Conjugate guided gradient (CGG) method for robust inversion and its application to velocity-stack inversion [a]

[a]Published in Geophysics, 71, no. 4, R59-R67, (2006)

Jun Ji

ABSTRACT

This paper proposes a modified conjugate gradient (CG) method, called the conjugate guided gradient (CGG) method, that is a robust iterative inversion method producing a parsimonious model estimation. The CG method for solving least-squares (LS) (i.e. ℓ^2-norm minimization) problems is modified to solve for different norms or different minimization criteria by guiding the gradient vector appropriately during iteration steps. The guiding is achieved by iteratively reweighting either the residual vector or the gradient vector during iteration steps like IRLS (Iteratively Reweighted Least Squares) method does. The robustness is achieved by weighting the residual vector and the parsimonious model estimation is obtained by weighting the gradient vector. Unlike the IRLS method, however, the CGG method doesn't change the corresponding forward operator of the problem and is implemented in a linear inversion template. Therefore the CGG method requires less computation than the IRLS method does. Since the solution in the CGG method is found in a least-squares sense along the gradient direction guided by the weights, the solution found by the CGG method can be interpreted as the LS solution located in the guided gradient direction. Guiding the gradient gives us more flexibility in choice of weighting parameters than the one of the IRLS method. I applied the CGG method to velocity-stack inversion, and the results show that the CGG method gives a far more robust and parsimonious model estimation than the standard ℓ^2-norm solution, with the results comparable to the ℓ^1-norm IRLS solution.

INTRODUCTION

The inverse problem has received considerable attention in various geophysical applications. One of the most popular inverse solutions is the least squares (LS) solution. The LS solution is a member of a family of generalized ℓ^p-norm solutions that are deduced from a maximum-likelihood formulation. This formulation allows the design of various statistical inversion solutions. Among the various ℓ^p-norm solutions, the ℓ^1-norm solution is more robust than the ℓ^2-norm solution, because it is less sensitive

to spiky, high-amplitude noise Claerbout and Muir (1973); Taylor et al. (1979); Scales and Gersztenkorn (1987); Scales et al. (1988). In order to take advantages of both ℓ^2 and ℓ^1 norm solutions, hybrid ℓ^1/ℓ^2 - norm solutions are also tried Huber (1973); Bube and Langan (1997); Guiiton and Symes (2003). However, the implementation of the algorithm to find ℓ^1-norm solutions is not a trivial task, which uses linear programming techniques Taylor et al. (1979) and needs a large quantity of computer memory. Iterative inversion algorithm called Iteratively Reweighted Least Squares (IRLS) method Gersztenkorn et al. (1986); Scales and Gersztenkorn (1987); Scales et al. (1988); Bube and Langan (1997) is a good choice for solving ℓ^p-norm minimization problems for $1 \leq p \leq 2$. The IRLS approach which was originally developed for nonlinear inversion can be adapted to solve linear inverse problems in ℓ^p-norm sense by modifying the iterative inversion method such as conjugate gradient (CG) method Darche (1989); Nichols (1994); Claerbout (2004).

The ℓ^p-norm minimizing IRLS inversion can be used to any inversion problem whose required properties are the robustness to spiky noise and the parsimony of the model, and the velocity-stack inversion is one of them. The velocity-stack inversion is useful not only for velocity analysis but also for various data processing applications. The applications of the velocity-stack inversion include non-hyperbolic noise removal in CMP gathers Nichols (1994); Guiiton and Symes (2003), multiple-removal Thorson and Claerbout (1985); Hampson (1986); Foster and Mosher (1992); Kostov and Nichols (1995); Lumley et al. (1995); Kabir and Marfurt (1999); Herrmann et al. (2000) and missing offset reconstruction Ji (1994); Sacchi and Ulrych (1995), and so on. In these applications, the velocity-stack panels obtained by inversion are usually required to be as spiky and sparse as possible. Then the hyperbolic events represented by the isolated peaks in the velocity-stack panel are more easily distinguished from the rest of the noise.

This paper introduces a modification of the conventional CG method for solving LS problem so as to be robust and produce a parsimonious model estimation. The modified CG method is called Conjugate Guided Gradient (CGG) method. The modification of the CG method is performed by guiding the gradient vector during the iteration steps. Guiding the gradient vector is achieved by iteratively reweighting either the residual vector or the gradient vector during iteration steps like IRLS (Iteratively Reweighted Least Squares) method does. The weighting of the residual vector makes the CGG method robust and the weighting of the gradient vector makes the CGG method produce a parsimonious model estimation. In the first section, I review the conventional CG method for solving LS problems and show how the IRLS approach differs from the standard LS approach. Next, I explain the CGG method and contrast it with both LS and IRLS methods. Finally the proposed CGG method is tested on velocity-stack inversions with both synthetic and real data and the results of the CGG method are compared with conventional LS and ℓ^1-norm IRLS results.

CG METHOD FOR LS INVERSION

Most inversion problems start by formulating the forward problem, which describes the forward operator $\mathbf{L}$ that transforms the model vector $\mathbf{m}$ into the data vector $\mathbf{d}$

$$\mathbf{d} = \mathbf{L}\mathbf{m}. \tag{1}$$

In general, the measured data $\mathbf{d}$ may be inexact, and the forward operator $\mathbf{L}$ may be ill-conditioned. In that case, instead of solving the above equation directly, different approaches are used to find an optimum solution $\mathbf{m}$ for given data $\mathbf{d}$. The most popular method is finding a solution that minimizes the misfit between the data $\mathbf{d}$ and the modeled data $\mathbf{Lm}$. The misfit is often referred as residual vector $\mathbf{r}$ and is described as follows:

$$\mathbf{r} = \mathbf{L}\mathbf{m} - \mathbf{d}. \tag{2}$$

In least-squares inversion, the solution $\mathbf{m}$ is the one that minimizes the squares of the residual vector as follows:

$$min_m(\mathbf{r}^T r) = min_m(\mathbf{L}\mathbf{m} - \mathbf{d})^T(\mathbf{L}\mathbf{m} - \mathbf{d}). \tag{3}$$

Most iterative solvers for the LS problem search the minimum solution on a line or a plane in the solution space. In the CG algorithm, not a line, but rather a plane is searched. A plane is made from an arbitrary linear combination of two vectors. One vector is chosen to be the gradient vector. The other vector is chosen to be the previous descent step vector. Following Claerbout (1992), a conjugate-gradient algorithm for the LS solution can be summarized as shown in Algorithm 1.

Algorithm 1 CG method for LS solution

```
r ⇐ Lm − d
while condition do
  Δm ⇐ L^T r
  Δr ⇐ LΔm
  (m, r) ⇐ cgstep(m, r, Δm, Δr)
end while
```

In Algorithm 1, the condition represents a convergence check such as the tolerance of residual vector $\mathbf{r}$, a maximum number of iteration, and so on. The subroutine cgstep() updates model $\mathbf{m}$ and residual $\mathbf{r}$ using the previous iteration descent vector in the conjugate space $\Delta\mathbf{s} = L(\mathbf{m}_i - \mathbf{m}_{i-1})$, where i is the iteration step, and the conjugate gradient vector $\Delta\mathbf{r}$. The update step size is determined by minimizing the quadrature function composed from $\Delta\mathbf{r}$ (the conjugate gradient) and $\Delta\mathbf{s}$ (the previous iteration descent vector in the conjugate space) as follows Claerbout (1992):

$$Q(\alpha, \beta) = (\mathbf{r} - \alpha\Delta\mathbf{r} - \beta\Delta\mathbf{s})^T(\mathbf{r} - \alpha\Delta\mathbf{r} - \beta\Delta\mathbf{s}).$$

Notice that the gradient vector ($\Delta\mathbf{m}$) in the CG method for LS solution is the gradient of the squared residual and is determined by taking the derivative of the squared residual (i.e. the ℓ^2-norm of the residual, $\mathbf{r}^T\mathbf{r}$) with respect to the model $\mathbf{m}^T$:

$$\Delta\mathbf{m} = \frac{\partial}{\partial\mathbf{m}^T}(\mathbf{Lm}-\mathbf{d})^T(\mathbf{Lm}-\mathbf{d}) = \mathbf{L}^T\mathbf{r}. \tag{4}$$

CG method for Iteratively Reweighted Least Squares (IRLS)

Instead of the ℓ^2-norm solution obtained by the conventional LS method, ℓ^p-norm minimization solutions, with $1 \le p \le 2$, are often tried. Iterative inversion algorithms called IRLS (Iteratively Reweighted Least Squares) algorithms have been developed to solve these problems, which lie between the least-absolute-values problem and the classical least-squares problem. The main advantage of IRLS is that it provides an easy way to compute the approximate ℓ^p-norm solution. Among the various ℓ^p-norm solutions, ℓ^1-norm solutions are known to be more robust than ℓ^2-norm solutions, being less sensitive to spiky, high-amplitude noise Claerbout and Muir (1973); Taylor et al. (1979); Scales and Gersztenkorn (1987); Scales et al. (1988).

The problem solved by IRLS is a minimization of the weighted residual/model in the least-squares sense. The residual to be minimized in the weighted problem is described as

$$\mathbf{r} = \mathbf{W}_r(\mathbf{LW}_m\mathbf{m}-\mathbf{d}) \tag{5}$$

where $\mathbf{W}_r$ and $\mathbf{W}_m$ are the weights for residual and model, respectively. These residual and model weights are for enhancing our preference regarding the residual and model. They can be applied separately or together according to a given inversion goal. In this section, for simplicity, the explanation will be limited to the case of applying both weights together, but the examples given in a later section will show all the cases including the residual and the model weights separately for comparison. Those weights can be any matrices, but diagonal matrices are often used for them and this paper will assume all weights are diagonal matrices. Then the gradient for the weighted least-squares becomes

$$\mathbf{W}_m^T\mathbf{L}^T\mathbf{W}_r^T\mathbf{r} = \frac{\partial}{\partial\mathbf{m}^T}(\mathbf{LW}_m\mathbf{m}-\mathbf{d})^T\mathbf{W}_r^T\mathbf{W}_r(\mathbf{LW}_m\mathbf{m}-\mathbf{d}). \tag{6}$$

A particular choice for the residual weight $\mathbf{W}_r$ is the one that results in minimizing the ℓ^p-norm of the residual. Choosing the i^{th} diagonal element of $\mathbf{W}_r$ to be a function of the i^{th} component of the residual vector as follows:

$$diag(\mathbf{W}_r)_i = |r_i|^{(p-2)/2}, \tag{7}$$

the ℓ^2-norm of the weighted residual is then

$$\|\mathbf{W}_r r\|_2^2 = \mathbf{r}^T\mathbf{W}_r^T\mathbf{W}_r\mathbf{r} = \mathbf{r}^T\mathbf{W}_r^2\mathbf{r} = \|\mathbf{r}\|_p^p. \tag{8}$$

Therefore, the minimization of the ℓ^2-norm of the weighted residual with an weight as shown Equation (7) can be thought as a minimization of the ℓ^p-norm of the residual. This method is valid for ℓ^p-norms where $1 \leq p \leq 2$. When the ℓ^1-norm is desired, the weighting is as follows:

$$diag(\mathbf{W}_r)_i = |r_i|^{-1/2}.$$

This weight will reduce the contribution of large residuals and improve the fit to the data that is already well-estimated. Thus, the ℓ^1-norm-based minimization is robust, less sensitive to noise bursts in the data. In practice the weighting operator is modified slightly to avoid dividing by zero. For this purpose, a damping parameter ϵ is chosen and the weighting operator is modified to be:

$$diag(\mathbf{W}_r)_i = \begin{cases} |r_i|^{-1/2}, & |r_i| > \epsilon \\ \epsilon, & |r_i| \leq \epsilon \end{cases}$$

The choice of this parameter is related to the distribution of the residual values. Some authors choose it as a relatively small value like $\epsilon = \max|\mathbf{d}|/100$ and others choose it as a value that corresponds to a small percentile of data as 2 percentile (which is the value with 98% of the values above and 2% below). In this paper, I used the percentile approach to decide the parameter ϵ because it can reflect the distribution of the residual values in it and shows more stable behavior in the experiments performed in this paper.

The use of the model weight $\mathbf{W}_m$ is to enhance our preference regarding the model, for example the parsimony or the smoothness of the solution. The introduction of the model weight corresponds to applying precondition and solving the problem

$$\mathbf{L}\mathbf{W}_m\hat{\mathbf{m}} = d$$

followed by

$$\mathbf{m} = \mathbf{W}_m\hat{\mathbf{m}}.$$

The iterative solution of this system minimizes the energy of the new model parameter $\hat{\mathbf{m}}$

$$\|\hat{\mathbf{m}}\|_2^2 = \hat{\mathbf{m}}^T\hat{\mathbf{m}} = \mathbf{m}^T\mathbf{W}_m^{-T}\mathbf{W}_m^{-1}\mathbf{m}.$$

In the same vein as the residual weight, the model weight $\mathbf{W}_m$ can be chosen as

$$diag(\mathbf{W}_m)_i = |m_i|^{(2-p)/2}. \tag{9}$$

Then the weighted model energy that is minimized is now

$$\mathbf{m}^T\mathbf{W}_m^{-2}\mathbf{m} = \|\mathbf{m}\|_p^p,$$

which is the ℓ^p-norm of the model. When the minimum ℓ^1-norm model is desired, the weighting is as follows:

$$diag(\mathbf{W}_m)_i = |m_i|^{1/2}.$$

The IRLS method can be easily incorporated in CG algorithms by including the weights $\mathbf{W}_r$ and $\mathbf{W}_m$ such that the operator $\mathbf{L}$ has a postmultiplier $\mathbf{W}_r$ and a premultiplier $\mathbf{W}_m$ and the adjoint operator $\mathbf{L}^T$ has a premultiplier $\mathbf{W}_m^T$ and postmultiplier $\mathbf{W}_r^T$ Claerbout (2004). However, the introduction of weights that change during iterations leads us to implement a nonlinear CG method with two nested loops. The outer loop is for the iteration of changing weights and the inner loop is for the iteration of the LS solution for a given weighted operator. Even though we do not know the real residual/model vector at the beginning of the iteration, we can approximate the real residual/model with a residual/model of the previous iteration step, and it will converge to a residual/model that is very close to the real residual/model as the iteration step continues. This method can be summarized as Algorithm 2, where $f(\mathbf{r})$ and $f(\mathbf{m})$ represent functions of residual and model described in Equation (7) and Equation (9), respectively.

Algorithm 2 CG method for IRLS solution

$\mathbf{r} \Leftarrow \mathbf{Lm} - \mathbf{d}$
while condition **do**
  $diag(\mathbf{W}_r) \Leftarrow \mathrm{f}(\mathbf{r})$
  $diag(\mathbf{W}_m) \Leftarrow \mathrm{f}(\mathbf{m})$
  $\mathbf{r} \Leftarrow \mathbf{W}_r(\mathbf{LW}_m\mathbf{m} - \mathbf{d})$
  while condition **do**
    $\mathbf{\Delta m} \Leftarrow \mathbf{W}_m^T\mathbf{L}^T\mathbf{W}_r^T\mathbf{r}$
    $\mathbf{\Delta r} \Leftarrow \mathbf{W}_r\mathbf{LW}_m\mathbf{\Delta m}$
    $(\mathbf{m}, \mathbf{r}) \Leftarrow \mathrm{cgstep}(\mathbf{m}, \mathbf{r}, \mathbf{\Delta m}, \mathbf{\Delta r})$
  end while
  $\mathbf{m} \Leftarrow \mathbf{W}_m\mathbf{m}$
end while

For efficiency, Algorithm 2 is often implemented not to wait until the convergence of the inner loop and instead implemented to finish the inner loop after a certain number of iterations and recomputes the weights and the corresponding residual Darche (1989); Nichols (1994). In order to take advantage of plane search in CG, however, the number of the iterations of the inner loop should be more than or equal to two. The experiments performed for the examples of this paper have shown almost no differences between the results of different iteration steps of the inner loop. In this paper, therefore, the IRLS algorithm is implemented to finish the inner loop after two iterations.

CONJUGATE-GUIDED-GRADIENT (CGG) METHOD

From the algorithmic viewpoint of the CG method, the IRLS algorithm can be considered as an LS method, but with its operator, $\mathbf{L}$, modified by the weights, $\mathbf{W}_r$ and $\mathbf{W}_m$. The only change in the problems to solve that distinguishes the IRLS algorithm from the LS one is the substitution of $\mathbf{W}_r\mathbf{LW}_m$ and $\mathbf{W}_m^T\mathbf{L}^T\mathbf{W}_r^T$ for $\mathbf{L}$ and

$\mathbf{L}^T$, respectively. Since the weights $\mathbf{W}_r$ and $\mathbf{W}_m$ are functions of the residual and the model, respectively, and the residual $\mathbf{r}$ and the model $\mathbf{m}$ are changing during the iteration, the problem that IRLS method solves is a nonlinear problem. Therefore, the IRLS method obtains the ℓ^p-norm solution at the cost of nonlinear implementation. I propose another algorithm that obtains ℓ^p-norm solution without breaking the linear inversion template. Instead of modifying the operator which results in nonlinear inversion, we can choose a way to guide the search to find the minimum ℓ^2-norm solution in a specific model subspace so as to obtain a solution that meets a user's specific criteria. The specific model subspace could be guided by a specific ℓ^p-norm's gradient or constrained by an *a priori* model. Such guiding of the model vector can be realized by weighting the residual vector or gradient vector in the CG algorithm. Since the weights are basically changing the direction of the gradient vector in the CG algorithm, this proposed algorithm is named as Conjugate Guided Gradient (CGG) method.

CGG with residual weight guide

Suppose we apply the same residual weight $\mathbf{W}_r$ as the one we used in the IRLS method, to the residual when we compute the gradient $\mathbf{\Delta m}$ but do not apply the weight when we compute the conjugate gradient $\mathbf{\Delta r}$. This means that we do not change the operator from $\mathbf{L}$ to $\mathbf{W}_r\mathbf{L}$, and the weight affects only the gradient direction. This corresponds to guiding the gradient direction with a weighted residual, and the resultant gradient will be the same gradient as we used for the ℓ^p-norm residual solution in the IRLS method. Unlike the IRLS method, however, we don't need to recompute the residual when the weight has changed since we did not change the operator while the iteration goes and the problem is the same problem as before we change the weight (i.e. we are solving a linear problem). This algorithm can be implemented as shown in Algorithm 3.

Algorithm 3 CGG method with residual weight guide

$\mathbf{r} \Leftarrow \mathbf{Lm} - \mathbf{d}$
while condition **do**
  $\text{diag}(\mathbf{W}_r) \Leftarrow \text{f}(\mathbf{r})$
  $\mathbf{\Delta m} \Leftarrow \mathbf{L}^T\mathbf{W}_r^T\mathbf{r}$
  $\mathbf{\Delta r} \Leftarrow \mathbf{L\Delta m}$
  $(\mathbf{m}, \mathbf{r}) \Leftarrow \text{cgstep}(\mathbf{m}, \mathbf{r}, \mathbf{\Delta m}, \mathbf{\Delta r})$
end while

Notice that Algorithm 3 is different from the original CG method (Algorithm 1) only at the step of gradient $\mathbf{\Delta m}$ computation; the modification of the gradient is performed by changing the residual before the gradient is computed from it. By choosing the weight as a function of the residual of the previous iteration step, as we did in the IRLS method, we can guide the gradient vector to the gradient vector of the ℓ^p-norm. Thus the result obtained by weighting the residual in the CGG method

could be interpreted as a localized LS solution in the subspace composed by the ℓ^p-norm gradient vectors, not in the whole solution space. The minimum ℓ^2-norm location is unlikely to be located along the gradient direction of the different ℓ^p-norm which is guided by the applied weight. Therefore, it is more likely that the solution will be close to the minimum ℓ^p-norm location which is guided by the applied weight.

CGG with model weight guide

Another way to modify the gradient direction is to modify the gradient vector after the gradient is computed from a given residual. Since the gradient vector is in the model space, any modification of the gradient vector imposes some constraint in the model space. If we know some characteristics of the solution which can be expressed in terms of weighting in the solution space, we can use the weight to redirect the gradient vector by applying the weight to it. Again, by keeping the forward operator unchanged, we don't need to recompute the residual when the weight has changed. This algorithm can be implemented as shown in Algorithm 4.

Algorithm 4 CGG method with model weight guide

$\mathbf{r} \Leftarrow \mathbf{Lm} - \mathbf{d}$
while condition **do**
  $\text{diag}(\mathbf{W}_m) \Leftarrow \text{f}(\mathbf{m})$
  $\mathbf{\Delta m} \Leftarrow \mathbf{W}_m^T \mathbf{L}^T \mathbf{r}$
  $\mathbf{\Delta r} \Leftarrow \mathbf{L \Delta m}$
  $(\mathbf{m}, \mathbf{r}) \Leftarrow \text{cgstep}(\mathbf{m}, \mathbf{r}, \mathbf{\Delta m}, \mathbf{\Delta r})$
end while

Even though the model weighting has different meaning from from residual weighting in the inversion result, the analyses are similar in both cases. As we redefined the contribution of each residual element by weighting it with the absolute value of itself to some power, we can do the same thing with each model element in the solution,

$$diag(\mathbf{W}_m)_i = |m_i|^p, \tag{10}$$

where p is a real number that depends on the problem we wish to solve. If the operator used in the inversion is close to unitary, the solution obtained after the first iteration already closely approximates the real solution. Therefore, weighting the gradient with some power of the absolute value of the previous iteration means that we down-weight the importance of small model values and improve the fit to the data by emphasizing model components that already have large values.

CGG with residual and model weights guide

In the previous two subsections, we examined the meaning of weighting the residual vector and the gradient vector, respectively. Since applying the weighting in both

residual space and model space is nothing but changing the direction of the descent for the solution search, the weighting is not limited either to residual or to model space. We can weight both the residual and the gradient as shown in Algorithm 5.

Algorithm 5 CGG method with residual and model weights guide

$\mathbf{r} \Leftarrow \mathbf{Lm} - \mathbf{d}$
while condition **do**
  $\text{diag}(\mathbf{W}_r) \Leftarrow \text{f}(\mathbf{r})$
  $\text{diag}(\mathbf{W}_m) \Leftarrow \text{f}(\mathbf{m})$
  $\mathbf{\Delta m} \Leftarrow \mathbf{W}_m^T \mathbf{L}^T \mathbf{W}_r^T \mathbf{r}$
  $\mathbf{\Delta r} \Leftarrow \mathbf{L \Delta m}$
  $(\mathbf{m}, \mathbf{r}) \Leftarrow \text{cgstep}(\mathbf{m}, \mathbf{r}, \mathbf{\Delta m}, \mathbf{\Delta r})$
end while

Again, Algorithm 5 is different from the conventional CG method (Algorithm 1) only in the step of gradient computation. Whether we modify the gradient in the residual sense or in the model sense, it changes only the gradient direction (i.e. the direction in which the solution is sought) and the solution is found in the least-squares sense in that direction. Therefore, the problem solved by the CGG method is a linear problem and the CGG algorithm always converges to a solution, which is different from the LS solution that is located along the original gradient direction. Notice that the CGG algorithm (Algorithm 5) is simpler than the IRLS algorithm (Algorithm 2), but the CGG method gives a similar solution as the one of the IRLS method, which are demonstrated with examples shown in the following section.

APPLICATION OF THE CGG METHOD IN VELOCITY-STACK INVERSION

The CGG method described in the previous section can be used to any inversion problem whose required properties are the robustness to spiky noise and the parsimony of the model. In this section, the CGG method is tested on a velocity-stack inversion which is useful not only for velocity analysis but also for various data processing applications. The conventional velocity-stack is performed by summing or estimating semblance Taner and Koehler (1969) along the various hyperbolas in a CMP gather, resulting in a velocity-stack panel. Ideally a hyperbola in a CMP gather should be mapped onto a point in a velocity-stack panel. Summation along a hyperbola, or hyperbolic Radon transform (HRT), does not give such resolution. To obtain a velocity-stack panel with better resolution, Thorson and Claerbout (1985) and Hampson (1986) formulated it as an inverse problem in which the velocity domain is the unknown space. If we find an operator $\mathbf{H}$ that transforms a point in a model space (velocity-stack panel) $\mathbf{m}$ into a hyperbola in data space (CMP gather) $\mathbf{d}$,

$$\mathbf{d} = \mathbf{Hm}, \tag{11}$$

and also find its adjoint operator $\mathbf{H}^T$, we can pose the velocity-stack problem as an inverse problem. The adjoint operator $\mathbf{H}^T$ corresponds to the velocity stacking operator for a given range of velocities(or slownesses), which generates a velocity-stack panel and can be described as

$$\mathbf{m}(s,\tau) = \sum_{h=h_{min}}^{h_{max}} \mathbf{d}(h, t = \sqrt{\tau^2 + h^2 s^2}) \tag{12}$$

where $\mathbf{d}(h,t)$ denotes the common-midpoint (CMP) gather and $\mathbf{m}(s,\tau)$ denotes velocity-stack panel. The slowness-time pair (s,τ) are the coordinate axes of the velocity-stack and the offset-time pair (h,t) are the coordinate axes of the CMP gather. A straightforward definition for the forward operator $\mathbf{H}$ is the adjoint of the operator $\mathbf{H}^T$ defined by Equation (12). Through the suitable definition of the inner product, $\mathbf{H}$ turns out to be simply the process of reverse NMO and stacking (Thorson and Claerbout, 1985):

$$\mathbf{d}(h,t) = \sum_{s=s_{min}}^{s_{max}} \mathbf{m}(s, \tau = \sqrt{t^2 - h^2 s^2}). \tag{13}$$

Inverse theory helps us to find an optimal velocity-stack panel which synthesizes a given CMP gather via the operator $\mathbf{H}$. The usual process is to implement the inverse as the minimization of a least-squares problem and calculate the solution by solving the normal equation:

$$\mathbf{H}^T\mathbf{H}\mathbf{m} = \mathbf{H}^T\mathbf{d}. \tag{14}$$

Since the number of equations and unknowns may be large, an iterative least-squares solver such as CG is usually preferred to solving the normal equation directly.

The least-squares solution has some attributes that may be undesirable. If the model space is overdetermined and has busty noise in data, the least-squares solutions usually will be spread over all the possible solutions. Other methods may be more useful if we desire a parsimonious representation of the solution. To obtain a more robust solution, Nichols (1994) and Trad et al. (2003) used the IRLS method for ℓ^1-norm minimization, and Guiiton and Symes (2003) used a quasi-Newton method called limited-memory BFGS Broyden (1969); Fletcher (1970); Goldfarb (1970); Shanno (1970); Nocedal (1980) for Huber-norm minimization. Another possibility is the CGG method proposed in the preceding section. In the next subsections the results of the CGG method for the velocity-stack inversion are compared with the results of conventional LS method and ℓ^1-norm IRLS method.

Examples on synthetic data

To examine the performance of the proposed CGG method, a synthetic CMP data set with various types of noise is used. Figure 1 shows the synthetic data with three types of noise — Gaussian noise in the background, busty spike noises, and a trace with only Gaussian noises. Figure 1 (right) is the same data as Figure 1 (left), but displayed in wiggle format to clearly show the busty spike noises that were not discernible because

of the clipping in the raster format display. The relative amplitudes of three noise types are compared to the maximum amplitude of the hyperbolic data: ten times for the busty spikes, two times for the noisy trace, and 0.2 times for the Gaussian noise, respectively.

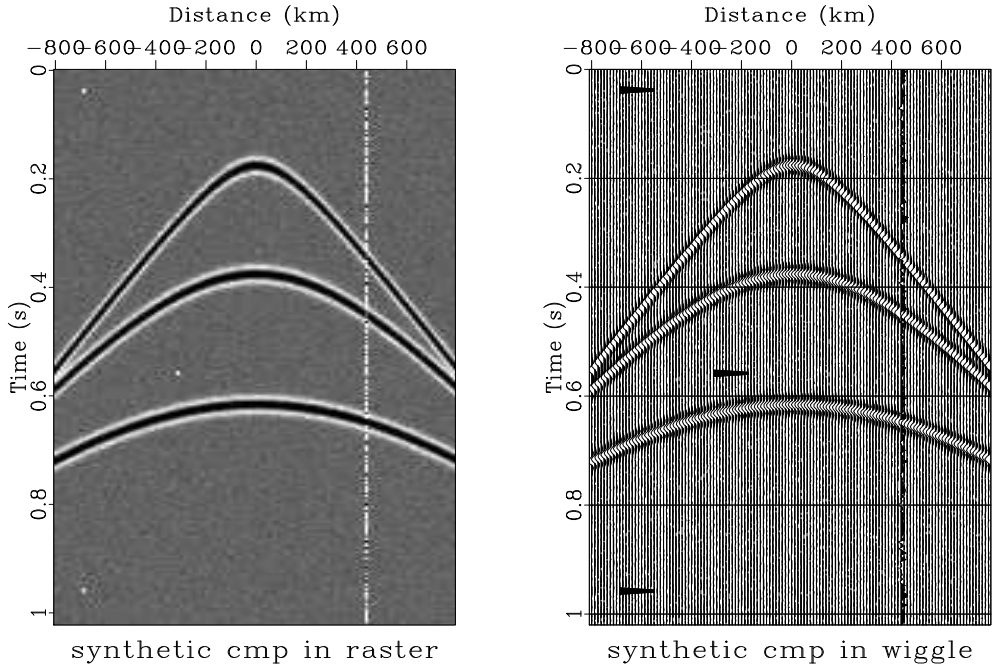

Figure 1: Synthetic data with various types of noise in raster format (left) and in wiggle format (right).

Figure 2 shows the inversion result (Figure 2b) obtained using the conventional CG algorithm for LS solution and remodeled data (Figure 2a) from it. In the CG method, the iteration was performed 30 times and the same number of iterations was also used for all the examples presented in this paper (including the number of iterations in the inner loop in the case of IRLS CG method). From Figure 2 we can clearly see the limit of ℓ^2-norm minimization. In the remodeled data, the noise with Gaussian statistics was removed quite well, but some spurious events were generated around the busty noise spikes and noisy trace. The inversion result obtained as a velocity-stack panel also shows many noisy values that correspond to the noise part of the data which was not removed completely.

Figures 3d through 3f show the inversion results obtained using the IRLS algorithm with ℓ^1-norm residual weight only, ℓ^1-norm model weight only, and ℓ^1-norm residual and model weights together, respectively. Figures 3a through 3c show the remodeled data from the corresponding inversion results. From the results of ℓ^1-norm residual weight (Figures 3a and 3d), we can see the robustness of ℓ^1-norm residual minimization for the busty noise and the successful removal of the Gaussian noise, too. From the results of ℓ^1-norm model weight (Figures 3b and 3e), we can see the improvement in the parsimony of the model compared to the result of LS inversion

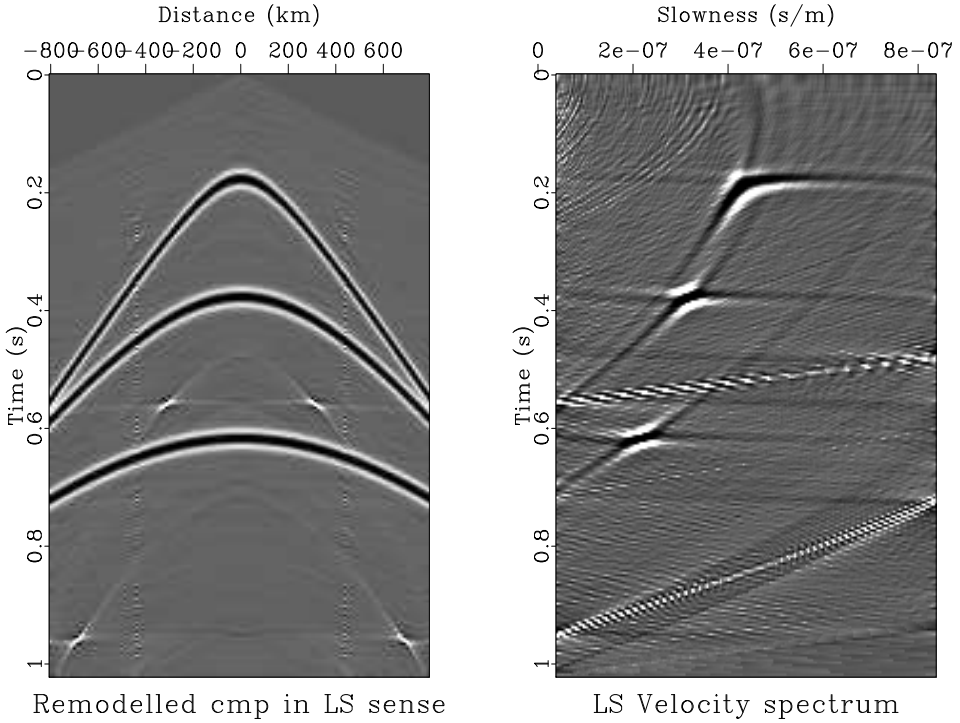

Figure 2: The remodeled synthetic data (a) from the velocity-stack (b) obtained by LS inversion using CG method for the noisy synthetic data (Figure 1).

(Figure 2b). The ℓ^1-norm model weight also seems to have some effect to reduce low amplitude noise quite well but the result shows some limit in reducing high amplitudes noises by making some spurious event around the busty spike noises (Figure 3b). From the results with ℓ^1-norm residual and model weights together (Figures 3c and 3f), we can see that the IRLS method can achieve both goals, the robustness to the busty noises and the parsimony of the model representation, quite well.

Figures 4d through 4f show the inversion results obtained using the CGG algorithm with the residual weight only, the model weight only, and the residual and the model weights together, respectively. Figures 4a through 4c show the remodeled data from the corresponding inversion results. From the results of the residual weight (Figures 4a and 4d), we can also see the robustness of the residual weight for the busty spike noises, and also the success in the removal of the Gaussian noise. Here the residual weight used was the same residual weight as the one used in the ℓ^1-norm residual minimizing IRLS method. Thus we can say that guiding the gradient using the ℓ^1-norm-like residual weight in the CGG method seems to behave as the ℓ^1-norm residual minimizing IRLS method does. From the results of model weight (Figures 4b and 4e), we can also see the improvement in the parsimony of the model estimation compared to the result of LS inversion (Figure 2b) and the similar behavior in reducing noises as the ℓ^1-norm model minimizing IRLS method does. For the model weight, I used $diag(\mathbf{W}_m)_i = |m_i|^{1.5}$, where the exponent 1.5 was decided empirically.

If we want the model weight as the one used in the ℓ^1-norm model weight in the IRLS method, the model weight $diag(\mathbf{W}_m)_i$ would be $|m_i|^{0.5}$, but the result of it was not as successful as the IRLS method. So the appropriate value for the exponent was decided to be 1.5 after experiments with several exponent values from 0.5 to 3. From the results of the residual and model weights together (Figures 4c and 4f), we can see that the CGG method also achieves both goals, the robustness to the busty noises and the parsimony of the model representation, quite well, and the results of the CGG method are comparable to the results of the IRLS method (Figures 3c and 3f).

Figure 5 shows the differences of the results of the IRLS method and of the CGG method from the original synthetic data, respectively. We can see that both differences contain nothing but the noise part of data. This demonstrates that both the IRLS method and the CGG methods are very successful in removing various types of noises. Therefore, we can say that the CGG inversion method can be used to achieve the same goal to make an inversion robust and to produce parsimonious model estimation as the IRLS method does. In addition, the CGG method requires less computation than the IRLS method by solving a linear inversion problem (which requires one iteration loop) instead of solving a nonlinear inversion problem (which requires two nested iteration loops).

Examples on real data

I tested the proposed CGG method on a real data set that contains various types of noise. The data set is a shot gather from a land survey. However the trajectories of the events in the data set look "hyperbolic" enough to be tested with a hyperbolic inversion.

Figure 6 shows the real data set used for testing and the results, the velocity-stack (Figure 6c) and the remodeled data (Figure 6b) from it, when we used the conventional LS inversion. We can see that the real data (Figure 6a) originally contains various types of noise such as the strong ground roll, the amplitude anomalies early at near-offset and late at 0.8km offset, and the time shifts around offsets 1.6 km and 2.0 km. The conventional LS inversion generally does a good job in removing most dominant noises except some whose characteristics are somehow busty (i.e. the amplitude anomalies early at near-offset and late at 0.8km offset and the time shift around offsets 1.6 km and 2.0 km). The resultant velocity-stack panel (Figure 6c) was filled with various noises that requires some more processing if we want to perform any velocity-stack oriented processing such as velocity picking, multiple removal, and so on.

Figure 7 shows the remodeled data from the inversion results (Figure 8) obtained using the IRLS method with three different weighting combinations (ℓ^1-norm residual only, ℓ^1-norm model only, and ℓ^1-norm residual and model together). We can see that all three remodeled results show quite successful removal of most noises similarly. The

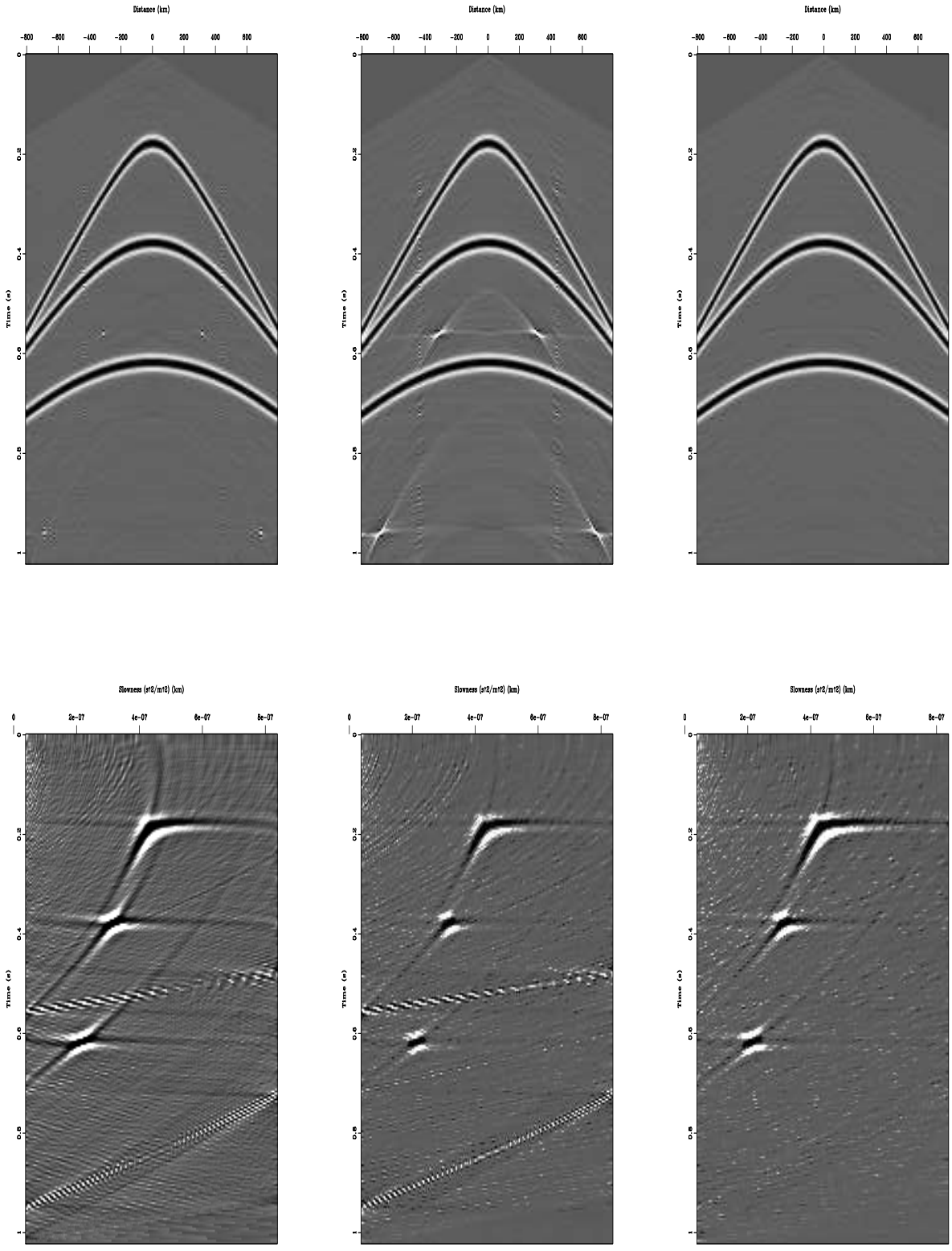

Figure 3: The remodeled data and the velocity-stack inversion results obtained by IRLS method with three different norm criteria : (a) and (d) are for the ℓ^1-norm residual weight only, (b) and (e) are for the ℓ^1-norm model weight only, and (c) and (f) are for the ℓ^1-norm residual/model weights together.

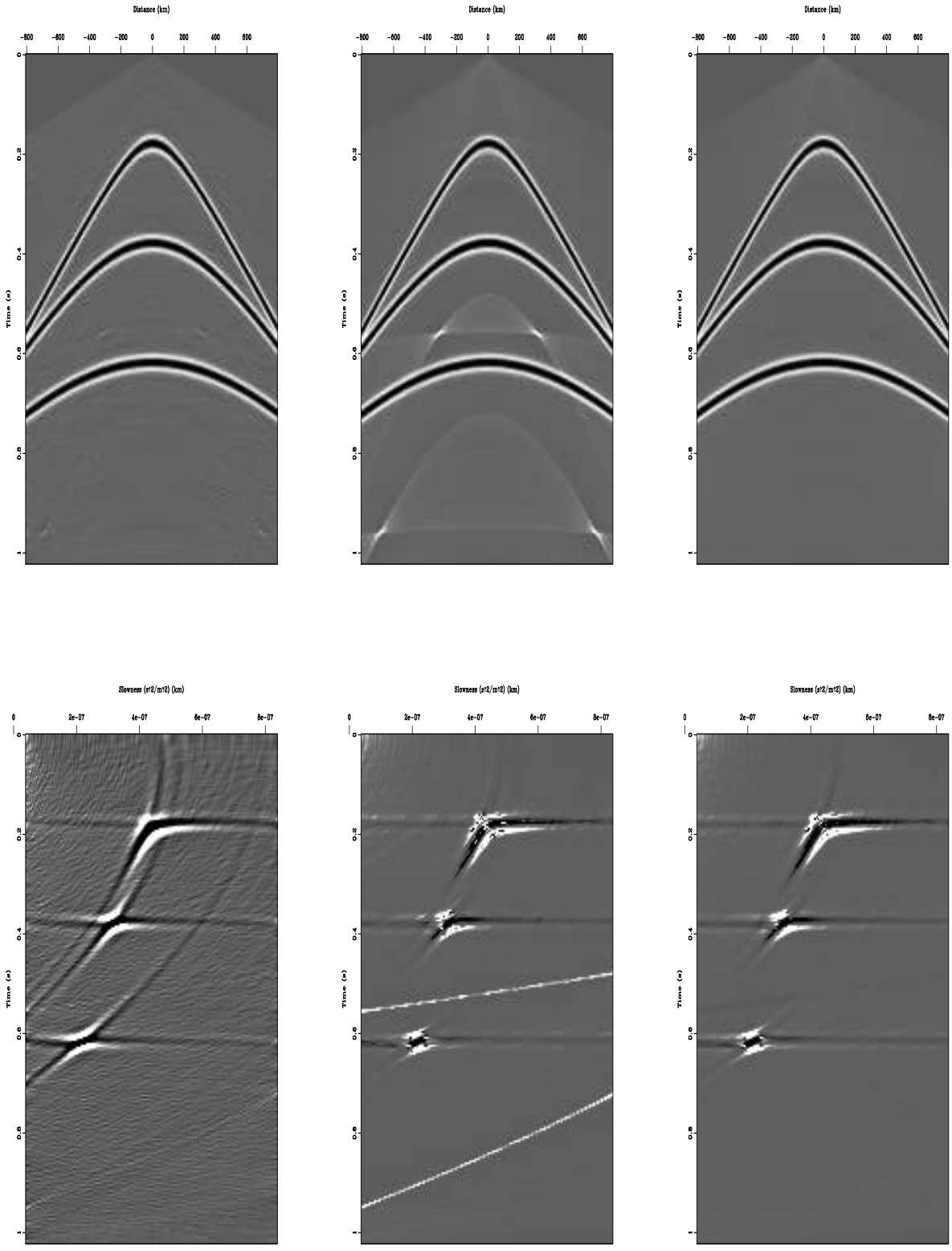

Figure 4: The remodeled data and the velocity-stack inversion results obtained by CGG method with three different guiding weights : (a) and (d) are for the residual weight only, (b) and (e) are for the model weight only, and (c) and (f) are for the residual/model weights together.

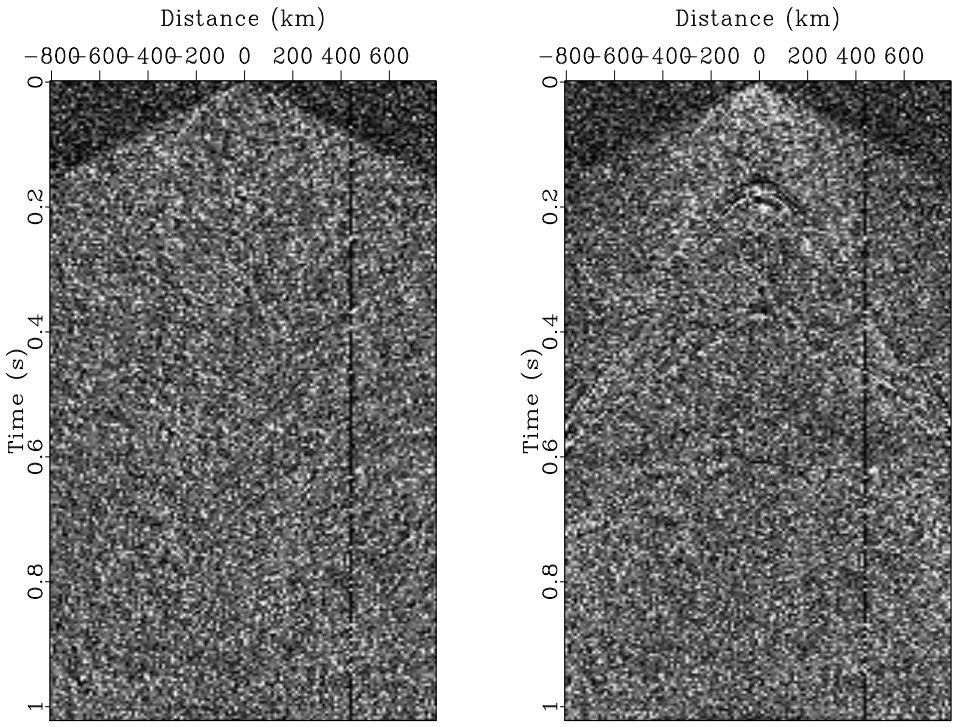

Figure 5: (a) The difference of the remodeled data obtained by IRLS method from the original synthetic data: the original noisy synthetic data (Figure 1) was subtracted from the remodeled data with IRLS method (Figure 3c), (b) The difference of the remodeled data obtained by CGG method from the original synthetic data: the original noisy synthetic data (Figure 1) was subtracted from the remodeled data with CGG method (Figure 4c).

main difference among the three inversion results is the degree of parsimony of the corresponding velocity-stacks as shown in Figure 8. Even though the approach of the ℓ^1-norm residual weight can reduce many noisy signals in the velocity-stack, the result of the ℓ^1-norm model weight shows better parsimony of the velocity-stack.

Figure 9 shows the remodeled data from the inversion results (Figure 10) obtained using the CGG algorithm with three different guiding types (the residual weight only, the model weight only, and the residual and the model weights together). All three remodeled data (Figures 9a through 9c) show quite similar quality as the ones obtained with IRLS method (Figure 7). The differences in the parsimony of the velocity-stacks among the different guiding types are also clearly shown in the Figure 10 and they are similar to the results of IRLS method. In the case of the guiding with the residual weight (Figure 9a) I used the weight, $diag(\mathbf{W}_r)_i = |r_i|^{-0.75}$ to achieve the similar quality in the noise removal as the one of the ℓ^1-norm residual minimizing IRLS method. The exponent value -0.75 is also decided empirically after experiments with various exponents values.

In the real data example above, the values of exponent of the weight functions in the CGG method were decided empirically and were sometimes different from the exponents of the weights used in the ℓ^1-norm IRLS method. In the IRLS approach, the meaning of the exponent for the weight functions can be explained either with the ℓ^p-norm sense or with the relative weighting for each values of the residual/model. Even though the two explanations are closely related, the meaning of the exponent for the weight function in the CGG method can be explained better with the latter since it only changes the gradient vector and doesn't minimize ℓ^p-norm. So if we increase the value of exponent of the model weight function, the relatively high amplitude model values get more emphasis in fitting the model to the data. Likewise, if we decrease the value of exponent of the residual weight function which is a negative value, the relatively high amplitude residual values get less emphasis in fitting the model to the data. The optimum value of exponent or relative emphasis in the model and the residual depends on the distribution of the values of the model/residual and could be found empirically as performed in this paper. The experiments performed in the paper demonstrate that the value of exponent of weight function used for ℓ^1-norm residual/model in the IRLS approach are good choices for a start, but could be increased or decreased appropriately for each case if any further improvement is required.

CONCLUSIONS

The proposed CGG (Conjugate Guided Gradient) inversion method is a modified CG (Conjugate Gradient) inversion method, which guides the gradient vector during the iteration and allows the user to impose various constraints for residual, model, or both of them. The guiding is implemented by weighting the residual vector and the gradient vector, either separately or together. Weighting the residual vector with the residual itself corresponds to guiding the solution search toward the ℓ^p-norm

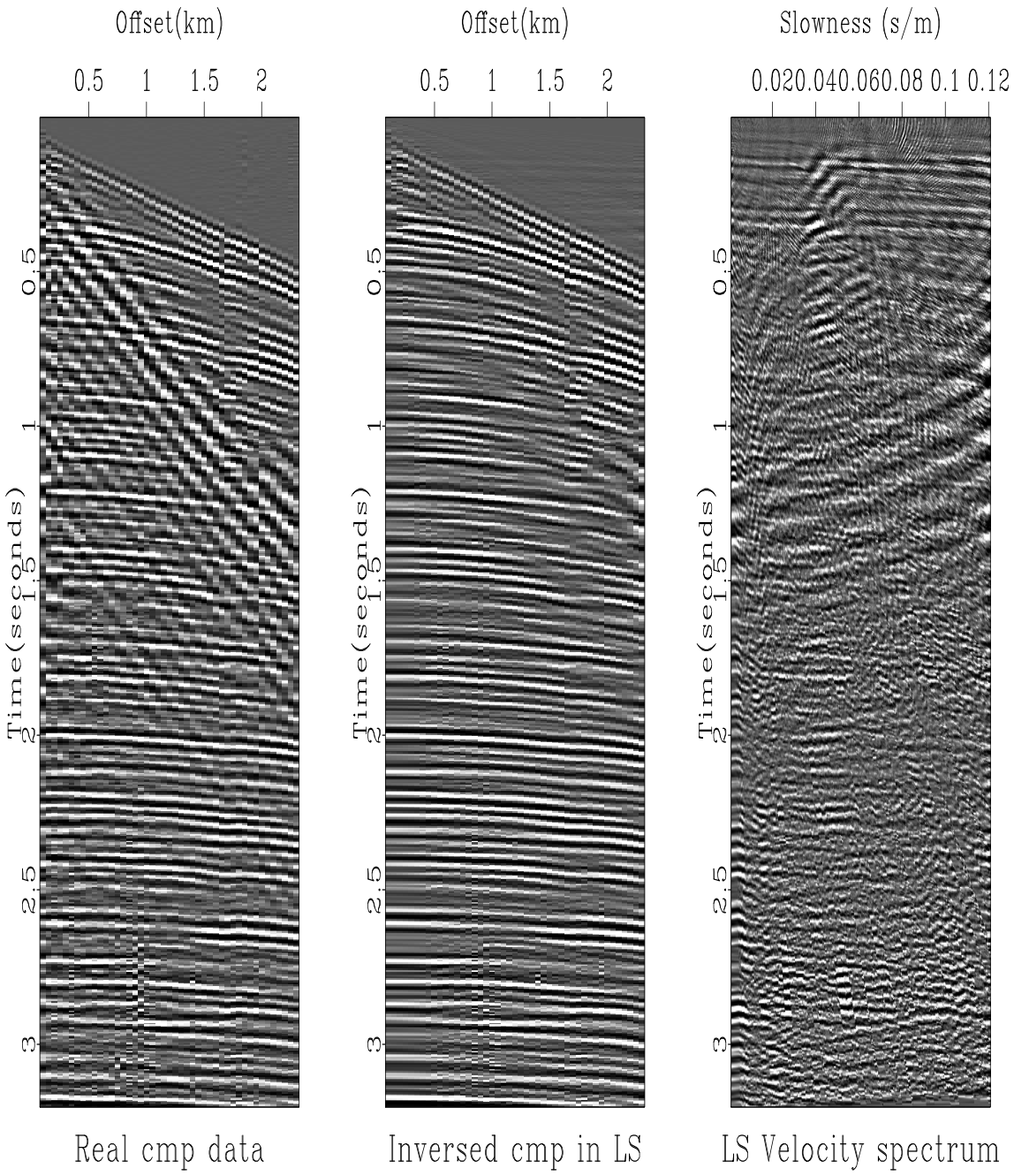

Figure 6: The real data set (a), the remodeled data from the inversion result (b), and the LS inversion result (c).

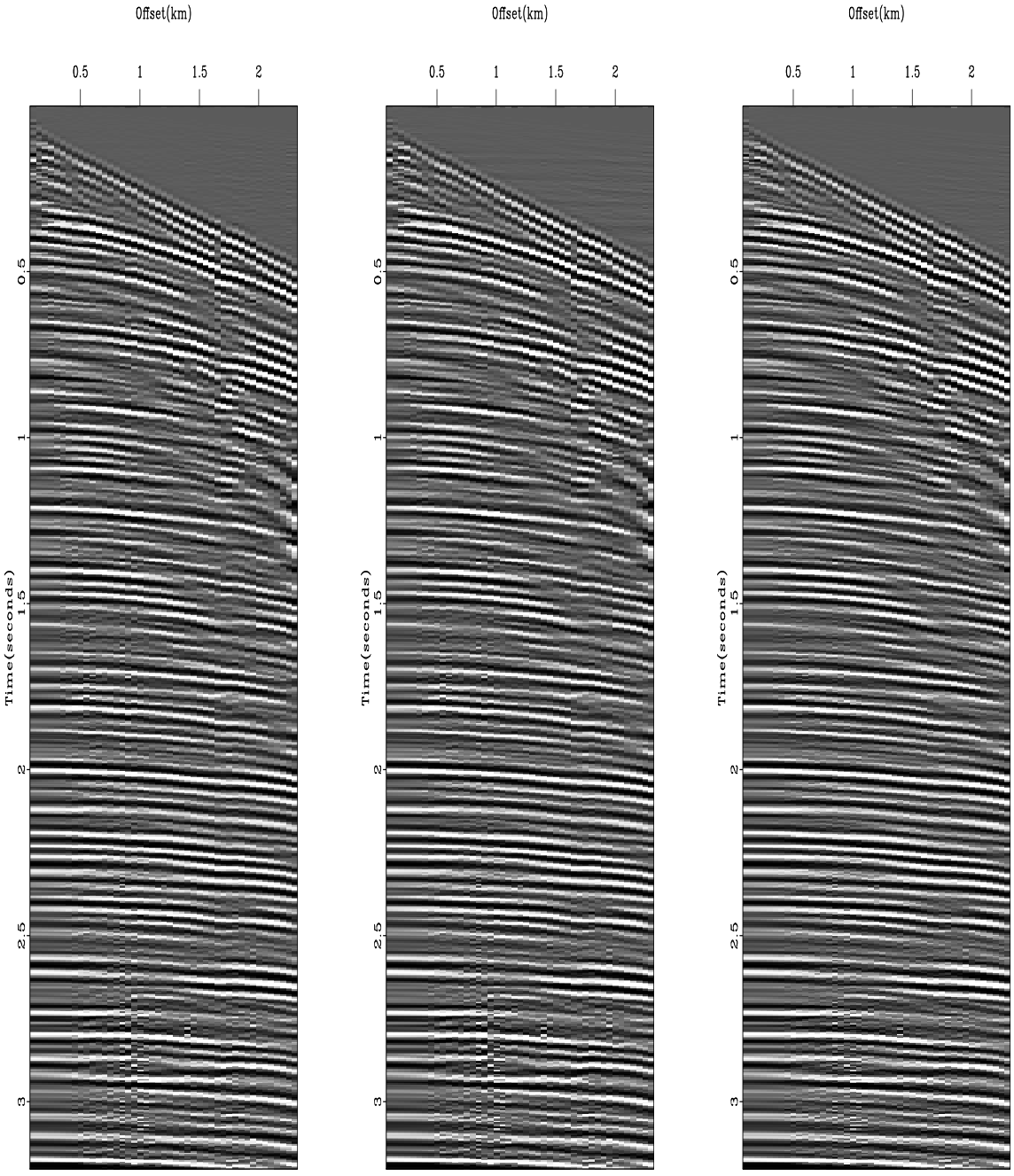

Figure 7: The remodeled data from the velocity-stack inversion results (Figure 8) of the real data (Figure 6a) using the IRLS method with different norm criteria: ℓ^1-norm residual only (a), ℓ^1-norm model only (b), and ℓ^1-norm residual/model together (c).

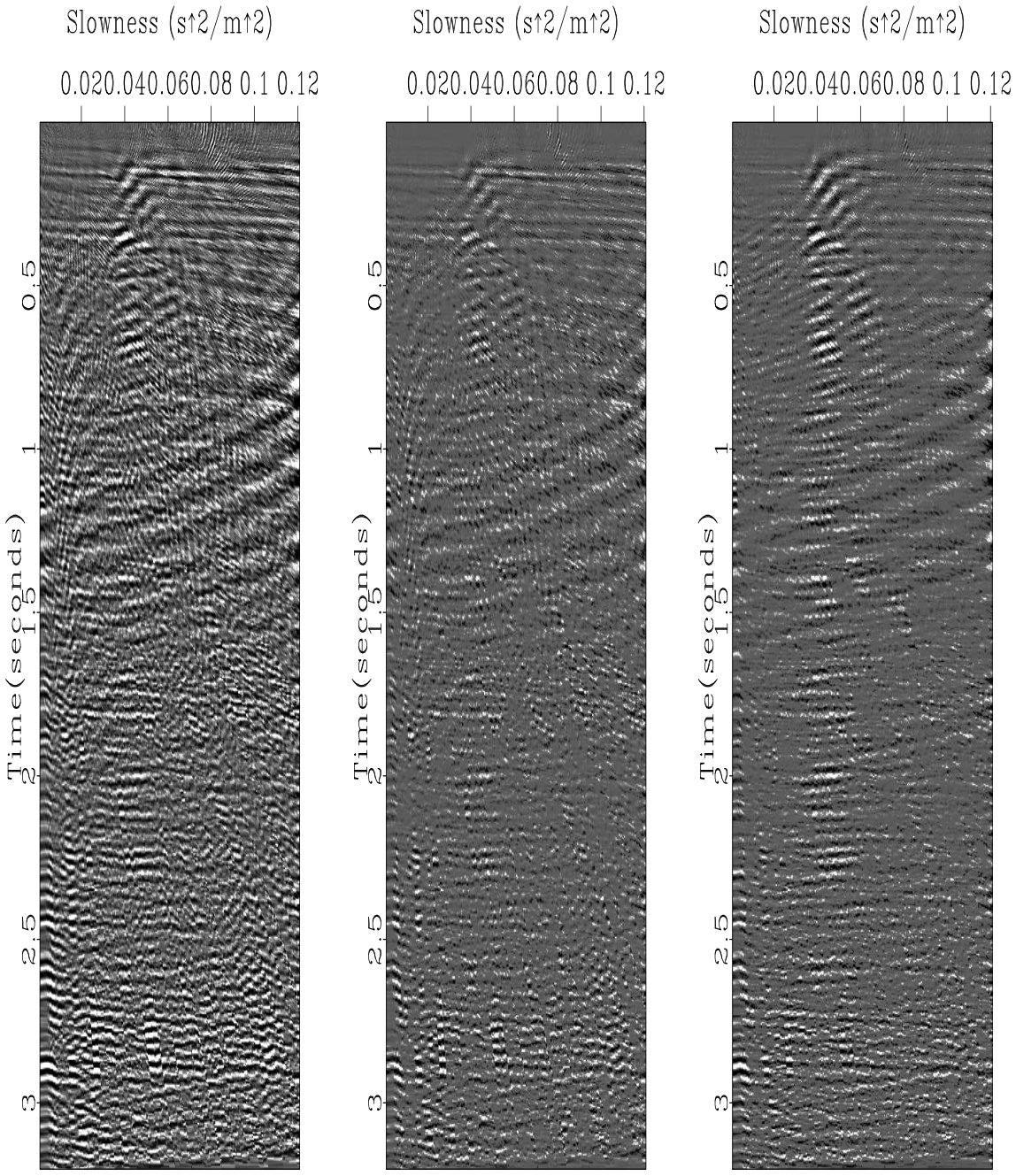

Figure 8: The velocity-stack inversion results of the real data (Figure 6a) using the IRLS method with different norm criteria: ℓ^1-norm residual only (a), ℓ^1-norm model only (b), and ℓ^1-norm residual/model together (c).

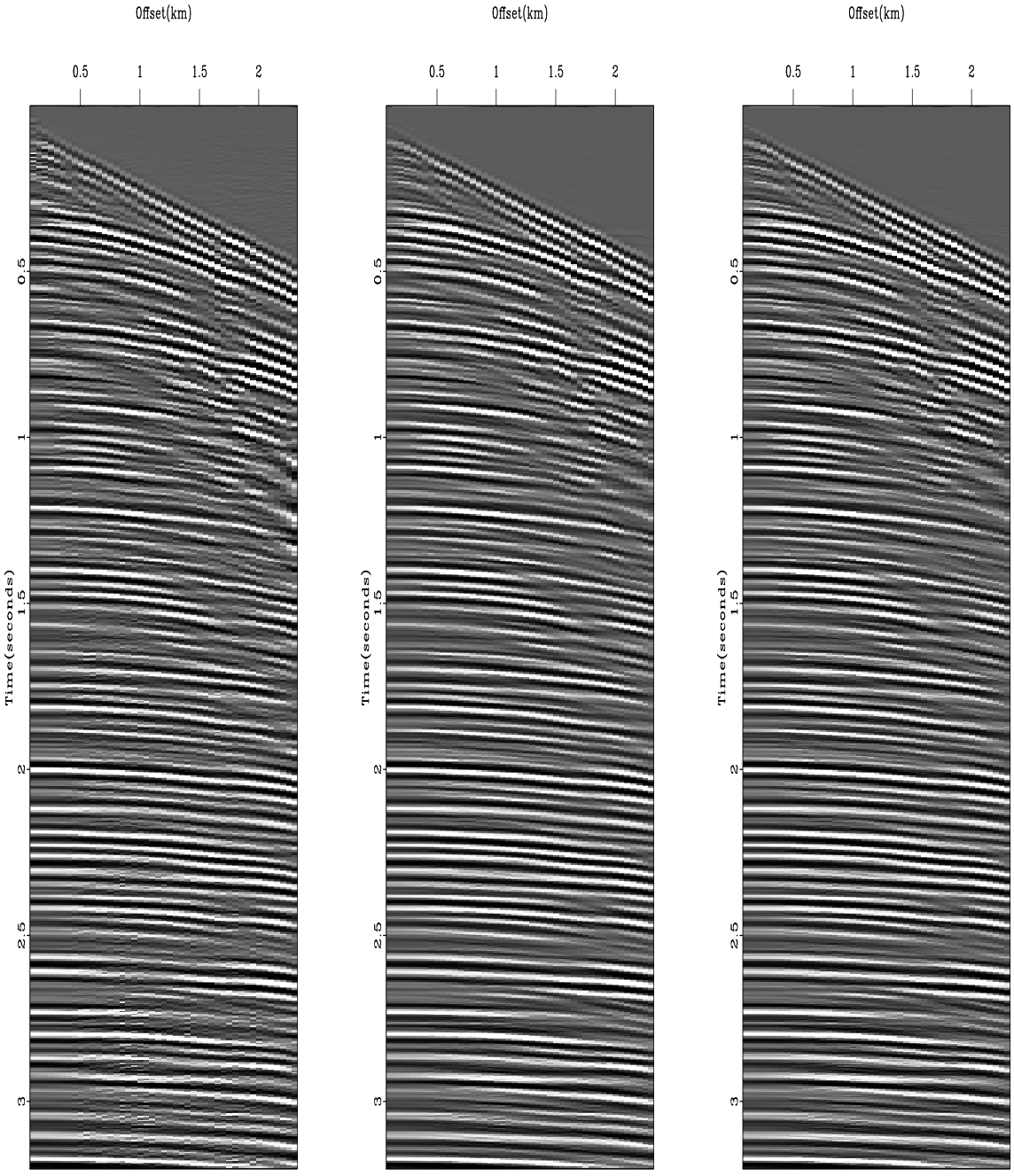

Figure 9: The remodeled data from the velocity-stack inversion results (Figure 10) of the real data (Figure 6a) using the CGG method with different guiding weights: the residual weight only (a), the model weight only (b), and the residual/model weights together (c).

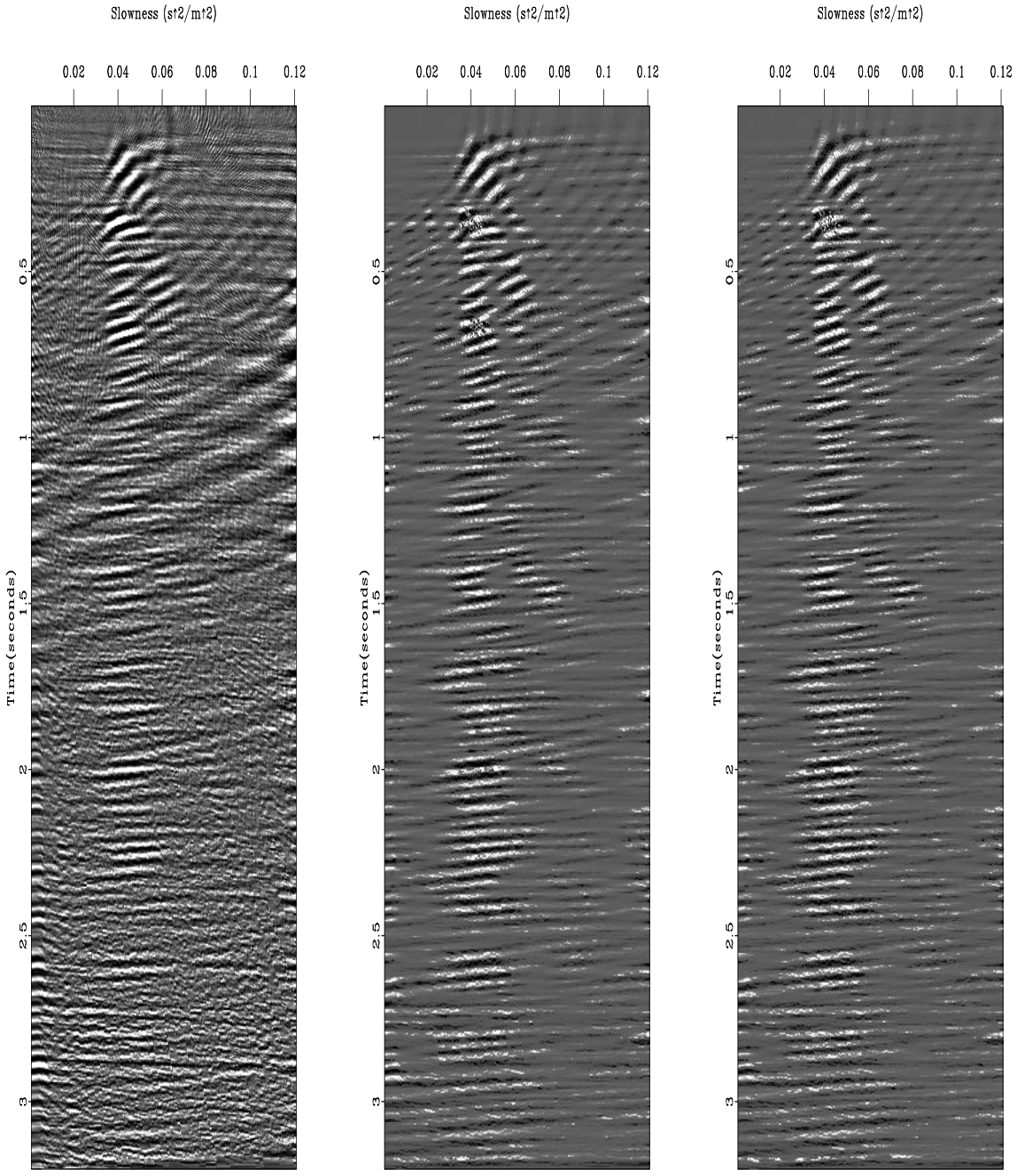

Figure 10: The velocity-stack inversion results of the real data (Figure 6a) using the CGG method with different guiding weights: the residual weight only (a), the model weight only (b), and the residual/model weights together (c).

minimization; weighting the gradient vector with the model itself corresponds to guiding the solution search toward *a priori* information imposed. Testing the CGG algorithm for the velocity-stack inversion of synthetic and real data demonstrates that the guiding with residual weighting gives a robust model estimation comparable to the IRLS method and the guiding with model weighting produces a parsimonious velocity spectrum also comparable to the IRLS method. So we can say that the CGG method can be use to achieve the same goals as the IRLS method does, but with less computation by solving the linear problem instead of solving nonlinear problem and more flexibility in choice of weighting parameters. Therefore, the CGG method seems to be a good alternative to the IRLS method for robust and parsimonious model estimation inversion of seismic data.

ACKNOWLEDGEMENT

This research was financially supported by Hansung University in the year of 2006. I thank two anonymous reviewers and the associate editor for their helpful and constructive comments.

REFERENCES

Broyden, C. G., 1969, A new double-rank minimization algorithm: Notices of the American Mathematical Society, **16**, 670.

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

Claerbout, J. F., 1992, Earth Soundings Analysis, Processing versus Inversion: Blackwell Scientific Publication.

———, 2004, Image Estimation by Example: `http://sepwww.stanford.edu/sep/prof/index.html`.

Claerbout, J. F., and F. Muir, 1973, Robust modeling with erratic data: Geophysics, **38**, 826–844.

Darche, G., 1989, Iterative l_1 deconvolution, *in* SEP-61: Stanford Exploration Project, 281–302.

Fletcher, R., 1970, A new approach to variable metric methods: The Computer Journal, **13**, 317–322.

Foster, D. J., and C. C. Mosher, 1992, Suppression of multiple reflections using the Radon transform: Geophysics, **57**, 386–395.

Gersztenkorn, A., J. B. Bednar, and L. R. Lines, 1986, Robust iterative inversion for the one-dimensional acoustic wave equation: Geophysics, **51**, 357–368.

Goldfarb, D., 1970, A family of variable metric methods derived by varia- tional means: Mathematics of Computation, **24**, 23–26.

Guiiton, A., and W. Symes, 2003, Robust inversion of seismic data using the Huber norm: Geophysics, **68**, 1310–1319.

Hampson, D., 1986, Inverse velocity stacking for multiple elimination: Journal of the Canadian Society of Exploration Geophysicists, **22**, 44–55.

Herrmann, P., T. Mojesky, M. Magesan, and P. Hugonnet, 2000, De-aliased, high-resolution Radon transforms: 70th Ann. Internat. Mtg, Soc. of Expl. Geophys., 1953–1956.

Huber, P. J., 1973, Robust regression: Asymptotics, conjectures, and Monte Carlo: Ann. Statist., **1**, 799–821.

Ji, J., 1994, Near-offset interpolation in wavefront synthesis imaging, *in* SEP-82: Stanford Exploration Project, 195–208.

Kabir, M. M. N., and K. J. Marfurt, 1999, Toward true amplitude multiple removal: The Leading Edge, **18**, 66–73.

Kostov, C., and D. Nichols, 1995, Moveout-discriminating adaptive subtraction of multiples: 65th Ann. Internat. Mtg, Soc. of Expl. Geophys., 1464–1467.

Lumley, D. E., D. Nichols, and T. Rekdal, 1995, Amplitude-preserved multiple suppression: 65th Ann. Internat. Mtg, Soc. of Expl. Geophys., 1460–1463.

Nichols, D., 1994, Velocity-stack inversion using $\mathbf{L_p}$ norms, *in* SEP-82: Stanford Exploration Project, 1–16.

Nocedal, J., 1980, Updating quasi-Newton matrices with limited storage: Mathematics of Computation, **35**, 339–353.

Sacchi, M. D., and T. J. Ulrych, 1995, High-resolution velocity gathers and offset space reconstruction: Geophysics, **60**, 1169–1177.

Scales, J. A., and A. Gersztenkorn, 1987, Robust methods in inverse theory, *in* Geophysical imaging, symposium of geophysical society of Tulsa: Soc. of Expl. Geophys., 25–50.

Scales, J. A., A. Gersztenkorn, S. Treitel, and L. R. Lines, 1988, Robust optimization methods in geophysical inverse theory: 58th Ann. Internat. Mtg, Soc. of Expl. Geophys., Session:S7.1.

Shanno, D. F., 1970, Conditioning of quasi-Newton methods for function minimization: Mathematics of Computation, **24**, 647–657.

Taner, M. T., and F. Koehler, 1969, Velocity spectra - Digital computer derivation and applications of velocity functions: Geophysics, **34**, 859–881. (Errata in GEO-36-4-0787).

Taylor, H. L., S. C. Banks, and J. F. McCoy, 1979, Deconvolution with the L-one norm: Geophysics, **44**, 39–52.

Thorson, J. R., and J. F. Claerbout, 1985, Velocity stack and slant stochastic inversion: Geophysics, **50**, 2727–2741.

Trad, D., T. Ulrych, and M. Sacchi, 2003, Latest views of the sparse Radon transform: Geophysics, **68**, 386–399.