adjoint of the up-down separation operator $\mathbf{S}'$ and zeroing the appropriate data component in the up-down space. Let d_o and d_u be the over and under data synthesized as described in the last section. To filter out the down-going energy in d_o and d_u, we can zero the down-going data in the up-down data space before applying the adjoint separation operator $\mathbf{S}'$ as shown below:

$$\begin{bmatrix} \tilde{d_o^{\uparrow}} \\ \tilde{d_u^{\uparrow}} \end{bmatrix} = \mathbf{S}' \left(\mathbf{S} \begin{bmatrix} d_o \\ d_u \end{bmatrix} \right)_{d_\downarrow = 0} = \mathbf{S}' \begin{bmatrix} d_\uparrow \\ d_\downarrow \end{bmatrix}_{d_\downarrow = 0} .$$

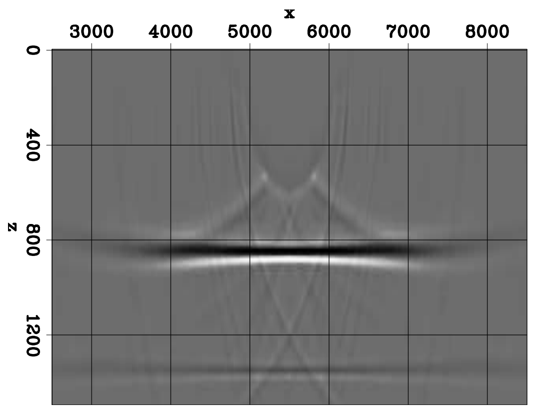
Note that the resulting data $\tilde{d_o^{\uparrow}}$ and $\tilde{d_u^{\uparrow}}$ are still in the over-under data space but contain only up-going energy. Similarly, to filter out the up-going energy in d_o and d_u, we can use:

$$\begin{bmatrix} \tilde{d_o^{\downarrow}} \\ \tilde{d_u^{\downarrow}} \end{bmatrix} = \mathbf{S}' \left(\mathbf{S} \begin{bmatrix} d_o \\ d_u \end{bmatrix} \right)_{d_\uparrow = 0} = \mathbf{S}' \begin{bmatrix} d_\uparrow \\ d_\downarrow \end{bmatrix}_{d_\uparrow = 0} .$$

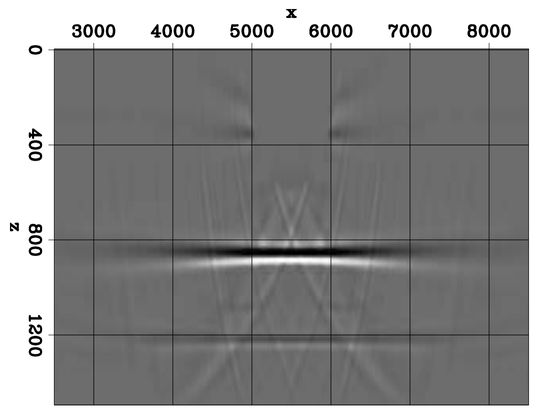
The results of performing RTM with only the down-energy and only the up-energy are shown in Figure 5(b) and Figure 5(c) respectively. Both figures have the same clip. In Figure 5(b), the reflector at 800 m is much wider than the corresponding one in Figure 5(c). This is because migrating with down-going energy is primarily imaging with the receiver ghost signal. On the other hand, migrating with up-going energy is primarily imaging with the primary signal. Imaging with the receiver ghost data provides a wider subsurface illumination, because its point of reflection is further away from the receiver than that of the primary reflection. Another observation is that the amplitude of the 800 m reflector in figure 5(c) is stronger than the amplitude in Figure 5(b). The primary signal has stronger amplitude than any signal of higher order (ghost or multiples) because of geometric spreading. Therefore, imaging with the primary gives a higher-amplitude reflectivity model than imaging with the receiver ghost.



(a)



(b)



(c)

Figure 5: (a) The migration result using the over-under data. There is a reflector below the 800 m depth. In addition, there are also two weaker coherent artifacts at 1200 m and 1300 m depth. (b) The migration result of over-under with up-going only energy; (c) the migration result of over-under data with down-going only energy. **[CR]** mandy2/. ou0,ou1,ou2

Artifacts in Ocean-Bottom RTM

The two artifacts that appear in Figure 5(a) can be identified when migrating with only up-energy and with only down-energy. The deeper artifact ($\approx$ 1300 m) appears in Figure 5(b) while the 1200 m artifact appears in figure 5(c). These artifacts do not appear in streamer RTM. When injecting the down-going data into the finite difference grid, the signal actually spreads both upward and downward in the computation grid. This creates spurious events when correlating, because down-going energy should only travel upward in reverse time. Similarly, up-going data should only traverse downward when injected in reverse time to the finite-difference grid. To alleviate this problem, we propose a joint inversion scheme. Such a scheme not only addresses the artifacts problem, it also provides a way to constructively combine up- and down-going migration results.

JOINT INVERSION OF UP-GOING AND DOWN-GOING OCEAN-BOTTOM DATA

The goals of our joint inversion scheme are threefold:

1. To migrate two of the four-component ocean-bottom data with acoustic modeling. Field data is collected as pressure (P)and vertical particle velocity (Z). To make use of the Z measurement, we would need to apply elastic modeling. Our proposed joint inversion scheme will avoid this difficulty.
2. To alleviate the migration artifacts in ocean-bottom RTM caused by to its acquisition geometry.
3. To combine up-going and down-going reflectivity solutions in a constructive way.

Our proposed scheme can be summarized using the flowchart in Figure 6. Field data d_P and d_Z can be converted into up-going and down-going data at the receiver level by applying the PZ summation operator $\mathbf{P}$. There are many ways to implement PZ summation; one of the simplest is described in Wong et al. (2009).

$$\begin{bmatrix} \tilde{d}_{\uparrow} \\ \tilde{d}_{\downarrow} \end{bmatrix} = \mathbf{P} \begin{bmatrix} d_P \\ d_Z \end{bmatrix}.$$

Next, up- and down-going data can be converted into over- and under- data using the adjoint of the separation operator $\mathbf{S}$:

$$\begin{bmatrix} \tilde{d}_o \\ \tilde{d}_u \end{bmatrix} = \mathbf{S}' \begin{bmatrix} \tilde{d}_{\uparrow} \\ \tilde{d}_{\downarrow} \end{bmatrix}.$$

RTM is performed using over and under data via the RTM operator, denoted as $\mathbf{F}'$:

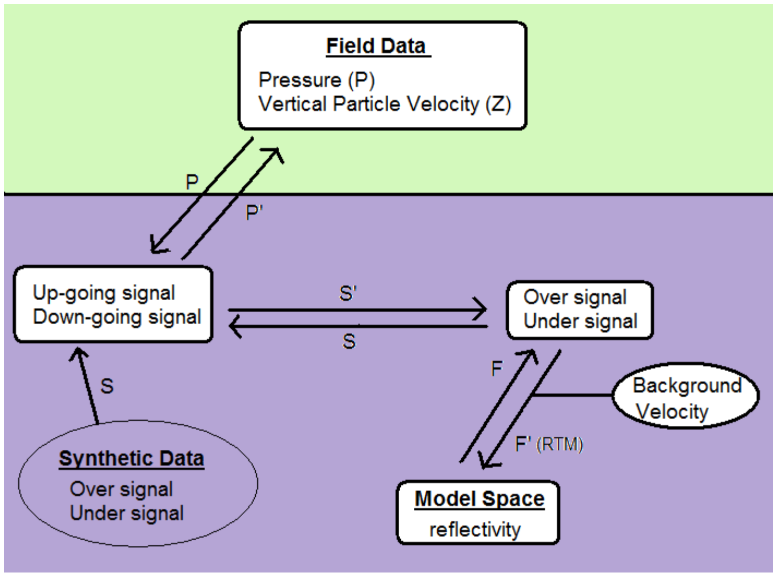


Figure 6: This flowchart shows the relationship between multi-component pressure (P) and vertical particle velocity (Z) data, up- and down-going data, and over/under data. Reverse time migration (RTM) is performed with over/undre data and generate reflectivity in migration using a background velocity model. **[NR]** mandy2/. Flowchart2

$$\mathbf{m} = \mathbf{F}' \begin{bmatrix} \tilde{d_o} \\ \tilde{d_u} \end{bmatrix}.$$

Since all three operators are linear, we can do acoustic RTM on d_P and d_Z by applying a cascade of these operators, denoted as $\mathbf{L}'$:

$$\mathbf{m} = \mathbf{F}'\mathbf{S}'\mathbf{P} \begin{bmatrix} \tilde{d_P} \\ \tilde{d_Z} \end{bmatrix} = \mathbf{L}' \begin{bmatrix} \tilde{d_P} \\ \tilde{d_Z} \end{bmatrix}.$$

We claim that by applying the inverse operator $\mathbf{L}^{-1}$ instead of the adjoint operator $\mathbf{L}'$, the three goals discussed at the beginning of the section can be achieved. The testing of this theory is the focus of our current research.

CONCLUSION

We discuss our progress in migrating multi-component ocean-bottom data, focusing on joint inversion of up- and down-going data using two-way acoustic wave equation. We observe that applying RTM to up-going energy and down-going energy shows different illumination of the sub-surface. Migration of the up-going data gives strong reflector amplitude and weaker artifacts. However, its area of sub-surface illumination is narrower than migrating with the down-going signal due to mirror imaging. In addition, a new class of artifacts appears because of the injection of receiver wavefields with the ocean-bottom geometry. We propose a joint-inversion scheme that can constructively achieve three goals; (1) combining the up- and down-going migration result, (2) removing spurious artifacts in the final reflectivity model and (3) migrating multi-components ocean bottom data with only acoustic modeling.

ACKNOWLEDGMENTS

We also thank SeaBird Explorations for the release of the field dataset.

REFERENCES

Amundsen, L., 1993, Wavenumber-based filtering of marine point-source data: Geophysics, **58**, 1497–150.

Barr, F. J. and J. I. . Sanders, 1989, Attenuation of water-column multiples using pressure and velocity detectors in a water-bottom cable: 59th annual international meeting: SEG Expanded Abstracts, 653–656.

Baysal, E., D. Kosloff, and J. W. C. Sherwood, 1983, Reverse time migration: Geophysics, **48**, 1514–1524.

Dash, R., G. Spence, R. Hyndman, S. Grion, Y. Wang, and S. Ronen, 2009, Wide-area imaging from obs multiples: Geophysics, in press.

Ebrom, D., X. Li, and D. Sukup, 2000, Facilitating technologies for permanently instrucmented oil fields: The Leading Edge, **19**, 282–285.

Etgen, J. T., 1986, Prestack reverse time migration of shot profiles: SEP-Report, **50**, 151–170.

Sonneland, L., L. E. Berg, P. Eidsvig, A. Haugen, B. Fotland, and J. Vestby, 1986, 2-d deghosting using vertical receiver arrays: SEG Expanded Abstracts, **5**, 516.

White, J. E., 1965, Seismic waves: radiation, transmission, and attenuation: McGraw-Hill.

Whitmore, N. D., 1983, Iterative depth migration by backward time propagation: SEG Expanded Abstracts, **2**, 382.

Wong, M., B. L. Biondi, and S. Ronen, 2009, Joint inversion of up- and down-going ocean bottom data: SEP-Report, **138**, 225–234.

Yan, J. and P. Sava, 2008, Isotropic angle-domain elastic reverse-time migration: Geophysics, **73**, S229.

Theory and practice of interpolation in the pyramid domain

Antoine Guitton and Jon Claerbout

ABSTRACT

With the pyramid transform, 2-D dip spectra can be characterized by 1-D prediction-error filters (pefs) and 3-D dip spectra by 2-D pefs. This transform takes data from (ω, x)-space to data in $(\omega, u = \omega \cdot x)$-space using a simple mapping procedure that leaves empty locations in the pyramid domain. Missing data in (ω, x)-space create even more empty bins in (ω, u)-space. We propose a multi-stage least-squares approach where both unknown pefs and missing data are estimated. This approach is tested on synthetic and field data examples where aliasing and irregular spacing are present.

INTRODUCTION

For interpolation, it is often assumed that the seismic data are made from a superposition of locally planar events [Claerbout (1992); Symes (1994); Fomel (2002)]. Since plane-waves are highly predictable, interpolating seismic data with prediction-error filters (pefs) is both very effective and efficient, either in the Fourier [Spitz (1991)] or time domain [Crawley et al. (1999)].

In the Fourier domain and for aliased data, pefs from lower frequencies are used to de-alias higher frequencies [Spitz (1991)]. Therefore, many pefs need to be estimated for complete processing. Sun and Ronen (1996) introduce a frequency dependent sampling in the (ω, x) domain such that a data vector $\mathbf{d}$ at each frequency in the (ω, x) domain is mapped into a new vector $\mathbf{m}$ (pyramid domain) according to

$$m(\omega, u) = d(\omega, x = u/\omega), \tag{1}$$

where u has units of velocities and x is a spatial axis (offset or mid-point position). We demonstrate in this paper that this transform has the advantage of making the pefs frequency independent, thus offering useful properties (for stationary data):

- Only one pef is necessary for interpolation
- This single pef can be estimated from all the frequencies
- For noisy data, many regressions are available from all the frequencies, yielding robust pef estimation

This paper investigates the pyramid domain method and introduces a linear operator called the pyramid transform. First, we illustrate the properties of the pyramid transform and explain why, in theory, only one pef is necessary for interpolation. Second, we identify mapping artifacts from (ω, x)-space to $(\omega, u = \omega \cdot x)$-space and propose a strategy both to attenuate them and to interpolate seismic data. This strategy works for both aliased and irregularly-sampled data. Finally, we illustrate the proposed algorithm on synthetic and real data cases.

THEORY: INTRODUCING THE PYRAMID TRANSFORM

In this section, we first introduce the pyramid transform, which is a linear operation that remaps the Fourier transformed data in (ω, x) into the pyramid domain in (ω, u). Then we introduce some properties of the pyramid domain and their implications for pef estimation and missing data interpolation.

The pyramid transform

To begin introduce the forward transformation from the pyramid domain (ω, u) to the Fourier space (ω, x) as follows:

$$d(\omega, x) = m(\omega, u = \omega \cdot x). \tag{2}$$

The adjoint transformation is defined in equation (1). These definitions can be extended in 3-D easily for the forward case:

$$d(\omega, x, y) = m(\omega, u = \omega \cdot x, v = \omega \cdot y), \tag{3}$$

and for the adjoint case:

$$m(\omega, u, v) = d(\omega, x = u/\omega, y = v/\omega), \tag{4}$$

where y is a spatial axis in the crossline direction (offset of mid-point position) and v the dual of y in the pyramid domain. Equations (2) and (3) can be rewritten in a more compact form:

$$\mathbf{d} = \mathbf{L}\mathbf{m}, \tag{5}$$

where $\mathbf{L}$ is the pyramid transform. Similarly, equations (1) and (4) can be rewritten as

$$\mathbf{m} = \mathbf{L}'\mathbf{d}, \tag{6}$$

where $\mathbf{L}'$ is the adjoint of $\mathbf{L}$. Note that the remapping between x and u (plus y and v in 3-D) requires an interpolation operator. A linear interpolation process looping over the data space $d(\omega, x)$ is applied in all our results.

Properties

The equation of a plane-wave in (t, x) is given by

$$P(t, x) = f(t - px), \tag{7}$$

where f is a waveform and p is a constant slope. The prediction operator B from one trace to another at a distance Δx in (ω, x) for the plane-wave in equation (7) is given by Fomel (2002):

$$B(\omega, \Delta x) = e^{i\omega p \Delta x}. \tag{8}$$

If we now introduce the new variable $\Delta u = \omega \Delta x$ in equation (8), then we have a new prediction operator A in the (ω, u) domain:

$$A(\omega, \Delta u) = e^{ip\Delta u}. \tag{9}$$

From Equation (9), we notice that the dependency in ω of the prediction operator has vanished in the pyramid domain. Sun and Ronen (1996) extend this property to more than one plane-wave. We illustrate the pyramid domain in Figure 1. Figure 1a shows two plane-waves with the same wavelet (Ricker 2 with a fundamental frequency of 20Hz). Figure 1b displays the real part of the Fourier transformed data in the (ω, x) domain and Figure 1c the real part of the (ω, u) pyramid domain. Notice that we limited the range of frequencies for display purposes only in Figures 1b and 1c. In 2-D, the pyramid domain maps into a triangle-shaped area. In 3-D, it will map into a pyramid-shaped volume. Consequently, each trace in (ω, x) is transformed into a radial trace in (ω, u).

As anticipated from the definition of the prediction operator in equation (9), the information at each frequency in Figure 1c is independent of ω, which means that any scheme involving pefs in the pyramid domain will require only one filter (one 1-D filter for 2-D data, and one 2-D filter for 3-D data). In addition, we observe that the slope p now acts as a wavenumber on the u axis. This fact implies that the low velocity event in Figure 1a will look like a high wavenumber component on the u-axis whereas the high velocity event will look like a low wavenumber. An added feature is that since we only have one pef for the whole domain (in theory), the filter estimation should be relatively robust to the noise present in the data (especially for random noise).

Transformation artifacts

Going back and forth between the (ω, x) and (ω, u) domains will leave artifacts in (t, x). There are two main reasons for this. First, the linear interpolation operator we use to transform one domain to the other is not unitary. Second, the pyramid transform (forward and adjoint operators) compresses and stretches the horizontal axis in ways that can affect the reconstruction of the frequency content, especially for the low frequencies [remember that a trace in (ω, x)-space is mapped into a radial trace in (ω, u)-space]. We illustrate this effect in Figure 2a. This Figure is the result of the following operation:

$$\tilde{\mathbf{d}} = \mathbf{L}\mathbf{L}'\mathbf{d} \tag{10}$$

where $\mathbf{d}$ is the Fourier transform of Figure 1a and $\tilde{\mathbf{d}}$ contains the reconstructed data. In this example, because the u axis is too coarse, the information of the slowest event (ringiest on the u axis) disappears.

In order to mitigate these effects, we propose making the u axis very dense. Theoretically, we could derive the maximum bin size δu to accommodate the slowest event. From simple Fourier analysis, we can establish that

$$\delta u \leq \frac{1}{2p_{max}}, \tag{11}$$

where p_{max} is the slope of the slowest event. This relation is a necessary, but not sufficient, condition for δu since some of the artifacts are also due to the linear interpolation itself (i.e., the linear interpolation operator is not unitary). Therefore in practice, smaller δu's than the one in equation 11 are necessary. Having very fine sampling in u will help attenuate most of the transformation errors seen in Figure 2. Figure 3 shows $\mathbf{L}\mathbf{L}'\mathbf{d}$ for the same dataset, but with a sampling 12 times finer on the u axis than on Figure 2b. Now, Figure 3a shows the two events with some remaining artifacts due to the linear interpolation operator only. In the next section, we introduce an algorithm that will both remove these remaining artifacts and also allow us to interpolate missing data.

ALGORITHM FOR MISSING DATA INTERPOLATION

Having a very fine sampling on the u axis attenuates most of the transformation artifacts of the pyramid transform. However, as seen in Figure 3a, some noise due to the linear interpolation operator used for the mapping between the (ω, x) and (ω, u) domains still remains. This section introduces an algorithm that will (1) remove the remaining artifacts and (2), interpolate missing data (regularly or irregularly spaced).

Mitigating mapping effects

The remaining artifacts in Figure 3a can be attenuated in many ways. For instance, Hung et al. (2004) propose using a sinc interpolation instead of a linear interpolation. Otherwise, if missing-data interpolation is not required, the effects of linear interpolation can be attenuated by inserting a tridiagonal solver within the adjoint in equations (1) and (4). Alternatively, we can recast this attenuation as an inverse problem, in which we want to minimize the energy of the residual vector $\mathbf{r}$ where

$$\mathbf{r} = \mathbf{Lm} - \mathbf{d}, \tag{12}$$

and minimize $f(\mathbf{m}) = \|\mathbf{r}\|_2$ iteratively to find the minimum $\tilde{\mathbf{m}} = (\mathbf{L}'\mathbf{L})^{-\mathbf{1}}\mathbf{L}'\mathbf{d}$. The results of this iterative formulation can be seen in Figure 4: The plane-waves are recovered without any noise. Note that the iterative solution and the tridiagonal solver approach are equivalent. However, the missing data problem we are trying to solve makes the tridiagonal solver difficult to use.

Putting everything together

We are now ready to present the missing data interpolation algorithm. One side effect of the fine sampling of the u axis is that empty bins will appear in the pyramid domain. Missing data in $\mathbf{d}$ will add even more empty locations. We propose interpolating the missing data in $\mathbf{d}$ by filling the empty bins in $\mathbf{m}$. For this, we follow the approach of Claerbout and Fomel (2002). First, we want to honor the data where they are known by introducing the residual vector

$$\mathbf{r_d} = \mathbf{W_d}(\mathbf{Lm} - \mathbf{d}), \tag{13}$$

where $\mathbf{W_d}$ is a masking operator equal to unity where data are known, and zero where data are missing. Solving for $\mathbf{m}$ minimizing the amplitude of $\mathbf{r_d}$ only will not fill the empty bins. We need to add a regularization term that will enforce a certain multivariate spectrum to the vector $\mathbf{m}$:

$$\mathbf{r_m} = \mathbf{Am}, \tag{14}$$

where $\mathbf{A}$ is a pef in the model space. Assuming that the pef $\mathbf{A}$ is known, we can fill the empty locations in $\mathbf{m}$ by minimizing $f(\mathbf{m}) = \|\mathbf{r_d}\|_2 + \epsilon\|\mathbf{r_m}\|_2$, where ϵ is a balancing operator between data fitting and model space regularization.

There are two issues with this approach. The first issue is that the convergence towards a solution will be slow. To accelerate this process we introduce a new variable $\mathbf{q} = \mathbf{Am}$ and rewrite equations (13) and (14) as follows:

$$\begin{aligned} \mathbf{r_d} &= \mathbf{W_d}(\mathbf{LA^{-1}q} - \mathbf{d}) \\ \mathbf{r_q} &= \mathbf{q} \end{aligned} \tag{15}$$

We then minimize $f(\mathbf{q}) = \|\mathbf{r_d}\|_2 + \epsilon\|\mathbf{r_q}\|_2$ and compute $\tilde{\mathbf{m}} = \mathbf{A^{-1}}\tilde{\mathbf{q}}$, where $\tilde{\mathbf{q}}$ minimizes $f(\mathbf{q})$. The term $\mathbf{A^{-1}}$ is computed by applying a polynomial division to $\mathbf{q}$ which yields fast filling of the empty bins. Because $\mathbf{A}$ is a miminum-phase filter, the polynomial division is stable and will not cause the solution to blow-up. This preconditioning of the problem has been used in numerous geophysical problems [Clapp et al. (2004); Fomel and Guitton (2006); Herrmann et al. (2009)]. In practice, we set $\epsilon = 0$ and minimize $\mathbf{r_d}$ only [Trad et al. (2003); Guitton and Claerbout (2004)].

The second issue is that both $\mathbf{q}$ and $\mathbf{A}$ are unknown in equation (15). To circumvent this problem we bootstrap the pef estimation by first assuming that $\mathbf{A}$ is a 1-D gradient. To make sure that we can apply the polynomial division, we set the second coefficient of the gradient to -0.96 instead of -1. We then minimize $f(\mathbf{q})$ with this first pef, find a new $\tilde{\mathbf{m}}$ and estimate a better pef $\mathbf{A}$ from it. Having a better pef, we can minimize $f(\mathbf{q})$ again:

```
$\mathbf{A} \leftarrow (1 - 0.96)^T$
iterate {
    minimize $f(\mathbf{q})$
    $\tilde{\mathbf{m}} \leftarrow \mathbf{A^{-1}}\tilde{\mathbf{q}}$
    estimate $\mathbf{A}$ from $\tilde{\mathbf{m}}$
    }
```

We are essentially solving the non-linear problem in a step-wise fashion by keeping $\mathbf{A}$ constant within each non-linear loop. In practice, we notice that only 4 to 5 non-linear iterations are necessary to converge towards a pef that yields accurate interpolation of the missing data.

In the next section, we apply this algorithm to synthetic and field data examples in 2-D. We show that aliased and irregularly-sampled data can be interpolated.

EXAMPLES

Results in this section illustrate how the pyramid domain can be used to interpolate missing data. One interesting feature of the preceding algorithm is that no assumptions are made concerning the type of problem that can be tackled: here, we show interpolation results for aliased and irregularly-spaced data.

Interpolation of aliased data

We illustrate in Figure 5 the dealiasing properties of the interpolation algorithm. Figure 5a shows an aliased dataset (Δx=50 m) as a first example. Its FK spectrum is displayed in 5b and illustrates the aliasing effects above 13 Hz. We interpolate this data set on a 25 m grid by first binning the data in Figure 5a onto a 25 m grid. This binning will leave every other trace empty: the weighting operator $\mathbf{W_d}$ in equation (15) is set to zero at these locations. Figures 5c and 5d display the interpolated data in (t, x) and (f, k)-spaces. Most of the aliasing has been removed, except for the slowest event. We completely de-alias this dataset by binning Figure 5c on a 12.5 m grid and interpolating the missing traces in the pyramid domain. This final interpolation is shown in Figures 5e and 5f.

Now, we interpolate non-stationary data for two shots from one synthetic and one field experiment. Being a frequency domain approach, the interpolation in the pyramid domain forces us to decompose the data in patches, or time windows, first. The size of these windows is one second in time and 500 m in offset. Each window is processed independently. Figure 6a shows the input data for the synthetic example with a 50 m offset sampling. Its amplitude FK spectrum is displayed in Figure 6b: some events are aliased for frequencies above 15 Hz. After interpolation on a 25 m grid in Figure 6c, most of the aliased energy is gone (Figure 6d), while all the main events are preserved. Note that in principle, more interpolation steps would be necessary to remove all aliased energy (above 30 Hz).

Finally, we interpolate a shot gather from a field data experiment in the Gulf of Mexico (Figure 7). The close-up in Figure 7a shows primaries above 4 seconds and multiples below. The FK spectrum in Figure 7b shows some aliasing for the slowest events around 40 Hz. We interpolate this shot from a 26 m to a 13 m grid in Figure 7c. Most of the aliasing artifacts have been attenuated (Figure 7d).

Interpolation of irregularly-sampled data

The interpolation of irregularly-sampled data in the pyramid domain requires a binning of the data onto a regular grid first. This binning is done with a simple nearest-neighbor scheme. Like what is done for the regular case, the weighting operator $\mathbf{W_d}$ in equation (15) is set to zero at the empty trace locations. For all following examples, 50% of the traces were set to zero randomly in order to emulate a realistic acquisition geometry.

Figure 8a shows the data to interpolate with its amplitude spectrum (Figure 8b). Many methods exploit the FK domain directly in order to interpolate missing data [Xu et al. (2005); Abma and Kabir (2006); Zwartjes and Sacchi (2007)], with or without nonuniform Fourier transforms. The interpolation result in Figure 8c proves that the proposed algorithm works in this case as well. The FK domain in Figure 8d validates these findings.

Figure 9a and 9c display the input data and the interpolation results, respectively. This dataset is similar to the one used in Figure 6a. The missing traces have been properly interpolated almost everywhere. Where gaps are big, however, the proposed algorithm might have some difficulties which could be overcome by using a multiscaling strategy. Note, in Figures 9b and 9d, the clean up of the FK domain after interpolation.

Finally, we interpolate irregularly-spaced data for a field data example shown in Figure 10a. This dataset was also used in Figure 7a. Like what we observed in the previous example, the interpolation result in Figure 10c exemplifies how the proposed algorithm can interpolate missing data: the reconstructed traces look very similar to the original ones. The FK spectra in Figures 10b and 10d show the attenuation of artifacts due to the random sampling in Figure 10a.

Parameter estimation

One of the most important parameters in our algorithm is the sampling in the pyramid domain. We find that for interpolating irregularly-spaced data, half of the value of δu given in equation 11 is usually enough. For interpolating aliased data, however, the sampling becomes more difficult to establish. In this case, the sampling becomes smaller as the maximum non-aliased frequency becomes smaller. In practice, we tend to find the right sampling by trial and error, starting from the value we would use for the irregularly-spaced data case. This approach is unsatisfying and we feel that more work should be done to automate this important parameter selection.

CONCLUSION

The pyramid transform creates a domain where linear events in (t, x)-space are linear in (ω, u)-space. This property offers many opportunities for interpolation schemes based on prediction-error filters. First, one filter can predict all frequencies. Second, this pef can be estimated from all frequencies. Finally, the filter estimation should be more robust to the noise present in data (although not shown here).

One challenge with the pyramid transform is the remapping between the (ω, x) and (ω, u) domains which can introduce artifacts. We propose mitigating these effects by making the pyramid axis u very dense and using a simple linear interpolation for the transform. One consequence of this proposal is that many empty bins appear in the pyramid domain. Realizing that missing data will add even more empty locations, we introduce a non-linear algorithm that both interpolates missing data (regularly or irregularly-spaced data) and fills the empty bin locations (those resulting from the transform).

Our synthetic and field data experiments prove that the proposed algorithm works and that the pyramid domain is a sensible complement to our existing interpolation toolbox. Although not presented here, the extension to 3-D would be straightforward. Interpolating aliased or irregularly-space data does not require any change of the algorithm. However, we notice that smaller δu's are required when de-alasing is needed.

Comparing the pyramid transform to other domains for missing data interpolation goes beyond the scope of this paper, but more work needs to be done to understand how the proposed algorithm fairs when compared to more popular techniques such as FX interpolation.

The pyramid transform could benefit other applications. For instance, the irregular sampling case could be treated with a combination of nonuniform Fourier transform and pyramid transform. The signal/noise separation problem could be easily recast in the pyramid domain where only 1-D (for 2-D data) and 2-D (for 3-D data) projection filters are necessary [Soubaras (1994)].

ACKNOWLEDGMENTS

We thank Shuki Ronen and Xukai Shen for numerous discussion on this topic. We also thank the sponsors of the Stanford Exploration Project for their financial support.

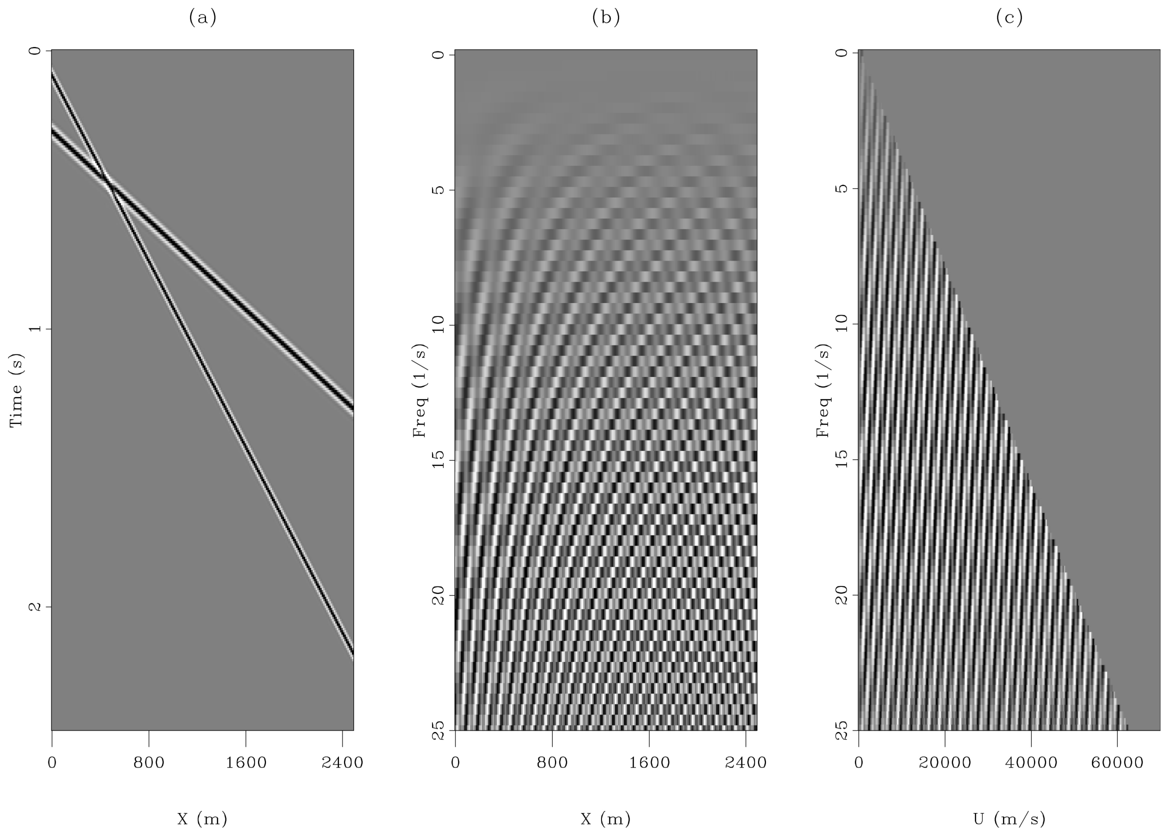


Figure 1: Two plane waves in (a) (t, x), (b) (ω, x) and (c), (ω, u) domains.(b) and (c) show the real parts only. **[NR]** **antoine1/. comp-tx-fx-fu-synthplane3**

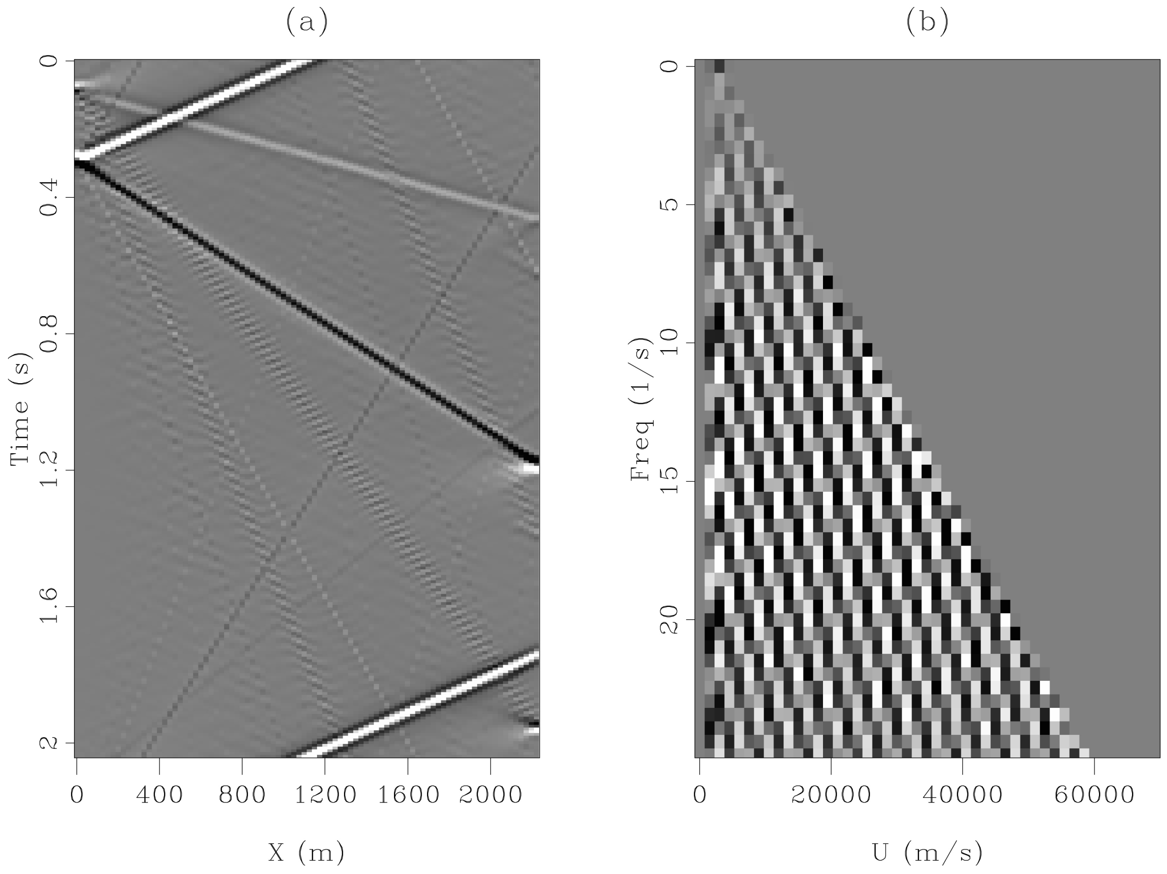


Figure 2: Illustration of transformation artifacts: (a) is $\mathbf{LL'd}$ back in the (t, x) domain and (b) is the real part $\Re(\mathbf{L'd})$, where $\mathbf{d}$ is the data in Figure 1a. The slowest event, with the highest wavenumber component on the u axis, disappears due to the parameterization of the pyramid transform. **[NR]** **antoine1/. comp-tx-fu-pyramid-60-synthplane2**

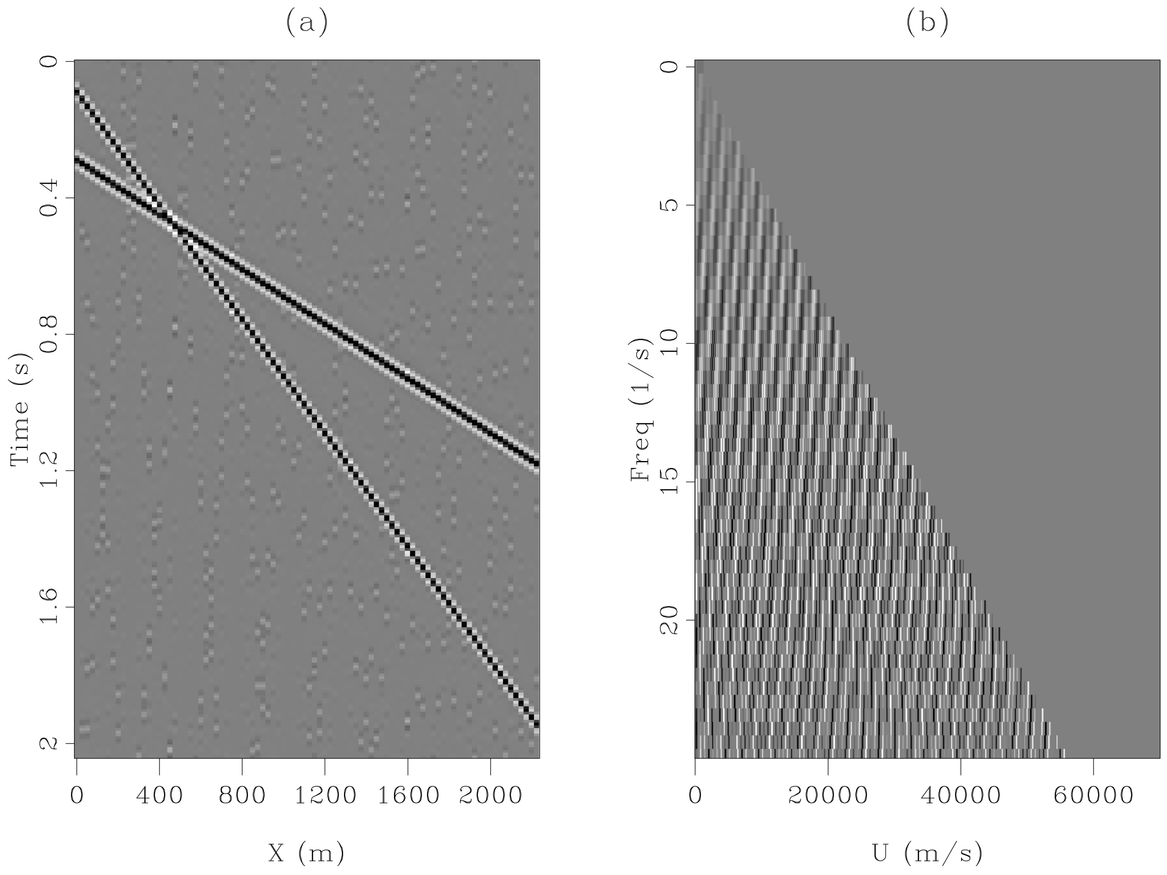


Figure 3: Same as Figure 2 but with a 12 times finer horizontal sampling on the u axis. The two plane-waves are recovered. Some noise is still present due the linear interpolation operator. **[NR]** **antoine1/. comp-tx-fu-pyramid-5-synthplane2**

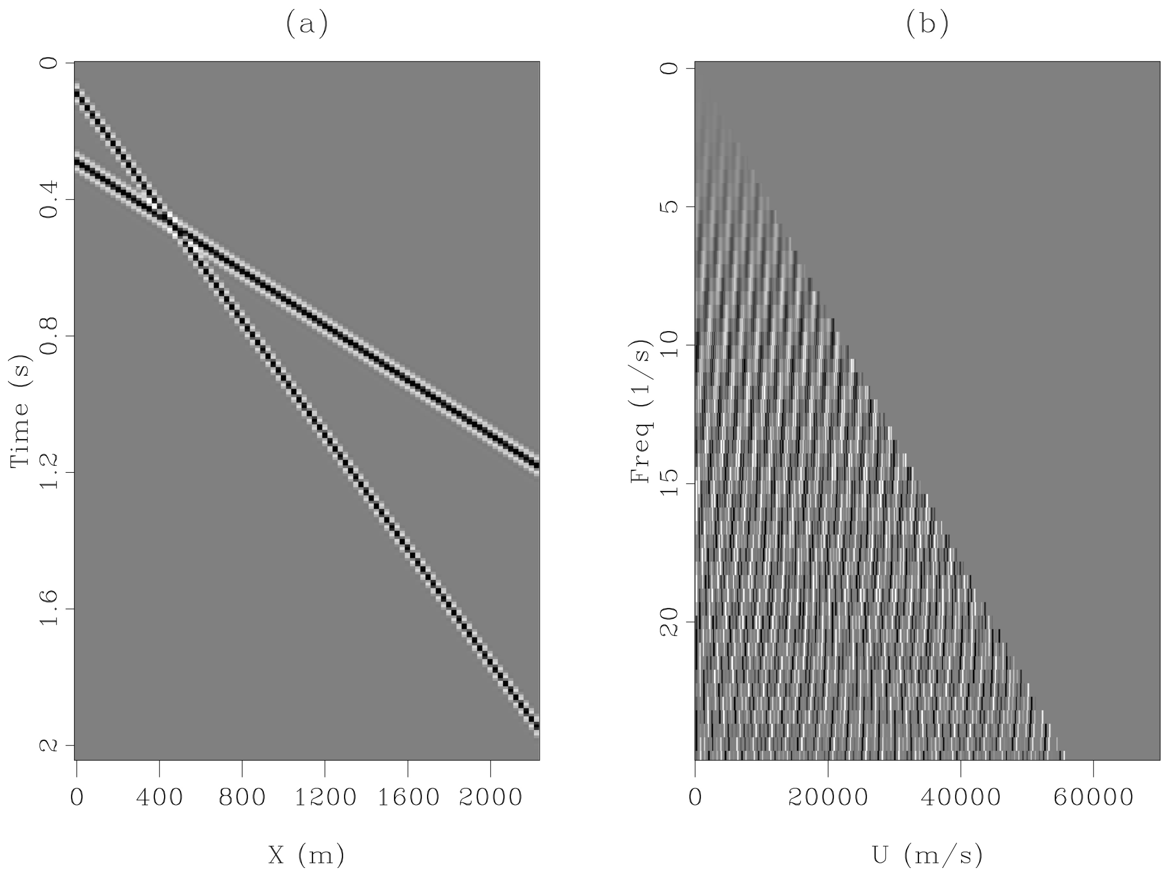


Figure 4: Illustration of the iterative process to attenuate the effects of linear interpolation. (a) is $\mathbf{L}\tilde{\mathbf{m}} = \mathbf{L}(\mathbf{L}'\mathbf{L})^{-1}\mathbf{L}'\mathbf{d}$ back in the (t, x) domain and (b) is $\Re(\tilde{\mathbf{m}})$. The noise in Figure 3a has been attenuated. **[NR]** **antoine1/. comp-tx-fu-pyramid-5iter-synthplane2**

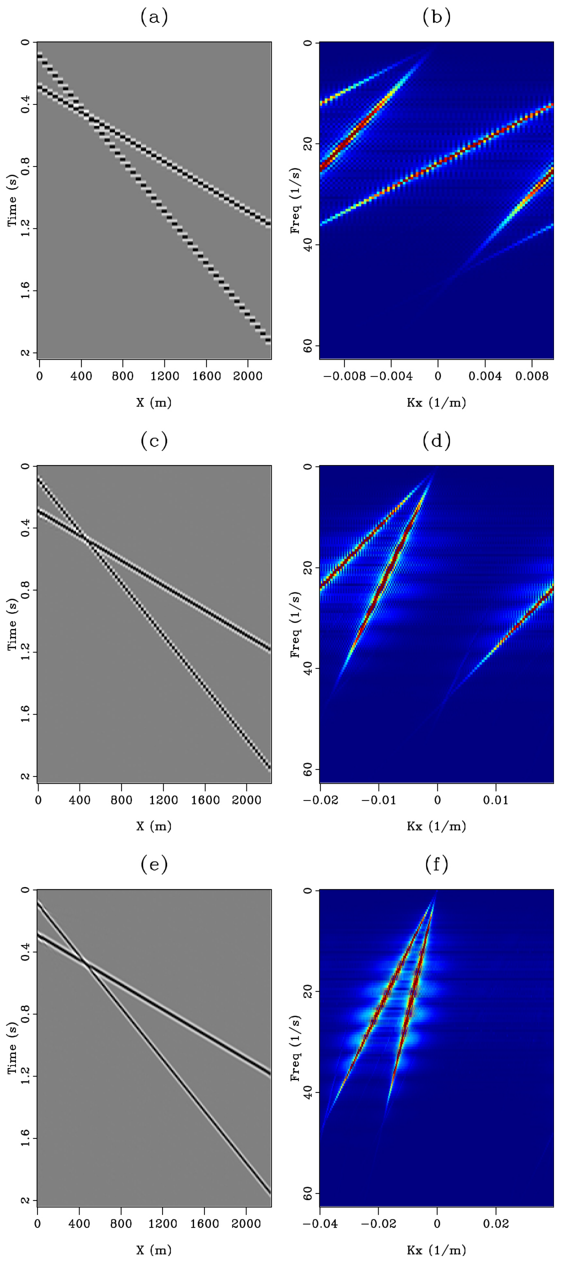


Figure 5: Interpolation results for aliased data: (a) Shows the input data at Δx=50 m and its corresponding FK spectrum in (b). The slowest event is aliased for frequencies above 13 Hz and the fastest for frequencies above 22 Hz. (c) Shows the interpolation result with Δx=25 m and the corresponding FK spectrum in (d). The slowest event is still aliased above 22 Hz. (e) Shows the interpolation result with Δx=12.5 m and the corresponding FK spectrum in (e). The data in (a) have been dealiased for all frequencies. **[NR]** $\boxed{\textbf{antoine1/. aliased-synthetic2}}$

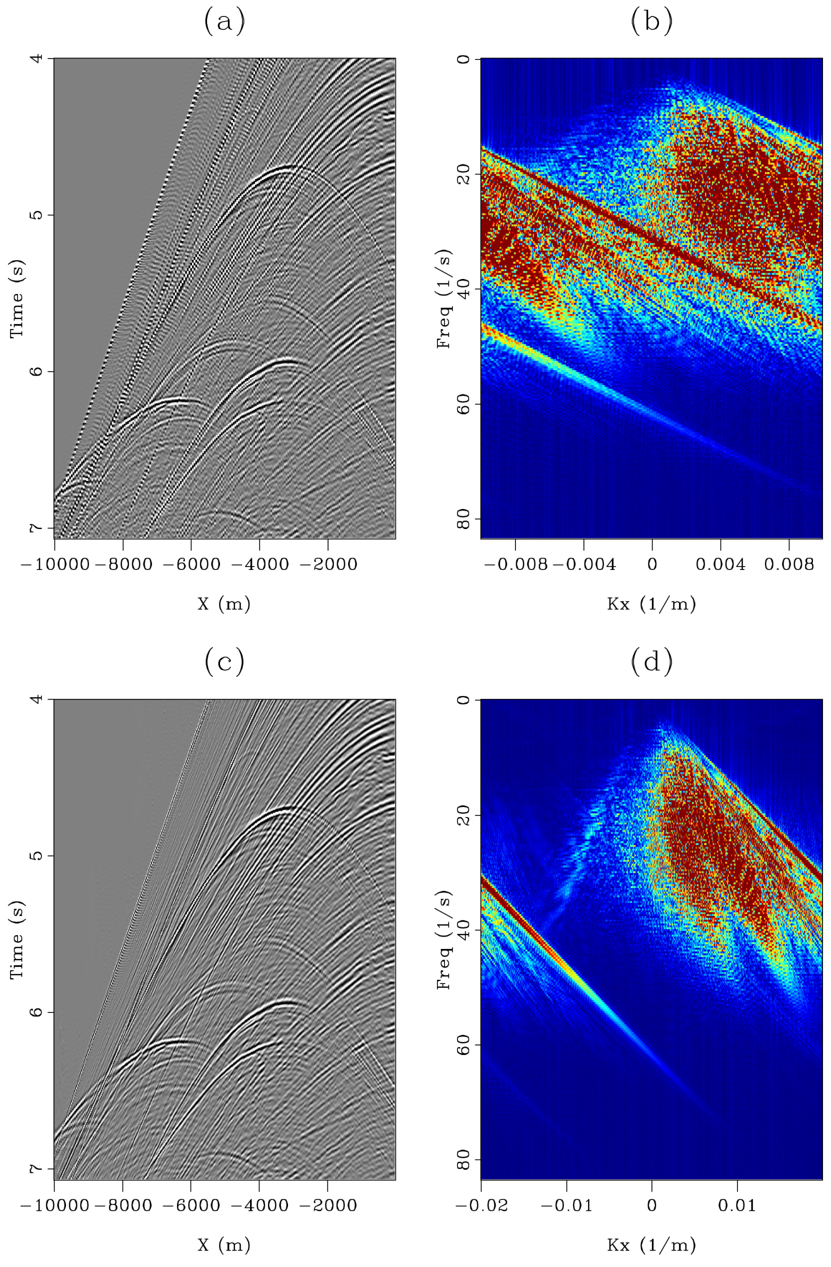


Figure 6: Interpolation results of a realistic synthetic data experiment. (a) Shows the input data on a 50 m grid with its FK amplitude spectrum in (b). (c) Shows the same data after interpolation on a 25 m grid (FK spectrum in (d)). All the events have been correctly interpolated but some aliasing remains. **[NR]** **antoine1/. aliased-bp**

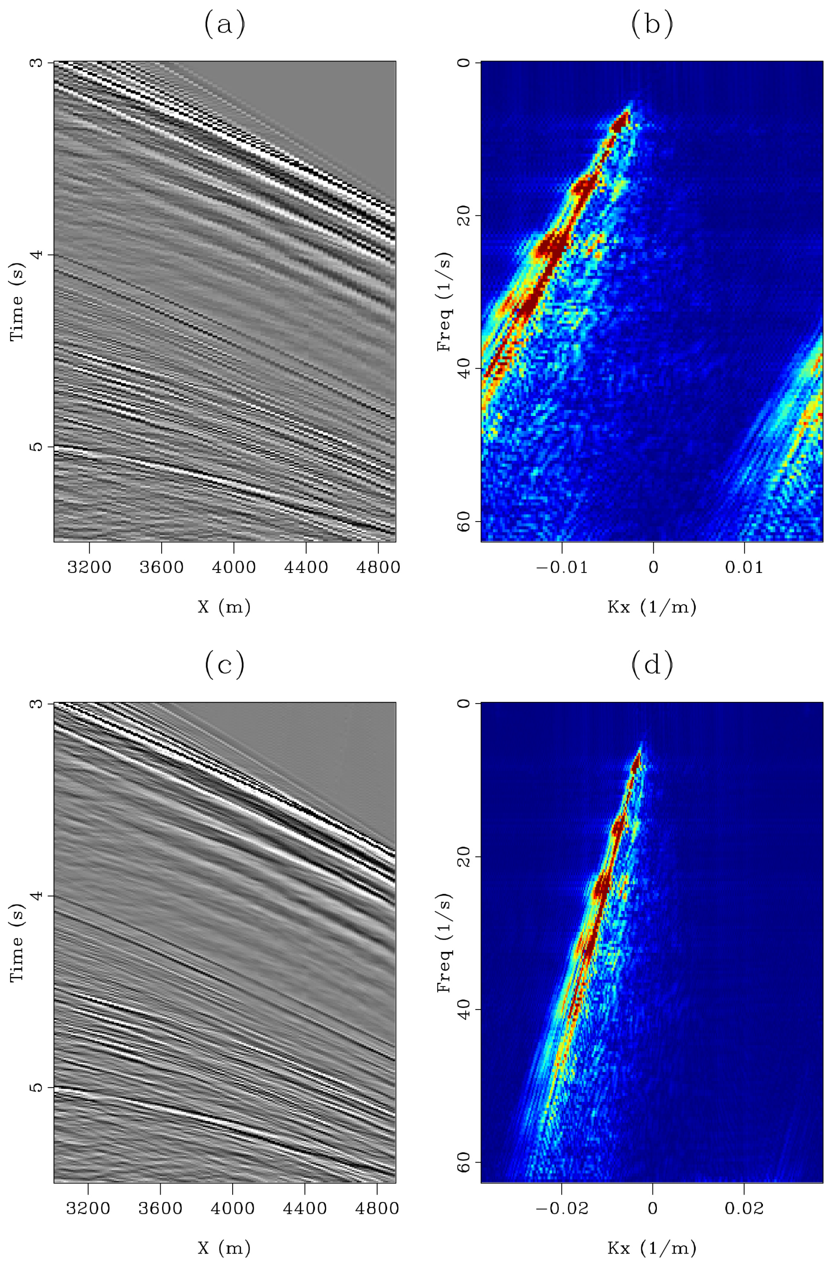


Figure 7: Interpolation results of a shot gather from the Gulf of Mexico. (a) Shows a close-up of the input data on a 26 m grid with its FK amplitude spectrum in (b). (c) Shows the same data after interpolation on a 13 m grid (FK spectrum in (d)). All the events have been correctly interpolated. **[NR]** $\boxed{\textbf{antoine1/. aliased-gm}}$

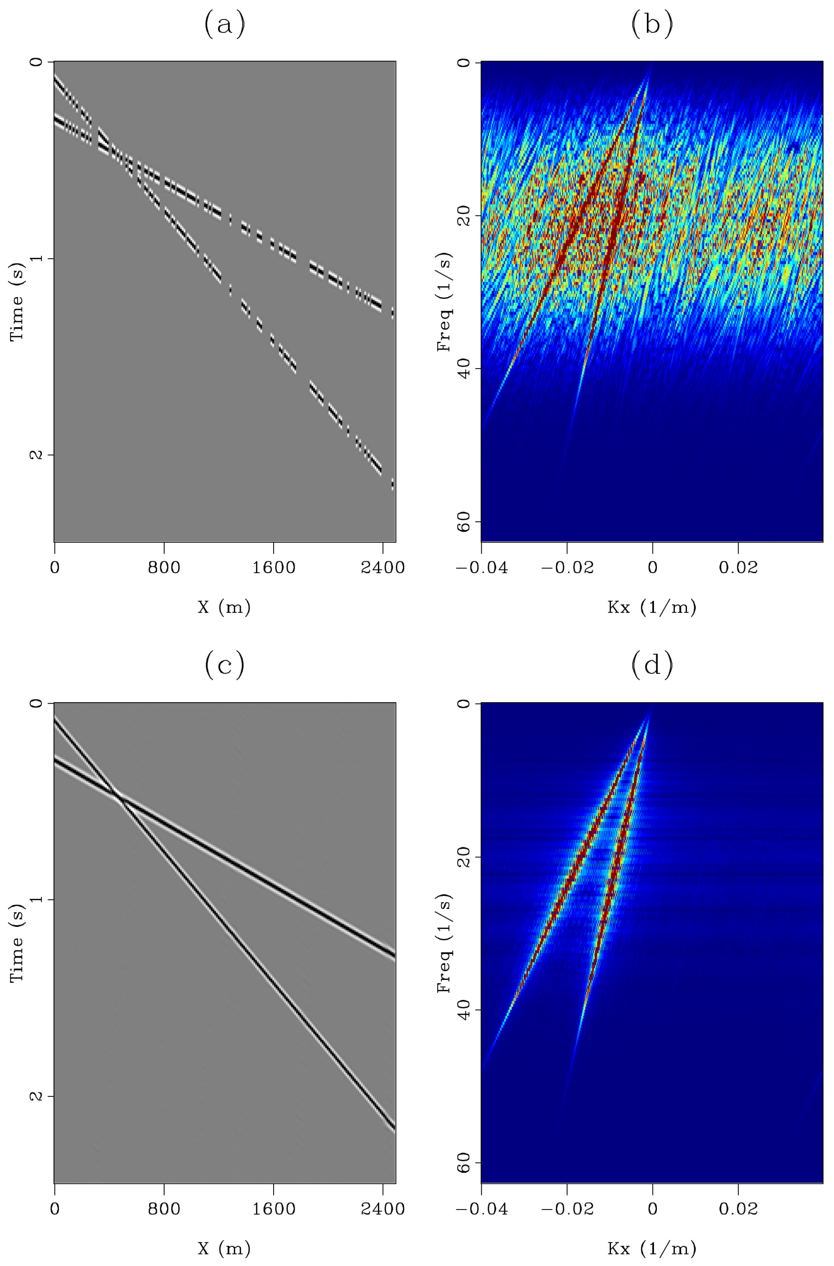


Figure 8: Interpolation of irregularly-sampled data. (a) Shows the input data binned onto a regular grid before interpolation where 50% of the traces are missing and its corresponding FK spectrum in (b). Interpolation results are shown in (c) (FK spectrum in (d)): the linear events are recovered. **[NR]** **antoine1/. irregular-synth**

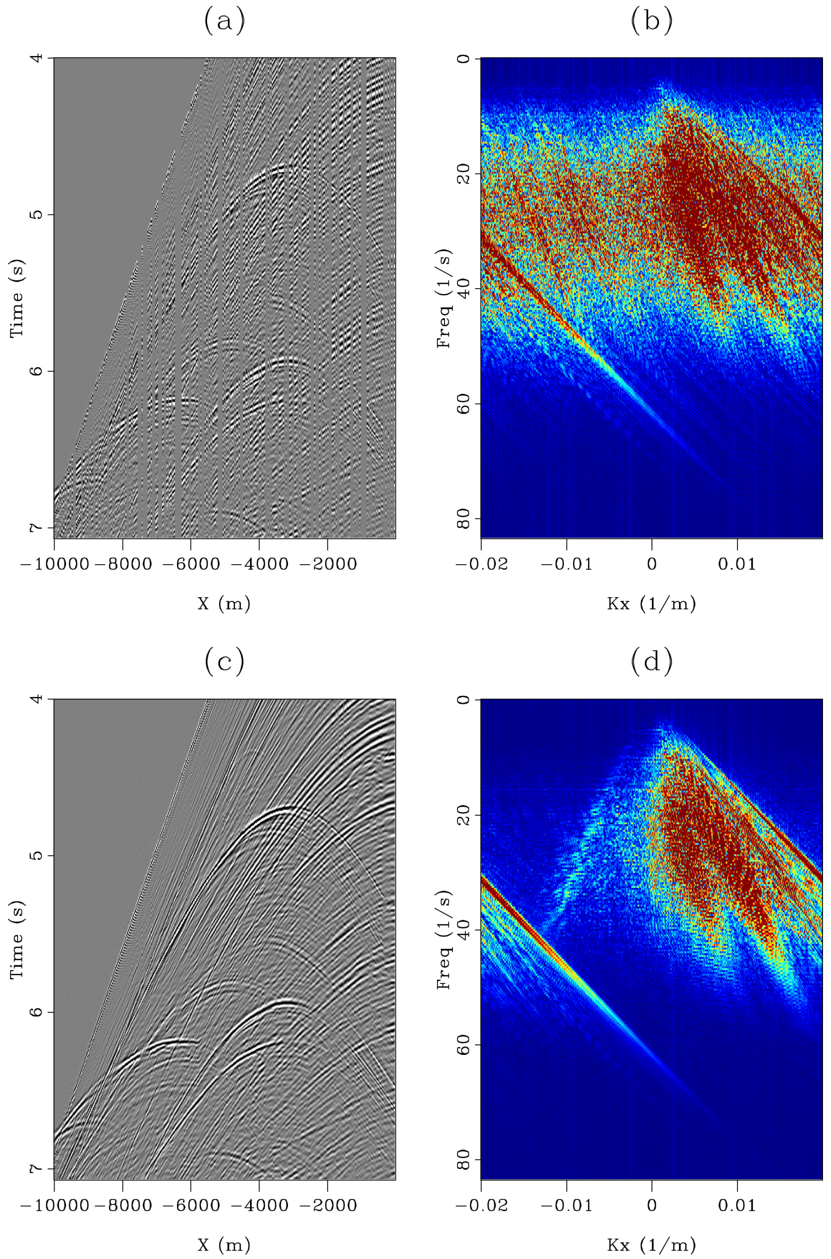


Figure 9: Interpolation of a synthetic shot gather with irregular sampling. (a) Shows the input data binned onto a regular grid before interpolation and its corresponding FK spectrum in (b). Interpolation results are shown in (c) (FK spectrum in (d)). Our proposed algorithm recovers the missing traces where conflicting dips are present. **[NR]** **antoine1/. irregular-bp**

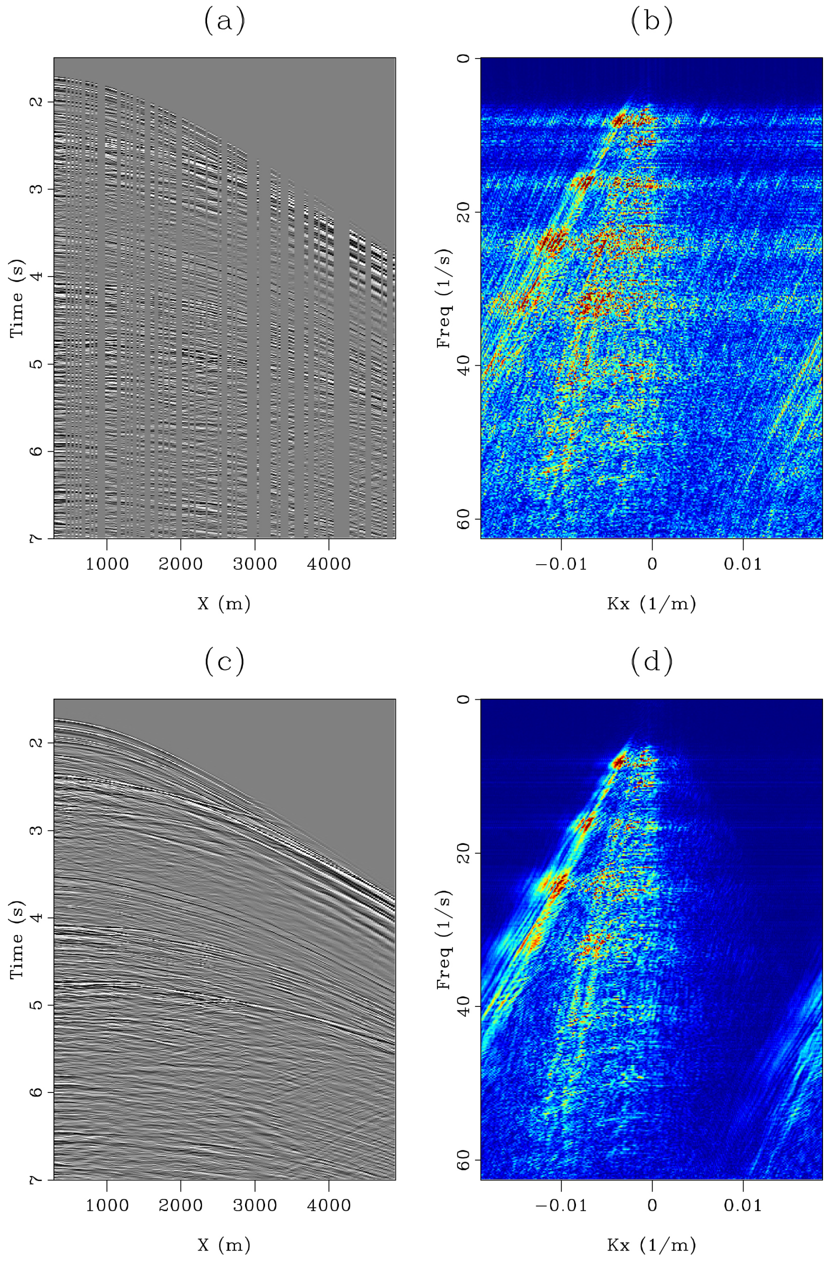


Figure 10: Interpolation of a shot gather from the Gulf of Mexico with irregular sampling. (a) Shows the input data binned onto a regular grid before interpolation and its corresponding FK spectrum in (b). Interpolation results are shown in (c) (FK spectrum in (d)). The missing traces are reconstructed and there is no noticeable footprint left by the interpolation algorithm. **[NR]** **antoine1/. irregular-gm**

REFERENCES

Abma, R. and N. Kabir, 2006, 3D interpolation of irregular data with a POCS algorithm: Geophysics, **71**, E91–E97.

Claerbout, J., 1992, Earth Sounding Analysis, Processing versus Inversion: Blackwell Scientific Publications.

Claerbout, J. and S. Fomel, 2002, Image Estimation by Example: Geophysical Soundings Image Construction: Class notes, http://sepwww.stanford.edu/sep/prof/index.html.

Clapp, R. G., B. L. Biondi, and J. F. Claerbout, 2004, Incorporating geologic information into reflection tomography: Geophysics, **69**, 533–546.

Crawley, S., R. Clapp, and J. Claerbout, 1999, Interpolation with smoothly nonstationary prediction-error filters: SEG Technical Program Expanded Abstracts, **18**, 1154–1157.

Fomel, S., 2002, Applications of plane-wave destruction filters: Geophysics, **67**, 1946–1960.

Fomel, S. and A. Guitton, 2006, Regularizing seismic inverse problems by model reparameterization using plane-wave construction: Geophysics, **71**, A43–A47.

Guitton, A. and J. Claerbout, 2004, Interpolation of bathymetry data from the Sea of Galilee: A noise attenuation problem: Geophysics, **69**, 608–616.

Herrmann, F. J., C. R. Brown, Y. A. Erlangga, and P. P. Moghaddam, 2009, Curvelet-based migration preconditioning and scaling: Geophysics, **74**, A41–A46.

Hung, B., C. Notfors, and S. Ronen, 2004, Seismic trace interpolation using the pyramid transform: SEG Technical Program Expanded Abstracts, **23**, 2017–2020.

Soubaras, R., 1994, Signal-preserving random noise attenuation by the F-X projection: SEG Technical Program Expanded Abstracts, **13**, 1576–1579.

Spitz, S., 1991, Seismic trace interpolation in the F-X domain: Geophysics, **56**, 785–794.

Sun, Y. and S. Ronen, 1996, The pyramid transform and its application to signal/noise separation: SEP-Report, **93**, 161–176.

Symes, W. W., 1994, The plane-wave detection problem: Inverse Problems, **10**, 1361–1391.

Trad, D., T. Ulrych, and M. Sacchi, 2003, Latest views of the sparse Radon transform: Geophysics, **68**, 386–399.

Xu, S., Y. Zhang, D. Pham, and G. Lambaré, 2005, Antileakage Fourier transform for seismic data regularization: Geophysics, **70**, V87–V95.

Zwartjes, P. M. and M. D. Sacchi, 2007, Fourier reconstruction of nonuniformly sampled, aliased seismic data: Geophysics, **72**, V21–V32.

Schoenberg's angle on fractures and anisotropy: A study in orthotropy

James G. Berryman

ABSTRACT

For vertical-fracture sets at arbitrary orientation angles to each other – but not perfectly randomly oriented, I present a detailed model in which the resulting anisotropic fractured medium generally has orthorhombic symmetry overall. Analysis methods of Schoenberg are emphasized, together with their connections to other similarly motivated and conceptually related methods by Sayers and Kachanov, among others. Examples show how parallel vertical fracture sets having HTI symmetry turn into orthotropic fractured media if some subsets of the vertical fractures are misaligned with the others, and then the fractured system can have VTI symmetry if all the fractures are aligned either randomly, or half parallel and half perpendicular to a given vertical plane. Another orthotropic case of vertical fractures in an otherwise VTI earth system treated previously by Schoenberg and Helbig is compared to, and contrasted with, other examples treated here.

INTRODUCTION

The present work covers various issues related to fractures and anisotropy, especially in relation to some of the published work of Michael Schoenberg (Schoenberg, 1980; Schoenberg and Muir, 1989; Schoenberg and Sayers, 1995; Schoenberg and Helbig, 1997). Details of methods presented here will also make use of an approach outlined by Sayers and Kachanov (1991), and used previously by Berryman (2006, 2008, 2009) in a recent series of published papers. [However, it is important to recognize that in earlier work it has also been shown (Berryman, 2006) that the general results of Bakulin et al. (2000) (for example) for the Thomsen (1986, 2002) weak anisotropy seismic parameters and contained in their Figure 6, are both qualitatively and even (reasonably) quantitatively consistent with each other, as well as being consistent with results from the method of Sayers and Kachanov (1991) treated explicitly here.] Thus, a high degree of consistency has been established among fracture-influence results that are based in part on the linear-slip model of fractures by Schoenberg (1980) and in part on penny-shaped (or approximately penny-shaped) cracks. [Also see Grechka et al. (2006).] This fact is important to the theme of the paper, because it shows that the details are often less important than the grand scheme of how fractures affect both the elastic system response and the wave propagation results.

FRACTURE ANALYSIS

For waves propagating in the $[x_1\text{-}x_3]$-plane with wavenumbers $k_1 = k\sin\theta$ and $k_3 = k\cos\theta$ where $k^2 = k_1^2 + k_3^2$, Tsvankin (1997) shows that we have the following equations [patterned here after the notation of Berryman (1979)]:

$$\rho\omega_\pm^2 = \frac{1}{2}\left[(c_{11}+c_{55})k_1^2 + (c_{33}+c_{55})k_3^2 \pm R\right], \tag{1}$$

where

$$R \equiv \sqrt{\left[(c_{11}-c_{55})k_1^2 - (c_{33}-c_{55})k_3^2\right]^2 + 4(c_{13}+c_{55})^2k_1^2k_3^2} \tag{2}$$

and where ρ (with no subscript) is the inertial density. Equation (1) determines the two wave speeds

$$V_\pm^2 = \frac{\omega_\pm^2}{k^2}. \tag{3}$$

The quantities $\omega_\pm$ have dimensions of angular frequency, but they are introduced mostly to simplify the form of the equations. The pertinent phase speeds V_+ for quasi-P-waves and V_- for quasi-SV-waves are given respectively by values corresponding to the + and − subscripts in the velocity equation (3). Group velocities [Brillouin (1946); Rüger (2002); Tsvankin (2005)] can be computed relatively easily as well when using the methods outlined earlier by Berryman (1979).

Sayers and Kachanov (1991) consider a model with two sets of possibly nonorthogonal fractures, also possibly having two different fracture density values ρ_a and ρ_b. Total fracture density is therefore $\rho_f = \rho_a + \rho_b$. These authors found that the pertinent fracture influence parameters were multiplied in this situation, when the angle between the fracture sets is ϕ, either solely by ρ_f itself or by one of the two factors:

$$\begin{aligned} A &= \rho_f + \left[\rho_f^2 - 4\rho_a\rho_b\sin^2\phi\right]^{1/2}, \\ B &= \rho_f - \left[\rho_f^2 - 4\rho_a\rho_b\sin^2\phi\right]^{1/2}. \end{aligned} \tag{4}$$

Table 1 shows the Sayers and Kachanov (1991) results for corrections to the isotropic background values of compliance (in Voigt 6×6 matrix notation — the original paper had results expressed in terms of tensor notation). Those background values are specfically for one model considered having Poisson's ratio $\nu = 0.4375$ (dimensionless), effective bulk modulus $K = 16.87$, shear modulus $\mu = 2.20$, and Young's modulus $E = 6.325$, with all moduli measured in units of GPa. For the assumed inertial density $\rho = 2200.0$ kg/m^3, the resulting isotropic background compressional wave speed is $V_p = 3.0$ km/s and shear wave speed is $V_s = 1.0$ km/s. For our computations, we also need the isotropic background compliance values, which are $S_{11} = S_{22} = S_{33} = 6.325$, $S_{12} = S_{13} = S_{23} = -2.767$, and $S_{44} = S_{55} = S_{66} = 0.4545$. The fracture influence factors η_1 and η_2, found for this specific model by Berryman and Grechka (2006), are displayed in Table 2. Some higher order fracture-influence factors were also obtained in the earlier work, but I will not be considering such factors in this short paper.

In the examples that follow, I will consider only the case of equal fracture densities $\rho_a = \rho_b = \rho_f/2$. For this somewhat simpler situation, I also have

$$\begin{aligned} A &= \rho_f(1+\cos\phi), \\ B &= \rho_f(1-\cos\phi). \end{aligned} \tag{5}$$

There may also be some uncertainty about exactly which of these factors is which in this degenerate case, because of the sign ambiguity in taking the square root of $\cos^2\phi$. But I will not concern myself with this detail here.

ΔS_{11}	$=$	$(\eta_1+\eta_2)A$
ΔS_{22}	$=$	$(\eta_1+\eta_2)B$
ΔS_{33}	$=$	0
ΔS_{12}	$=$	$\eta_1\rho_f$
ΔS_{13}	$=$	$\eta_1 A/2$
ΔS_{23}	$=$	$\eta_1 B/2$
ΔS_{44}	$=$	$\eta_2 B$
ΔS_{55}	$=$	$\eta_2 A$
ΔS_{66}	$=$	$2\eta_2\rho_f$

Table 1: Compliance matrix correction values for the vertical fracture model considered in Eq. 4, which are also true for the specific limit of Eq. 5, as a special case of the general result.

Fracture parameter	GPa^{-1}
η_1	-0.0192
η_2	0.3944

Table 2: Fracture-influence parameters (see Table 1 for usage) in a model reservoir having isotropic background with Poisson's ratio $\nu = 0.4375$, $V_p = 3.0$ km/s, and $V_s = 1.0$ km/s.

VANISHING OF THE ANELLIPTICITY PARAMETERS

One observation made immediately upon computing sample results for the model specified here is that the quasi-SV-wave propagating in the $[x_1\text{-}x_3]$-plane apparently has constant (or very nearly constant to numerical accuracy) wave speed at all angles in this plane (see Table 3). This result is startling when first seen, but it has been remarked upon previously in the literature by Gassmann (1964) and Schoenberg and Sayers (1995). One useful interpretation of this fact is obtained by noting that for quasi-SV-waves to have constant velocity in the plane, it is necessary for the pertinent effective anellipticity parameter to vanish for this plane of propagation. If it does not vanish identically, then it must at least vanish to first order in the fracture dependent correction factors (shown here in Table 1) in order to explain the numerical results for the relatively small fracture densities considered here. The condition required for exact vanishing of the pertinent anellipticity parameter in terms of stiffness coefficients turns out to be:

$$c_{11}c_{33} - c_{13}^2 = c_{55}\left(c_{11} + c_{33} + 2c_{13}\right), \tag{6}$$

as was already known to Gassmann (1964). But going farther in the analysis, it takes a fair amount of algebra to show that — to first order in the correction factors — the result (6) amounts to a condition relating compliance correction factors:

$$\Delta S_{55} = \Delta S_{11} + \Delta S_{33} - 2\Delta S_{13}. \tag{7}$$

This condition holds when we can ignore the higher order terms $O(\Delta^2)$, as well as all still higher orders (think of the proportionality $\Delta \propto \rho_f$). Then, we find that (7) is in fact satisfied identically when the expressions in Table 1 are substituted. It is important to notice as well that exact satisfaction of the condition in (7) is true for the general form of the definitions in Table 1, and not just for the restricted definitions used in the examples we computed, where the simplified definitions of (5) were employed to reduce our bookkeeping load. So the result in (7) is more general than just the specific examples I have computed. But the result is certainly not expected to be true for arbitrary compliance matrices. Nevertheless, it does seem to be true for a wide range of compliance matrices having vertical fractures in an otherwise isotropic earth.

ϕ	V_{sv} (km/s)
0^o	0.8602
30^o	0.8678
45^o	0.8771
60^o	0.8896
90^o	0.9222

Table 3: Constant V_{sv} wave speeds in the [x_1-x_3]-plane found for various values of the angle ϕ between fracture planes, and for the fixed value of fracture density $\rho_f = 0.20$.

This result surely seems quite interesting all by itself, but the curiously symmetric nature of the full fracture model can be highlighted further by noting that the following two expressions analogous to (7):

$$\Delta S_{44} = \Delta S_{22} + \Delta S_{33} - 2\Delta S_{23}, \tag{8}$$

and

$$\Delta S_{66} = \Delta S_{11} + \Delta S_{22} - 2\Delta S_{12}, \tag{9}$$

which are also both satisfied identically by the same set of expressions found in Table 1. These facts indicate that the model also has vanishing anellipticity factors (at least to the precision at which we are working) in the other orthogonal planes of propagation [x_2-x_3] and [x_1-x_2], as well.

As is well-known [see Gassmann (1964)], vanishing of the anellipticity factors means that there will be no triplications of wave arrivals for these models from propagation in any of these planes we have considered. Triplications arise because of complications from taking the derivatives required to compute group velocity from the phase velocities quoted here [also see Berryman (1979) for examples]. Group velocity determines the wave speed of signals and/or pulses of seismic energy [see Brillouin (1946) for a discussion].

VERTICAL FRACTURES IN VTI EARTH

Another model in a very similar context that has been discussed frequently by Schoenberg and Helbig (1997), Bakulin et al. (2000), and others is the model of a VTI earth system (where the background elastic medium is transversely isotropic with vertical axis of system, as it would be in a layered earth model having isotropic layers [Backus (1962)]) with superposed vertical fractures. A result that is often quoted in this context concerns a condition that is necessarily satisfied by the elastic stiffness matrix elements for such a system:

$$c_{13}(c_{22} + c_{12}) = c_{23}(c_{11} + c_{12}). \tag{10}$$

Using the same ideas applied here already, we can reduce this equation to a simple statement about the system compliances. The resulting statement is the formula:

$$S_{13} = S_{23}, \tag{11}$$

by which we mean to say that the only requirement imposed on the compliances after the introduction of the vertical fractures to the VTI earth background is that the new overall system compliance must satisfy the conditions in (11) after the changes due to the fractures are included in the values of these two compliance components. No other special constraints appear.

To see that (11) is the correct condition, note that

$$\begin{aligned} S_{13} &= (c_{12}c_{23} - c_{13}c_{22})/\det(C) \\ S_{23} &= (c_{12}c_{13} - c_{23}c_{11})/\det(C), \end{aligned} \tag{12}$$

where $\det(C)$ is the determinant of the upper left 3×3 sub-matrix of the orthotropic stiffness matrix C. Equating these two expressions from (12) and rearranging the final result gives a formula that is precisely the same condition (10) given by Schoenberg and Helbig (1997). What this condition implies for the physical system — since the background earth medium is assumed to be VTI with vertical axis of symmetry and also since $S_{13} = S_{23}$ for the background medium itself (before the fractures are added to it) — is that our final result must be the condition:

$$\Delta S_{13} = \Delta S_{23}. \tag{13}$$

This very simple equality means the changes (i.e., increases) in these off-diagonal compliances — when due to the addition of the vertical fractures to this model — have just one constraint, and that single constraint is that changes in these two off-diagonal compliances S_{13} and S_{23} must occur in unison. This result is also seen as a limiting case found in Table 1, when $A = B$, which occurs only when $\phi = 0^o$. Since $\phi = 0^o$ means that all the vertical fractures are parallel, and therefore being all aligned fractures, I therefore then have exactly the case studied explicitly by Schoenberg and Helbig (1997).

DISCUSSION OF VARIOUS FRACTURE MODELS

Among others, there are two main methods that are typically used in the seismic exploration literature for modeling the effects of fractures on seismic wave propagation: One is the method introduced originally by Schoenberg (1980) and the other was introduced at about the same time by Kachanov (1980). These two methods have both been used extensively in the exporation community, especially since the work of Sayers and Kachanov (1991) and Schoenberg and Sayers (1995). Connections, including many similarities and a few differences, are discussed in an overview by Schoenberg and Sayers (1995). I have some personal preference for the analytical version of Sayers and Kachanov (1991), because it is so explicit and also permits deep connections to be made to effective medium theory, especially through the work of Eshelby (1957). In particular, some recent and related work on fractures in outcrops using an effective medium theory approach by Berryman and Aydin (2009) has made use of the approach put forward by Sayers and Kachanov (1991) for modeling higher fracture density media. This study might have been more difficult to carry through using the original Schoenberg (1980) approach. But, once these higher fracture density results were known, it became straightforward to incorporate them into the layer-averaging approach emphasized by Backus (1962) and Schoenberg and Muir (1989). Thus, the effective medium theory for layering was found to be most useful for providing a means of determining fracture-fracture interactions when the fracture sets are close, but not actually intersecting.

Effects of fluids on the fracture behavior have not been emphasized here, but this issue is obviously a very important one for our applications, and it has been treated in other recent work by Daley et al. (2006) and Berryman (2006). Effects that liquids have on elastic moduli may be incorporated fairly easily using results of Gassmann (1951) and Skempton (1954), as shown by Berryman (2006).

CONCLUSIONS

In this paper I have treated various methods for quantifying the geomechanical effects of fracture sets on reservoirs. Special emphasis has been given to recent work, and also to the influence that the work of Michael Schoenberg has had on this subject. I conclude that the methods currently in use provide a consistent and presumably quite accurate picture of the influence of fractures on wave propagation in many cases, and that the various methods in use, although sometimes presented quite differently, are actually very closely related both conceptually and also in the quantitative details of their predictions.

Of course, elastic orthotropy is not universal in the earth [see Sayers et al. (2009)], so we should not and I do not assume that all the problems in exploration seismology will be solved by such relatively simple models. One final conclusion is that, although good progress has been made, there is also clearly more work to be done relative to the influence of fractures and anisotropy on seismic waves.

ACKNOWLEDGMENTS

I thank Atilla Aydin, Vladimir Grechka, Mark Kachanov, Francis Muir, Seiji Nakagawa, Steven R. Pride, Colin Sayers, Leon Thomsen, Ilya Tsvankin, and especially Michael Schoenberg — to whose memory this paper is dedicated — for many helpful, stimulating, and enlightening discussions about fractures and anisotropy.

REFERENCES

Backus, G. E., 1962, Long-wave elastic anisotropy produced by horizontal layering: Journal of Geophysical Research, **67**, 4427–4440.

Bakulin, A., V. Grechka, and I. Tsvankin, 2000, Estimation of fracture parameters from reflection seismic data – Part I: HTI model due to a single fracture set: Geophysics, **65**, 1788–1802.

Berryman, J. G., 1979, Long-wave elastic anisotropy in transversely isotropic media: Geophysics, **44**, 896–917.

———, 2006, Seismic waves in rocks with fluids and fractures: Geophysical Journal International, **171**, 954–974.

———, 2008, Exact seismic velocities for transversely isotropic media and extended Thomsen formulas for stronger anisotropies: Geophysics, **73**, D1–D10.

———, 2009, Aligned vertical fractures, HTI reservoir symmetry and Thomsen seismic anisotropy parameters for polar media: Geophysical Prospecting, **57**, no. 2, 193–208.

Berryman, J. G. and A. Aydin, 2009, Quasi-static analysis of elastic behavior for some systems having higher fracture densities. submitted for publication 2009.

Berryman, J. G. and V. Grechka, 2006, Random polycrystals of grains containing cracks: Model of quasistatic elastic behavior for fractured systems: Journal of Applied Physics, **100**, 113527.

Brillouin, L., 1946, Wave Propagation in Periodic Structures: Dover.

Daley, T. M., M. Schoenberg, J. Rutqvist, and K. T. Nihei, 2006, Fractured reservoirs: An analysis of coupled elastodynamic and permeability changes from pore-pressure variation: Geophysics, **71**, no. 5, O33–O41.

Eshelby, J. D., 1957, The determination of the elastic field of an ellipsoidal inclusion, and related problems: Proceedings of the Royal Society of London A, **241**, no. 1226, 376–396.

Gassmann, F., 1951, Über die Elastizität poröser Medien: Vierteljahrsschrift der Naturforschenden Gesellschaft in Zürich, **96**, 1–23.

———, 1964, Introduction to seismic travel-time methods in anisotropic media: Pure and Applied Geophysics, **58**, 53–112.

Grechka, V., I. Vasconcelos, and M. Kachanov, 2006, Influence of crack shape on the effective elasticity of fractured rocks: Geophysics, **71**, no. 5, D153–D160.

Kachanov, M., 1980, Continuum model of medium with cracks: Journal of the Engineering Mechanics Division of ASCE, **106**, 1039–1051.

Rüger, A., 2002, Reflection Coefficients and Azimuthal AVO Analysis in Anisotropic Media: Society of Exploration Geophysicists, Tulsa, OK.

Sayers, C. M. and M. Kachanov, 1991, A simple technique for finding effective elastic constants of cracked solids for arbitrary crack orientation statistics: International Journal of Solids and Structures, **27**, 671–680.

Sayers, C. M., A. D. Taleghani, and J. Adachi, 2009, The effect of mineralization on the ratio of normal to tangential compliance of fractures: Geophysical Prospecting, **57**, no. 3, 439–446.

Schoenberg, M., 1980, Elastic wave behavior across linear slip interfaces: Journal of the Acoustical Society of America, **68**, 1516–1521.

Schoenberg, M. and K. Helbig, 1997, Orthorhombic media: Modeling elastic wave behavior in a vertically fractured earth: Geophysics, **62**, no. 6, 1954–1974.

Schoenberg, M. and F. Muir, 1989, A calculus for finely layered anisotropic media: Geophysics, **54**, no. 4, 581–589.

Schoenberg, M. and C. M. Sayers, 1995, Seismic anisotropy of fractured rock: Geophysics, **60**, no. 1, 204–211.

Skempton, A. W., 1954, The pore-pressure coefficients A and B: Geotechnique, **4**, 143–147.

Thomsen, L., 1986, Weak elastic anisotropy: Geophysics, **51**, 1954–1966.

———, 2002, Understanding Seismic Anisotropic in Exploration and Exploitation: Society of Exploration Geophysicists, Tulsa, OK.

Tsvankin, I., 1997, Anisotropic parameters and P-wave velocity for orthorhombic media: Geophysics, **72**, 1292–1309.

———, 2005, Seismic Signatures and Analysis of Reflection Data in Anisotropic Media: Elsevier, Oxford, UK.

Stanford Exploration Project, Report SEP139, October 19, 2009

Poroelastic measurements resulting in complete data sets for granular and other anisotropic porous media

James G. Berryman

ABSTRACT

Poroelastic analysis usually progresses from assumed knowledge of dry or drained porous media to the predicted behavior of fluid-saturated and undrained porous media. Unfortunately, the experimental situation is often incompatible with these assumptions, especially when field data (from hydrological or oil/gas reservoirs) are involved. The present work considers several different experimental scenarios typified by one in which a set of undrained poroelastic (stiffness) constants has been measured using either ultrasound or seismic wave analysis, while some or all of the dry or drained constants are normally unknown. Drained constants for such a poroelastic system can be deduced for isotropic systems from available data if a complete set of undrained compliance data for the principal stresses is available, together with a few other commonly measured quantities such as porosity, fluid bulk modulus, and grain bulk modulus. Similar results are also developed here for anisotropic systems having up to orthotropic symmetry if the system is granular (*i.e.*, composed of solid grains assembled into a solid matrix, either by a cementation process or by applied stress) and the grains are known to be elastically homogeneous. Finally, the analysis is also fully developed for anisotropic systems with nonhomogeneous (more than one mineral type), but still isotropic, grains — as well as for uniform collections of anisotropic grains as long as their axes of symmetry are either perfectly aligned or perfectly random.

INTRODUCTION

Poroelastic analysis (Gassmann, 1951; Biot and Willis, 1957; Brown and Korringa, 1975; Rice and Cleary, 1976; Thigpen and Berryman, 1985; Cheng, 1997; Wang, 2000) usually progresses from assumed knowledge of dry or drained porous media to the predicted behavior of fluid-saturated and undrained porous media. When field data (say from oil, gas, or hydrological reservoirs) are involved, the experimental situation is generally and unfortunately incompatible with these assumptions.

For this reason, I want to consider several different experimental scenarios typified by one in which a set of undrained constants has been measured using either ultrasound (in the laboratory) or seismic wave analysis (for field data), while some or all of the dry or drained constants are usually unmeasured and therefore unknown. Drained constants for such a poroelastic system can be deduced from available data if a complete set of undrained compliance data (as could be calculated by inverting the stiffness data, which are almost directly obtained, within a factor of the density, from wave speed measurements) is available, together with a few other commonly measured quantities such as porosity, fluid bulk modulus, and grain bulk modulus for isotropic systems. Similar results are developed here for anisotropic systems having up to orthotropic symmetry if the system is granular (*i.e.*, composed of solid grains assembled into solid either by a cementation process or by applied stress) and the grains are known to be elastically homogeneous.

In the later sections, the analysis is also fully developed for anisotropic systems with inhomogeneous (meaning more than one mineral type is present), but still isotropic, grains. Also studied is the case for uniform collections of the same types of anisotropic grains, as long as the grain symmetry axes are either perfectly aligned or perfectly random. I show how many poroelastic data are needed in order to consider the data sets complete, and which types of data are in some sense redundant. Some combinations can be used to replace other data types that remain missing when lab experimental and/or field limitations prevent direct measurements of all the poroelastic coefficients.

ISOTROPIC POROELASTICITY

Homogeneous grains

The famous equation for undrained bulk modulus named for Gassmann (1951) can be written in the form

$$K^u = K^d + \alpha^2/[(\alpha - \phi)/K^g + \phi/K_f] \tag{1}$$

for isotropic systems, where $\alpha \equiv 1 - K^d/K^g$ is the Biot-Willis coefficient or effective stress (Biot and Willis, 1957) coefficient, K^g is the solid modulus of the grains (assumed homogeneous), K^d is the drained modulus of the porous medium, K_f is the pore fluid modulus, and ϕ is the porosity. The formula becomes more complicated if the solids constituting the porous medium are heterogeneous. But we will delay discussion of this point to the next subsection and for now assume that the solids are truly homogeneous. For notational convenience, I next introduce a modulus for a fluid suspension having the same solid and fluid components as well as the same porosity, but having drained modulus $K^d \equiv 0$. Then I find that the effective modulus is given by

$$K_{susp} = \left[\frac{1-\phi}{K^g} + \frac{\phi}{K_f}\right]^{-1}. \tag{2}$$

In fact this result follows directly from Gassmann's formula (1) by setting $K^d = 0$ everywhere, since then $K^u = K_{susp}$. But of course this result is also well-known in mechanics and acoustics (Wood, 1955) for these types of fluid-solid suspensions.

Deducing drained moduli from undrained: Isotropic system with homogeneous grains

Rewriting Gassmann's formula in these terms, I first find that

$$K^u = K^d + \frac{(1 - K^d/K^g)^2}{1/K_{susp} - K^d/(K^g)^2}. \tag{3}$$

Note that all explicit porosity (ϕ) dependence is now imbedded in the modulus K_{susp}. Now if I simply multiply through by the denominator on the right hand side, then I find

$$K^u\left(\frac{1}{K_{susp}} - \frac{K^d}{(K^g)^2}\right) = 1 - 2\frac{K^d}{K^g} + \frac{K^d}{K_{susp}}. \tag{4}$$

Also notice that two terms of the form $(K^d/K^g)^2$ have cancelled from this expression. Once these convenient cancellations have occurred, K^d appears only linearly in the resulting expression. The equation can therefore be solved immediately for drained modulus K^d in terms of the undrained modulus K^u and the other factors that are also assumed to be known (and in fact these other factors are usually easier to measure than either K^u or K^d). Finally, I obtain:

$$K^d = \left(\frac{K^u}{K_{susp}} - 1\right)\left[1/K_{susp} - 2/K^g + K^u/(K^g)^2\right]^{-1}. \tag{5}$$

This result shows that the drained modulus can be deduced from measurements of the undrained modulus, together with knowledge of ϕ, K_f, and K^g. Note that this result was first derived by Zhu and McMechan (1990), but apparently published only in an SEG conference proceedings.

Heterogeneous grains

When the grains in a granular packing are no longer composed of elastically homogeneous and isotropic materials, or if they are homogeneous but anisotropic while nevertheless being distributed in a randomly oriented way in space, then — as has been pointed out previously in Brown and Korringa (1975), Rice and Cleary (1976), and the work of others [*e.g.*, Wang (2000)] — I need to introduce a more general notation to deal with these circumstances.

Recall that the Reuss (1929) average of the grain bulk moduli for a heterogeneous medium with a distribution of grain types is given by:

$$\frac{1}{K_R^g} \equiv \sum_{m=1,\ldots,n} \frac{v_m}{K_m}, \tag{6}$$

where v_m is the volume fraction (out of all the solid material present, so that $\sum_m v_m = 1$) of the m-th isotropic grain having bulk modulus K_m. This average should be distinguished from that of the Voigt (1928) average

$$K_V^g \equiv \sum_{m=1,\ldots,n} v_m K_m, \tag{7}$$

which is known (Hill, 1952) to satisfy $K_V^g \geq K_R^g$. Furthermore, these two measures are also known (Reuss, 1929; Voigt, 1928; Hill, 1952) to satisfy $K_V^g \geq K_g^* \geq K_R^g$, where K_g^* is the effective bulk modulus of an isotropic elastic composite consisting only of the minerals $m = 1, \ldots, n$ in the same volume proportions given by the v_m values. However, this fact actually is not pertinent here, as the only averages of this type that play a direct role in the poroelastic equations are always those of the Reuss-type, as will be shown in the further developments.

To clarify later usage of the same notation K_R^g, I emphasize here that when (or if) the grains in our assemblage are all anisotropic — but nevertheless of the same type and oriented randomly in space — then the pertinent average is again the Reuss average. But in this case the average is determined by the equation

$$\frac{1}{K_R^g} = \sum_{i,j=1,2,3} s_{ij}^g, \tag{8}$$

where the s^g_{ij} for $i, j = 1, 2, 3$ are the principal components of the compliance matrix for the anisotropic grain material itself. It is easy to see that this must be the case by referring back to the equations above, specifically those requiring the suspension result K_{susp}. The formula as quoted in (2) was only written for the case of homogeneous grains. But if we generalize this formula slightly as:

$$K_{susp} \equiv \left[\frac{1-\phi}{K^g_R} + \frac{\phi}{K_f}\right]^{-1}, \tag{9}$$

then we see that it holds equally true: (a) for homogeneous isotropic grains (when $K^g_R \equiv K^g$), (b) for heterogeneous volumes of isotropic grains [when K^g_R is given by (6)], or (c) for anisotropic grains when they are randomly oriented in the fluid [and then K^g_R is given by (8)]. In all these cases, I do assume that this mixture of grains and fluid is close to being a true suspension, by which we mean that individual grains are acted on similarly by changes in fluid pressure.

If the clumpings are loose enough, then the fluid can act equally on all the individual grains, and the result in (9) holds true regardless of the heterogeneity. However, if this is not the case, then there must be elastically distinct clumpings of grains forming solid composites locally – so the individual grains are no longer uniformly surrounded by the pore fluid. Then, each grain's fluid environment is different, due to welded contacts with other contiguous grains. I am assuming for the present purposes that such effects are negligible in the types of comparatively homogeneous porous media (on the meso- and macroscales, but not necessarily on the microscale) being studied here. In fact, some types of more heterogeneous systems can be treated, and some of these have already been studied (Berryman and Milton, 1991; Berryman and Pride, 2002) when the porous system is composed of just two distinct types of grain clumpings; however, I will not be discussing such double-porosity and/or multi-porosity effects in the present paper.

Heterogeneous pores

Another important type of heterogeneity that can occur in practice involves heterogeneity of the pore space. One obvious issue is whether the pores are all connected to each other, or whether there may be two (or more) distinct, but intertwining, pore systems. One well-known example of this situation is the double-porosity concept (Berryman and Pride, 2002; Gurevich et al., 2009; Barenblatt and Zheltov, 1960), in which one type of pore has high volume but low permeability, while the other has low volume (imagine a system of very flat cracks or fractures) and high permeability. I can also consider that some pores might be interior to some grains and not connected to any other pores (and might therefore also be empty of pore fluid), while other subsets of the grains have no inherent porosity of this type, and so are truly solid grains.

I will not try to deal with all these cases simultaneously, as even enumerating all the possibilities quickly becomes burdensome. I will limit myself instead to one of the more typical scenarios, considered for example by Brown and Korringa (1975) and by Rice and Cleary (1976) — and also see the recent related work of Gurevich et al. (2009).

Heterogeneity of the pore space is most important when considering flow of fluid into and out of the boundaries of a porous sample. Then, the concept of increment of fluid content ζ comes into play, and special care is required. A straightforward definition of this dimensionless parameter (just as the strains e_{11}, e_{22}, ..., e_{13} are all dimensionless) is given by:

$$\zeta \equiv \frac{\delta(\phi V) - \delta V_f}{V} \simeq \phi \left(\frac{\delta V_\phi}{V_\phi} - \frac{\delta V_f}{V_f} \right), \tag{10}$$

where V is the overall volume of the initially fully fluid-saturated porous material at the first instant of consideration, $V_\phi = \phi V$ is the pore volume, with ϕ being the fluid-saturated porosity of the volume, and V_f is the volume occupied by the pore-fluid, making $V_f = \phi V$ at the start. The δ's indicate small changes in the quantities following. For "drained" systems, there must be a reservoir of the same fluid just outside the volume V that can either supply more fluid or absorb any excreted fluid as needed during the nonstationary phase of the poroelastic process; the amount of pore fluid can therefore either increase or decrease from the initial amount of pore fluid, and at the same time the pore volume can also be changing, but not necessarily at exactly the same rate as the pore fluid itself. The one exception to these statements is when the surface pores of the total volume V are sealed, in which case the system is "undrained" and $\zeta \equiv 0$, identically. In these circumstances, it is still possible that V_f and $V_\phi = \phi V$ are both changing, but because of the imposed undrained boundary conditions, they are necessarily changing at the same rate. The result is that, for an isotropic system, I have:

$$\zeta = \phi \left[\frac{\delta \sigma_c}{K_p} + \delta p_f \left(\frac{1}{K_p} - \frac{1}{K_R^\phi} + \frac{1}{K_f} \right) \right], \tag{11}$$

and where the various moduli in (11) are defined by the following relations [see Brown and Korringa (1975)]:

$$-\frac{\delta V_f}{V_f} = \frac{\delta p_f}{K_f}, \tag{12}$$

$$-\frac{\delta V}{V} = \frac{\delta p_d}{K_R^d} + \frac{\delta p_f}{K_R^g}, \tag{13}$$

and

$$-\frac{\delta V_\phi}{V_\phi} = \frac{\delta p_d}{K_p} + \frac{\delta p_f}{K_R^\phi}. \tag{14}$$

The changes in fluid pressure and differential pressure are respectively δp_f and $\delta p_d \equiv \delta p_c - \delta p_f$, where $\delta p_c = -\delta \sigma_c$ is the uniform confining pressure, if the external confining pressure is uniform. If not, then this quantity is replaced in the definition of δp_c by $-\delta \sigma_m$, which is the change in the mean confining pressure and where $\sigma_m \equiv (\sigma_{11}+\sigma_{22}+\sigma_{33})/3$ is the definition of the mean principal stress. Clearly, if the confining principal stress is uniform ($\sigma_{11} = \sigma_{22} = \sigma_{33}$), then the mean stress equals this uniform confining stress. If not, then there can be additional shearing effects that need to taken into account, but these do not play any role in the changes of fluid content since this quantity is effectively a measure only of the total number of fluid particles contained in the pertinent pore volume.

It can also be shown using poroelastic reciprocity (and I will show this later as it very clearly develops in the following anisotropic analysis) that

$$\frac{\phi}{K_p} = \frac{\alpha_R}{K_R^d} = \frac{1}{K_R^d} - \frac{1}{K_R^g}. \tag{15}$$

So generalizing Gassmann's formula for undrained modulus gives:

$$\frac{1}{K_R^u} = \frac{1-\alpha_R B}{K_R^d}, \tag{16}$$

where $\alpha_R = 1 - K_R^d/K_R^g$ and

$$B = \left(\frac{1}{K_R^d} - \frac{1}{K_R^g}\right)\left[\left(\frac{1}{K_R^d} - \frac{1}{K_R^g}\right) + \phi\left(\frac{1}{K_f} - \frac{1}{K_R^\phi}\right)\right]^{-1} \tag{17}$$

is the second coefficient of Skempton (1954). Combining these terms, I find that the most general form of the equation for the undrained bulk modulus in the isotropic case is:

$$\frac{1}{K_R^u} = \frac{1}{K_R^d} - \left(\frac{1}{K_R^d} - \frac{1}{K_R^g}\right)^2\left[\left(\frac{1}{K_R^d} - \frac{1}{K_R^g}\right) + \phi\left(\frac{1}{K_f} - \frac{1}{K_R^\phi}\right)\right]^{-1}, \tag{18}$$

or, alternatively,

$$\frac{1}{K_R^u} = \frac{1}{K_R^d} - \frac{(\alpha_R/K_R^d)^2}{\alpha_R/K_R^d + \phi\left(\frac{1}{K_f} - \frac{1}{K_R^\phi}\right)}, \tag{19}$$

which is the isotropic result of Brown and Korringa (1975), and should also be compared directly to (1). So, if the pore modulus and grain modulus are equal with $K_R^\phi = K_R^g$, then (19) reduces exactly to (1). Although this result is the same as that of Brown and Korringa (1975), I nevertheless write it differently to emphasize different features.

Deducing drained constants from undrained: Heterogeneous grains and pores

I was able to deduce K_R^d from our knowledge of K_R^u, K_R^g, K_f, and ϕ in subsection 2.2. But even though I am still assuming the system is isotropic, I have now introduced some additional degrees of freedom by permitting the grains and pores to be heterogeneous. It is clear that I cannot deduce K_R^d if I just have the same amount of information as before. In particular, it does seem fairly straightforward to measure K_R^g, as I have already described its meaning in the earlier discussion and even given formulas for it — if I have information about the constituents and their volume fractions, or alternatively about the principal components of elastic compliance and/or stiffness matrices. But I have another variable now, which is the pore modulus K_R^ϕ, and this bulk modulus is not so easy either to model or to measure directly (Lockner and Stanchits, 2002). However, by adding one more piece of information — namely the second Skempton coefficient $B = p_f/p_c$, which is a fact that should typically be known in poroelastic systems — then it turns out that I can solve for both K_R^d and K_R^ϕ. Again, I assume that K_R^g and K_R^u are known. But now I also assume that B is also known experimentally. Working through the algebra, I find that

$$K_R^d = \frac{1-B}{1/K_R^u - B/K_R^g} \tag{20}$$

[which is a rearrangement of $K_R^u = K_R^d/(1-\alpha_R B)$], and similarly that

$$\frac{1}{K_R^\phi} = \frac{1}{K_f} - \left(\frac{1-B}{\phi B}\right)\left(\frac{1}{K_R^d} - \frac{1}{K_R^g}\right) = \frac{1}{K_f} - \left(\frac{1}{\phi B}\right)\left(\frac{1}{K_R^u} - \frac{1}{K_R^g}\right). \tag{21}$$

In (21), I used the previous result (20) for K_R^d to simplify the final formula.

These forms are very useful for many applications in poroelasticity, but so far they apply only to the fully isotropic case. I show next that a very similar set of formulas applies to the anisotropic cases under consideration. I am able to attain greater clarity at this point by switching to the more general anisotropic problem, where it can seen more easily how poroelastic reciprocity comes directly into play.

ANISOTROPIC POROELASTICITY

If the overall porous medium is anisotropic due either to some preferential alignment of the constituent particles or to externally imposed stress (such as a gravity field and weight of overburden, for example), I consider the orthorhombic anisotropic version of the poroelastic equations:

$$\begin{pmatrix} e_{11} \\ e_{22} \\ e_{33} \\ -\zeta \end{pmatrix} = \begin{pmatrix} s_{11} & s_{12} & s_{13} & -\beta_1 \\ s_{12} & s_{22} & s_{23} & -\beta_2 \\ s_{13} & s_{23} & s_{33} & -\beta_3 \\ -\beta_1 & -\beta_2 & -\beta_3 & \gamma \end{pmatrix} \begin{pmatrix} \sigma_{11} \\ \sigma_{22} \\ \sigma_{33} \\ -p_f \end{pmatrix}. \tag{22}$$

From here on throughout the paper, I will drop the δ's from the stresses and strains, as this extra notation is truly redundant when they are all being treated as small (and therefore resulting in linear effects), as I do here.

The e_{ii} are strains in the $i = 1, 2, 3$ directions. The σ_{ii} are the corresponding stresses. The fluid pressure is p_f. The increment of fluid content is ζ. The drained compliances are $s_{ij} = s_{ij}^d$. Undrained compliances (not yet shown) are symbolized by s_{ij}^u. Coefficients $\beta_i = s_{i1} + s_{i2} + s_{i3} - 1/3K_R^g$, where K_R^g is again the Reuss average modulus of the grains. The drained Reuss average bulk modulus is defined by

$$\frac{1}{K_R^d} = \sum_{ij=1,2,3} s_{ij}^d. \tag{23}$$

For the Reuss average undrained bulk modulus K_R^u, I have drained compliances replaced by undrained compliances. A similar definition of K_R^g, with drained compliances replaced by grain compliances has already been introduced earlier in the discussion. The alternative Voigt (1928) average [also see Hill (1952)] of the stiffnesses will play no role in the present work. And, finally, $\gamma = \sum_{i=1-3} \beta_i / BK_R^d$, where B is the second Skempton (1954) coefficient, which will be defined carefully again later in our discussion.

The shear terms due to twisting motions (*i.e.*, strains e_{23}, e_{31}, e_{12} and stresses σ_{23}, σ_{31}, σ_{12}) are excluded from this discussion since they typically do not couple to the modes of interest for anisotropic systems having orthotropic symmetry, or any more symmetric system such as transversely isotropic or isotropic. I have also assumed that the true axes of symmetry are known, and make use of them in my formulation of the problem. Note that the s_{ij}'s are the elements of the compliance matrix $\mathbf{S}$ and are all independent of the fluid, and therefore would be the same if the medium were treated as elastic (*i.e.*, by ignoring the fluid pressure, or assuming that the fluid saturant is air). In keeping with the earlier discussions, I typically call these compliances the drained compliances and the corresponding matrix the drained compliance matrix $\mathbf{S}^d$, since the fluids do not contribute to the stored mechanical energy if they are free to drain into a surrounding reservoir containing the same type of fluid. In contrast, the undrained compliance matrix $\mathbf{S}^u$ presupposes that the fluid is trapped (unable to drain from the system into an adjacent reservoir) and therefore contributes in a significant and measureable way to the compliance and stiffness ($\mathbf{C}^u = [\mathbf{S}^u]^{-1}$), and also therefore to the stored mechanical energy of the undrained system.

Although the significance of the formula is somewhat different now, I find again that

$$\beta_1 + \beta_2 + \beta_3 = \frac{1}{K_R^d} - \frac{1}{K_R^g} = \frac{\alpha_R}{K_R^d}, \tag{24}$$

if I also define (as I did for the isotropic case) a Reuss effective stress coefficient:

$$\alpha_R \equiv 1 - K_R^d/K_R^g. \tag{25}$$

Furthermore, I have

$$\gamma = \frac{\beta_1 + \beta_2 + \beta_3}{B} = \frac{\alpha_R}{K_R^d} + \phi\left(\frac{1}{K_f} - \frac{1}{K_R^\phi}\right), \tag{26}$$

since I have the rigorous result in this notation (Berryman, 1997) that Skempton's B coefficient is given by

$$B \equiv \frac{1 - K_R^d/K_R^u}{1 - K_R^d/K_R^g} = \frac{\alpha_R/K_R^d}{\alpha_R/K_R^d + \phi(1/K_f - 1/K_R^\phi)}. \tag{27}$$

Note that both (26) and (27) contain dependence on the pore bulk modulus K_R^ϕ that comes into play when the pores are heterogeneous, regardless of whether the system is isotropic or anisotropic. I want to emphasize that all these formulas are rigorous statements based on the earlier anisotropic analysis. The appearance of K_R^d and α_R is not an approximation, but merely a useful choice of notation made here because it will make clear the similarity between the rigorous anisotropic formulas and the isotropic ones.

The β_i coefficients

I will now provide several results for the β_i coefficients, and then follow the results with a general proof of their correctness.

In many important and useful cases, the coefficients β_i are determined by

$$\beta_i = s_{i1}^d + s_{i2}^d + s_{i3}^d - \frac{1}{3K_R^g}. \tag{28}$$

Again, K_R^g is the Reuss average of the grain modulus, since the local grain modulus is not necessarily assumed uniform here as mentioned previously. Equation (28) holds true for homogeneous grains, such that $K_R^g = K^g$. It also holds true for the case when K_R^g is determined instead by (6). However, when the grains themselves are anisotropic, I need to allow again for this possibility, and this can be accomplished by defining three directional grain bulk moduli determined by:

$$\frac{1}{3\overline{K}_i^g} \equiv s_{i1}^g + s_{i2}^g + s_{i3}^g = s_{1i}^g + s_{2i}^g + s_{3i}^g, \tag{29}$$

for $i = 1, 2, 3$. The second equality follows because the compliance matrix is always symmetric. I call these quantities in (29) the partial grain-compliance sums, and the $\overline{K}^g_i$ are the directional grain bulk moduli. Then, the formula for (28) is replaced by

$$\beta_i = s^d_{i1} + s^d_{i2} + s^d_{i3} - \frac{1}{3\overline{K}^g_i}. \tag{30}$$

Note that the factors of three have been correctly accounted for because

$$\sum_{i=1,2,3} \frac{1}{3\overline{K}^g_i} = \frac{1}{K^g_R}, \tag{31}$$

in agreement with (8). If the three contributions represented by (29) for $i = 1, 2, 3$ happen to be equal, then each equals one-third of the sum (31).

The preceding results are for perfectly aligned grains. If the grains are instead perfectly randomly oriented, then it is clear that the formulas in (28) hold as before, but now K^g_R is determined instead by (8).

All of these statements about the β_i are easily proven by considering the case when $\sigma_{11} = \sigma_{22} = \sigma_{33} = -p_c = -p_f$. In this situation, from (22), I have:

$$-e_{ii} = \left(s^d_{i1} + s^d_{i2} + s^d_{i3}\right) p_c + \beta_i(-p_f) = (s^g_{i1} + s^g_{i2} + s^g_{i3})\, p_f \equiv \frac{p_f}{3\overline{K}^g_i}, \tag{32}$$

in the most general of the three cases discussed, and holding true for each value of $i = 1, 2, 3$. This is a statement about the strain e_{ii} that would be observed in this situation, as it must be the same if these anisotropic (or inhomogeneous) grains were immersed in the fluid, while measurements were taken of the strains observed in each of the three directions $i = 1, 2, 3$, during variations of the fluid pressure p_f. Consider this proof to be a thought experiment for determining the coefficients, in the same spirit as those proposed originally by Biot and Willis [see Biot and Willis (1957); Stoll (1974)] for the isotropic and homogeneous case.

Coefficient γ

The relationship of coefficient γ to the other coefficients is easily established because I have already discussed the main issue, which involves determining the role of the various other constants contained in the Skempton (1954) coefficient B. I have quoted this result in (17).

Again, from (22), I find that

$$-\zeta = 0 = -\left(\beta_1 + \beta_2 + \beta_3\right) \sigma_c - \gamma p_f, \tag{33}$$

for undrained boundary conditions. Thus, I again have

$$\frac{p_f}{p_c} \equiv B = \frac{\beta_1 + \beta_2 + \beta_3}{\gamma}, \tag{34}$$

where $p_c = -\sigma_c$ is the confining pressure. Thus, the scalar coefficient γ is determined immediately and given by

$$\gamma = \frac{\beta_1 + \beta_2 + \beta_3}{B} = \frac{\alpha_R/K^d_R}{B} = \alpha_R/K^d_R + \phi\left(\frac{1}{K_f} - \frac{1}{K^\phi_R}\right). \tag{35}$$

Alternatively, I could say that

$$B = \frac{\alpha_R}{\gamma K_R^d}. \tag{36}$$

I have now determined the physical/mechanical significance of all the coefficients in the poroelastic matrix (22). These results are as general as possible without considering poroelastic symmetries that have less than orthotropic symmetry, while also taking advantage of my assumption that I do typically know (or can often determine) the three directions of the principal axes of symmetry.

Inverting poroelastic compliance

The matrix in (22) is in compliance form and has extremely simple poroelastic behavior in the sense that all the fluid mechanical effects appear only in the single coefficient γ. I can simplify the notation a little more by lumping some coefficients together, combining the 3×3 submatrix in the upper left corner of the matrix in (22) as $\mathbf{S}$, and defining the column vector $\mathbf{b}$ by

$$\mathbf{b}^T \equiv (\beta_1, \beta_2, \beta_3). \tag{37}$$

The resulting 4×4 matrix and its inverse are now related by:

$$\begin{pmatrix} \mathbf{S} & -\mathbf{b} \\ -\mathbf{b}^T & \gamma \end{pmatrix} = \begin{pmatrix} \mathbf{A} & \mathbf{q} \\ \mathbf{q}^T & z \end{pmatrix}^{-1}, \tag{38}$$

where the elements of the inverse matrix can be shown to be written in terms of drained stiffness matrix $\mathbf{C}^d = \mathbf{C} = \mathbf{S}^{-1}$ by introducing three components: (a) scalar $z = \left[\gamma - \mathbf{b}^T\mathbf{C}\mathbf{b}\right]^{-1}$, (b) column vector $\mathbf{q} = z\mathbf{C}\mathbf{b}$, and (c) undrained 3×3 stiffness matrix (*i.e.*, the pertinent one connecting the principal strains to principal stresses) $\mathbf{A} = \mathbf{C} + z\mathbf{C}\mathbf{b}\mathbf{b}^T\mathbf{C} = \mathbf{C}^d + z^{-1}\mathbf{q}\mathbf{q}^T \equiv \mathbf{C}^u$, since $\mathbf{C}^d$ is drained stiffness and $\mathbf{A} = \mathbf{C}^u$ is clearly undrained stiffness by construction. This result is the same as that of Gassmann (1951) for anisotropic porous media, although his results were presented in a form somewhat harder to scan than the form shown here.

Also, note the important fact that the observed decoupling of the fluid effects occurs only in the compliance form (22) of the equations, and never in the stiffness (inverse) form for the poroelasticity equations.

From these results, it is not hard to show that

$$\mathbf{S}^d = \mathbf{S}^u + \gamma^{-1}\mathbf{b}\mathbf{b}^T. \tag{39}$$

This result emphasizes the remarkably simple fact that the drained compliance matrix can be found directly from knowledge of the inverse of undrained compliance, and the still unknown, but sometimes relatively easy to estimate, values of γ, together with the three distinct orthotropic β_i coefficients, for $i = 1, 2, 3$.

TABLE 1. Reuss (R), Voigt (V), and self-consistent effective ($*$) bulk moduli of various common anisotropic materials (Berryman, 2005): Water ice, cadmium, zinc, graphite, α-quartz, corundum, barium titanate, rutile, aluminum, copper, magnesia, spinel. Full references for the data used in both TABLES 1 and 2 are provided in Berryman (2005). Units of bulk modulus K are GPa.

Material	Symmetry	K_R	K^*	K_V	K_V/K_R
H_2O	Hexagonal	8.89	8.89	8.89	1.00
Cd	Hexagonal	48.8	54.7	58.1	1.19
Zn	Hexagonal	61.6	70.9	75.1	1.22
Graphite	Hexagonal	35.8	88.0	286.3	8.00
Al_2O_3	Trigonal	253.5	253.7	253.9	1.002
α-SiO_2	Trigonal	37.6	37.8	38.1	1.01
TiO_2	Tetragonal	209	213	218	1.04
$BaTiO_2$	Tetragonal	163.1	179.3	186.8	1.15
Al	Cubic	76.3	76.3	76.3	1.00
MgO	Cubic	162.4	162.4	162.4	1.00
$MgAl_2O_4$	Cubic	196.7	196.7	196.7	1.00
Cu	Cubic	138.0	138.0	138.0	1.00

Deducing coefficients from measurements: Anisotropic example with homogeneous grains

Now, further progress is made by considering the Reuss average again for both of the orthotropic drained and undrained compliances:

$$\frac{1}{K_R^d} \equiv \sum_{i,j=1,2,3} s_{ij}^d, \tag{40}$$

and

$$\frac{1}{K_R^u} \equiv \sum_{i,j=1,2,3} s_{ij}^u. \tag{41}$$

These effective moduli are the Reuss averages of the nine compliances in the upper left 3×3 of the full (including the uncoupled shear components) 6×6 compliance matrix for the two cases, respectively, when the pore fluid is allowed to drain from the porous system, and when the pore fluid is trapped by a jacketing material and therefore undrained.

TABLE 2. Reuss (R), Voigt (V), and self-consistent effective ($*$) shear moduli of various common materials [see Berryman (2005)]: Water ice, cadmium, zinc, graphite, α-quartz, corundum, barium titanate, rutile, aluminum, copper, magnesia, and spinel. Units of shear modulus G are GPa. The anisotropy parameter $\mathcal{A} \equiv 5\frac{G_V}{G_R} + \frac{K_V}{K_R} - 6$ [from Ranganathan and Ostoja-Starzewski (2008a,b)].

Material	Symmetry	G_R	G^*	G_V	G_V/G_R	$\mathcal{A}$
H_2O	Hexagonal	3.48	3.52	3.55	1.02	0.10
Cd	Hexagonal	22.1	24.3	26.4	1.19	1.14
Zn	Hexagonal	34.1	40.6	44.8	1.31	1.77
Graphite	Hexagonal	9.2	52.6	219.4	23.8	121.0
Al_2O_3	Trigonal	160.7	163.1	165.5	1.03	0.15
α-SiO_2	Trigonal	41.0	44.0	47.6	1.16	0.81
TiO_2	Tetragonal	99.5	114.5	124.9	1.26	1.34
$BaTiO_2$	Tetragonal	47.4	53.6	59.8	1.26	1.46
Al	Cubic	26.0	26.2	26.3	1.01	0.05
MgO	Cubic	123.9	126.3	128.6	1.04	0.20
$MgAl_2O_4$	Cubic	98.6	109.0	118.0	1.20	1.00
Cu	Cubic	40.0	46.3	51.3	1.28	1.41

Although the significance of the formula in the anisotropic case is somewhat different now, I find again that

$$\beta_1 + \beta_2 + \beta_3 = \frac{1}{K_R^d} - \frac{1}{K_R^g} = \frac{\alpha_R}{K_R^d}, \tag{42}$$

if I also define a Reuss effective stress coefficient:

$$\alpha_R \equiv 1 - K_R^d/K_R^g, \tag{43}$$

by analogy to the isotropic case. Furthermore, I have

$$\gamma = \frac{\beta_1 + \beta_2 + \beta_3}{B} = \frac{\alpha_R}{K_R^d} + \phi\left(\frac{1}{K_f} - \frac{1}{K_R^g}\right), \tag{44}$$

since I have the rigorous result in this notation (Berryman, 1997) that Skempton's B coefficient (Skempton, 1954) is given by

$$B \equiv \frac{1 - K_R^d/K_R^u}{1 - K_R^d/K_R^g} = \frac{\alpha_R/K_R^d}{\alpha_R/K_R^d + \phi(1/K_f - 1/K_R^g)}. \tag{45}$$

I want to emphasize that all these formulas are rigorous statements based on the earlier anisotropic analysis. The appearance of K_R^d and α_R is not an approximation. In fact it is important now in the anisotropic case (but not in the isotropic cases considered earlier as long as the grains were also homogeneous) to make this distinction between the Reuss and Voigt averages. Making this choice of notation will help demonstrate useful analogies between the rigorous anisotropic formulas and the isotropic ones. I have prepared the way for these analogies by using the Reuss averages in earlier notation, even though they were mostly redundant in those isotropic examples.

First note that, from (42) and (44), it follows that $\gamma^{-1} = \frac{BK_R^d}{\alpha_R}$ — also see (36). So I can now rearrange (39) to give the formal relationship

$$s_{ij}^d = s_{ij}^u + \frac{BK_R^d}{\alpha_R}\beta_i\beta_j, \quad \text{for} \quad i,j = 1,2,3. \tag{46}$$

At this point in the analysis, I know everything needed except for the three coefficients β_i, for $i = 1, 2, 3$. But, by taking appropriate sums of (46), I find that

$$\beta_i = s^d_{i1} + s^d_{i2} + s^d_{i3} - \frac{1}{3K^g_R} = s^u_{i1} + s^u_{i2} + s^u_{i3} - \frac{1}{3K^g_R} + B\beta_i. \tag{47}$$

Rearranging, I find that

$$\beta_i(1 - B) = s^u_{i1} + s^u_{i2} + s^u_{i3} - \frac{1}{3K^g_R}. \tag{48}$$

Formula (45) for the Skempton (1954) coefficient determines B exactly in terms of presumed known quantities. In the present case, the Skempton coefficient B, however, was not assumed to be known, since for homogeneous grains I can compute K^d_R relatively easily, and then B follows since I also know K^g_R. [For the case of heterogeneous or anisotropic grains, the necessary introduction of the additional variable K^ϕ_R requires still more measured data, and it turns out that the next easiest quantity to measure is B itself — as was already shown in the isotropic case.] So, all three β_i's (which are themselves drained constants) and γ are now precisely determined. All the remaining drained compliances s^d_{ij} can then be found directly from (46). Note that all the steps in this inversion procedure are linear; there was no need to solve any quadratic equation in this formulation of the undrained-to-drained inversion problem. There is also no iteration, and no fitting steps are required in this procedure.

Deducing anisotropic drained constants from undrained: Homogeneous grains and pores

I am now in position to develop the analogy between the isotropic and anisotropic Gassmann (1951) equations for the case of homogeneous grains. In particular, the equation for the suspension modulus in (2) does not change at all. In contrast, the equation for the effective undrained bulk modulus K^u, as shown in both (1) and (3), changes only in that the relationship is now between the Reuss averages K^u_R and K^d_R of these quantities. This result is completely analogous to (3), and so will not be shown here.

Since the remainder of the argument is virtually identical to the isotropic case, I therefore obtain:

$$K^d_R = \left(\frac{K^u_R}{K_{susp}} - 1\right)\left[1/K_{susp} - 2/K^g_R + K^u_R/(K^g_R)^2\right]^{-1}. \tag{49}$$

This formula shows how to invert for drained Reuss bulk modulus K^d_R from knowledge of K^u_R, ϕ, K_f and K^g_R in an anisotropic (up to orthotropic) poroelastic system.

TABLE 3. Data for the principal stiffness coefficients c_{ij} of orthorhombic sulfur (S), Rochelle salt, Benzophenone, and α-Uranium (α-U). All data from Musgrave (2003), but re-expressed in units of GPa.

Stiffness	Sulfur (S)	Rochelle Salt	Benzophenone	α-Uranium
c_{11}	24.0	25.5	107.0	215.0
c_{22}	20.5	38.1	100.0	199.0
c_{33}	48.3	37.1	71.0	267.0
c_{12}	13.3	14.1	55.0	46.0
c_{13}	17.1	11.6	16.9	22.0
c_{23}	15.9	14.6	32.1	107.0

TABLE 4. Data for various measures of bulk modulus K (Voigt, Reuss, and three partial sum moduli) for orthorhombic sulfur (S), Rochelle salt, Benzophenone, and α-Uranium (α-U). All data from Musgrave (2003) [see TABLE 3 here], while the expressions in the main text were used for the computations. All moduli in units of GPa.

Bulk Modulus	Sulfur (S)	Rochelle Salt	Benzophenone	α-Uranium
K_V	20.6	20.1	54.0	114.6
K_R	17.6	19.3	49.2	111.3
K_1	15.2	12.5	55.8	87.9
K_2	10.1	30.6	107.5	113.6
K_3	15.8	23.3	29.6	147.7

Clearly this formula does not yet provide the individual compliance matrix elements s^d_{ij} directly. Nevertheless, Equation (49) was the hardest step in the overall procedure. The rest of the steps follow easily once I have this rigorous result available to use.

To finish the analysis, I make use of the newly computed value of K^d_R, and substitute this number into the formula for B, which in this case is:

$$B = \frac{1 - K^d_R/K^u_R}{1 - K^d_R/K^g_R}. \tag{50}$$

Once I know Skempton coefficient B, this value can be substituted into (48) in order to determine the β_i coefficients for $i = 1, 2, 3$. The remaining coefficient is $\gamma = \alpha_R/BK^d_R$. So I have shown that the critical step in this process was determining the value of the drained Reuss bulk modulus K^d_R.

Deducing anisotropic drained constants from undrained: Heterogeneous grains and pores

One difficulty for heterogeneous grains comes from the additional constant K^ϕ_R that I do not know how to determine independently from the other poroelastic measurements. But this fundamental problem is actually no different for the anisotropic case than it was for the isotropic one, and the solution is also the same. In both cases, I need more information, and in both cases the necessary information will most likely come from our knowledge of the Skempton (1954) coefficient B. If I assume that B can be directly measured (which is plausible, since $B = p_f/p_c$ in the undrained case when a uniform confining pressure is applied to the system), then the problem is completely solved, because B is the key to solving for the coefficients β_i in (48). The only new difficulty is that the terms of the form $1/3K_R^g$ must also be replaced by the partial grain compliance sums $\frac{1}{3\overline{K}_i^g}$, as shown in (30). So I now have

$$\beta_i = s_{i1}^d + s_{i2}^d + s_{i3}^d - \frac{1}{3\overline{K}_i^g} = s_{i1}^u + s_{i2}^u + s_{i3}^u - \frac{1}{3\overline{K}_i^g} + B\beta_i. \tag{51}$$

Rearranging, I find that, for heterogeneous grains, the result is:

$$\beta_i(1-B) = s_{i1}^u + s_{i2}^u + s_{i3}^u - \frac{1}{3\overline{K}_i^g}. \tag{52}$$

So, I am almost done now, but I still need either to determine the values of the anisotropic grain correction terms $\frac{1}{3\overline{K}_i^g}$, or to find some way of avoiding the necessity of doing so.

In principle, this can be done experimentally by actually performing a test on the porous sample that applies the same pressure inside and outside. Then, measurements of the change in strain in the three orthogonal directions $i = 1, 2, 3$ would provide direct measures of the quantities $\overline{K}_i^g$ needed. So this approach is one that is experimentally feasible.

An alternative that I have not considered so far would be to perform shear tests by applying nonzero deviatoric stress changes (Skempton, 1954; Lockner and Stanchits, 2002). The undrained fluid pressure is given by $p_f = Bp_c = B(-\sigma_m)$, where the mean stress is $\sigma_m = (\sigma_{11} + \sigma_{22} + \sigma_{33})/3$. But, if the σ_{ii}'s are not uniform, then there are also deviatoric stresses present, due to the nonuniformity of the principal stresses.

Triaxial testing geometry

One common example of this type of measurement uses triaxial testing [see Lockner and Stanchits (2002)], where a two-sided confining stress is defined as $\sigma_{22} = \sigma_{33}$, and then the deviatoric stress is determined by

$$\sigma_{dev} \equiv (\sigma_{11} - \sigma_{33})/2. \tag{53}$$

In this situation, the general equation relating undrained pressure to the confining stresses is given by:

$$-p_f = B\sigma_m + 2\left(A - \frac{1}{3}\right)B\sigma_{dev}, \tag{54}$$

where the only new symbol is the first coefficient A of Skempton (1954).

TABLE 5. Data for the principal stiffness coefficients c_{ij} for $i, j = 1, 2, 3$, as well as c_{44}, of hexagonal minerals: cadmium (Cd), H_2O ice, β-quartz (SiO_2), titanium (Ti), and zirconium (Zr). All data from Simmons and Wang (1971) [entry numbers: 52473, 52563, 52643, 52726, and 52798, respectively], but re-expressed in units of GPa.

Stiffness	Cadmium (Cd)	H_2O Ice	β-Quartz	Titanium (Ti)	Zirconium (Zr)
c_{11}	115.30	13.85	116.6	163.9	137.0
c_{33}	51.20	14.99	110.4	181.6	160.7
c_{12}	39.24	7.07	16.7	91.3	75.6
c_{13}	40.22	5.81	32.8	68.9	65.4
c_{44}	20.40	3.19	36.1	47.2	30.1

TABLE 6. Data for various measures of bulk modulus K (Voigt, Reuss, and three partial-sum moduli) for hexagonal minerals: cadmium (Cd), H_2O ice, β-quartz (SiO_2), titanium (Ti), and zirconium (Zr). All data from Simmons and Wang (1971) [see TABLE 5], while the expressions in the main text were used for all the computations. All moduli in units of GPa.

Modulus	Cadmium (Cd)	H_2O Ice	β-Quartz	Titanium (Ti)	Zirconium (Zr)
K_V	57.89	8.90	56.47	107.51	94.17
K_R	48.61	8.90	56.37	107.50	94.02
$K_1 = K_2$	143.07	8.94	53.97	109.00	89.58
K_3	20.95	8.82	61.86	104.63	104.36

It is not difficult to show that, in terms of our previous definitions for the triaxial testing geometry, the coefficient A is given precisely by the ratio

$$A \equiv \frac{\beta_1}{\beta_1 + \beta_2 + \beta_3}. \tag{55}$$

For an isotropic system, $A = 1/3$, so this contribution always vanishes in (54). This fact explains why I did not encounter this coefficient before in the analysis. Note that there is no assumption here that the poroelastic system itself is necessarily transversely isotropic. Only the prescribed equality of the two applied transverse stresses, σ_{22} and σ_{33}, is assumed. Then, the formula (54) follows directly from the equations already presented.

Non-triaxial testing geometries

Clearly, it would also be natural to introduce other measures of the β_i coefficients as well, especially if the measurements are not being constrained to the triaxial testing configuration. So I might imagine that three such coefficients could be measured according to:

$$-p_f = B\left[\sigma_m + \sum_i A_i(\sigma_{ii} - \sigma_m)\right], \tag{56}$$

where

$$A_i = \frac{\beta_i}{\beta_1 + \beta_2 + \beta_3}, \tag{57}$$

for $i = 1, 2, 3$. In general, no more than two of these A_i coefficients can be independent since $\sum_{i=1,2,3} A_i \equiv 1$. But, for general testing configurations, there could be two useful and distinct measurements to be gathered from deviatoric response testing, although only one was available in the triaxial testing configuration.

In order to be able to deduce the values of the β_i's from the A_i's, I need to know the value of the sum $\beta_1 + \beta_2 + \beta_3 = \gamma B$. I also needed to know the value of B to determine any of the A_i's, but the value of γ is harder to determine independently. The values of K_R^u and the total K_R^g are both usually easier to determine, so it is likely enough information is available to compute the β_i sum this way:

$$\beta_1 + \beta_2 + \beta_3 = \frac{\frac{1}{K_R^u} - \frac{1}{K_R^g}}{1 - B} = \gamma B. \tag{58}$$

If the β_i sum has been computed using (58), then clearly I also have

$$\beta_i = A_i \left(\frac{\frac{1}{K_R^u} - \frac{1}{K_R^g}}{1 - B} \right). \tag{59}$$

Once I have computed the β_i's, I also find (if I want to, although it is not usually critical information) the values of the partial sums of the grain Reuss modulus:

$$\frac{1}{3K_i^g} \equiv s_{i1}^u + s_{i2}^u + s_{i3}^u - \beta_i(1 - B). \tag{60}$$

This additional information is therefore available if needed for some other reason such as determining how well-stirred the particles composing a given granular medium might be.

Deducing anisotropic drained constants from undrained for very heterogeneous porous media

At this point I have determined a data processing scheme that would provide all the drained constants for a poroelasticity system from measurements of the undrained constants. In the example of the preceding subsection, I needed to broaden the meaning of the undrained set of constants to include the Skempton A_i coefficients, which were not needed in earlier parts of the paper. But they could nevertheless be computed from the information found earlier, since I did show how to compute all the β_i's directly, and these coefficients provide just the information that I would need for determining these values from (57).

In realistic data collection situations, especially those involving field data, my previous assumptions concerning the nature and orientations of the constituent grains of the granular porous medium may sometimes – perhaps most times – be too idealized. Nevertheless, it is the case that the equations of poroelasticity never become any more complex than those shown here. What does change is the interpretation of the directional grain moduli. In the worst case scenario, equation (60) needs to be replaced by an equation of the same form, namely:

$$\frac{1}{3K_i^*} = s_{i1}^u + s_{i2}^u + s_{i3}^u - \beta_i(1 - B). \tag{61}$$

Measurements are exactly as before, but the interpretation of the resulting constant estimator K_i^* becomes that of an effective medium bulk modulus, *i.e.*, one that is (or at least could be) dependent on the directions $i = 1, 2, 3$ of measurement. Effective medium theories for random polycrystals generally assume [see Berryman (2005)] that the anisotropic grains are perfectly randomly oriented. Of course, this may not be true in practice. But to do a better job of predicting the outcome of experiments in situations where grain orientations are not perfectly random, I need information about these deviations from perfect randomness. In the present context, the information would come in the form of these measured constants K_i^*. Some effort could then be expended in showing how such moduli might arise if the constituents' nature and volume fractions are known. But in the absence of such knowledge, these constants are sufficient to analyze other results of most experiments of interest in poroelasicity.

To provide different ideas about how important the anisotropy, and the random orientation of the constituents might be in a few cases, TABLES 1 and 2 show some quantitative examples based on results of Berryman (2005) and Ranganathan and Ostoja-Starzewski (2008a,b). TABLES 3 and 4, respectively, provide input data for the types of orthorhombic solids (Musgrave, 2003), and the results for the Voigt, Reuss, and directional measures of bulk moduli for these particular materials. Note the significant finding that the directional moduli do not have to stay within the values set by the Voigt and Reuss estimators.

Tables 5–9 show similar results for selected data taken from the compendium assembled by Simmons and Wang (1971).

TABLE 7. Data for the principal stiffness coefficients c_{ij} for $i, j = 1, 2, 3$ and c_{44} of cubic symmetry minerals: aluminum (Al), copper (Cu), magnesia (MgO), and spinel ($MgAl_2O_4$). All data from Simmons and Wang (1971) [entry numbers: 10089, 10385, 10902, and 11877, respectively], but re-expressed in units of GPa.

Stiffness	Aluminum (Al)	Copper (Cu)	Magnesia (MgO)	Spinel ($MgAl_2O_4$)
c_{11}	107.30	170.98	297.08	298.57
c_{12}	60.80	123.99	95.36	153.72
c_{44}	28.30	75.45	156.13	157.58

TABLE 8. Data for various measures of bulk modulus K (Voigt, Reuss, and three partial-sum moduli) for cubic symmetry minerals: aluminum (Al), copper (Cu), magnesia (MgO), and spinel ($MgAl_2O_4$). All data from Simmons and Wang (1971) [see TABLE 7], while the expressions in the main text were used for all the computations. All moduli in units of GPa. Clearly, all the pertinent bulk moduli for each material are the same (*i.e.*, $K_V = K_R = K_1 = K_2 = K_3$, even though these cubic symmetry minerals are not isotropic.

Bulk Modulus	Al	Cu	MgO	$MgAl_2O_4$
$K_V = K_R = \dots$	76.3	139.65	162.6	202.00

SUMMARY AND CONCLUSIONS

There have been a great many experiments done on poroelastic systems through the years, and many attempts to measure complete poroelastic data sets. The work summarized here is designed to make this process easier by removing the need (whenever possible) for tedious fitting routines that have traditionally been used to find the pertinent drained constants for the measured fluid-saturated and undrained systems. It is of some real scientific importance to have methods like those treated here, because it is a well-known fact that the presence of the pore-fluid can alter the nature of the points of contact between neighboring grains, and therefore alter the values of the "drained" constants that were sought here and found via the methods developed for this purpose. Different values of the "drained" constants might be obtained if all the fluid is physically drained out of the system, so it is effectively "dry" rather than merely "drained" (*i.e.*, in the sense of pore-fluid having the capability of moving in and out of the boundaries as would happen in the absence of jacketing material). In the case of a fully dry system, the grain-to-grain contacts may act very differently than they do when saturated with certain fluids. At the very least, it could be important to check experimentally whether these constants are different or not in a variety of systems, and the present analysis will permit such studies to move forward.

One especially interesting aspect of the analysis presented is that in no case did the solution of any of the problems treated involve any analysis more complicated than solving a linear equation. There are no quadratic equations solved in this paper, and none that needed to be solved. The hardest calculation in the paper is the implicit inversion of a 3×3 matrix when the real data are poroelastic stiffnesses, rather than compliances. [Note: Matrix inversion requires only the calculation of various determinants, but not the solution of any quadratic or cubic equations. These additional difficulties can be avoided unless I also want or need to find the eigenvalues of the matrix. But this step is unnecessary in the work described here.] This situation does occur in practice whenever the elastic/poroelastic data are obtained using wave propagation methods. Then, the actual data have the form $v = \sqrt{c/\rho}$, where v is a wave speed, c is a stiffness or combination of stiffnesses, and ρ is the inertial density. A complete set of the stiffnesses for the principal stresses and strains is needed for the analysis because I require compliance data, and to obtain a complete set of compliance data from stiffness data, I also need a complete set of the corresponding stiffness data. All the elements of the undrained 3×3 compliance matrix for the principal stresses and strains must be known in order to proceed with the described analysis.

TABLE 9. Reuss (R) and Voigt (V) shear moduli of various common hexagonal and cubic materials (Simmons and Wang, 1971): cadmium, β-quartz, titanium, zirconium, aluminum, copper, magnesia, and spinel. Units of shear modulus G are GPa. All the formulas needed to compute the various effective moduli from the stiffness coefficients are given in Berryman (2005). The anisotropy parameter $\mathcal{A} \equiv 5\frac{G_V}{G_R} + \frac{K_V}{K_R} - 6$ [from Ranganathan and Ostoja-Starzewski (2008a,b)].

Material	Symmetry	G_R	G_V	G_V/G_R	$\mathcal{A}$
Cd	Hexagonal	22.1	26.4	1.197	1.174
SiO_2	Hexagonal	3.48	3.55	1.025	0.125
Ti	Hexagonal	34.1	44.8	1.31	0.154
Zr	Hexagonal	32.54	33.40	1.03	0.132
Al	Cubic	26.04	26.28	1.009	0.045
Cu	Cubic	40.04	54.67	1.365	1.825
MgO	Cubic	128.06	134.02	1.047	0.235
$MgAl_2O_4$	Cubic	107.17	123.52	1.152	0.76

The analysis has been restricted to systems having orthotropic poroelastic symmetry or higher. Lower symmetry systems might also be studied, but I purposely avoided them here because for such systems it is harder to know for sure from experimental data when you have determined the true axes of symmetry. Also, in such cases of orthotropic symmetry or higher, the system of equations to be studied is substantially reduced because there is no coupling of the fluid effects to the shear components associated with the strains e_{23}, e_{31}, e_{11}, or the the stresses σ_{23}, σ_{31}, σ_{12}. Shear effects are not ignored altogether however, as there are well-known shearing mechanisms in poroelastic media associated with the Skempton (1954) coefficient A [also see Lockner and Stanchits (2002)]. These effects were studied here in some detail, and were found to be very useful in accomplishing our main goals, since they provided a necessary mechanism for measuring some otherwise difficult to measure off-diagonal terms in the poroelastic equations.

I conclude that this analysis has been successful in solving the problem of obtaining drained constants from undrained constants in all the cases considered. The chosen set of cases (orthotropic or higher symmetry) is not very restrictive from a practical point of view, as the great majority of poroelastic systems studied in practice usually have hexagonal (transversely isotropic) symmetry or higher, and therefore are all explicitly included within the range of the present analyses.

ACKNOWLEDGMENTS

I thank Seiji Nakagawa and Leon Thomsen for helpful comments on the work. I thank Adam Halpert and Mandy Man Chu Wong for useful suggestions that improved the manuscript.

REFERENCES

Barenblatt, G. I. and Y. P. Zheltov, 1960, Fundamental equations of filtration of homogeneous liquids in fissured rocks: Sov. Phys. Dokl., **132**, 545–548.

Berryman, J. G., 1997, Transversely isotropic elasticity and poroelasticity arising from thin isotropic layers, *in* Teng, Y.-C., E.-C. Shang, Y.-H. Pao, M. H. Schultz, and A. D. Pierce, eds., Theoretical and Computational Acoustics 1997, 457–474, World Scientific, Newark, NJ.

———, 2005, Seismic waves in rocks with fluids and fractures: Journal of the Mechanics and Physics of Solids, **53**, 2141–2173.

Berryman, J. G. and G. W. Milton, 1991, Exact results for generalized gassmann's equations in composite porous media with two constituents: Geophysics, **56**, 1950–1960.

Berryman, J. G. and S. R. Pride, 2002, Models for computing geomechanical constants of double-porosity materials from the constituents' proeprties: Journal of Geophysical Research, **107**, 2052–1–2052–15.

Biot, M. A. and D. G. Willis, 1957, The elastic coefficients of the theory of consolidation: ASME Journal of Applied Mechanics, **24**, 594–601.

Brown, R. J. S. and J. Korringa, 1975, On the dependence of the elastic properties of a porous rock on the compressiblity of the pore fluid: Geophysics, **40**, 608–616.

Cheng, A. H.-D., 1997, Material coefficients of anisotropic poroelasticity: International Journal of Rock Mechanics, **34**, 199–205.

Gassmann, F., 1951, On elasticity of porous media, *in* Pelissier, M. A., H. Hoeber, N. van de Coevering, and I. F. Jones, eds., Classics of Elastic Wave Theory, volume **24**, 389–407, Society of Exploration Geophysicists, Tulsa, Oklahoma.

Gurevich, B., D. Makarynska, and M. Pervukhina, 2009, Ultrasonic moduli for fluid-saturated rocks: Mavko-Jizba relations rederived and generalized: Geophysics, **74**, N25–N30.

Hill, R., 1952, The elastic behaviour of crystalline aggregate: Proceedings of the Physical Society of London, **A65**, 349–354.

Lockner, D. A. and S. A. Stanchits, 2002, Relations between drained and undrained moduli in anisotropic poroelasticity: Journal of Geophysical Research, **107**, 2353–1–2353–14.

Musgrave, M. J. P., 2003, Crystal Acoustics: Introduction to the Study of Elastic Waves and Vibrations in Crystals: Acoustical Society of America, New York.

Ranganathan, S. I. and M. Ostoja-Starzewski, 2008a, Universal elastic anisotropy index: Physical Review Letters, **101**, 055502.

———, 2008b, Universal elastic anisotropy index: Physical Review B, **77**, 214308.

Reuss, A., 1929, Berechung der Fliessgrenze von Mischkristallen: Z. Angew. Math. Mech., **9**, 55.

Rice, J. R. and M. P. Cleary, 1976, Some basic stress diffusion solutions for fluid-saturated elastic porous media with compressible constituents: Reviews of Geophysics and Space Physics, **14**, 227–241.

Simmons, G. and H. Wang, 1971, Single Crystal Elastic Constants and Calculated Aggregate Properties: A Handbook: The M.I.T. Press, Cambridge, Massachusetts.

Skempton, A. W., 1954, The pore-pressure coefficients *a* and *b*: Géotechnique, **4**, 143–147.

Stoll, R. D., 1974, Acoustic waves in saturated sediments, *in* Physics of Sound in Marine Sediments, 19–39, Plenum, New York.

Thigpen, L. and J. G. Berryman, 1985, Mechanics of porous elastic materials containing multiphase fluid: International Journal of Engineering Science, **23**, 1203–1214.

Voigt, W., 1928, Lehrbuch der Kristallphysik: Teubner, Leipzig.

Wang, H. F., 2000, Theory of Linear Poroelasticity with Applications to Geomechanics and Hydrogeology: Princeton University Press, Princeton, NJ.

Wood, A. W., 1955, A Textbook of Sound: Bell, London, England.

Zhu, X. and G. A. McMechan, 1990, Direct estimation of the bulk modulus of the frame in a fluid-saturated elastic medium by Biot theory: 60th International Meeting, Expanded Abstracts, 787–790, Society of Exploration Geophysicists. Tulsa, Oklahoma.

SEP PHONE DIRECTORY

Name	**Phone**	**Login Name**
Almomin, Ali	723-3187	ali
Ayeni, Gboyega	723-6006	gayeni
Barak, Ohad	723-3187	ohad
Berryman, James	–	berryman
Biondi, Biondo	723-1319	biondo
Cardoso, Claudio	723-1250	claudio
Claerbout, Jon	723-3717	jon
Clapp, Bob	725-1334	bob
de Ridder, Sjoerd	723-1250	sjoerd
Guitton, Antoine	–	antoine
Halpert, Adam	723-6006	adam
Lau, Diane	723-1703	diane
Leader, Chris	723-0463	chrisl
Li, Elita	723-9282	myfusan
Maysami, Mohammad	723-9282	mohammad
Moussa, Nader	723-0463	nwmoussa
Shen, Xukai	723-0463	xukai
Tang, Yaxun	723-1250	tang
Taweesintananon, Kittinat	723-3187	kittinat
Wong, Mandy	723-9282	mandyman
Zhang, Yang	723-3187	yang

SEP fax number: (650) 723-0683

E-MAIL

Our Internet address is "*sep.stanford.edu*"; i.e., send Jon electronic mail with the address "*jon@sep.stanford.edu*".

WORLD-WIDE WEB SERVER INFORMATION

Sponsors who have provided us with their domain names are not prompted for a password when they access from work. If you are a sponsor, and would like to access our restricted area away from work, visit our website and attempt to download the material. You will then fill out a form, and we will send the username/password to your e-mail address at a sponsor company.

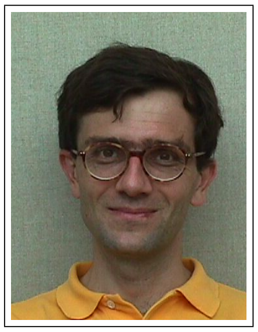
STEERING COMMITTEE MEMBERS, 2008-2009

Name	Company	Tel #	E-Mail
Raymond Abma	BP	(281)366-4604	abmar1@bp.com
Biondo Biondi	SEP	(650)723-1319	biondo@sep.stanford.edu
Robert Bloor (Co-chair, 2nd year)	ION/GX Technology	(281)781-1141	robert.bloor@iongeo.com
Jon Claerbout	SEP	(650)723-3717	jon@sep.stanford.edu
Richard Cook	Shell	(713)245-7195	richard.cook@shell.com
Stewart A. Levin	Halliburton	(303)488-3062	stewart.levin@halliburton.com
Yi Luo (Co-chair, 1st year)	Saudi Aramco	(966)38738520	yi.luo@aramco.com

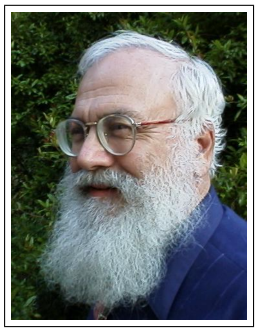
Research Personnel

James G. Berryman received a B.S. degree in physics from Kansas University (Lawrence) in 1969 and a Ph.D. degree in physics from the University of Wisconsin (Madison) in 1975. He subsequently worked on seismic prospecting at Conoco. His later research concentrated on seismic waves in rocks and sediments – at AT&T Bell Laboratories (1978-81) and at Lawrence Livermore National Laboratory (1981-), where he is currently a physicist in the Energy and Environment Directorate. He received the Maurice Anthony Biot Medal of the ASCE in May, 2005, for his work in the mechanics and acoustics of porous media containing fluids. Continuing research interests include acoustic, seismic, and electrical methods of geophysical imaging and studies of waves in porous media. He is a member of APS, AGU, ASA, and SEG.

Biondo L. Biondi graduated from Politecnico di Milano in 1984 and received an M.S. (1988) and a Ph.D. (1990) in geophysics from Stanford. SEG Outstanding Paper award 1994. During 1987, he worked as a Research Geophysicist for TOTAL, Compagnie Francaise des Petroles in Paris. After his Ph.D. at Stanford, Biondo worked for three years with Thinking Machines Co. on the applications of massively parallel computers to seismic processing. After leaving Thinking Machines, Biondo started 3DGeo Development, a software and service company devoted to high-end seismic imaging. Biondo is now Associate Professor (Research) of Geophysics and leads SEP efforts in 3-D imaging. He is a member of SEG and EAGE.



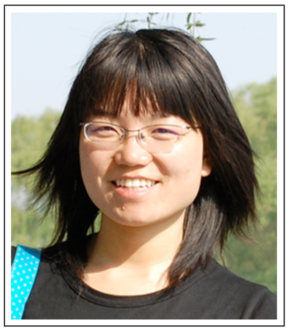
Jon F. Claerbout (M.I.T., B.S. physics, 1960; M.S. 1963; Ph.D. geophysics, 1967), professor at Stanford University, 1967. Emeritus 2008. Best Presentation Award from the Society of Exploration Geophysicists (SEG) for his paper, *Extrapolation of Wave Fields.* Honorary member and SEG Fessenden Award "in recognition of his outstanding and original pioneering work in seismic wave analysis." Founded the Stanford Exploration Project (SEP) in 1973. Elected Fellow of the American Geophysical Union. Authored three published books and five internet books. Elected to the National Academy of Engineering. Maurice Ewing Medal, SEG's highest award. Honorary Member of the European Assn. of Geoscientists & Engineers (EAGE). EAGE's highest recognition, the Erasmus Award.



Robert Clapp received his B.Sc. (Hons.) in Geophysical Engineering from Colorado School of Mines in May 1993. He joined SEP in September 1993, received his Masters in June 1995, and his Ph.D. in December 2000. He is a member of the SEG and AGU.

Claudio Guerra received his B.Sc. in Geology from Federal University of Rio de Janeiro, Brazil in 1988 and a M.Sc. from State University of Campinas, Brazil in 1999. Since 1989, he has been working for Petrobras, Brazil. He joined SEP in 2006 and is currently pursuing a Ph.D. in geophysics at Stanford University. He is member of SEG and SBGf.

Antoine Guitton received a M.Sc. in geophysics from Universite de Strasbourg, France in 1996 and from Stanford University in 2000. He received his Ph.D. from Stanford University in 2005. He was awarded Best Student Paper from the SEG in 1999 and received the EAGE Van Weelden award in 2004. He was research assistant at the Institut Francais du Petrole (Paris-1996/97) working on well seismic imaging and research geophysicist at CGG Houston (1997-98) working on multiples attenuation. He is now Senior Geophysicist for 3DGeo Inc. in Santa Clara, CA. His research interests include interpolation, noise attenuation, inversion and novel seismic-interpretation techniques. He is a member of the SEG and EAGE.

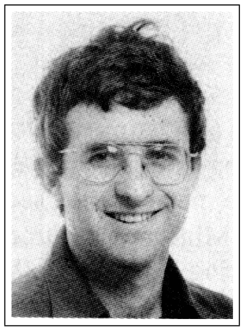


Yunyue (Elita) Li graduated from China University of Petroleum, Beijing in July 2008 with a B.S. in Information and Computational Science. She joined SEP in the fall of 2008, and is currently working toward a Ph.D. in Geophysics. She is a student member of the SEG.

Mohammad Maysami graduated in 2005 with two B.Sc. in Electrical Engineering and Petroleum Engineering from Sharif University of Technology, Tehran, Iran. After completing a M.A.Sc. degree in January 2008 in Geophysics at University of British Columbia in Vancouver, Canada, he joined SEP. He is working toward a Ph.D. in Earth, Energy, and Environmental Sciences (EEES) which is jointly supervised by SEP and SCRF (Stanford Center for Reservoir Forecasting). He is a student member of SEG, AGU, and SPE.

Nader Moussa graduated from North Carolina State University, where he studied Physics and Electrical and Computer Engineering. In 2008, he finished his M.Sc. at Stanford in Electrical Engineering, with focus on experimental radioscience and electromagnetic sensor instrumentation design. He is now working towards a Ph.D. in Geophysics with the Stanford Exploration Project.

Shuki Ronen is a geophysicist with Geco-Prakla working on various aspects of seismic data processing and acquisition. Previously he worked with GeoQuest on seismic data interpretation; with Schlumberger on reservoir characterization; with the Institute of Petroleum Research and Geophysics on seismic data processing and acquisition; with the Colorado School of Mines as a visiting professor; and with Saxpy Computer company as an engineer. He has a Ph.D. from Stanford in Geophysics, and a B.Sc. in Physics and Geology from Hebrew University. (Photo: December 1985, Geophysics, p. 2919)



Mandy Wong graduated in 2004 with a B.Sc. in Physics and Mathematics from the University of British Columbia (UBC) in Vancouver, Canada. In 2006, she obtained a M.Sc. degree in Condensed Matter Thoery at UBC. Afterward, Mandy worked for a geophysical consulting company, SJ Geophysics, based in Vancouver, Canada. Mandy joined SEP in 2008, and is working towards a Ph.D. in Geophysics. Her main research interest is imaging with multiples.

SEP ARTICLES PUBLISHED OR IN PRESS

Ayeni, G., A. Huck, and P. de Groot, 2008, Extending reservoir property prediction with psedo-wells: First Break. **26**, 11–15.

Biondi, B., 2007a, Angle-domain common image gathers from anisotropic migration: Geophysics, **72**, S81–S91.

———, 2007b, Residual moveout in anisotropic angle-domain common-image gathers: Geophysics, **72**, S93–S103.

Biondi, B. L., 2006, 3D Seismic Imaging: Society of Exploration Geophysicists.

———, 2007c, Concepts and Applications in 3D Seismic Imaging: Society of Exploration Geophysicists, in press.

Shan, G. and B. Biondi, 2008, Plane-wave migration in tilted coordinates: Geophysics, **73**, S185-S194.

de Ridder, S., Prieto, G.A., 2008, Seismic Interferometry and the Spatial Auto-Correlation Method on the Regional Coda of the Non-Proliferation Experiment: AGU Fall Meeting, S31A-1885.

de Ridder, S., G. Papanicolaou and B. Biondi, 2009, Kinematics of iterative interferometry in a passive seismic experiment: SEG Technical Program Expanded Abstracts, **28**, 1622-1626.

Gray, S., D. Trad, B. Biondi, and L. Lines, 2006, Towards wave-equation imaging and velocity estimation: Recoder, **31**, 47–53.

Guitton, A., B. Kaelin, and B. Biondi, 2007, Least-squares attenuation of reverse-time-migration artifacts: Geophysics, **72**, G19–G23.

Guerra, C., Y. Tang, and B. Biondi, 2009, Wave-equation tomography using image-space phase encoded data: SEG Technical Program Expanded Abstracts, **28**, 3964-3968.

Gunther, R. and M. Reshef, 2007, Dip corrections for velocity analysis in super-gathers: J. Seis. Expl., **16**.

Haines, S., A. Guitton, and B. Biondi, 2007a, Seismoelectric data processing for surface surveys of shallow targets: Geophysics, **72**, G1–G8.

Haines, S., S. Pride, S. Klemperer, and B. Biondi, 2007b, Seismoelectric imaging of shallow targets: Geophysics, **72**, G9–G20.

Halpert, A., Clapp, R.G., Lomask, J. and B. Biondi, 2008, Image segmentation for velocity model construction and updating: SEG Technical Program Expanded Abstracts **27**, 3088-3092.

Halpert, A., Clapp, R.G. and B. Biondi, 2008, Seismic image segmentation with multiple attributes: SEG Technical Program Expanded Abstracts **28**, 3700-3704.

Lomask, J., R. G. Clapp, and B. Biondi, 2007, Application of image segmentation to tracking 3D salt boundaries: Geophysics, **72**, P47–P56.

Maysami, M., and F. J. Herrmann, 2008, Lithology constraints from seismic waveforms: application to opal-A to opal-CT transition: SEG Technical Program Expanded Abstracts **27**, 2011-2015.

Rosales, D. and B. Biondi, 2006, Converted-wave azimuth moveout: Geophysics, **71**, S99–S110.

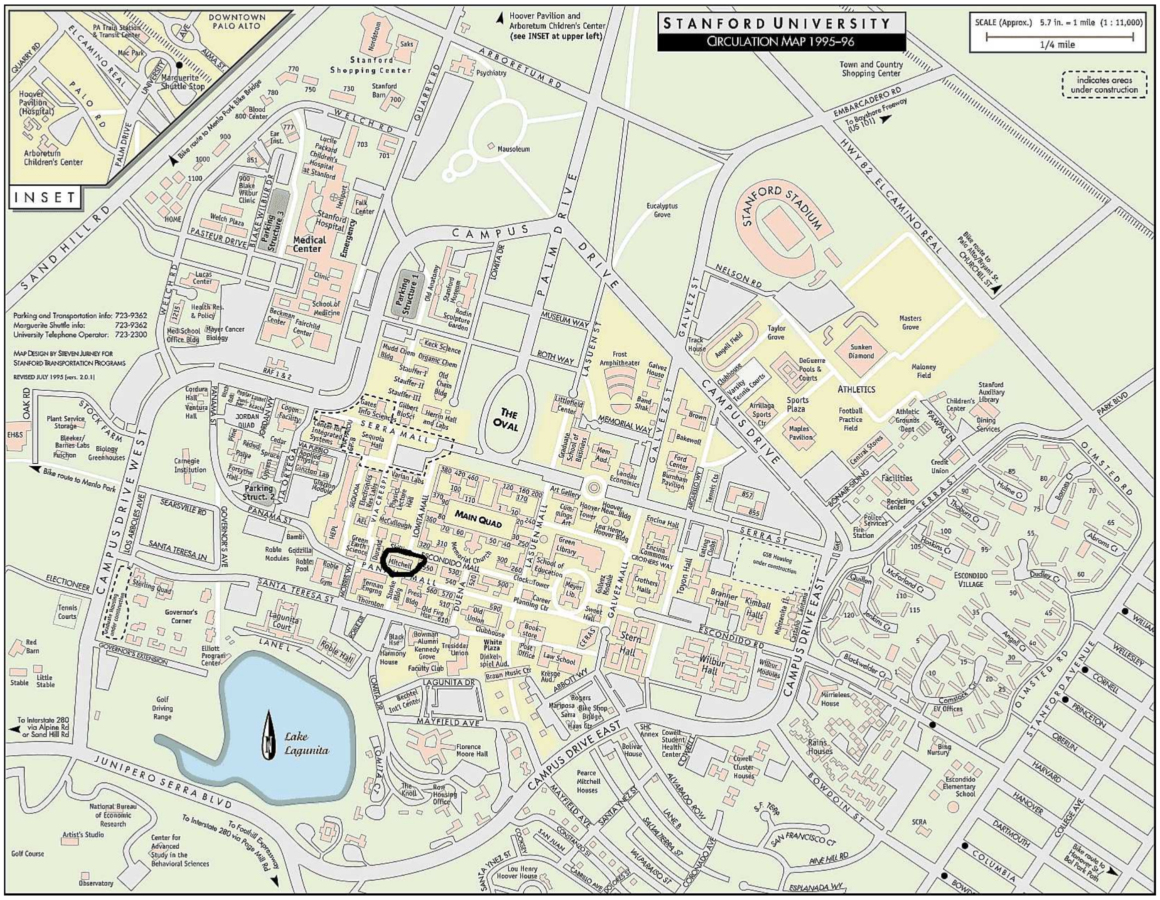
Rosales, D. A., S. Fomel, B. Biondi, and P. Sava, 2008, Angle-domain common-image gathers for converted waves: Geophysics, **73**, S17-S26.

———, 2008, Riemannian wavefield extrapolation: Nonorthogonal coordinate systems: Geophysics, **73**, T11–T21.

———, 2008, Wave-equation Hessian by phase encoding: SEG Technical Program Expanded

Abstracts **27**, 2201-2205.

Valenciano, A., B. Biondi, and A. Guitton, 2006, Target-oriented wave-equation inversion: Geophysics, **71**, A35.


STANFORD UNIVERSITY
CIRCULATION MAP 1995–96
SCALE (Approx.) 5.7 in. = 1 mile (1 : 11,000)
1/4 mile
indicates areas under construction
INSET
DOWNTOWN PALO ALTO
Hoover Pavilion and Arboretum Children's Center (see INSET at upper left)
Stanford Shopping Center
Medical Center
Stanford Hospital
CAMPUS DRIVE
PALM DRIVE
STANFORD STADIUM
THE OVAL
MAIN QUAD
SERRA MALL
Mitchell
ATHLETICS
ESCONDIDO VILLAGE
Lake Lagunita
JUNIPERO SERRA BLVD
SAND HILL RD
CAMPUS DRIVE WEST
CAMPUS DRIVE EAST
HWY 82 EL CAMINO REAL
Parking and Transportation info: 723-9362
Marguerite Shuttle info: 723-9362
University Telephone Operator: 723-2300
MAP DESIGN BY STEVEN JURNEY FOR STANFORD TRANSPORTATION PROGRAMS
REVISED JULY 1995 (vers. 2.0.1)