4. The convolution theorem and padding/windowing/smoothing:

Convolution of 2 functions is a *definite integral of their product*:

$$\theta(t) = \int_{-\infty}^{+\infty} g(\tau)h(t-\tau)\, d\tau$$

$$\theta = g * h$$

a. Convolution as smoothing: the concept of a *kernel* function

A powerful use of convolution comes from having one function be your data and the other be a "kernel" or "weighting function" or "window" or "sampling mask". For example, thinking about a sampling mask made of equally spaced delta functions is the math that connects continuous Fourier analysis to its discrete cousin. Another use is a *smoothing kernel*, a function with finite values around τ=0 but decaying away to 0 for large | τ |. This might be a square "top-hat" or "boxcar" function, or a Gaussian or other bell curve, or a "1-2-1 filter", which is a narrow boxcar stacked on a wider boxcar (plotted below).

The convolution of a time series h(t) with a smoothing kernel g(τ) can be thought of as replacing each data point on the graph of h(t) with a little copy of the kernel function g(τ), centered at that time t and scaled by the magnitude x(t). Now add up all these kernels, and you get a smoothed time series. In basic calculus, you thought of an infinitesimal area x(t)dt under a graph as a narrow square "tower". If you are convolving h(t) with a bell shaped g(τ) kernel, you instead replace this little tower at each value of t by a bell curve with the same area under it. Sum up all the little bell curves, and what do you get? A smoothed (smeared) version of h(t).

Figure 13 shows a graphical example of the above idea: (Forgive the small mis-offsets of colored bars relative to vertices of the dotted time series; a software annoyance).

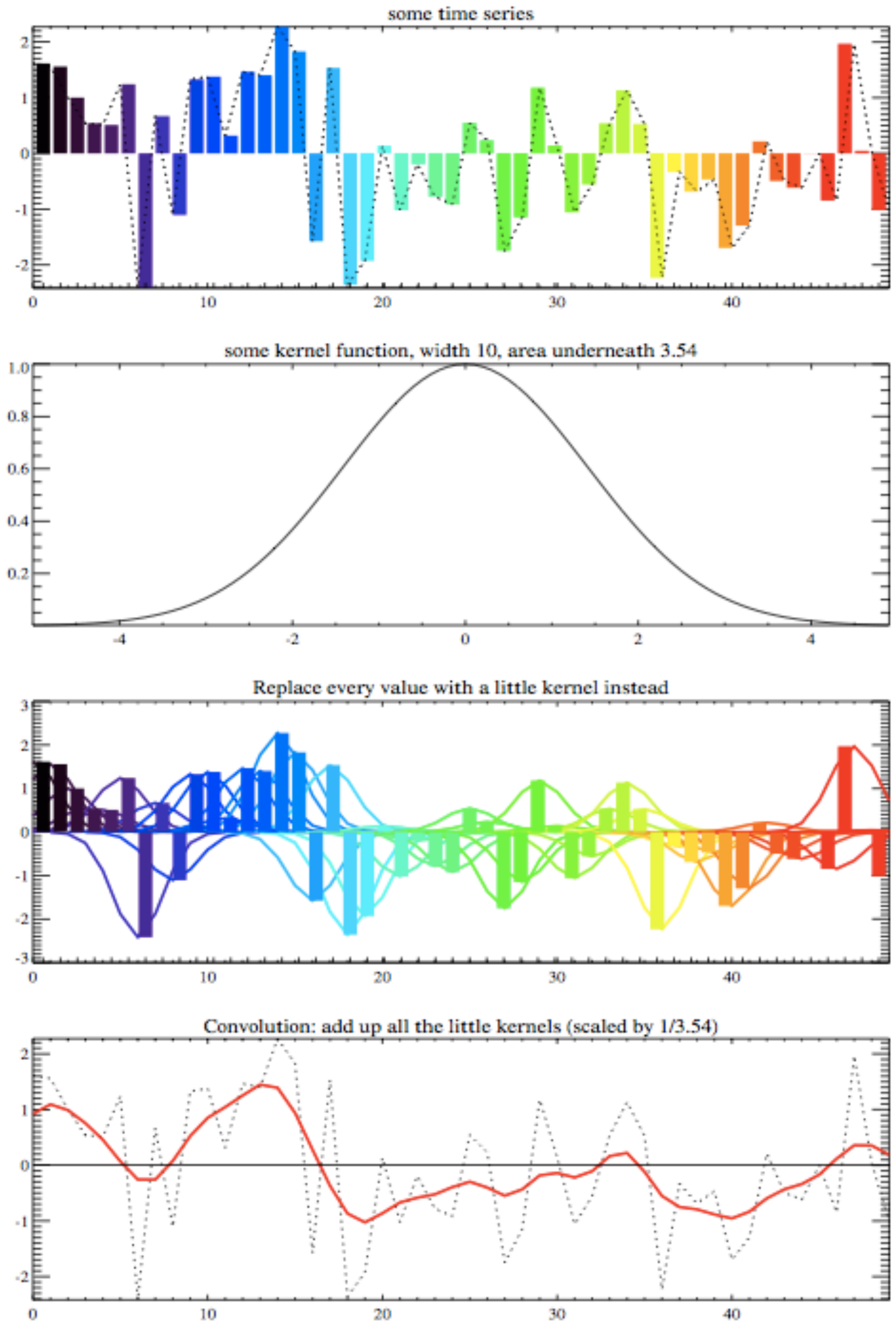
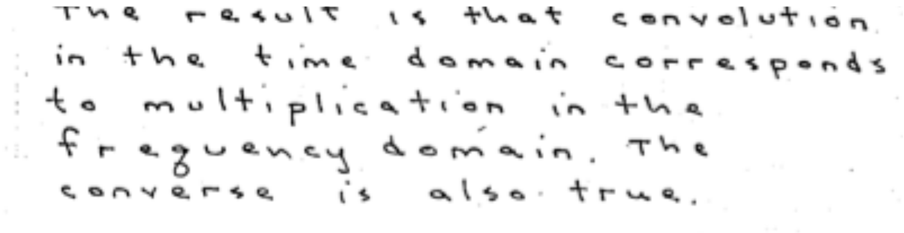


Figure 13. Illustration of the convolution of the dotted time series with a Gaussian kernel to get a smoothed time series (red).

b. The Convolution Theorem links spectral and physical spaces:

I learned this profound fact from the lectures and notes of Dr. Kevin Leaman (1949-2015), teacher of our predecessor course for many years.

The convolution theorem:



The result is that convolution in the time domain corresponds to multiplication in the frequency domain. The converse is also true.

Let's use this to revisit the problem of "**spectrum estimation**" – trying to understand the error bars in the *frequency domain* that result from various imperfections and finiteness problems with the data in the physical or *time domain*.

We saw that taking a finite time segment of time length T makes the power spectrum discrete, with *integer* frequencies 1,2,3... cycles/T. Padding with zeroes at both ends, out to infinity, makes the spectrum continuous. Such a zero-padded series is just a multiplication in time of an infinitely repeating set of copies of our time series on [0,T] (Fig. 1), multiplied by a boxcar(t) masking function which is 0 everywhere except on [0,T]. Transforming this product into spectral space, the multiplication becomes a convolution or smoothing process, where the true spectrum is convolved with a kernel that is the Fourier transform of the boxcar function.

TIME: Zero-padded series = periodic L(t) x boxcar(7 days)
FREQ: Power(" ") = $|\mathcal{L}(f)|^2$ *convolved w/* FT(boxcar)

What does that kernel look like?

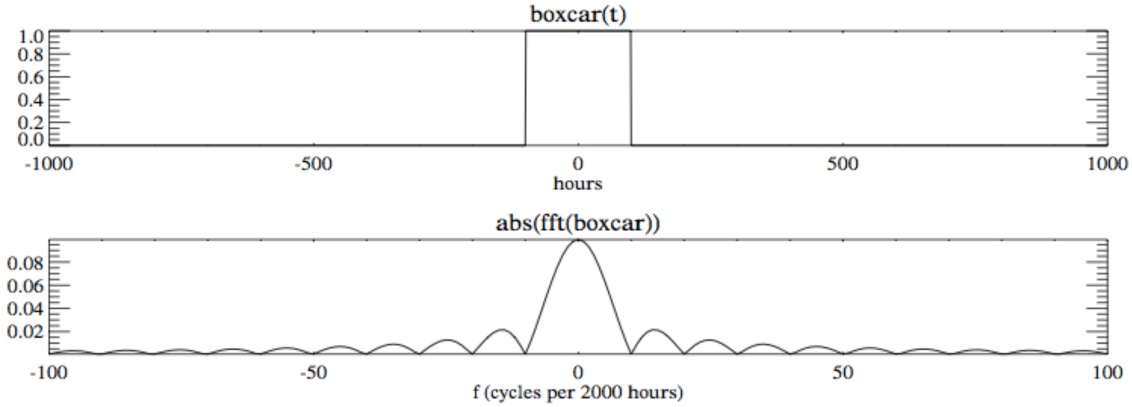


Figure 14. The boxcar function and its Fourier amplitude spectrum. The nonzero mean of boxcar(t) corresponds to nonzero power at frequency=0 in the transform (spectrum).

So if we pad our 7-day L(t) series from Fig. 1 with zeros, the spectrum of the resulting infinite series is the spectrum (Fig. 2) *convolved with* (smeared or smoothed by) the *kernel* function whose absolute value is in the lower panel of Fig. 14 (note: the lobes oscillate in sign). In other words, at every frequency in the spectrum of Fig. 2, you replace each local spectral power value with a copy of the kernel and add them up, as done in Fig. 13. A nice sharp peak in the true spectrum will become a smeared peak, with "echoes" from the side-lobes of Fig. 14 further spreading its power very widely across the frequency domain. This smearing is called "spectral leakage".

To reduce the side-lobes and long-range leakage, it be better to smoothly taper the ends of our padded data sequence. That is, we would multiply the infinite periodic data record by some kind of a rounded window rather than the square boxcar (mask) function above. Let's imagine the results, based on the fft of different window functions:

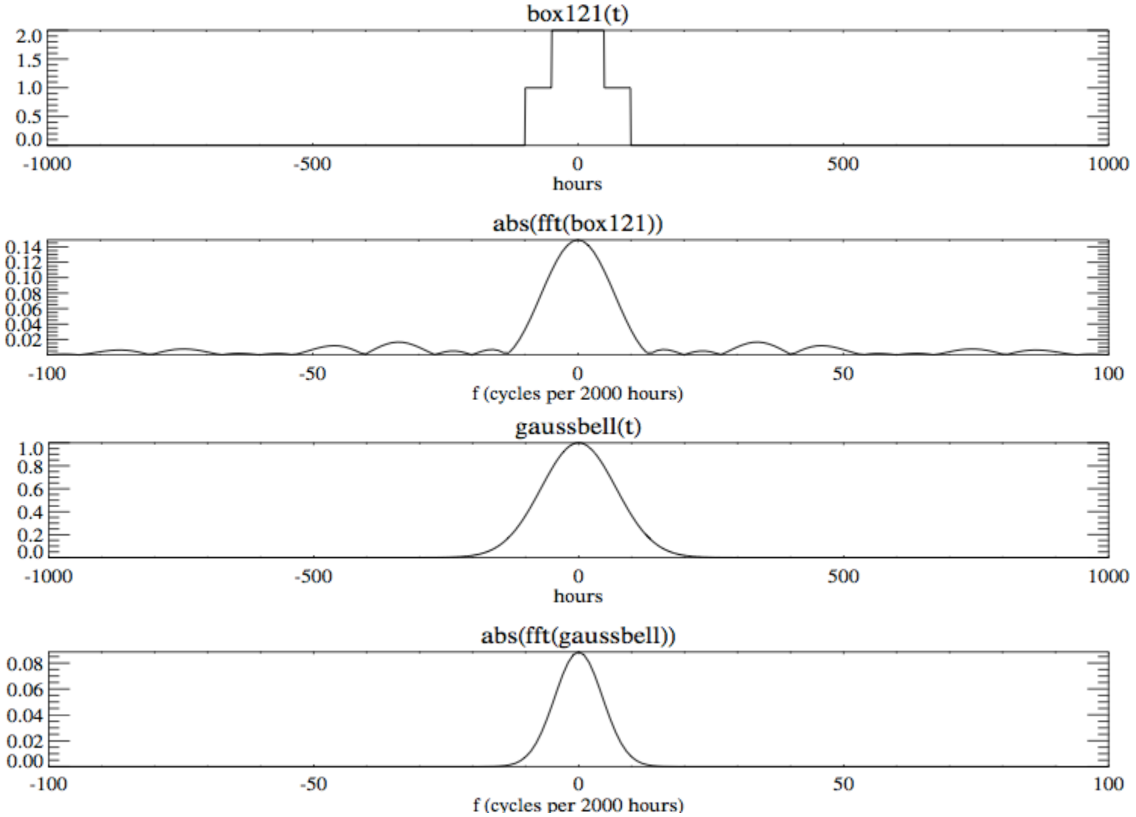


Figure 15. As in Fig. 14, but for "window" functions that taper, instead of a square boxcar.

WOW! **The Fourier transform of a Gaussian is a Gaussian** (spooky? or just beautiful?). *No side lobes.*

Now remember, this works both ways:

converse is also true.

So when somebody says they used a "1-2-1 smoother" on their data, it means they *convolved* their time series with the 121 function of Fig. 15. Often people will do this several times in succession to make things more smooth. Well, that is equivalent to multiplying their spectrum by the fft(box121) function above (perhaps many times). That is called a “low pass” filter: low frequencies are almost unaffected while frequencies comparable or higher than 1 cycle/(boxcar width) are damped (averaged away).

If you really want a pure low-pass filter, it should have a boxcar-like step-function structure in *frequency* (low frequencies completely passed, high frequencies completely blocked). To do that with a smoothing kernel, you’d need a complicated wiggly kernel like fft(boxcar) on Fig. 14. But that has a wide stencil or ‘footprint’ in the time domain, with all those sidelobes, so it’s impossible to apply near the ends of a time series. These are the inescapable trade-offs of having limited information content in a time series. There is an uncertainty principle to this: <u>there is an inescapable tradeoff between resolution in the time domain and resolution in the frequency domain</u>.

c. A glimpse of wavelet analysis

Wavelet analysis is an attempt to optimize that inescapable time-frequency tradeoff. Instead of decomposing a time series into sines and cosines that oscillate to infinity, a basis set of *wavelet functions* is used. Wavelet functions W(f,τ) are like localized wave *packets*. They oscillate at some frequency (characterized by some parameter f) around τ=0, but with an amplitude that decays at large |τ|. Wavelet basis functions probe the time series for *local* wiggly features, not just global periodicities that exist somewhere within the whole time series.

Recall that Fourier coefficients are the “projection” (the mean of the product) of your data with sine and cosine functions. Like a Fourier spectrum, the wavelet spectrum is given by the *projection* (or resemblance, measured by the integral of the product) of your data with the wavelet basis function. A **wavelet spectrum** is a function of both time and frequency, $\mathscr{L}_w$**(f,t).**

If your wavelet basis function has a lot of wiggles, it will act like sines and cosines with their intense orthogonality: the projection onto a time series will clearly distinguish the *frequency* of oscillations. But in order to have a lot of wiggles, it must have a wide footprint or stencil in the time domain. That means the projection will light up anytime that long wavelet stencil overlaps with a wiggle in the time series. As a result, the wavelet spectral power will be similar for all the times that are within its time stencil, leading to poor time resolution.

On the other hand, a more compact-in-time wavelet function like the DOG (derivative of Gaussian) or Mexican Hat (you can google image these) just has one sidelobe, so it can’t distinguish frequencies very sharply, but its localization in time makes it good at localizing time series peaks or “events”.

So you choose your wavelet function based on your purposes, and what you are trying to extract or emphasize about your time series. For a time series with “event-like” features, a compact wavelet is good; for a time series with “wave-like” features whose frequency you want to resolve, it makes sense to use a more wavelike wavelet function.

d. Revisiting the problem of aliasing using the convolution theorem

Let's revisit the aliasing problem of Fig. 8, the 3-hourly subsampling of 2-minute L(t) from section 3a above. Figure 15 shows the "shah" (comb or spikes) function that, when multiplied by the full time series, gives the 3-hourly samples. Its amplitude spectrum (bottom) is also a series of spikes, now in frequency, spaced 8 cycles/day apart. Notice that this is twice the Nyquist frequency of 4 cycles per day.

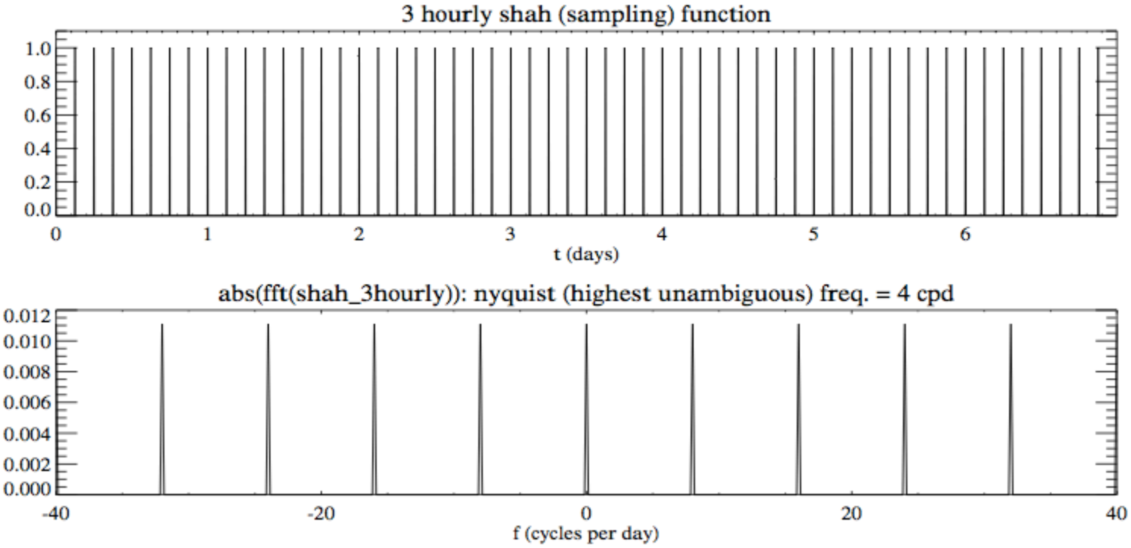


Figure 15. The comb or "shah" function: a set of equally spaced spikes. Its product with our L(t) data gives the subsampled data (red triangles in Fig. 8a). Its Fourier spectrum is shown at bottom. Notice that the nonzero mean of the function value gives a nonzero transform at frequency f=0.

According to the convolution theorem, the effect of multiplying L(t) by the 3-hourly shah(t) function in time is to *convolve* $\mathcal{L}$(f) with $\mathscr{shah}$(f) in the frequency domain. In this case, our spectrum $\mathcal{L}$(f) is the function that is more compact around the origin (since geophysical spectra usually are "red"), while $\mathscr{shah}$(f) is the function that extends to infinity (Fig. 15b). So it is probably clearer to think of our spectrum as the 'kernel' and $\mathscr{shah}$(f) the thing being 'smoothed' by that kernel.

Figure 16 shows the spectrum of the full data, plotted as amplitude (not power), and symmetric (not just the positive frequencies). It's the same information as Fig. 2 (repeated at right here as a reminder):

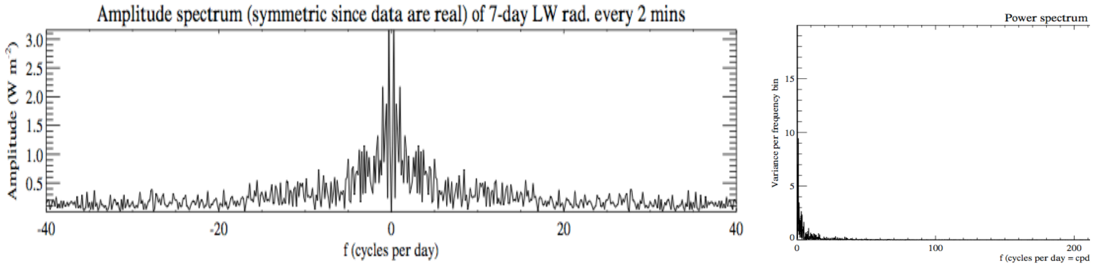


Figure 15. The amplitude spectrum of L(t), shown in symmetric form around f=0 for clarity.

Now we can build the convolution as a kernel sum, just like in Fig. 13:

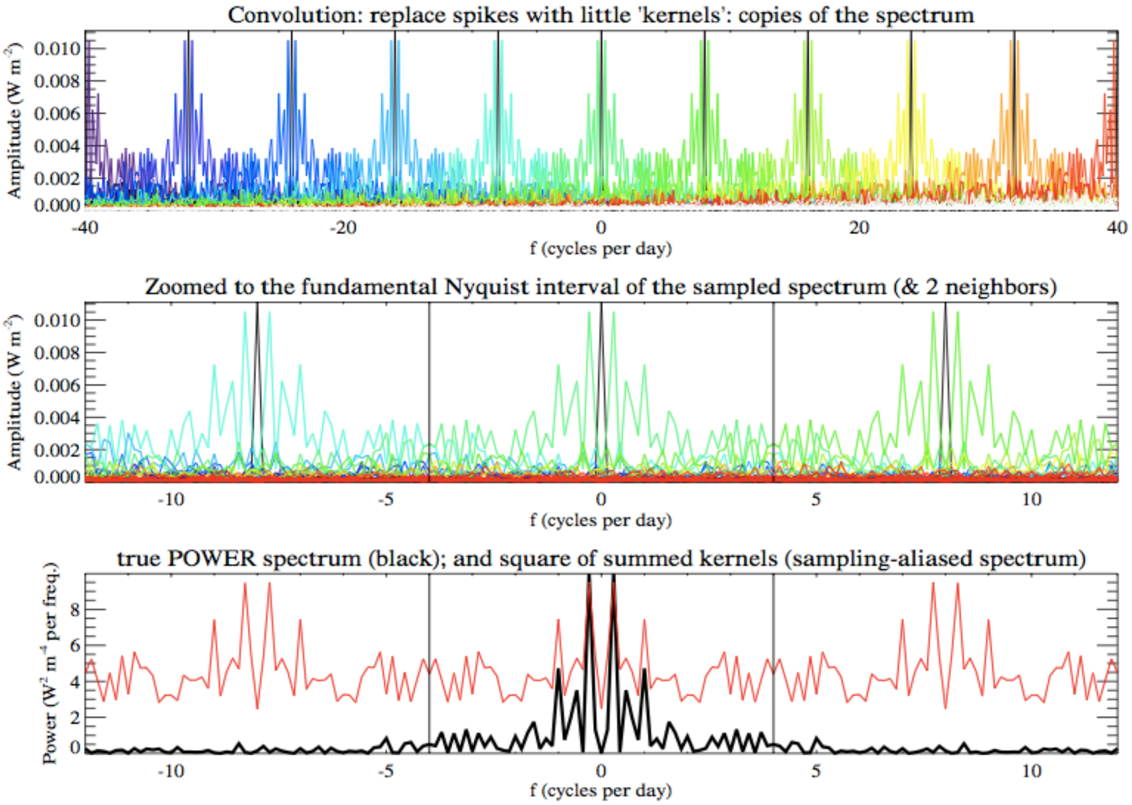


Figure 16. Summing of kernels as in Fig. 13. The true spectrum ℒ (f) is treated as a "kernel" for the sampling or shah function whose values are replaced by the kernel. In the bottom panel, the true power spectrum is in black, and the aliased power spectrum due to 3-hour undersampling is in red. The red-black gap is a noise level or error bar for all possible alignments of the 3h sampling.

When we sum up all the kernels in Fig. 16, the long tails of high-frequency power from all the faraway copies of the spectrum add up to a large amount of spurious (aliased) power. The red-black gap at bottom here is a vertical "error bar" for spectra. As we saw in Fig. 8, 3h subsampling gives modest error on our lowest frequency peak (2 cycles/7 days). For the diurnal peak (1cpd), 3h sampling produces about a ~100% error bar (spurious, aliased power about equal to true power). For f > 1cpd, the spurious power is larger than the true power. That's consistent with the results of our one particular sampling *realization* from Fig. 8 (Fig. 17): the low frequency is maybe mostly right, diurnal is problematic, and the rest is mostly spurious.

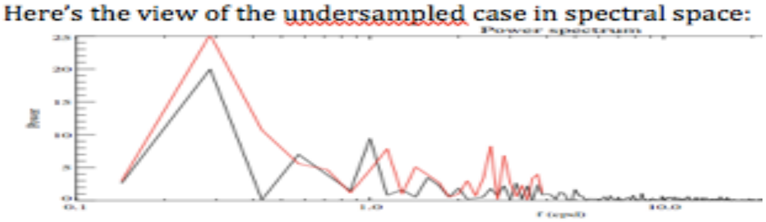

Here's the view of the undersampled case in spectral space:

Figure 17. Repeating Fig. 8: the sampling-aliased spectrum for 3-hourly sampling aligned with t=0.

5. Summary of “ERROR BARS” in frequency space arising from data shortcomings in physical space (or time)

Taking the Fourier transform of a **finite** segment [0,T] of data L(t) to get $\mathcal{L}$(f) tacitly assumes that the patterns in L(t) repeat periodically every interval T, forever in all t. The finiteness of T gives a **lower bound** on the lowest frequency (1 cycle per record length T) in $\mathcal{L}$(f). It also **discretizes the frequencies** (1,2,3,... cycles per T). This discrete frequency spacing (“**bandwidth**”) of 1 cycle/T is, in estimation terms, a form of horizontal “error bar” in frequency space.

Spectral “leakage” is the continuous version of this discretization of bandwidth. For example, say we pad the T length data sequence with 5T worth of zeros on each ends, giving a total padded record length of 11T. Analyzing this longer sequence (which, again, assumes this longer 11T sequence is periodically repeated to infinity) will give 11 x finer spectral resolution (1 cycle/11T instead of 1 cycle/T). But does this mean we will be able to actually discriminate frequencies that are closer together than 1 cycle/T? No, because spectral “leakage” or smearing will still act as a horizontal error bar (that is, in frequency). This should not seem surprising, if you think about it: information-wise, you can’t get something for nothing (where padding with zeros is literally nothing!).

When we have a **discrete** sequence of data values at times separated by dt, it gives an **upper bound** to the frequencies we can resolve: (1 cycle per 2dt), the **Nyquist frequency**.

Folding (aliasing) of power from beyond the Nyquist frequency is shown in Fig. 16: the tails of power beyond the central Nyquist interval (-4 to 4 cpd in Fig. 16) add up to produce aliased power, providing a **vertical error bar on our estimates of the power spectrum**. *Much the power in an undersampled spectrum is spurious (aliased)*. Revisit Fig. 9 for a sobering reminder of how much.

To go beyond these broad generalities about horizontal and vertical error bars on spectra, we will need to define some “null hypothesis” spectra for “colored noise” (red or Brown) to use for “significance testing” of spectral peak detection. After defining all those terms better…