Comments on fast-filtering and a partition of unity

John C. Bancroft

ABSTRACT

Fast-filtering is a process that differentiates a filter operator a number of times and integrates the input data the same number of times. The differentiation diminishes the operator to a few delta functions that reduce convolution to the summation of few integrated samples. The filter operators are typically formed by repeated convolution of a boxcar with itself a number of times that will tend to a Gaussian shape when the number of convolutions becomes large. A finite number of these fast-filter operators will not form a partition of unity and are not suitable for decomposing a trace into short time windows to approximate stationary analysis. They do however indicate how operators can be created that will form a partition of unity and maintain the features of fast-filtering.

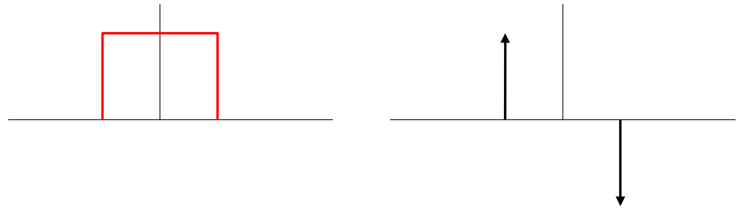
INTRODUCTION

Fast-filtering

Simple filter operators can be formed by convolving a boxcar with itself a number of times to produce a triangular, quadratic, etc. shaped window. The shape of these windows tends to a Gaussian shape as the number of iterations tends to infinity. All of these windows can be used for fast-filtering in which the operator can be differentiated back to delta or impulse functions where the order of differentiation equals one plus the number of convolutions. Input data is integrated to the same order. Fast-filtering is accomplished by the convolutional process of multiplying and summing, however, because the window is only composed of a few delta functions, the computations are significantly reduced.

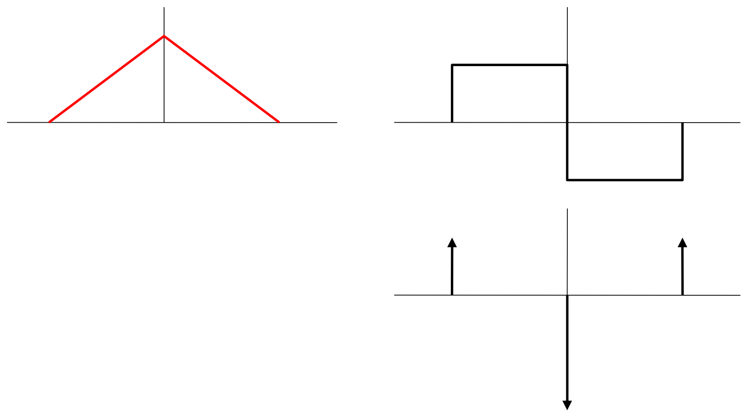
Consider a boxcar widow shaped filter that is applied to a seismic trace. Traditionally, one filtered sample is found by summing all samples in the window, and the sum divided by the number of samples within the window. In fast-filtering, integration produces a new trace where each point is the sum of all the previous samples. The difference between any two points on the integrated trace is the sum between those same points on the original trace. Differentiation of the boxcar produces two delta functions whose location and polarity define the beginning and end of the boxcar, and when they are multiplied with the integrated trace, the sum of these two points is identical to the sum over the boxcar window. Consequently a fast-filtered sample requires two summations, while the conventional method requires a number of summations defined by the size of the boxcar.

Triangular filtering Lumley (1994) uses two integrations and differentiations for fast-filtering and only requires four summations, (one on each end and two for the central point). In contrast, conventional filtering requires a multiplication of each point with the filter shape and a corresponding number of summations. The same concept applies to all higher order filters. The first three of these filter windows are shown in Figures 1 to 3.



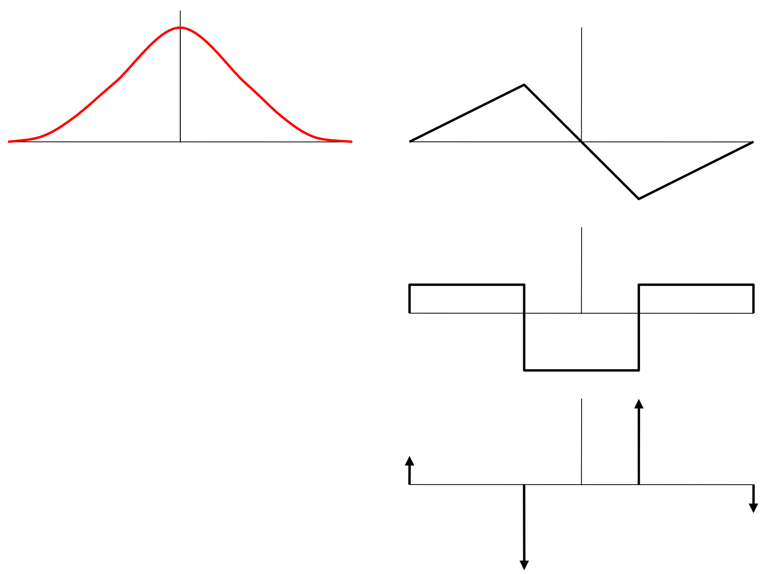
a) b)

FIG. 1. a) a boxcar window and b) one derivative to get delta functions.



a) b)

FIG. 2. a) A triangle window and b) two derivative to get delta functions.



a) b)

FIG. 3. a) A quadratic window and b) three derivatives to get delta functions.

Consider a triangle filter that may cover samples from -10 to 10. A conventional approach requires weighting and summing the 21 sample points or 42 floating point operations (FPO) for each filtered sample. Fast-filtering applies a second order differentiation to the triangle filter to produce three delta functions with an amplitude of +1 at n = -10, -2 at n = 0, and +1 at n = 10 as illustrated in Figure 2b. When applied to a doubly integrated trace, only four summations and one scaling multiply or 5 FPO are required for each filtered sample.

Comment 1. An alternate view of fast-filtering occurs in the Fourier transform domain where the convolution process of filtering $f*g$ becomes multiplication F G. Integration and differentiation are also multiplications in the transformed domain, but they are inverses of each other and the net effect is that they cancel.

Comment 2. The solutions of odd orders of integration and differentiation lie midway between sample points and produce a corresponding phase shift. However, even orders of integration can be approximated pairs of top-down and then bottom-up summations. Even orders of differentiation can be produced by pairs of forward and backward differences, which produce no phase shift.

Comment 3. Integration of the input data is performed once, as is the differentiation of the filter operator.

Comment 4. In applications such as interpolation (or migration), the delta functions of the filter window may not lie on a sample location. If necessary, interpolation of neighbouring samples may be required.

Applications of fast-filtering

Convolutional filtering will be more efficient (typically) when the size of the filter operator is less than 30 points, while Fourier transform filtering will be more efficient (again typically) if the time domain filter operator is larger than 50 points. Fast-filtering will therefore allow much larger windows than conventional methods before the Fourier transform methods are faster.

The fast-filtering method is ideal when used as an anti-aliasing filter (AAF) for Kirchhoff type migrations where one migrated sample is produced by summing many samples in a diffraction shaped window. As the slope of the diffraction window increases, increased high cut filtering of the local input data is required to reduce aliasing. Consequently, when the diffraction window is aligned with an input trace, local filtering of that data is required. That same data will be migrated to many other locations, each with a different AAF requirement, i.e., when spreading energy on a circle, the input trace will require a different AAF for all traces crossed by the circle.

The reason these filters are popular is that they can provide very fast filtering by differentiating the filter to a given order that reduces it to delta functions. The data to be filtered is integrated to the same order. Filtering is accomplished by weighting and summing only the input data that aligns with the delta functions.

Window shapes required for a partition of unity

One-dimensional data may be decomposed into many short segments that when summed together reproduce the original data. Summing the windows that created the segments forms the partition of unity. An example would be a boxcar window of any or varying size that is applied to the input data where each segment is butted to the previous and following segments. However, a segment of data created by a boxcar is not as useful as a segment in which the sides are tapered and are more suitable for Fourier transformation. The central portion of a window and the segments that overlap must sum to unity to form what mathematicians refer to as a partition of unity (PU).

In an overlap region, the trailing edge of a window $f_i(n)$ must sum to unity with the leading edge of the next window $f_j(n)$, i.e.,

$$f_i(n) + f_j(n) = 1.0\,, \tag{1}$$

where n is the sample number of the input data.

In other applications, the size of the windows may be much larger than the increment of the segment and there will be multiple overlaps from windows in the neighbourhood. Once again these multiple overlaps are required to sum to unity.

Fourier transform applications usually require smoothly varying window shapes, such as a raised cosine, or Hanning window. A feature of the Hanning window is that it complies with the PU requirement and its size is only twice the move-along of the segment. The shape of a Hanning window $h_{Han}(n)$ is defined by

$$h_{Han}(n) = 0.5\left[1+\cos\left(\pi n / N\right)\right], \quad -N \le n \le N \tag{2}$$

where N is the half size of the window. Summing the overlapping samples from 0 to N for a zero lag window and one with a lag (or advance) of N points, we have

$$\begin{aligned} h_{Han}(n) + h_{Han}(n+N) &= 0.5\left[1+\cos\left(\pi n / N\right)\right] + 0.5\left[1+\cos\left\{\pi\left(n+N\right)/N\right\}\right] \\ &= 1.0 + 0.5\left[\cos\left(\pi n / N\right) + \cos\left(\pi + \pi n / N\right)\right] \\ &= 1.0 \end{aligned} \quad . \tag{3}$$

The Fourier transform of the Hanning window has a main central lobe that is relatively narrow, which is convolved in the frequency domain to produce a reasonably smooth spectrum of the input data segment.

The shape of the fast-filter window

The first two windows of fast-filtering, the boxcar and triangle, are PU compatible, however the remainder, as defined above are not compatible. For example the third window of Figure 3 has three sections of quadratic shape, but they are not PU compatible. This can be observed from the amplitude and location of the transition points as defined by the location of the impulse functions of the amplitude of the second derivative. These amplitudes are proportional to the coefficient of the squared term that defines the shape of the curve, and since they are different, the overlapping amplitudes will not sum to unity. In addition, the locations of the transition points are not centered on each side.

Although these window are not compatible with the PU requirement for decomposing an input trace, they do indicate how a similar window could be constructed that is compatible. For example, we would want the transition points to be symmetrical about each side and we would want the amplitude of the appropriate derivative to be equal so that the curvatures of the original window are the same.

A QUADRATIC WINDOW THAT IS PU COMPATIBLE

A quadratic shaped window (Q2) that is PU compatible may be defined by locating the transition point midway on each side of the window and using equal amplitudes of the second derivative. This window and its three derivatives are shown in Figure 4 for N = 20.

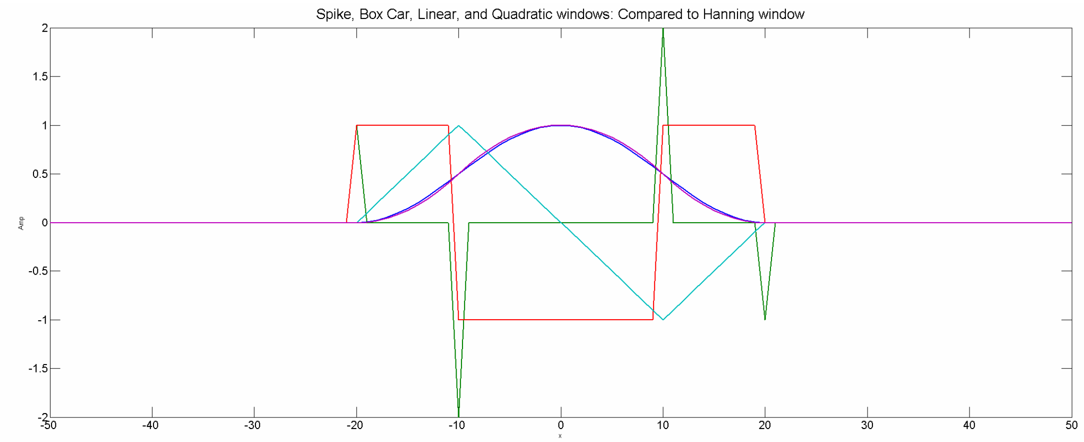


FIG. 4. A PU compatible quadratic window is shown in purple with the first, second, and third derivatives shown in cyan, red, and green. N = 20. A Hanning window is also shown in blue overlying the quadratic window (enlarged in Figure 5).

The difference between the quadratic and Hanning is illustrated in Figure 5, with the quadratic window in blue and the Hanning window in green.

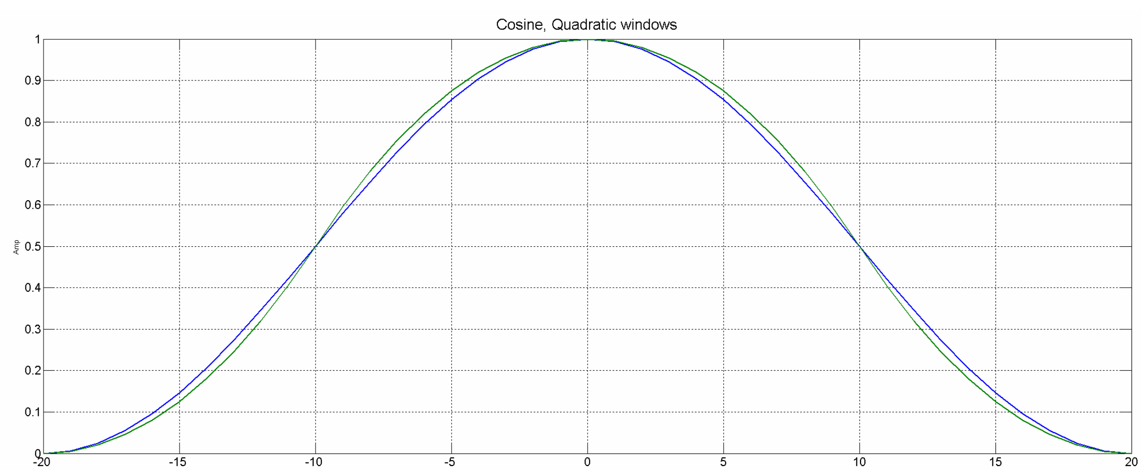


FIG. 5. A comparison between the PU compatible quadratic window with the Hanning window.

Although these windows are similar in appearance, their third derivative are significantly different. The Hanning window is still a sinusoid, while the quadratic window is reduced to four delta functions.

A goal of designing the Q2 wavelet was to reduce the computation time of the Gabor transform where we could integrate the input data and sum at the delta function locations to form the time varying spectra. It doesn't quite work out that simply as we need to include the exponential of the Fourier transform with the input trace in the integrations.

FAST-FILTERING THE GABOR TRANSFORM

Assume the convolution *c*(*t*) of a wavelet *w*(*t*) with a trace *g*(*t*),

$$c(t) = \int_{-\infty}^{\infty} w(t-\tau) g(\tau) \, d\tau \, , \tag{4}$$

where we integrate for one sample at time *t*. If we assume the wavelet to be symmetrical, and time limited to $\pm t_1$, we get

$$c(t) = \int_{t-t_1}^{t+t_1} w(\tau - t) g(\tau) \, d\tau \, . \tag{5}$$

If *w*(*t*) is differentiated to delta functions, and *g*(*t*) integrated the same number of times, this equation reduces to the fast-filtering form

$$\begin{aligned} c(t) &= \int_{t-t_1}^{t+t_1} \left(a\delta_{ta} + b\delta_{tb} + c\delta_{tc} + \ldots \right) g(\tau) \, d\tau \\ &= ag(t_a) + bg(t_b) + cg(t_c) + \ldots \end{aligned} . \tag{6}$$

Now consider the Fourier transform of *c*(*t*) defined as,

$$C(\omega) = \int_{-\infty}^{\infty} c(\tau) e^{-i\omega\tau} d\tau \tag{7}$$

This transform is illustrated with a cartoon in Figure 6, which contains a vertical vector representing the input signal, a horizontal vector that represents the amplitude of the Fourier transform, and a 2D array that show the input trace repeated for each discrete value of frequency. Also shown is a sinusoidal representation of the exponential weighting function. The amplitude of the spectrum is the vertical sum of the product of the trace and exponential.

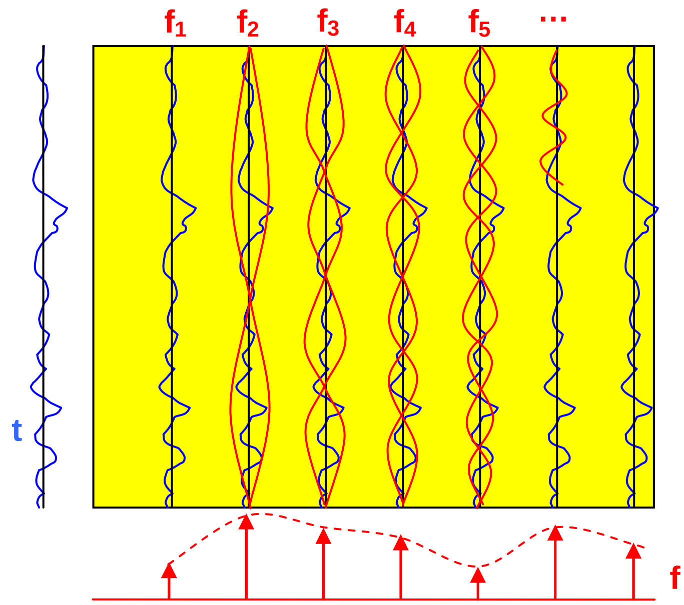


FIG. 6. A cartoon representation of the formal Fourier transform. The left side is a vector representing the input trace. The volume contains the trace repeated at each frequency location, and a sinusoid representing the weighting of the exponential. The bottom vector represents the amplitude spectrum that is formed from the vertical sum.

The Fourier transform of one windowed segment $S(\omega,t)$ where t is the window delay becomes,

$$S(\omega,t)=\int_{t-t_1}^{t+t_1} w(\tau-t)\,g(\tau)\,e^{-i\omega\tau}d\tau\ , \tag{8}$$

in which the integration is limited to $\pm t_1$, and is illustrated in Figure 7. This figure represents the formation of one output spectrum line of the Gabor transform at time t_a. using a triangular window.

Equation (8) is similar to equation (5) except that we now have to contend with the exponential term, i.e.,

$$S(\omega,t)=\int_{t-t_1}^{t+t_1} w(\tau-t)\,g(\tau)\,e^{-i\omega\tau}d\tau \tag{9}$$

We can differentiate $w(t)$ and integrate $g(\tau)e^{-i\omega\tau}$ for-fast-filtering and greatly increase the speed of this formal computation. That means we would pre-compute the two dimensional array containing $g(\tau)$ multiplied by $e^{-i\omega\tau}$ for each frequency component ω, and then integrate each τ array a number of times appropriate for the wavelet. In this example using a triangular wavelet, two integrations are required.

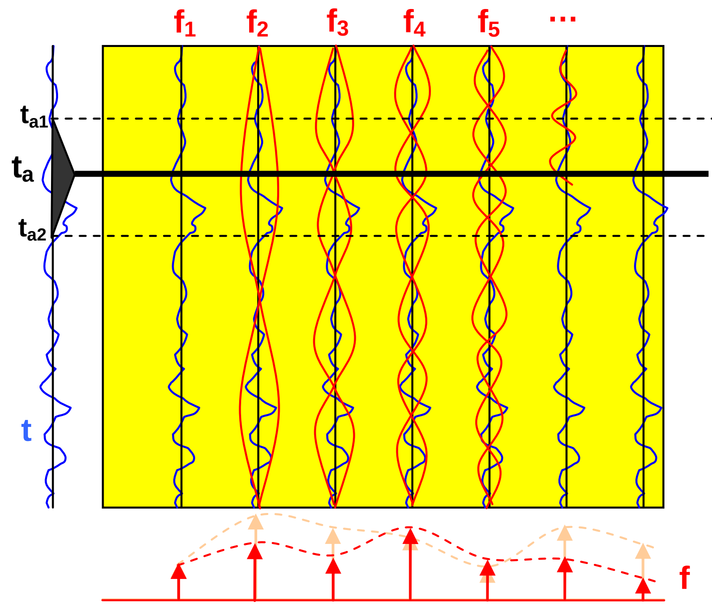


FIG. 7. Representation of one segment of the Gabor transform at t_a. The weighting and sum of a triangular shaped window is limited to the range of times from t_{a1} to t_{a2}.

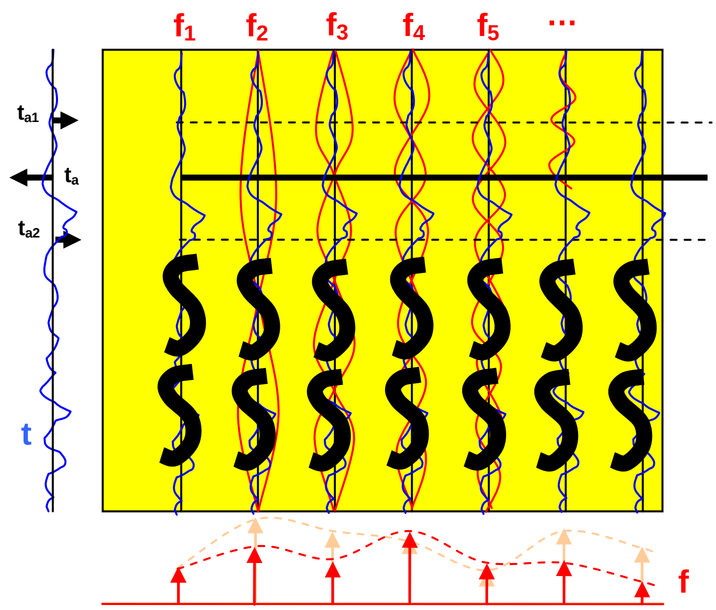


FIG. 8. Representation of one segment of fast-filtering for the Gabor transform at t_a. There are only three summation points at the three delta functions for each frequency. The black integral lines (S) imply that the product of the trace and exponential is integrated twice.

COMMENTS

The size of the front and back end of the quadratic filter window can vary as long as they match the preceding and following windows. This feature will allow time varying filter windows.

CONCLUSIONS

Fast-filtering can significantly increase the computation speed of convolution when the wavelet can be differentiated to delta functions. A quadratic shaped window was presented that is comparable to a Hanning window, but can be differentiated to four delta functions. This window with increments of half the window size will also form a partition of unity, suitable for a Gabor transform.

ACKNOWLEDGEMENTS

I am grateful to Chuck Ursenbach for useful discussions in relating fast-filtering to the Gabor transform.
NSERC and the CREWES sponsors.

REFERENCES

Lumley, D. L., Claerbout, J., F., and Bevc, D., 1994, Anti-aliased Kirchhoff 3-D migration: Expanded Abstracts, SEG National Convention.