Nuts and bolts of least squares Kirchhoff migration

Daniel Trad

ABSTRACT

Least squares migration (LSMIG) has been an important research topic in the academia for about two decades, but only recently it has attracted interest from the industry. The main reason is that from a practical point of view its ratio of benefit/cost has not been sufficient for its use in seismic exploration. Another problem is that these benefits are mixed with effects from the filtering techniques used to regularize the inversion, which are computationally much cheaper. In this report, I discuss some challenges with least squares Kirchhoff depth migration. This algorithm, although less precise than the more popular least squares reverse time migration, has the advantage of being fast enough to be applied in a production environment, and flexible enough to be applied without data regularization. This last characteristic makes it a good candidate to understand benefits in terms of footprint acquisition and aliasing. In addition, its limitations in terms of modelling/imaging accuracy make more evident some problems that exist but are often ignored when using reverse time migration with synthetic data. This work focuses on some implementation issues that are not often mentioned in other papers. First I discuss the effects of amplitude weights by testing on a flat synthetic reflector I compare spectra, amplitude preservation and convergence when using different weights and imaging condition. Second I apply LSMIG to the Marmousi data set, and compare the previously discussed effects. Also, I show the impact of traveltime tables on the convergence and the predictions. Finally, I explore some connections with seismic data interpolation, and show examples of data prediction using time and depth migration.

INTRODUCTION

Despite the existence of more accurate imaging algorithms, the most common migration algorithm for land seismic data is still Kirchhoff migration, mostly because of its flexibility to adapt to irregular sampling and its efficiency. Kirchhoff migration is basically a weighted mapping between data space (the original acquired data), and the image space (migrated section), where samples are weighted and shifted temporally and spatially. These weights come from the mathematics of integration and wave propagation, usually assuming infinite aperture and taking only partially into account the exact locations of shots and receivers. Hard work over many decades has led to weights that produce a good approximation to true amplitude reflectivity. However, these weights have limitations because they do not contain information about the sampling operator and acquisition illumination.

Least squares migration (LSMIG) is a different approach to migration where, rather than mapping and weighting to find the model, a modeling operator with the physics of wave propagation is inverted by optimization algorithms (Nemeth et al., 1999; Kuhl and Sacchi, 2003). The inverse of this operator applied to seismic data produces an image that by design can predict the acquired data in a least squares sense. Although the inverse of this modeling operator is closely related to the imaging (mapping) operator, it represents a different approach to the problem and it takes into account the actual geometry (sampling

and aperture) of the data. By applying this process instead of simple mapping, the physics of the process can be more closely matched because it contains more details of the seismic experiment than the standard migration mapping. By inverting a complete imaging operator that contains the acquisition pattern, we can in principle remove acquisition and illumination signature. But we will never be able to invert data from a defficient acquisition.

Probably the most attractive characteristic of least squares migration is that it falls into a wide range of algorithmic techniques used not only in seismic but in physics in general. The mathematics and physics of LSMIG allows one to include concepts like prior information, in the form of mathematical constraints that help to obtain better images. In practice, however, this is a very difficult process because of the high cost of applying the modeling/adjoint operator multiple times, and the challenging goal of creating an image that not only looks reasonable but is capable of predicting the data. Although stacking samples (mapping) is a very robust process, inverting for a multidimensional operator is usually an ill-conditioned problem.

From the practical point of view the advantages of LSMIG have not been shown in the industry except for specific examples, mostly 2D synthetic examples. Its application to 3D seismic processing is only starting to be tried as an alternative to traditional methods and it remains as an elusive application of academic research. This is particularly true for land data sets, where noise is complex and correct signal amplitudes are very difficult to predict correctly (Trad, 2015). However, many possible advantages over conventional migration keep motivating researchers to solve these difficulties. Among these possibilities we can mention (1) elimination of footprint noise caused by poor sampling and (2) better amplitudes in the image by compensation of illumination problems. In addition, the availability of the LSMIG engine provides a powerful and physical tool for interpolation, and de-noising.

The goal of this research is to understand some aspects of least squares Kirchhoff migration that are not often discussed in the geophysical literature. A 2D/3D least squares Kirchhoff time and depth migration is implemented following an Object-Oriented Programming (OOP) approach. Many variants are added into this code to examine how different weights and constraints affect the result, how much of the acquisition pattern can be removed by inverting the modeling operator, and the applications for creating data from a velocity/reflectivity model to interpolate data and reduce aliasing constraints.

LEAST SQUARES INVERSION AS A GENERAL FRAMEWORK

My LSMIG implementation uses a general philosophy widely used to solve for geophysical problems that originates from the Stanford Exploration Project and is nicely explained in Claerbout (1992). Given a transformation of some model $\mathbf{m}$ to some data $\mathbf{d}$ through some mathematical operator $\mathbf{L}$,

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

we can approximately solve the inverse problem of recovering the model that produced the data by inverting the operator in a least squares sense. The meaning of the different components of equation (1) depends on the details of the operator and are often associated

with well-known transformations. For the case of LSMIG, $\mathbf{L}$ is a synthesis or modeling operator, that generates data from reflectivity using Green functions as basis functions. For the case of Fourier Transform $\mathbf{L}$ is a synthesis of the data in terms of plane waves, and for Radon transform is a synthesis of the data in terms of arbitrary curves. Although these three examples can be thought of as physical representations of seismic data in some domains, strictly speaking they are all pure mathematical constructions. We could say that the Kirchhoff modeling operator is more physical than the other two because it resembles the physical process of wave propagation. However, any transformation could be used to generate the same data by this procedure, independently of whether it is physical or not. What is different and requires some consideration is the meaning we assign to the model. Even for very similar operators, small variations can change what the model represents. For example, by changing the weights in the operator we can switch the model from reflectivity (Kirchhoff) to velocity perturbations (Born).

This framework is interesting because it broadens the possible applications for LSMIG. For example, using Kirchhoff time migration for structured data may not make sense for estimating reflectivity but it can be useful for interpolation or noise attenuation. We just need to be careful that, if the operator does not contain the proper migration weights, or if we include nonphysical constraints, the predicted data will be still fine, but the model will not correspond to the reflectivity. This is in fact one reason why LSMIG is more complicated than interpolation or noise attenuation. In those cases, we are only interested in the data space so we have complete freedom to modify the operator at will. By contrast, in LSMIG we are interested in the model itself, therefore we are constrained by physics. As an example, for the case of interpolation by Fourier transforms we use a non-physical mapping because the model we are inverting for is just a temporary space which is not used for any physical interpretation. For LSMIG, on the other hand, the transformation is constrained by the physics of wave propagation, at least if we want to be able to use the model for reflectivity.

Once the meaning of the model is defined by the operator chosen, we can estimate the model by inverting the operator. To invert operators, we define a cost or objective function, which is a mathematical expression that measures the undesired characteristics of the model. The most common is to find a model that honors the data in a least error sense, measured on some norm, and has a minimum of information not required by the data. This statement of goals is commonly presented as

$$\begin{aligned} &\text{minimize } \|\mathbf{W}_m\mathbf{m}\|_p^p \\ &\text{subject to } \|\mathbf{W}_d(\mathbf{d}-\mathbf{L}\mathbf{m})\|_q^q = \phi_d \end{aligned} \tag{2}$$

where ϕ_d is some estimate of the noise level in the data plus a residual due to the failure of the proposed model to explain the data. p and q indicate that different norms can be applied to measure the norm of vectors. $\mathbf{W}_d$ could be a matrix or vector of data weights, often a diagonal matrix containing the inverse of the standard deviation of the data, but more generally it could be any kind of filter that leaves out bad data. If the LSMIG were going to be used for interpolation for example, $\mathbf{W}_d$ should nullify any information from unrecorded traces, otherwise the model is forced to predict zeroes on those traces (honoring the empty traces). $\mathbf{W}_m$ is an operator of model weights that can be customized to enhance

our preferences regarding the model. For example, $\mathbf{W}_m$ might be a gradient (roughening) operator, then minimizing $\|\mathbf{W}_m\mathbf{m}\|$ will produce a smooth model.

For LSMIG we usually need to favor, **but not enforce**, smooth models that lead to continuous structures. A typical choice for Kirchhoff type of migrations, is to use $\mathbf{W}_m$ to enforce smooth changes of reflectivity with offset or angle. This can be done as in Nemeth et al. (1999) by taking differences between consecutive reflectivities at different offsets, as in Kuhl and Sacchi (2003) by a sparse tau p transform of the reflectivity, or as in Moghaddam and Herrmann (2005) by a sparse curvelet transform of the reflectivity. A generic choice for $\mathbf{W}_m$ is the inverse of a multidimensional operator that takes a multi-offset migrated volume and performs smoothing, for example a triangular filter, a dip filter, a fxy filter or a Radon transform. In Fomel (2007), this formulation is generalized with the name of shaping filter. Notice that this type of constraint forces us to compute many common offset migrations simultaneously, opposite to common practice in industry of migrated common offset volumes independently.

After these choices, by minimizing the cost function obtained from equations (2) we can obtain a model that better approximates the desired solution. Thus, the LSMIG model can be found by solving the system of equations

$$(\lambda\mathbf{W}_m^T\mathbf{W}_m + \mathbf{L}^T\mathbf{W}_d^T\mathbf{W}_d\mathbf{L})\mathbf{m} = \mathbf{L}^T\mathbf{W}_d^T\mathbf{W}_d\mathbf{d}, \tag{3}$$

where λ is a trade-off parameter that will allow a different weight to be assigned to the misfit and model constraints. To eliminate this parameter and simplify this system of equations, we incorporate these weights and filters into a general operator $\tilde{\mathbf{L}} = \mathbf{W_d}\mathbf{L}\mathbf{W_m^{-1}}$, which is just a change of variables in the fundamental equations, with $\tilde{\mathbf{m}} = \mathbf{W}_m\mathbf{m}$ and $\tilde{\mathbf{d}} = \mathbf{W}_d\mathbf{d}$. This is equivalent to a right and left preconditioning, transforming the modeling equation (1) to

$$\mathbf{W}_d\mathbf{d} = \mathbf{W}_d\mathbf{L}\mathbf{W}_m^{-1}\mathbf{W}_m\mathbf{m} \tag{4}$$

and the system of equations (3) to

$$(\lambda\mathbf{I} + \tilde{\mathbf{L}}^T\tilde{\mathbf{L}})\tilde{\mathbf{m}} = \tilde{\mathbf{L}}^T\tilde{\mathbf{d}}\,. \tag{5}$$

The system (5) can then be solved by setting $\lambda = 0$ (no regularization) and limiting the number of iterations. A great simplification of all these transformations is that now the operator contains all our choices and the numerical solver is completely generic. This solver is usually iterative, like a conjugate gradient method (CG).

IMPLEMENTATION

Parallel Object-Oriented approach to optimization

Least squares migrations of different types have many pieces in common. For example, least squares time and depth migrations use identical computer codes except for the calculation of traveltimes. On the other hand, the framework that is required to efficiently implement a parallel 3D LSMIG of any kind is significantly larger than the migration algorithm itself. To take maximum advantage of the similarities, and minimize future development overhead, the LSMIG module implemented for this work is designed with an

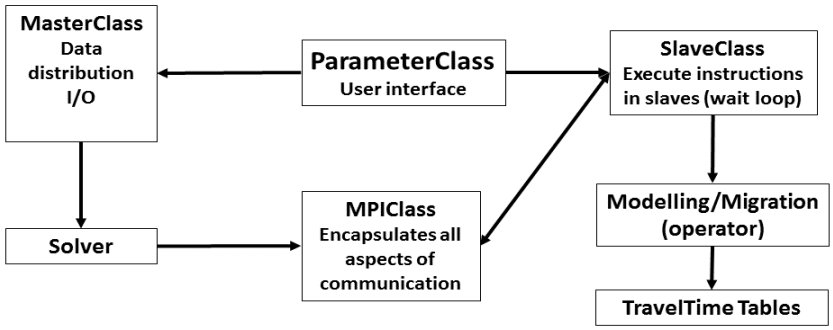

FIG. 1. Parallel Object Oriented implementation for Least Squares Migration. Different migration types can be implemented by changing only the operator.

Object Oriented Programming (OOP) approach to optimization. All operations related to the particular migration algorithm are encapsulated into a migration class. Similarly, all operations related to optimization are encapsulated into a solver class. Operations in the "master" and "slaves" are driven by a master and slaves classes. With this design, different least squares migration can be implemented by keeping most of the code intact, just redefining the operator class.

Data are distributed across nodes, but every node contains a complete copy of the model. This makes sense because the data are usually much larger than the model. Each node migrates and models (operators $\tilde{\mathbf{L}}^T$ and $\tilde{\mathbf{L}}$) a subset of the data using all available threads, but the inversion involves all the distributed data simultaneously. At each conjugate gradient (CG) iteration, model information across nodes must be transferred to the master, where the CG algorithm actually takes place, by using Message Passing Interface (MPI). Also, in each iteration the new model must be broadcast from master to slaves. This scheme permits to accommodate parallelization through OPENMPI and OPENMP simultaneously, and it allows the distribution of the data across nodes regardless of issues like migration aperture. The MPI calls are all encapsulated in a MPI class, and they are designed in such a way that new algorithms can be tested by modifying only the master. Basically, the algorithm looks like a serial CG solver with matrices or vector calls replaced by calls to members of the MPI class. This class contains methods to apply all the common solver operations, like forward, adjoint, residuals and dot-products. To facilitate the synchronization between master and slave calls, each node is continuously waiting for instructions. Once an instruction is received and executed the node goes back to waiting mode. This design allows one to program the algorithm as if it were executed only on the master and avoid common problems when master/slave steps go out of sync. Finally, using C++ Standard Template Library containers, the dimensionality of the model can change (2D, 3D, and 4D=3D+offset) without changing the code.

Weighting

In principle, we might expect that least squares inversion of the modeling operator will compensate for some of the numerical weights normally used in migrations, since these weights are designed to make the migration operator approximate the inverse of $\mathbf{L}$. However, removing these weights produces suboptimal results because LSMIG inversion is never performed completely due to the high cost of the forward and adjoint operators (never iterate to convergence). Results are just a partial inversion, so numerical weights similar to the ones used during migration are important.

In LS Kirchhoff migration, it is not trivial to achieve the optimal weighting scheme. At first we could choose to use the best possible weighted migration operator, design its adjoint such that it passes the adjoint test (Claerbout, 1992), and use these two operators into a CG algorithm. Weights in migration algorithms often take the form of deconvolution (DC) weights, which come from the imaging principle (Claerbout, 1971) as a ratio of scattered (u_s) over incident (u_i) fields as:

$$\mathbf{m} \approx \frac{u_s}{u_i} \tag{6}$$

This leads to migration weights that are proportional to a ratio of shot and receiver wavefield amplitudes, for example for shot migration (Docherty, 1991)

$$a_{migration} \approx \frac{A_r}{A_s} \tag{7}$$

or, after approximating for constant velocity which allows us to replace amplitudes by inverses of traveltimes (Dellinger et al., 1999; Zhang et al., 2000)

$$a_{migration} \approx \frac{t_s}{t_r} \tag{8}$$

Nemeth et al. (1999) sets the operator $\mathbf{L}$ as a modelling rather than a migration (Casasanta, personal communication). This approach changes the imaging condition from deconvolution (migration) to cross-correlation (modelling). In the cross-correlation imaging condition (XC) the LS weights are proportional to the product of amplitudes, which leads, after using similar approximations, to a product of traveltimes:

$$a_{lsmig} \approx \frac{1}{t_s\, t_r} \tag{9}$$

Testing with simple flat synthetics supports this interpretation. Figure 2a-b illustrates that deconvolution weights produce the correct amplitudes for migration (a), but as we iterate during the inversion, the amplitudes get distorted (b). Figure 2c-d show that cross-correlation weights produce the wrong image for migration (c), but the amplitudes become correct after a few iterations (d).

Another related issue, is whether the phase filter operator should be included inside the modelling/migration operator so it would act for every iteration, or should be only pre-applied as a data preconditioner, or not applied at all. The exact formulation for the phase

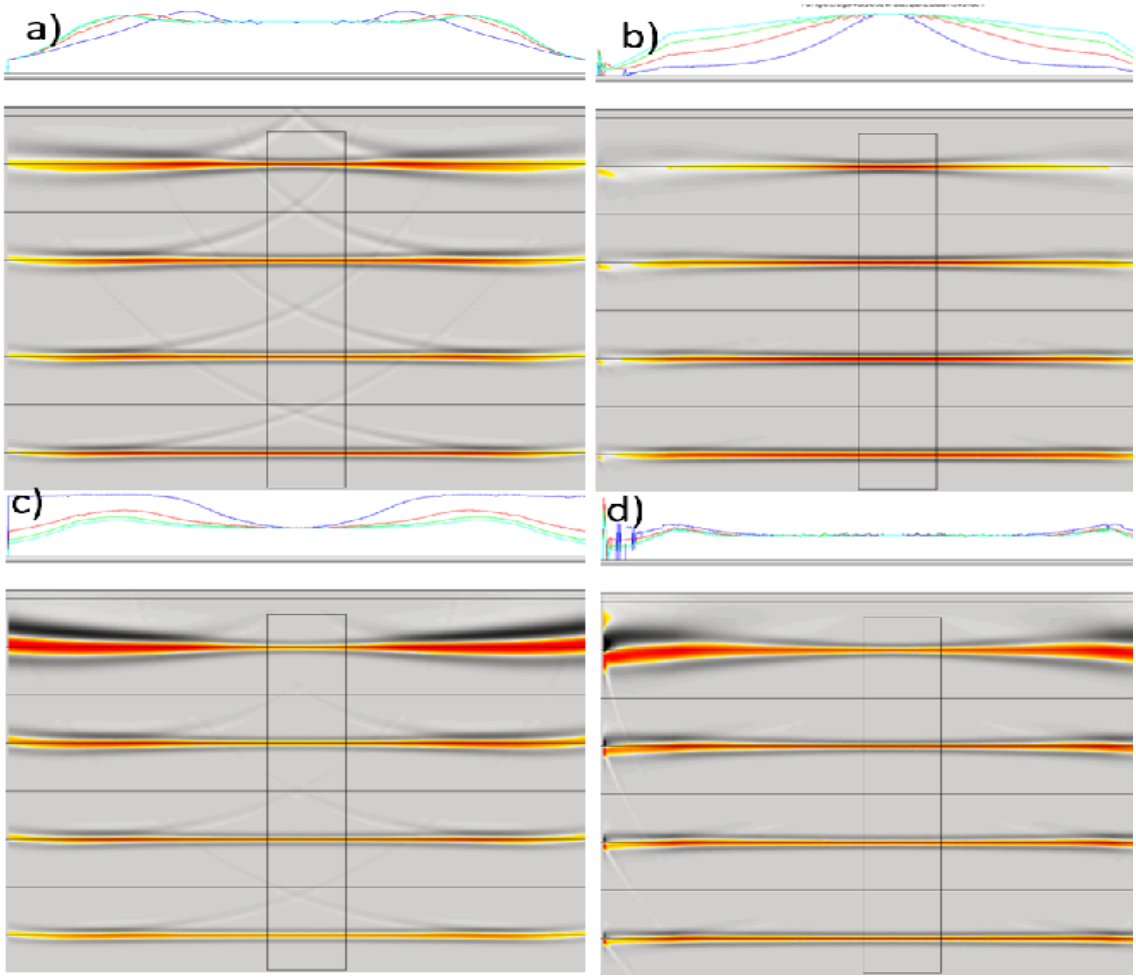

FIG. 2. Comparison of amplitudes for deconvolution weights (above) vs cross-correlation weights (below), and their evolution with iterations: migration (left) vs lsmig (right)

shift operator depends on the dimensionality of the data acquisition, being equal to $i\omega$ for 3D, $\sqrt{|\omega|}$ for 2.5D or $|\omega|$ for 2D (equations 2 in Bleistein and Gray (2001)). In this paper, I refer to this filter simply as ω filter. Figure 3 shows the convergence for flat reflectors (2D) in four different scenarios: deconvolution weights with ω filter pre-applied, with ω filter inside operator, cross-correlation weights with pre-applied filter and cross-correlation weights with filter inside operator. Certainly, including the ω inside the operator seems to help with the convergence. But clearer evidence comes from comparing the amplitude spectra (Figure 4). These tests show that if the ω filter is not included inside the operator, then the spectrum of the image shift towards higher frequencies with iterations. Another factor to consider with data weights is whether to include or not velocity weights. In theory amplitude weights contain a power of velocity (v^{-2} for 2D, v^{-3} for 3D). Although in migration this factor can be replaced by a depth gain after migration, in LSMIG it has a different effect because by including the velocities inside the weights the convergence is focused to areas with higher velocity. This is one of many situations where simplifications taken in migration do not carry over to LSMIG.

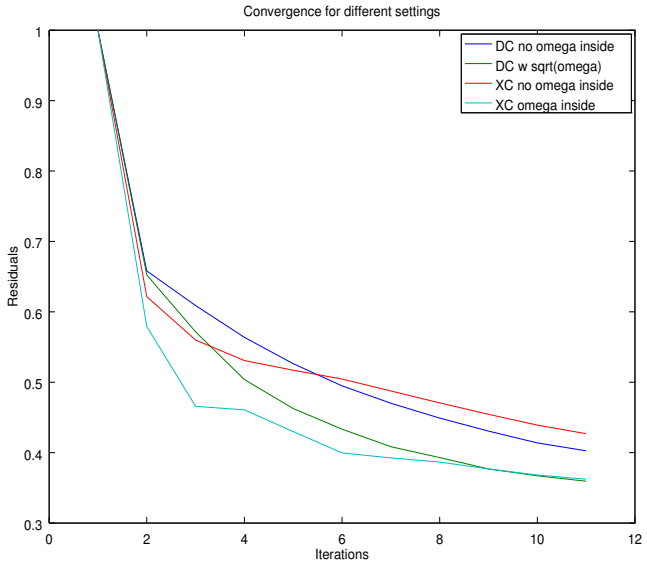

FIG. 3. Comparison of convergence for LSMIG using deconvolution weights (DC) vs cross-correlation weights (XC) with/without ω filter.

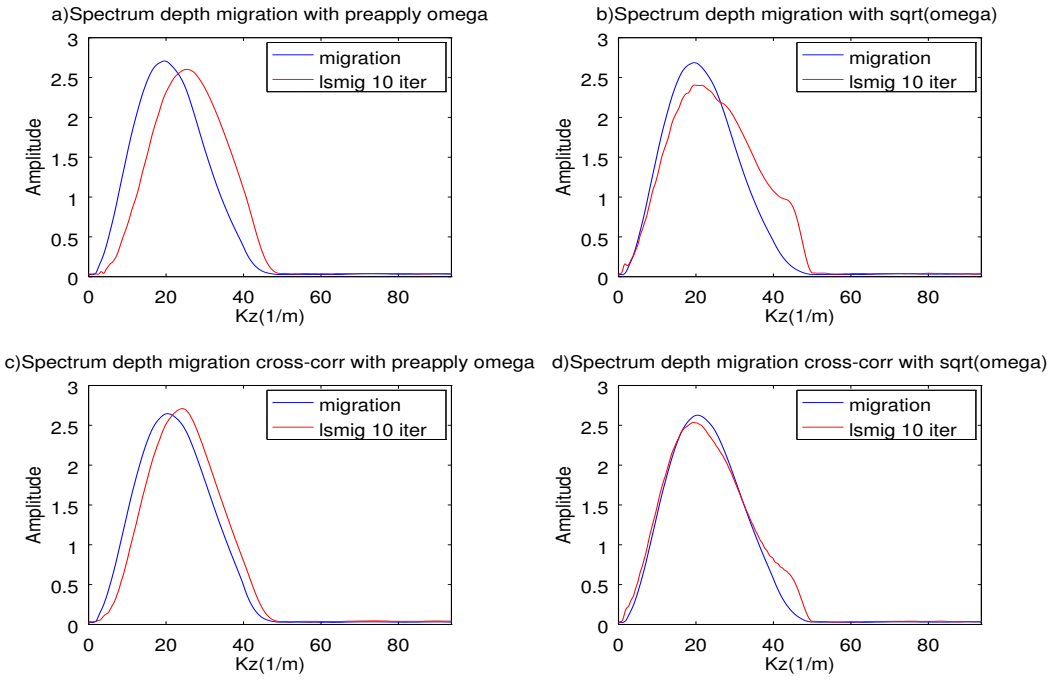

FIG. 4. Comparison of spectrum for migration vs LSMIG using deconvolution weights (DC) vs cross-correlation weights (XC), with/without ω filter.

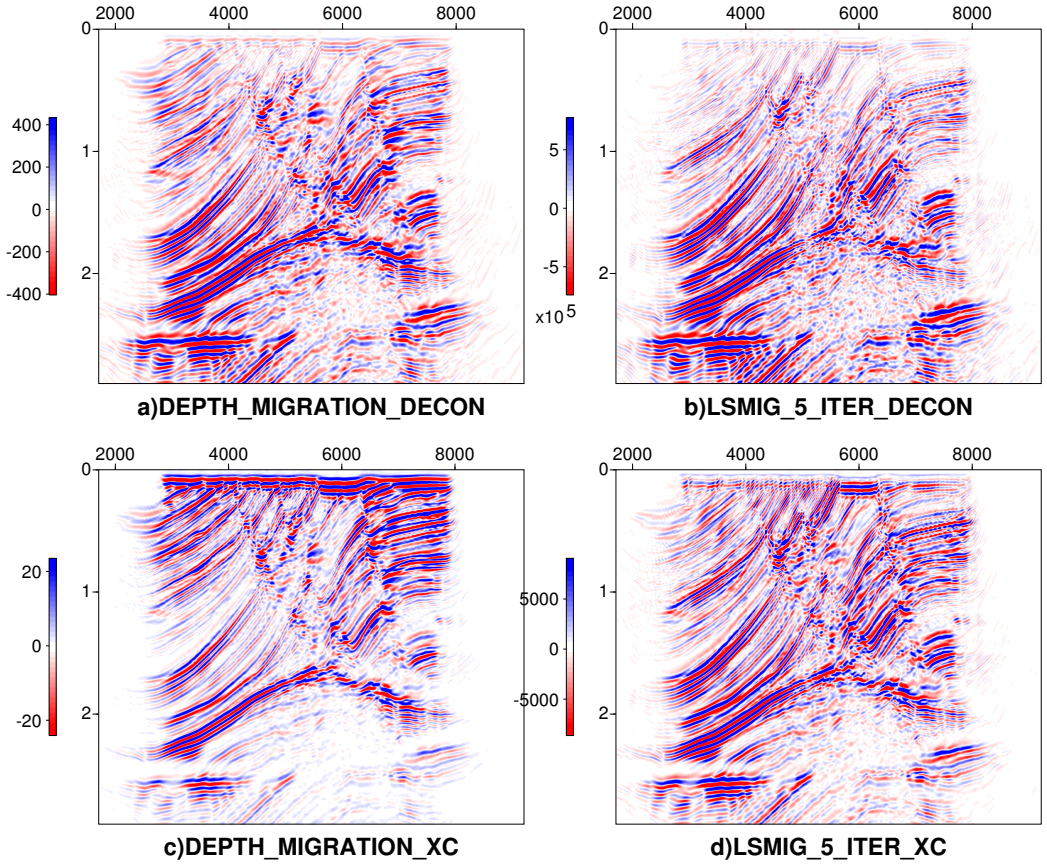

FIG. 5. Comparison of migration and LSMIG with deconvolution and cross-correlation weights

The choices stated above are not intended to be definitive answers to the question of which amplitude weights are required for LSMIG, but instead a declaration that such question exists, and this technology cannot evolve unless this question is properly answered. Conclusions achieved in this simple flat reflector test may not stand on structured real data tests. In practice, these effects are more complex in real data with noise, or even in structured synthetics. In general, the cross-correlation weights make the effect on LS Kirchhoff migration less noticeable, but preserves the reflectivity spectrum better with iterations. As we saw for the flat data examples, the deconvolution weights approximate the results of inversion in one iteration, but tend to distort the amplitudes with iterations. On the other hand, with the cross-correlation weights, the image is farther from the inversion result in the first iteration but improves with iterations. However, because the number of iterations is insufficient to completely solve for the inversion, the LS image from XC weights maybe quite similar or sometimes worse than the image from DC weights. In Figure 5 we see a comparison for Marmousi using deconvolution (above) and cross-correlation weights (below). Both results show some increase on resolution but also of noise with iterations. It is not easy to see which of the two weights produce a better image, although shallow amplitudes with XC condition seem more uniform.

Noise control

One of the most important problems in LSMIG is the accumulation of noise with iterations. This noise comes from broken contributions to the model as the algorithm tries to fit the predictions to small details on the data that are not properly mapped by the migration operator. This effect seems to be highly accentuated with discontinuities or gaps in Kirchhoff depth migration traveltime tables, as shown in a later section, because the operator becomes a poorer representation of the physics. For this reason, LSMIG noise is more obvious with a Kirchhoff migration than with reverse time migration (RTM). In RTM, the noise often disappears if the data are created by the same numerical approximation used in the migration engine. On the other hand, RTM may not show this problem if tested with finite difference data, but the noise may appear in real data processing as the numerical approximation deviate from the actual physics that produced the data.

As usual in ill-conditioned inversions, this problem requires some form of noise control by means of numerical regularization. Regularization in my implementation can be performed by a multidimensional triangular filter (the $\mathbf{W}_m$ operator in equations 2). The filter acts on the inline, crossline and offset dimensions. The length of the filter, which is independent across dimensions, permits one to control the noise effectively, but it is limited by the structural complexity of the image. In the example of Figure 5 the applied filter is very mild to preserve structure (length along inline was 5, length along offset was 5). Increasing the length would make the result cleaner, but some faults would be smoothed. In addition to triangular filter I have tried four other operators, dip filtering, fx filtering and linear and parabolic Radon transforms. All these are more computationally intensive and did not perform better in my tests than the simple triangle filter. However, having a directional smoothing filter (Fomel, 2007) certainly would allow one to increase the length of the filter to follow reflections, but also could smooth faults and enforce a pre-defined structure into the image.

Anti-alias filtering

Another way to address numerical noise introduced by iterations is by using an anti-alias filter. In the example in Figure (5) the anti-alias filter (AAF) was turned off completely. Figure (6) shows the same results with AAF at 100%. Because this synthetic data set has a good sampling (as would normally be the case for 2D data), the effect of AAF is minimal. If we apply LSMIG to more complex data sets in realistic 3D acquisitions the AAF becomes very important. However, one of the expected benefits of LSMIG is to make the AAF less necessary. In practice, it is unclear whether removing the AAF completely is a good idea. For one side, it controls aliasing noise, but from the other data fitting at larger offsets requires energy eliminated by an AAF. Therefore, when the AAF is applied, iterations slowly tend to fill back the filtered energy. What is less clear is if, by applying controlled regularized data fitting, this energy can contribute to the model without aliasing noise.

Although migration AAF implementation is well known, some special care is required when implementing the forward/adjoint pairs required in LSMIG. If these pairs do not pass the adjoint test (Claerbout, 1992), then the step sizes calculated during any conjugate

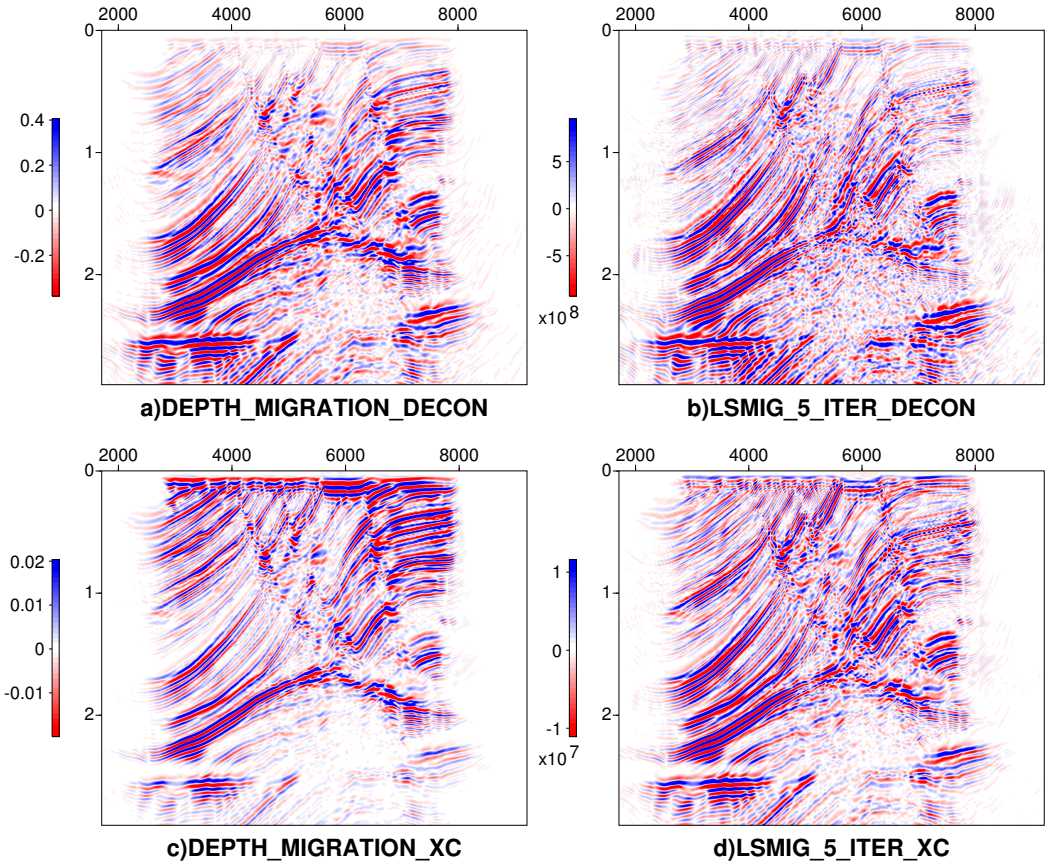

FIG. 6. Comparison of migration and LSMIG with deconvolution and cross-correlation weights with anti-alias filtering included

gradient algorithm will be wrong, and there will not be convergence. Because of the large size of data and model spaces for migration, even a small error in the adjoint test will produce order of magnitude errors in the step size. Following Lumley et al. (1994) we need the following steps when performing migration:

- resample data to fine sampling (usually 1 ms for efficiency)
- apply phase filter + double integration to input ($\times\omega^{-2}$)
- calculate and apply triangular filter
  - calculate mapping and amplitude weights from traveltime tables ($t_{shot}+t_{receiver}$)
  - calculate AAF length (which produces three different traveltimes)
  - add the three data samples to model with corresponding weights $(-1, 2, -1)$

When performing modeling, the order of operations change to:

- calculate and apply triangular filter
  - calculate mapping and amplitude weights from traveltime tables ($t_{shot}+t_{receiver}$)
  - calculate AAF length (which produces three different traveltimes)
  - spread value from model to three different data samples with corresponding weights $(-1, 2, -1)$
- apply conjugate phase filter + double integration to output ($\times\omega^{-2}$)
- resample data to original sampling rate

Convergence

A critical aspect for a successful application of LSMIG in an industrial setting is to improve its convergence. Figure (7) shows how convergence improves when the phase filter is applied, and it agrees with similar tests for the flat reflectors (Figure 3). Another factor that has a large effect on convergence is the dimensionality of the model. When the model is defined in terms of offset gathers instead of their summation, convergence improves dramatically (Figure 8). This is understandable because of less mixing, which makes prediction easier, and therefore convergence faster. Using offset gathers has the effect of subdividing the problem into smaller sub-inversions. Similarly, convergence for a part of the data is much faster than for the whole data. The trade-off, however, is that by using part of the data inversion is only partially accomplished, even if data fitting is better.

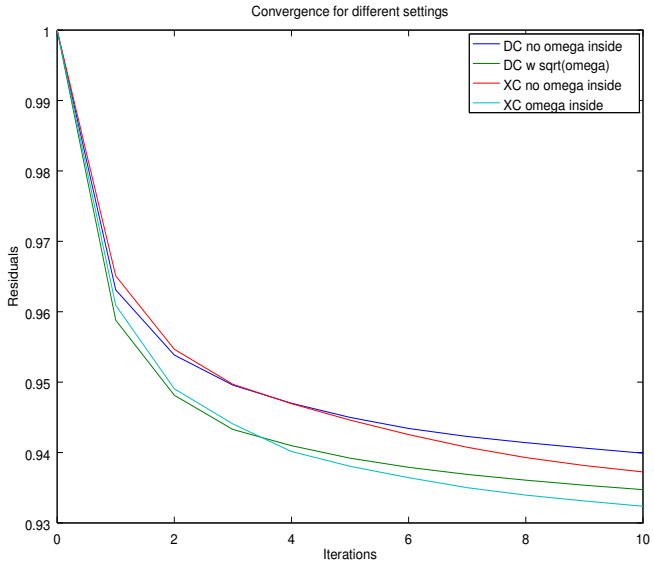

FIG. 7. Comparison of convergence for LSMIG using deconvolution weights (DC) vs cross-correlation weights (XC) for the Marmousi tests, and when the phase filter is included inside the operator or not.

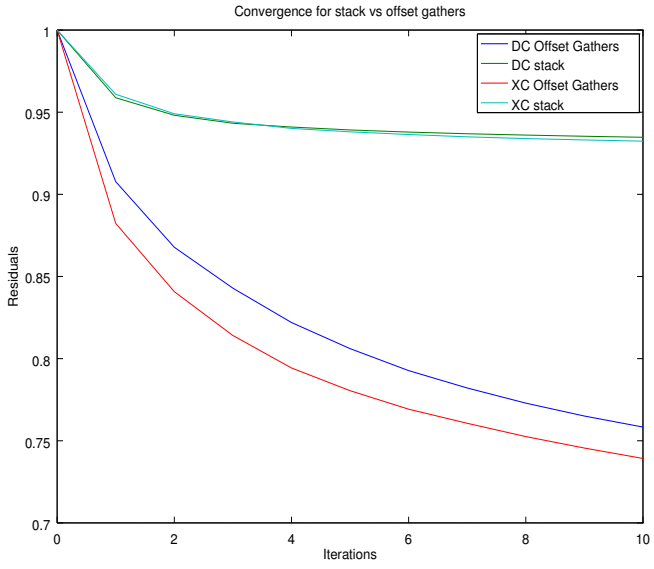

FIG. 8. Comparison of convergence for LSMIG offset gathers vs stack, in Marmousi tests.

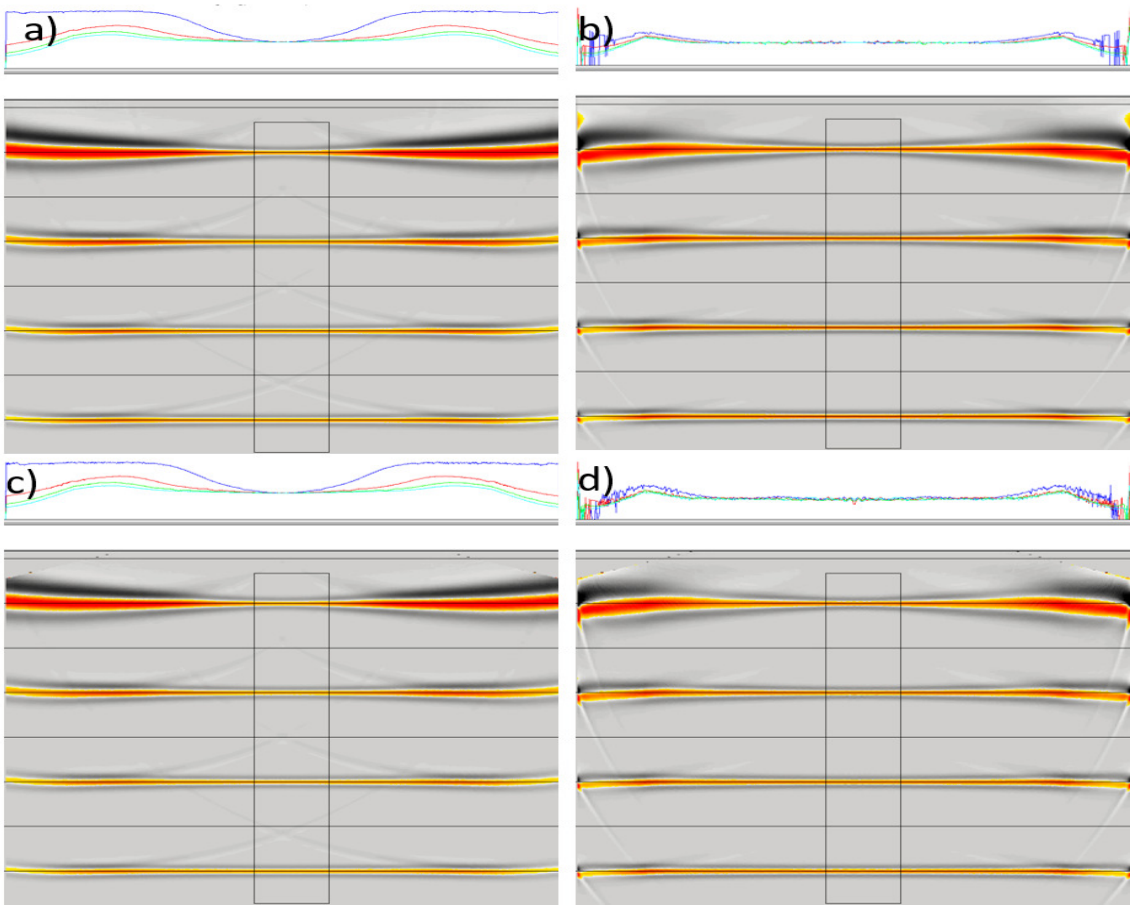

FIG. 9. Comparison of amplitudes for cross-correlation weights (above) for analytical time functions vs interpolated travel-time tables and its evolution with iterations: migration (left) vs lsmig (right)

Effects of traveltime table problems with iterations

LSMIG can be, in some respects, more robust than simple migration, but in others much less robust. This is particularly true when considering problems caused by a poor approximation of physics by the migration operator. A clear example is the effect of travel time table imperfections on the final image. Figure 9 shows, for flat reflectors, migration results (left) vs LSMIG results (right). The differences between (a-b) and (c-d) is that the figures in the top were calculated using analytical calculations for traveltimes, and the figures in the bottom were calculated using actual ray tracing (which includes interpolation for traveltime tables). We see that although migration results are nearly identical, the LSMIG results show some increase in noise when numerical traveltime tables are used. The effects we see are subtle for simple flat reflectors, where traveltime tables are almost perfect, but the problem becomes much worse with data complexity. Figure 10 shows this effect for LSMIG for the Sigsbee2a data set. The area on the left, where ray tracing is relatively simpler because of stratigraphy, shows much less noise than the area on the bottom right, where the effect of traveltime errors due to the salt are starting to be more apparent. This effect is produced by the iterative use of traveltime derivatives, and illustrate why LSMIG is much harder than migration for one side, and why LSMIG in depth is harder than LSMIG in time.

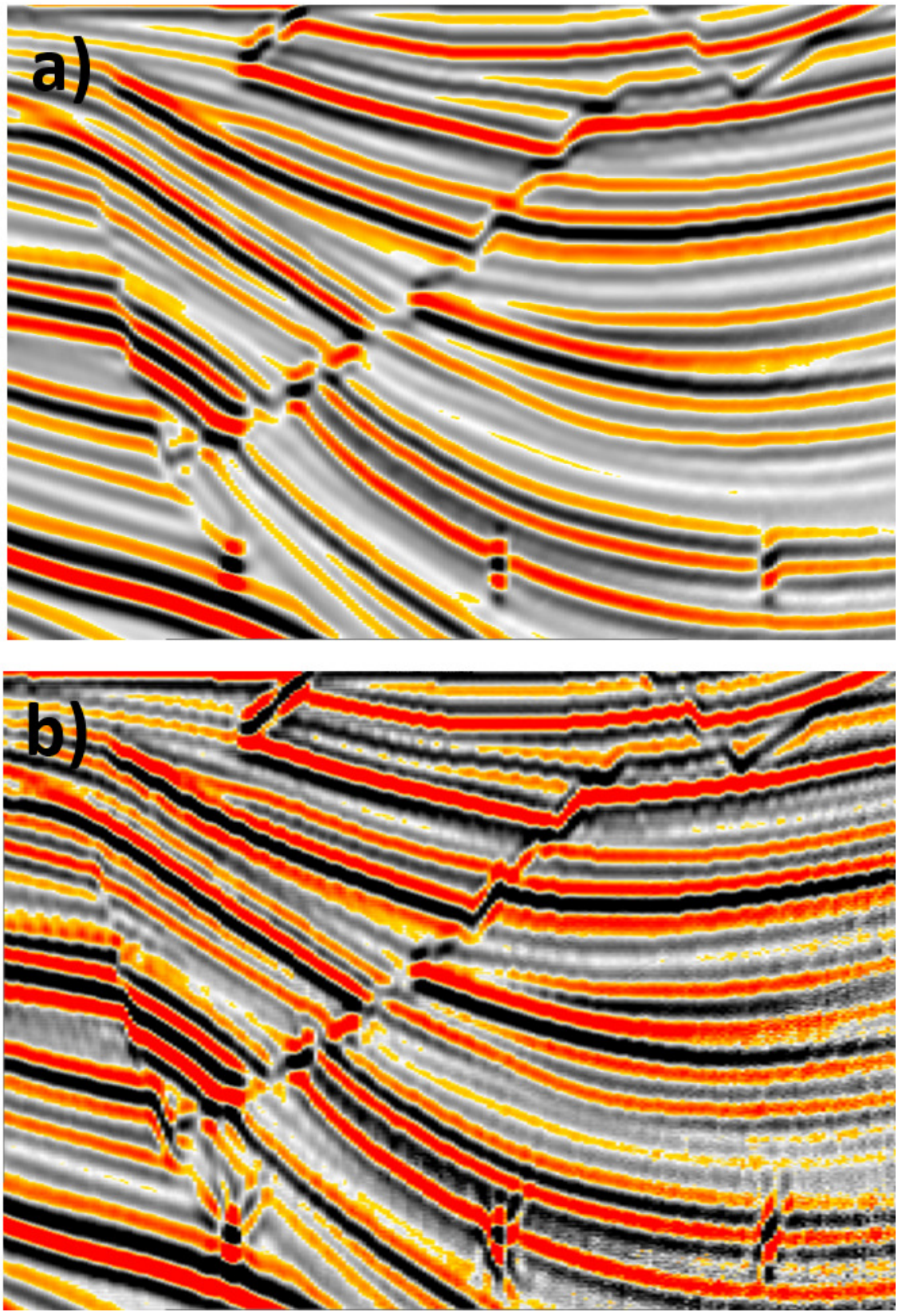

FIG. 10. Effect of complexities of traveltime tables for LSMIG for the Sigsbee model: migration (above) vs LS migration (below)

Modelling/Migration as an interpolator operator

LSMIG is a decomposition of seismic data in terms of Green's functions of the earth, just like a multidimensional Fourier transform is a decomposition of the data in terms of plane waves. Any such decomposition could be used to predict missing data. The advantage of using Green's functions instead of plane waves is that the Green's functions incorporate our knowledge about velocities and the physics of wave propagation. Even more, they can include information obtained from wells or other geophysical methods.

On the other hand, it can be argued that interpolation is a pre-conditioner for migration, so once migration is done there is no need for interpolation. However, there are several reasons why we may want to examine and/or use LSMIG predictions of missing and existing data. An obvious use of these predictions is for quality control of our algorithm. If unrecorded data are predicted with right amplitudes and phase, that means that the inversion has succeeded in removing the acquisition signature from the data. Similarly, predictions of existing or missing data inform us how well the operator approximates the physics of seismic wave propagation, and more importantly, how good our image is. Parts of the model that the operator is unable to see will be absent from predictions, indicating that probably our image is also missing some features that are responsible for part of the data. This is important information that will increase our confidence on the image we have obtained, and the validity of our algorithm, although does not necessarily ensures that the image is right.

Most of the LSMIG results in the geophysical literature focus on examining how well migration artifacts are removed from the final image. This is valuable in practice, but it may be a serious pitfall to justify the heavy computational burden of LSMIG by this effect, if this can also be obtained by much cheaper algorithms like filtering. Most of the acquisition footprint attenuation observed in these tests may well be due to filters applied inside the operator, which could do the same for the final image if applied after a simple migration. Therefore it is critical to understand whether the benefits observed in these tests are related to filtering and illumination compensation in elements along the diagonal of the Hessian matrix $\mathbf{L}^T\mathbf{L}$, or to elimination of acquisition footprint that lies on the off-diagonal elements of the Hessian. If LSMIG can predict data in unrecorded locations that would prove that the inversion has succeeded and done something that could not be obtained by simple post-imaging processing.

A difficulty often observed with LSMIG is that changes in the image tend to be subtle, even after many iterations. On the other hand, predictions change quickly. This is because by definition iterative LSMIG is designed to improve predictions, not the image. From physics, we can infer that if predictions get better, the image should also become better. However, the amount of improvement that we can observe in the image, from a given amount of improvement in the predictions, is very hard to infer. It most likely depends on the complexity of the seismic experiment. The mapping between data and model space follows a complex pattern and makes LSMIG more difficult, from a practical point of view, than other decomposition methods with simpler mapping, like Fourier transforms.

As an example let us examine the evolution of predictions and image for LSMIG for the

Marmousi model, focusing on the most difficult part of the structure close to location 5000 (central part). In Figure 11 we see the image, predictions and residuals after 1 iteration. Predictions are missing much detail, containing only the stronger features of the input shots. On the other hand, the image seems to have most of the final features, except some weaker horizons where illumination is poor, and other features that cannot be mapped because their information is missing in the traveltime tables, for example because of multi-pathing or poor ray tracing. In Figure 12 we see the same example after 5 iterations. Residuals have decreased and predictions have improved noticeable, but changes in the image are very subtle.

Another question is to what degree the quality of predictions depends on the quality of the operator approximation for the physics of the process. This question has very practical meaning in cases like interpolation, where the goal resides on the data space instead of the model space. Much effort is required to obtain better modeling operators, both in the implementation and in the computer effort during calculation. This effort brings clear benefits when the goal is to obtain a better image, but it is unclear if the same is true when the goal is to predict the data. For example, Fourier transform has proven very useful in interpolating data, even when the multidimensional Fourier spectrum has only an approximated physical meaning. To perform this test, I implemented a time migration version of LSMIG and compare the results for different iterations. Figure 13 shows the equivalent for Figure 11 but changing the operator to time instead of depth. Despite the image suffering serious deficiencies since the RMS assumptions for time migration are broken in this model, we see that the predicted data match the recorded data well, at least in part. In fact, some predicted events seem to match the data better than the prediction from depth migration. This could be explained by the advantage of time migration of having smoothing varying traveltime values, defined everywhere, different from depth where traveltime tables suffer from some discontinuities due to the complexity of the ray tracing across the complex structure. Similarly, Figure 14 shows the equivalent for Figure 12 after 5 iterations. We can conclude that time migration benefits from smoothness of the operator, which brings less numerical noise in the model with the progress of the iterations.

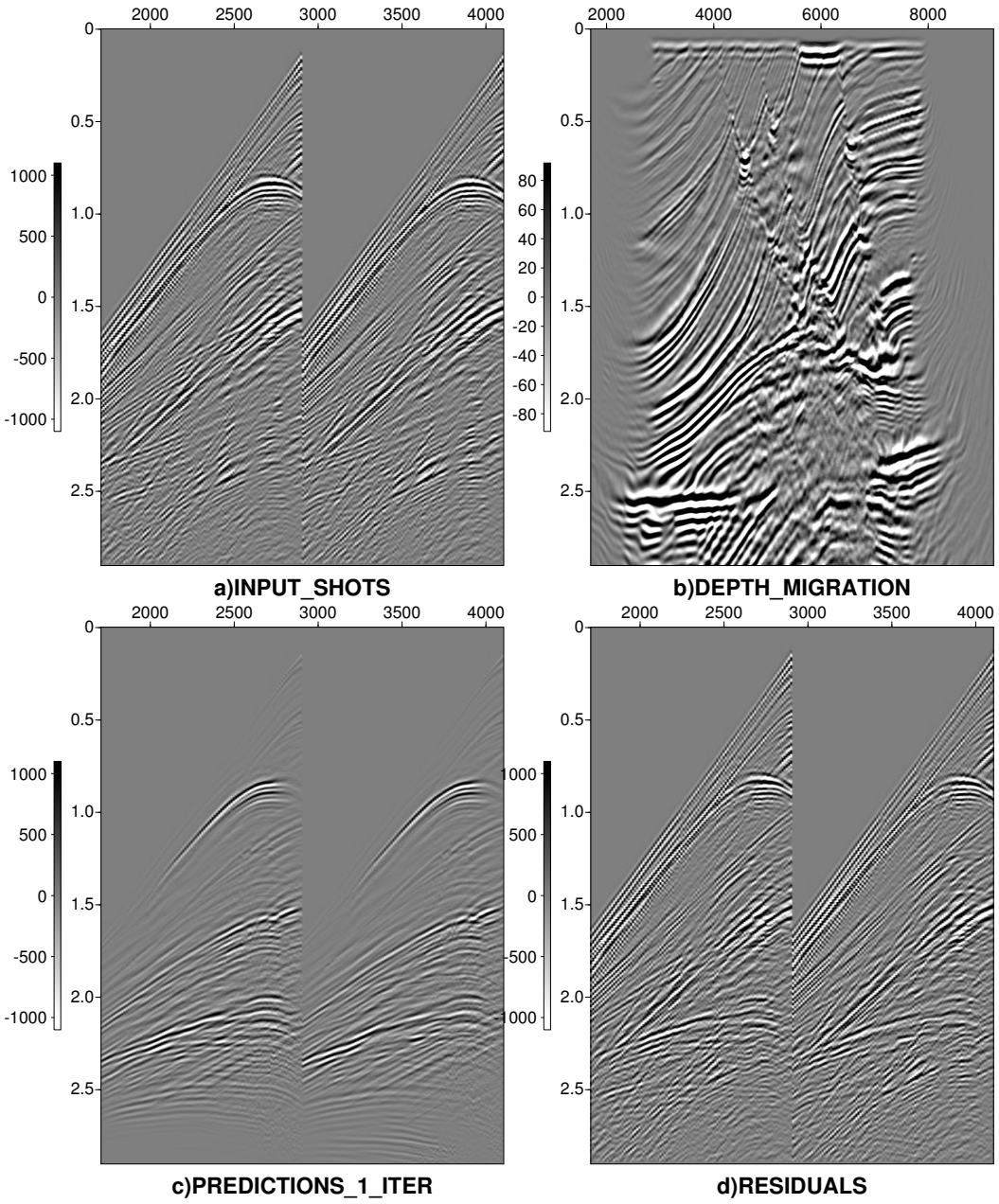

FIG. 11. Predicted shots using depth migration (1 iteration of LSMIG). a) input shots, b) migration, c) predictions, d) residuals. These are the shots in top of the most difficult part of the Marmousi model.

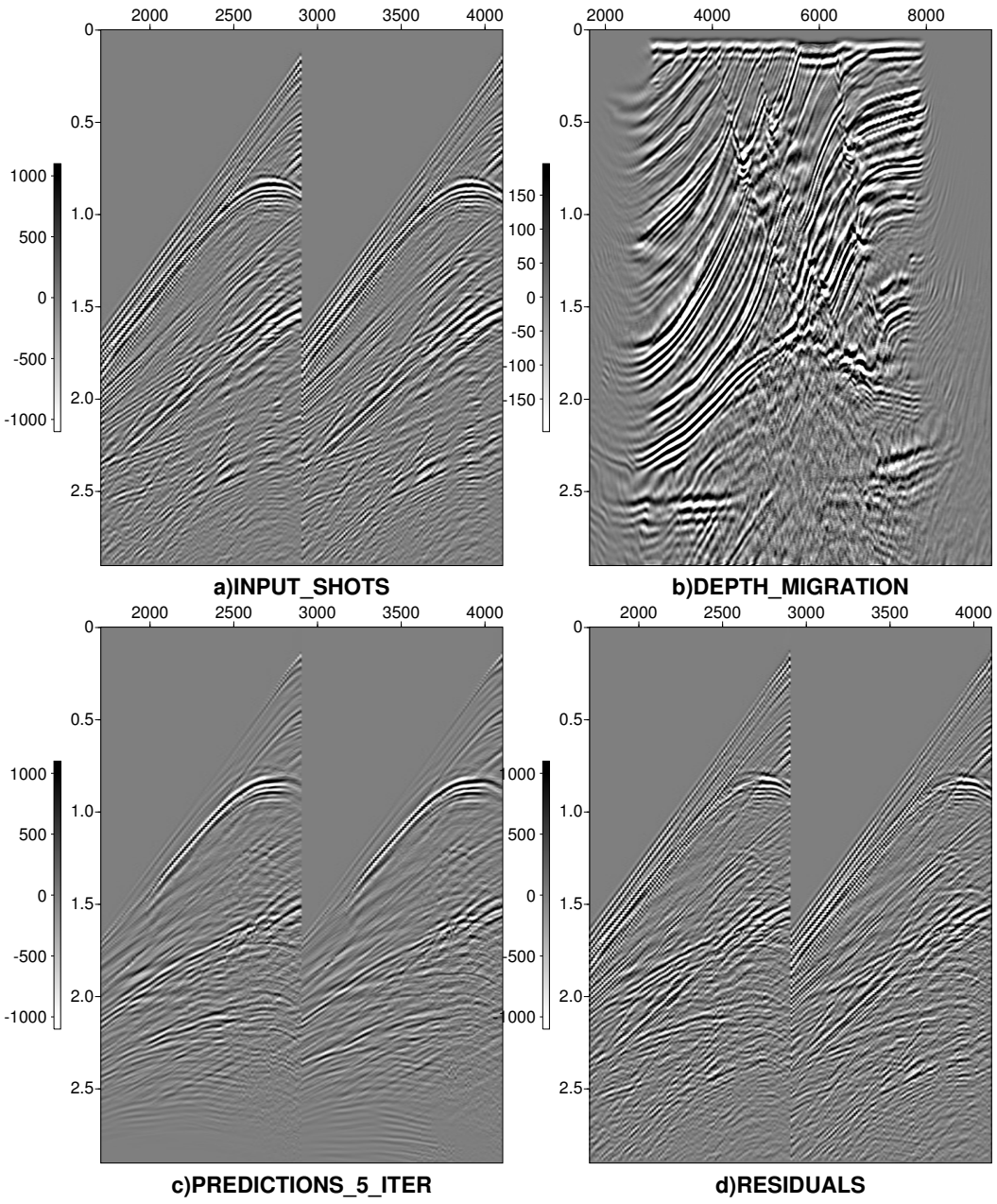

FIG. 12. Predicted shots using LSMIG with 5 iterations. a) input shots, b) migration, c) predictions, d) residuals. These are the shots in top of the most difficult part of the Marmousi model.

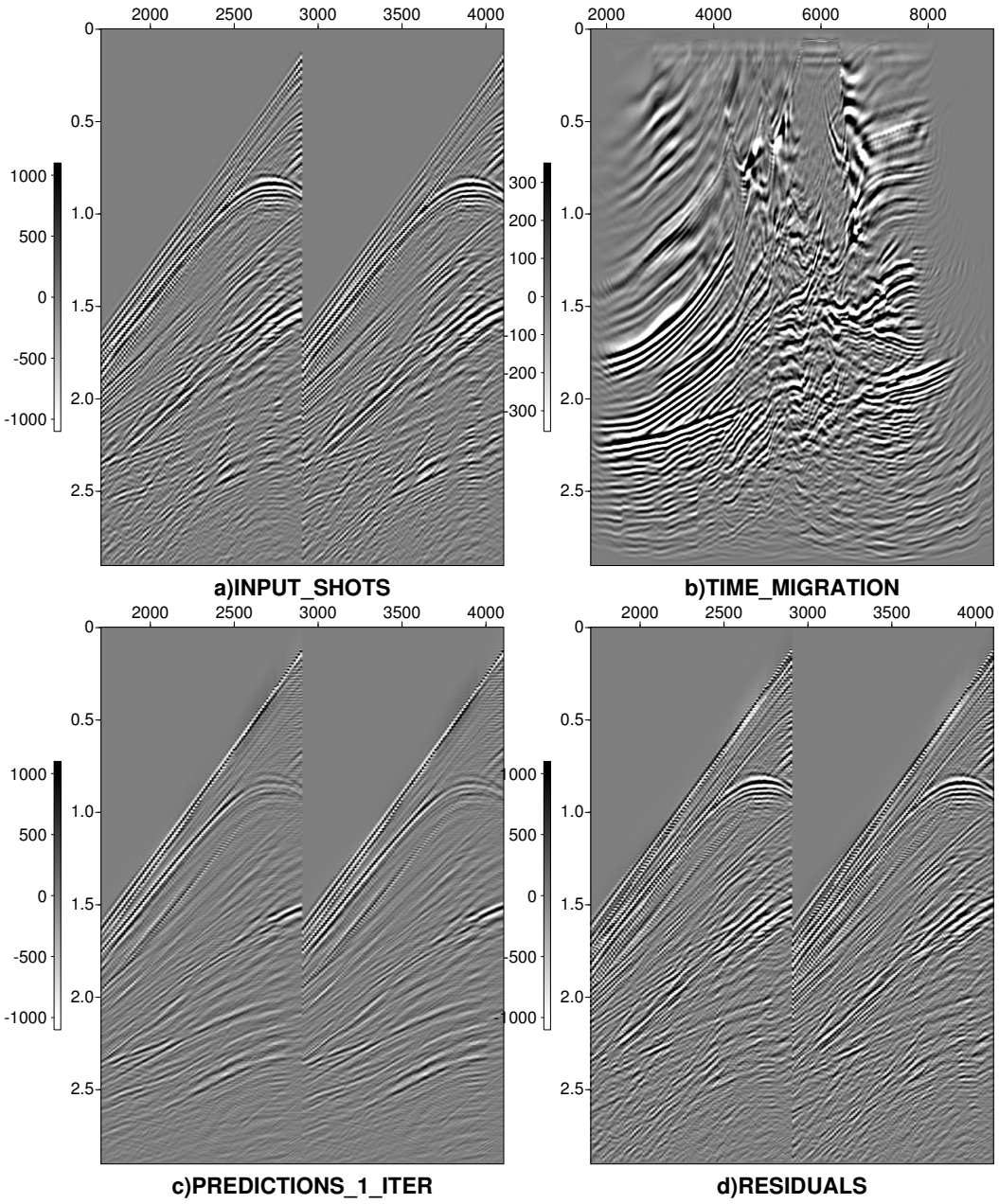

FIG. 13. Predicted shots using time migration (1 iteration of LSTMIG). a) input shots, b) migration, c) predictions, d) residuals. These are the shots on top of the most difficult part of the Marmousi model.

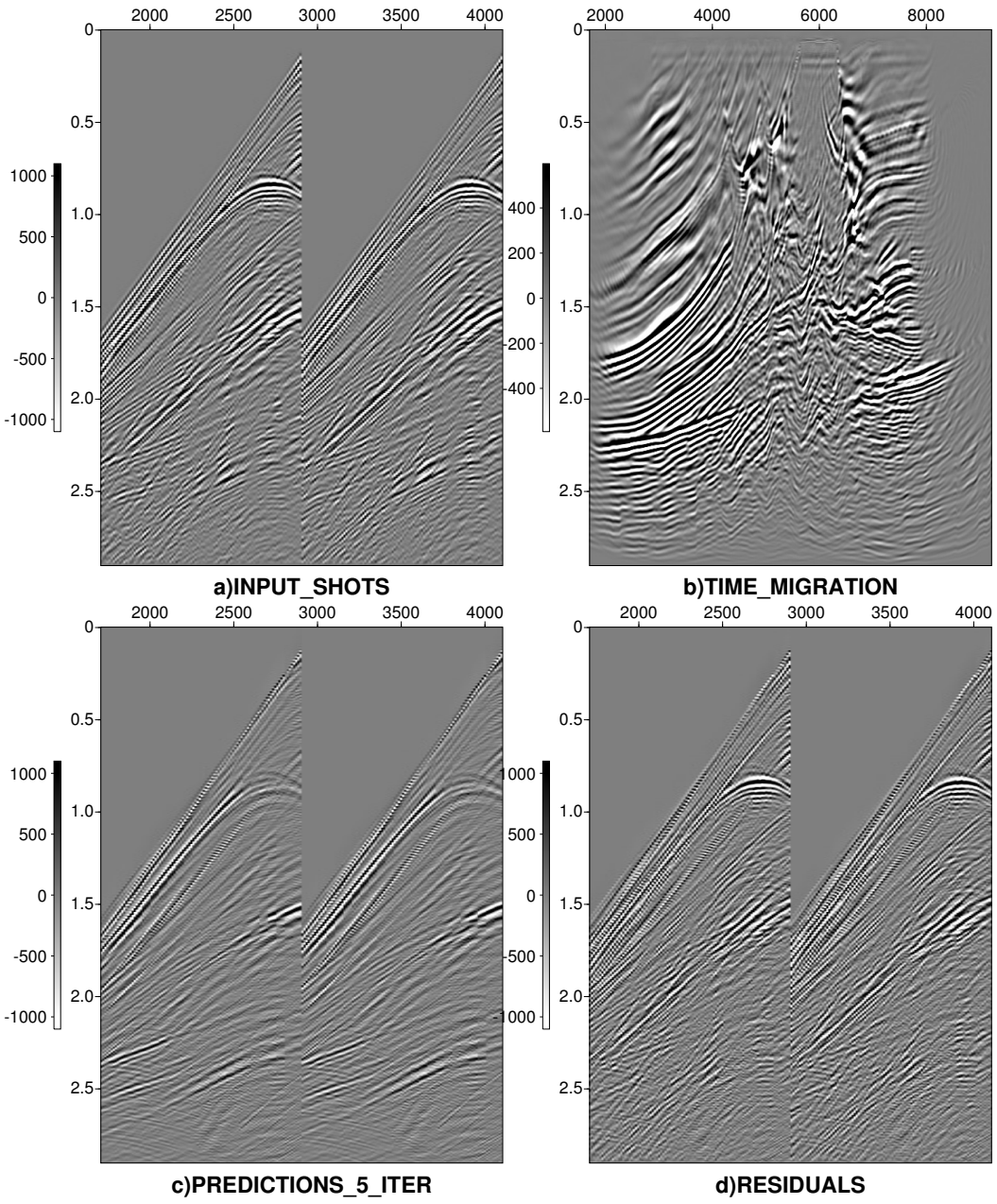

FIG. 14. Predicted shots using LSTMIG with 5 iterations. a) input shots, b) migration, c) predictions, d) residuals. These are the shots in top of the most difficult part of the Marmousi model.

CONCLUSIONS

This paper discussed several aspects of LSMIG in the context of physical transforms, and showed different possibilities in the operator design and implementation. Although straightforward in principle, there are many options related to the operator design whose choice can have significant influence in results and efficiency, and are not obvious from the theoretical point of view. Using synthetic tests, we can try to find an optimal implementation, but this is also challenging because the outcome of these tests depends on the complexity of the synthetic data. Also, by looking not at the quality of the image but at the predictions, we can obtain useful information. This is a practical approach because predictions show larger changes with iterations than images, since LSMIG is designed to adjust predictions. Finally, we go further in these experiments by testing how much the quality of predictions depends on the quality of the operator. Results show that a smooth, approximate operator seems to have the capability of creating acceptable data predictions, even when the related image may be wrong. This suggests a possible benefit of using time migration rather than depth migration for interpolation.

ACKNOWLEDGMENTS

My gratitude to Lorenzo Casasanta and Sam Gray for many enlightful discussions about migration during my work at CGG. I also thank CREWES sponsors for contributing to this seismic research. I also gratefully acknowledge support from NSERC (Natural Science and Engineering Research Council of Canada) through the grant CRDPJ 461179-13.

REFERENCES

Bleistein, N., and Gray, S. H., 2001, From the hagedoorn imaging technique to kirchhoff migration and inversion: Geophysical Prospecting, **49**, No. 6, 629–643.

Claerbout, J., 1992, Earth sounding analysis, Processing versus inversion: Blackwell Scientific Publications, Inc.

Claerbout, J. F., 1971, Toward a unified theory of reflector mapping: GEOPHYSICS, **36**, No. 3, 467–481.

Dellinger, J., Murphy, G., Etgen, J., Fei, T., and Gray, S., 1999, Efficient 2.5-d trueâĂŘamplitude migration: The Leading Edge, **18**, No. 8, 946–949.

Docherty, P., 1991, A brief comparison of some kirchhoff integral formulas for migration and inversion: GEOPHYSICS, **56**, No. 8, 1164–1169.

Fomel, S., 2007, Shaping regularization in geophysical-estimation problems: GEOPHYSICS, **72**, No. 2, R29–R36.

Kuhl, H., and Sacchi, M. D., 2003, Least squares wave equation migration for avp/ava inversion: GEOPHYSICS, **68**, No. 1, 262–273.

Lumley, D. E., Claerbout, J. F., and Bevc, D., 1994, AntiâĂŘaliased kirchhoff 3âĂŘD migration, 1282–1285.

Moghaddam, P. P., and Herrmann, F. J., 2005, Migration preconditioning with Curvelets, 2204–2207.

Nemeth, T., Wu, C., and Schuster, G. T., 1999, Least squares migration of incomplete reflection data: GEOPHYSICS, **64**, No. 1, 208–221.

Trad, D., 2015, Least squares kirchhoff depth migration: implementation, challenges, and opportunities, *in* 2015 SEG Annual Meeting, Society of Exploration Geophysicists, 4238–4242.

Zhang, Y., Gray, S., and Young, J., 2000, Exact and approximate weights for Kirchhoff migration, 1036–1039.