Lecture 5:

Reading: Bendat and Piersol, Ch. 4.5.1, 11.1, 11.2

Recap

Last time we looked at least-squares fitting. We derived the formula for a least-squares fit and showed that we could find a linear trend and a sinusoidal variation. Then we asked how many functions we could fit to our data. In the limit of maximum fitting, if we have N data points, we can fit N functions to our data. We noted that a set of N sines and cosines (so $N/2$ pairs of sine and cosine) ranging from frequency 0 to frequency $N/2$ cycles per N points (the Nyquist frequency) will complete fit our data. This is because the set of sines and cosines are orthogonal to each other (even for discrete data) and they fully span the data.

Least-squares fits and misfit

You'll recall that last time we considered a least-squares fit of the form

$$\mathbf{Ax} + \mathbf{n} = \mathbf{y}, \tag{1}$$

and then we took this to the limit in which $\mathbf{A}$ was an $N \times N$ square matrix, so we were solving for as many unknowns in $\mathbf{x}$ as we had data in $\mathbf{y}$. In this case, what happens to our noise $\mathbf{n}$? By using N orthogonal functions, we obtain a perfect fit and the noise is zero. That's convenient, but it loses any information that we might have had about uncertainties in our data. If we made noisy measurements, then it should not be possible to obtain a perfect fit.

Formally we can evaluate this using a χ^2 test. (Yet another use of χ^2.) If I believe my *a priori* uncertainties in my data are σ, then I expect that my misfit should roughly match my uncertainty so I can define a weighted summed misfit:

$$\chi^2 = \sum_{i=1}^{N} \left(\frac{y_i - \sum_{j=1}^{M} a_{i,j} x_j}{\sigma_i} \right)^2 . \tag{2}$$

Here we're summing the squared misfit of each row in our matrix equation, weighted by our uncertainty. If our error bars make sense, then this should yield about N, reduced a bit by the number of functions were fitting. So we expect that χ^2 will be about $N - M$, which is the number of degrees of freedom. Formally we can decide if our fit is too good to be true by evaluating χ^2 using the incomplete gamma function, which yields the probability that this should occur by chance:

```
p=gammainc(chi_squared/2,nu/2)
```

This tells you the probability of finding a fit this good purely by chance. If p is near 1, it can tell us that our fit is too good to be true. If p is too small, it can tell us that our fit isn't properly representing the data. In this case, if χ^2 is zero, p will be 1, warning us that we're over-fitting our data.

The Fourier Transform

So our least-squares fit of N data to N sinusoids was clearly too good to be true, but we're not doing fitting here, so we're going to proceed along this line of reasoning anyway. Our goal is to rerepresent all of the information in our data by projecting our data onto a different basis set. In this case we'll take the projection, warts and all, and we want to make sure we don't lose any information.

So we want to represent our data via sines and cosines:

$$x(t) = \frac{a_0}{2} + \sum_{q=1}^{\infty} \left(a_q \cos(2\pi q f_1 t) + b_q \sin(2\pi q f_1 t)\right), \tag{3}$$

where $f_q = 1/T_p$, and T_p is the duration of the record (following Bendat and Piersol). Formally we should assume that the data are periodic over the period T_p. We find the coefficients a and b by projecting our data onto the appropriate sines and cosines:

$$a_q = \frac{1}{T_p} \int_0^{T_p} x(t) \cos(2\pi q f_1 t)\, dt \tag{4}$$

and

$$b_q = \frac{1}{T_p} \int_0^{T_p} x(t) \sin(2\pi q f_1 t)\, dt \tag{5}$$

solved for $q = 0, 1, 2, ...$

It's not much fun to drag around these cosines and sines, so it's useful to recall that

$$\cos\theta = \frac{\exp(i\theta) + \exp(-i\theta)}{2} \tag{6}$$

$$\sin\theta = \frac{\exp(i\theta) - \exp(-i\theta)}{2i}, \tag{7}$$

which means that we could redo this in terms of $e^{i\theta}$ and $e^{-i\theta}$. In other words, we can represent our data as:

$$x(t) = \sum_{q=-\infty}^{\infty} \left[\hat{a}_q \exp(i 2\pi q f_1 t)\right] = \sum_{q=-\infty}^{\infty} \left[\hat{a}_q \exp(i \sigma_q t)\right] \tag{8}$$

where $\sigma_q = 2\pi q/T$, and $\hat{a}_q$ represents a complex Fourier coefficient. If we solved for our coefficients for cosine and sine, then we can easily convert them to find the complex coefficients $\hat{a}_q$ for $\exp(i\sigma_q t)$ and $\exp(-i\sigma_q t)$. Consider :

$$a\cos\theta + b\sin\theta = \frac{a}{2}(e^{i\theta} + e^{-i\theta}) + \frac{b}{2i}(e^{i\theta} - e^{-i\theta}) \tag{9}$$

$$= \frac{a - ib}{2} e^{i\theta} + \frac{a + ib}{2} e^{-i\theta}. \tag{10}$$

This tells us some important things. The coefficients for $e^{i\theta}$ and $e^{i\theta}$ are complex conjugates. And there's a simple relationship between the sine and cosine coefficients and the $e^{\pm i\theta}$ coefficients. Instead of computing $\sum_{j=1}^{N} a_j \cos(\omega_j t)$ and $\sum_{j=1}^{N} b_j \sin(\omega_j t)$, we can instead find $\sum_{j=1}^{N} \hat{a}_j \exp(i\omega_j t)$ and then use the real and imaginary parts to represent the cosine and sine components. This gives us a quick shorthand for representing our results as sines and cosines.

Fourier transform in continuous form

Bracewell's nice book on the Fourier transform refers to the data as $f(x)$ and its Fourier transform as $F(s)$, where x could be interpreted as time, for example, and s as frequency. Here's I've rewritten to follow the same notation as above. In continuous form, the Fourier transform of $x(t)$ is $X(\omega)$ (where $\omega = q f_1$), and the process can be inverted to recover $x(t)$.

$$X(\omega) = \int_{-\infty}^{\infty} x(t) e^{-i2\pi t\omega}\, dt \tag{11}$$

$$x(t) = \int_{-\infty}^{\infty} X(\omega) e^{i2\pi t\omega}\, d\omega \tag{12}$$

(following Bracewell).

But there are lots of alternate definitions in the literature:

$$X(\sigma) = \int_{-\infty}^{\infty} x(t)e^{-it\sigma}\,dt \tag{13}$$

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\sigma)e^{it\sigma}\,d\sigma \tag{14}$$

or

$$X(\sigma) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} x(t)e^{-it\sigma}\,dt \tag{15}$$

$$x(t) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} F(s)e^{it\sigma}\,d\sigma \tag{16}$$

So we always have to be careful about our syntax.

The same questions about choices of notation apply in the discrete form that we consider when we analyze data. And we can get ourselves really confused. So we have to keep in mind one rule: we don't get to create energy. That means that we need to have the same total variance in our data set in the time domain as we have in the frequency domain. This is Parseval's theorem, and we'll return to it.

One of the glories of the Fourier transform is that we can take all of these projections and make them extremely efficient through the Fast Fourier Transform (FFT). In principle, FFT's are most efficient if you compute them for records that are a power of 2 in length, so 64 or 128 or 256 points for example. But modern FFTs are fast even if your data set doesn't have 2^n elements. Moreover, a year doesn't have 2^n days, so trying to force a data record to conform to a length of 2^n can suppress some of the natural periodicity.

Mathematically the Matlab definitions look like this:

$$X_k = \sum_{n=1}^{N} x_n \exp(-i2\pi(k-1)(n-1)/N), \tag{17}$$

where frequency labels k and data labels n go from 1 to N. Here capital letters are used to denote Fourier transformed variables. Matlab computes this using the command "fft".

The inverse of the Fourier transform is computed using "ifft" and is defined to be:

$$x_n = \frac{1}{N}\sum_{k=1}^{N} X_k \exp(i2\pi(k-1)(n-1)/N) \tag{18}$$

In Matlab the Fourier transform and inverse Fourier transform become:

```
f=fft(x)
x_new=ifft(f)
```

To make Parseval's theorem work, the variance of our data has to equal the variance of the Fourier transform. Thus we'll want to compare:

```
sum(x.^2)
sum(abs(f).^2)
f'*f
sum(f.*conj(f))
```

They don't quite agree, so we'll see that we should divide the Fourier transform by N, the number of data points.

What do we gain by Fourier transforming our data?

We live life in the time domain, so it's sometimes hard to think about the world as seen in the frequency domain. While linear trends aren't well represented by the Fourier transform, the Fourier tranform is particularly effective for representing sinusoidal oscillations. Solar radiation that warms the Earth varies on a 365.25 day cycle with the seasons, and on a 24 hour cycle, with the rising and setting of the sun. Ocean tides vary at semidiurnal (12.4 hour) and diurnal frequencies (as well as being modulated on fortnightly and monthly intervals.) In fact Thus if you look at data from a tide gauge, you see oscillatory fluctuations at a variety of different frequencies, as shown in the slides. If we solve for the tidal amplitudes, we find for example:

Symbol	Frequency (cpd)	Amplitude (cm)	Greenwich Epoch
O1	0.92953571	8.91	217
P1	0.99726209	5.32	224
K1	1.00273791	16.12	225
M2	1.93227361	9.97	354
S2	2.00000000	6.45	357

These complex Fourier coefficients might seem confusing, but they give us a lot of information about our data, allowing us, for example to tell whether there is more energy at frequency σ_j compared with frequency σ_l. The Fourier coefficients are complex so this comparison might seem confusing, but we'll just examine the squared magnitudes of the coefficients: $|a_j|^2$.

Of course, if we knew the frequency exactly, we could just do a least-squares fit, but often we aren't exactly sure of the frequencies in question—there might be energy spread over a broad range of frequencies, and the Fourier tranform provides us with a way to examine our data in terms of oscillatory signals.