Sampling & Windowing

In this chapter, we discuss two sort of unrelated topics. The first is how to determine a correct sampling interval **Δ**, and the second discusses the concept of windowing.

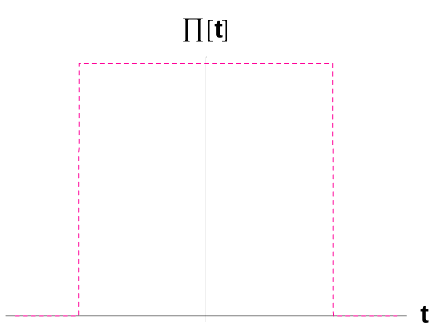
■ Sampling

In the previous chapter we saw that if one is collecting a time series and will later wish to do a Fourier transform of the data, then the number of data points should be a factor of two. Here we discuss another factor in experiment design: how often we need to sample the signal.

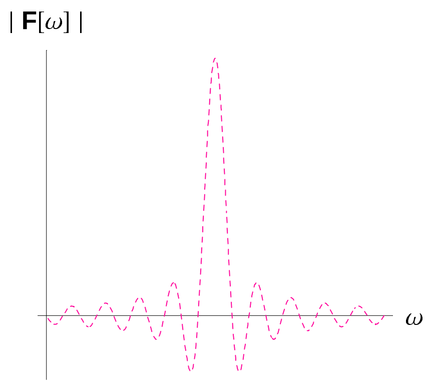
We have seen that if we have a sine wave that is non-zero for only a limited number of cycles, its Fourier transform has sidebands. These sidebands arise for any function which is non-zero for only a finite time interval. In the context of experimental measurement, since we can never collect data for an infinite length of time, the Fourier transform will *always* have such sidebands.

In this context, it is interesting to note that the Fourier transform of a Gaussian is itself is the Gaussian. This makes the Gaussian unique. Of course the Gaussian only approaches zero asymptotically as **t** approaches ±∞.

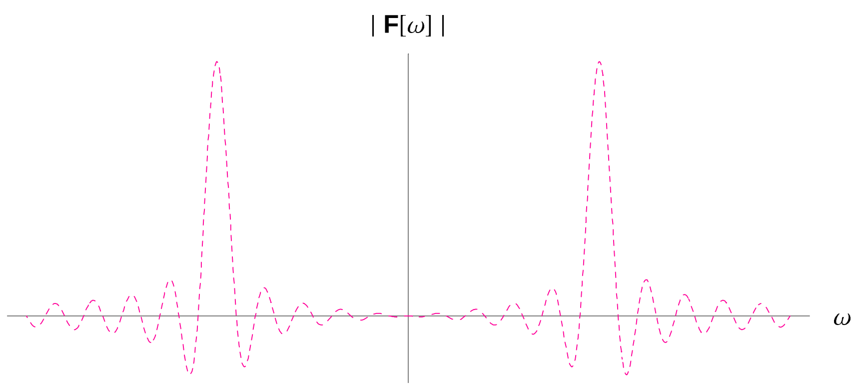
We also saw in the previous chapter that the *amplitude* of the *discrete* Fourier transform is an even function, so the frequency spectrum for positive frequencies is reflected for negative frequencies. For example if we have a time series of a rectangle function **∏[t]**:



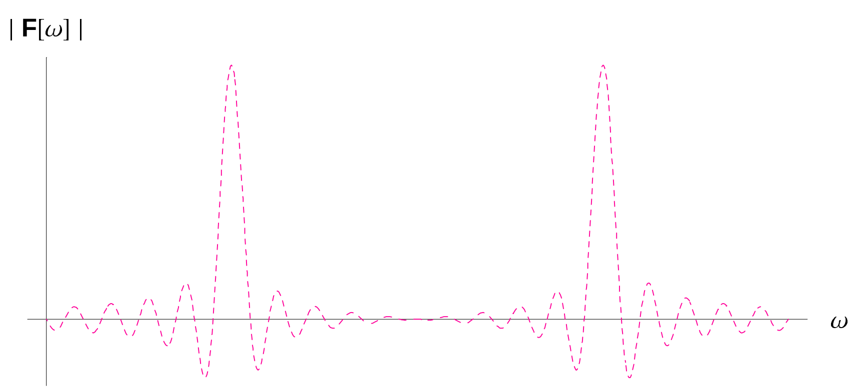
its Fourier transform is a Sinc:



Because we are dealing with the discrete Fourier transform, there is an "alias" or "Nyquist pair" of the above transform.



Mathematica reflects the alias at the end of the positive frequency part of the transform:



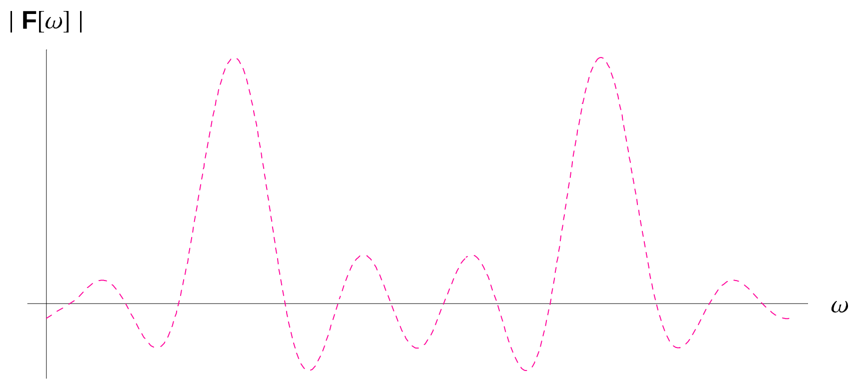
The reflection occurs about some frequency ω which is at the half-way point of the frequency axis. The value of the reflection point is called the *critical* or *Nyquist* frequency. Recall that the frequency of a term in the type-1 Fourier series is given by:

$$\omega_j = j\left(\frac{2\pi}{n\Delta}\right),\ j = 0,\ 1,\ 2,\ \ldots,\ n-1$$

Thus, the critical frequency is:

$$\omega_c = \frac{\pi}{\Delta}$$

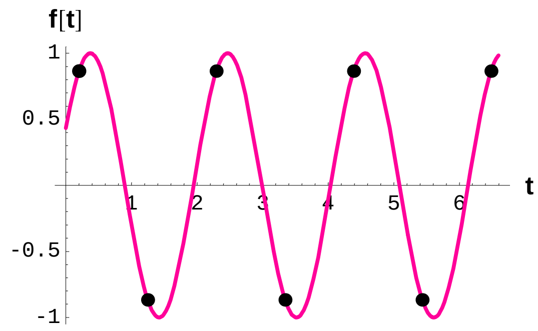
In the above figure, note that there is no significant overlap between the signal and its alias, because the value of the transform is so small at the Nyquist frequency. If the rectangle function **∏[t]** becomes more narrow, we know that the Fourier transform becomes wider. Thus, for a narrow rectangle time series we could have a Fourier transform that looks like:



Now the alias overlaps with the transform. Thus, in this case it is impossible to sort out which is the transform and which is the alias.

Since usually we can't change the frequency components of the signal we are sampling, one way to avoid minimise this problem is to increase the critical frequency; this is accomplished by decreasing the sampling interval **Δ.** The effect is to push the alias away from the signal.

Say we have a continuous sine wave with angular frequency of 3 radians/sec. If we sample it so that the critical frequency is also 3 radians/sec., this corresponds to **Δ** equal to $\pi/3$. Thus, we end up sampling the sine wave at exactly two points in each cycle:



Thus sampling at this rate allows us to exactly determine the frequency of the sine wave. We ignore for now the possibility that we begin sampling when the sine wave is zero; in this unlucky case we end up with all zeroes in the time series.

Now imagine some system in which the frequency spectrum is bandwidth limited, such as a music amplifier that produces no output for inputs greater than 100 kHz $= 100 \times 2\pi \times 10^3$ radians/sec. Then if we sample it such that the critical frequency is equal to this upper frequency limit *we have completely determined **f[t]*** since we know all the frequency components. For the example we have been using, this works out to be:

$$\Delta = \frac{\pi}{\omega} = \frac{1}{200000}\ \text{secs}$$

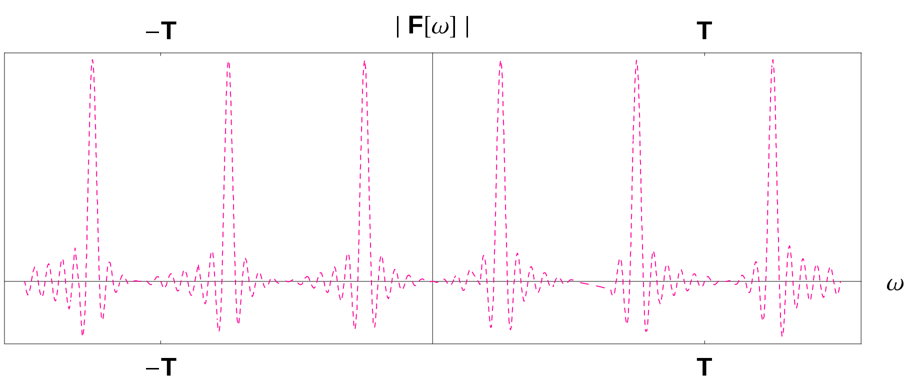
The result we have just obtained is somewhat remarkable, and is thus promoted to theorem-status:

Sampling Theorem: if a signal is bandwidth limited, it is possible to completely determine the signal by sampling it at a rate such that the critical frequency is equal to its upper bandwidth.

The bad news is that if the signal is not bandwidth limited, we have seen that frequencies higher than the critical frequency are "aliased" into the region from **{ $-\omega_c$, $+\omega_c$}**. Once this has happened there is little that can be done remove the aliased components.

Thus, in designing an experiment (1) know the bandwidth of the system in advance or impose one by analog filtering of the continuous signal, and then (2) sample at a rate sufficiently rapid to give two points per cycle of the highest frequency component.

Finally, you may be thinking that if *Mathematica* did not put the negative frequency components at the end of the positive frequency ones we would not have a problem. Sadly, this won't work. We have been skirting around the question of whether or not the Fourier transform assumes that **f[t]** is periodic. However, for the discrete Fourier transform F_j, examining the definition makes it clear that we get the same results for frequencies that are $\pm N 2\pi$ of the ones between $\{-\omega_c, \omega_c\}$ where N is an integer. Thus, we can consider the transform as being periodic.



Note that the region in the above plot between 0 and **T** is identical to the form from *Mathematica*. Here, though, we interpret the "alias" as the first half of the next period of the transform.

■ Windowing

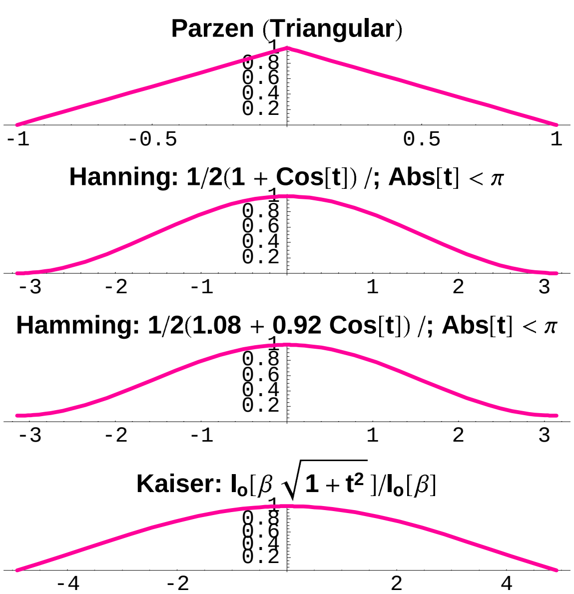
We saw in the previous chapter that an ideal low pass filter is not possible because the inverse Fourier transform of the rectangle function ∏ is a Sinc function, and the Sinc function extends from -∞ to ∞. Put another way, it has an infinite impulse response.

Recall the Convolution Theorem, that states that the convolution of two functions is equal to the inverse Fourier transform of the product of the Fourier transforms of the two functions. The converse is also true: multiplication in the time domain is equivalent to convolution in the frequency domain.

Thus, if we can limit the **Sinc** function to some finite range of times we have a physically realisable filter. One way to do this is to multiply the **Sinc** by some function that is non-zero only in a finite range of time. Such a function is called a *window*. The choice of windows is a somewhat arcane art, and practically every function which rises from zero to a peak and than falls to zero again is associated with somebody's name. You will use windows again when we study power spectrum estimation.

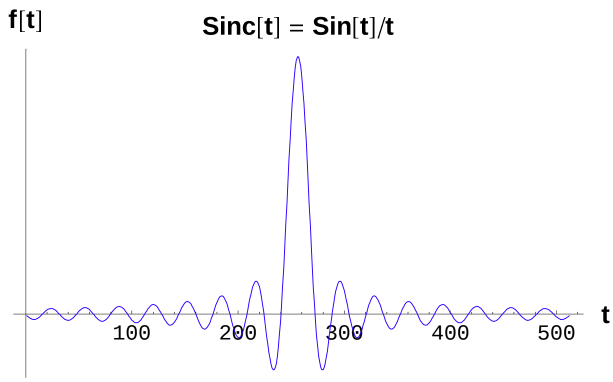
Note that the effect of the windowing will be to "spread" the rectangle in the frequency domain.

Some common windows are:



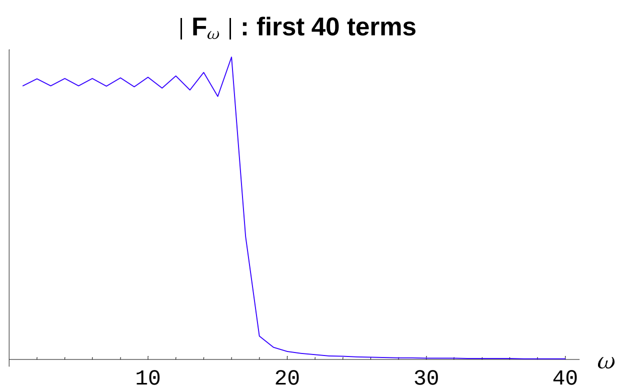
Note that the *Hamming* window does not go to zero at the extremes. Also, for the *Kaiser* window above, I_o is the zeroth-order modified Bessel function, and the window function has a second parameter β; *Mathematica*'s zero-th order modified Bessel function is **BesselI[0, t]**; for the plot shown β is 0.5.

We finish this section with an example. We have a 512 element time series of a **Sinc** function:



Obviously, we have had *Mathematica* "connect the dots" in the above plot; we will do this for all subsequent plots too.

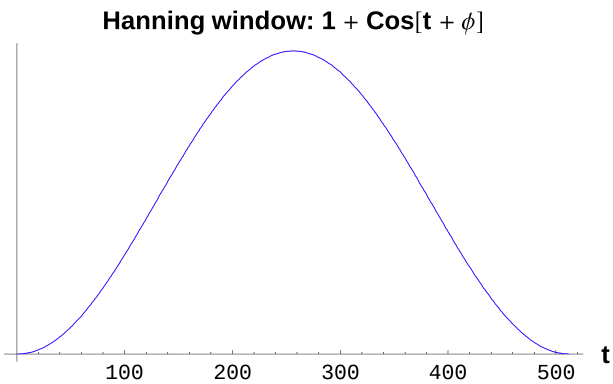
The first 40 elements of the absolute value of the Fourier transform of the above time series is:



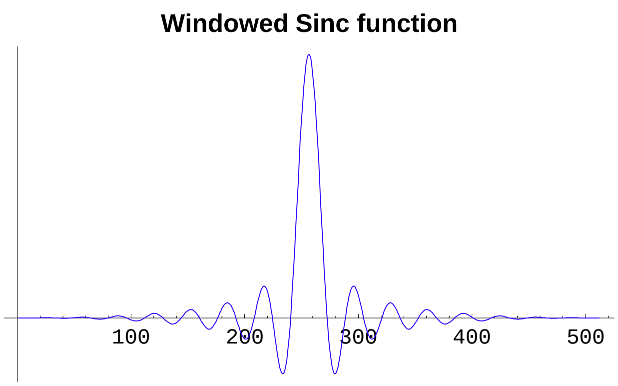
These are the positive frequency terms of the low pass filter. Since the **Sinc** has been cut off for **k** < 0 and **k** > 511, which is equivalent to imposing a rectangular window, the filter is far from ideal.

We have used *Mathematica*'s type-3 **Fourier** in the above. The only difference from a type-1 transform is the relative signs of the real and imaginary parts, which we have wiped out by taking the absolute value, and the overall normalisation, which we don't care about here.

We construct a 512 term Hanning window:

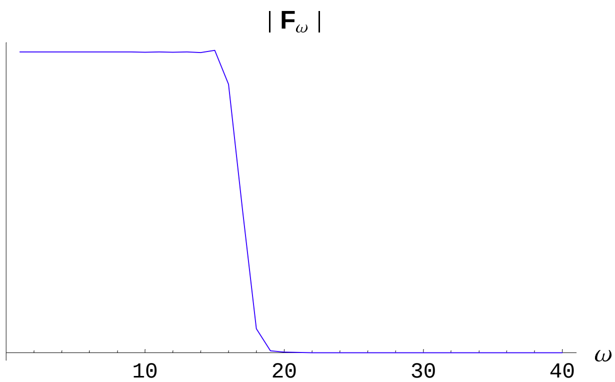


We multiply the original **Sinc** time series by the window:



The effect is to have the extremes more-or-less gradually go to zero.

The absolute value of the Fourier transform of this time series is a much better filter:



■ Author

 It was prepared using *Mathematica,* which produced a PostScript file; *Adobe Distiller* was then used to prepare the PDF file. This is version 1.3 of the document, date (m/d/y) 01/30/99.