Lecture 11:

Recap

Last time we looked at methods for computing spectra by filtering in the frequency domain, and we looked at the use of the covariance to compute spectra. We had a few loose ends on the covariance approach, so we'll revisit that now.

Using the auto-covariance to think about spectra (Part 2).

We went through the equations to compute a spectrum from the autocovariance, but that can seem a bit mind-bending, so we really need to look at a couple of examples.

Using the auto-covariance to compute spectra requires averaging, and the example I went through last time had a little problem, but this one will get it right. First let's take a moment to think about the autocovariance. If I compute the autocovariance of white noise, that's

$$R_{xx}(\tau) = \int_{-\infty}^{\infty} x(t)x(\tau+t)dt = \begin{cases} 0 & \text{for } \tau \neq 0 \\ 1 & \text{for } \tau = 0 \end{cases} \tag{1}$$

This is true, because white noise is uncorrelated except at zero lag.

Alternatively, if I consider red noise, then the noise will be correlated from point to point, and the autocovariance will have a bit of spread. We can test this out:

```
% define red data with autoregressive process
a=randn(1000,1);
b(1)=a(1);
for i=2:1000
  b(i)=b(i-1)+a(i);
end

BB=xcorr(b,b)/max(xcorr(b,b));
BB_unbiased=xcorr(b,b,'unbiased')/max(xcorr(b,b,'unbiased'));
plot(-999:999,[BB' BB_unbiased'],'LineWidth',3)
```

Notice that we consider both the 'biased' and the "unbiased" estimator. There are arguments for either choice. The difference depends on how we normalize our discrete autocovariance. In the unbiased case, we're computing

$$R(\tau)_{unbiased} = \frac{1}{N-m}\sum_{n=1}^{N-m} x(t_n)x(t_{n+m}). \tag{2}$$

In the biased case, we change how we normalize:

$$R(\tau)_{biased} = \frac{1}{N}\sum_{n=1}^{N} x(t_n)x(t_{n+m}), \tag{3}$$

which means that as the number of values we consider becomes smaller, we constrain the magnitude of the autocovariance by continuing to divide by N. Emery and Thomson note that the biased estimator acts like a triangle window.

So what happens if we use this to estimate our spectrum

1. Suppose we start with a big matrix of white noise, and we compute the autocovariance for each column of our matrix, then Fourier transform, and use these to compute a spectrum. We'll end up doing something along these lines:

```
A=randn(1000,100);
B(1,:)=A(1,:);
for i=2:1000
 B(i,:)=B(i-1,:)+A(i,:);
end

for i=1:100
 AcA(:,i)=xcov(A(:,i),A(:,i),'unbiased');  % autocovariance for
 BcB(:,i)=xcov(B(:,i),B(:,i),'unbiased');  % autocovariance for
end
fAcA=fft(AcA(500:1500,:)); % Fourier transform of autocovariance
fBcB=fft(BcB(500:1500,:)); % Fourier transform of autocovariance
frequency=(0:500)/1000;
figure(1)
subplot(1,2,1)
loglog(frequency,abs(mean(fAcA(1:501,:),2)),'LineWidth',3)
set(gca,'FontSize',16)
xlabel('Frequency (cycles per data point)','FontSize',16)
ylabel('Spectral energy','FontSize',16)
subplot(1,2,2)
loglog(frequency,abs(mean(fBcB(1:501,:),2)),'LineWidth',3)
set(gca,'FontSize',16)
xlabel('Frequency (cycles per data point)','FontSize',16)
ylabel('Spectral energy','FontSize',16)
```

2. Alternatively, we could average all of the autocovariances, and then Fourier transform:

```
mean_AcA=mean(AcA,2);
mean_BcB=mean(BcB,2);
fmean_AcA=fft(mean_AcA(500:1500));
fmean_BcB=fft(mean_BcB(500:1500));
subplot(1,2,1)
hold on
loglog(frequency,abs(fmean_AcA(1:501,:))*1.1,'r','LineWidth',3)
subplot(1,2,2)
hold on
loglog(frequency,abs(fmean_BcB(1:501,:))*1.1,'r','LineWidth',3)
legend('average of FFTs of many autocovariances',...
     'FFT of averaged autocovariance (scaled by 1.1)')
```

Examples are illustrated in Figure 1.

You shouldn't be surprised that averaging before or after the FFT leads to the same results, since averaging has no impact on the FFT. But this might give you an idea of how you can take advantage of the autocovariance to compute spectra from gappy data.

All of this means that we could compute spectra without needing to chunk our data and compute lots of ffts, provided that we had a good estimate of the autocovariance. In the days before the development of the FFT, the autocovariance was a natural pathway to determining the spectrum, since it was clean and easy to compute. And now, with modern computing, you might not feel like there's any need to take advantage of the FFT anymore. If you can obtain the best possible estimate of the autocovariance, by whatever means necessary, then you should be able to compute one FFT and obtain reasonable estimate of the spectrum, without concern for data gaps or computational speed.

Getting the units right

So far we've been a tiny bit sloppy about the units in our spectra, but we do need to make this right. Our fundamental principle is that we want Parseval's theorem to work. But this gets a tiny bit messy when we average multiple frequencies. Still the basic rule of Parseval's theorem is not that the sum of the squares equals the sum of the squared Fourier coefficients, but rather that the integrated variance equals the integral under the spectrum.

In the discrete Fourier transform, we have:

$$\sum_{n=0}^{N-1} x_n^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X_k|^2 \tag{4}$$

and we often work with this. But in continuos form, we had a form more like this:

$$\int_{-\infty}^{\infty} x^2(t)dt = \frac{1}{2\pi} \int_{-\infty}^{\infty} |X(\sigma)|^2 \, d\sigma = \int_{-\infty}^{\infty} |X(2\pi f)|^2 \, df \tag{5}$$

where $\sigma = 2\pi f$, and f is frequency in cycles per unit time. If we want this integral form to work for our real data, then we have to be a bit careful with our normalizations. We're going to want the area under the curve in our spectrum to be equal to the total variance integrated over time. So if total integrated variance is

$$variance = \sum_{n=0}^{N-1} x_n^2 \, \Delta t \tag{6}$$

then the integrated spectrum should be

$$variance = \sum_{k=0}^{N-1} |X_k|^2 \, \Delta f \tag{7}$$

where $\Delta f = 1/(N\Delta t)$. This implies that we might imagine normalizing our spectra to have:

$$\sum_{n=0}^{N-1} x_n^2 \Delta t = \frac{\Delta t}{N\Delta f} \sum_{k=0}^{N-1} |X_k|^2 \Delta f = (\Delta t)^2 \sum_{k=0}^{N-1} |X_k|^2 \Delta f \tag{8}$$

Or maybe this makes for units that aren't easily compared, so we could normalize our spectra to represent the average energy per unit time in the time domain, and adjust the frequency domain accordingly:

$$\begin{aligned}
\frac{1}{T}\sum_{n=0}^{N-1} x_n^2 \Delta t &= \frac{1}{N\Delta t}\sum_{n=0}^{N-1} x_n^2 \Delta t && (9)\\
&= \frac{(\Delta t)^2}{N\Delta t}\sum_{k=0}^{N-1} |X_k|^2 \Delta f && (10)\\
&= \frac{(\Delta t)}{N}\sum_{k=0}^{N-1} |X_k|^2 \Delta f && (11)\\
&= \frac{1}{N frac N N\Delta t}\sum_{k=0}^{N-1} |X_k|^2 \Delta f && (12)\\
&= \frac{1}{N2\frac{N/2}{N\Delta t}}\sum_{k=0}^{(N-1)/2} 2|X_k|^2 \Delta f && (13)\\
&= \frac{1}{N2 f_{Nyquist}}\sum_{k=0}^{(N-1)/2} 2|X_k|^2 \Delta f && (14)
\end{aligned}$$

so we could divide our spectrum by twice the Nyquist frequency to have energy in units appropriate for comparing if we wanted to have our integrals match.

This isn't always the way we think about this, but it serves as our reminder that we should think about the units of our spectrum. What we know is that integral of our spectrum over a cdertain frequency range should give a measure of the signal variance:

$$\text{variance in a band} = \int_{f-\Delta f/2}^{f+\Delta f/2} |X(f)|^2\, df \qquad (15)$$

So if we expand this out, this implies that the units of $|X(f)|^2$ should be equivalent to variance divided by frequency, so it's our reminder that we'll label the y-axis units as the squared units of x divided by frequency, with a normalization to account for the units of time in our data.

Variance preserving spectra

The problem with plotting our spectra in log-log space is that we can't actually learn much from the area under the curve. That might not bother you, but sometimes that visual representation might seem helpful. Consider a spectrum $S_{xx}(f)$ derived from our Fourier amplitudes $X(f)$. In log space, the area under our curves is a pseudo-variance s_*^2:

$$s_*^2 = \int_{f-\Delta f/2}^{f+\Delta f/2} \log(S_{xx}(f))\, df \qquad (16)$$

If we want to plot something that is more directly representative of variance, we can try this:

$$s^2 = \int_{f-\Delta f/2}^{f+\Delta f/2} S_{xx}(f)\, df = \int_{f-\Delta f/2}^{f+\Delta f/2} f S_{xx}(f) d[\log(f)] \qquad (17)$$

This means that instead of plotting log$S$ vs log(f), we could plot $fS_{xx}(f)$ in linear space vs f in log space. This is especially useful for features that have strong peaks not exactly at the lowest frequency.

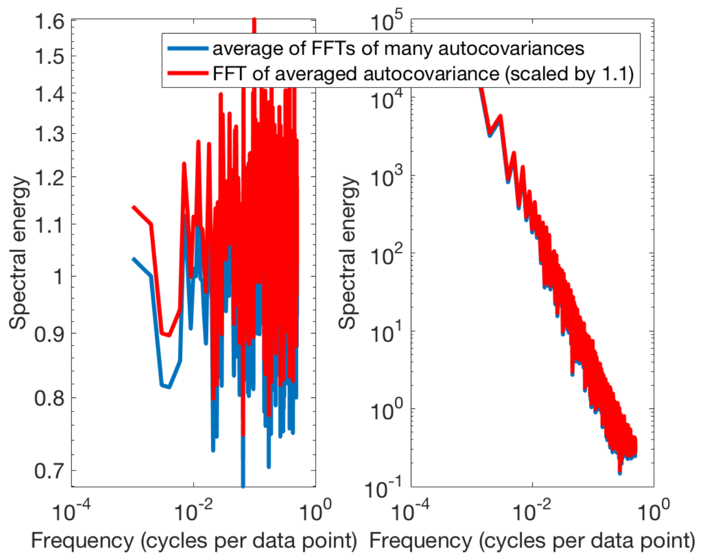

Figure 1: Spectra for white noise (left) and red noise (right), computed by Fourier transforming 100 realizations of the autocovariance function (blue), or by Fourier transforming a smoothed autocovariance function computed from 100 realizations of the data (red). The red line is scaled upward by a factor of 10 to allow both lines to be seen.