Lebesgue-type partitions of unity for operator localization

Michael P. Lamoureux and Gary F. Margrave

ABSTRACT

Partitions of unity (POUs) that track geological features give an effective means to reduce computational complexity in time-frequency methods. Such methods typically localize operators using windowed Fourier transforms to implement Gabor multipliers or more general pseudodifferential operators. We give a description of a method to create non-uniform partitions of unity and show the approximations obtained by these non-uniform versions are as good as the uniform method, while reducing complexity and numerical errors due to window edge artifacts. This windowing is a key step in Gabor decon and Gabor wavefield extrapolation.

INTRODUCTION

Seismic imaging occurs in the context of inhomogeneous, anisotropic media, and mathematical modeling of the physical propagation of waves must take into account the local and global variations of parameters that characterize physical properties of the geology. Processing of data should respect these inhomogenities, and typically must be designed in a nonstationary manner, to track the known parameter variations.

A specific implementation of nonstationary filtering that our research group has been developing uses the Gabor transform and Gabor multipliers in a variety of seismic data processing applications such as spectral deconvolution, depth migration, and reverse time migration. Some recent work on this include (Wards et al., 2008), (Ma and Margrave, 2008), (Ismail, 2008), (Ma and Margrave, 2007a), (Ma and Margrave, 2007b), (Henley and Margrave, 2007), (Montana and Margrave, 2006), (Margrave and Lamoureux, 2006), and (Grossman, 2005). Some foundational references include (Margrave et al., 2003b), (Margrave et al., 2003a) and (Margrave and Lamoureux, 2002). Gabor techniques are an extension of the Fourier transform methods applied to localized signals, allowing mathematical models with inhomogeneities in the physical material being studied.

The key step in the Gabor method is to break up a signal into small, localized packets by multiplying the signal with a window function. Typically, the window is a smooth “bump” function, such as a Gaussian, localized at the point of interest in the signal. The localized packet can then be analyzed or modified using Fourier techniques. This is done for a collection of windows, covering the entire extent of the window. Finally, all the processed packets are re-assembled into one full, processed signal which is the result of the nonstationary filtering.

An illustration of this windowing method is presented Figure 1, which shows a sinusoid signal aligned above a bump function. The product of the two is shown in the bottom third of the figure, illustrating a short packet of signal in the middle of the time axis. This packet is a localization of the signal.

More generally, in Figure 2, a signal is decomposed into a sum of several wave packets,

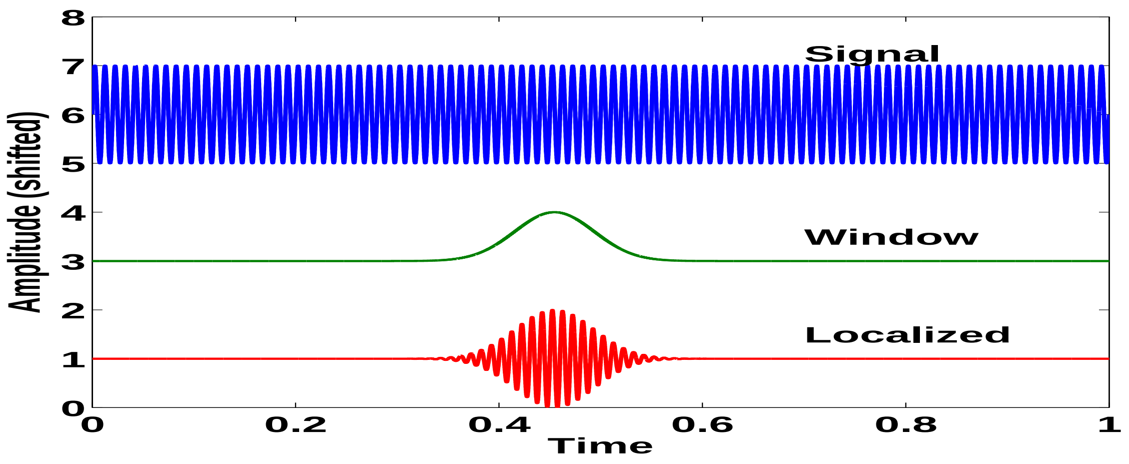

FIG. 1. Localizing a signal with a window.

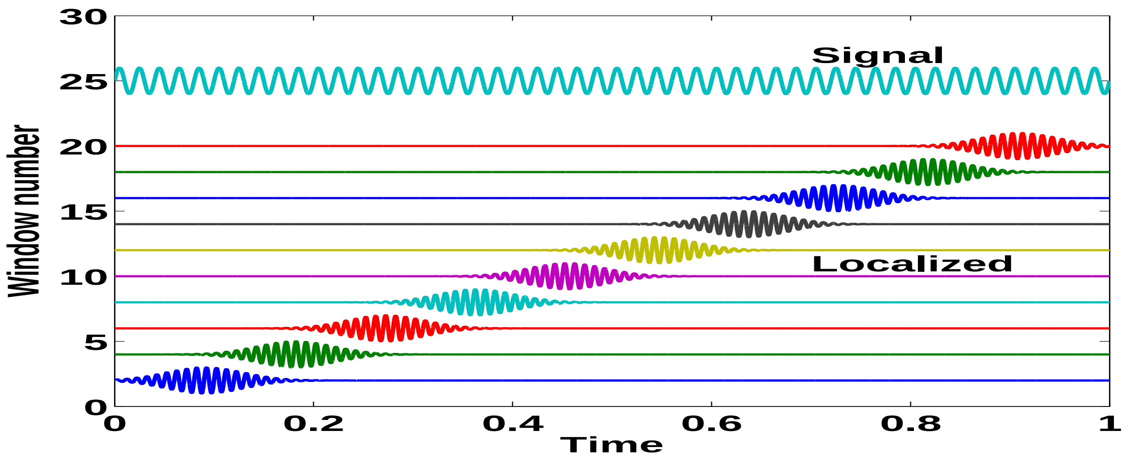

FIG. 2. Breaking up a signal into localized pieces.

illustrated in the figure as a set of 10 packets, localized to different positions on the time axis. For this particular example, the packets are the same except for their location. For a more complex signal, the packets would be different, with each packet capturing the local information of the signal in the time interval framed by the corresponding window.

The choice of windows is not completely arbitrary. For instance, the whole collection of windows should cover all of signal space – if they miss an interval, then there is an interval of signal that is not "seen" by the decomposition into wave packets. More precisely, we expect the window functions $w_k(t)$ should give enough information for us to rebuild the original signal from its constituent packets. A useful way to achieve this is to require that windows to satisfy the *partition of unity* condition, namely that these window functions should add up to one:

$$\sum_{k=1}^{M} w_k(x) = 1. \tag{1}$$

The idea is that the windows "partition" the constant function one (unity) into a series of little bumps. This is illustrated in Figure 3, where a series of ten bump functions are aligned

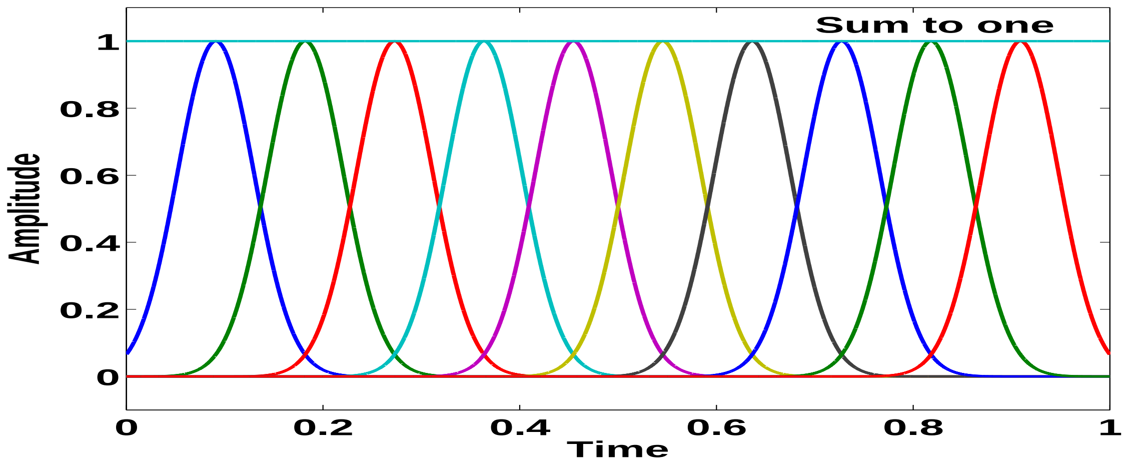

FIG. 3. Windows that sum up to one: a partition of unity.

that add up, uniformly, to one.[*]

Given a signal $f(x)$, the localized packets of signal are given by multiplying by the window function, which give the functions

$$f_k(x) = w_k(x)f(x). \tag{2}$$

Because of the partition of unity condition, the signal can be recovered from the localized packets simply by summing, so

$$f(x) = \sum_k f_k(x). \tag{3}$$

That is, since $\sum_k w_k(x) = 1$, we have

$$f(x) = \left(\sum_k w_k(x)\right) f(x) = \sum_k w_k(x)f(x) = \sum_k f_k(x). \tag{4}$$

A nonstationary filter is created by applying different filters to each signal packet, and then adding together the results. For instance, for each index k one fixes a convolution operator C_{g_k} which acts by convoluting the signal packet with a given function g_k. The resulting nonstationary filter is an operator C that acts on signal $f(x)$ by the formula

$$Cf = \sum C_{g_k} f_k = \sum_k g_k * f_k, \tag{5}$$

where the notation $g_k * f_k$ is just the ordinary convolution of g_k with f_k.

Using M_{w_k} to represent the operation of multiplication by the window function w_k, we can write the nonstationary filter operator C as a sum of convolution and multiplication

[*]In this example we have dropped a couple of windows at the ends, for a simpler picture.

operators,

$$C = \sum_k C_{g_k} M_{w_k}. \tag{6}$$

It is worth noting that a similar, but subtly different, nonstationary filter is obtained by reversing the order of the products, giving a new operator

$$C' = \sum_k M_{w_k} C_{g_k}. \tag{7}$$

This operator C' filters the original, unlocalized signal f by each of the convolution operators C_{g_k}, localizes each result, and sums together. Such an operator turns out to be the adjoint of operators of the form in Equation 6, and is useful for creating inverse operators.

It is also handy to split the window into two symmetric halves, and put them on either side of the convolution operator. This gives a new nonstationary filter, defined as

$$C'' = \sum_k M_{w_k}^{1/2} C_{g_k} M_{w_k}^{1/2}. \tag{8}$$

Such a filter first localizes the signal, filters it, localizes the result, and then sums together all the localized results. Such filters are particularly useful when careful control over the operator bounds is required. (See (Ismail, 2008).) Such a division of the windows is called a split partition of unity.

The goal of this paper is to point out that the computational effort in applying one of these filters (Equations 6,7,8) grows with the total number of windows. There are also numerical artifacts introduced by the filtering operation that correspond to the transitions present at the window edges. Thus, a key to reducing computational effort and minimizing numerical artifacts is to reduce the number of windows and their edges.

However, one must also be able to successfully approximate the varying physical parameters in mathematical models of the media being studied. What we show here is that using a Lebesgue-type partition of unity, one can use few windows while retaining a highly accurate approximation.

NON-UNIFORM WINDOWS

There is no particular reason that the windows used in these nonstationary filters must all look alike. It is quite possible to use windows of different forms. All that is required is the partition of unity condition.

As a simple example, consider Figure 4 which shows three boxcar windows spanning a time frame from $t = 0$ to $t = 1$. The two boxcars at the ends are rather short, and the third in the middle is long. Together they add to one, so the POU condition is satisfied on the interval of interest.

As localized filtering with sharp cutoff windows can have painful artifacts, it is useful to smooth these boxcar windows, as shown in the bottom half of Figure 4. Again, in the

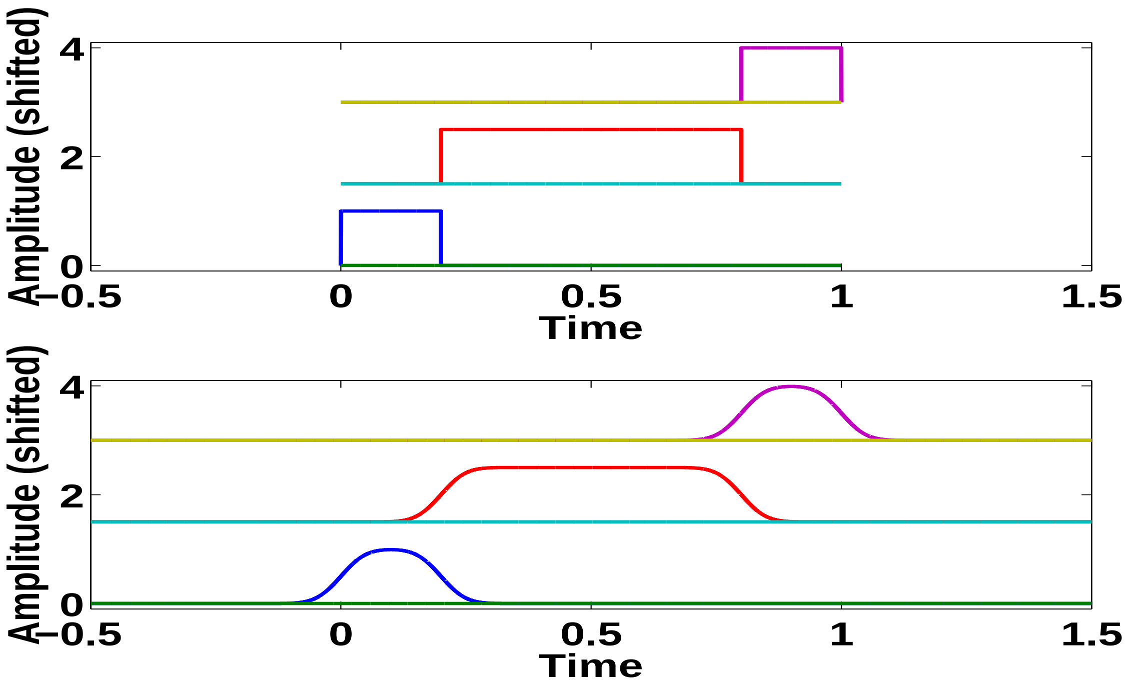

FIG. 4. Boxcar windows of assorted sizes, then smoothed.

interval of interest, the windows sum to one, so even the smooth windows satisfy the POU condition.

There is a simple trick to smooth a POU and have the resulting windows also satisfy a POU condition. Suppose

$$\sum_{k=1}^{M} w_k(x) = 1 \tag{9}$$

is a collection of windows satisfying the POU condition. Fix a smoothing function $g(x)$, say a Gaussian bump function, that has been normalized to unit mass. That is, we have

$$\int g(x)\,dx = 1. \tag{10}$$

Define the smoothed windows as the convolution with the bump function, so

$$w_k' = g * w_k. \tag{11}$$

The new windows w_k' satisfy the POU condition, since

$$\sum_k w_k' = \sum_k (g * w_k) = g * \left(\sum_k w_k\right) = g * 1 = \int g(x)\,dx = 1, \tag{12}$$

by the normalization condition on the smoother.

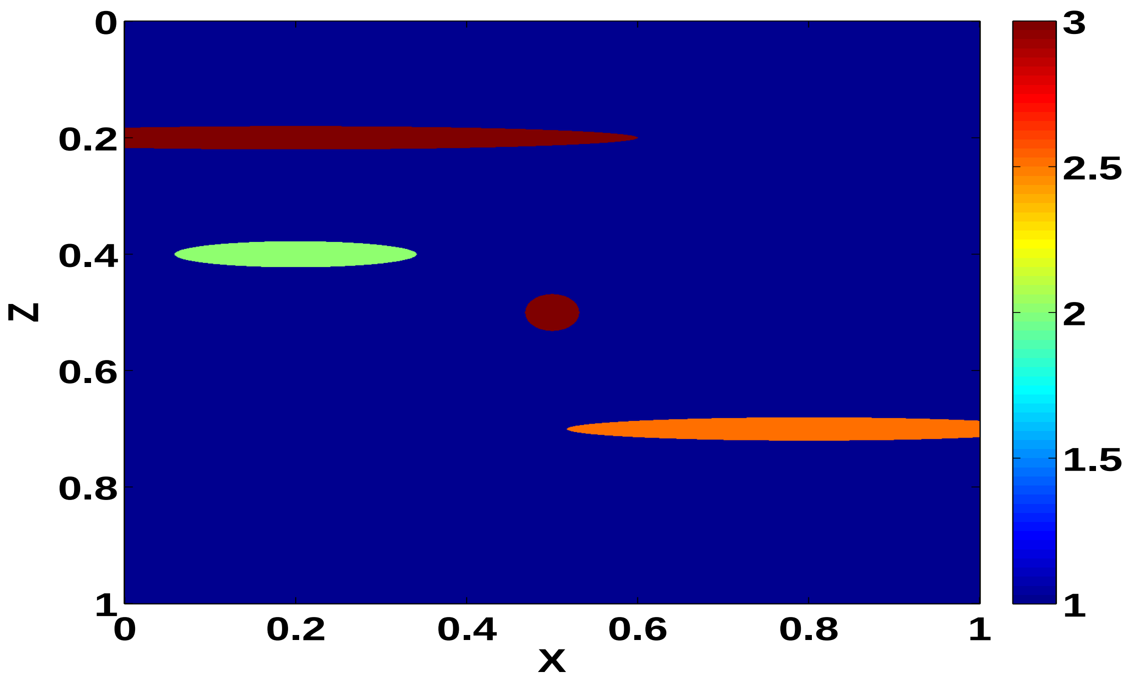

FIG. 5. A 2D velocity field.

NON-UNIFORM WINDOWS IN 2D

We illustrate the efficiency of non-uniform windows in two dimensions by considering a model of a heterogeneous velocity field shown in Figure 5. This model shows long and narrow, horizontal geological features. To model these features well with uniform windows – that is, to localize a nonstationary filter to match these geological trends – one would require many small, symmetric windows that tile the plane quite densely, as in Figure 6. From these 400 small windows, one selects the few that line up with the geology, and creates a window map as in Figure 7. Here, some 27 windows are needed to track the features. Another 373 windows are required to track the background.[†]

Obviously, the computational effort for dealing with the 400 windows will be large.

A better way is to build windows that specifically track the features. As an example, Figure 8 show a boxcar-type window which is equal to one along the top horizontal feature, and zero elsewhere. This 0-1 function is called an indicator function; where it takes the value one indicates the location of the feature it is tracking. Choosing a 2D Gaussian smoother as shown in Figure 9, the boxcar window can be rounded out to create a smoothed window as shown in Figure 10.

A similar construction is used to create three other smooth windows, and the four geological features of the velocity field are tracked by the four windows shows in Figure 11.

[†]In fact, the 400 windows shown in Figure 6 are just a representative sample. A careful analysis of this problem shows something like five to ten times more uniform windows are needed to accurately model this velocity profile.

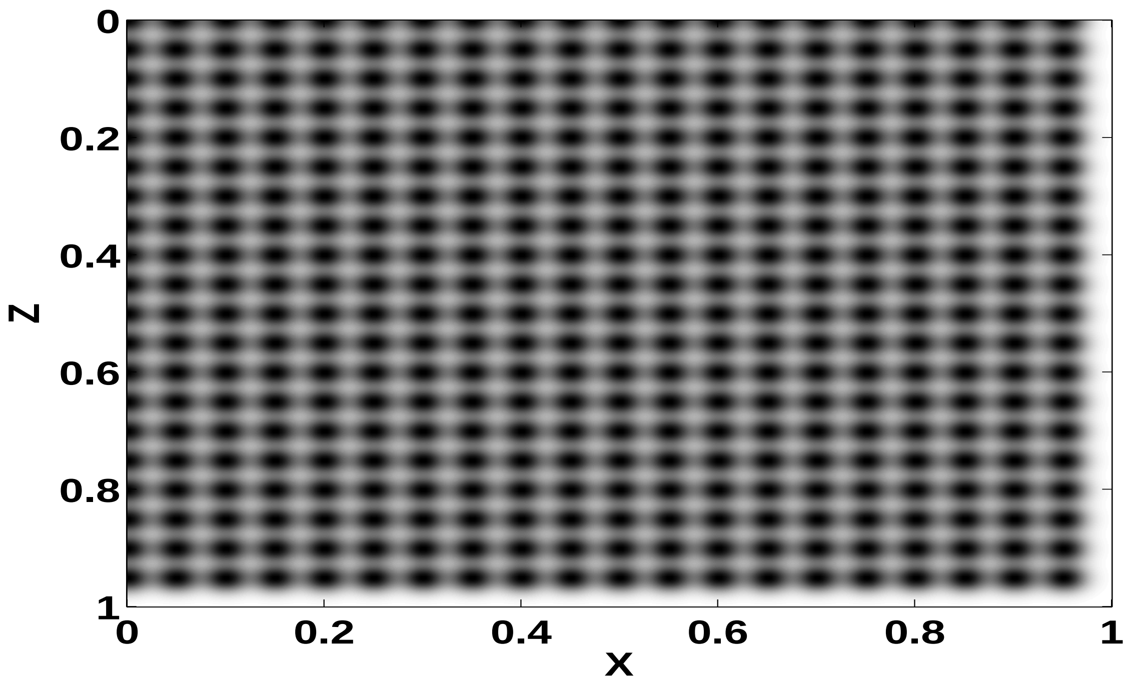

FIG. 6. A tiling with 400 small windows.

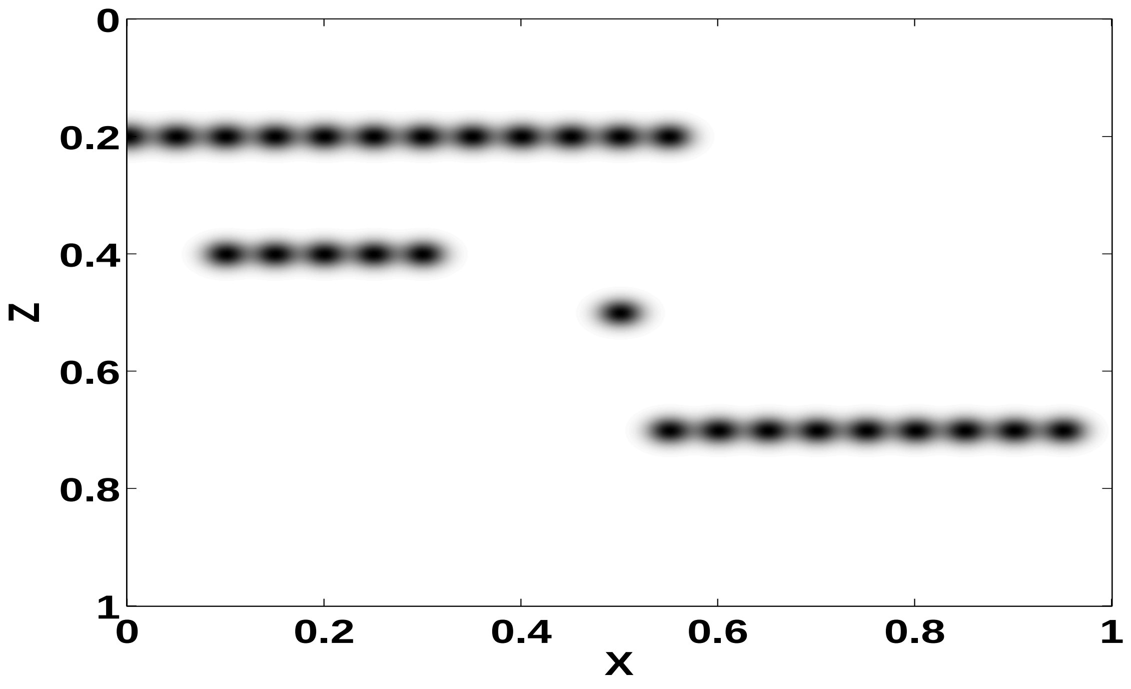

FIG. 7. Twenty seven small windows tracking the geological features.

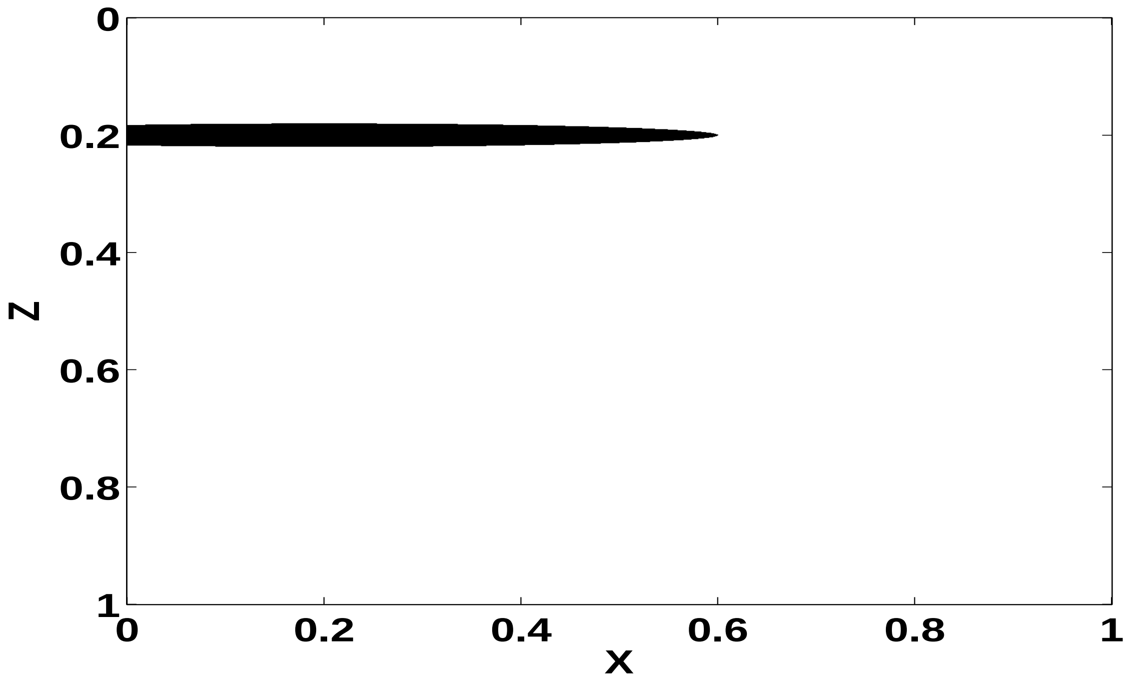

FIG. 8. A simple 0-1 window tracking one geological feature.

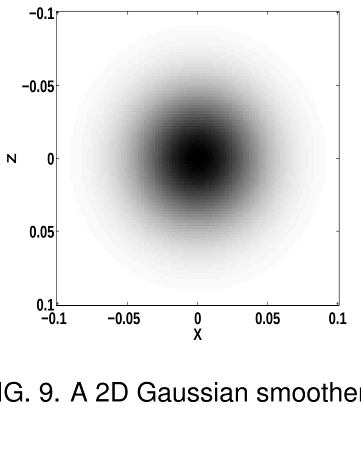

FIG. 9. A 2D Gaussian smoother.

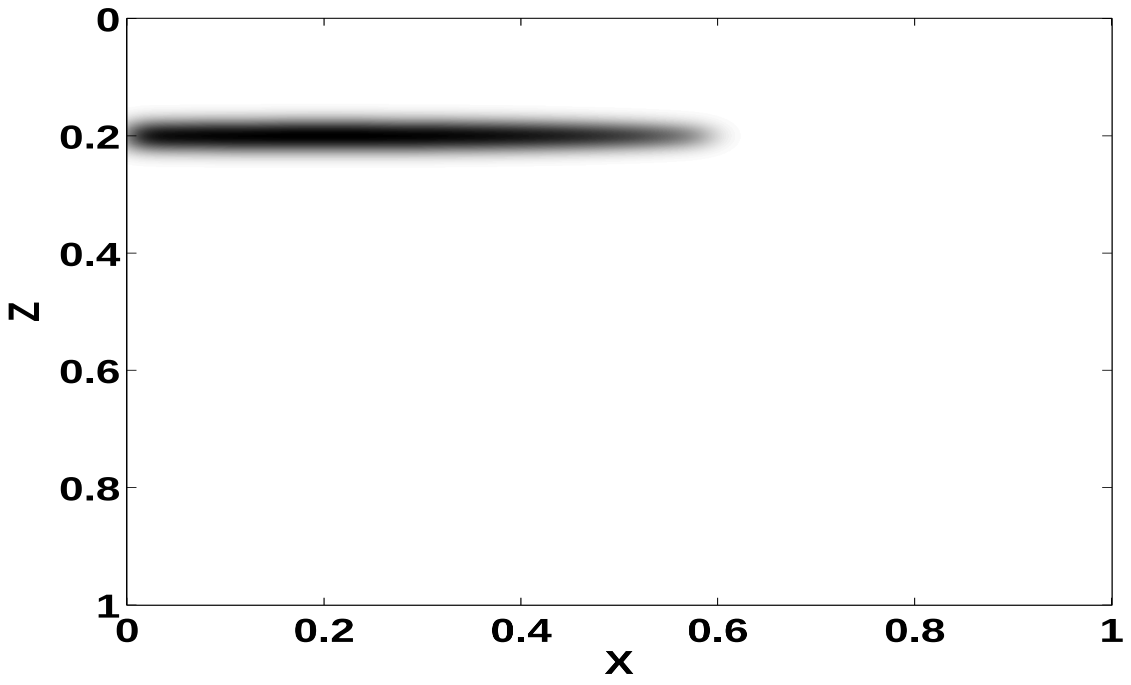

FIG. 10. A smoothed window, tracking one geological feature.

A fifth window is used to model the background. Thus with a total of only five windows, the velocity field is approximated to as good an estimate as with the 400 small, uniform windows.

THE APPROXIMATION RESULT

The construction of feature-tracking windows can be automated. The key step is to select a range of sample velocities $v_1, v_2, \ldots, v_M$ that represent the typical values expected in the velocity fields, and define the corresponding window w_k as an indicator function that equals one on the positions where the velocity $v(x)$ is closest to sample velocity v_k. So, we write

$$w_k(x) = 1 \text{ if } v_k \text{ is closest to } v(x), \text{ and 0 otherwise.} \tag{13}$$

This creates a range of window functions $w_1(x), w_2(x), \ldots, w_m(x)$. By definition, these windows w_k satisfy the partition of unity condition. These windows can be made smoother by convolving with a bump function (smoother) of unit mass, as mentioned in the last section. Thus we have created a set of smooth windows $w'_k(x)$ which also satisfy the POU condition.

The windowed approximation to the velocity field $v(x)$ is given by the sum

$$\tilde{v}(x) = \sum_{k=1}^{M} v_k w'_k(x). \tag{14}$$

In practice, this approximation is very close to the original velocity field. For a piece-wise constant velocity field, it can often be made exact except for a set which follows a

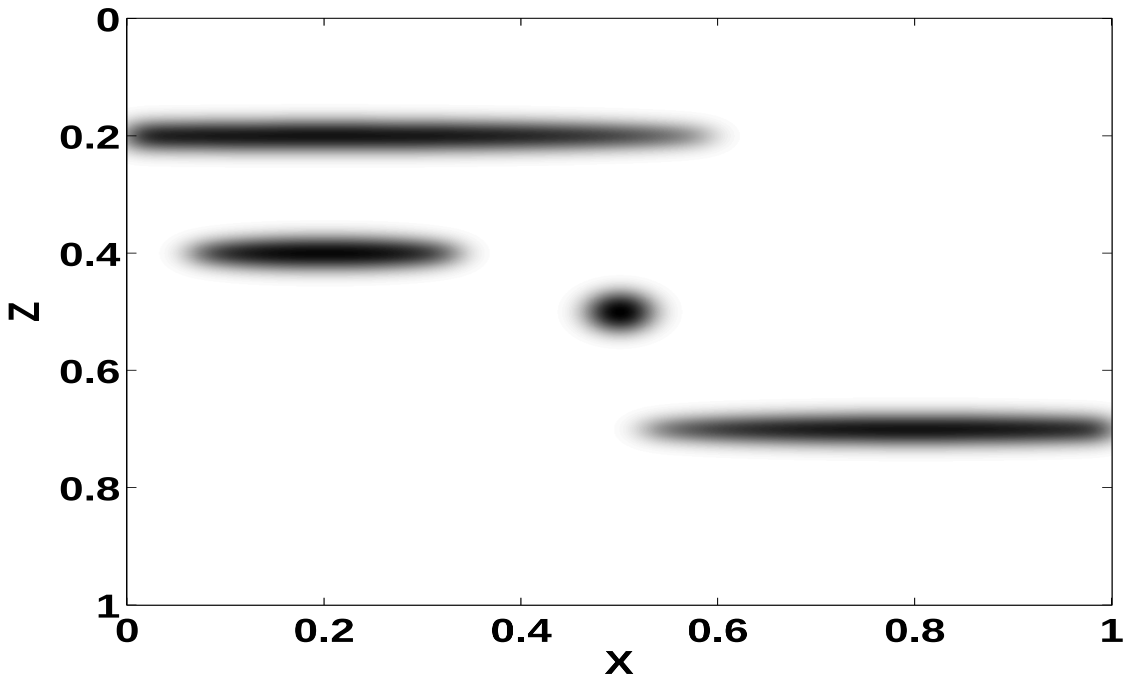

FIG. 11. Four smoothed windows, one for each feature.

blurred outline of the path of discontinuities (essentially, along the window edges).

A precise mathematical statement can be made:

Theorem 1 *Given a piecewise continuous velocity field $v(x)$ with upper and lower limits v_{min}, v_{max}, the construction above can be selected with M windows to give an approximating function $\tilde{v}(x)$ which satisfies*

$$|\tilde{v}(x) - v(x)| \leq \frac{v_{max} - v_{min}}{M}, \tag{15}$$

at all points x except those within some ϵ of the discontinuity points of $v(x)$, where ϵ is the width of the smoother.

In our 2D example of the previous section, we saw that $M = 5$ windows were enough to give an accurate approximation to the velocity field, a great improvement over the use of 400 small, uniform windows.

This approximation of the velocity field is key to the approximation of Gabor multipliers that represent, say, the wave equation in an inhomogeneous medium. For details on this, we refer the reader to (Lamoureux et al., 2008).

A more difficult result is to show that the corresponding nonstationary filters of the form

$$C = \sum_{k=1}^{N} C_{g_k} M_{w_k} \tag{16}$$

will be accurate representations of the physics of wave propagation in an inhomogeneous medium, where C_{g_k} is the constant velocity model for the corresponding velocity v_k. We have numerical evidence to suggest this is a good approximation and have been developing algorithms accordingly.

CONCLUSIONS

Windowed filtering is an effective means of implementing nonstationary filters, which are a useful method of modeling wave propagation in nonhomogeneous media. An automated procedure has been presented that provides a robust method for creating nonuniform windows that accurately track inhomogeneities and allows for good approximations with few windows. The small number of windows allows for computational efficiencies and the reduction of window edge artifacts.

ACKNOWLEDGEMENTS

This research is supported by grants from NSERC, MITACS, and the sponsors of the POTSI and CREWES consortia.

REFERENCES

Grossman, J. P., 2005, Theory of adaptive, nonstationary filtering in the gabor domain with applications to seismic inversion: Ph.D. thesis, University of Calgary.

Henley, D. C., and Margrave, G. F., 2007, Gabor deconvolution: surface and subsurface consistent, Tech. rep., CREWES.

Ismail, S., 2008, Nonstationary filters: M.Sc. thesis, University of Calgary.

Lamoureux, M. P., Margrave, G. F., and Ismail, S., 2008, Solving physics pde's using Gabor multipliers, Tech. rep., CREWES.

Ma, Y., and Margrave, G. F., 2007a, Gabor depth imaging with topography, Tech. rep., CREWES.

Ma, Y., and Margrave, G. F., 2007b, Gabor depth migration using a new adaptive partitioning algorithm: CSEG Expanded Abstracts.

Ma, Y., and Margrave, G. F., 2008, Seismic depth imaging with the Gabor transform: Geophysics, **73**, S91–S97.

Margrave, G. F., Dong, L., Gibson, P. C., Grossman, J. P., Henley, D. C., and Lamoureux, M. P., 2003a, Gabor deconvolution: extending wiener's method to nonstationarity: The CSEG Recorder, **Dec**.

Margrave, G. F., Henley, D. C., Lamoureux, M. P., Iliescu, V., and Grossman, J. P., 2003b, Gabor deconvolution revisited: SEG Technical Program Expanded Abstracts, **73**.

Margrave, G. F., and Lamoureux, M. P., 2002, Gabor deconvolution: CSEG Expanded Abstracts.

Margrave, G. F., and Lamoureux, M. P., 2006, Gabor deconvolution: The CSEG Recorder, **Special edition**, 30–37.

Montana, C., and Margrave, G. F., 2006, Surface-consistent gabor deconvolution: SEG Technical Program Expanded Abstracts, **76**.

Wards, B., Margrave, G. F., and Lamoureux, M. P., 2008, Phase shift time stepping for reverse time migration: the Marmousi data experience, Tech. rep., CREWES.