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

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

We noted that the misfit is defined as a squared quantity so should follow the χ^2 statistic. (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 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.

What happens in the limit when we fit N data points with N columns in matrix $\mathbf{A}$? The matrix $\mathbf{A}$ is an $N \times N$ square matrix, and we are solving for as many unknowns in $\mathbf{x}$ as we had data in $\mathbf{y}$. In this case, if χ^2 is zero, p will be 1, warning us that we're over-fitting our data. 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, we might not have any reason to expect a perfect fit, but we'll have one anyway.

Orthogonality and Sines and Cosines

Last time we talked about the importance of having independent columns in our matrix $\mathbf{A}$ and noted that sines and cosines are particularly useful since they are orthogonal. Let's work through this a little more carefully.

Consider a record of duration T with N data points. I can imagine squeezing into the period T, one sine wave, or two, or three, or four. How do I tell if my records are orthogonal?

$$\begin{aligned}\int_0^T \sin\left(\frac{2\pi nt}{T}\right)\sin\left(\frac{2\pi mt}{T}\right)\,dt &= \frac{1}{2}\int_0^T \cos\left(\frac{2\pi(n-m)t}{T}\right)-\cos\left(\frac{2\pi(n+m)t}{T}\right)\,dt \\ &= \frac{1}{2}\frac{T}{2\pi}\left[\frac{\sin\left(\frac{2\pi(n-m)t}{T}\right)}{n-m}-\frac{\sin\left(\frac{2\pi(n+m)t}{T}\right)}{(n+m)}\right]\Bigg|_0^T \\ &= 0 \qquad \text{unless } n=m\end{aligned}$$

By extension the same applies for two cosines, or a sine multiplied by a cosine.

This means that I can set up a matrix of sines and cosines, in which every column will be orthogonal to every other column.

$$A=\begin{bmatrix} 1 & \cos(\omega t) & \sin(\omega t) & \cos(2\omega t) & \sin(2\omega t) & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \end{bmatrix}, \tag{3}$$

where $\omega = 2\pi/T$ and T is the total duration of the data record. The dot product of any two columns i and j of $\mathbf{A}$ is zero if $i \neq j$. If I have data at N evenly spaced time increments, $t_1, t_2, ...t_N$, then this orthogonality property holds for all frequencies from ω through $N\omega/2$. Since I have a sine and cosine at each frequency (up to frequency $N\omega/2$ where sine might be zero at all points in time), this means that I can define a total of N independent orthogonal columns in $\mathbf{A}$.

What happens if we want to fit the frequency $\omega/2$? In this case, it won't be orthogonal to my other functions over the range of this data. For example, between 0 and T, $\sin(\omega/2)$ varies from 0 to 1 to 0 and is always positive, meaning that it will be positively correlated with a constant. In fact, sines and cosines with frequencies that are ω multiplied by integers rangings from 0 to $N/2$ make a complete set that spans all space, and there are no additional N-element vectors that I can add to $\mathbf{A}$ that would also be orthogonal to all other columns of $\mathbf{A}$.

The orthogonality of the columns of $\mathbf{A}$ is really important. It means that my solution for x_1 is completely independent of my solution for x_2. Here are some results for a set of 128 random numbers, b.

$$\hat{b} = -0.0629 - 0.0620\cos(\omega t) - 0.1339\sin(\omega t) \tag{4}$$

$$\hat{b} = -0.0629 - 0.0960\cos(2\omega t) + 0.1117\sin(2\omega t) \tag{5}$$

$$\hat{b} = -0.0629 - 0.0620\cos(\omega t) - 0.1339\sin(\omega t) - 0.0960\cos(2\omega t) + 0.1117\sin(2\omega t), \tag{6}$$

where $\hat{b}$ is our fitted approximation to b. You can see that the amplitudes of $\cos(\omega t)$ and $\cos(2\omega t)$ are the same regardless of whether $\mathbf{A}$ contains 3 columns or 5 columns.

If we take a time series of N elements, then the lowest frequency that we can resolve is 1 cycle per N elements, so $\cos(2\pi i/N)$, where our counter i runs from 1 to N (or from 0 to $N-1$). We can find two coefficients for this: one for the $\cos$ component and one for the $\sin$ component.

Actually, maybe a better way to think about this is that the lowest frequency we can resolve is $\cos(0i/N) = 1$, which is a constant and represents the mean. Since $\sin(0) = 0$, there is only a cosine component for frequency 0.

At any rate, after considering 1 cycle per N points, the next frequency we can resolve that will actually be fully orthogonal is 2 cycles per N points. We can keep counter upward: 3 cycles per N points, 4 cycles per N points, and so forth. All of these are guaranteed to be orthogonal over our domain of N points.

What is the maximum number of cycles that we can resolve in N points? One possibility would be that the maximum is N cycles per N points. That would require a full sinusoidal oscillation squeezed between data element 1 and data element 2. But if you think about it, we wouldn't expect to have enough information to determine the amplitude of a sine wave that had to squeeze itself between consecutive observations. Moreover if N cycles per N points were the maximum, this would mean that we'd be solving for $2N$ coefficients with only N data points. Clearly that would require more information than we have available.

So the maximum must be less than N. There are two ways to think about this. One way is to think about the information content of our data. At the most basic level, if I have N elements in my data matrix, then at best, I can hope to have N independent equations, so I can fit N coefficients as an exactly determined problem. So I can solve for frequencies from 0 cycles per N points to $N/2$ cycles per N points, with 2 coefficients for all but the end members of my record. The other way to think of this is to observe that you'll need at least 2 data points per cycle to determine even a minimum amount of information about the sine or cosine amplitudes of your data. So the highest frequency that you can possibly detect of $N/2$ cycles per N data points. This is the *Nyquist frequency* and it is one of the most central concepts in time series analysis.

And the strategy of least-squares fitting all possible frequencies that can be resolved represents the *discrete Fourier transform*. It's a slow and inefficient Fourier transform, but it is the essence of this class and it will be the building block for everything we do in the remainder of the quarter.

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:

$$x(t) = \frac{a_0}{2} + \sum_{q=1}^{\infty} \left(a_q \cos(2\pi q f_1 t) + b_q \sin(2\pi q f_1 t) \right), \tag{7}$$

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:

$$a_q = \frac{1}{T_p} \int_0^{T_p} x(t) \cos(2\pi q f_1 t)\, dt \tag{8}$$

and

$$b_q = \frac{1}{T_p} \int_0^{T_p} x(t) \sin(2\pi q f_1 t)\, dt \tag{9}$$

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

$$\cos\theta = \frac{\exp(i\theta) + \exp(-i\theta)}{2} \tag{10}$$

$$\sin\theta = \frac{\exp(i\theta) - \exp(-i\theta)}{2i}, \tag{11}$$

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:

$$x(t) = \sum_{q=-\infty}^{\infty} [\hat{a}_q \exp(i2\pi q f_1 t)] = \sum_{q=-\infty}^{\infty} [\hat{a}_q \exp(i\sigma_q t)] \tag{12}$$

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 :

$$\begin{aligned} a\cos\theta + b\sin\theta &= \frac{a}{2}(e^{i\theta} + e^{-i\theta}) + \frac{b}{2i}(e^{i\theta} - e^{-i\theta}) &(13)\\ &= \frac{a-ib}{2}e^{i\theta} + \frac{a+ib}{2}e^{-i\theta}. &(14)\end{aligned}$$

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 $\omega = qf_1$), and the process can be inverted to recover $x(t)$.

$$\begin{aligned} X(\omega) &= \int_{-\infty}^{\infty} x(t) e^{-i2\pi t\omega}\, dt &(15)\\ x(t) &= \int_{-\infty}^{\infty} X(\omega) e^{i2\pi t\omega}\, d\omega &(16)\end{aligned}$$

(following Bracewell).

But there are lots of alternate definitions in the literature:

$$\begin{aligned} X(\sigma) &= \int_{-\infty}^{\infty} x(t) e^{-it\sigma}\, dt &(17)\\ x(t) &= \frac{1}{2\pi}\int_{-\infty}^{\infty} X(\sigma) e^{it\sigma}\, d\sigma &(18)\end{aligned}$$

or

$$\begin{aligned} X(\sigma) &= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} x(t) e^{-it\sigma}\, dt &(19)\\ x(t) &= \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{\infty} X(\sigma) e^{it\sigma}\, d\sigma &(20)\end{aligned}$$

So we always have to be careful about our syntax.

Given the vast array of notation, we're going to try very hard to stick to Bendat and Piersol's forms:

$$\begin{aligned} X(f) &= \int_{-\infty}^{\infty} x(t) e^{-i2\pi ft}\, dt &(21)\\ x(t) &= \int_{-\infty}^{\infty} X(f) e^{i2\pi ft}\, df &(22)\end{aligned}$$

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:

$$X_k = \sum_{n=1}^{N} x_n \exp(-i2\pi(k-1)(n-1)/N), \tag{23}$$

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:

$$x_n = \frac{1}{N} \sum_{k=1}^{N} X_k \exp(i2\pi(k-1)(n-1)/N) \tag{24}$$

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:

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

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

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.