Stanford Exploration Project, Report 129, May 6, 2007, pages 117–129

ℓ_1 Regularized Inversion

Ben Witten

ABSTRACT

Here an interior point inversion method is presented that solve a least squares problem with ℓ_1 regularization. Velocity inversion can benefit from ℓ_1 regularization because the sparse solution creates blocky velocity models. This is often more geologically accurate than smooth models. In this paper an efficient method is present for solving ℓ_1 regularized least squares problems. Its usefulness is shown through comparisons of previous methods on an example using a least squares formulation for Dix inversion.

INTRODUCTION

Interval velocity estimation is an fundamental problem in reflection seismology. An accurate velocity model is essential to creating an interpretable image from seismic data. There are many techniques for estimating velocity (Clapp, 2001; Sava, 2004) in complex geological settings, but these are often very expensive due to, not only, the operator but also the non-linear nature of the problem and coherent noise that can lead the linear problem to local minima. In this paper an inversion technique is presented for ℓ_1 regularized problems that could potentially decrease the computation time for velocity estimation.

Grid based techniques have an additional drawback, in that they tend to create smooth models even where sharp contrast exists. When considering velocity inversion problems, ℓ_1 regularization can be used to create a sparse solution, resulting in more "blocky" velocity models. The ℓ_1 regularization preserves sharp geologic boundaries, such as channel margins, salt bodies, or carbonate layers. Recently, a specialized interior point method has been presented for efficiently solving ℓ_1 regularized least squares problems (Kim et al., submitted).

A modified version of that algorithm is presented here. To exemplify its utility it will be used to solve the least squares Super Dix equations, originally presented by Clapp et al. (1998). Expanding on this work,Valenciano et al. (2003) introduced ℓ_1 regularization to the problem formulation using a nonlinear iterative approach that approximated an ℓ_1 regularization. Witten and Grant (2006) solved the same problem using a MATLAB based convex optimization solver. MATLAB, however, was pushed to its limits to solve even this small problem.

In this paper, the algorithm will be described. The method is first applied to a simple synthetic model. Then it is applied to a real data set from the Gulf of Mexico. It initial results compared to previous methods for this same dataset.

THE METHOD

The general form of the problem examined by Kim et al. (submitted) is

$$\text{minimize } |\mathbf{Cu}-\mathbf{d}|_2^2+\Sigma_{i=1}^n\epsilon|\mathbf{u}_i|, \tag{1}$$

where $\mathbf{u}$ is the model, $\mathbf{C}$ is an operator relating the model to the data $\mathbf{d}$ and $\mathbf{u_i}$ is the model value at point i.

In this paper a very similar problem, of the form,

$$\text{minimize } |\mathbf{W}(\mathbf{Cu}-\mathbf{d})|_2^2+\Sigma_{i=1}^n\epsilon_x|\mathbf{D}_x\mathbf{u}_i|+\Sigma_{i=1}^n\epsilon_\tau|\mathbf{D}_\tau\mathbf{u}_i|, \tag{2}$$

will be explored. Here $\mathbf{u}$ is a vector whose components range over vertical travel time depth τ and whose values are the interval velocity squared v_{int}^2 and $\mathbf{d}$ is the data vector which has the same range as $\mathbf{u}$, but whose values are the scaled root-mean squared (RMS) velocities squared, $\tau v_{RMS}^2/\Delta\tau$, where $\tau/\Delta\tau$ is the index on the time axis. $\mathbf{C}$ is the casual integration operator, and $\mathbf{W}$ is a weight matrix which is proportional to our confidence in the RMS velocities. As well, $\mathbf{D_x}$ and $\mathbf{D}_\tau$ are the first order finite-difference derivatives along the midpoint and travel-time axis, respectively, and ϵ_x and ϵ_τ are the regularization parameters that control the importance of the two model residuals, effectively controlling the smoothing.

This problem can be transformed to a convex quadratic problem with linear inequality constraints,

$$\begin{array}{lll} \text{minimize} & |\mathbf{W}(\mathbf{Cu}-\mathbf{d})|_2^2+\Sigma_{i=1}^n\epsilon_x\mathbf{v}_i^x+\Sigma_{i=1}^n\epsilon_\tau\mathbf{v}_i^\tau & \\ \text{subject to} & -\mathbf{v}_i^x\le\mathbf{D}_x\mathbf{u}_i\le\mathbf{v}_i^x & i=1,\dots,n \\ & -\mathbf{v}_i^\tau\le\mathbf{D}_\tau\mathbf{u}_i\le\mathbf{v}_i^\tau & i=1,\dots,n, \end{array} \tag{3}$$

where $v^{x,\tau}$ serve to remove the absolute value from the problem. The new problem (3) can be solved by interior point methods (e.g. (Ye, 1997; Wright, 1997)). With this goal in mind, we can now construct logarithmic barrier functions, which approximate an inequality constraint by increasing to infinity as the point approaches the constraint. For a simple problem,

$$\begin{array}{ll} \text{minimize} & f_0(x) \\ \text{subject to} & f_i(x)\le 0,\quad i=1,\dots,m, \end{array} \tag{4}$$

the logarithmic barrier function is

$$-\left(\frac{1}{t}\right)\log(-f_i(x)), \tag{5}$$

where $t>0$ is a parameter the determines how closely you approximate the constraint (Boyd and Vandenberghe, 2004).

For the bound constraints in equation 3 the barrier functions are:

$$\Phi^x(\mathbf{u},\mathbf{v}^x)=-\Sigma_{i=1}^n\log(\mathbf{v}_i^x+\mathbf{D}_x\mathbf{u}_i)-\Sigma_{i=1}^n\log(\mathbf{v}_i^x-\mathbf{D}_x\mathbf{u}_i)=-\Sigma_{i=1}^n\log(\mathbf{v}_i^{x\,2}-(\mathbf{D}_x\mathbf{u}_i)^2), \tag{6}$$

and

$$\Phi^{\tau}(\mathbf{u},\mathbf{v}^{\tau}) = -\Sigma_{i=1}^{n}\log(\mathbf{v}_i^{\tau}+\mathbf{D}_{\tau}\mathbf{u}_i) - \Sigma_{i=1}^{n}\log(\mathbf{v}_i^{\tau}-\mathbf{D}_{\tau}\mathbf{u}_i) = -\Sigma_{i=1}^{n}\log(\mathbf{v}_i^{\tau\,2}-(\mathbf{D}_{\tau}\mathbf{u}_i)^2), \quad (7)$$

Now we can define the centering problem as,

$$\text{minimize } \phi_t(\mathbf{u},\mathbf{v}^{x},\mathbf{v}^{\tau}) = t|\mathbf{W}(\mathbf{C}\mathbf{u}-\mathbf{d})|_2^2 + t\,\Sigma_{i=1}^{n}\epsilon_x\mathbf{v}_{\mathbf{i}}^{\mathbf{x}} + \mathbf{t}\Sigma_{\mathbf{i=1}}^{\mathbf{n}}\epsilon_{\tau}\mathbf{v}_{\mathbf{i}}^{\tau} + \Phi^{\mathbf{x}} + \Phi^{\tau}. \quad (8)$$

The centering problem is an equivalent representation to problem (3) and has a unique solution parametrized by t, called the central path which leads to an optimal solution (Boyd and Vandenberghe, 2004). Newtons method can now be applied to the centering problem, which involves solving a system on linear equations,

$$H\begin{bmatrix}\Delta\mathbf{u}\\ \Delta\mathbf{v}^{\mathbf{x}}\\ \Delta\mathbf{v}^{\tau}\end{bmatrix} = -g, \quad (9)$$

where $H = \nabla^2\phi_t(\mathbf{u},\mathbf{v}^{\mathbf{x}},\mathbf{v}^{\tau})$ is the Hessian and $g = \nabla\phi_t(\mathbf{u},\mathbf{v}^{\mathbf{x}},\mathbf{v}^{\tau})$ is the gradient. Conjugate gradients is used to find an approximate solution to this system. We differ from Kim et al. (submitted) by choosing not to solve the whole system with conjugate gradients. Instead, $\mathbf{v}^{\mathbf{x}}$ and $\mathbf{v}^{\tau}$ will be solved analytically, decreasing the size of the system of equations needed to be solved solve from $3n$ to n, substantially reducing computational time.

'To solve for $\mathbf{v}^{\mathbf{x}}$ analytically, take the derivative of $\phi_t(\mathbf{u},\mathbf{v}^{\mathbf{x}},\mathbf{v}^{\tau})$ with respect to $\mathbf{v}^{\mathbf{x}}$, then solve for $\mathbf{v}^{\mathbf{x}}$. This gives

$$\mathbf{v}_i^{x} = \frac{1}{t\epsilon_x} + \sqrt{\left(\frac{1}{t\epsilon_x}\right)^2 + (\mathbf{D}_x\mathbf{u}_i)^2}. \quad (10)$$

The same can be done for $\mathbf{v}^{\tau}$. We can also write the Hessian and gradient succinctly as,

$$H = t\nabla^2|\mathbf{W}(\mathbf{C}\mathbf{u}-\mathbf{d})|_2^2 + \nabla^2\Phi^{x}\left(\mathbf{u},\mathbf{v}^{\mathbf{x}}\right) + \nabla^2\Phi^{\tau}\left(\mathbf{u},\mathbf{v}^{\tau}\right) = 2t\mathbf{W}\mathbf{C}^{\mathbf{T}}\mathbf{C} + \mathbf{D}, \quad (11)$$

where

$$\mathbf{D} = diag\left(2\left(\frac{\mathbf{v}_1^{x\,2}+(\mathbf{D}_x\mathbf{u}_1)^2}{\mathbf{v}_{\mathbf{1}}^{\mathbf{x}2}-(\mathbf{D}_x\mathbf{u}_1)^2} + \frac{\mathbf{v}_1^{\tau\,2}+(\mathbf{D}_\tau\mathbf{u}_1)^2}{\mathbf{v}_1^{\tau\,2}-(\mathbf{D}_\tau\mathbf{u}_1)^2}\right),\ldots,2\left(\frac{\mathbf{v}_n^{x\,2}+(\mathbf{D}_x\mathbf{u}_n)^2}{\mathbf{v}_n^{x\,2}-(\mathbf{D}_x\mathbf{u}_n)^2} + \frac{\mathbf{v}_n^{\tau\,2}+(\mathbf{D}_\tau\mathbf{u}_n)^2}{\mathbf{v}_n^{\tau\,2}-(\mathbf{D}_\tau\mathbf{u}_n)^2}\right)\right) \quad (12)$$

where $diag$ denotes a diagonal matrix with elements $2\left(\frac{\mathbf{v}_1^{x\,2}+(\mathbf{D}_x\mathbf{u}_1)^2}{\mathbf{v}_1^{x\,2}-(\mathbf{D}_x\mathbf{u}_1)^2} + \frac{\mathbf{v}_1^{\tau\,2}+(\mathbf{D}_\tau\mathbf{u}_1)^2}{\mathbf{v}_1^{\tau\,2}-(\mathbf{D}_\tau\mathbf{u}_1)^2}\right)$. The gradient can be written as

$$\begin{aligned} g = & \quad \nabla_u\phi_t(\mathbf{u},\mathbf{v}^{x},\mathbf{v}^{\tau}) \\ = & \quad t(2\mathbf{W}\mathbf{C}^{T}(\mathbf{C}\mathbf{u}-\mathbf{d}) + \left[\frac{2\mathbf{u_i}}{\mathbf{v_i^{x2}}-\mathbf{D_x}\mathbf{u_i^2}} + \frac{2\mathbf{u_i}}{\mathbf{v_i^{\tau\,2}}-\mathbf{D}_\tau\mathbf{u}_i^2},\ldots,\frac{2\mathbf{u_i}}{\mathbf{v_i^{x2}}-\mathbf{D_x}\mathbf{u_i^2}} + \frac{2\mathbf{u_i}}{\mathbf{v_i^{\tau\,2}}-\mathbf{D}_\tau\mathbf{u}_i^2}\right]^{T}. \end{aligned} \quad (13)$$

Since the Hessian is constructed from more than the linear operator (it incorporates the barrier functions), matrix multiplication is used to solve the system of equations in the Newton

system. Thus each step of the conjugate gradient is slow, but time is saved by reducing the overall number of conjugate gradient steps.

The final algorithm is:
given the update parameters for t, set the initial values $t = 1/\epsilon$, $\mathbf{u} = \mathbf{0}$, and $\mathbf{v}^{\mathbf{x},\tau} = \mathbf{1}$.

repeat
1. Calculate $\mathbf{v}^{\mathbf{x}}$ and $\mathbf{v}^{\tau}$
2. Compute the search direction Δu using conjugate gradients
3. Compute the step size s by backtracking line search
4. Update $\mathbf{u} = \mathbf{u} + \mathbf{s}(\Delta \mathbf{u})$
5. Calculate $\mathbf{v}^{\mathbf{x}}$ and $\mathbf{v}^{\tau}$ update
6. Evaluate the duality gap and quit if appropriate (see Boyd and Vandenberghe (2004) for more on Duality)
7. Update t

SYNTHETIC EXAMPLE

A synthetic data example is created to test the algorithm to make sure that it works properly. A simple layer-cake earth model is used, shown in Figure 1. RMS velocities are then created from this model as input data to the algorithm. If the inversion is run on this simple model, the result is almost perfect, as shown in Figure 2. The inversion is off by a maximum of 3%, which occurs at the bottom-most interface. This error could most likely be reduced further if we decrease the stopping criterion.

Now 1 and 5 percent Gaussian noise is added to the RMS velocities to simulate real data. The inversion of this noisy data with very little smoothness applied is shown in Figures 3 and 4. The noise introduced to the model shows up as block features. As more noise is added the layers become harder to distinguish from each other. If we increase the smoothing parameters on the ℓ_1 regularization, then much of the noise is smoothed out in the result (Figure 5 and Figure 6). If the regularization parameters, $\epsilon_{x,\tau}$ are increased further then the result will be even smoother (Figure 7 and Figure 8).

As seen in these examples, it is important to correctly choose the regularization parameter to get a good inversion result that is compromise between desired blockiness and introducing spurious elements into the model in the form high spatial frequency events. It can be seen that not all the noise is smoothed out in either Figure 7 or Figure 8. This is because if boundaries are sharp then the ℓ_1 regularization preserves them. The sharper the boundary, the higher the ϵ needs to be smooth them out. Much of the sharp contrast, however, is also smoothed away. From this test it became clear the smoothing along the midpoint is not currently working properly.

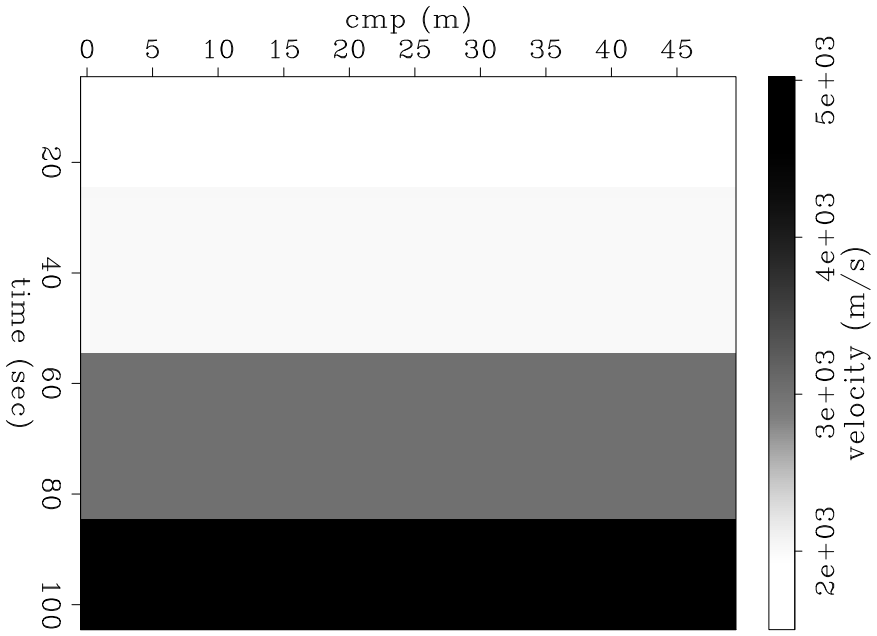

Figure 1: A simple layer cake earth model ben2-model [ER]

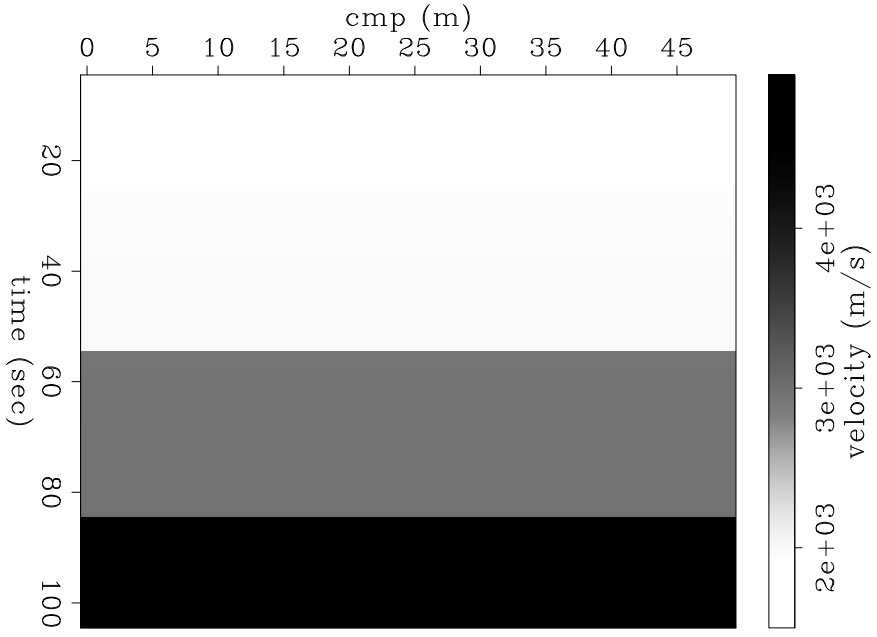

Figure 2: Inversion result for simple model using the algorithm above. For the simple model shown in Figure 1, the result is almost perfect. ben2-Modvint [ER]

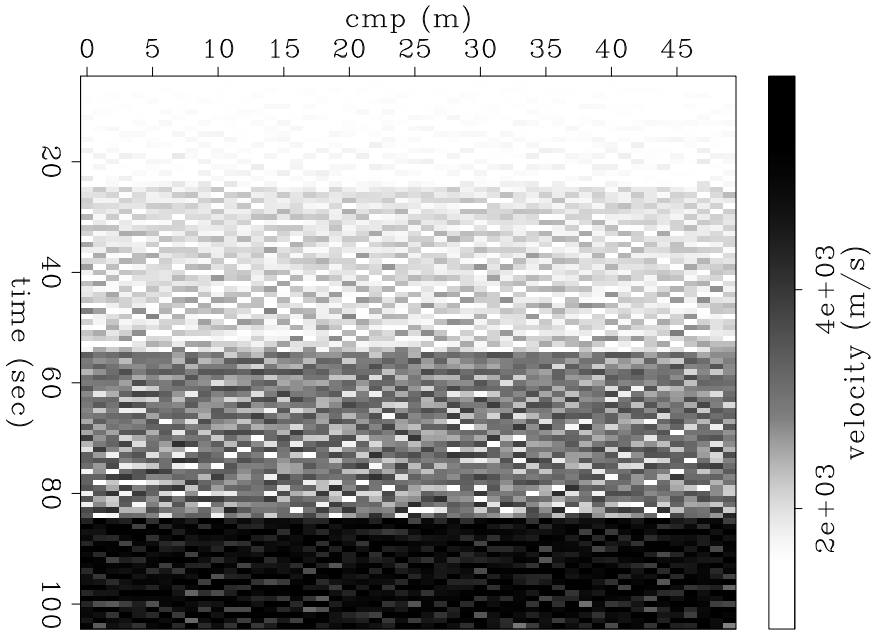

Figure 3: Inversion of the data with 1% noise. The layer boundaries are still visible, but the noise pollutes the results substantially. ben2-ModvintNoise [ER]

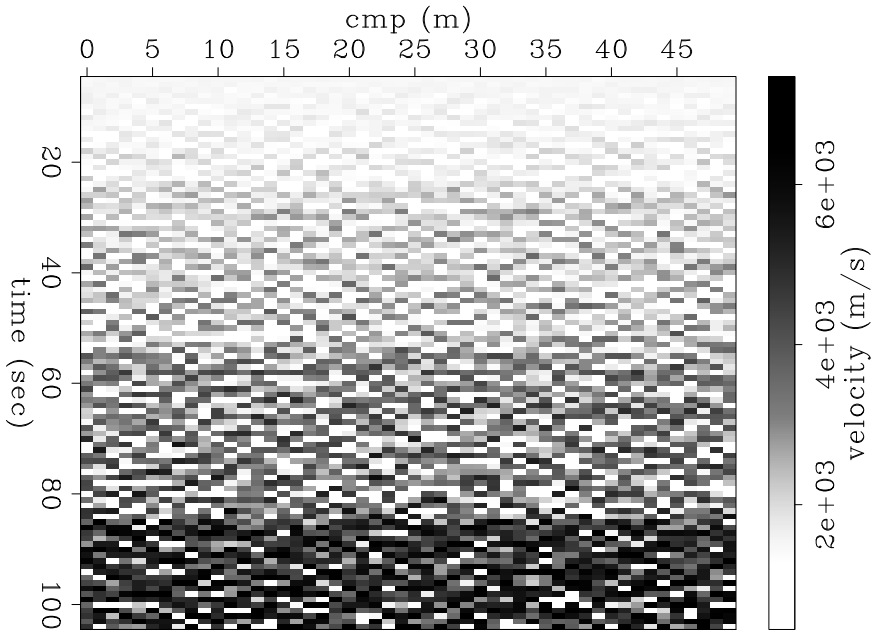

Figure 4: Inversion of noisy data with 5% noise. The noise is severe enough that the boundary layers are no longer discernible. ben2-ModvintNoise5 [ER]

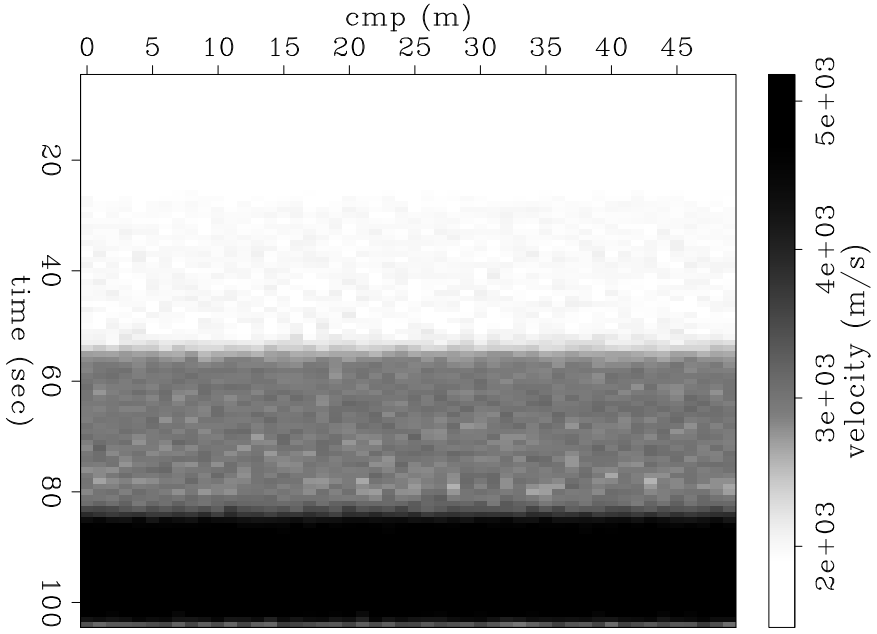

Figure 5: Smoothed version the data with 1% noise. Much of the noise has been smoothed out, but the sharp boundary contacts are still clear. ben2-ModvintNoiseSmooth [ER]

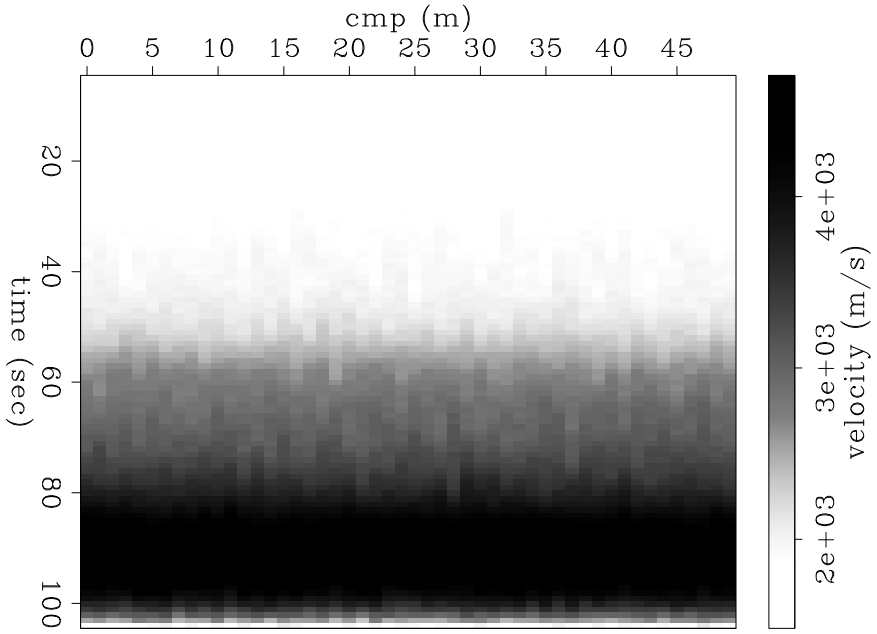

Figure 6: Smoothed version of the data with 5% noise. The smoothing parameter had to be increased to get rid of much of the noise. The boundaries are more evident here than in Figure 6, but the layer boundaries are smoothed out. ben2-ModvintNoiseSmooth5 [ER]

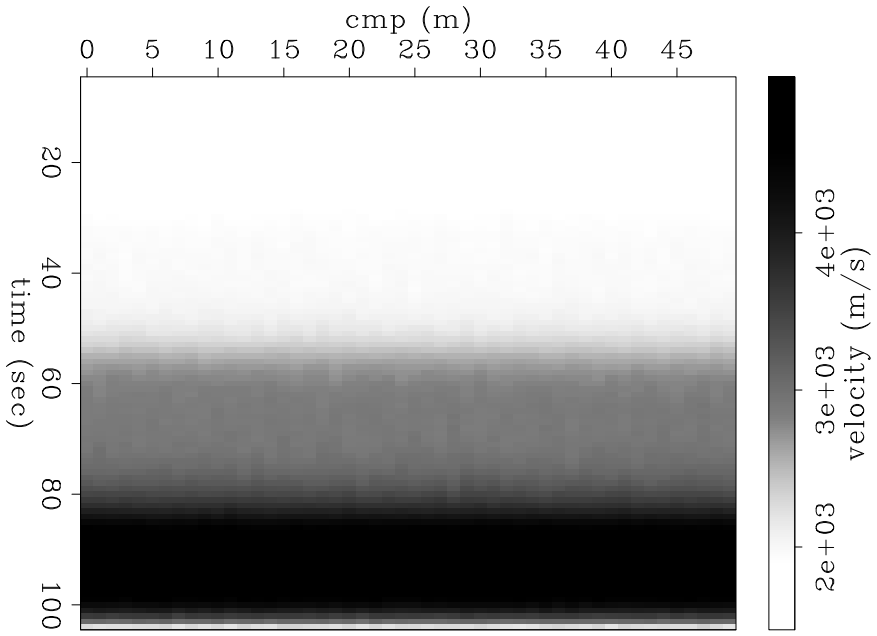

Figure 7: The smoothing parameters were increased further for the 1% noise. Most of the noise is no longer visible, but the layer boundaries are not as sharp. ben2-ModvintNoiseSmooth2 [ER]

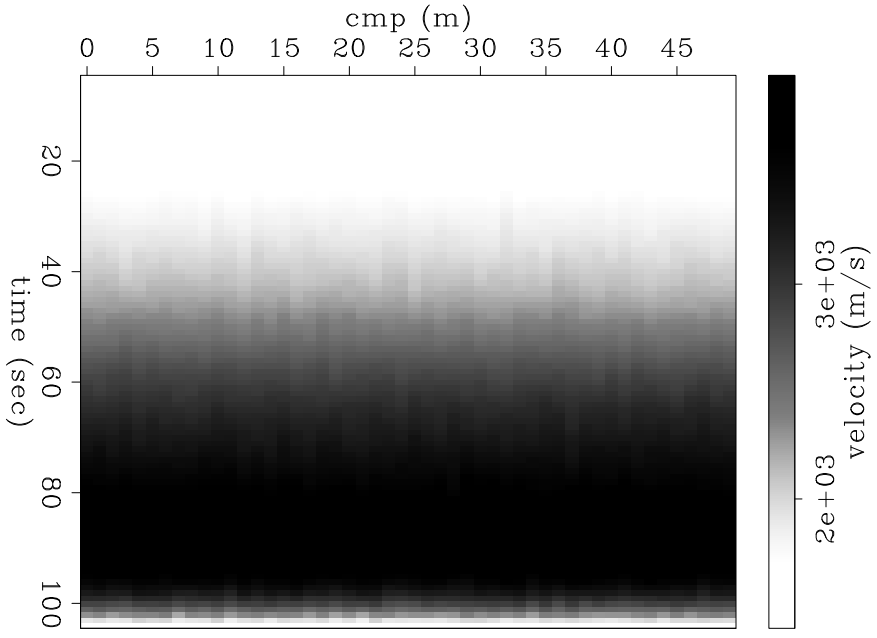

Figure 8: Smoothing parameters were increased for the 5% noise to the point that almost no noise is visible. Doing this, however, has smoothed out the entire result. ben2-ModvintNoiseSmooth2-5 [ER]

REAL DATA EXAMPLE

Real data from the Gulf on Mexico will now be examined. Figure 9 shows the RMS velocity. Figures 10 and 11 show the previous inversion results done by Valenciano et al. (2003) and

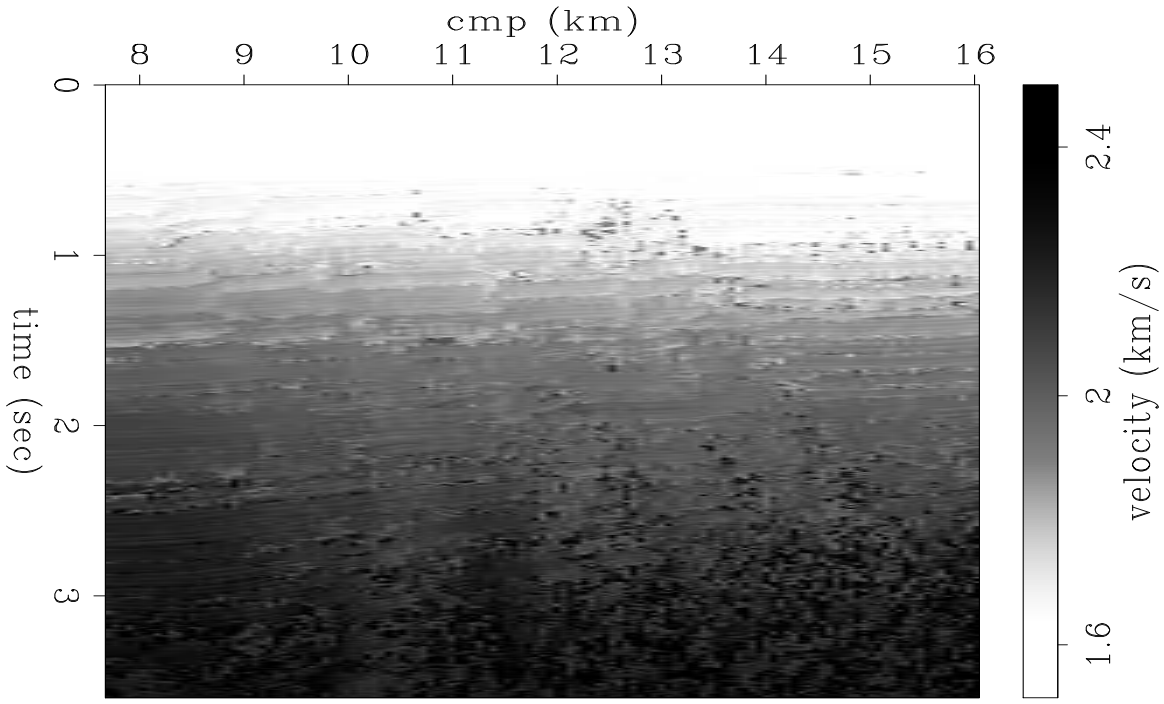

Figure 9: RMS velocity for Gulf of Mexico data. ben2-vrms [ER]

Witten and Grant (2006), respectively. Notice that Figure 10 is smoother than Figure 11. This is because Figure 10 uses an approximate ℓ_1 norm.

The results from both previous inversion schemes have limitations. Valenciano et al. (2003) uses an approximate ℓ_1 norm and has no stopping criterion. Thus it requires as many iterations as the user is willing to execute. For the result shown here, 800 conjugate gradient steps were taken. Witten and Grant (2006) is hampered by the use of the MATLAB based ıcvx software (Grant et al., 2006), limiting its use to small problems.

Although Figure 12 is not perfect, the inversion result for the method presented in this paper is close to the previous results. The result in Figure 12 took only 108 iterations. All of the major features are present in all three inversion results. The main difference is in the smoothness of the result.

CONCLUSIONS AND FUTURE WORK

The algorithm presented in this paper offers a fast and efficient method for ℓ_1 regularized inversion problems. By using ℓ_1 regularization the result is often more consistent geologically

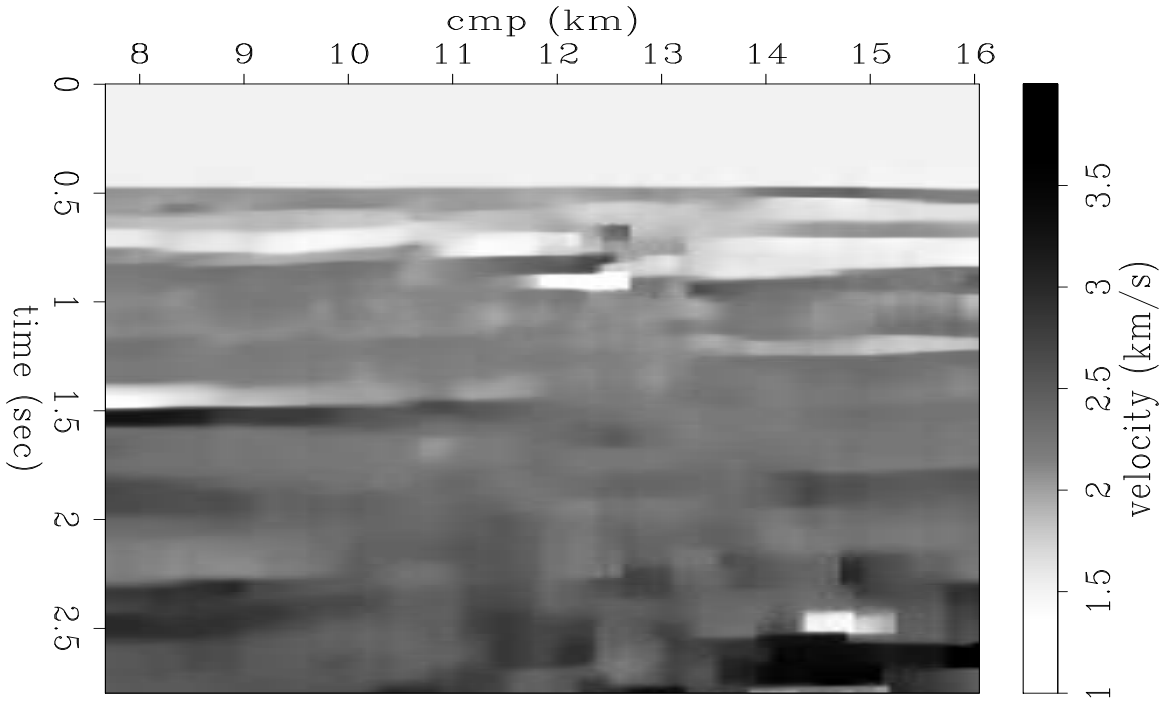

Figure 10: Inversion result from Valenciano et al. (2003). ben2-vintOld [ER]

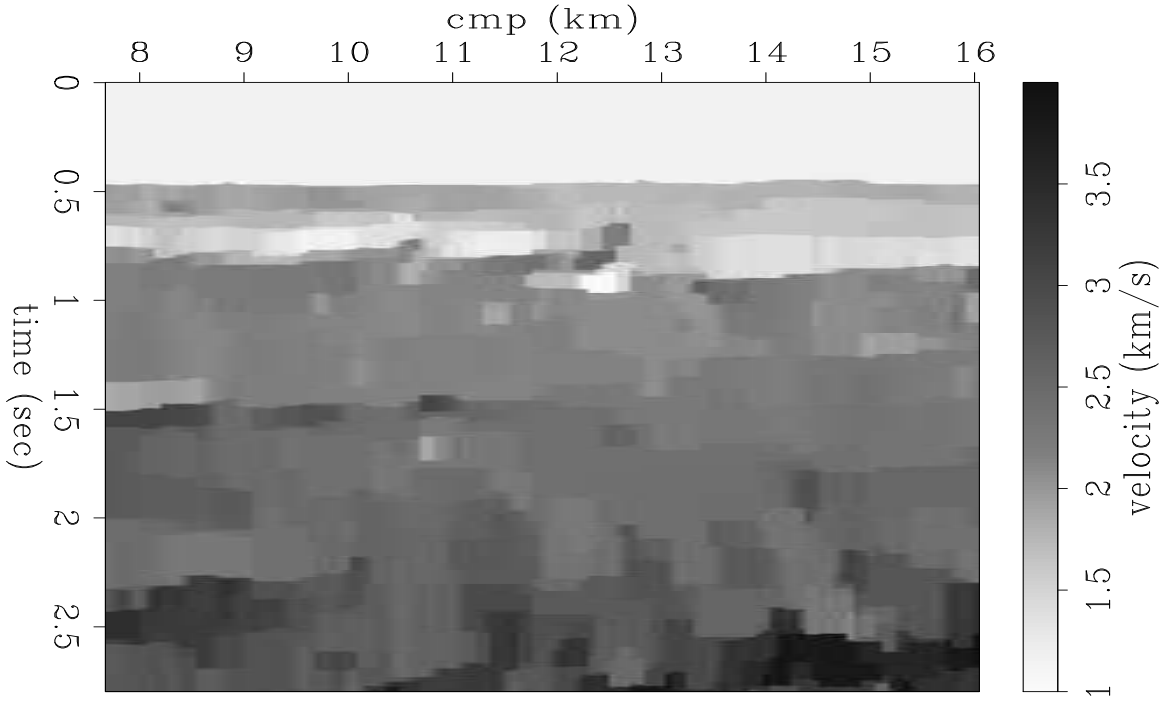

Figure 11: Inversion result from Witten and Grant (2006). ben2-vintOldcvx [CR]

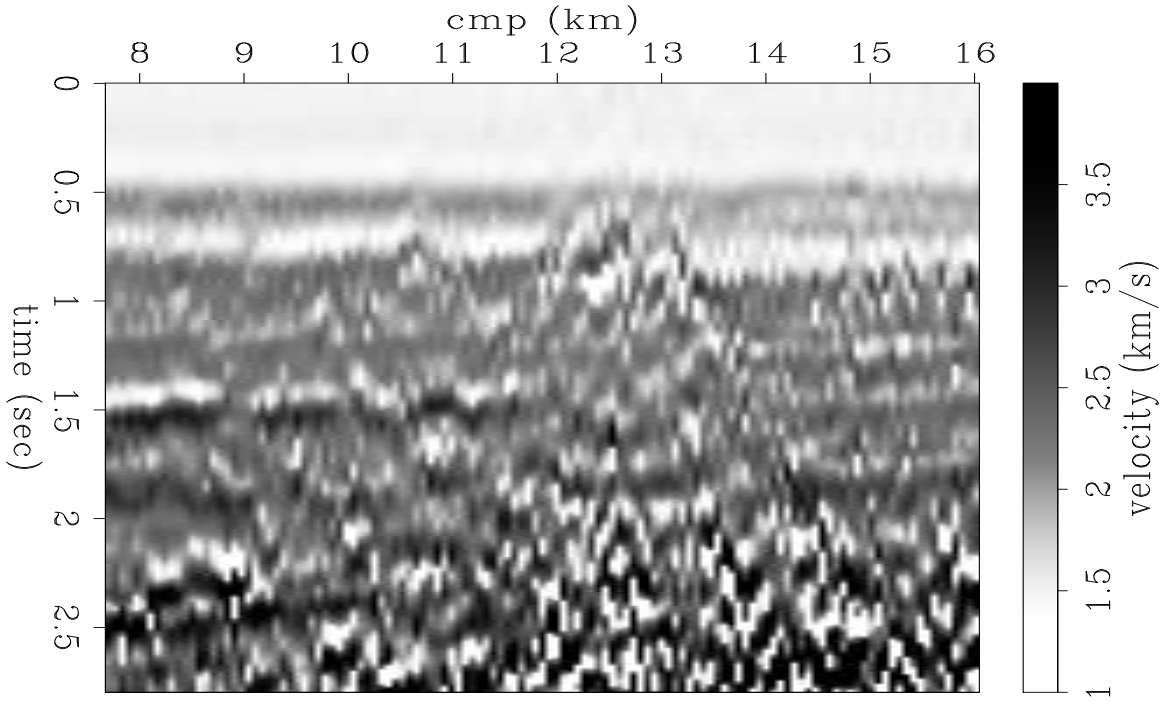

Figure 12: Inversion result from scheme presented in this paper. The inversion appears to be working properly as all of the main features seen in the Figures 10 and 11. The smoothing along the midpoint axis does not work. ben2-vint [ER]

because the sharp boundaries are preserved. Even though this method reduces the number of conjugate gradients steps, it could be slow since it must use matrix multiplication. This can be overcome, however, if we implement the matrix multiplication in a parallel fashion as presented by Lomask and Clapp (2006).

Further work involves applying the smoothness along the midpoint-axis and adding hard equality and inequality constraints to the inversion, limiting the range of velocities acceptable in different parts of the model. This would ensure that bad data did not put an impossible velocity in a particular area, such as an ultra-low velocity at great depths. It would also allow for other forms of data, especially ground truth through well logs, to be easily incorporated into the inversion.

ACKNOWLEDGMENTS

I would like to Michael Grant for his help throughout this project.

REFERENCES

Boyd, S. and L. Vandenberghe, 2004, Convex optimization: Cambridge University Press.

Clapp, R. G., P. Sava, and J. F. Claerbout, 1998, Interval velocity estimation with a null-space: SEP–**97**, 147–156.

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

Grant, M., S. Boyd, and Y. Ye, 2006, cvx: Matlab software for disciplined convex programming: http://www.stanford.edu/ boyd/cvx/.

Kim, S., a. L. M. Koh, H., S. Boyd, and D. Gorinevsky, submitted, A method for large-scale ℓ_1-regularized least squares problems with applications in signal processing and statistics:.

Lomask, J. and R. G. Clapp, 2006, Parallel implementation of image segmentation for tracking 3d salt boundaries: SEP–**124**.

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

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

Witten, B. and M. Grant, 2006, Interval velocity estimation through convex optimization: SEP-**125**.

Wright, S., 1997, Primal-daul interior-point methods: Society for Industrial and Applied Mathematics, Philadelphia, PA, USA.

Ye, Y., 1997, Interior point algorithms: theory and analysis: John Wiley & Sons.