Signal Stacking: Phase-Weighted Stack (PWS)

2021-01-16 · signal stacking · 5488 times read

CONTENTS

1 Introduction

In realistic world, seismic signals are contaminated by random noise, but we need signals with high signal-to-noise ratios (SNRs) to conduct the following work of, such as probing the interiors of our planet. Therefore, signal processing experts or seismologist came up with some signal stacking strategies to enhance the SNRs. Linear stacking is usually used to suppress the noise, and generally, we can obtain clear signals using linear stacking. However, we want to accelerate the convergence of signals in ambient noise cross-correlation applications (e.g., Shapiro & Campillo, 2004; Dias et al., 2015) if we only have small data sets. Linear stacking may not work well in such situation. Fortunately, seismologists proposed some novel signal stacking rules to accelerate the convergence of coherent signals. Here, we introduce the phase-weighted stack rule (Schimmel & Paulssen, 1997) and give two examples to present this signal stacking rule.

2 Basic principles

Closely following the study of Schimmel & Paulssen (1997), analytical signal can be obtained from the real signal and its corresponding HIlbert transfom (the imaginary signal). That is

$$\begin{aligned} s(t) &= x(t) + iy(t) \\ &= A(t)e^{i\phi(t)} \end{aligned}, \tag{1}$$

where, $x(t)$ is the real signal, $y(t)$ is the imaginary signal, $A(t)$ is the envelope of the analytical and $e^{i\phi(t)}$ is the phase term.

No amplitude information is involved, the coherence can be defined by

$$c(t) = |\frac{1}{N}\sum_{k=1}^{N} e^{i\phi_k(t)}|. \tag{2}$$

We can obtain phase-weight stacked signal by the multiplication of the real signal and coherence terms,

$$s(t) = \frac{1}{N}\sum_{j=1}^{N} x(t)_j c(t)^{\nu}, \tag{3}$$

specially, this formula means linear stacking when $\nu = 0$.

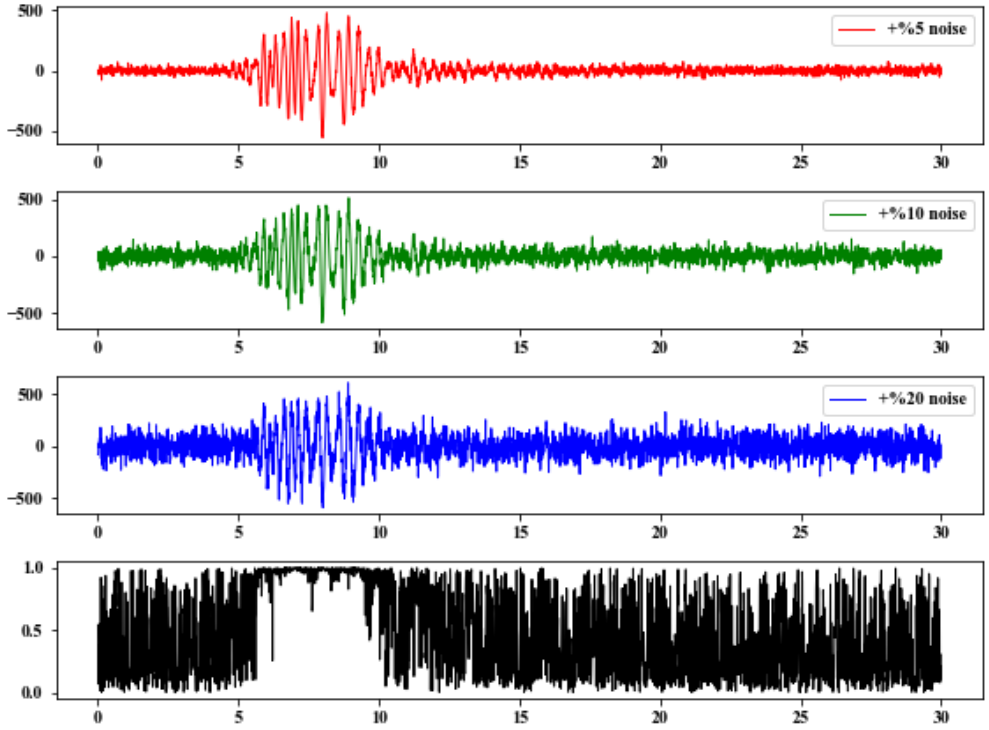
3. A numerical example

Here, we give an example of measuring the coherence of several signals. First, we add different noise levels to the original signal. Using eq. (2), the coherence of these signals can be obtained.

Python

```python
import numpy as np
import matplotlib.pyplot as plt
from obspy import read
from scipy.signal import hilbert

# Gte the coherence of input stream.
def get_coh(st, v, sm=False, sl=20):
    m = len(st)
    n = st[0].stats.npts
    dt = st[0].stats.delta
    t = np.arange(n) * dt
    ht = np.zeros((m, n), dtype=complex)
    c = np.zeros(n)
    for i, tr in enumerate(st):
        ht[i] = hilbert(tr.data)
    pha = ht / abs(ht)
    for i in range(n):
        c[i] = abs( sum(pha[:, i]) )
    # Smooth the coherence if necessary.
    if sm:
        c = np.convolve(c/m, np.ones(sl)/sl, 'same') ** v
    else:
        c = ( c/m ) ** v
    return t, c

v = 2
st = read()
```

```
st.filter('bandpass', freqmin=2.5, freqmax=5, corners=4, zerophase=True)
d1 = st[0].data.copy()
st[0].data = d1 + np.random.randn(len(d1))*d1.max()*0.05
st[1].data = d1 + np.random.randn(len(d1))*d1.max()*0.10
st[2].data = d1 + np.random.randn(len(d1))*d1.max()*0.20

t, c = get_coh(st, v)

plt.figure(figsize=(8, 6))
plt.subplot(411)
plt.plot(t, st[0].data, lw=1, color='r', label='+%5 noise')
plt.legend(loc='upper right')
plt.subplot(412)
plt.plot(t, st[1].data, lw=1, color='g', label='+%10 noise')
plt.legend(loc='upper right')
plt.subplot(413)
plt.plot(t, st[2].data, lw=1, color='b', label='+%20 noise')
plt.legend(loc='upper right')
plt.subplot(414)
plt.plot(t, c, lw=1, color='k')
plt.tight_layout()
plt.show()
```



In the time window of $5-10\ s$, the coherence $c(t)$ is approximately 1 , because this window contains the lower-frequency coherent signals.

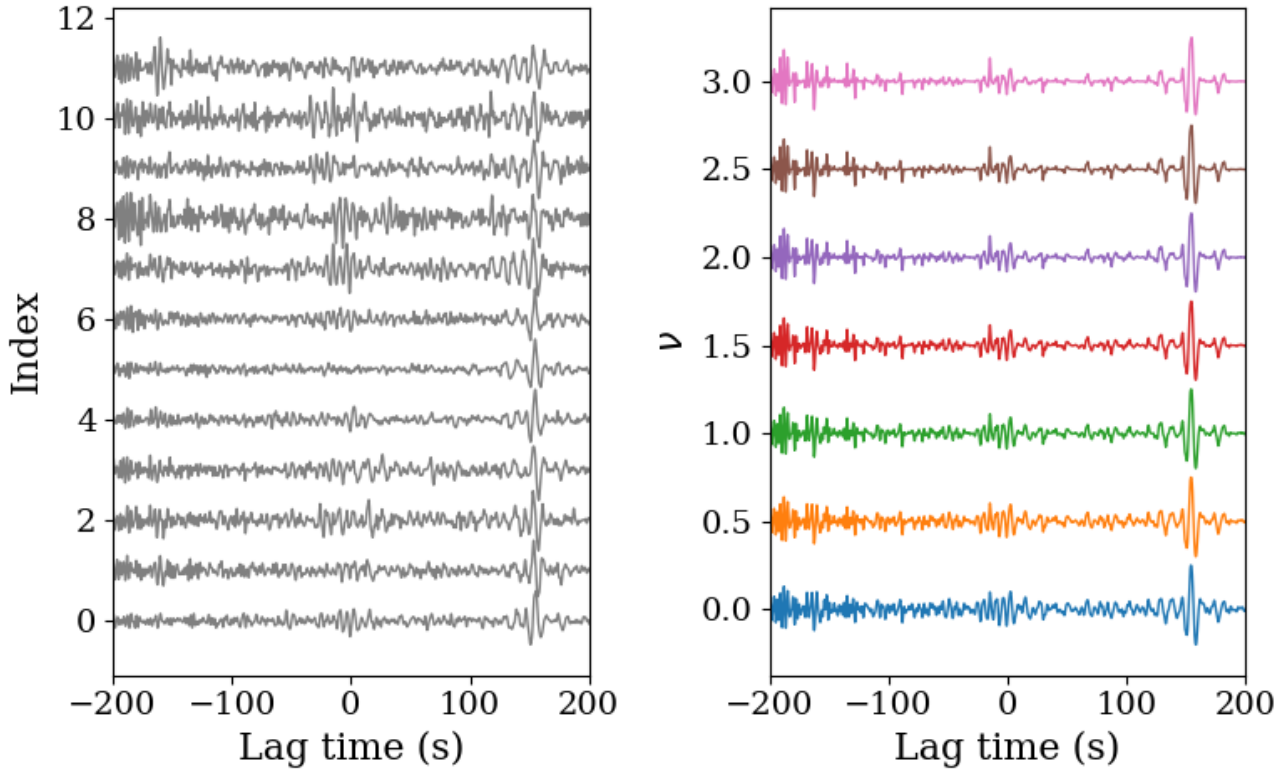
4. A realistic application

Some researchers utilize the PWS to accelerate the covergence of noise cross-correlation time functions (e.g., Dias et al., 2015; Dantas et al., 2018), and we use the PWS strategy to suppress the noise contaminating the noise cross-correlation functions from hurricane events. The PWS is given by the following Python code.

Python

```python
def PWS(st, v, sm=False, sl=15):
    m = len(st)
    n = st[0].stats.npts
    dt = st[0].stats.delta
    t = np.arange(n) * dt
    c = np.zeros(n, dtype=complex)
    for i, tr in enumerate(st):
        h = hilbert(tr.data)
        c += h/abs(h)
    c = abs(c/m)
    if sm:
        operator = np.ones(sl) / sl
        c = np.convolve(c, operator, 'same')
    stc = st.copy()
    stc.stack()
    tr = stc[0]
    tr.data = tr.data*c**v
    return tr
```

Here, we compare the stacked signals from PWS with different power numbers ν and linear stacking.



The power number ν of coherence $c(t)$ are indicated by the numbers on the right panel. Note that the SNRs become higher with the increase of the power number ν. However, one must understand that the waveforms from PWS may change, especially for the larger power number ν.

5 An Extension of PWS: tf-PWS

The stacked waveform using PWS changes much due to the non-linear influence. Here, Schimmel et al. (2011) have developed an extension of the PWS: tf-PWS in time-frequency domain. The steps are

(1) Transform the trace into time-frequency domain with Stockwell transform or S transform;

$$S(\tau, f) = \int_{-\infty}^{\infty} u(t) w(\tau - t, f) e^{-i2\pi f t}, \tag{4}$$

where $w(\tau-t,f)=\frac{|f|}{k\sqrt{2\pi}}e^{-f^2(\tau-t)^2/(2k^2)}$ with $k>0$ is an Gaussian window function. The new coherence function is defined by

$$c(\tau,f)=|\frac{1}{N}\sum_{n=1}^{N}\frac{S_n(\tau,f)e^{i2\pi ft}}{|S_n(\tau,f)|}|^{\nu}. \tag{5}$$

The stacked waveform with tf-PWS in time-frequency domain is

$$S_{tf-PWS}(\tau,f)=c(\tau,f)S_{linear}(\tau,f), \tag{6}$$

where, $S_{linear}(\tau,f)$ is the linear stacking of all traces in time-frequency domain using S transform.

We apply an inverse S transform to $S_{tf-PWS}(\tau,f)$ and obtain the tf-PWS waveform $s(\tau)$.

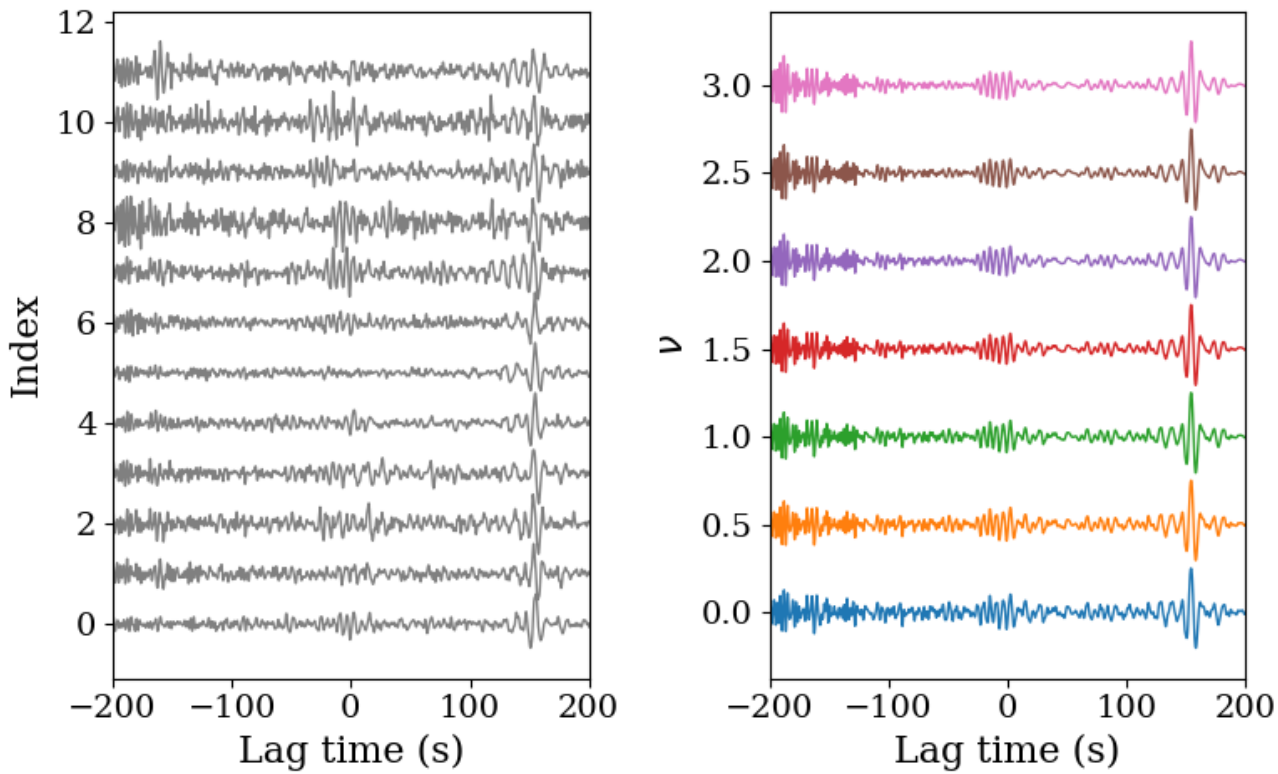
6 An Application of tf-PWS

We use the same noise cross-correlations to check the tf-PWS. You need to install the Python library stoc kwell first if you want to try the following example.

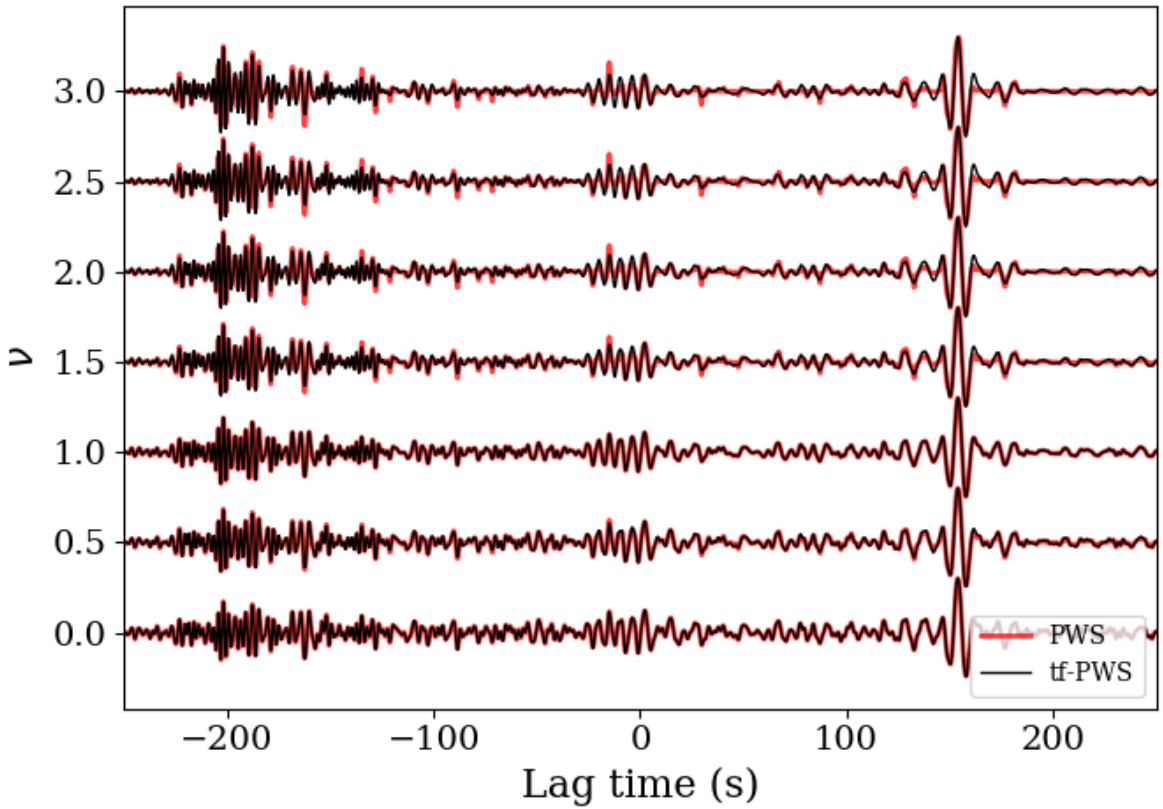
Python

```python
from stockwell import st

def tf_PWS(stream, v, f1, f2):
    dt = stream[0].stats.delta
    b = stream[0].stats.sac.b
    t = np.arange(stream[0].stats.npts) * dt + b
    df = 1 / (stream[0].stats.sac.e-b)
    fn1 = int(f1/df); fn2 = int(f2/df)
    for i, tr in enumerate(stream):
        d = tr.data
        s = st.st(d, fn1, fn2)
        if i < 1:
            f = np.linspace(f1, f2, len(s[:, 0]))
            T, F = np.meshgrid(t, f)
            c = np.zeros_like(s)
        ph = s / abs(s) * np.exp(2j*np.pi*F*T)
        c += ph
    c /= len(stream)
    stc = stream.copy()
    stc.stack()
    ds = st.st(stc[0].data, fn1, fn2)
    pws = st.ist(ds*abs(c)**v, fn1, fn2)
```

```
23      tr.data = pws
24      return tr
```



We can find that the stacked waveforms with tf-PWS do not change much but noise is suppressed. We compare the stacked waveforms with the PWS and tf-PWS, and we can see that the difference is more obvious when the power number ν increases.



References

Dias, R.C., Julià, J. & Schimmel, M. Rayleigh-Wave, Group-Velocity Tomography of the Borborema Province, NE Brazil, from Ambient Seismic Noise. *Pure Appl. Geophys.* **172,** 1429–1449.

Dantas, Odmaksuel Anísio Bezerra, Do Nascimento, A. F. , & Schimmel, M. . (2018). Retrieval of body-wave reflections using ambient noise interferometry using a small-scale experiment. *Pure & Applied Geophysics*.

Schimmel, M., and H. Paulssen (1997), Noise reduction and detection of weak, coherent signals through phase weighted stacks, Geophys. J. Int., 130, 497 – 505.

M. Schimmel, E. Stutzmann2and J. Gallart. Using instantaneous phase coherence for signal extraction

from ambient noise data at a local to a global scale. *Geophysical Journal International*(1), 494-506.

Shapiro, N. M., and M. Campillo (2004). Emergence of broadband Rayleigh waves from correlations of the ambient seismic noise, Geophys. Res. Lett. 31, no. 7, doi: 10.1029/2004GL019491.

Author : Geophydog
LastMod : 2021-01-16

#signal processing #PWS #signal stacking

‹ Ocean and Climate Data Source

Matlab GUI Design: popupmenu ›