

Calculating the FT and IFT in the finite duration (finite length sequences in both TD and FD), fully discrete (both in TD and FD) domain.

We can distinguish 4 types of Fourier transform based on what we've been developing so far.

A signal can be either **continuous** or **discrete** and either **periodic** or **aperiodic**. This results in four types of signals -

i) aperiodic continuous - such as decaying exponentials and gaussians

ii) periodic continuous -such as sines and square waves

iii) aperiodic discrete - similar to (i), but sequence rather than continuous

iv) periodic discrete - similar to (ii), but sequence rather than continuous

Each of these classes of signal extend to \pm infinity (even for finite length signals in the computer, since the basis functions - sine and cosine - are defined to \pm infinity). We also already know that you need an infinite set of frequencies to synthesize an aperiodic signal.





Type of Transform	Example Signal
Fourier Transform <i>signals that are continuous and aperiodic</i>	
Fourier Series <i>signals that are continuous and periodic</i>	
Discrete Time Fourier Transform <i>signals that are discrete and aperiodic</i>	
Discrete Fourier Transform <i>signals that are discrete and periodic</i>	

Figure from Smith

This leads to four types of Fourier Transform pairs (**ignoring normalization constants**)

Aperiodic continuous (Fourier Transform)

$$X(\omega) = \int_{-\infty}^{\infty} x(t)e^{-i\omega t} dt \quad x(t) = \int_{-\infty}^{\infty} X(\omega)e^{i\omega t} d\omega$$

Periodic continuous (Fourier Series)

$$\tilde{X}(\omega) = \int_{-T_0/2}^{T_0/2} \tilde{x}(t)e^{-i\omega t} dt \quad \tilde{x}(n) = \sum_{n=-\infty}^{\infty} \tilde{X}(\omega)e^{i\omega n}$$

Aperiodic discrete (call it the Discrete Time Fourier Transform)

$$X(\omega) = \sum_{k=-\infty}^{\infty} x(k)e^{-i\omega k} \quad x(n) = \int_{-\pi}^{\pi} X(\omega)e^{i\omega n} d\omega$$

Periodic discrete (Discrete Fourier Series, if also finite-duration with proper interpretation becomes Discrete Fourier Transform, continuous ω goes to discrete k in Frequency Domain)

$$\tilde{X}(k) = \sum_{n=0}^{N-1} \tilde{x}(n)e^{-i(2\pi/N)nk} \quad \tilde{x}(n) = \sum_{k=0}^{N-1} \tilde{X}(k)e^{i(2\pi/N)nk}$$

Consider a periodic sequence with period N (note, N is not the length of the sequence, which is still infinite). We will denote periodic sequences with a tilde, $\tilde{x}(n)$, and we have by the definition of periodicity

$$\tilde{x}(n) = \tilde{x}(n + kN) \quad \text{for all integer } k.$$

The DFS can be found from the FT we have been using as follows -

First, remember how we discretized the “time” variable when we went from the continuous case to a sequence, t went to $k\Delta t$.

Now we do the same to the frequency variable, ω goes to $2\pi n\Delta f$. And $\Delta f = (1/N\Delta t)$, so ωt , which went to ωk before, now goes to $2\pi n k / N$ and the FT becomes

$$\tilde{X}(\omega) = \sum_{k=-\infty}^{\infty} \tilde{x}(k) e^{-i\omega k} = \sum_{k=0}^{N-1} \tilde{x}(k) e^{-i2\pi n k / N} = \tilde{X}(n\Delta\omega)$$

Where the sum is now from 0 to $N-1$, since there are only N complex exponential terms (the others are the “same” due to aliasing

$e^{-i2\pi n(k+N)/N} = e^{-i2\pi n k / N}$), the n is from the Fourier series frequency domain discretization, and the k is from the time domain discretization.

Discrete Fourier Series (DFS) pairs.

$$\tilde{x}(n) = \frac{1}{N} \sum_{k=0}^{N-1} \tilde{X}(k) W_N^{-nk}$$

$$\tilde{X}(k) = \sum_{n=0}^{N-1} \tilde{x}(n) W_N^{nk}$$

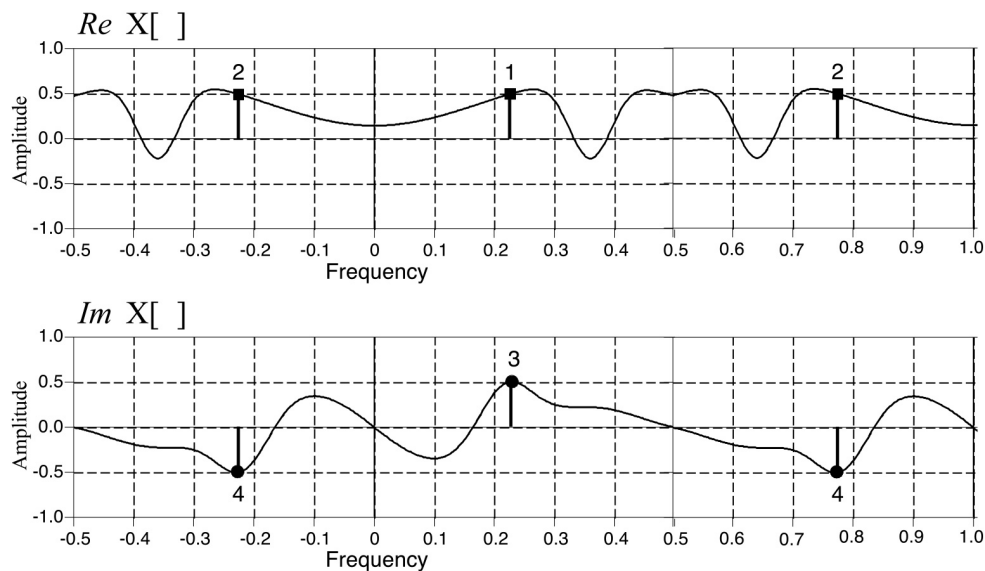
both of which have period N .

Summary for real $x(n)$.

$$\tilde{x}^*(n) \Leftrightarrow \begin{cases} \tilde{X}(k) = \tilde{X}^*(-k) \\ \Re[\tilde{X}(k)] = \Re[\tilde{X}(-k)] \\ \Im[\tilde{X}(k)] = -\Im[\tilde{X}(-k)] \\ |\tilde{X}(k)| = |\tilde{X}(-k)| \\ \arg[\tilde{X}(k)] = -\arg[\tilde{X}(-k)] \end{cases}$$

$$\tilde{x}_e^*(n) \Leftrightarrow \Re[\tilde{X}(k)]$$

$$\tilde{x}_o^*(n) \Leftrightarrow i\Im[\tilde{X}(k)]$$



DFT of real sequence showing symmetries of real and imaginary parts of complex DFT. One and a half cycles shown to see 2 most common x axis ranges -0.5 to 0.5 and 0 to 1 (in units of π).

With the proper interpretation we can directly change the DFS into the Discrete Fourier Transform (DFT). So far we have seen that we can represent a finite-duration sequence of length N by a periodic sequence with period N , one period of which is identical to the finite-duration sequence (we usually assume that the finite-duration sequence is in the region $0 \leq n \leq N-1$, but this is not necessary). In the sense that the periodic sequence has a unique DFS representation, so does the original finite-duration sequence, since we can compute a single period of the periodic sequence, and thus the finite-length sequence, from the DFS.

If we want to be rigorous - we should re-derive everything with the following modifications.

Define the rectangular sequence $R_N(n)$ (also known as a boxcar window) by

$$R_N(n) = \begin{cases} 1 & 0 \leq n \leq N - 1 \\ 0 & \textit{otherwise} \end{cases}$$

The finite duration sequence is defined by extracting one period.

$$x(n) = \begin{cases} \tilde{x}(n) & 0 \leq n \leq N - 1 \\ 0 & \textit{otherwise} \end{cases}$$

Which can be written

$$x(n) = \tilde{x}(n)R_N(n)$$

(Notice that we now have a **multiplication** in the ***TD*** by a “boxcar”, so we get convolution in the ***FD***).

How does the Discrete Fourier Transform (DFT) relate to the Continuous Fourier Transform (CFT)? This figure shows that we can consider the DFT to be a special case of the FT. (if one insists that discrete math is second class).

(A) shows the continuous function $x(t)$ and its continuous FT $X(f)$.

To determine the FT by digital means we first have to sample it. This is accomplished by multiplying the signal by the sampling function shown in (B). The sampling interval is T .

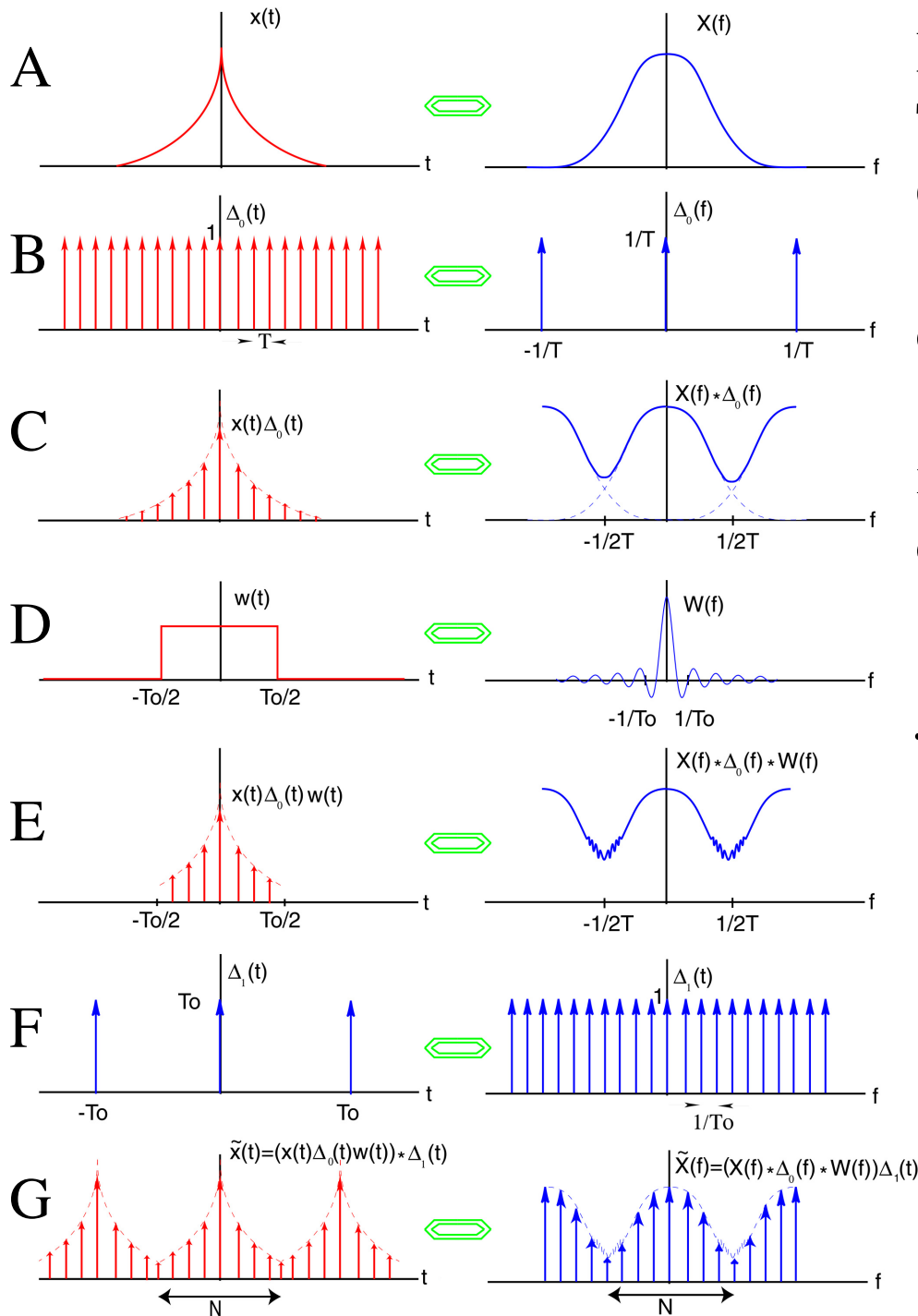
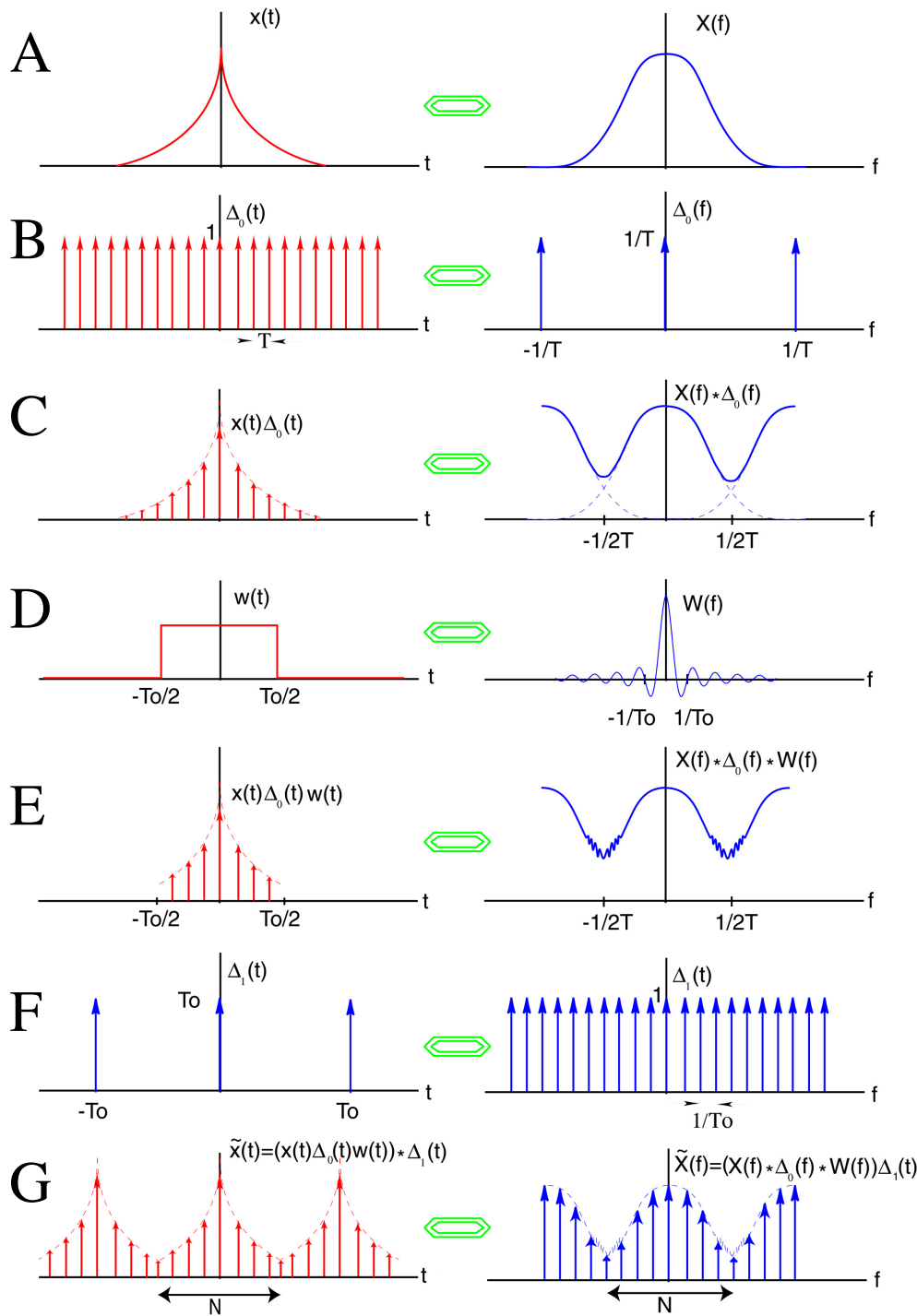
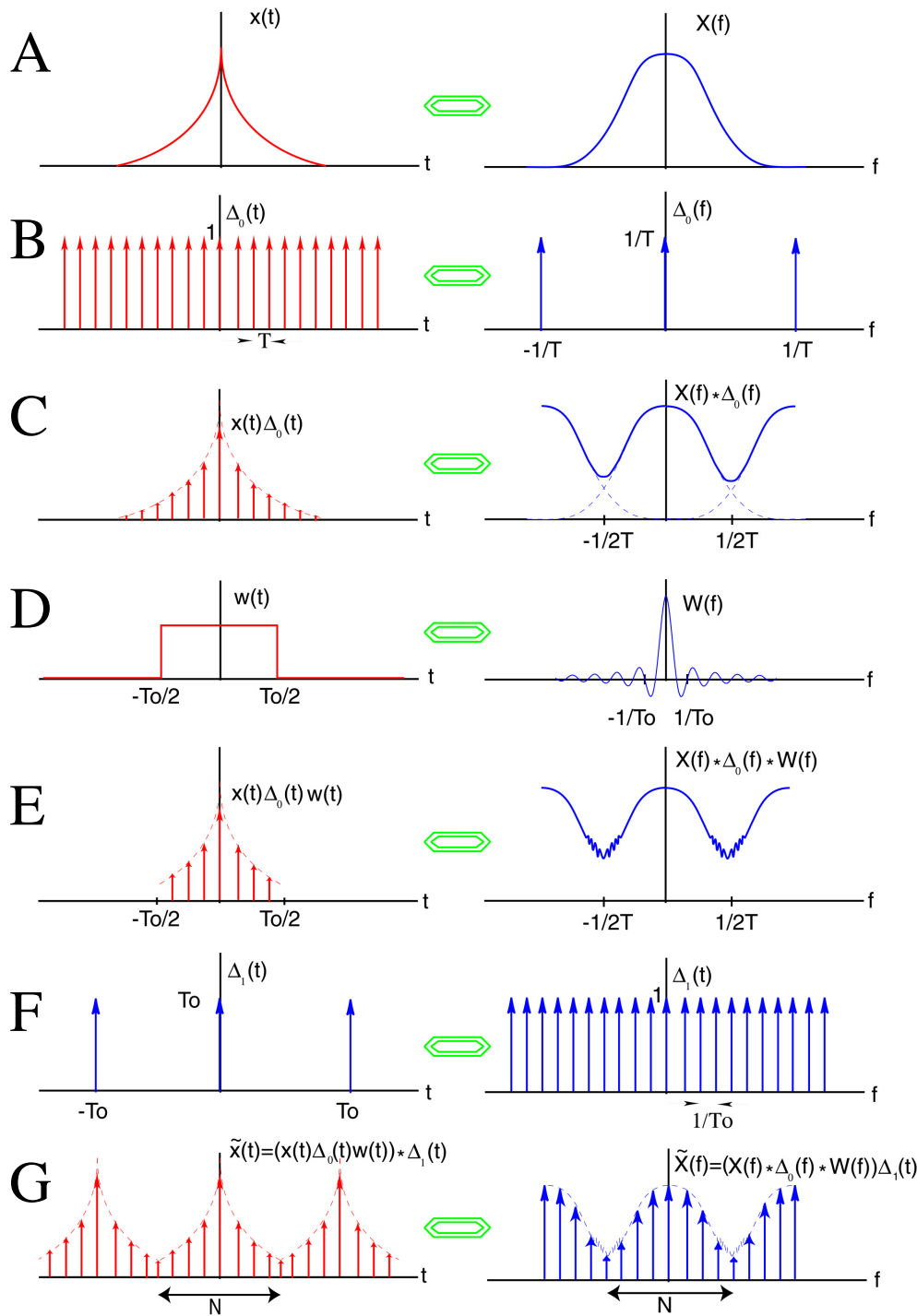


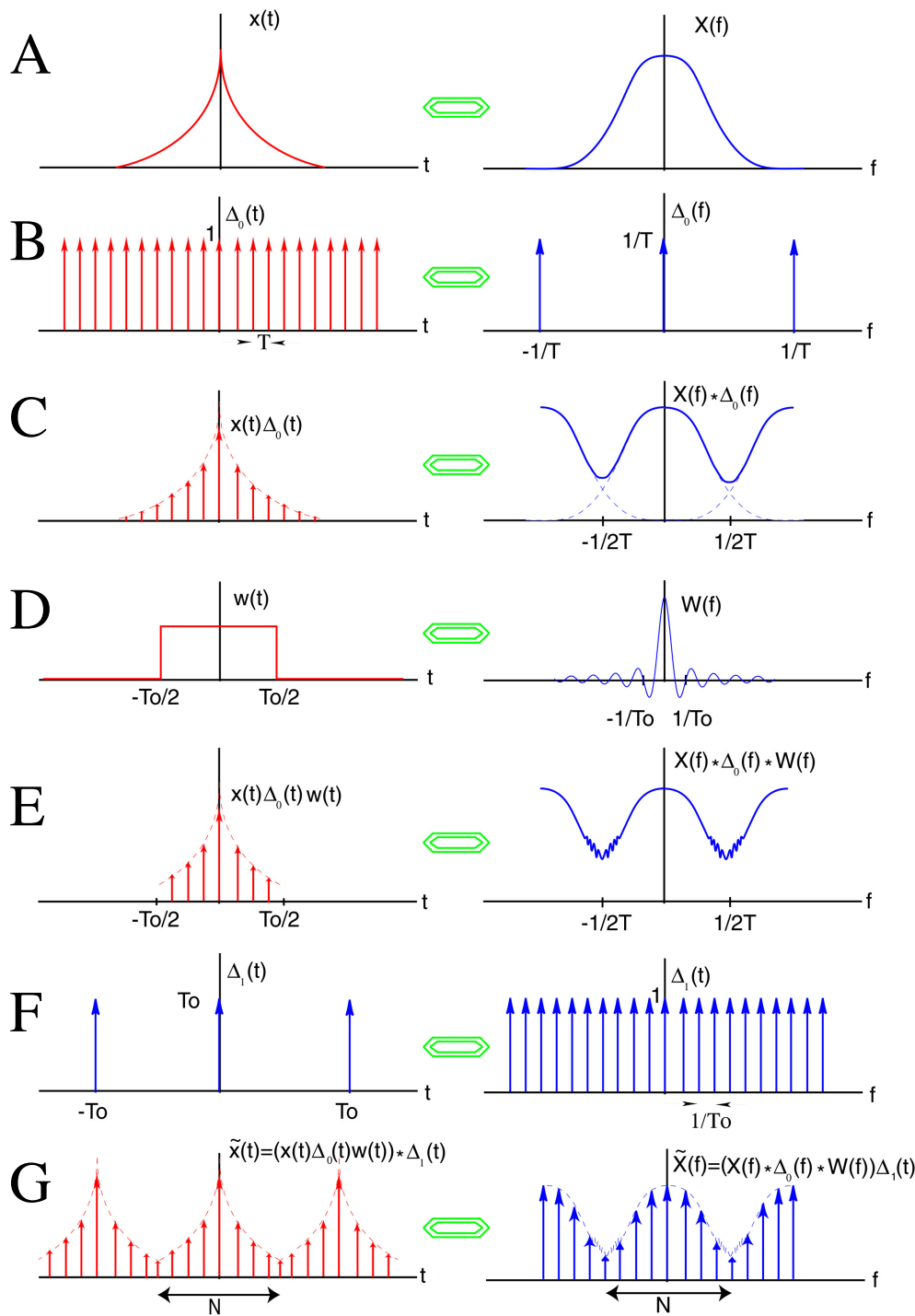
Figure after Brigham



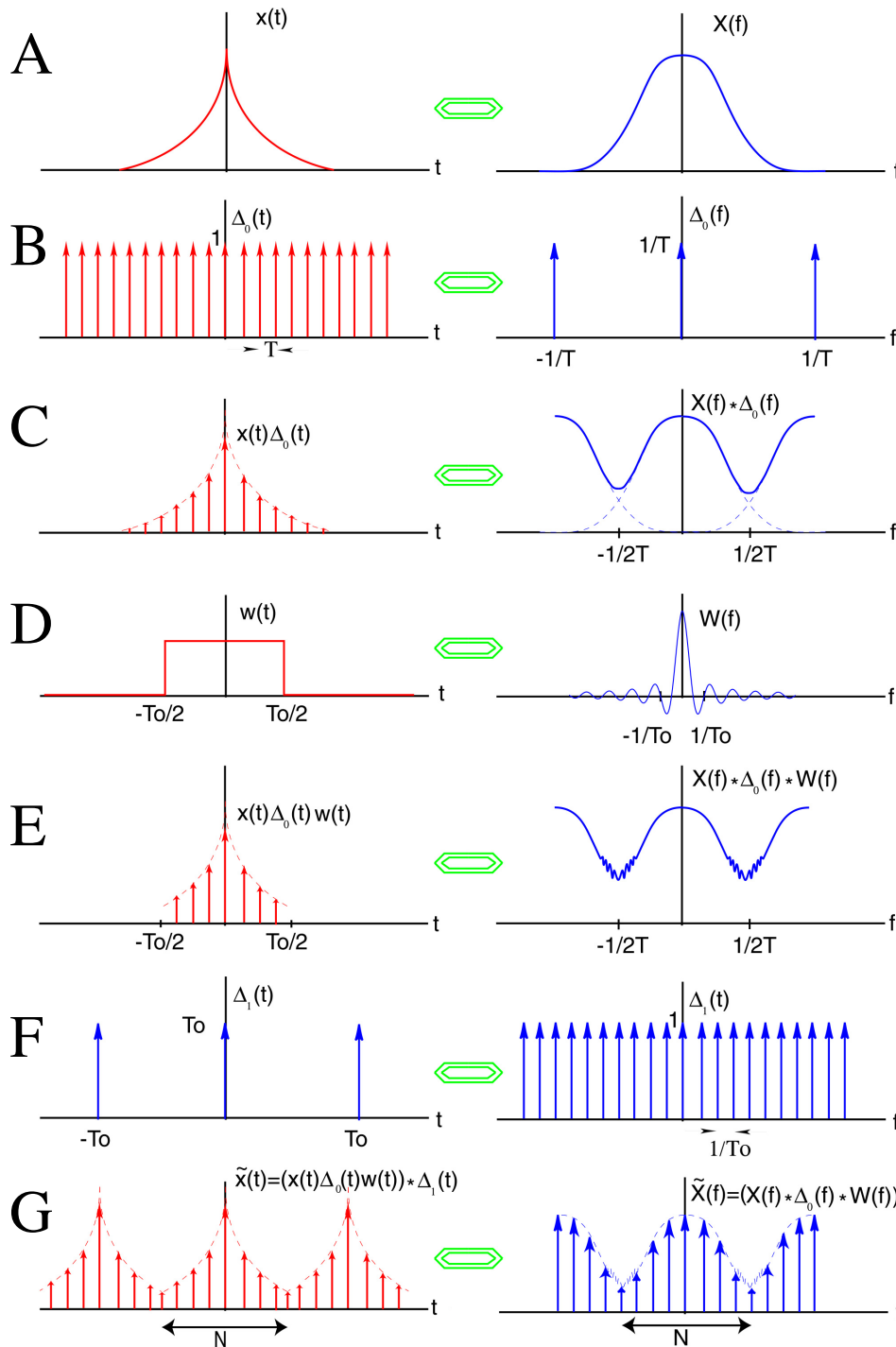
(C) shows the sampled function and its FT. The signal and FT are both changed. TD multiplication, forming the sampled signal, becomes FD convolution of the CFT of the continuous signal with the CFT of the sampling function making the CFT periodic. Note that in this case, the spikes in the CFD are not far enough apart to keep the repeated spectra apart, so they overlap and sum (aliasing). Since the spacing of the CFD spikes is $1/T_0$, one must sample faster to spread them apart and prevent aliasing (or remove the higher frequencies from the signal before sampling).



We cannot feed (C) into the computer yet, because it is infinitely long. We have to truncate it to a finite length. To do this, multiply by a rectangular (truncation or boxcar) window in the TD (D) producing the length N finite-duration, discrete TD signal in (E). This effects the FD representation, requiring a convolution with the sinc function (CFT of the rectangular window). This introduces ripple into the CFT due to the wiggles in the tails of the sinc function. We're still not ready for the computer however, because the FD signal is still continuous.¹¹

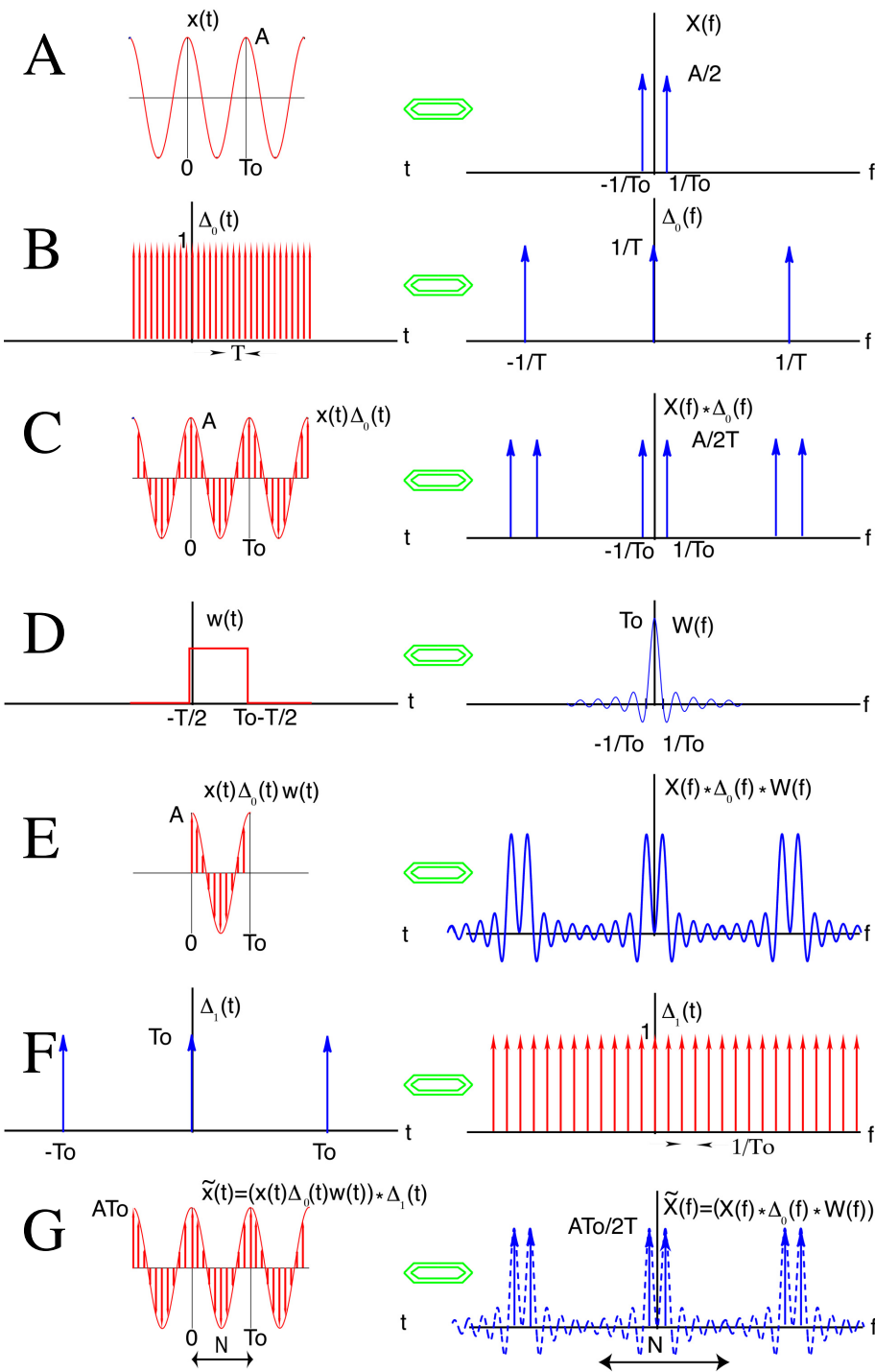


This is addressed in (F), where we now discretize the FD. As for the TD sampling, this is done with a comb of N (same length as TD) δ functions producing the DFT on the bottom right. FD multiplication requires convolution in the TD, so we convolve the finite-duration sampled TD sequence with the IFT of the FD sampling comb producing a periodic signal in the TD. In the computer, of course, we have only 1 cycle of the TD and FD representations and only positive frequencies. The math thinks things go to \pm infinity and includes negative frequencies¹².



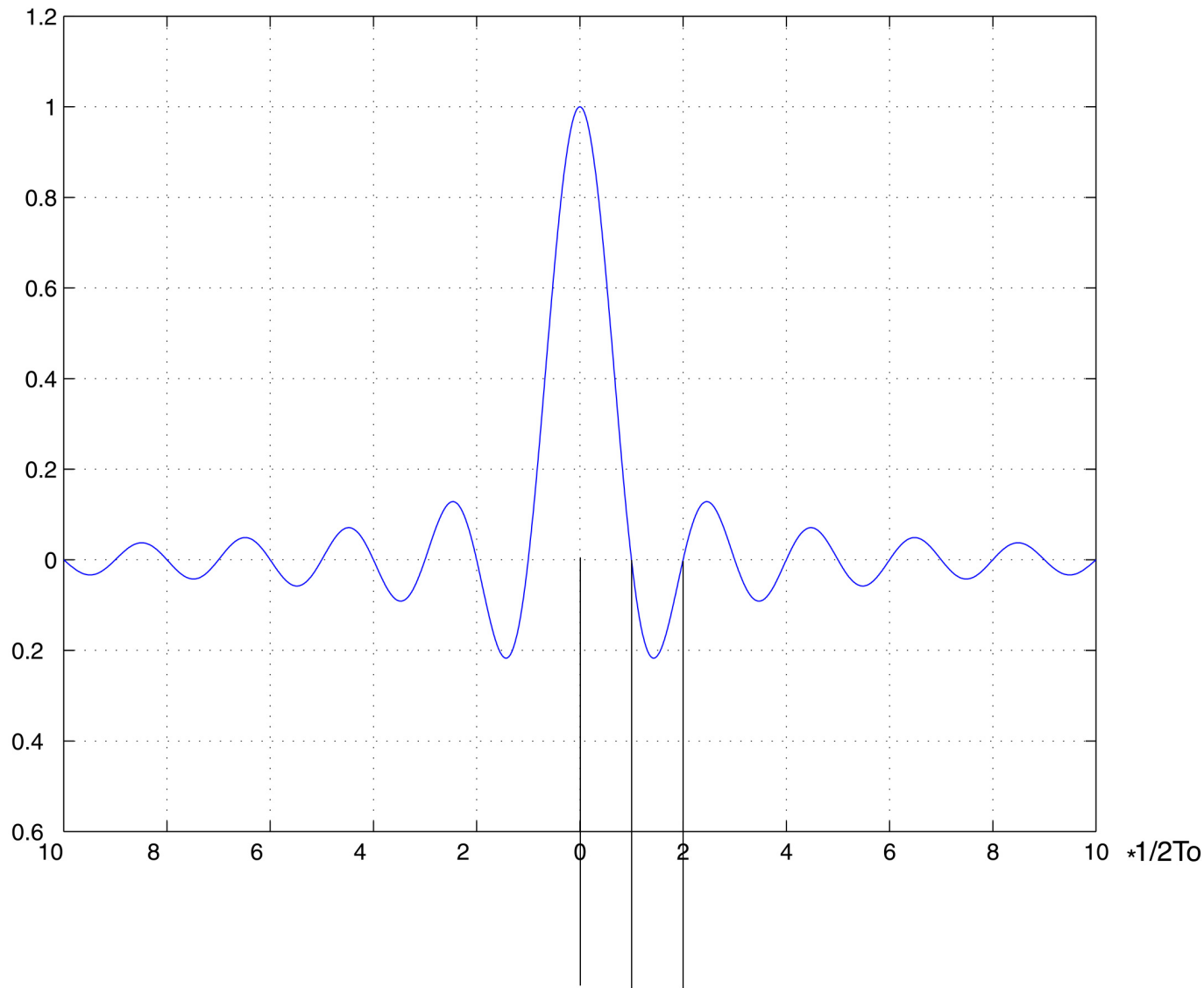
What about the ripples that showed up in (E)? Do they make it into (G)?

It depends (we can never have a simple answer!) on the frequency content in (A). As we will see, if the continuous signal just happens to contain only those frequencies contained in the comb of delta functions in (F), then the ripples all occur “between” the FD samples so they “don’t” exist (since discrete signals don’t exist there). If there is a continuum of frequencies in the continuous signal in (A) then the ripples will not all be zero on the FD samples and they will add into the DFT.



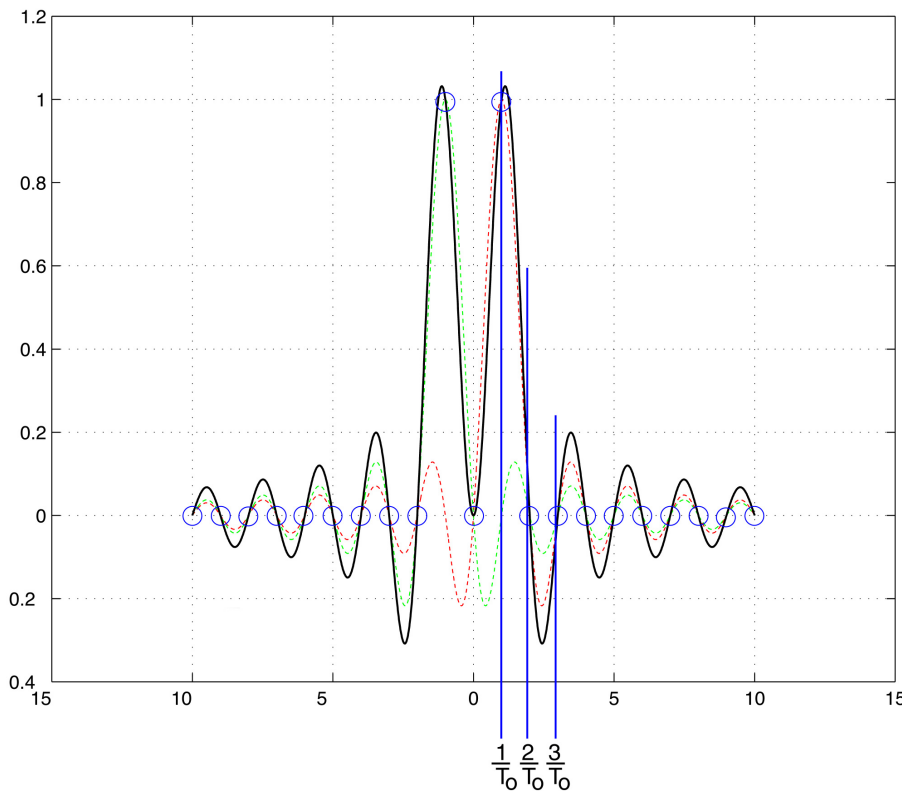
Truncation interval = period (“on” a DFT “line”). Same development as before. Minor change - the window function goes from $-T/2$ to $T_0-T/2$, this prevents the sampled TD data from having a point that is duplicated when (F) makes the TD periodic. Since there is only one frequency component we can follow the amplitudes exactly. There are 10 samples/cycle. (you will notice if you look carefully that the FD in (E) is not quite right - we need one more oscillation - two more zero crossings - in there [to be fixed in future {maybe}])

The TD representation in (G) looks the same as the TD analog signal in (A) since the analog signal has the same period at the DFT so we don't notice the imposed periodicity, while the FD shows this periodicity.

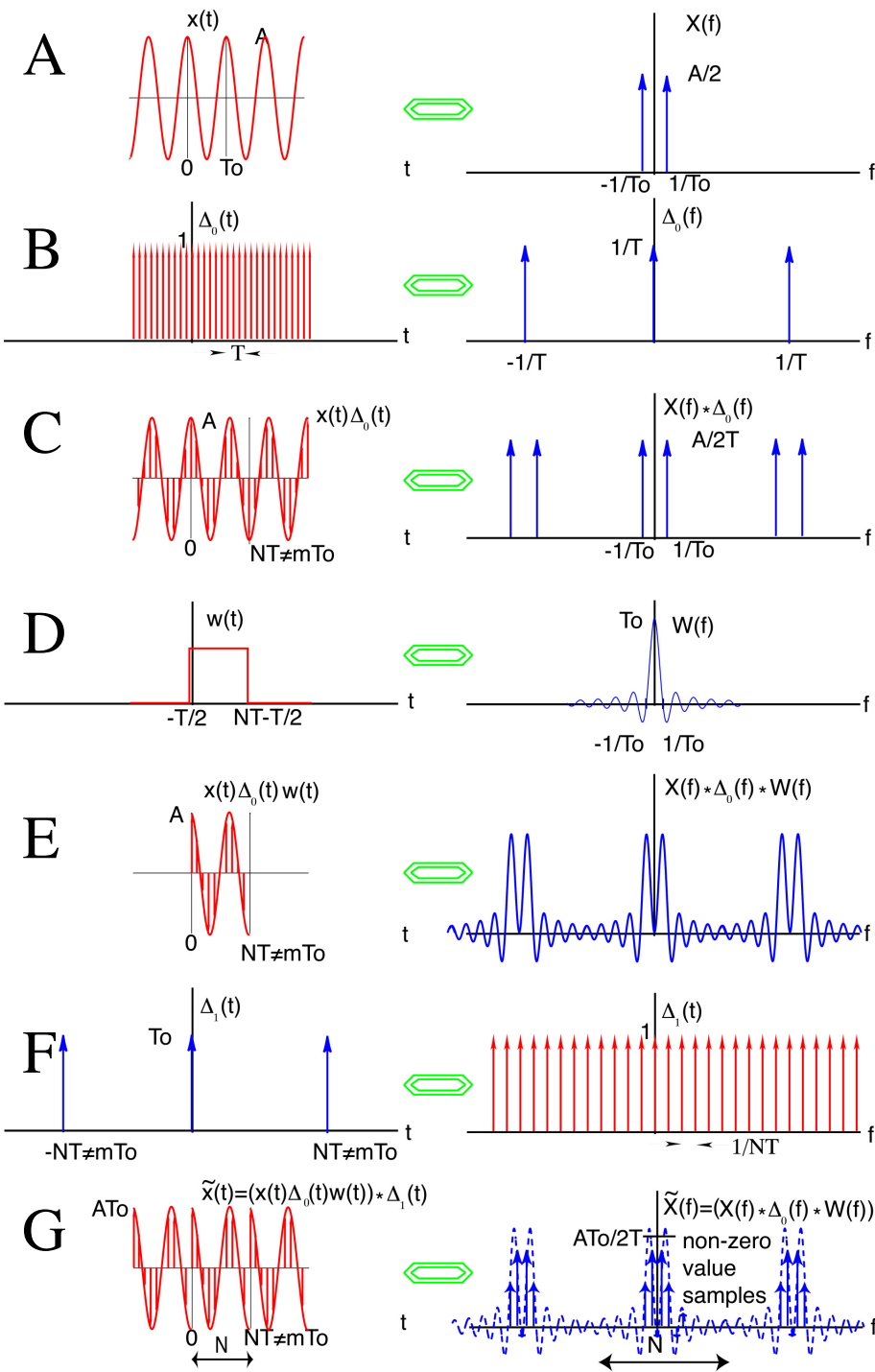


A quick look at the sinc function -- it is **1** at $x=0$, and zero at $x=n/2T_0$ for all integer $n \neq 0$ and oscillates around, while decaying to zero, in between.

FD zoom when the truncation interval = the period. Convolution of the two δ 's at $\pm f$ with the sinc results in the black line - somewhat ugly. The dashed red and green lines show the shifted sinc function components in the convolution that sum to this result. The blue lines show where the continuous (black) FT is sampled (multiplied by δ comb) in the FD to make the discrete spectrum shown by the blue circles. Except for the two samples at the "fundamental", the sinc functions and convolution are zero at the FD sampling points, while at the fundamental the convolution value is 1. We get exactly what we



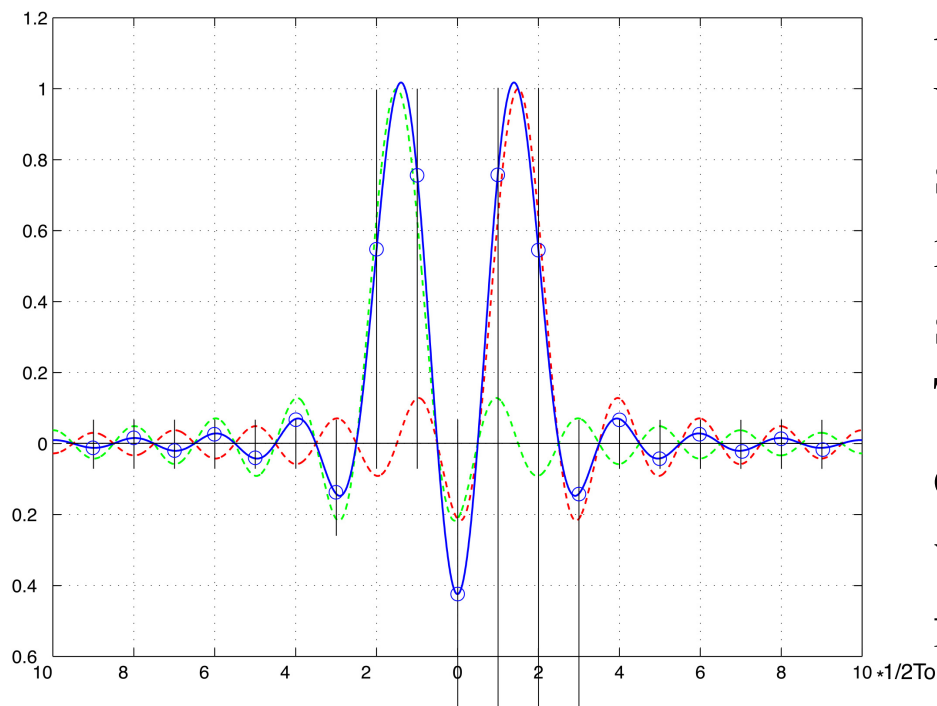
“expect” from our intuitive understanding of the real world and the CFT - a sharp single spectral line at the input frequency giving the amplitude of the signal (not worrying about difference between peak and RMS amplitude determinations or the 50-50 power split between + and - frequencies)



Truncation interval \neq period (input frequency not on DFT line). Same development as before. Since there is only one frequency component we can attempt to follow the amplitude exactly.

We now notice two “non-intuitive” (actually confusing) things - first the TD representation in (G) is not the same as the TD analog signal in (A) and we do not get a single spike in the DFT and the amplitude determination is “off”! This is a result of the “math” making a signal that is not periodic in N in the analog world, periodic in N in the discrete world on the computer.

Zooming-in in the FD we see the result of convolution of the continuous FT with the sinc function from the windowing. So far we get the same thing as before - the same ugly continuous function. The “problem” arises when we sample this with the δ comb in the FD. Now the continuous frequency locations of the fundamental and the zeros of the shifted sinc functions and convolution do not correspond to locations of the δ 's in the comb - so the ugliness carries through the sampling into our result and we get two points in the sequence with large “incorrect” amplitude values and lots of points in the sequence with non-zero values.



Putting a little more effort into understanding this it actually makes sense. The DC value is not zero for instance. Remember that the sequence is “periodic” in the TD. The DC value is the average over one window cycle of the TD signal, which is not zero for arbitrary frequency cosine waves.

To sum up, the effect of truncation of a TD sinusoidal signal at other than multiples of the period of the signal is to create a periodic TD function with sharp discontinuities. Intuitively we expect the addition of these discontinuities in the TD to result in the need for additional (high) frequency components in the FD. In the FD, the TD truncation is equivalent to convolution with the sinc function. After this convolution in the FD we no longer have a single impulse, but a “smooth” continuous function with local maximum (correctly) centered at the original impulse and a series of other peaks called sidelobes. The next “bad” effect comes with FD sampling, which samples this mess based on the TD truncation period and not the sinusoidal signal period resulting in a multiple spike peak and low amplitude tails. The effect whereby the energy is spread-out over multiple samples is called “spectral leakage”. This effect is inherent in the DFT because of the required TD truncation (it is aliasing – need high frequencies for the discontinuity). We will look at some techniques for reducing spectral leakage (at a cost) later (you already have seen this - windows that remove/smooth out the discontinuities in the periodic view of the TD).

If the signal to be analyzed is finite duration or periodic, one can skip the “windowing” step by looking at a sequence with a length equal to the finite duration of the signal or one period and get the “right” answer in the “math” view in both the TD and FD (remember what happened when we sampled the “ugly” convolution result for the case of a cosine wave truncated at its period - it removed the ugliness by sampling only the “good” points). If the original signal is not band-limited, however, we would get aliasing errors.

For a general signal the DFT gets hit with a double whammy - one from the truncation effect and one from aliasing. We already “know” (but have not looked in detail at) how to fix both problems

Truncation - use a smooth window function (not a boxcar)

Aliasing problem - faster sampling and anti-aliasing filtering before sampling – can’t fix discontinuity – fix same way – smoothing window.

Let's look at what can happen when we decide to do DSP on the computer. We have some sequence and would like to know its spectrum. Assume it's a GDSN seismogram, so we don't have to worry about the a-d hardware, sampling, bits/sample, etc. We have a good, properly prepared signal to process. Let's further assume that this signal is a pure sine wave (the "spherical chicken" of DSP).

The first question we have to answer is how much data to process. The DFT needs TD data to \pm infinity and a continuum of FD frequencies, which we can't do on the computer. So we need to make some decisions and compromises. We know we need to sample at least two samples at the sinusoids extrema to get the amplitude and that the more samples we have the better for the DFT, but we run into computer space and calculation time problems. So we compromise on some finite duration sequence that gets a number of cycles (say between 10 and 1000)

Next we have to decide on the number of frequencies we will use in our finite sum. We get the fundamental frequency from the sampling rate.

We will now calculate some DFTs using the definition of the DFT (the slow - correlation method).

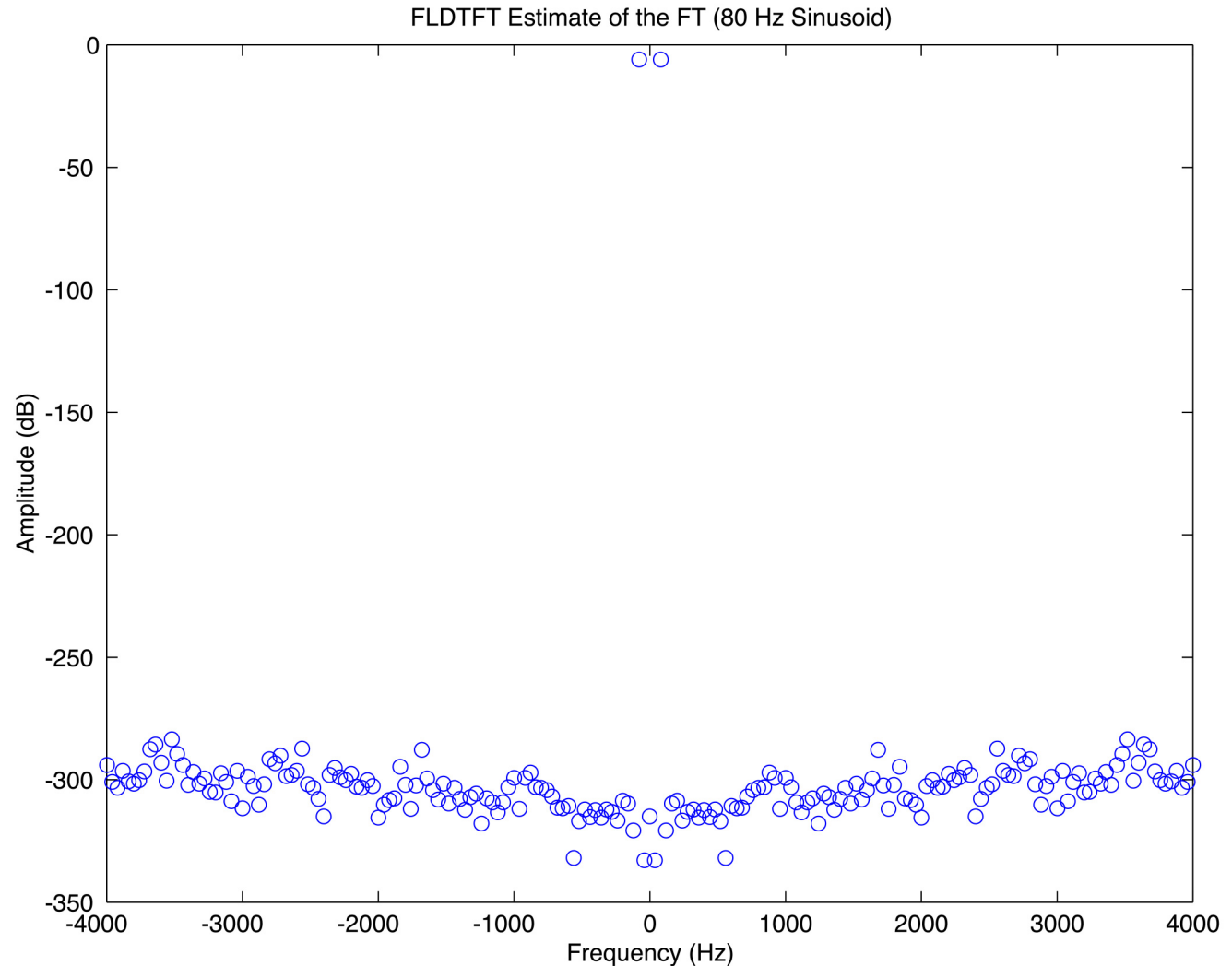
Take an 80 Hz sinewave and sample at 8kHz (that's 100 samples per cycle) for 100 msec (that's 8 cycles) for a total of 801 samples.

Given the sampling frequency we know that the spectrum will only have frequencies between ± 4 kHz. Since we can't calculate the continuous spectrum, let's pick 200 of them (notice that we're doing something non-standard here - for educational purposes, normally we'd use 801 frequencies for an 801 point complex input sequence), which gives a fundamental frequency of 40 Hz and a spacing in the discrete FD of 40 Hz. Notice that 80 Hz is an even multiple of the fundamental.

We now compute
$$X(n) = \sum_{k=0}^{800} x(k)e^{-i2\pi nk/801}$$
 for the 201 frequencies

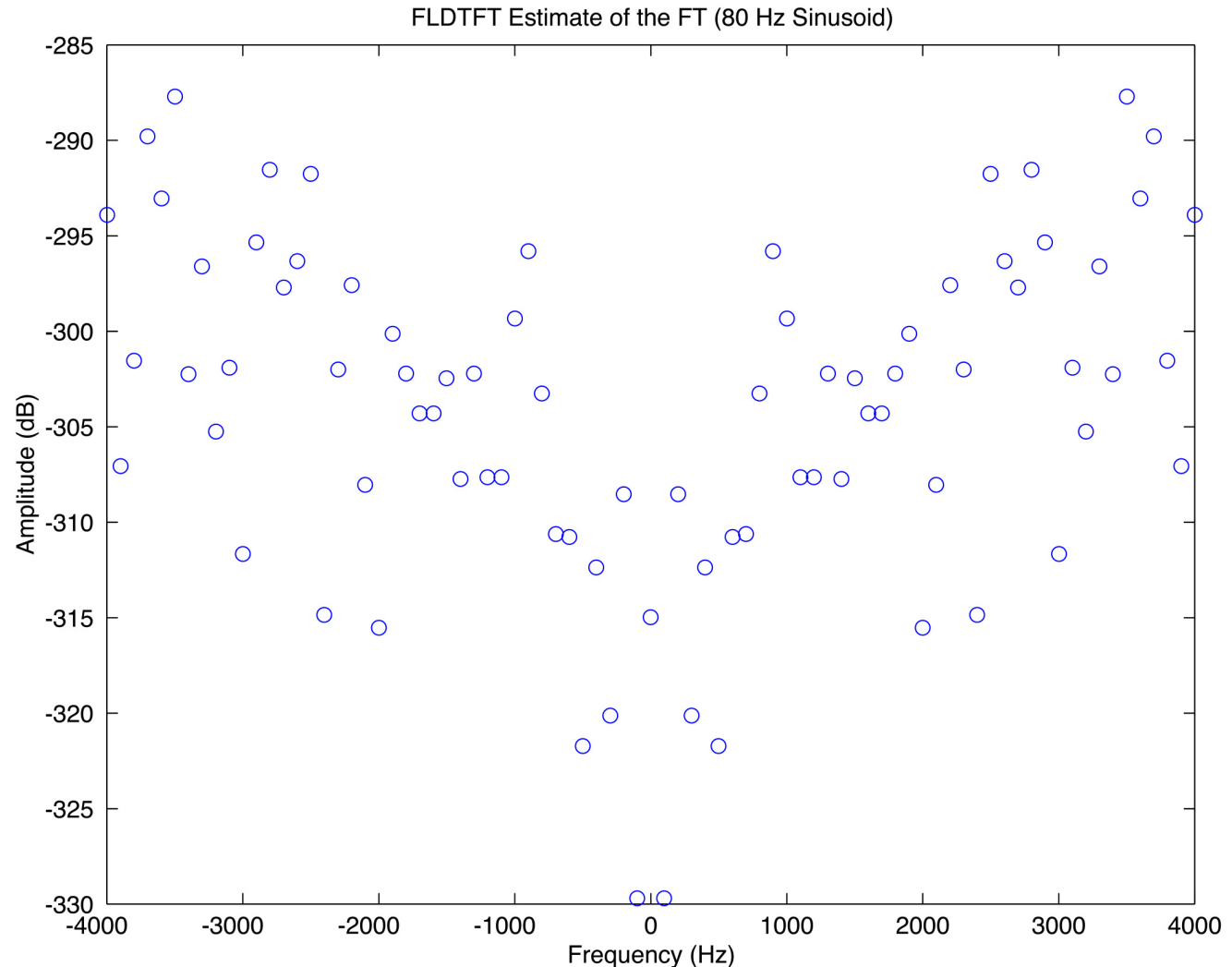
between -4kHz and 4kHz ($-100 \leq n < 100$) and get.

We get a near “perfect” result, all the energy is basically in two lines at ± 80 Hz. The stuff down at -300 dB is effectively zero and is due to round off error associated with the finite precision math (MATLAB uses double precision).



The result in the previous plot was so nice because we cheated - it was “set-up” to come out that way. We knew the frequency of the input and “ginned-up” the spectrum calculation to include the known frequency in the set of frequencies used for the correlation's. Suppose we did not know what the input frequency was and arbitrarily decided to sample the FD every 100 Hz instead of every 40 Hz? (again for educational purposes). Now what do we get?

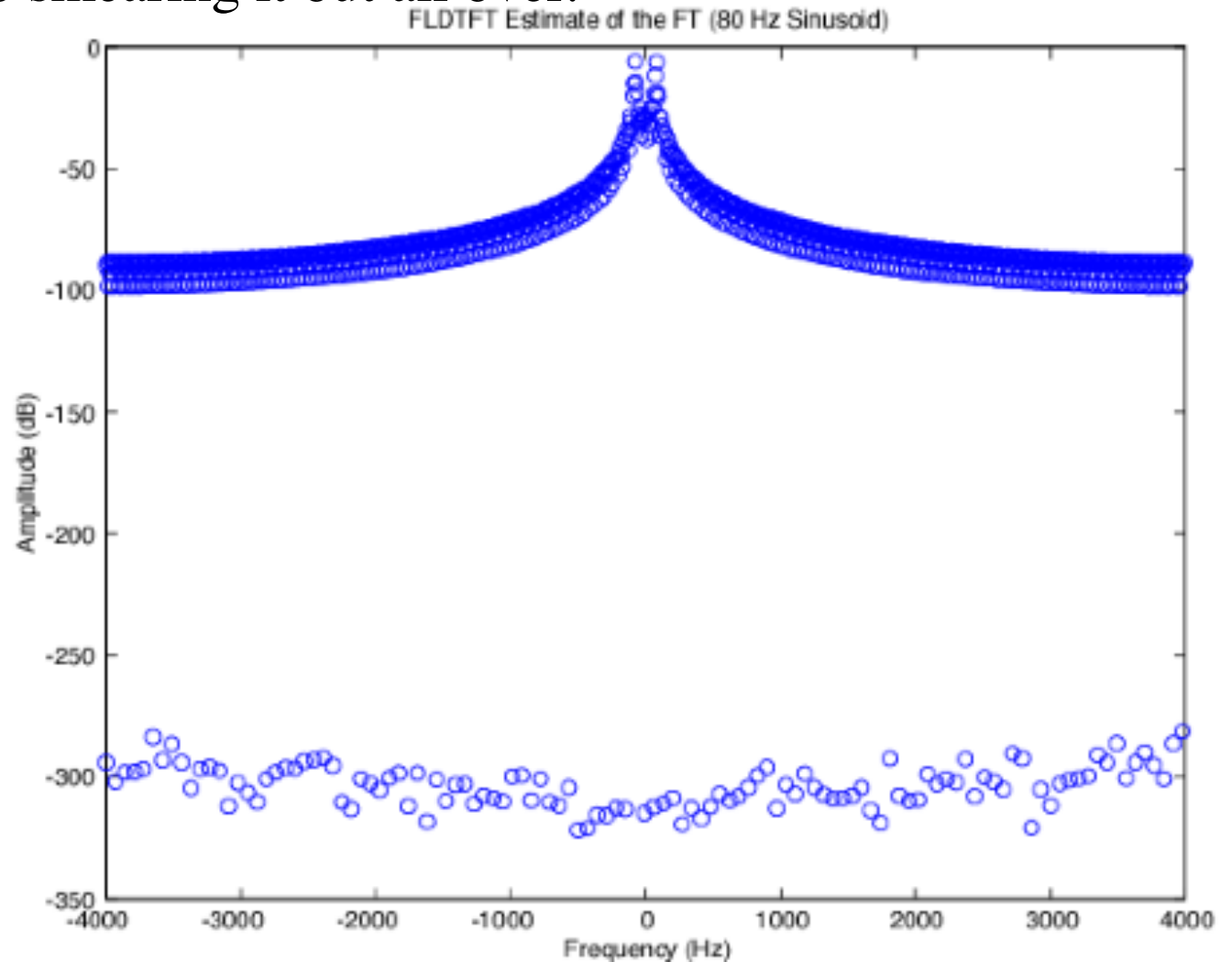
WOW! Where'd our sine wave go? Everything's down at -300 dB, i.e. there's no signal. But we know we put one in. Remember that we're correlating the input with a set of sinusoidal waves. Also remember that sinusoidal waves are orthonormal, so we get ZERO (or near zero) for all of our correlation's, since the frequencies we're using for a basis are every 100 Hz, none match up with 80 Hz. This is pretty bad - we know we put in a signal and get NOTHING out.



Ok, so we understand why we don't see the 80 Hz sine wave, we did not sample at a sub-harmonic of 80 Hz (so that $n * \textit{fundamental} = 80 \text{ Hz}$). But what does this imply for taking measurements in the real world where our inputs will most likely be unknown frequencies from a continuous spectrum? Can we really mess up so badly?

Let's try sampling in the FD at 7 Hz to see what happens.

WOW again! We went from the frying pan into the fire. First we saw no sine wave in the output and now we see them everywhere (plus some frequencies with zero - the guy's down at -300 dB). Granted we do have peaks at ± 80 Hz, but what's all that other stuff? So we went from missing our input altogether to smearing it out all over.



Ok - so what's going on.

First - the previous examples were meant to scare you.

When you look at the equations, correlation over some finite set of frequencies seems like a reasonable way to approximate the DTFT. In practice this will produce

Good solutions - Showing all terms, and at the right frequencies

Bad solutions - Missing important terms

Ugly solutions - Moving important terms to the wrong frequencies

(let's call our correlation method the Finite-Length-Discrete-Time-Fourier-Transform = FLDTFT)

To confidently predict the FT from the DFT/FFT/DFS/FLDTFT you need to know what happened in these examples. Fortunately there is a fairly easy explanation

Recall there are two problems that prevented us from calculating the DTFT - we don't have a signal over all time, and we can't evaluate the DTFT at all frequencies.

Start by examining the first problem. We only have $x[n]$ over a finite time interval, but we usually assume the analog signal, $x(t)$, extends out to $t=\pm\textit{infinity}$ (or at least extends for a lot longer time than we used in the FLDTFT calculation). We essentially multiplied $x(t)$ by a 'window' that looks like a square pulse by ignoring data outside some finite time interval in our calculation of the FLDTFT,.

Rather than looking at the difference between the DTFT and FLDTFT calculations - look at the difference between the spectrum of $x(t)$ and the spectrum of $x(t)w(t)$.

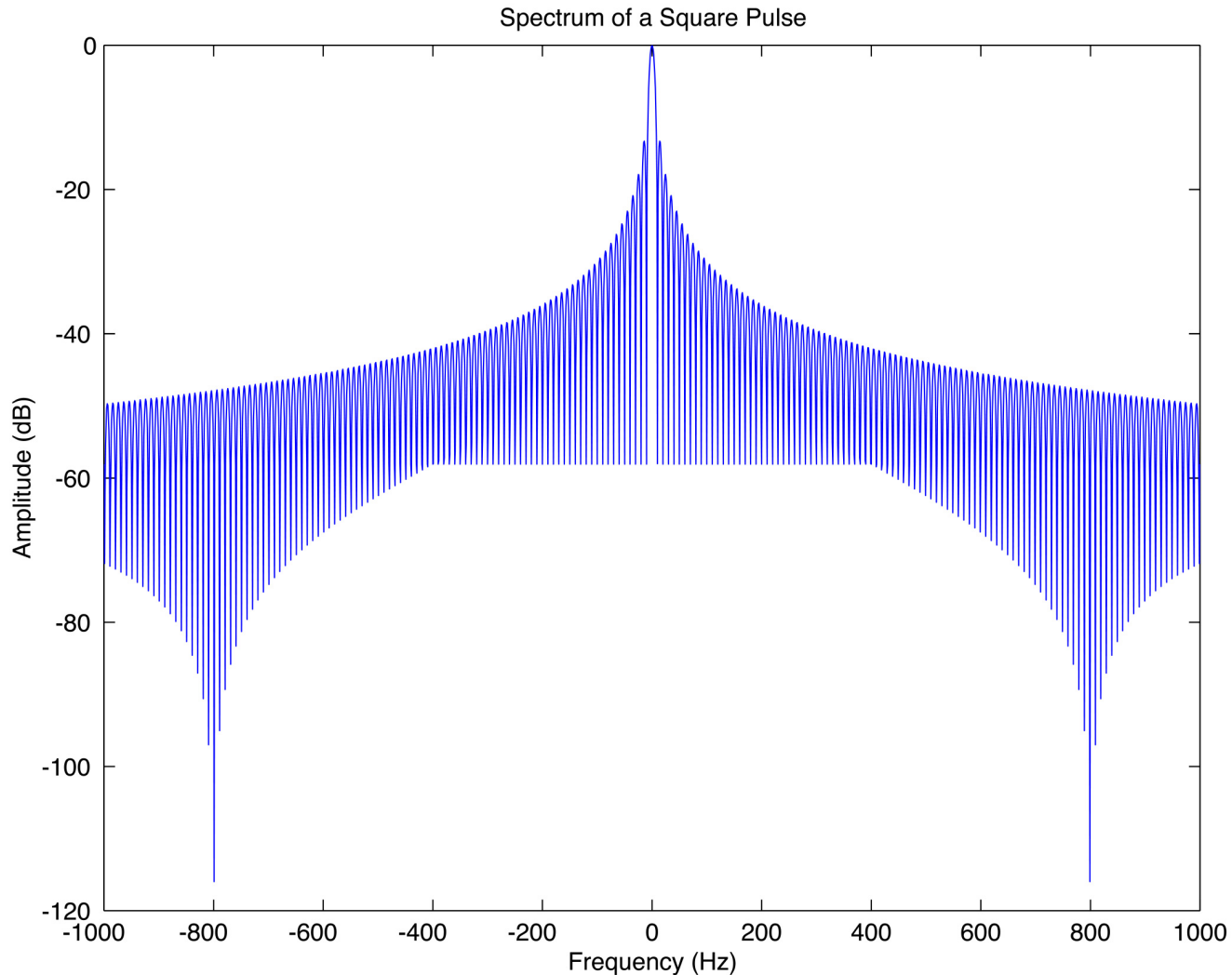
Recall that multiplication in the TD, is convolution in the FD. Also recall that the spectrum of a square pulse (the truncation window) is a sinc function.

If the pulse has a width of T_p seconds, the sinc spectrum will hit zero every $1/T_p$ Hz. In our example the window was 100 msec long, so it's spectrum will hit zero every 10 Hz.

To get a clear view of this spectrum, just take the FLDTFT of a constant

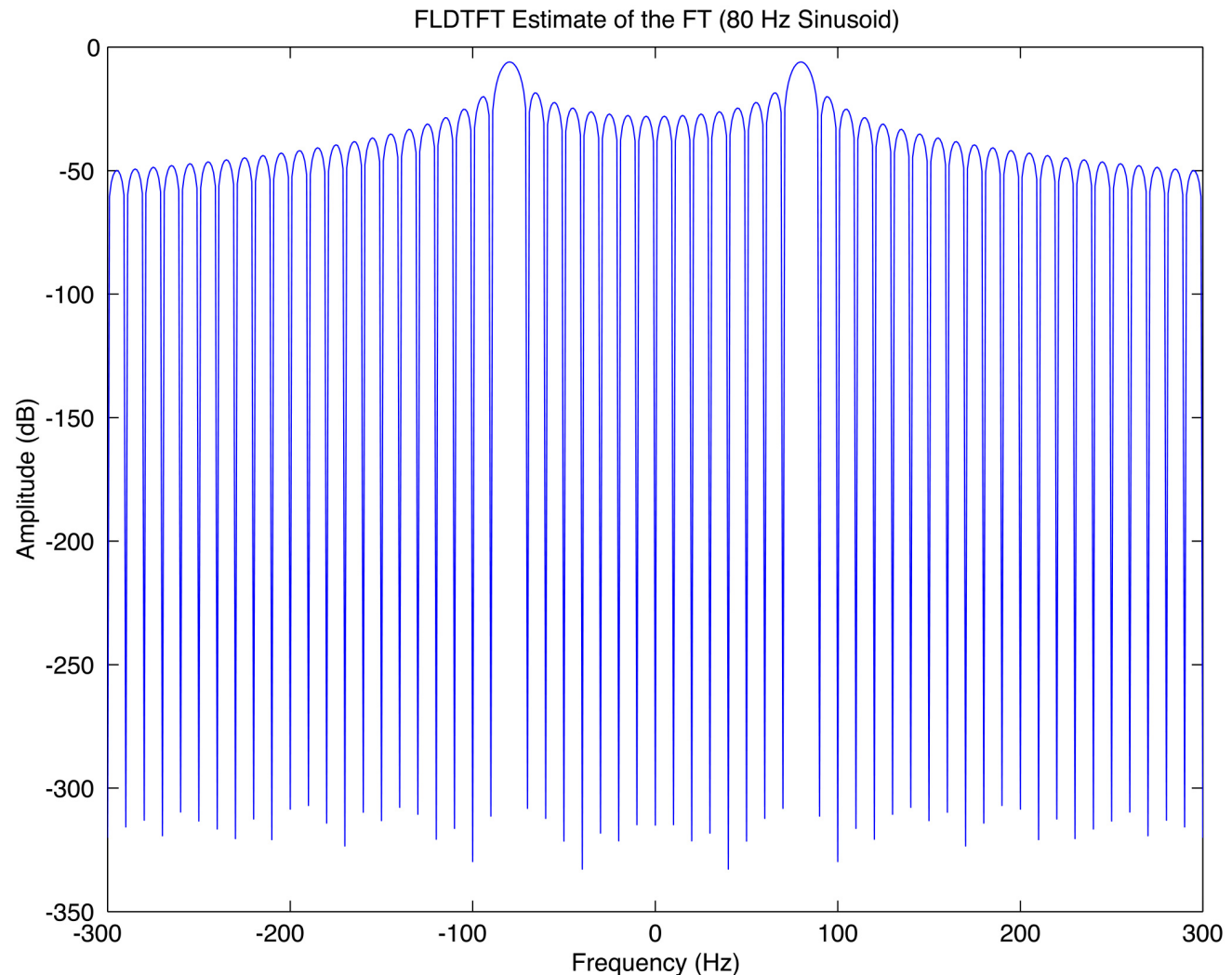
(notice that we're repeating what we saw in the multi-panel CFT to DFT development)

Spectrum of the truncation function (boxcar, rectangular pulse) that results from changing a infinite duration signal into a finite duration signal. (no sampling yet).



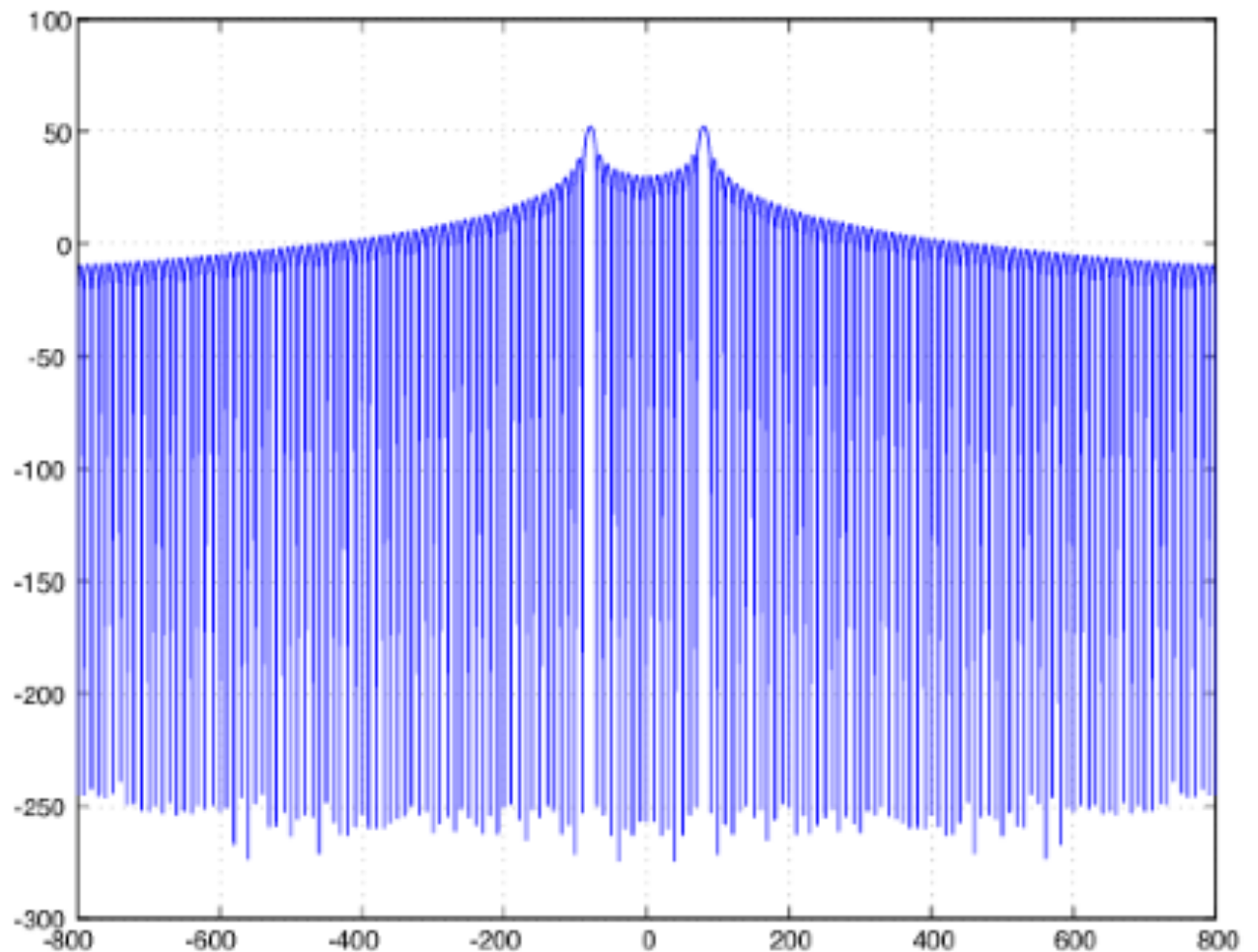
Convolving the sinc function with the deltas changes the original spectrum of $x(t)$. When $x(t)$ is a sinusoid this is easy. Using the numbers in this example, the spectrum of $x(t)$ is just two impulses, at ± 80 Hz. When these impulses are convolved with a sinc function, we should see

two copies of the sinc function, centered at ± 80 Hz. These sinc functions will hit zero every 10 Hz, at: ... -110, 100, -90, -70, -60, -50 50, 60, 70, 80, 90, 100, 110 ... Hz.

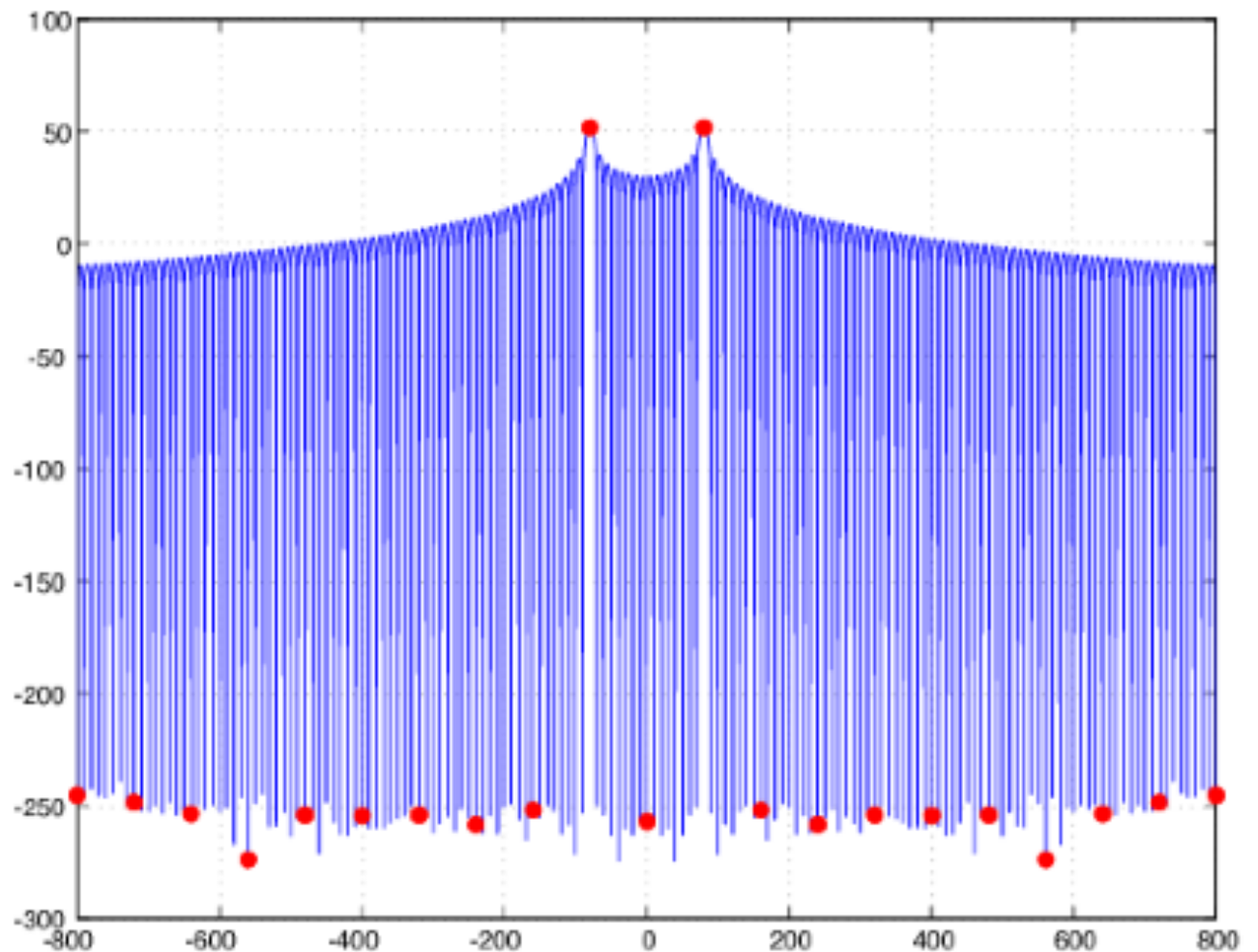


So now we can explain exactly what happened in our 3 cases and relate it to the “true” spectrum.

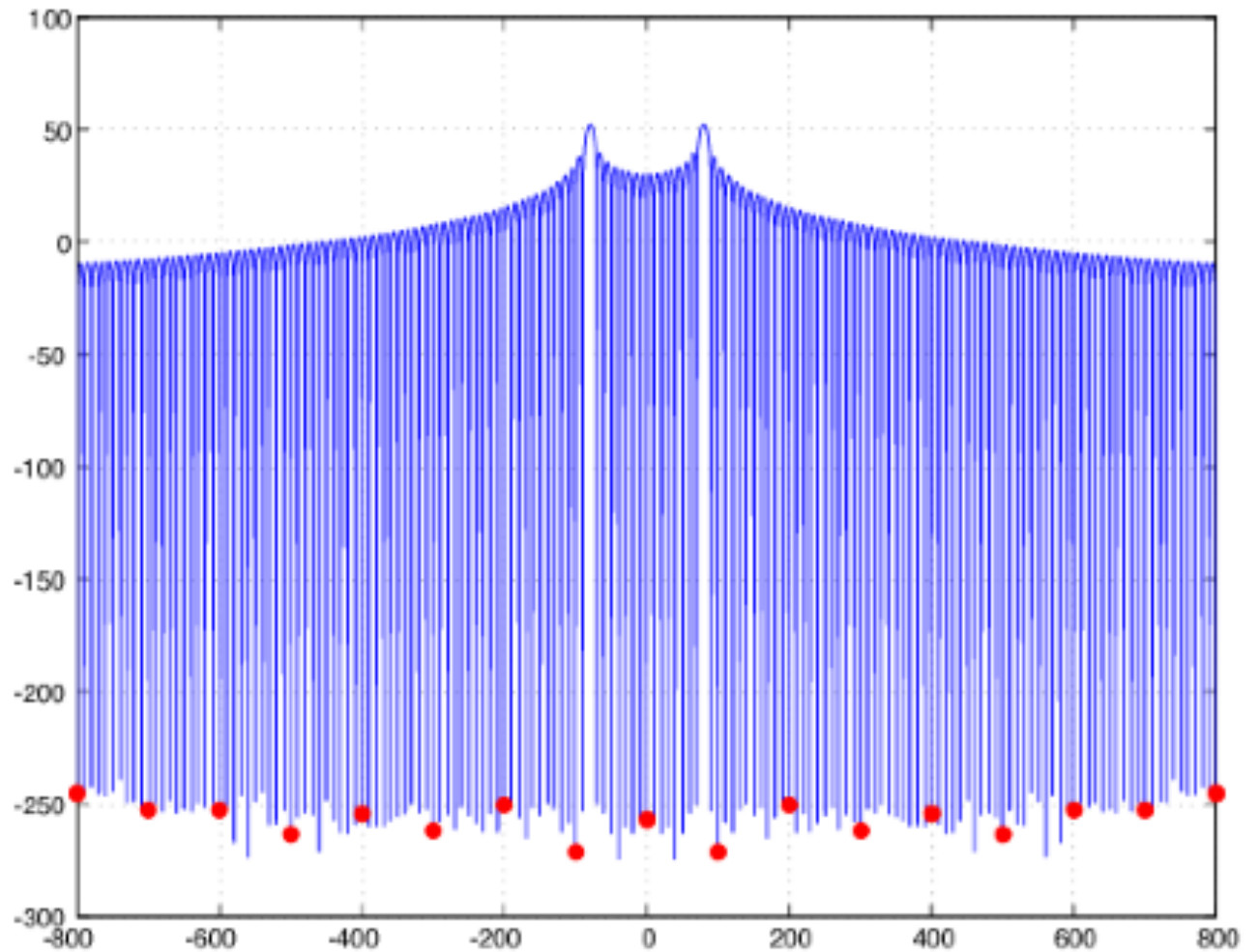
True (no FD sampling, is continuous [well it looks continuous, it's still done on the computer so it can't be continuous - so just pretend]). Take enough frequencies in the FD to see the full $x(t)w(t)$ (i.e. both the signal and weighting function)



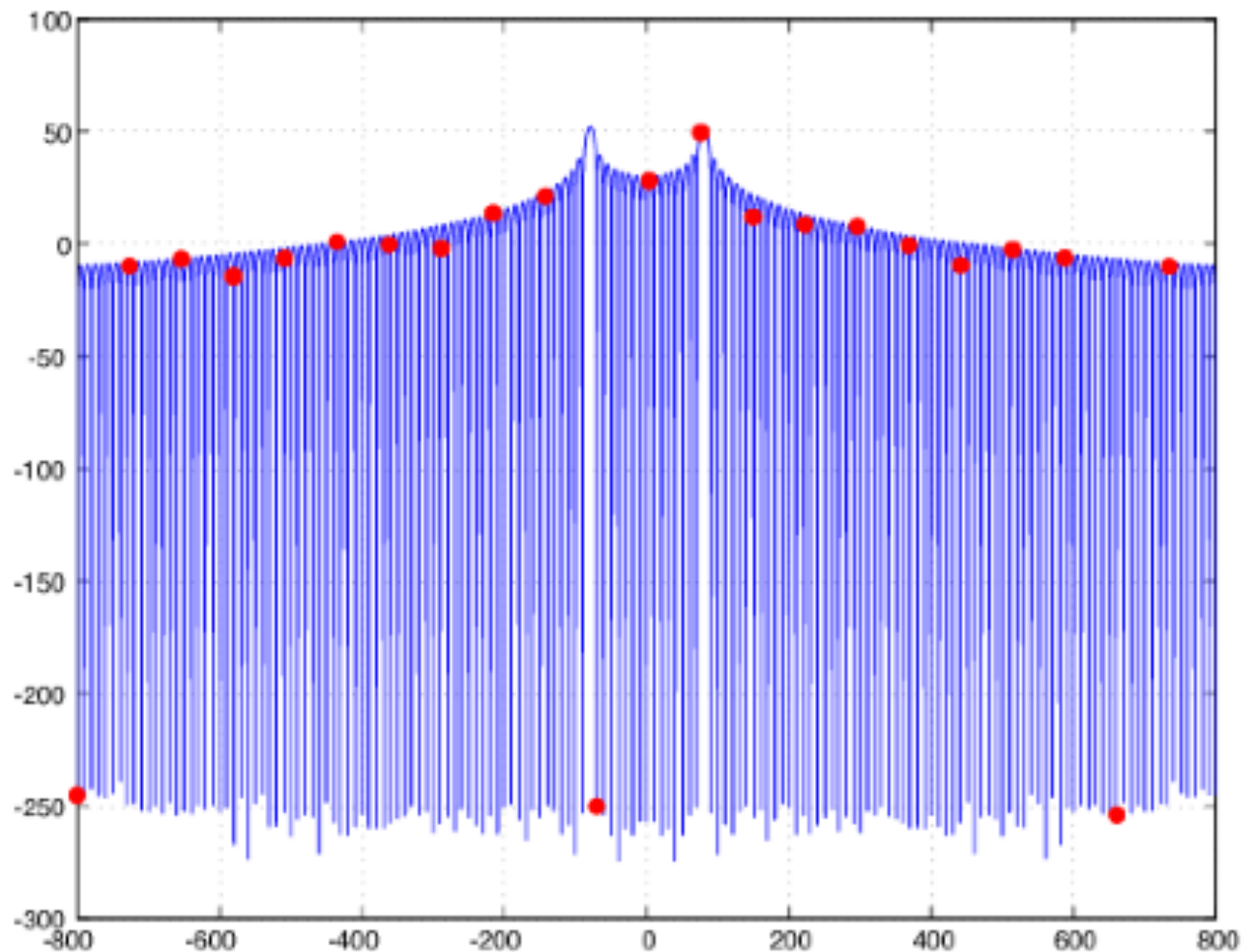
If we only sample the sinc function at the peak and nulls we get the “excellent” result shown by the red dots. Unfortunately this requires knowing the answer beforehand or lots of luck.



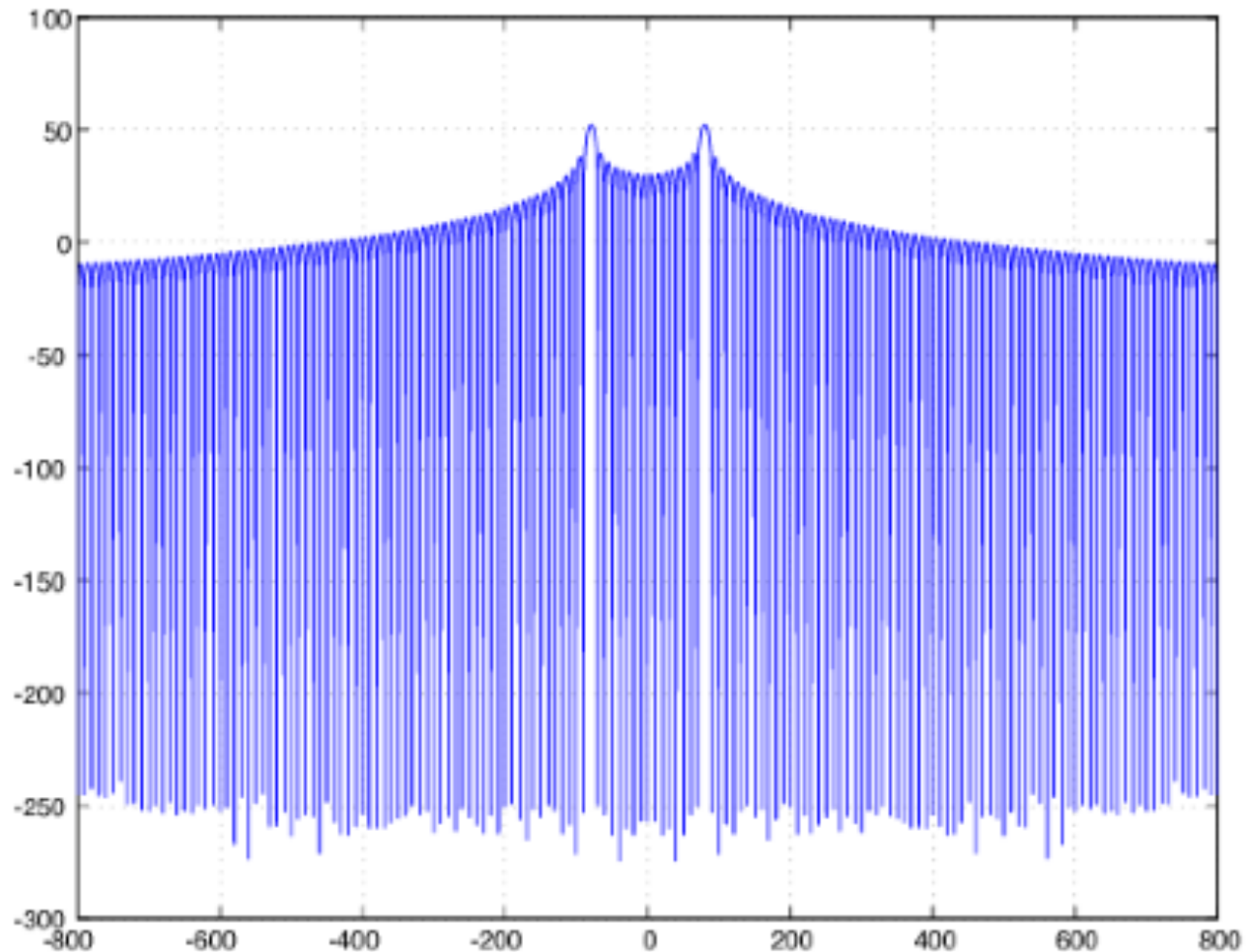
If we're unlucky enough to sample the spectrum only at the zeros of the sinc function we completely miss the input!



And if we sample randomly we might get the peak (if we sample on one of the sinc peaks), we move frequencies around by sampling on the sinc ripples, and we get zeros when sampling on the zeros of the sinc. YECH.



Another thing to worry about -- those pesky negative frequencies. Since the sinc function goes to \pm infinity, when we do the convolution of the truncation function with the deltas (for a single sinusoidal input) the tail going to +infinity of the sinc function coming from the negative frequency spike can “contaminate” the positive frequency spectrum (and similarly for the negative side of the spectrum).



The “educational” modification we made to the implementation of the FLDTFT was to select the frequencies in the FD. Normally these are “set”. But the truncation window effects are still there.

When you calculate a spectrum of a real-world signal, you should ALWAYS think about the finite window length effects. The spectrum you see on the computer screen (or the the screen of a spectrum analyzer) will be the convolution of the true spectrum with the spectrum of the weighting window. In the simple case where you use a rectangular window (the approach described on this page), the true spectrum will be convolved with a sinc function that has nulls every $1/T$ Hz (where T is the length of time you observed the signal in seconds). If you want to get a clear picture of the spectrum, you will need to look at frequency samples spaced every $1/10T$ to $1/100T$ Hz apart. If you ignore this effect - you may get very pretty, or very ugly spectra. At low frequencies, you need to remember that the sinc from the negative frequencies may alter the spectrum at positive frequencies.

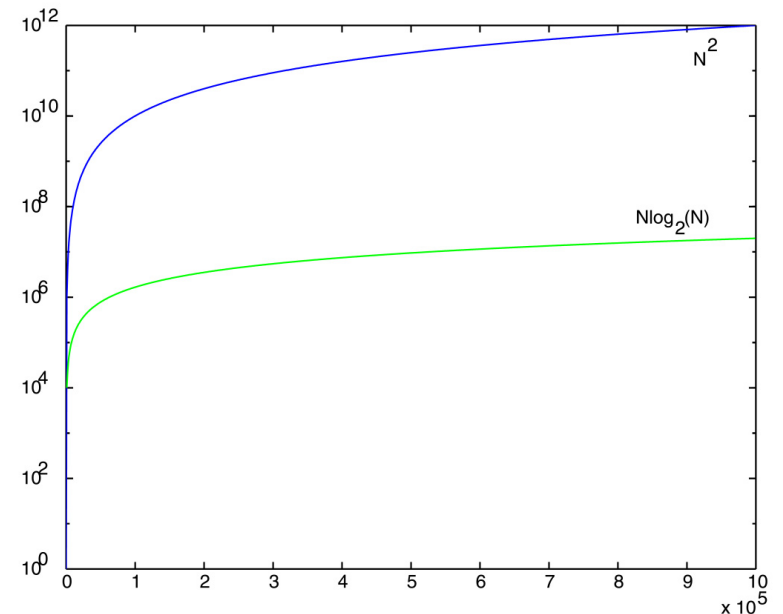
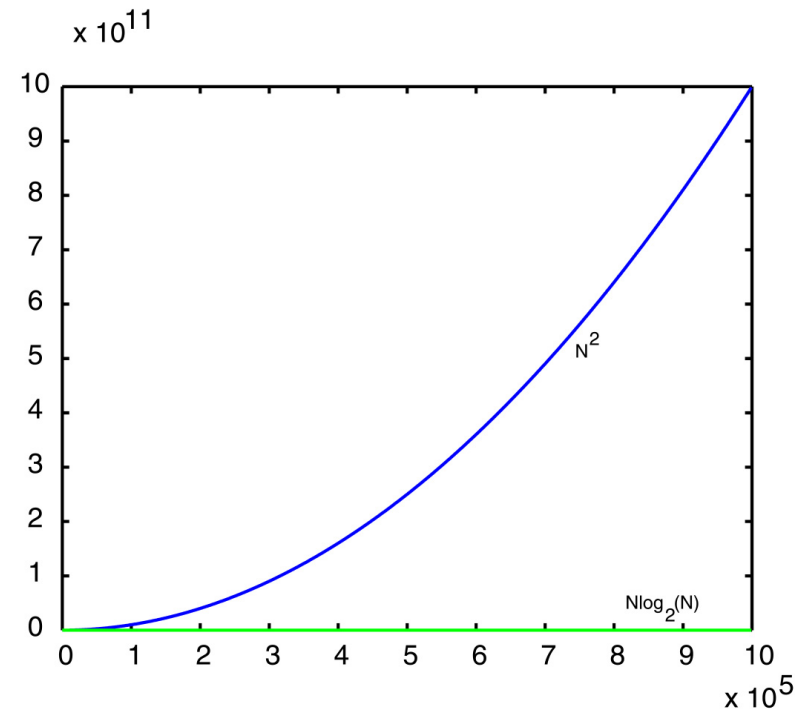
All this stuff is directly applicable to DSP in seismology . How long of a record must one analyze to resolve the frequencies of interest?

Comparison of number of complex multiplication's (equivalent to time) needed for DFT and FFT for sequences up to a million points long. Once the sequence is longer than about 64 points, the FFT wins hands down.

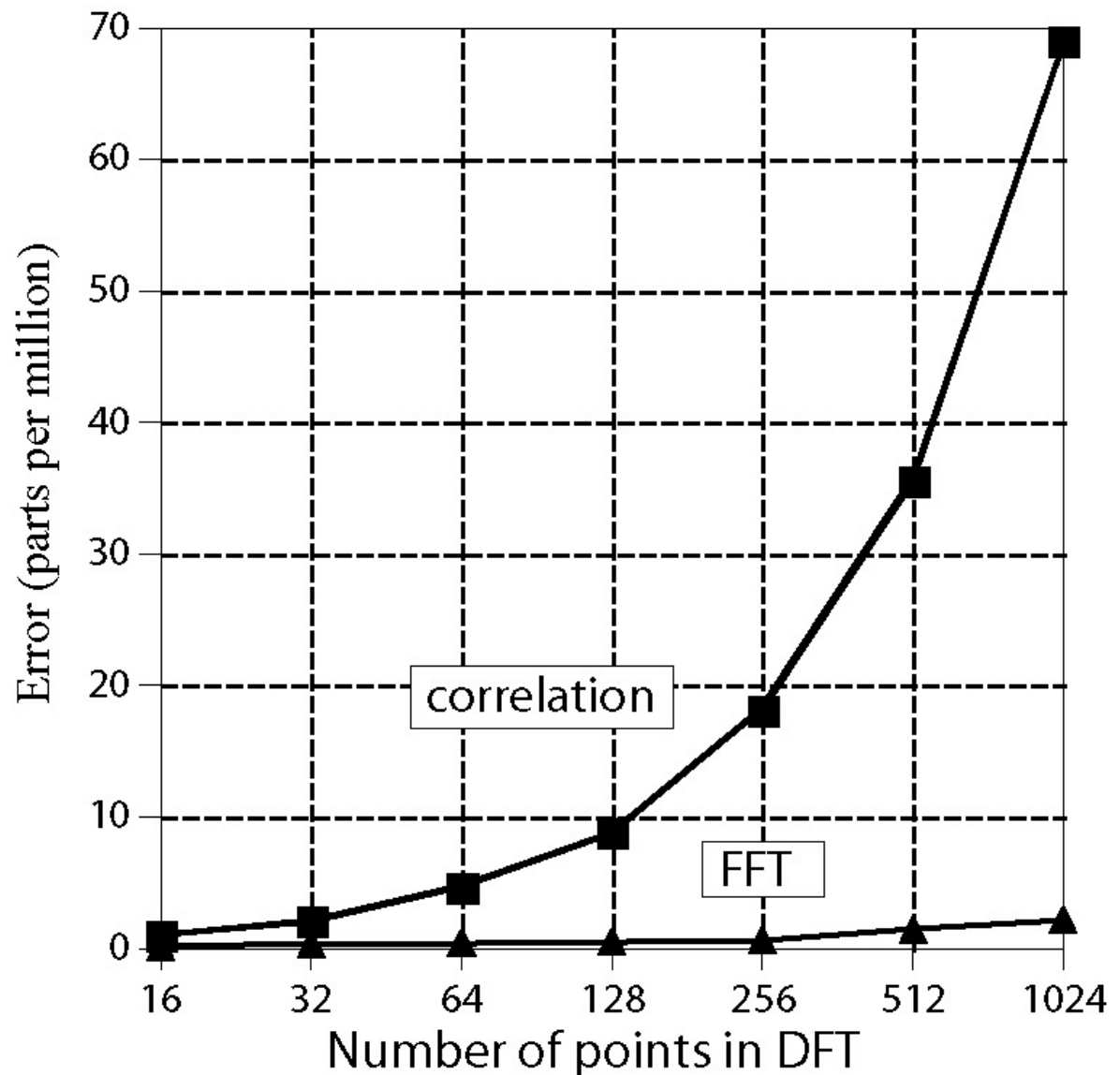
$$\text{Execution time}_{\text{DFT}} \propto N^2$$

$$\text{Execution time}_{\text{FFT}} \propto N \log_2(N)$$

Proportionality constant depends on whether or not sines are precalculated and stored and looked up or calculated on the fly. Further speed gains by noticing duplicated values in sine table. These gains are not order of magnitude.



Another big plus for the FFT is that the FFT calculates the FT more precisely because fewer calculations result in smaller round off error.



Making it faster still. There are some special techniques to make the FFT go faster, but the speed gains are minimal and in the range of 20-40% (not order of magnitude like the original FFT).

One of the go-faster techniques is interesting because it makes use of the symmetry properties of the DFT when computing the DFT of a real sequence. In this technique, one cuts the sequence in half and puts one half in the real part and one half in the imaginary part of the input sequence. (or one could stuff in two signals, one in the real part and one in the imaginary part). One then takes the FFT and gets a mess, but it's a special mess.

Can get a little faster using complex FFT and FD symmetries. Top 2 panels show real signal in complex FFT (wasting imaginary space). Notice that the real part of the FD is symmetric around $N/2$ and 0 , while the imaginary part is anti-symmetric around $N/2$ and 0 . If we put the sequence into the imaginary part of the complex input sequence and put zeros in the real part, the symmetry in the FD is reversed. So can put sequences into real and imaginary parts in TD and do even/odd decomposition in FD to recover the individual FT.

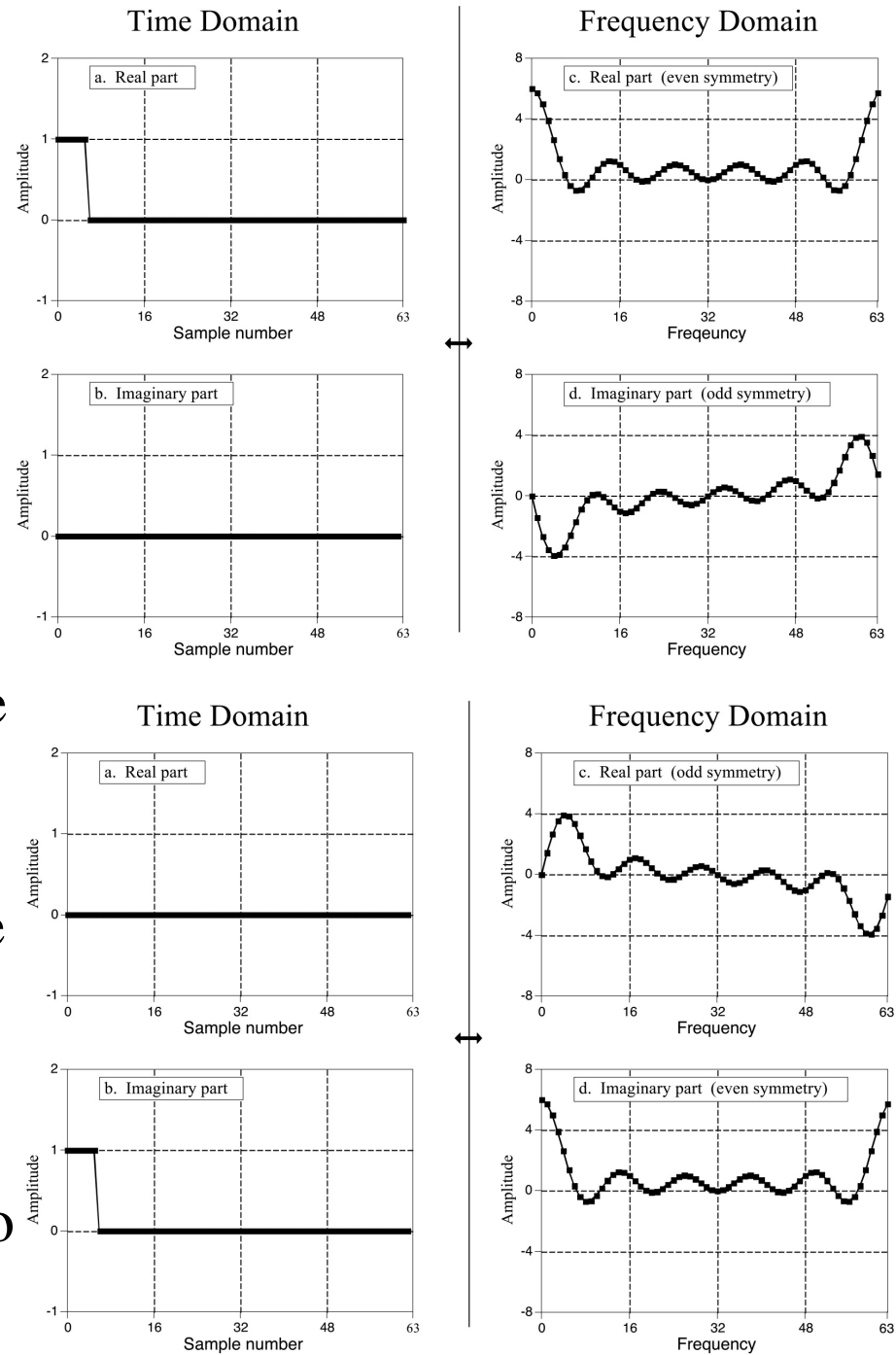
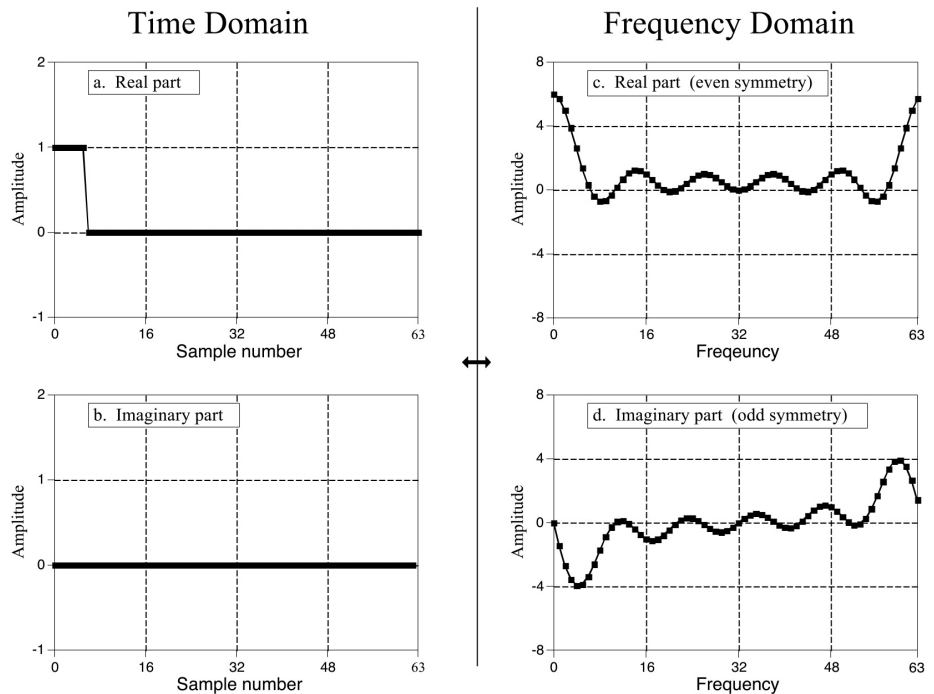
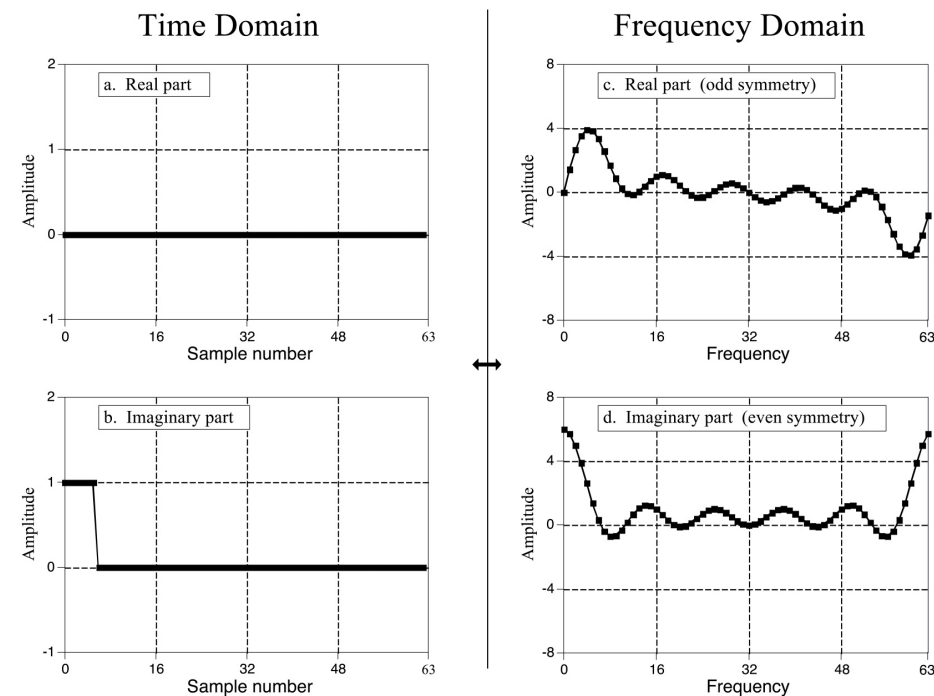


Figure from Smith



Remember that for a real input, the real part of the FT is an even function and the imaginary part is odd. We get the reverse for an imaginary input, the real part of the FT is odd and the imaginary part is even. Since any function can be uniquely (<important) decomposed into an even and odd function, we can decompose the mess that comes out of our FT due to the funky input we gave it. Decompose the real and imaginary parts of the FT into even and odd functions to obtain the individual FTs. (this also tells how the real and imaginary TD parts, in general, map into the real and imaginary FD parts.)⁴⁶

Figure from Smith



Getting useful results from the FFT.

What we want - the FT. Unfortunately this is not what we get on the computer.

So how do we get the best estimate of the FT from the FFT (since this is the only computer method efficient enough to be useful).

Remember the various flavors of Fourier Analysis

FT - for aperiodic analog signals - frequency domain continuous and infinite. TD continuous and infinite extent. Can't do on the computer.

Function in TD has to have finite energy. (Signals that are NOT finite energy: all periodic signals, ramps, steps, exponentials, DC. Signals that are finite energy: one shot pulses)

FS - for periodic analog signals - frequency domain discrete but infinite extent. TD is finite-duration (the period). Also can't do on the computer.

So far can't evaluate on the computer. Probably can't evaluate analytically either.

Next comes the DFT - for periodic, discrete signals (probably better to call it the DFS, but convention is to call it the DFT). Both TD and FD discrete and finite-duration. Can do on the computer (but VERY slow).

DTFT - for aperiodic, discrete signal - frequency domain continuous between -0.5 and 0.5 cycle per sample. (Outside this range repeats - aliasing). TD discrete but infinite extent. Can't do on the computer.

FLDTFT - modify DTFT to be finite-duration and discrete in FD (this is what we did with the step-by-step analysis of what happens on the computer). (there is no standard name for this technique). Can do on the computer (but VERY slow). Since most signals will not be periodic, this is the form (in the sense of the derivation) that we want to use on the computer.

Typically we want to know the FT for some analog signal, which we digitize, changing the FT flavor to the DTFT. We have to further limit it to a finite length of signal which changes it to the FLDTFT. We have 2 choices on the computer.

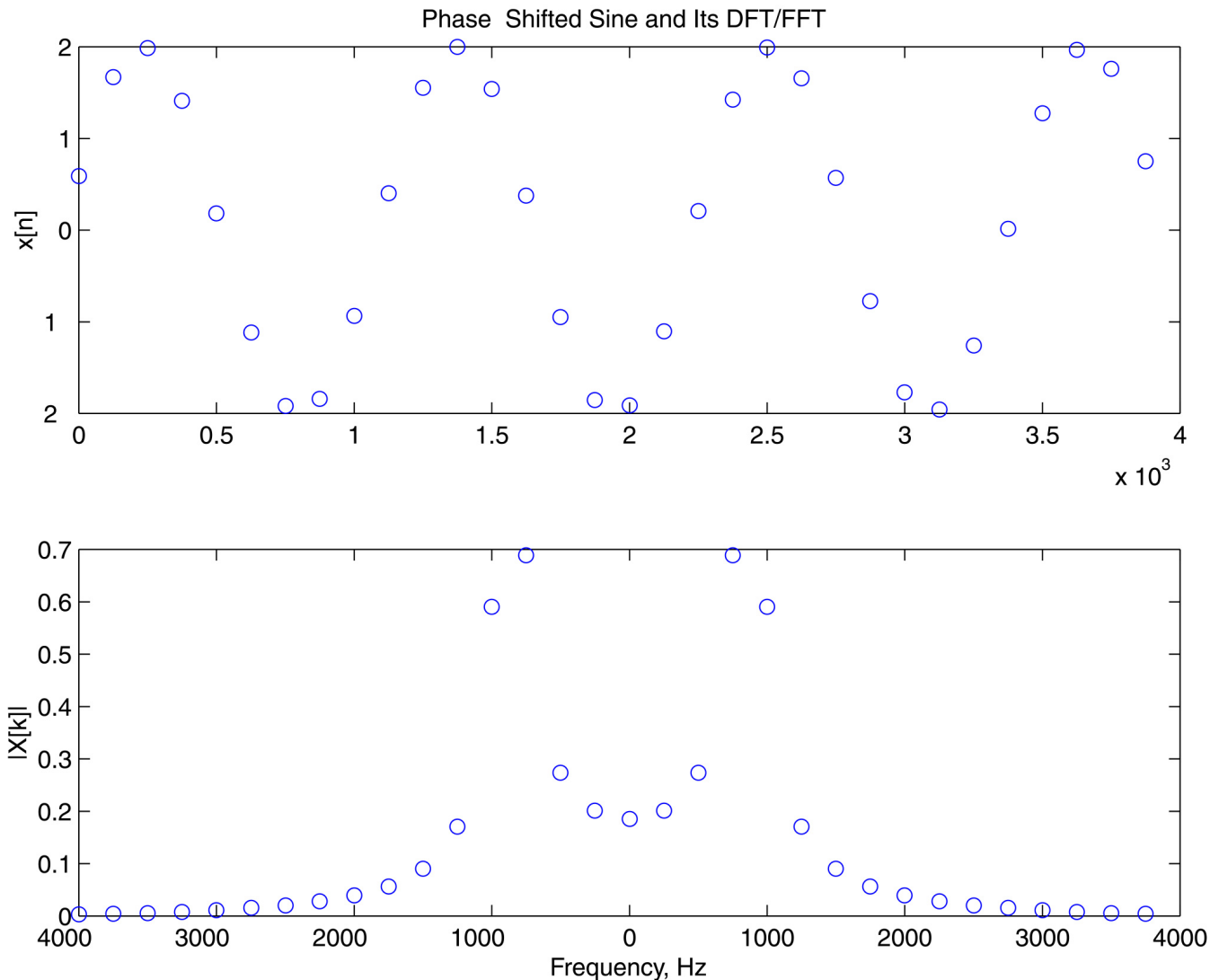
1) Evaluate the FLDTFT and realize the spectrum will be “smeared” with the sinc function. In addition to the spectrum smearing, it will also be SLOW.

2) Evaluate the FFT and realize that the signal will be made periodic whether we want it to be or not (since it’s really a FS, not a FT). We will get a FAST (possibly wrong, or at best confusing) result. But it is so fast it’s worth the effort to find out how to make it “correct”, or at least usable.

So the question is - how to use the FFT to get the “best” approximation to the FT?

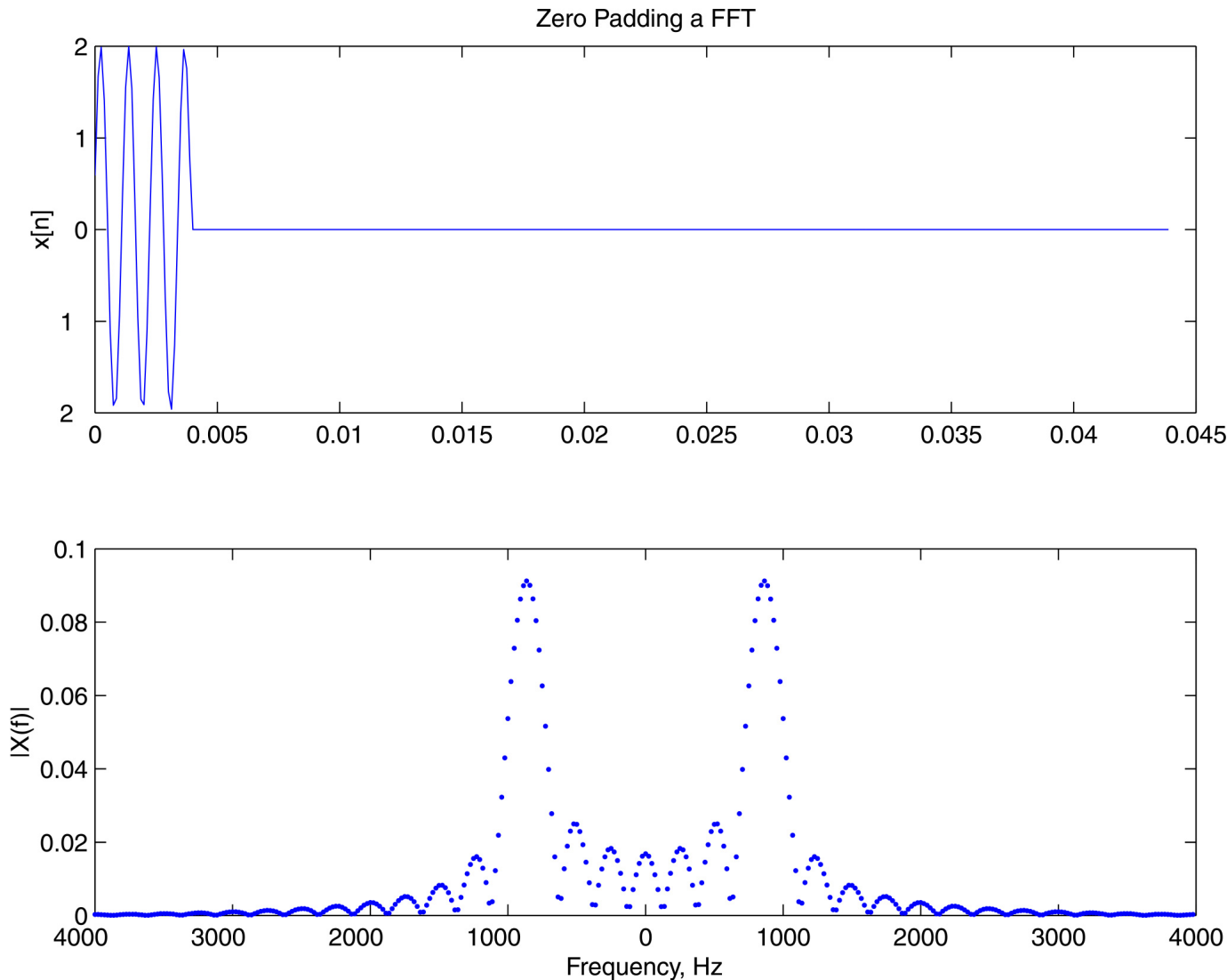
We want an estimate of the FT of an aperiodic signal, but the FFT works with periodic signals. An aperiodic signal can be thought of as a periodic signal with infinite period. Since we can't have infinite periods in the computer, we can approximate an infinite period by making the period very large. This can be easily done by ZERO PADDING the signal (you can put them before the signal, at the end, some before and some after - it does not matter -- why?).

Look at example with numbers (this enables us to put actual time and frequency labels on the x axis, else we have sample number only) - 3.5 cycles of a sine wave in 32 points, sampled at $f_s=8\text{KHz}$, and its FFT.



Notice that the peak is not very precisely defined (is wide) and we have the annoying sinc factor, although it's not obvious that it's a sinc.

Now try the same data (3.5 cycles of a sine) that has been extended with zeros to a total length of 320 points, 10 times the original length.



We get much better frequency resolution (more samples in FD), and the sinc is now “obvious”, but the peaks are still “wide”.

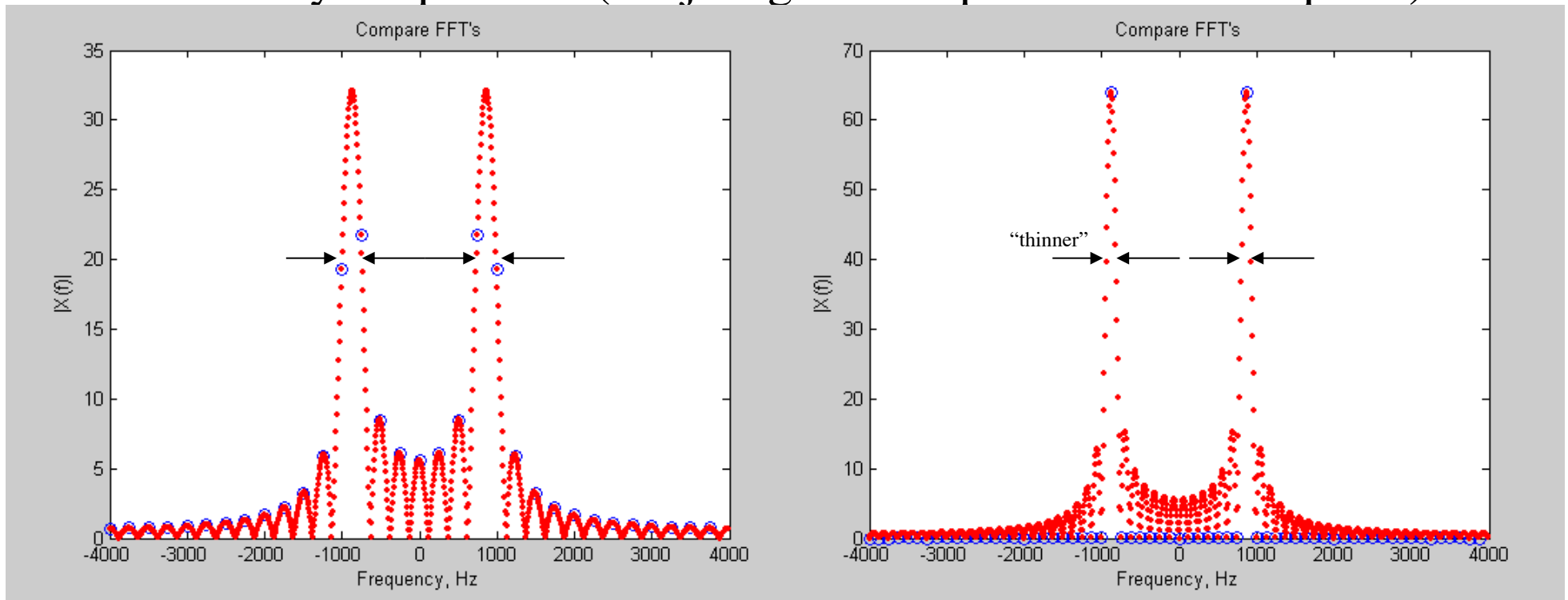
So the sinc function is back!

Zero padding in the TD and taking the FFT essentially does sinc interpolation in the FD. (Similarly zero padding in the FT before taking an IFFT does sinc interpolation in the TD).

Final advice -- if you want nice looking FD plots, pad with zeros in the time domain to extend the TD signal to be at least 10 times as long. This will give finer resolution in the FD samples, i.e. the set of discrete frequencies is larger.

Since the sampling has not changed, the Nyquist frequency is still the same, so we have a finer sampling of FD points (frequencies closer together as we stuff more samples into the frequency range 0 to Nyquist).

Important note: width of peaks is same for zero padded and non-zero padded input, so we do not get an improvement in our ability to separate two nearby frequencies (we just get more points on the freq axis).



To make peaks narrower, we have to take a longer data window (wider boxcar in TD, which gives narrower sinc in FD). On left we have 32 & 1024 point (zero padding from pts 33 to 1024) FFTs with 3.5 cycles (32 pt FFT not on FFT “line”), on right we have 64 & 1024 point (zero padding from pts 65 to 1024) FFTs with 7 cycles (64 pt FFT now “on” FFT “line” for 64 pt FFT – so one point, at freq, hi and the rest “0”).

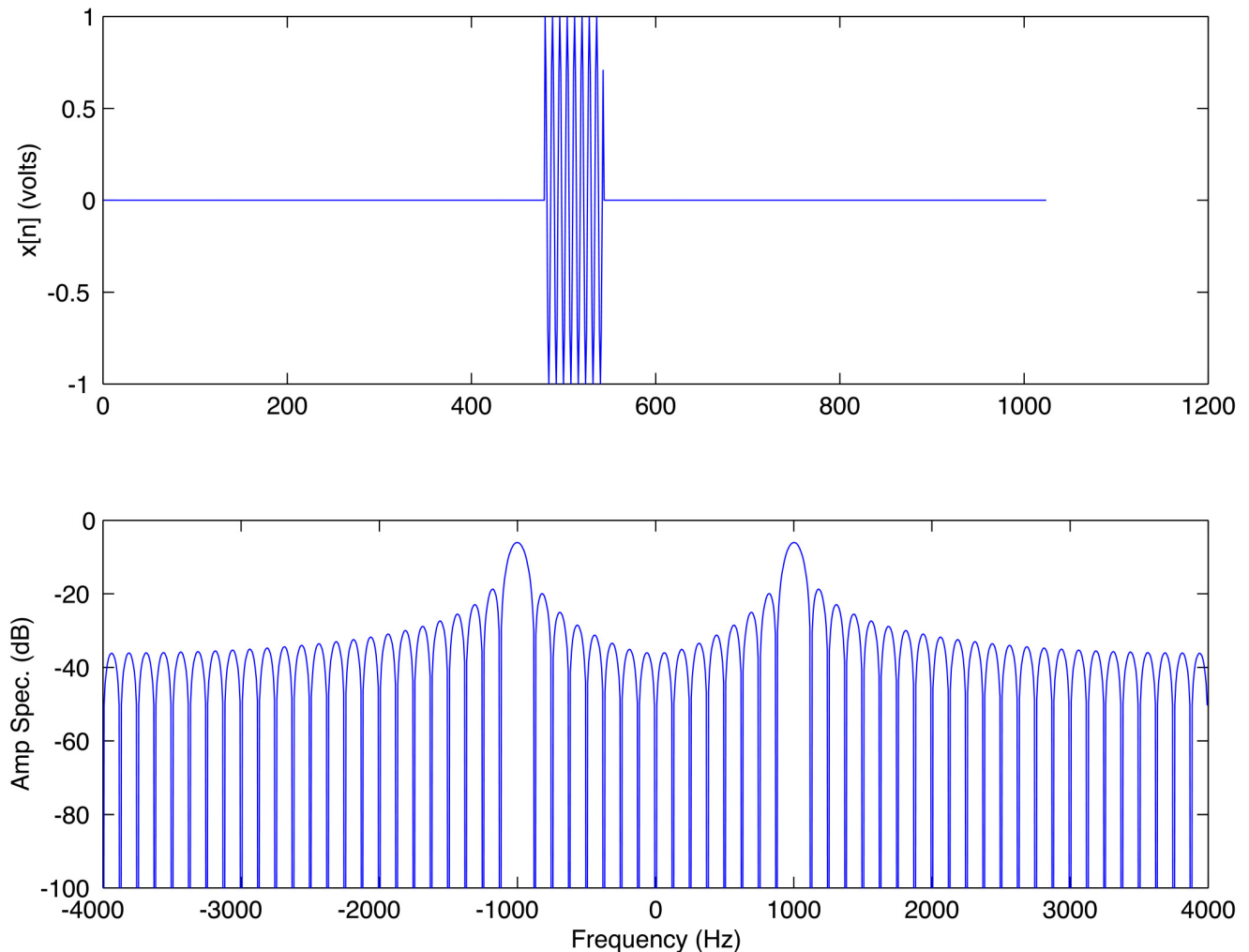
We have seen several times that one of the big problems (with respect to differences between the analog world and the finite-duration digital world) is the effect of the periodicity that the finite-duration digital world enforces on everything. We saw how this creates discontinuities in the infinite-extent, periodic TD representation that require high frequencies in the FD representation and how these higher frequencies get “aliased” into the FD representation (the wings, skirts, tails, etc).

Since the problem comes from the discontinuities at the ends of the finite-duration sequence when we duplicate it to make it periodic, what happens if we modify/distort the input sequence so that these discontinuities are suppressed?

This is the idea behind windowing with weighted windows (non boxcar windows).

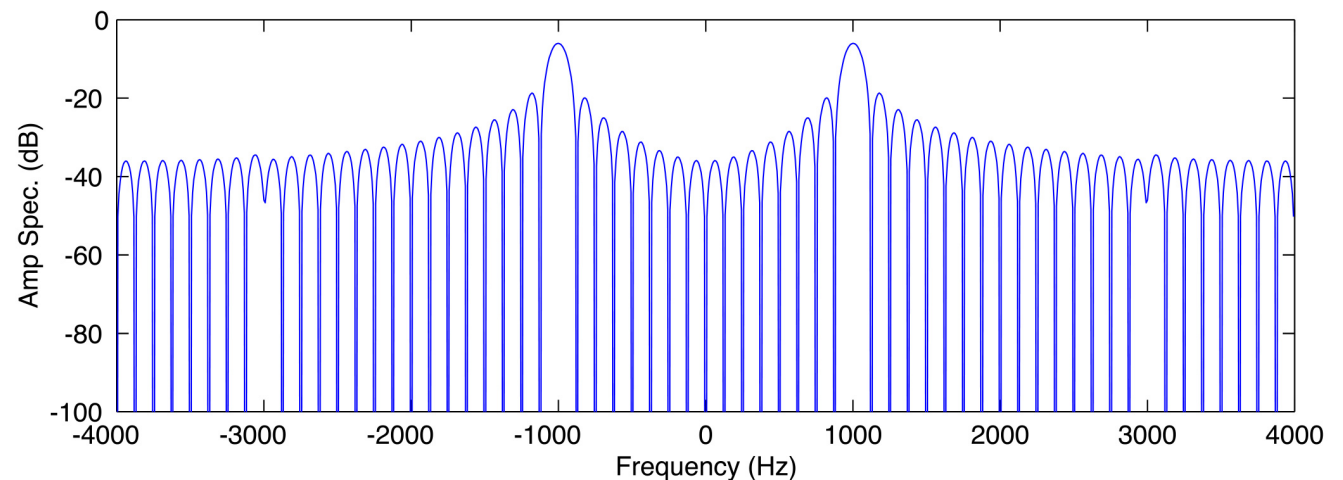
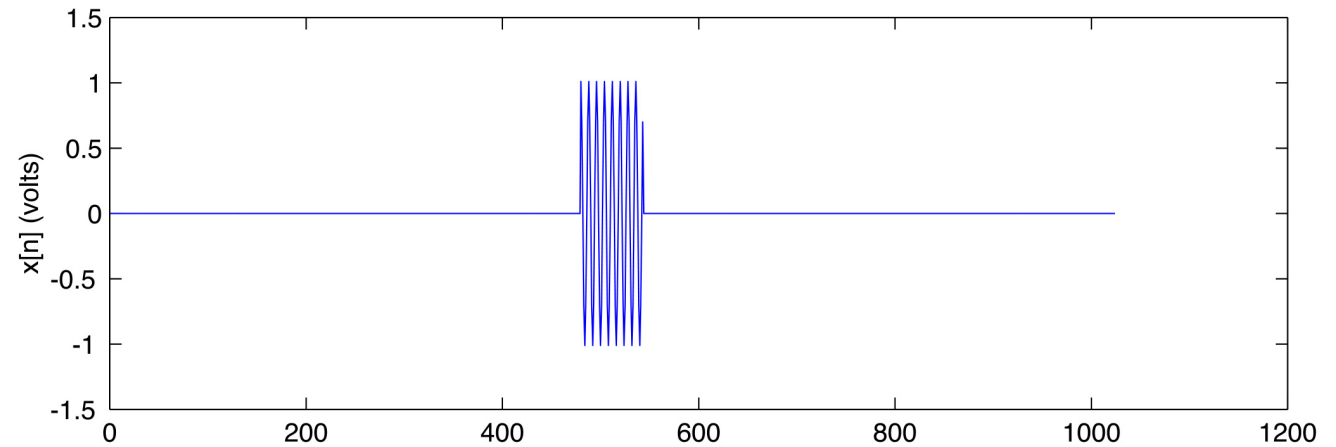
Problem - Finite Data Length = Frequency Spreading. Taking finite amount of data goofs up DTFT calculation. The true amplitude spectrum is convolved with a sinc function causing a sine wave that has an impulse for a DTFT to spread out into a signal that seems to occupy all frequencies.

You might be able to live with this if the input signal was a single frequency.



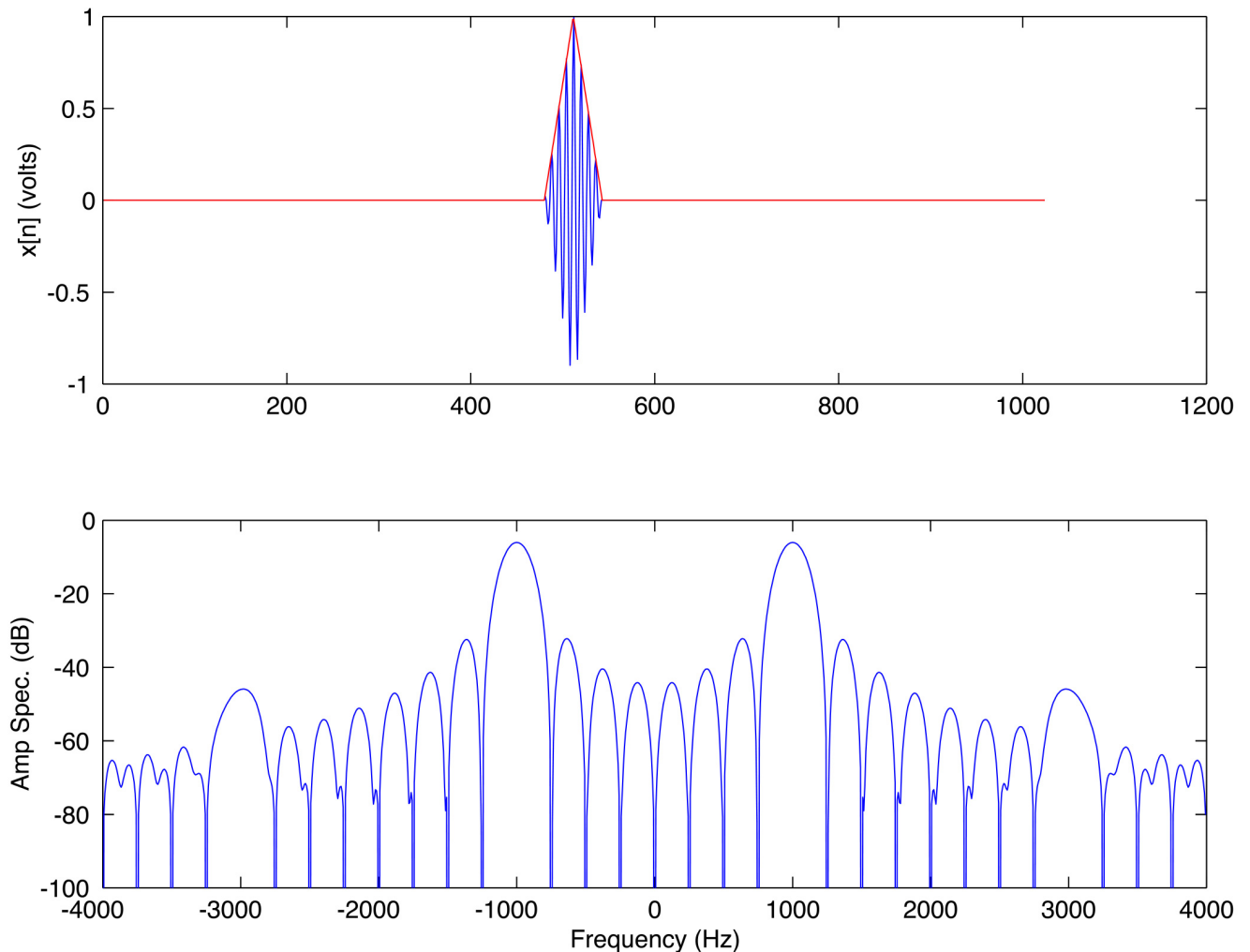
What if the signal was composed of 2 frequencies - add a second sine wave at 3kHz that is 40 dB down. You can't see it in the TD plot. One would like to be able to see it in the FD plot. Well - there is something at 3kHz, but it's not too useful.

So now the frequency spreading has gone from a nuisance to a fatal problem.



Enter the weighted window to the rescue (sort of). Since the “problem” comes from the discontinuities at the end, distort the data in the TD by multiplying in the TD with an envelope function that goes to zero at the ends (you should already see the implications of this -- FD convolution with the FT of the window function. So it’s not free).

Here’s the FFT of the same input as the last example with a triangular window applied. Notice that we can now clearly “see” the second signal at -40 dB.

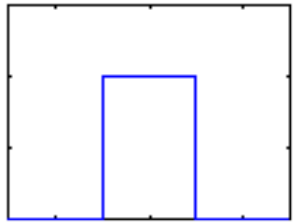


The triangular window solved the problem of discontinuities in the TD signal, but did not completely remove the contamination from the high frequency terms. This is because the derivatives are discontinuous.

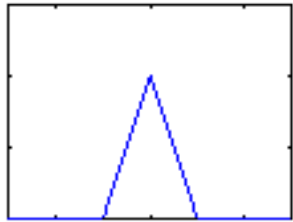
(Can see why need continuous derivatives looking at case of 1.5 cycles of a sine wave in the window - get a sharp “V” shaped join at ends, and therefore need high frequencies to reproduce this. Here the function is continuous but its first derivative is not. It gets harder to see this kind of “high frequency” effect as we have more high order continuous derivatives, but it’s still there.

The leakage envelope more or less goes (decays) as $1/f^{n+1}$, where n is the order of the first discontinuous derivative [$n=0$ when function itself is discontinuous, e.g. boxcar])

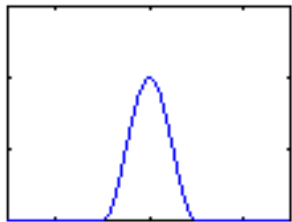
This lead to the development of other types of windows



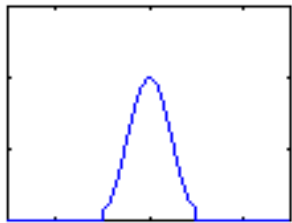
Boxcar - easy to apply (if you use it to shorten a sequence). It is the one you get whether you realize it or not! Sub-optimal effect on spectrum.



Triangular window - easy to compute. Improves spectrum. Good start. (all derivatives discontinuous - has effect of not removing high frequencies)



Hanning window - a “raised” sinusoid (add 1 to cosine and chop at $\pm\pi$). Window and first derivative continuous. Even better.

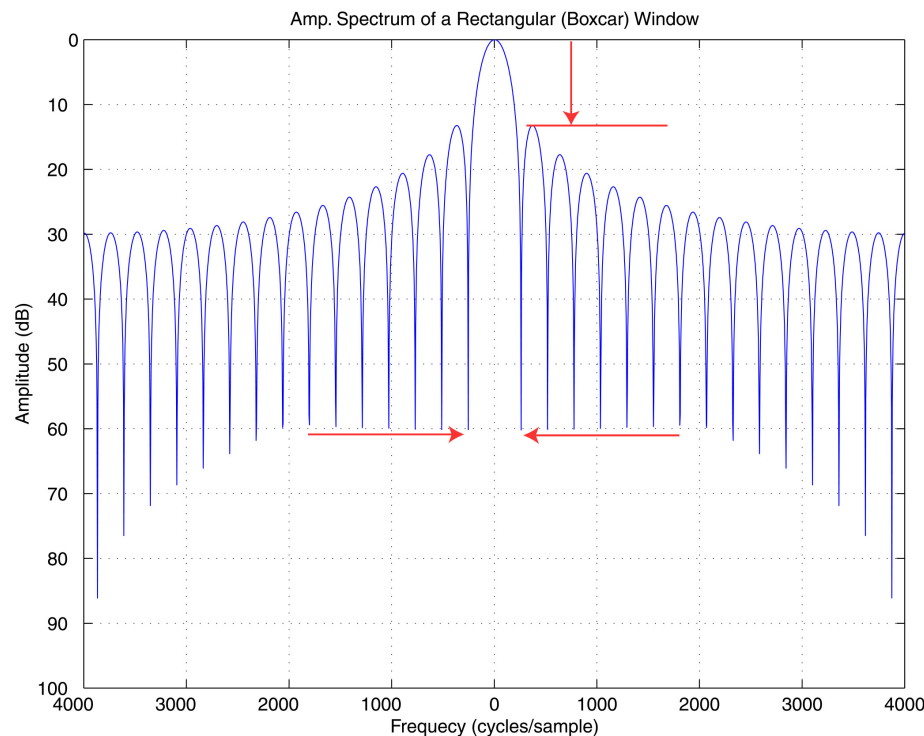


Hamming window - add a small DC offset to Hanning. Window and derivatives discontinuous again.

What do these windows look like in the FD? What determines how “good” they are?

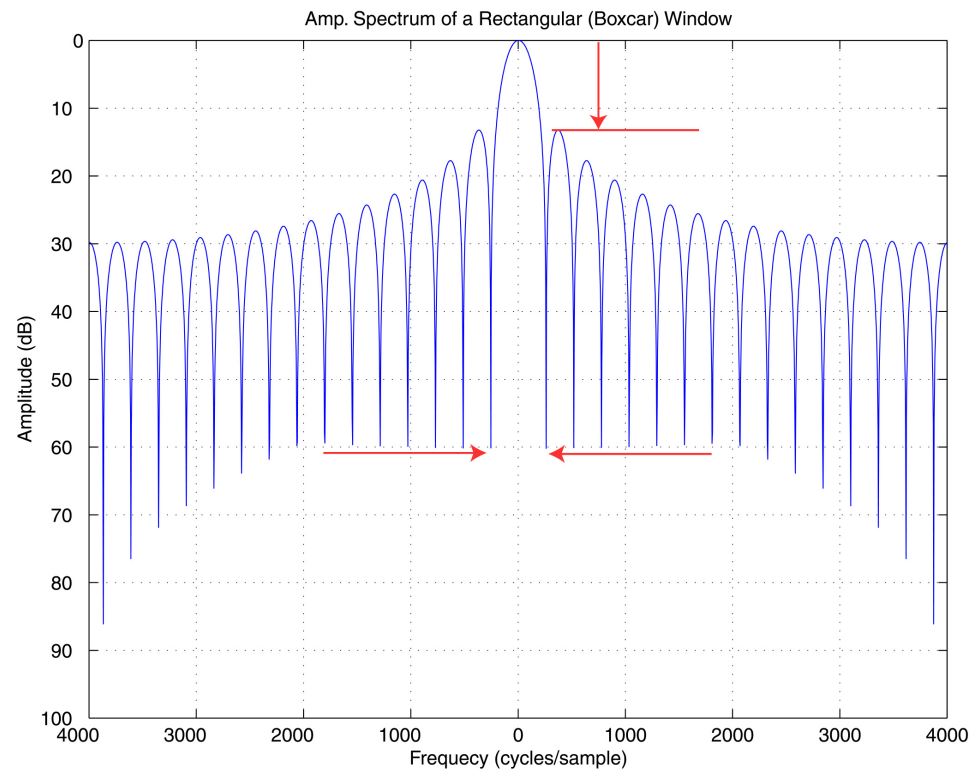
There is no single best weighting window. Different applications use different windows. When deciding which window to use, it is handy to characterize the window in some way.

Start by looking at the FT of the weighting windows. In virtually every case, the amplitude spectrum looks like a series of bumps, or lobes.



The tallest, and usually widest, bump is at DC, and is called the Main Lobe.

All of the other bumps are at higher frequencies and are called Side Lobes.



Blurring, smearing, or main lobe width.

All windows tend to smear an impulse over a range of frequencies close to the true frequency. This is the same thing that an out-of-focus optical lens will do to a point of light, so it is also called blurring.

This distortion makes it difficult to tell if a bump in the spectrum was generated by a single sine wave (single impulse) or a series of two or more closely spaced impulses (sinusoids at closely spaced frequencies).

To measure the severity of this problem, look at the width of the main lobe.

Small is good. You can measure the main lobe width in Hz, rad/sec, cycles/sample or rad/sample.

This width is a function of 2 things - the sampling frequency and the length of the window.

Leakage or side lobe attenuation.

If blurring is a local problem, leakage is a long-distance problem. A strong signal will generate terms at frequencies far from its frequency.

When there is a strong signal present, it may mask weak signals - even if they are separated by a great distance in frequency. (remember previous example)

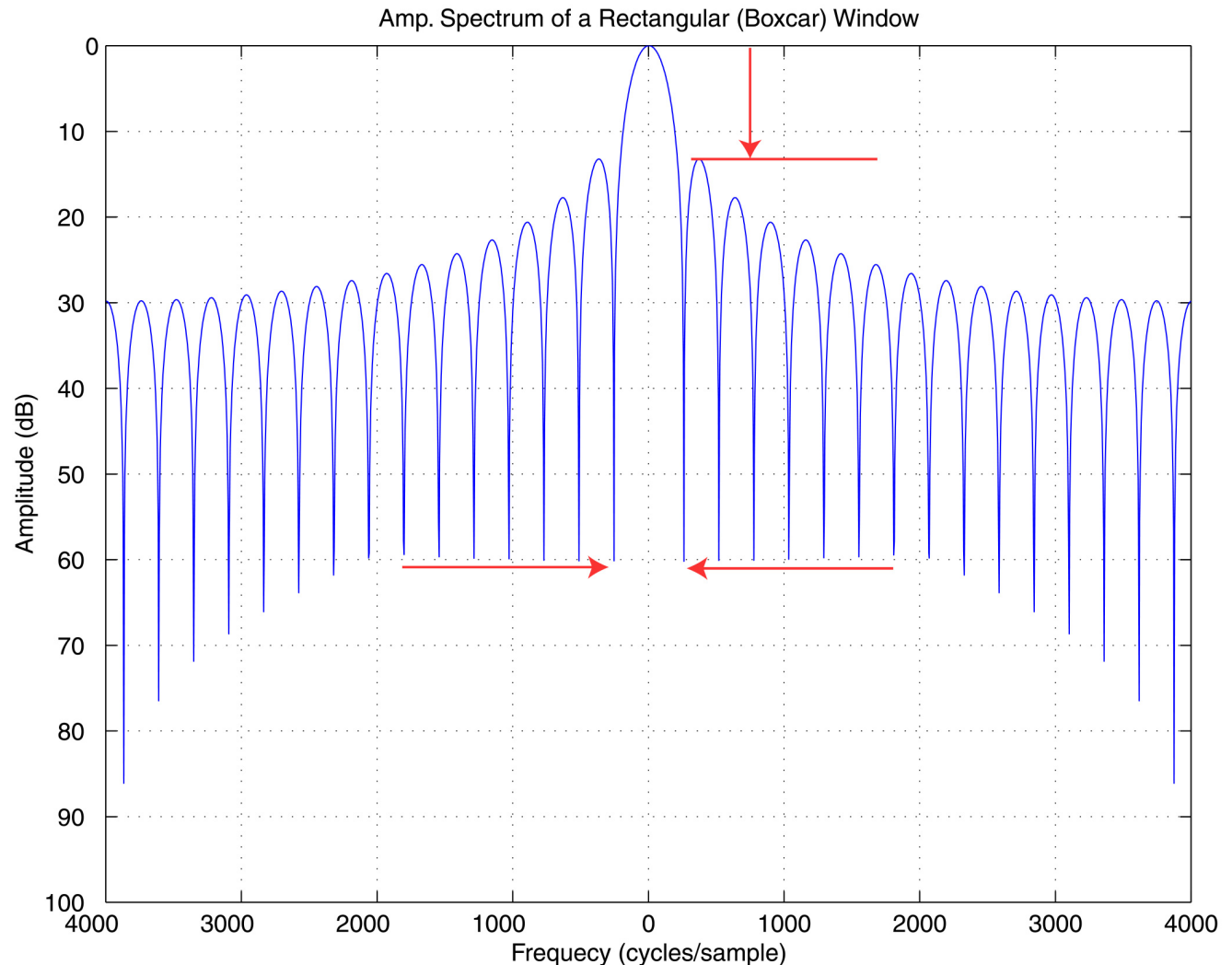
To measure the severity of this effect/problem, look at the difference between the amplitude of the main lobe, and the amplitude of the highest of the side lobes.

This difference in amplitude, usually measured in dB, is called the side lobe attenuation.

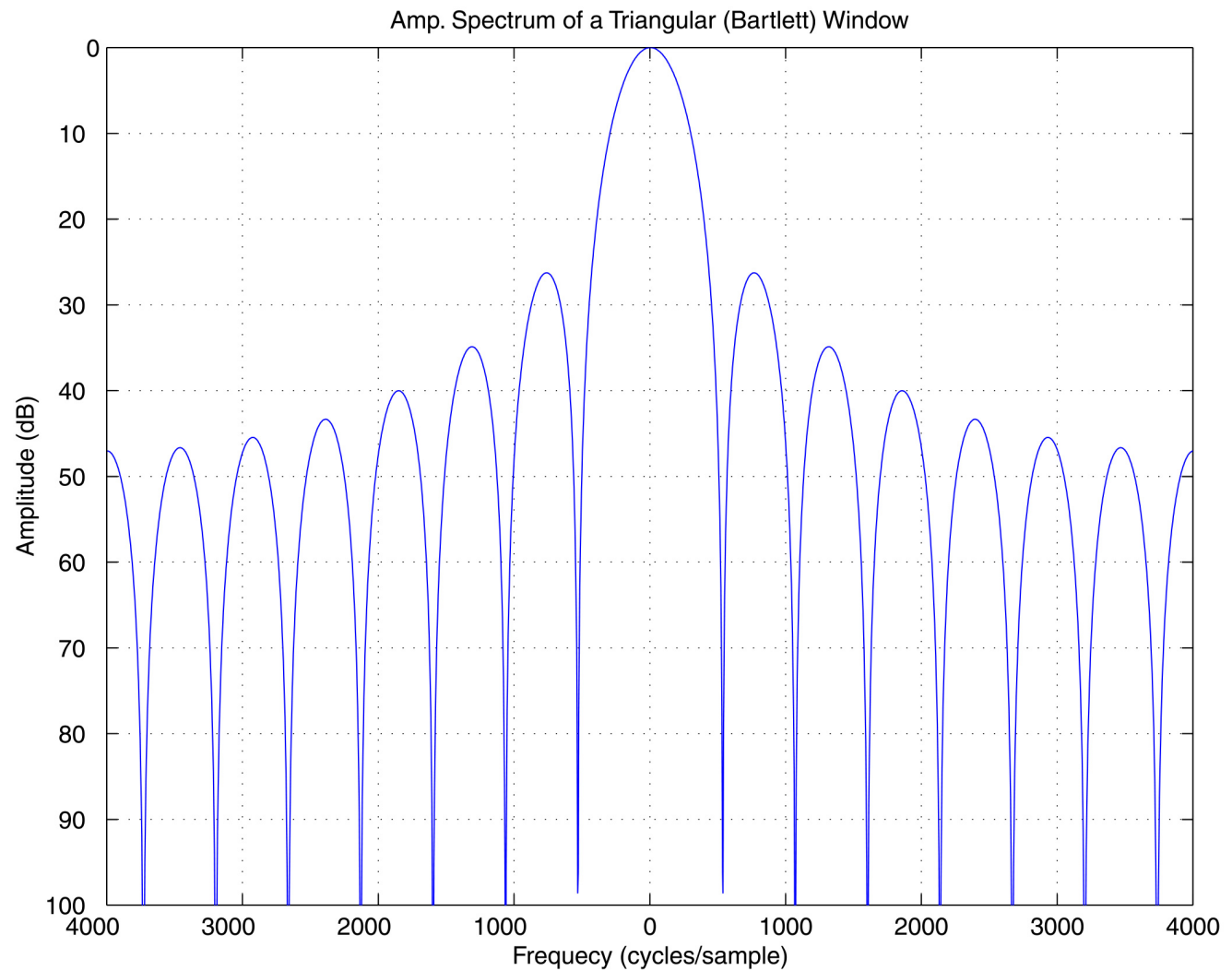
Bigger (higher attenuation) is better.

So looking at the various windows we've mentioned. Can distinguish two characteristics with any window's spectrum - main lobe width and side lobe "rejection"/leakage.

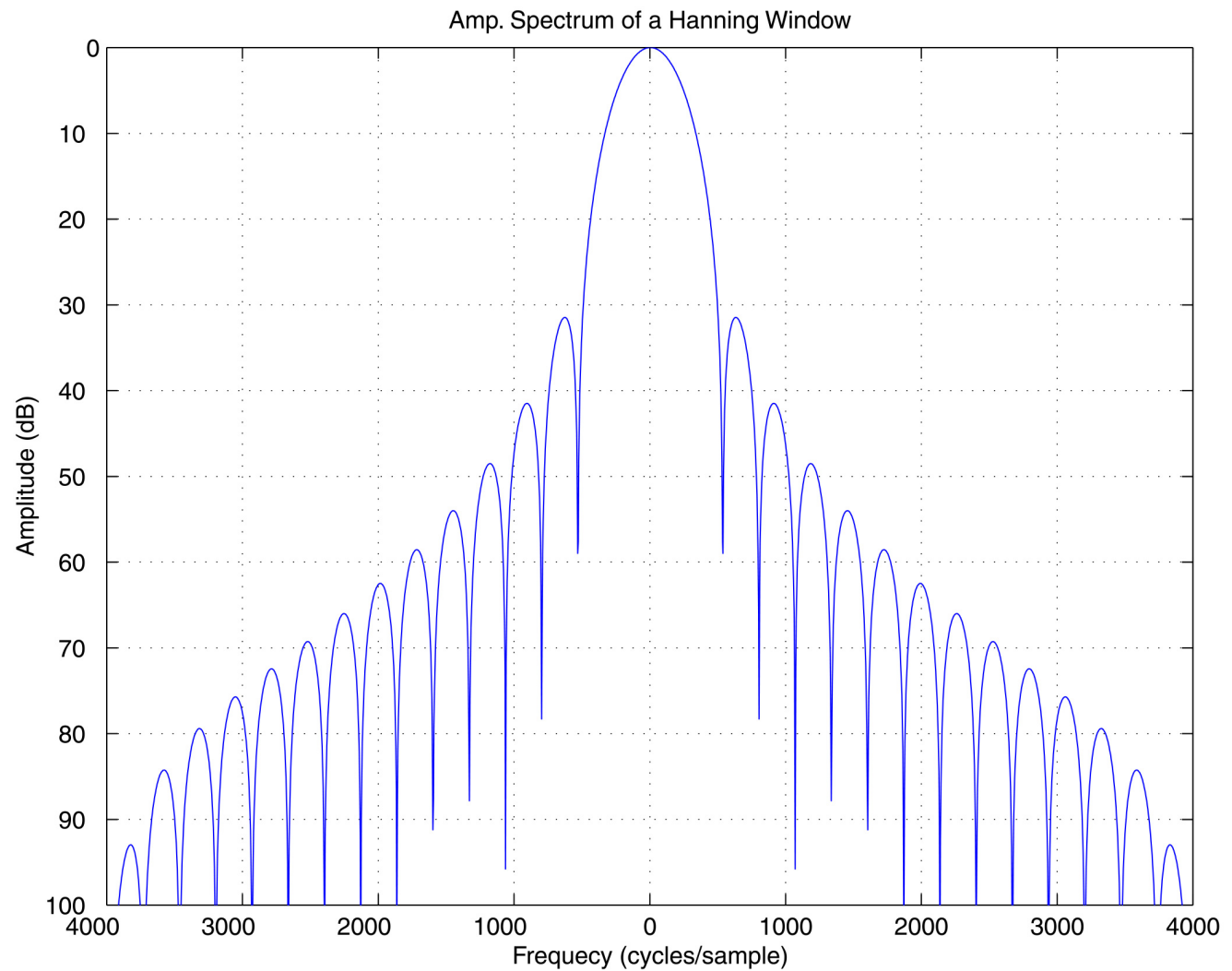
Boxcar -
narrowest central peak (remember that running average was "sharpest" low-pass filter), horrible leakage. Decays as $1/f$.



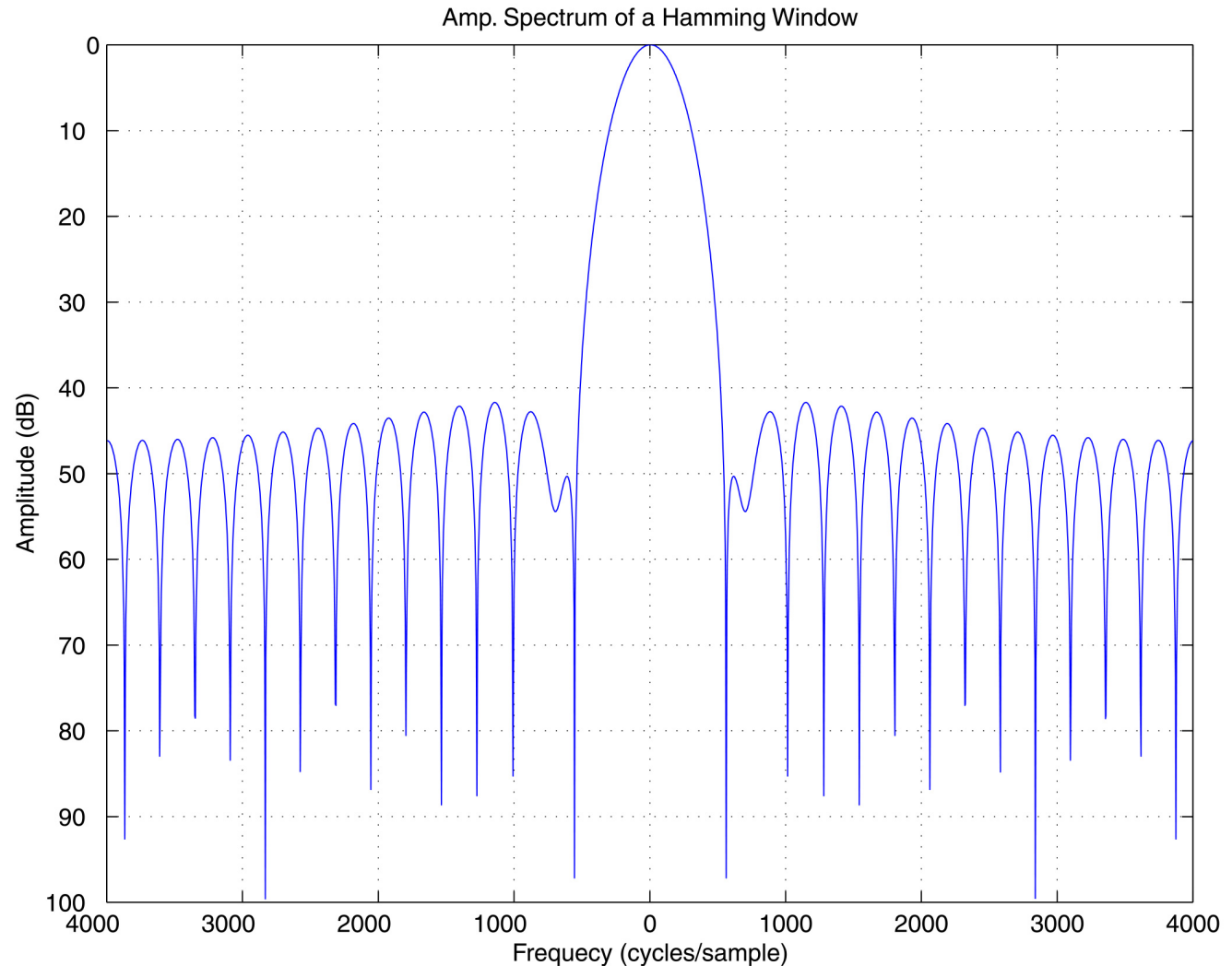
Triangular (Bartlett) - compared to boxcar is wider (not so good) but first side lobe is lower (better). Decays as $1/f^2$.



Hanning - compared to boxcar is wider (not so good) but compared to boxcar and triangle first side lobe and higher side lobes are lower (better). Decays as $1/f^3$.



Hamming (Hanning with a DC offset) - compared to boxcar is wider (not so good) but compared to boxcar, triangle and Hanning first side lobe is lower (better) and higher side lobes are lower (better). Leakage worse than Hanning. Decays as $1/f$.



Window	Main Lobe Width (Hz)	Side Lobe Atten. (dB)
Rectangle	f_s/N	13
Triangle	$2f_s/N$	26
Hanning	$2f_s/N$	31
Hamming	$2f_s/N$	42
Blackman	$3f_s/N$	58
Kaiser	f_s/N to $f_s/2$	13 to infinity
Lanczos	f_s/N to $f_s/2$	13 to infinity
Tukey	f_s/N to $2f_s/N$	13 to 31

Various windows and their main lobe width and side lobe attenuation. The rectangular window gives the narrowest main lobe but lousy side lobe attenuation. There are an infinite number of window types - each one optimized for some application.

When not to use windows -

Windows are like drugs - they distort reality for people and computers that can't handle the truth. They are fun to play with, they let you see things in a different light, and lots of people use them, but whenever possible, you should Just Say No (© 1988, R. Regan).

Weighting windows are useful, and arguably necessary, when the signal extends for a far longer time span than you can feed into a computer or when the input signal has a continuum of frequencies (always?!).

Weighting windows are not necessary, and should not be used, when the signal is a finite length pulse (e.g. P or S wave). The FT is really only supposed to be used for finite energy, or pulse-type, waveforms. Surface waves are usually not pulse type waveforms.

Filter impulse responses usually are pulse type waveforms. To calculate a filter's frequency response by the FFT, do not use weighting windows.⁷⁰