PREFACE

This document is fragments from an article by the same authors in SEP report 170 along with results found later.

Multichannel spectral factorization results

Kaiwen Wang and Jon Claerbout

ABSTRACT

Results, finally, and some minor updates about the Unitary transformations.

INTRODUCTION

Good morning everyone. The multichannel structure of Figure 1 arises in diverse physical settings.

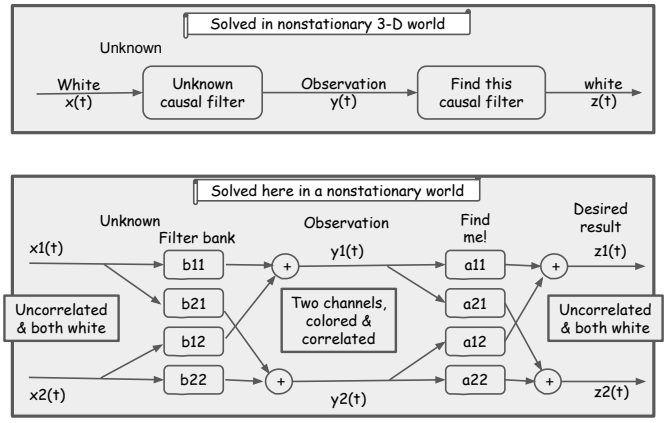

Figure 1: The left side of the diagram hypothesizes nature outside our view. Given the correlated observations $\mathbf{y}$ in the middle we here design the causal process $\mathbf{A}$ to create the outputs $\mathbf{z}$ on the right. We construct $\mathbf{A}$ and the uncorrelated white $\mathbf{z}$. Then we hope $\mathbf{B} \approx \mathbf{A}^{-1}$ and $\mathbf{z}$ approximates the underlying physical world $\mathbf{x}$.

Multichannel code is the matrix generalization of this scalar code:

```
a(1) = 1.0           #               Syntax:   "a+=b" means "a=a+b"
do over time t {     # e(t)     = nonstationary prediction error.
        do tau= 1, na
                       e(t)    +=  a(tau) * y(t-tau+1)      # forward
        do tau= 1, na
                       da(tau) +=  e(t)   * y(t-tau+1)      # adjoint
        da(1) = 0.                                          # constraint
        do tau= 1, na
                       a(tau)  -=  da(tau) * epsilon
        }
```

UNDERSTANDING PEF IN MACHINE LEARNING SETTING

We derive the scalar PEF in machine learning setting and expand it to the multicomponent case.

A scalar case

In a scalar case, we use linear regression to make prediction in a time series $x(t)$. Our assumption is that each data point could be predicted by a linear combination of its previous data points. As shown in Figure 2, $y^{(i)} = x(t+n)$ is predicted by $w^T x^{(i)}$, where $x^{(i)}$ is a preceding data segment $[x(t), x(t+1), x(t+2), \cdots, x(t+n-1)]$ of length n. The loss function L is set to be l_2 norm of the prediction error.

$$L(w) = \|y^{(i)} - \hat{y}^{(i)}\| = (y^{(i)} - w^T x^{(i)})^2 \tag{1}$$

We calculate the gradient of loss function at each time step,

$$\frac{dL(w)}{dw} = -2(y^{(i)} - w^T x^{(i)})x^{(i)} \tag{2}$$

Then we apply gradient descent at each time step and update the parameter vector w using the gradient. α is a chosen learning rate.

$$w = w - \alpha \frac{dL(w)}{dw} \tag{3}$$

By running the update with time, we will get a prediction error vector. The prediction error $y - \hat{y}$ is a time series of length m. The physical meaning is unpredictable part of $x(t)$.

If we assume stationary relationship, w^T will be time invariant. We define cost function J as the mean of loss function at each training data$(x^{(i)}, y^{(i)})$,

$$J(w) = \frac{1}{m}\sum_{i=1}^{m} \|y^{(i)} - \hat{y}^{(i)}\| = \sum_{i=1}^{m} (y^{(i)} - w^T x^{(i)})^2 \tag{4}$$

The gradient of cost function is

$$\frac{dJ(w)}{dw} = -2\sum_{i=1}^{m} (y^{(i)} - w^T x^{(i)})x^{(i)} \tag{5}$$

Applying gradient descent update will give prediction error $y - \hat{y}$.

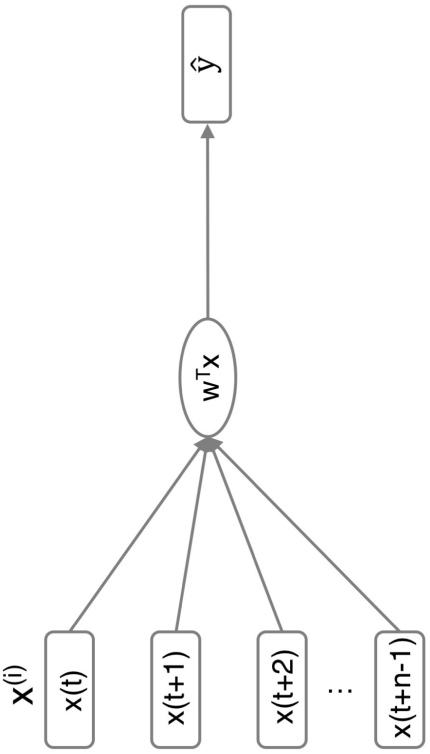

Figure 2: Diagram of scalar PEF. $x^{(i)}$ is a data segment beginning at time t. $\hat{y}$ is the prediction.

multicomponent case

Figure 3 shows how two different sources contribute to recordings at a two-component instrument. The Green's function and source time function are both unknowns. x_1 and x_2 are our observations that are two-component seismograms. Our goal is to perform blind deconvolution to recover the Green's function and source time function.

The network we present is shown in Figure 4. We propose that an inverse rotation should be the first step to remove the dependence between traces. This is an inverse of the projection step on the right of Figure 3. In this step, parameter θ_1 and θ_1 could be either time variant or invariant, and is learned through gradient updates. The forward relationship in Figure 3 is

$$\begin{aligned} x_1 &= q_1 \cos\theta_1 + q_2 \cos\theta_2, \\ x_2 &= q_1 \sin\theta_1 + q_2 \sin\theta_2 \end{aligned} \tag{6}$$

The inverse is then

$$\begin{aligned} q_1 &= \frac{x_1 \sin\theta_2 - x_2 \cos\theta_2}{\cos\theta_1 \sin\theta_2 - \sin\theta_1 \cos\theta_2}, \\ q_2 &= \frac{x_1 \sin\theta_1 - x_2 \cos\theta_1}{\cos\theta_2 \sin\theta_1 - \sin\theta_2 \cos\theta_1} \end{aligned} \tag{7}$$

After the inverse rotation, q_1 and q_2 are inputted into a linear regression step. Now the problem reduces to a scalar problem as we discussed in the previous section. The loss function is defined as

$$L(a_1, a_2, \theta_1, \theta_2) = \|q_1^{(i)} - \hat{q}_1^{(i)}\| + \|q_2^{(i)} - \hat{q}_2^{(i)}\| = (q_1(t+n) - a_1^T q_1^{(i)})^2 + (q_2(t+n) - a_2^T q_2^{(i)})^2 \tag{8}$$

Then we apply gradient descent at each time step and update the parameters $a_1, a_2, \theta_1, \theta_2$ using the gradient. α is a chosen learning rate.

$$\begin{aligned} a_1 &= a_1 - \alpha \frac{dL}{da_1}, \\ a_2 &= a_2 - \alpha \frac{dL}{da_2}, \\ \theta_1 &= \theta_1 - \alpha \frac{dL}{d\theta_1}, \\ \theta_2 &= \theta_2 - \alpha \frac{dL}{d\theta_2} \end{aligned} \tag{9}$$

The gradients are calculated as follows,

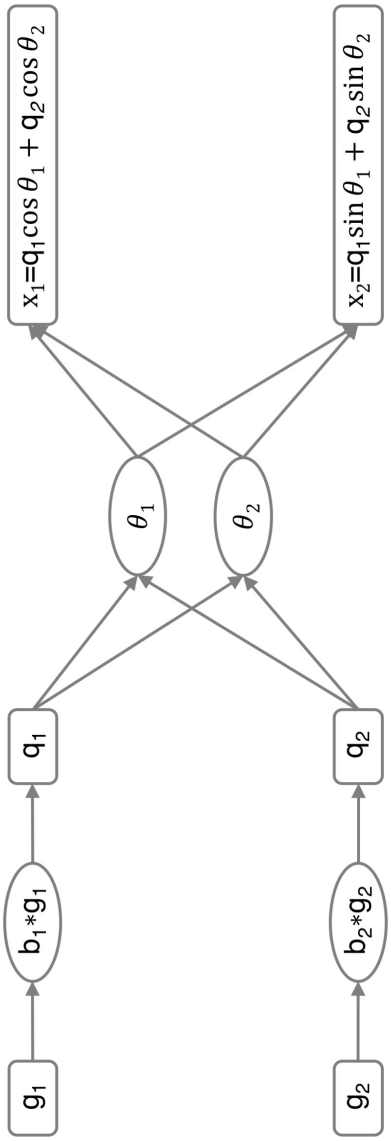

Figure 3: Diagram of how an earthquake generates multicomponent recordings. g_1 and g_2 denote Green's function of the two sources. Convolution of the Green's function and source time function b_1 and b_2 gives q_1 and q_2, as the recordings along the particle motion direction, respectively. Then the two waves q_1 and q_2 are projected to the horizontal and vertical directions.

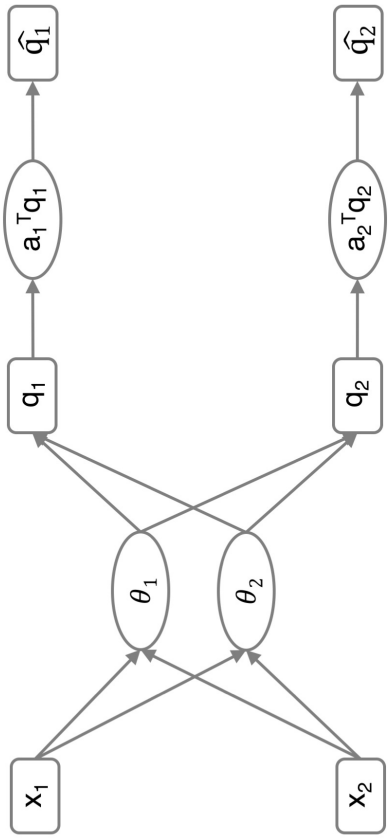

Figure 4: The network for deconvolution. x_1 and x_2 are multicomponent recordings as the input of the network. They first go through an inverse rotation step to rotate back to the directions of their own particle motions. After the inverse rotation, they could be treated as two separate scalar cases, where the two set of parameters a_1 and a_2 are learned independently.

$$
\begin{aligned}
\frac{dL}{da_1} &= -2(q_1(t+n) - a_1^T q_1^{(i)}) q_1^{(i)}, \\
\frac{dL}{da_2} &= -2(q_2(t+n) - a_2^T q_2^{(i)}) q_2^{(i)}, \\
\frac{dL}{d\theta_1} &= \frac{dL}{dq_1^{(i)}}\frac{dq_1^{(i)}}{d\theta_1} + \frac{dL}{dq_2^{(i)}}\frac{dq_2^{(i)}}{d\theta_1} = -2(q_1(t+n) - a_1^T q_1^{(i)}) a_1^T \frac{dq_1^{(i)}}{d\theta_1} - 2(q_2(t+n) - a_2^T q_2^{(i)}) a_2^T \frac{dq_2^{(i)}}{d\theta_1}, \\
\frac{dL}{d\theta_2} &= \frac{dL}{dq_1^{(i)}}\frac{dq_1^{(i)}}{d\theta_2} + \frac{dL}{dq_2^{(i)}}\frac{dq_2^{(i)}}{d\theta_2} = -2(q_1(t+n) - a_1^T q_1^{(i)}) a_1^T \frac{dq_1^{(i)}}{d\theta_2} - 2(q_2(t+n) - a_2^T q_2^{(i)}) a_2^T \frac{dq_2^{(i)}}{d\theta_2}
\end{aligned}
\tag{10}
$$

If we want to separate P and S waves, then we have approximately $\theta_1 + \theta_2 = \frac{\pi}{2}$. Set $\theta_1 = \theta, \theta_2 = \frac{\pi}{2} - \theta$, the forward and backward relationship reduces to

$$
\begin{aligned}
x_1 &= q_1 \cos\theta + q_2 \sin\theta, \\
x_2 &= q_1 \sin\theta + q_2 \cos\theta
\end{aligned}
\tag{11}
$$

The inverse is then

$$
\begin{aligned}
q_1 &= \frac{x_1 \cos\theta - x_2 \sin\theta}{\cos 2\theta}, \\
q_2 &= \frac{x_2 \cos\theta - x_1 \sin\theta}{\cos 2\theta}
\end{aligned}
\tag{12}
$$

Figure 5 is our first test data, synthetic data with a vertical component and a horizontal component. Both a P wave and an S wave are emerging at a fairly steep angle so the vertical is mostly a P is corrupted by a little S, while on the horizontal it is the opposite.

On Figure 6 we notice that the spike estimates become sharper and sharper with time as the filter **A** adapts with time. Oddly, there is some crosstalk on the P channel that does not seem to be diminishing with time. I don't know why that is. Perhaps I should run the program over the panel a zillion times, at each end, capturing the filter and reinstalling it at the beginning.

On Figure 7 the P and S channels contain two signals, the original spikes, and their estimates. We notice that crosstalk nearly diminishes to zero on the P channel, likewise on the S channel.

Unequal cross-over spectra

The method and code of this paper covers a more general model than the examples exhibit. In the real world, the tiny S wave signal seen on the vertical data need

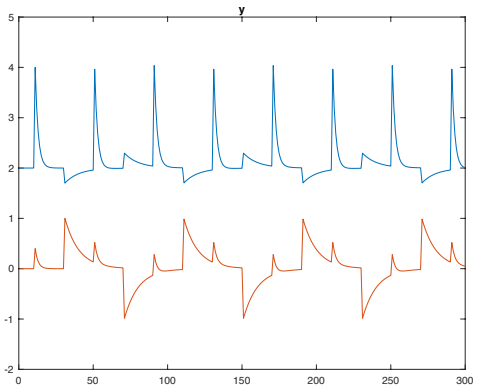

Figure 5: Synthetic data input is vertical and horizontal components. Model is a mix of sharp, unipolar P waves and S waves of lower frequency with alternating polarity. Stronger P waves on the vertical, and stronger S waves on the horizontal.

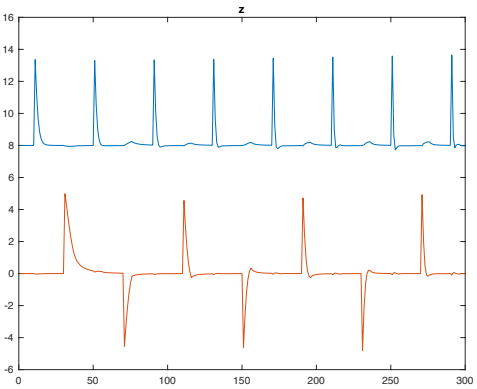

Figure 6: Output results: Deconvolved P wave on vertical component (top), S on horizontal (bottom). Spiking improves with time.

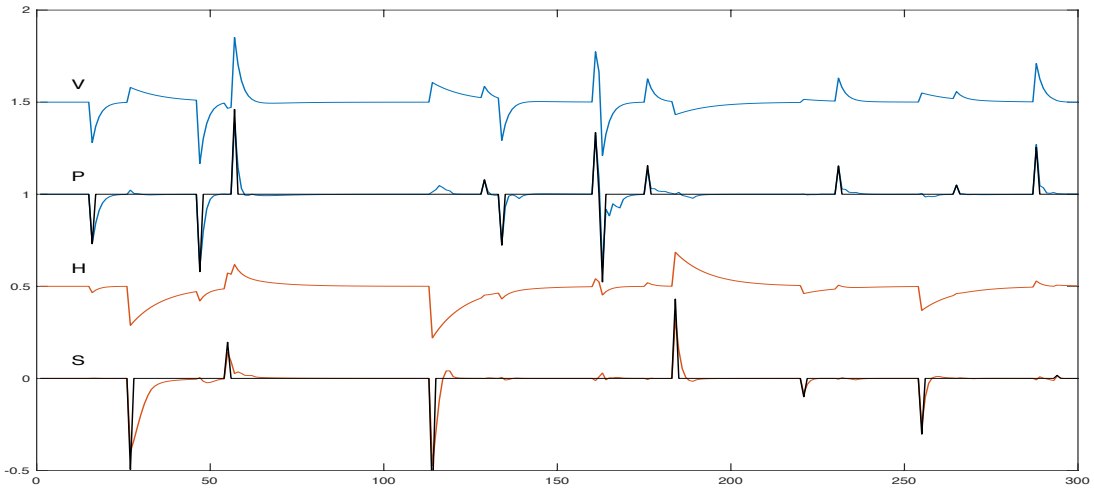

Figure 7: V=vertical, H=horizontal. The traces P and S are overlays of the original impulsive waves and their attempted reconstruction from (V,H). The pulses get sharper with time as the PEFs adapt.

not have the same spectrum as the big S wave seen on the horizontal. This could happen when the physical mechanism of a vertical seismograph differed from that of a vertical. A more dramatic example is an ocean-bottom node measuring both pressure and vertical velocity with an observer trying to extract up- and down-going waves.

Figure 8 has a much denser collection of spikes than Figure 7.

UNITARY TRANSFORMATIONS

Simplifying the unitary transformation

SEP progress report 170 closed before we had any results. We have some here now. We've also been struggling with the unitary transformations. In report 170 I imagined the channel mixing might change with time, but coding that led to complications I am not prepared to pursue now, so my current advice to Kaiwen is to apply the unitary transformation only after all the prediction error and Cholesky work is done. Then find the best time-independent angle θ for component rotation. Here is how: Define entropy by a ratio of an ℓ_1-norm sum to an ℓ_2-norm sum over time. The ℓ_1-norm sum contains terms such as $|z_1(t)| + |z_2(t)|$. Scan entropy as a function of rotation angle, and chose the rotation angle with the minimum entropy.

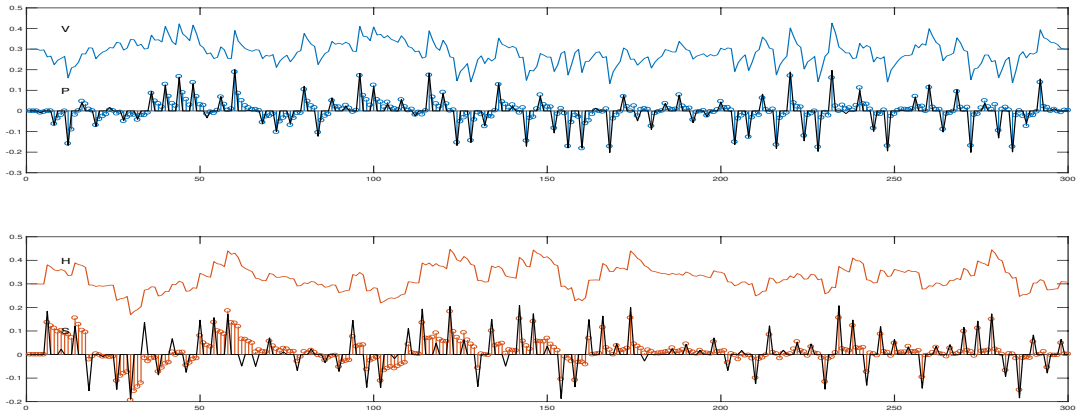

Figure 8: Like figure 7 but denser spikes, a spike every 4 pixels, so three quarters of the dots should be zero valued with the rest at spike tops. Notice the vertical trace (top) being dominated by P waves is higher frequency than the horizontal trace "H" which is dominated by S waves. Results are about the same quality showing that having lots of wavelet overlap causes no real problems. Fitting on the S channel gets much better with time. Fitting on the P channel is so good near the beginning that we hardly notice improvement with time. Preparing this talk I wondered why the fitting is not becoming near perfect. Then it hit me. We are using here a constant $\epsilon = 1/\lambda$. We need a bigger one early and a smaller one later. That is easy to arrange. At the beginning where the filter is steadily growing we could increase ϵ say 25% at each step. Later we could decrease it. The decision would depend on the polarity of $\Delta\mathbf{a} \cdot \Delta\mathbf{a}_{\text{previous}}$.

Channel order and polarity

Although our first synthetic data had the strongest pressure wave on the first channel, our first successful run yielded the pressure wave on the second channel. The channel flip operation is

$$\begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} \tag{13}$$

Now we flip channels when we find the expression $|\mathbf{z}_1 \cdot \mathbf{y}_1| + |\mathbf{z}_2 \cdot \mathbf{y}_2| < |\mathbf{z}_1 \cdot \mathbf{y}_2| + |\mathbf{z}_2 \cdot \mathbf{y}_1|$.

Our initial P-wave result had a flipped polarity. The operation for flipping the polarity for channel 1 is

$$\begin{bmatrix} -1 & 0 \\ 0 & 1 \end{bmatrix} \tag{14}$$

We change the polarity of channel 1 if $(\mathbf{y}_1 \cdot \mathbf{z}_1) < 0$ and likewise for channel 2.

It is easy to show for signals with an identity $\mathbf{I}$ correlation matrix, the channel flip and polarity change operations do not change the $\mathbf{I}$ correlation matrix. We can imagine data for which flip and polarity should change with time, but we have not yet synthesized such data for testing.

Can you separate body waves from surface noises?

Take the main signal to be a fast P or S wave. Take the noise to be slow surface waves, one inflowing, one out. From 2-D 2-component data containing signal (be it P or S) and both incoming and outgoing ground roll we will not be able to extract 3 wave types. Try ignoring the incoming ground roll. Then we might have a good "noise remover," but I don't have the confidence to expect improvement over traditional methods. Today we have bigger fish to fry.

What if original sources are not minimum phase?

We are converting a collection of signals to impulses using a causal method. If the original source waveforms had aspects of delay (non-minimum phase) those aspects will not be invertable by a causal method. (Then we'll see impulses smeared out by the amount of delay contained. Methods to deal with that put non-Guassian demands on the data.) The spiking code should continue to work with the spikes being smeared. But, the code here for `w(ichan,t)` might be unstable.

I cannot reasonably guess when we are likely to encounter instability for `w(ichan,t)`. Stability of inverse PEFs is a well studied topic in the older literature summarized at moderate length in my old book FGDP. Prediction filters will predict growing exponentials, and that by definition is instability. In the stationary world stability can be assured by the Burg method (FGDP again), but I do not know how that might be generalized to the nonstationary world.

My feeling is for small enough epsilon (big enough lambda) we are effectively in the stationary world and should not encounter instability. On the other hand instability may often be triggered by our startup environment. I can only hope the explosive nature of instability is quickly captured by our adaptive setup. We'll see.

As a first experiment we might rerun our first test case with the pressure waves time reversed. We should do that experiment even before we test `w(ichan,t)` production.

Restoring source spectra

White signals are not popular. Before corruption from channel 2, channel 1 had the spectrum of b_{11}. Consider restoring to the white output $\mathbf{z}_1$ the original spectrum P, namely b_{11}. Since $\mathbf{B} = \mathbf{A}^{-1}$ we can deduce b_{11}.

$$\begin{bmatrix} b_{11} & b_{12} \\ b_{21} & b_{22} \end{bmatrix} = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}^{-1} = \frac{1}{a_{11}a_{22} - a_{21}a_{21}} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix} \tag{15}$$

Under the assumption that the crossover filters are less significant than the pass-through filters we may simplify the result for initial trials.

$$b_{11} = a_{22}/(a_{11}a_{22} - a_{21}a_{21}) \approx 1/a_{11} \tag{16}$$
$$b_{22} = a_{11}/(a_{11}a_{22} - a_{21}a_{21}) \approx 1/a_{22} \tag{17}$$

I believe we should test the stability of the simplified approximation before thinking of adding in the complicating terms. We do this by appending some code to our SEP 170 code. The result of polynomial division $w(Z) = e(Z)/A(Z)$ recognizable in the code by $\mathbf{w}_t =$ `w(ichan,t)`. Here is the fragment we will will be appending to:

```
do it= na, nt {

    (  To reduce clutter, 13 lines were removed from here. )

        aa(ic,jc,ia) -= eps * (e(ic,it)/sige(ic)) * ( y(jc, it-ia+1) /sigy(jc))
        }}}
```

Code below goes after code above, before Cholesky and all the rotations and reflections.

```
        w(1,t) = e(1,t)
    do ia=2,na                                      # w1(Z) = e1(Z)/a11(Z)
        w(1,t) -= aa(1,1,ia) * w(1,t-ia+1)

        w(2,t) = e(2,t)
    do ia=2,na                                      # w2(Z) = e2(Z)/a22(Z)
```

```
            w(2,t) -= aa(2,2,ia) * w(2,t-ia+1)

    # In here goes Cholesky and all the unitary transformations.
    # Parameters used in those transformations must also be used on w(ichan,t).

    }     # This is the end of the outer t loop.
```

The parameter found for doing each of these scaling and unitary operations should be used a again on `w(ichan,t)`.

Singular value decomposition

I've been asked how this approach might compare to singular-value decomposition. I don't know. The method here easily flexes into ℓ_1 norm, etc; it has nonstationarity built in; and it has some limitations connected with causality. If you are willing to buy stationarity and throw causality to the wind it is straightforward to form the spectral matrix; then take its square root at each frequency.

The illustrations you see here required $2 \times 4 \times$ `na=5` $\times$ `nt=300` $= 12{,}000$ flops.

Discussion of references

The method and pseudocode is in Claerbout and Wang (2017), the basics of nonstationarity (Fomel et al., 2016), and the GIEE book, (Claerbout, 2014). These references are also available at `http://sep.stanford.edu/sep/jon/`.

REFERENCES

Claerbout, J., 2014, Geophysical image estimation by example: Lulu.com.

Claerbout, J. and K. Wang, 2017, Multichannel data: separating independent causes : SEP-Report, **170**, 189–206.

Fomel, S., J. Claerbout, S. Levin, and R. Sarkar, 2016, Streaming nonstationary prediction error (II): SEP-Report, **163**, 271–277.