Fast dispersion curves from ambient noise

Eileen Martin

ABSTRACT

To calculate dispersion curves, phase velocity vs. frequency, from ambient seismic noise, many researchers calculate virtual source responses in the time domain, then stack. This algorithm scales as $O(n^2)$, where n is the number of sensors. If the data are regularly sampled in space and time, an existing $O(n \log n)$ algorithm involves an F-K transform of the data cube followed by a transform into velocity vs frequency. I propose a new $O(n)$ algorithm which only takes the Fourier transform of the data in time. The new algorithm is conceptually simple, parallelizes easily, and does not require regular spacing between sensors. I show similar results for dispersion curves resulting from both the $O(n)$ algorithm and the $O(n^2)$ algorithm applied to data collected in a field trial of a trenched distributed acoustic sensing array at the Richmond Field Station. There are fewer opportunities for accumulating numerical error in the new $O(n)$ algorithm, so it yields a sharper image than the $O(n^2)$ algorithm.

INTRODUCTION

Dispersion curves are a simple way to synthesize ambient seismic noise data. Dispersion images roughly show how much surface wave energy is traveling at any given frequency and velocity. Dispersion curves are the result of picking curves along the peaks in frequency and velocity in these images. They sometimes allow us to see not only fundamental surface wave modes, but also higher order modes (de Ridder, 2014).

Let n be the number of sensors in a linear array that passively records seismic data. We are interested in continuous monitoring applications over large regions with many sensors (ideally, n should scale well into the 10,000's). In particular, scaling with the number of sensors of interest when processing ambient seismic noise data collected by a distributed acoustic sensing (DAS) array due to the dense sensor spacing. We assume ambient seismic noise is processed in small chunks (on the order of 1 minute with sampling on the order of 1 kHz).

We can reduce the communication and computation cost of calculating dispersion images (sometimes the peaks of these images are referred to as dispersion curves) from many sensors acting as virtual sources from $O(n^2)$ to $O(n)$ by solving this problem in the Fourier domain instead of the time domain. Following the derivation of this new algorithm, we show dispersion images on a small field dataset collected by a distributed acoustic sensing array.

First, we review some fundamental properties of the Fourier transform that are used in the derivation of the new $O(n)$ algorithm:

- The Fourier transform is linear, so

  $$\mathcal{F}_t \sum_{r=0}^{n} u(x_r, t) = \sum_{r=0}^{n} \mathcal{F}_t u(x_r, t) = \sum_{r=0}^{n} \hat{u}(x_r, \omega).$$

  I use the notation that $\hat{u}(\omega)$ is the Fourier transform of a function $u(t)$ throughout this report.

- A time shift is a frequency shift of the Fourier transform, i.e.

  $$e^{2\pi i\omega p x}\hat{u}(x, \omega) = \mathcal{F}_t u(x, t + px)$$

- A cross-correlation in the time domain is a multiplication in the frequency domain. If we define cross-correlation as

  $$d(x_1, t) \times d(x_2, t) = \int_{-\infty}^{\infty} d^*(x_1, \tau) d(x_2, \tau + t) d\tau,$$

  then the Fourier transform of the cross-correlation is

  $$\mathcal{F}_t \left(d(x_1, t) \times d(x_2, t)\right) = \hat{d}^*(x_1, \omega)\hat{d}(x_2, \omega)$$

  where d^* denotes complex conjugation.

SUMMARY OF EXISTING ALGORITHMS

A common and conceptually simple algorithm to calculate dispersion curves requires calculating source responses in the time domain then slant-stacking, but this method scales as $O(n^2)$. A slightly less intuitive but more efficient $O(n \log n)$ algorithm involves a Fourier transform of the data in both space and time, followed by a transform into slowness vs. frequency. The new $O(n)$ algorithm is inspired by an $O(n^2)$ algorithm, so we describe this existing $O(n^2)$ in detail before moving to the derivation of the $O(n)$ algorithm.

Let $d(x_r, t)$ be an ambient seismic noise trace recorded for some time period at a sensor in position x_r, perhaps with some filtering and preprocessing applied (see Bensen et al. (2007) for general outline of preprocessing). Then let

$$u_s(x_r, t) = d(x_s, t) \times d(x_r, t)$$

be a cross-correlation that is some realization of a random variable with a mean that is the response to a virtual source placed at x_s (this is sometimes loosely referred to as a Green's function). Let p represent slowness (if v is the velocity then $p = 1/v$). One conceptually simple method for calculating the dispersion curve, c_s, from the response to virtual source, s, would be to calculate slant stacks along the responses of all sensors to the virtual source at x_s as is done in Chang (2013):

$$\begin{aligned} c_s(p,t) &= \textstyle\sum_{r=1}^{n} u_s(x_r, t + p(x_r - x_s)) \\ \hat{c}_s(p,\omega) &= \mathcal{F}_t(c_s(p,t)) \end{aligned}$$

Clearly, this approach requires calculating $O(n)$ virtual source responses per virtual source, and must be carried out for $O(n)$ virtual sources to get an understanding of spatial variability across the array.

PROPOSED $O(N)$ FREQUENCY DOMAIN PROCESSING

We will use the basic facts about cross-correlations and the Fourier transform to rewrite the dispersion curve c_s for the response to a virtual source at x_s:

$$\begin{aligned} \hat{c}_s(p,\omega) &= \mathcal{F}_t\left(\sum_{r=1}^{n} u_s(x_r, t + p(x_r - x_s))\right) \\ &= \sum_{r=1}^{n} \mathcal{F}_t(u_s(x_r, t + p(x_r - x_s))) \\ &= \sum_{r=1}^{n} \hat{u_s}(x_r,\omega) e^{2\pi i p (x_r - x_s)\omega} \\ &= \sum_{r=1}^{n} \hat{d}^*(x_s,\omega)\hat{d}(x_r,\omega) e^{2\pi i p (x_r - x_s)\omega} \\ &= \hat{d}^*(x_s,\omega) \sum_{r=1}^{n} \hat{d}(x_r,\omega) e^{2\pi i p (x_r - x_s)\omega} \\ &= \hat{d}^*(x_s,\omega) e^{-2\pi i p x_s \omega} \sum_{r=1}^{n} \hat{d}(x_r,\omega) e^{2\pi i p x_r \omega} \end{aligned}$$

Let $\sigma := \sum_{r=1}^{n} \hat{d}(x_r,\omega) e^{2\pi i p x_r \omega}$. Clearly, σ can be reused in calculating the dispersion curves for all sources $\hat{c}_s(p,\omega)$, and it only takes $O(n)$ calculations. Also, these calculations are highly parallelizable over the number of sensors. Only a single round of communication is required for the reduction to calculate σ. A point-wise multiplication in the frequency domain must be calculated for each source, but that is a constant. Thus, we can calculate dispersion curves for n sensors in $O(n)$ time.

FIELD DATA EXAMPLE

Using just ten minutes of ambient noise data, we were able to extract coherent virtual source responses from a trenched distributed acoustic sensing (DAS) array deployed at Richmond Field Station. The channel length was 1 meter and the guage length was 10 meters. The preprocessing of traces is detailed in the companion paper Martin et al. (2015). Although we demonstrate this algorithm on data collected by a distributed acoustic sensing array, the algorithm can also be applied to traditional point sensors.

Algorithm 1 $O(n)$ algorithm:

```
Given a short chunk of traces $d(x_i, t)$ from receivers at $x_1, x_2, \ldots, x_n$
Initialize $\sigma(p, \omega) = 0$
Calculate $\sigma$, the sum of phase shifted data spectra
for $i = 1, \ldots, n$ do
    Despike and temporal normalization of $d(x_r, t)$
    Bandpass and whitening of $\hat{d}(x_r, \omega) = FFT(d(x_r, t))$
    $\sigma(p, \omega) += \hat{d}(x_r, \omega) e^{2\pi i p x_r \omega}$
end for
for $i = 1, \ldots, n$ do
    If stored, make filtered version of $\hat{d}(x_s, \omega)$ available. If not stored, do any filtering needed to calculate $\hat{d}(x_s, \omega)$
    Dispersion curve for virtual source at $x_s$ is $\hat{c}_s(p, \omega) = \hat{d}^*(x_s, \omega) e^{-2\pi i p x_r \omega} \sigma(p, \omega)$
end for
```

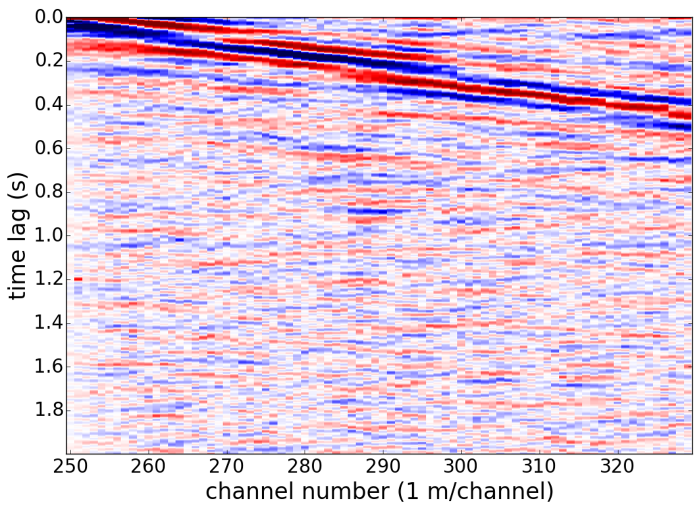


Figure 1: Estimated response to virtual source at channel 250 from 10 minutes of data filtered in the 5 to 50 Hz range. Virtual source response estimates such as this one must be calculated for each virtual source in the $O(n^2)$ algorithm, but can be avoided in the $O(n)$ algorithm. [**ER**]

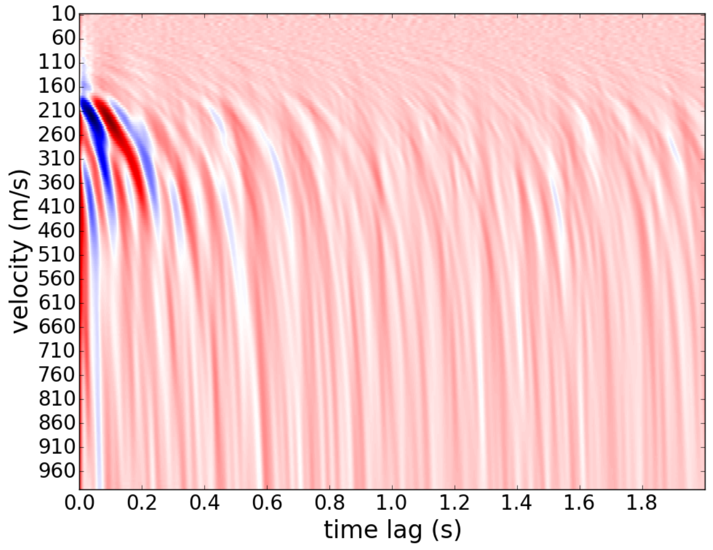


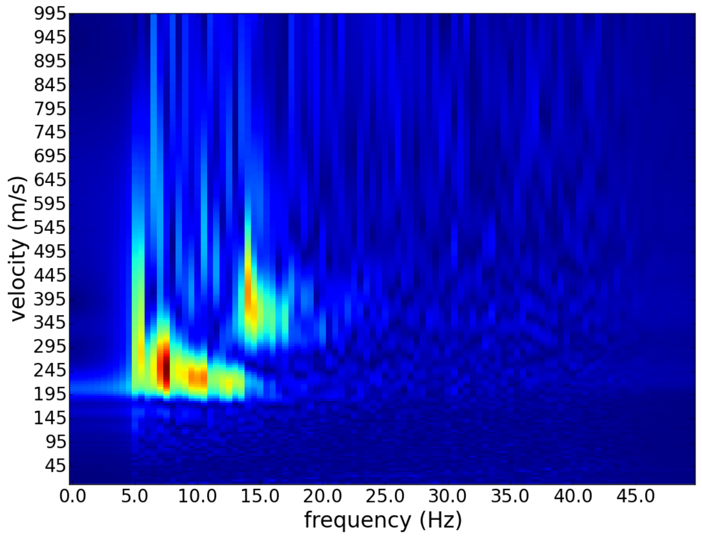
Figure 2: Velocity versus time lag resulting from $\tau - p$ transform of response estimate to virtual source at channel 250. This is a necessary step in the $O(n^2)$ algorithm, but can be avoided in the $O(n)$ algorithm. [**ER**]

In the $O(n^2)$ algorithm, the first step is to calculate virtual source responses. We show these response estimates (folded over zero time lag) in Figure 1 for one virtual source at the end of the array based on only ten minutes of ambient data. The next step in the $O(n^2)$ algorithm is to stack the response estimates over a variety of time lags and velocities, as seen in Figure 2.

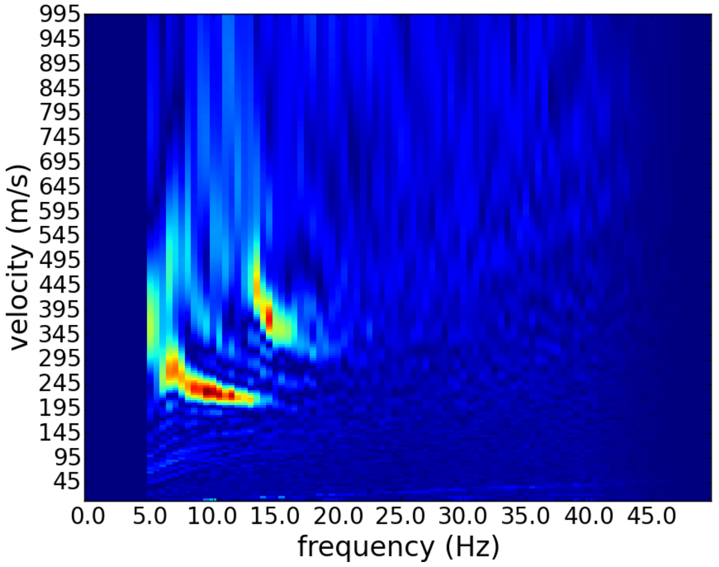
In the end, we are really interested in what the dispersion image looks like. For this data set we can see the dispersion image that only includes a virtual source at channel 250 in Figures 3(a) and 3(b). In the $O(n^2)$ algorithm, we can get this dispersion image by taking the FFT of the $\tau - p$ transform along the τ direction. However, in the $O(n)$ algorithm we can skip calculating the virtual source response estimates and the $\tau - p$ transforms, and directly calculate the dispersion image. Both algorithms yield two strong modes at approximately the same velocity and frequency.

There are some significant differences in these results due to numerical error despite the results being equivalent in theory for exact arithmetic. Switching between the time and frequency domains multiple times in the $O(n^2)$ algorithm leads to small nonzero values below 5 Hz, despite the bandpass filter, which should cut out energy below 5 Hz. However, the new $O(n)$ algorithm respects the bandpass filter. The new $O(n)$ algorithm results in a much sharper dispersion image, most likely because it has fewer Fourier transforms and fewer opportunities for numerical error to accumulate.

There are some features visible in the $O(n)$ algorithm's dispersion image that are not apparent in the $O(n^2)$ algorithm's dispersion image. Between the two most readily apparent modes in the $O(n)$ algorithm's dispersion image, there appears to be another



(a)



(b)

Figure 3: (Top) Velocity versus frequency dispersion image calculated by the $O(n^2)$ algorithm which takes an FFT of the $\tau - p$ transform in the τ direction. (Bottom) Dispersion image calculated by the $O(n)$ algorithm directly from the data spectra, then binned into 1 Hz intervals. Both plots show the two strongest modes at approximately the same velocity and frequency, but the $O(n)$ algorithm results in a much sharper image. [**ER**]

smaller peak for each frequency. It is possible this is another mode, but more data must be included before we can say anything conclusive.

In the $O(n)$ algorithm's dispersion curve, there are some faint lines running from the low velocity & low frequency regime up to the high velocity & high frequency regime. The slope of these lines is approximately 10 meters. More data need to be incorporated before we can draw any more conclusions about whether this is a coherent feature, and what the interpretation of this feature could be.

CONCLUSIONS

We propose a fast algorithm to calculate dispersion images from the data spectra of receivers in a passive seismic survey. This new algorithm scales linearly with the number of sensors. As demonstrated on an ambient noise data set collected by a distributed acoustic sensing array, the new algorithm also yields sharper images. The dispersion images resulting from the new $O(n)$ algorithm are most likely sharper because the simplified algorithm requires fewer operations that may accumulate numerical errors and spread them over the dispersion image space.

ACKNOWLEDGEMENTS

I would like to thank Jason Chang, Bob Clapp, and Sjoerd de Ridder for helpful conversations about existing methods to calculate dispserion curves and this algorithm. I would also like to thank my collaborators Jonathan Ajo-Franklin, Nate Lindsey, Shan Dou, Tom Daley, Barry Freifeld, Michelle Robertson, Craig Ulrich, and Anna Wagner, who collected the data with the support of the US Department of Defense under SERDP grant RC-2437 "Developing Smart Infrastructure for a Changing Arctic Environment Using Distributed Fiber-Optic Sensing Methods". I am supported by the Department of Energy Computational Science Graduate Fellowship, provided under grant number DE-FG02-97ER25308.

REFERENCES

Bensen, G., M. Ritzwoller, M. Barmin, A. Levshin, F. Lin, M. Moschetti, N. Shapiro, and Y. Yang, 2007, Processing seismic ambient noise data to obtain reliable broadband surface wave dispersion measurements: Geophysics Journal International, **169**, 1239–1269.

Chang, J. P., 2013, Velocity dispersion at Long Beach: SEP-Report, **150**, 101–108.

de Ridder, S., 2014, Passive seismic surface-wave interferometry for reservoir-scale imaging: PhD thesis, Stanford University.

Martin, E., J. Ajo-Franklin, N. Lindsey, T. Daley, B. Freifeld, M. Robertson, C. Ulrich, S. Dou, and A. Wagner, 2015, Applying interferometry to ambient seismic noise recorded by a trenched distributed acoustic sensing array: SEP-Report, **158**, 247–254.