Statistics, Data Analysis, and Machine Learning for Physicists
Timothy Brandt
Spring 2020

Lecture 19

Last class we discussed the discrete Fourier transform, which is a pretty standard data analysis tool. If your research involves time series, there is a good chance that you have already encountered Fourier transforms. Many fields of research are used to operating in Fourier space. This is nothing to be afraid of, but you always need to remember the limitations of your measurements: the window function.

But what if your data are not evenly sampled? And what if they have uncertainties that vary from one measurement to the next? The discrete Fourier transform cannot deal with either of these complications. The alternative that we'll discuss here is called the Lomb-Scargle periodogram. You can get to the result in a few different ways. My favorite is via a χ^2 fit to the time series with a single sine function. The sine function has a pre-determined frequency (the trial frequency), and an amplitude and phase that must be fit to the data. We can use the identity

$$A \sin(2\pi f t - \phi) = A' \sin(2\pi f t) + B' \cos(2\pi f t) \tag{1}$$

to fit a time series $\{y(t_i)\}$:

$$\chi^2 = \sum_i \left(\frac{(y_i - A' \sin(2\pi f t_i) - B' \cos(2\pi f t_i))^2}{\sigma_i^2} \right). \tag{2}$$

This is a linear model! You can differentiate with respect to A' and B' and solve a 2×2 matrix equation to get their best-fit values. This is important. Suppose that we wanted to fit a different periodic function to the data, for example a power of sine. *I would no longer be able to express a phase shift as a linear combination of two basis functions.* Sine is special, and you can think of this as a reason why we use sinusoids in the periodogram.

I return to the task of fitting this model with two linear parameters for a sine wave at arbitrary phase. I'll define helper variables

$$S_{sy} = \sum_i \frac{y_i \sin(2\pi f t_i)}{\sigma_i^2} \tag{3}$$

$$S_{cy} = \sum_i \frac{y_i \cos(2\pi f t_i)}{\sigma_i^2} \tag{4}$$

$$S_{ss} = \sum_i \frac{\sin^2(2\pi f t_i)}{\sigma_i^2} \tag{5}$$

$$S_{cc} = \sum_i \frac{\cos^2(2\pi f t_i)}{\sigma_i^2} \tag{6}$$

$$S_{sc} = \sum_i \frac{\sin(2\pi f t_i)\cos(2\pi f t_i)}{\sigma_i^2} = \sum_i \frac{\sin(4\pi f t_i)}{2\sigma_i^2} \tag{7}$$

The derivatives are then

$$\frac{\partial \chi^2}{\partial A'} = -2\left(S_{sy} - S_{ss}A' - S_{sc}B'\right) = 0 \tag{8}$$

$$\frac{\partial \chi^2}{\partial B'} = -2\left(S_{cy} - S_{sc}A' - S_{cc}B'\right) = 0 \tag{9}$$

and the best-fit coefficients are

$$A' = \frac{S_{cc}S_{sy} - S_{sc}S_{cy}}{S_{ss}S_{cc} - S_{sc}^2} \tag{10}$$

$$B' = \frac{S_{ss}S_{cy} - S_{sc}S_{sy}}{S_{ss}S_{cc} - S_{sc}^2}. \tag{11}$$

I can them compute the improvement in χ^2 from fitting this extra sine function. **The spectrum of χ^2 improvements (as a function of frequency f) defines the Lomb-Scargle periodogram**. The classical Lomb-Scargle periodogram uses unit variances for the measurements, but there is really no reason to neglect them.

There are two slight, but important, modifications to the periodogram that you may find useful in practice. The first is to fit a model with an offset term at each frequency. In this case, you will fit the linear model

$$\chi^2 = \sum_i \left(\frac{(y_i - A' \sin(2\pi f t_i) - B' \cos(2\pi f t_i) - C')^2}{\sigma_i^2} \right). \tag{12}$$

It is still linear in the three parameters, so it really isn't any harder to compute the best-fit coefficients (you need to solve a 3×3 matrix equation rather than a 2×2 matrix equation). This is important if the mean of your data set is not zero: for example, if you are trying to measure the oscillations of a system about some nonzero equiliubrium state. In that case, you should also fit for the mean in the absence of a sinusoidal signal. This is equivalent to fitting a constant to your data, or subtracting the inverse-variance weighted mean. If you do this, you have one free parameter to your fit without a sinusoid, and three free parameters with a sinusoid. You are still adding two free parameters, so the χ^2 thresholds should still be well-defined, and they should be the same as in the case without a fitted constant.

The second modification is to replace the inverse variances with the inverse covariance matrix of the data if you have correlated errors. As usual, in order to do this, you need to know the covariance matrix of your data!

You can use the Lomb-Scargle periodogram to measure the strength of a given sinusoidal signal in your data. If your signal is pure Gaussian noise properly characterized by your uncertainties σ_i (or your full covariance matrix!), the χ^2 improvements will follow a χ^2 distribution. Your sinusoid fit adds two degrees of freedom, so that gives the expected improvement in the absence of a signal. You can then set significance thresholds using the survival function of the relevant χ^2 distribution just as we did earlier in the course. Remember, however, that if you compute the periodogram at a large number of frequencies (say, 1000), then you have 1000 chances to get a false positive. If you want a 0.3% chance of a false signal over your entire data set, you need a 0.0003% chance (0.3% divided by 1000) for each trial frequency. Your χ^2 threshold (or Lomb-Scargle threshold) would then be the inverse survival function of the χ^2 distribution with two degrees of freedom:

```
scipy.stats.chi2.isf(3e-6, 2)
```

or 25.4. Contrast this with the 0.3% false positive threshold without the extra factor of 1000. In that case (without the trials factor), your threshold would be `scipy.stats.chi2.isf(3e-3, 2)`, or 11.6.

One important limitation of the Lomb-Scargle periodogram, especially if you are interpreting it using χ^2, is that you can only treat $\exp(\Delta\chi^2/2)$ as a probability if you think a pure sinusoid plus noise is a good model for your data. You could choose other periodic basis functions if you wanted. Perhaps your application presents a natural set of period basis functions that are not pure sinusoids. Recall, however, that we used Equation (1) in a very important way: it transformed a fundamentally *nonlinear* fit into a fundamentally *linear* fit. If, for example, your basis function is a periodic delta-function impulse (a Dirac comb), then you cannot do this. You can no longer optimize over phase using linear least squares. This is important, for example, in looking for exoplanet transits. A good model for a transit is a periodic rectangular window, and this cannot be phase-shifted by just adding a compensating function.

An important difference between the Lomb-Scargle periodogram and the discrete Fourier transform is that the Nyquist limit doesn't apply in general to the periodogram. As long as your data are irregularly sampled, you can often access frequencies much higher than your typical sampling frequency. The preceding discussion also glossed over the window function. It's still there, lurking. You can think of the Lomb-Scargle periodogram as a good estimate of the power spectrum of your data convolved with your window function, so it's still a good idea to take the periodogram of your window function to make sure that it isn't responsible for your statistically significant peak. There have been very high-profile claimed discoveries that turned out to be wrong. The claimed planet Alpha Centauri Bb, for example, turned out to be just a window function.

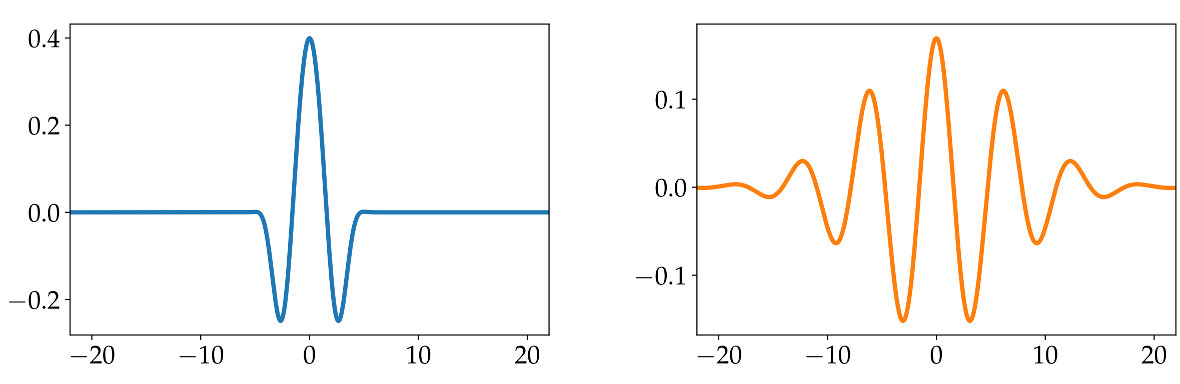
As a final topic, we'll spend a little bit of time talking about Fourier transforms over a limited region of data, or, a closely related idea, wavelet transforms. These are useful if you are looking for periodic signals that shift or change, or if you want to look for quasiperiodicity. This field is known as *time-frequency analysis*. You want a temporally localized periodicity search or periodicity measure. In the Fourier case, this is called the *short-time Fourier transform*; you just perform a Fourier transform over small regions of your data. You have to tune the length of these short regions though, and this length tuning determines the lowest frequency you can access (recall aliasing!).

Probably more common and standard today is the wavelet transform. You choose a template function, like a sinusoid, but one that is *localized in time*. This is called a *wavelet*. The wavelet should be normalized in the sense that it integrates to zero, it should oscillate so that its square integrates to a positive number, and it should be zero (or very nearly zero) outside some range. Typically this range may be from one to a few oscillations. With a suitable wavelet (think of an oscillating pulse), you can then define the *continuous wavelet transform* of a signal $g(t)$ using a wavelet ψ as

$$X(l, \tau) = \frac{1}{\sqrt{l}} \int_{-\infty}^{\infty} g(t) \psi \left[\frac{t - \tau}{l} \right]. \tag{13}$$

You convolve your signal with your wavelet to get the cross-correlation as a function of time. In this case, l is a time scale stretching the wavelet—kind of like a period—and τ is the time at which I represent my time series in this way. The Fourier transform gave me a series that had only frequency resolution, not temporal resolution. The wavelet transform gives me both. You can choose a different streching parameter to fill out your two-dimensional transform in frequency space.

The wavelets should generally be oscillating, but fall to zero outside of a range. Extending the range allows you to fit more oscillations. This improves your frequency resolution (you are matching more peaks, so the phase needs to line up across more wavelengths). This frequency resolution comes at the expense of time resolution, kind of like an uncertainty principle. The figure below shows two example wavelets that integrate to zero. Both are sensitive to the same frequency, but the one on the left has greater temporal resolution at the expense of frequency resolution. Like an oscillator with a lower quality factor, it will respond to a broader range of frequencies. I have normalized both so that $\int \psi^2 = 1$.



In practice, people use a discrete wavelet transform for real data. Like the Fourier transform, this give results at specific stretchings (analogous to Fourier frequencies) and times. You can construct your wavelets to have additional desirable properties, for example, orthogonality. This allows you to go back and forth between your original time series and its wavelet representation using the same recipe.

You can also use a wavelet transform for things other than time series. Wavelet transforms are used in music, but also in video and imaging. As an example, jpeg image compression uses a wavelet transform and achieves compression by storing these coefficients to limited precision. Because you need a lot of finely-tuned Fourier components (or wavelets) to resolve a sharp edge, jpeg introduces artifacts at sharp edges and doesn't work well for text.

There are a few very common and standard choices for wavelets that you can look up. I've never used a wavelet transform in my own work, but it is an appropriate choice for studying quasiperiodic phenomena or oscillations of oscillations, when you want to study the time-resolved frequency composition of your data.