Stanford Exploration Project, Report 79, November 16, 1997, pages 279–287

Short Note

Parallel computing exhumes slow Fourier transform in Stolt migration

Patrick Blondel and Francis Muir[1]

keywords: *Fourier transform, interpolation, algorithm*

INTRODUCTION

Stolt migration is a remapping of the data in the frequency-wave-number domain. The mapping function may be an expression of the wave equation either in isotropic media or in anisotropic media (Dellinger et al., 1993; Ecker and Muir, 1993). Both cases require a frequency-domain interpolation that can be a source of many numerical artifacts (Ronen, 1982; Harlan, 1982). Ronen and Harlan proposed frequency-domain interpolators that attenuate these artifacts. In this report, Lin *et al.* (1993) shows how to suppress them with a more appropriate interpolation filter. This filter is derived from the Fourier transform of the data for the irregular range of input frequencies that map into a regular range of output frequencies (Popovici et al., 1993). In other words, applying this interpolation filter in the frequency domain is equivalent to performing a slow Fourier transform for an irregular range of frequencies followed by an inverse fast Fourier transform. The aim of this note is to show that the slow Fourier transform, which is a naturally parallel process, can compete with the fast Fourier transform followed by interpolation on a parallel computer.

FOURIER TRANSFORM FOR IRREGULAR FREQUENCY SAMPLING

At any angular frequency ω, the Fourier transform of a discrete, regular function of time is given by the following summation:

$$F(\omega) = \sum_{n=0}^{N-1} f_n e^{-i\omega n \Delta t} \tag{1}$$

where N is the number of time samples, Δt is the time sampling rate, and f_n is the discrete signal. We notice that equation (1) relates a discrete signal to its continuous

[1]**email:** not available

Fourier transform. This expression can also be interpreted as a dot product between the discrete time function and a vector of powers of exponentials that depend on the particular frequency ω. Then, for a set of frequencies ω_p, the estimates of the Fourier transform result from a matrix multiplication with the discrete time function.

The implementation of this method is straightforward. The calculation of the matrix of exponentials is coded in parallel, for all frequencies ω_p. A built-in function of Fortran 90 performs an efficient vector-matrix multiplication directly yielding the remapped frequency domain.

NON-PARALLEL FEATURES OF INTERPOLATION FILTERS

Two features make the use of interpolation filter difficult to implement in parallel. First, the filters are different for each frequency to interpolate and this frequency varies with the output frequency and wave-number, according to the mapping function. Secondly, although the filter length is fixed, the filter is applied at different locations in the input space. Thus, the interpolation is not a convolution of the input space with a fixed filter, but the application of various weighted masks at irregular locations of this space. Therefore, neither the frequency axis nor the wave-number axis can be used as a parallel dimension.

TIMING ON THE CM5

Both the slow Fourier transform (SFT) and interpolation algorithms have been implemented on the CM5 and their running times compared. The SFT involves a matrix multiplication and therefore is an algorithm in $N \times N$. However, a parallel machine with P processors will perform these computations in a time proportional to $N \times N/P$ with some additional but smaller communication cost.

The serial implementation of the interpolation filter requires a number of computations proportional to $N \times m$, where m is the length of the filters. For large values of N, the interpolation algorithm will prove more efficient than the SFT algorithm. However, because the SFT algorithm runs P times faster on a parallel machine whereas the interpolation algorithm does not improve its speed, what was considered a large number N appears now as a smaller number. Indeed, for a trace length of 2048, the SFT algorithm runs four times as fast as the other:

```
SFT:                111.7 real      100.2 user       0.7 sys
Interpolation:      489.0 real      471.8 user       4.5 sys
```

Moreover, as the trace length decreases, the running time ratio increases in such a way that, for 1024 time samples (a usual trace length), it becomes one to eight:

```
SFT:                 99.7 real      30.0 user       1.7 sys
Interpolation:      447.8 real     241.0 user       6.0 sys
```

The `user` times are more significant than the `real` times which depend a lot on the current load of the machine.

The slow Fourier transform algorithm proves efficient not only for its speed but also for the artifact-free migration that it produces. Figure **??** illustrates this point.

CONCLUSION

A slow Fourier transform step in Stolt migration can be substituted for the usual fast Fourier transform and frequency interpolation step. The implementation on a parallel computer proved time efficient because the slow Fourier transform is fully parallel. This method produces a cleaner image than the application of truncated interpolation filters.

From a more general point of view, we can see that a parallel computer does not only increase the execution speed of a program but also leads the programmer to rethink all the algorithms and even exhume relics.

REFERENCES

Dellinger, J., Muir, F., and Karrenbach, M., 1993, Anelliptic approximations for TI media: Journal of Seismic Exploration, **2**, 23–40.

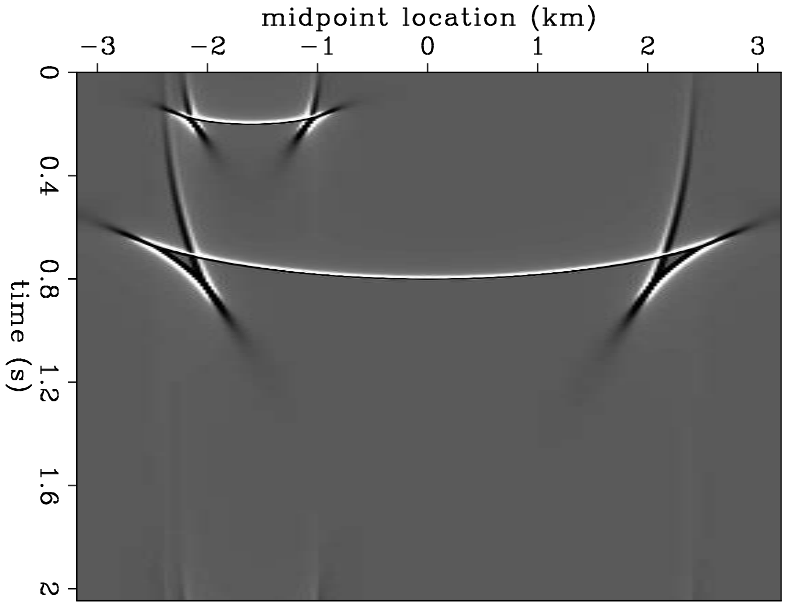
Ecker, C., and Muir, F., 1993, Extending Stolt migration in time to general dispersion relations: SEP–**79**, 249–254.

Harlan, W. S., 1982, Avoiding interpolation artifacts in Stolt migration: SEP–**30**, 103–110.

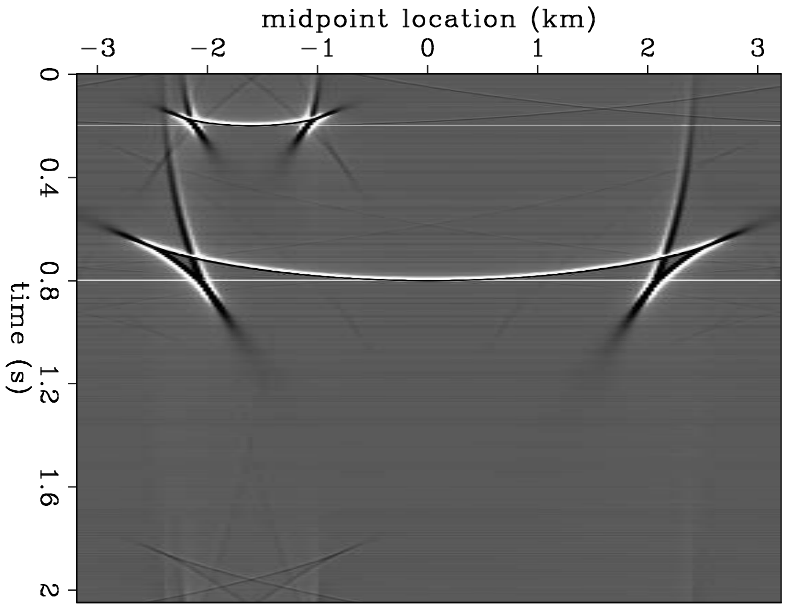
Lin, J., Teng, L., and Muir, F., 1993, Comparison of different interpolation methods for Stolt migration: SEP–**79**, 255–260.

Popovici, A. M., Blondel, P., and Muir, F., 1993, Interpolation in Stolt migration: SEP–**79**, 261–264.

Ronen, J., 1982, Stolt migration interpolation artifacts: SEP–**30**, 95–102.



Stolt migration with slow−FT



Stolt migration with Harlan s filter

Figure 1: Both figures are impulse responses of Stolt migration. The top figure was obtained with the algorithm involving the slow Fourier transform step. The bottom figure was obtained with the algorithm applying Harlan's 10-point filter in the frequency-wave-number domain. Harlan's filter removes only a part of the artifacts caused by frequency interpolation, whereas the slow Fourier transform algorithm yields a clean image. In both cases, a three-point filter (1,2,1) removes the Nyquist noise inherent to the frequency limitation of the discrete Fourier domain in both t and x. Click on the following button to see a movie alternating the two pictures. intime4-compare [ER]