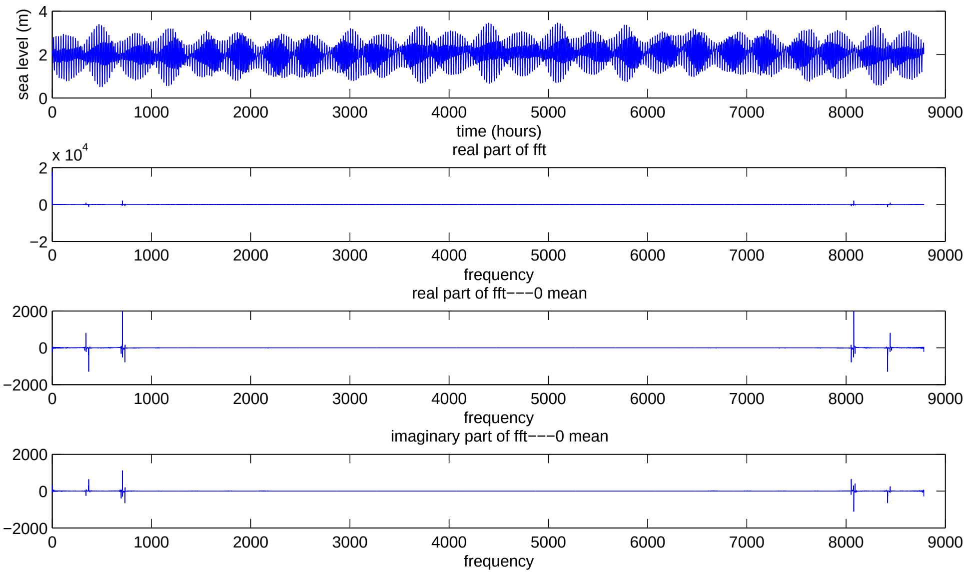

Figure 1: (top) Time series of sea level from Scripps Pier in La Jolla. (second panel) Real part of Fourier transform of time series. (third panel) Real part of Fourier transform of demeaned time series. (bottom) Imaginary part of Fourier transorm of time series.

Example: Fourier Transform of a Data Record

The top panel of Figure 1 shows a year-long time series (from 2000) of sea level measured at the La Jolla Pier. The second panel shows the real part of the Fourier transform. That's

```
plot(real(fft(data)));
```

You can't see much in this case, because frequency 0 contains the mean, which is large compared with the variability.

So that we don't have to look only at the mean, the third panel is the real part of the Fourier transform of the data record.

```
plot(real(fft(data-mean(data))));
```

It is symmetric about the middle, with a couple spikes at either end that correspond roughly to one cycle per day and two cycles per day—the dominant tidal frequencies. The bottom panel is the imaginary part of the Fourier transform.

```
plot(imag(fft(data-mean(data))));
```

or

```
plot(imag(fft(data)));
```

It is anti-symmetric about the middle.

The Fourier transform in this raw form isn't very informative, so we'll do a little more work to digest our results. For each frequency we can compute an amplitude. If I have data T that vary at only one frequency ω_k, I can represent them as $T = a_k \cos(\omega_k t) + b_k \sin(\omega_k t)$. Now suppose that I can have positive and negative frequencies. That means that I can have $T = a_k \cos(\omega_k t) + b_k \sin(\omega_k t) + c_k \cos(-\omega_k t) + d_k \sin(-\omega_k t) = (a_k + c_k) \cos(\omega_k t) + (b_k - d_k) \sin(\omega_k t)$. The average value of T is zero, since sine and cosine both have zero means—that's not so interesting. The variance of the signal is the mean squared amplitude:

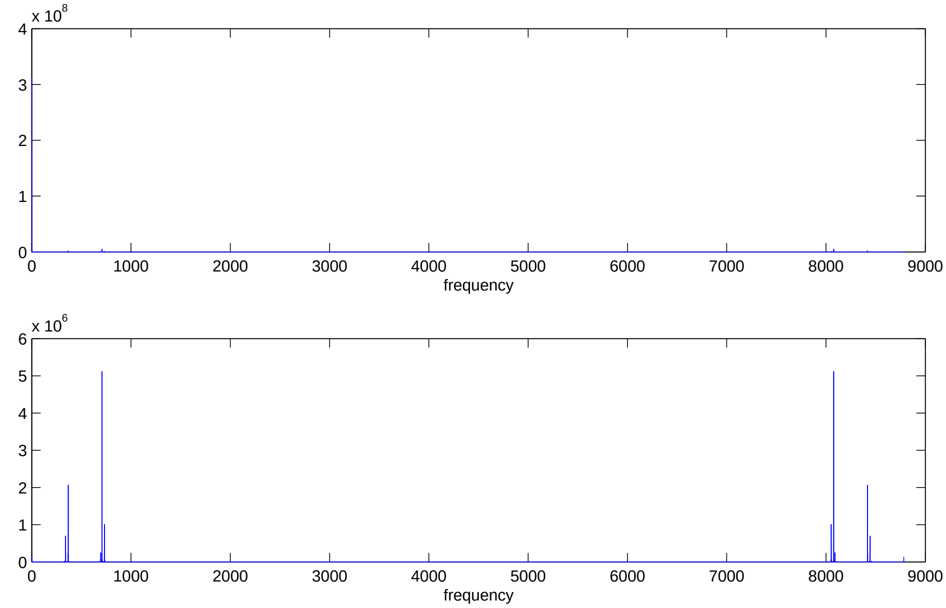

Figure 2: (top) Amplitude of Fourier transform as a function of frequency with mean included. (bottom) Amplitude of demeaned time series.

$\sum_{i=1}^{N}(a_k + c_k)^2 \cos^2(\omega_k t_i) + (b_k - d_k)^2 \sin^2(\omega_k t_i)$, where the terms containing products of cosine and sine disappear since sine and cosine are uncorrelated. In the same way that the integral $1/(2\pi)\int_0^{2\pi} \cos^2(t)\, dt = 1/2$, the sum $1/N \sum_{i=1}^{N} \cos^2(\omega t_i) = 1/2$. Thus the total variance of T is $((a_k + c_k)^2 + (b_k - d_k)^2)/2$.

Even if T represents a superposition of a lot of different sinusoidal variations, since sines and cosines are orthogonal (if they're resolved in our time domain), the presence of other frequencies won't influence the total variance at the frequency ω_k. That means we can use the same rule to find the variance of T corresponding to each specific frequency.

We're not done yet. We still have to look at the relationship between a_k, b_k, c_k and d_k. Clearly the amplitudes associated with positive and negative frequencies $\pm\omega_k$ aren't really independent. In our Fourier transform, we found that the amplitude of positive frequencies was the complex conjugate of the amplitude of negative frequencies. That means that if positive frequencies ω_k are represented by $a_k + ib_k$, then negative frequencies $-\omega_k$ have amplitudes $a_k - ib_k = c_k + id_k$, so $a_k = c_k$ and $b_k = -d_k$. Therefore the total variance of T is $((2a_k)^2 + (2b_k)^2)/2 = 2(a_k^2 + b_k^2)$, which is twice the squared magnitude of the amplitude that we find by Fourier transforming.

Figure 2 shows the amplitude of the Fourier transform $\Re(X_k)^2 + \Im(X_k)^2$ as a function of frequency.

```
plot(abs(fft(data)).^2);
```

The record is strongly dominated by the mean (at the lowest frequency). To avoid seeing nothing but the mean, we can subtract the mean, as shown in the bottom panel of Figure 2.

```
plot(abs(fft(data-mean(data))).^2);
```

but this result is still completely dominated by a couple of energetic frequencies.

A more helpful strategy is to plot the amplitudes on a semilog plot, as shown in Figure 3.

```
semilogy(abs(fft(data)).^2);
```

The resulting plot is completely symmetric. Frequencies indexed from 1 to $N/2$ can be thought of as positive frequencies. Frequencies from $N/2 + 1$ to N are equivalent to negative frequencies. Matlab's notation is a

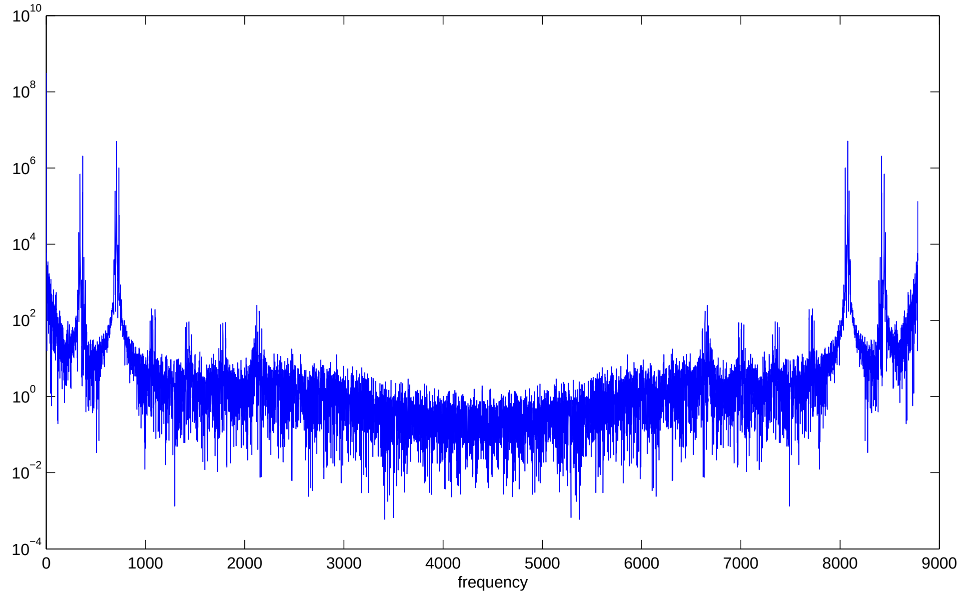

Figure 3: Amplitude of Fourier transform as a function of frequency with mean included, plotted with log scale for y-axis.

little confusing since index 1 corresponds to frequency 0, and index $N/2+1$ corresponds to the maximum resolved frequency.

Parseval's Theorem

One of the most important features of the Fourier transform is that it rerepresents data in frequency space, but it doesn't alter the overall "power" or "energy". Thus in continuous form:

$$\int_{-\infty}^{\infty} |x(t)|^2 \, dt = \int_{-\infty}^{\infty} |X(f)|^2 \, df. \tag{1}$$

and in discrete form:

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

Thus in our 8-point example above, $\sum y^2 = 28$ and $\sum |Y|^2 = 224 = 28 \times 8$.

For complex numbers $|Y_k| = \sqrt{\Re(Y_k)^2 + \Im(Y_k)^2}$. We can compute this as $|Y_k|^2 = Y_k Y_k^*$, where Y_k^* is the complex conjugate of Y_k. In Matlab, the transpose is also the complex conjugate, so if Y is a column vector, then we can compute $|Y|^2 = Y' * Y$ or you can specify that $|Y|$ = abs(Y).

Spectra

With Parseval's theorem in mind, we can interpret our squared amplitudes as a measure of the total "energy" in our time series. To plot energy as a function of frequency, we consider only frequencies between 1 and $N/2$, since frequency indices between $N/2+1$ and N simply repeat the same information. However, to make sure that we account for all of the variability in the system, we need to multiply our Fourier transform amplitudes by 2, except at zero frequency. In Matlab we compute:

```
N=length(data);
s=abs(fft(data)).^2;
spectra(1)=s(1);
spectra(2:N/2+1)=2*s(2:N/2+1);
```

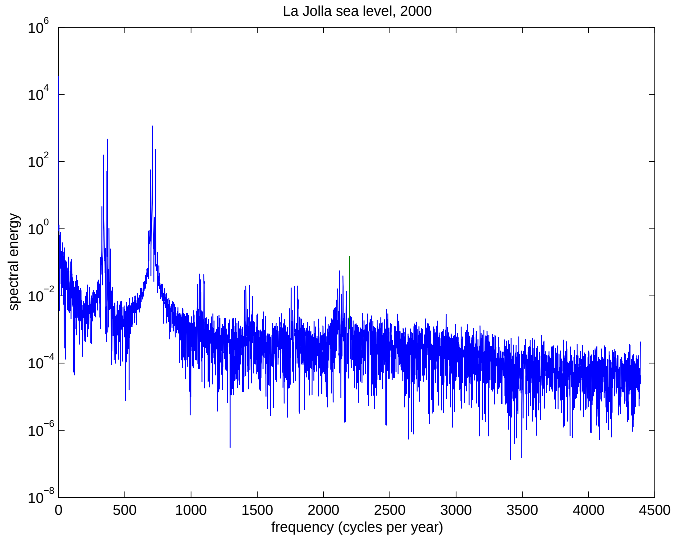

Figure 4: Spectral energy for La Jolla sea level in 2000.

We still have to decide how to normalize our spectra. Definitions are not always very consistent. In general people worry more about the slope of the spectra and the size of the peaks, and don't often interpret the absolute value of the spectra. I advocate normalizing your spectra so that Parseval's theorem is true: making the total sum of squares in the original time-domain data equal the total sum of squares in the frequency domain data. With Matlab, that means that you'll divide the computed spectrum by N.

```
s=abs(fft(data)).^2;
spectra(1)=s(1)/N;
spectra(2:N/2+1)=2*s(2:N/2+1)/N;

semilogy(0:N/2,spectra);
xlabel('frequency (cycles per N data points)')
```

Figure 4 shows the normalized spectrum for the La Jolla sea level time series. You can see two big peaks, corresponding to 1 and 2 cycles per day, and a lot of noise at other frequencies, which we don't really take to be a serious indication of statistically relevant variability. Clearly we need to compute error bars for our spectrum.