Lecture 19:

Recap

We've made it to the final week of the quarter. Last week we went through some details on coherence and transfer functions, and today we're going to wrap up some additional refinements by asking about the role of noise in coherence and transfer functions, and by considering the ever-trendy topic of multi-taper spectra.

Noise and coherence

Let's take a quick look at what noise does to coherence and to the transfer function. Suppose we start with two related signals:

$$y(t) = c_1 x(t) + n(t), \tag{1}$$

where $n(t)$ is noise. Can we determine c_1 and $n(t)$ using the cross-spectrum? Formally, the cross spectrum is:

$$\begin{aligned} S_{XY}(f) &= \frac{\langle X^*(f)Y(f)\rangle}{T} && (2)\\ &= \frac{c_1\langle X^*(f)X(f)\rangle + \langle X^*(f)N(f)\rangle}{T} && (3)\\ &= c_1 S_{XX} && (4)\end{aligned}$$

The squared coherence is of course:

$$\text{Coh}(f)^2 = \gamma_{XY}^2 = \frac{|S_{XY}|^2}{S_{XX}S_{YY}} \tag{5}$$

So what is S_{YY}? We can compute the spectrum of y:

$$\begin{aligned} S_{YY} &= \frac{[c_1^2\langle X^*X\rangle + c_1(\langle X^*N\rangle + \langle XN^*\rangle + \langle N^*N\rangle]}{T} && (6)\\ &= c_1^2 S_{XX} + S_{NN}, && (7)\end{aligned}$$

where X and N are assumed to be uncorrelated. Then the squared coherence is:

$$\begin{aligned} \gamma_{XY}^2(f) &= \frac{c_1^2 S_{XX}(f)^2}{S_{XX}(c_1^2 S_{XX} + S_{NN})} && (8)\\ &= \frac{1}{\left(1 + \frac{S_{NN}(f)}{c_1^2 S_{XX}(f)}\right)} && (9)\end{aligned}$$

The final term in the denominator is a measure of the noise-to-signal ratio. (In our example, we imposed it from the beginning.) So if we knew a lot about the causal relations between our records, we could use the coherence to extract a measure of the noise-to-signal ratio.

Noise and the transfer function

Now let's introduce noise in the Fourier transform domain, which perhaps implies a slightly more complicated relationship between x and y:

$$Y(f) = X(f)H(f) + N(f) \tag{10}$$

If we multiply through by X^*:

$$\langle X^*Y\rangle = \langle X^*X\rangle H(f) + \langle X^*N\rangle \tag{11}$$

Since the signal is uncorrelated with noise, this still gives us

$$H(f) = \frac{G_{xy}(f)}{G_{xx}(f)}, \tag{12}$$

so noise appears to have no impact on the results, but is it all so rosy?

Alternatively, we might imagine that our forcing x is noisy, so that

$$Y(f) = [X(f) + N(f)]\, H(f) \tag{13}$$

In doing this, we assume that the noise associated with X is uncorrelated with the signal Y—in other words, we assume that the response $y(t)$ should be responding to $x(t)$, but we've mismeasured the forcing as $x(t) + n(t)$. Then:

$$\langle (X+N)^*Y\rangle = \langle (X+N)^*(X+N)\rangle H(f) \tag{14}$$

And since y and n are uncorrelated in this formulation,

$$\hat{H}(f) = \frac{G_{xy}(f)}{G_{xx}(f) + G_{nn}(f)}, \tag{15}$$

which is biased low relative to the true response. (But the phase is unbiased.) This is analogous to the noise-related formulation that we looked at for coherence.

We could test all of this with an example:

```
x2=x+randn(5000,1);
xx2=[reshape(x2,500,10) reshape(x2(251:4750),500,9)];
fxx2=fft(xx2.*(hanning(500)*ones(1,19)));
sxx2=abs(fxx2(1:251,:)).^2;
sxy2=conj(fxx2(1:251,:)).*fyy(1:251,:);
transfer2=mean(sxy2,2) ./mean(sxx2,2);

transfer3=mean(sxy2,2) ./mean(syy,2);

semilogx(0:250,abs(transfer),0:250,abs(transfer2),0:250,abs(transfer3))
set(gca,'FontSize',14)
xlabel('frequency','FontSize',14)
ylabel('transfer function','FontSize',14)
legend('H_{xy}','H_{x+n,y}','H_{y,x+n}')
```

Uncertainties for transfer function

Finally we should comment briefly on the uncertainty of the transfer function or gain. Bendat and Piersol go through a detailed discussion of this (see their section 9.2.4) and ultimately write that the relative uncertainty of the transfer function is:

$$\epsilon\left[|\hat{H}_{xy}|\right] = \frac{s.d.\left[\left|\hat{H}_{xy}\right|\right]}{|H_{xy}|} \approx \frac{(1-\gamma_{xy}^2)^{1/2}}{|\gamma_{xy}|\sqrt{2n_d}}, \tag{16}$$

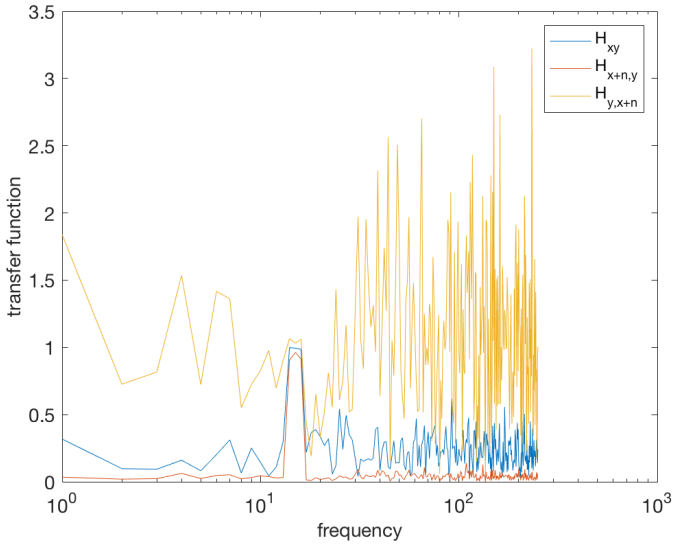

Figure 1: Transfer function for baseline case, vs examples with noise added. Noise substantially modifies the transfer function, and its impact depends on whether it appears in the denominator.

where $s.d.$ indicates the standard deviation. This expression is in fact the same as Bendat and Piersol's expression for the standard deviation of the phase.

Multi-tapers and spectral peaks

Spectra can come in two flavors. Some have distinct single peaks, associated with tides. Some have large-scale structure associated with the general red structure of the ocean. If we want to find exactly the right peaks, then we can try different strategies to what we use when we want to find the general structure.

When we have narrow peaks, they aren't always easy to differentiate, particularly if our sampling is a bit coarse compared with the signals we'd like to detect. Consider the following case of a sinusoidal cycle that might or might not be well sampled, depending how long our instruments survive:

```
time=1:.5:120;
A=2*cos(2*pi*time/30)+cos(2*pi*time/60);
B=A(1:200);
C=[A(1:200) zeros(1,40)];

plot(time,A,time(1:200),B,'LineWidth',3)
set(gca,'FontSize',16)
xlabel('time','FontSize',16)
ylabel('amplitude','FontSize',16)

fA=fft(A);
fB=fft(B);
fC=fft(C);

frequency1=(0:120)/120;
frequency2=(0:100)/100;
```

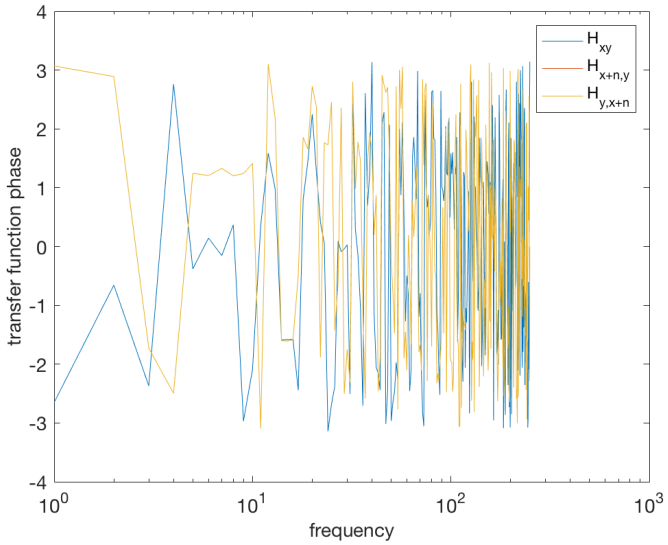

Figure 2: Phase for the transfer function examples shown in Figure 1. Phase depends on the signal, but does not depend on the spectral normalization that appears in the denominator, so the red and yellow lines are the same.

```
loglog(frequency1,abs(fA(1:121)).^2,frequency2,abs(fB(1:101)).^2,...
  frequency1,abs(fC(1:121)).^2,'LineWidth',3)
set(gca,'FontSize',16)
xlabel('frequency','FontSize',16)
ylabel('spectral density','FontSize',16)
legend('full record','truncated record','zero padding')
```

When you look at this example, you might conclude that without perfect sampling of full sinusoidal cycles, we'll never find the correct spectral peaks. In essence this is a windowing problem. When we have narrow peaks, they aren't always easy to differentiate, particularly if our sampling is a bit coarse compared with the signals we'd like to detect. See the slides for an example (from Rob Pinkel.)

If we don't have adequate resolution what are our options?

1. Possibility 1. Pad the short record with zeros to make it as long as we want. Since resolution is $f = 1/(N\Delta t)$. In this case, we'll see the impact of a sinc function bleeding into the frequencies that we'd like to resolve. Clearly this doesn't fully solve our problem.

2. Possibility 2. Obtain a longer record. This will be critical if we really want to resolve our signal.

Even if our record is norminally long enough, we also need to figure out how to optimize our detection of spectral peaks. Earlier we looked at the impact of windows, and examples from the Harris (1978) study showed how much impact a good windowing strategy can have in identifying spectral peaks. (For continuous spectra, windowing approaches work well.)

Formally, you'll recall that we can represent our record length problems using a convolution of our data with a finite width filter:

$$\hat{X}(f_n) = \int_{-\infty}^{\infty} X(f_m) W(f_n - f_m)\, df_m, \tag{17}$$

where

$$W(f) = \frac{\sin(2\pi fT)}{2\pi fT} = \text{sinc}\,(2\pi fT) \tag{18}$$

This means that the spectrum is essentially convolved with $W(2\pi fT)^2$. But as we noted earlier, we can switch from a boxcar window to a triangle window or something a bit more Gaussian than that and cut down on the sidelobes in our window to obtain a cleaner spectrum, although we have to widen the central peak of the window in the frequency domain, which means de-emphaizing the beginning and end of the data series.

What if we want to improve our resolution. Consider the example of a record equivalent to

$$x = 100\cos(2\pi 20.5/10001) + 80\cos(2\pi 30.4/100001) + 100\cos(2\pi 40.8/100001) + 10\cos(2\pi 50.3/100001) \tag{19}$$

The quality of our spectral estimate will depend on the length of our record. (Why is that? The resolution is the lowest resolved frequency.) So what can we do to improve resolution. One strategy would be to pad our record with zeros to make it as long we want. That buys us something, but it gives us plenty of spectral ringing.

If we want to optimize resolution, we can try a multitaper approach. (See for example Ghil et al, Reviews of Geophysics, 2001). In a multitaper approach, we replace our single window with a set of tapers. The tapers are designed to minimize spectral leakage, and they are referred to as "discrete prolate spheroidal sequences" or "Slepian" tapers (after Slepian, who studied them). Tapers are what we've been calling windows—they pre-multiply the data, Fourier transforms are computed, and then the spectrum is computed as a weighted sum of all of the squared Fourier transforms. This effectively averages over an ensemble of windows to minimize variance. This is very effective for extracting narrow peaks that would otherwise be undetectable. Matlab has a multi-taper method package ('pmtm'), but if you really want this to work, you probably want to dig into the guts of the algorithm a bit further.

Here's the Matlab example, modified slightly to make a longer record:

```
fs = 1000;                  % Sampling frequency
t = (0:3*fs)/fs;              % One second worth of samples
A = [1 2];                  % Sinusoid amplitudes
f = [150;140];              % Sinusoid frequencies

xn = A*sin(2*pi*f*t) + 0.1*randn(size(t));
pmtm(xn,4,[],fs)
```

This produces an impressive two spectral peaks. Of course this example isn't too tricky. Here's what we get if we take the same data and split them into 6 non-overlapping segments, even with no windowing or detrending:

```
fa=fft(reshape(xn(1:3000),500,6));
semilogy(mean(abs(fa(1:250,:)').^2))
```

These are reassuringly similar results.