ELECTRONOTES
1016 Hanshaw Road
Ithaca, NY 14850

APPLICATION NOTE NO. 426

November 15, 2015

A DFT CURIOSITY – INVESTIGATING BOTTOM UP

The presentation below is overtly about a curious case of using a DFT to estimate a Fourier Series [1, 2], with a subtext of suggesting that one can often find an explanation by working at the problem from the answer back. It comes from a real experience in an EE teaching lab (at least 25 years ago!). The same curiosity reappeared to me recently as an outright question from a reader of Electronotes. Originally, it came to me as a report of an apparent bug in a Pascal program such that a DFT gave a slightly wrong answer. It was said that no one could explain what was going on (long list of professors cited!). They were not asking me – or maybe they were. I knew pretty much what was going on. Not that I figured it out on the spot. It just happened that I had investigated it before (perhaps only weeks before) when I found something essentially the same, and ran a logical investigation, working back from the answer (bottom up). Sure I told them the cause.

Here the school was doing a practical exercise in Fourier Series (FS), also using a companion lab implementation with a sampling system followed by a DFT. The task was to sample a square wave from a function generator, use the Pascal program (written by a TA), and relate the result to the well-known mathematical FS of a square wave. Not much thought had been given to the limitations, and it would be sufficient if the spectrum out of the DFT had odd harmonics with the first few falling off as 1, 1/3, and 1/5 (1/k), or roughly so. Stopping there basically avoided the issue of folding (aliasing). This pretty much worked.

As I recall, the acquisition part of the experiment was not much different from just typing in a sequence. You stopped the hardware part when you caught it right. Only the square wave from an analog function generator was retained (no speech for example) and the captured data was adjusted to a series of positive samples followed by an equal number of negative values – exactly one cycle. In a few tries, with steady hands, this was achieved. We can thus easily simulate what happens thereafter using Matlab.

Fig. 1 shows the case of a length 64 sequence of samples with 32 samples of value +1 followed by 32 of value -1 (Fig. 1a). These 64 samples are processed to the frequency domain with a DFT (Matlab ***fft***) giving the magnitude (Fig. 1b) and the real and imaginary parts (Fig. 1c and Fig. 1d). This is exactly as we should expect.

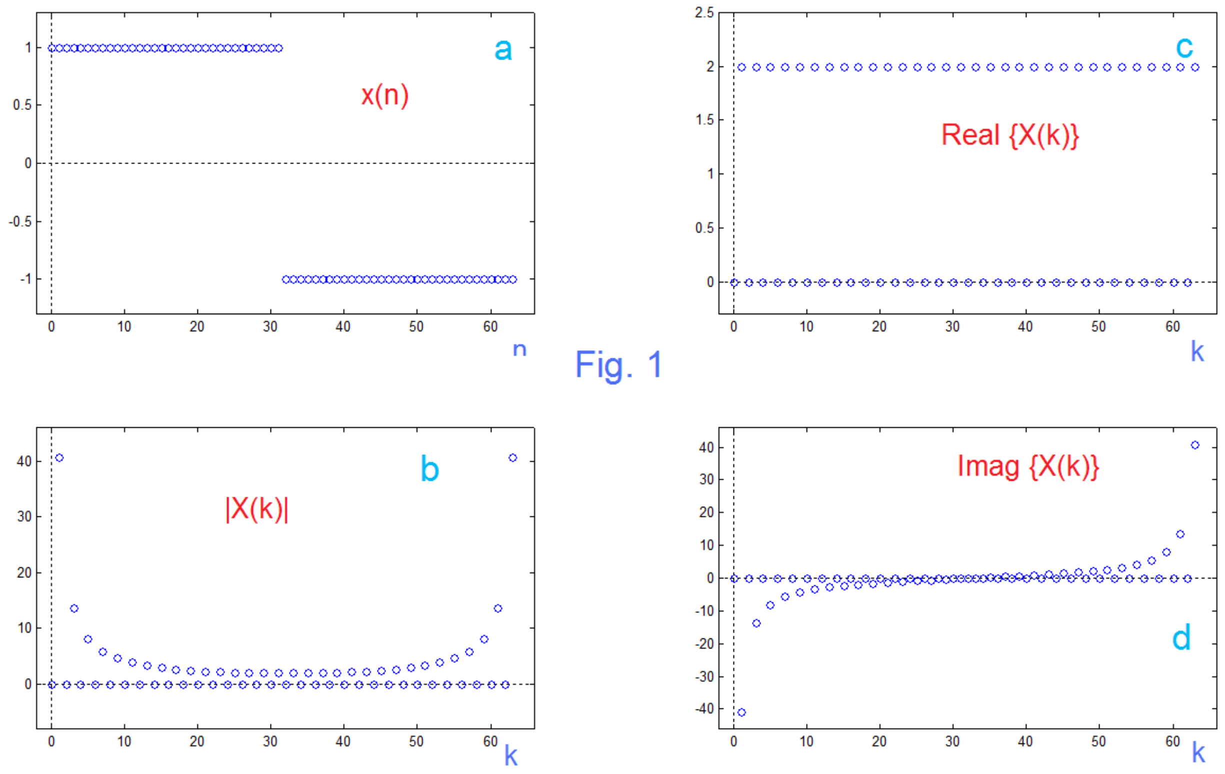
Fig. 1

Everyone was however expecting a purely imaginary result (odd symmetry), yet this has a real part which is a constant value of 2 for odd harmonics, 0 for even harmonics. Blaming the error on the hardware and sampling made no sense. [For example, a DC error would have made X(0) non-zero.] Because the sampling was "fudged" until a "correct" sequence was captured (could have been just typed in), the Pascal program for the DFT was falsely blamed. There was nothing wrong with the DFT.

I do not know how much effort went into trying to find a program bug. Evidently, a lot. My approach had been previously, as I show it here: to separate out the real part, and invert the DFT of that. That will tell you, in the time domain, what is messing with us here. This is what I mean by "bottom-up" or working back from the answer. Fig. 2 shows the result of the test. The "offending" component is two non-zero samples, a +1 at n=0 and a -1 at n=32. How strange.

Now, Fig. 1 and Fig. 2 tell the whole story of the finding here, but we will nonetheless be redundant and print out a result (in the style of Fig. 1) that corresponds to subtracting off the portion of x that caused the constant real part. In fact this is interesting in that it shows that in order to get rid of that real part, we replace the first sample of x (which is +1) with a sample of zero. Likewise the 32nd sample changes from -1 to zero. So two samples go away. Why!

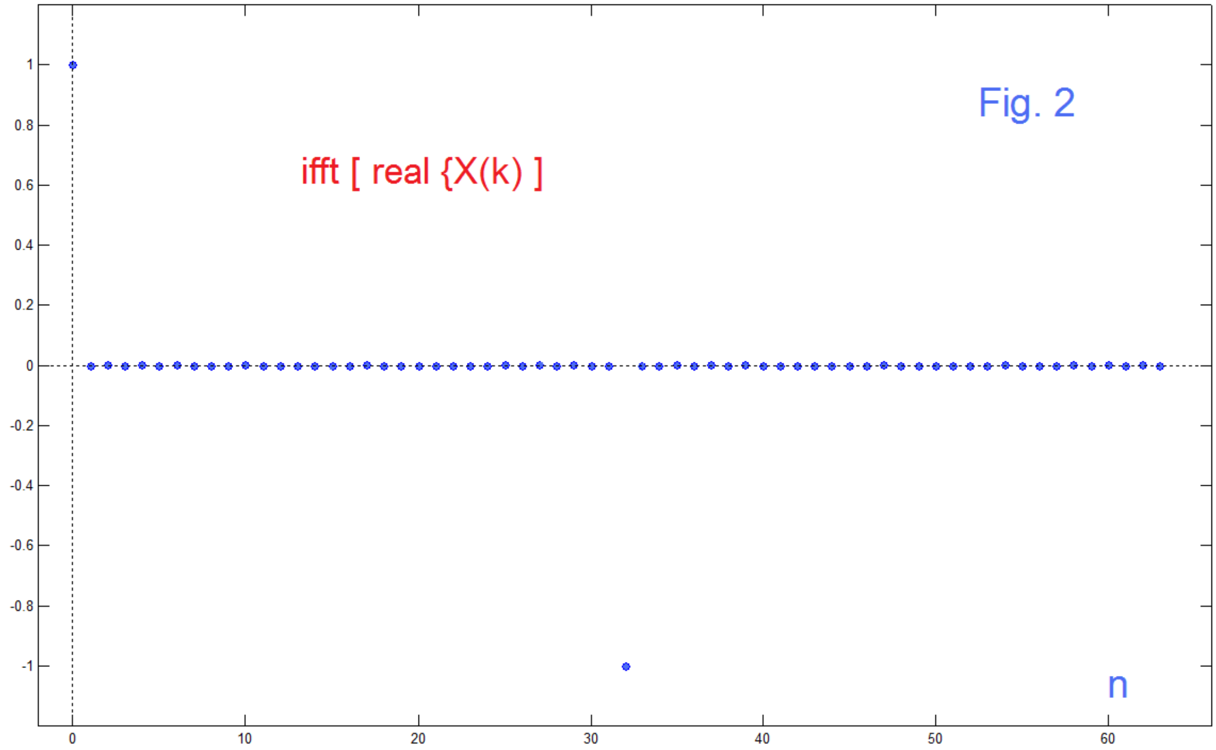
ifft [real {X(k)]
n
1
0.8
0.6
0.4
0.2
0
-0.2
-0.4
-0.6
-0.8
-1
0
10
20
30
40
50
60

Fig. 2

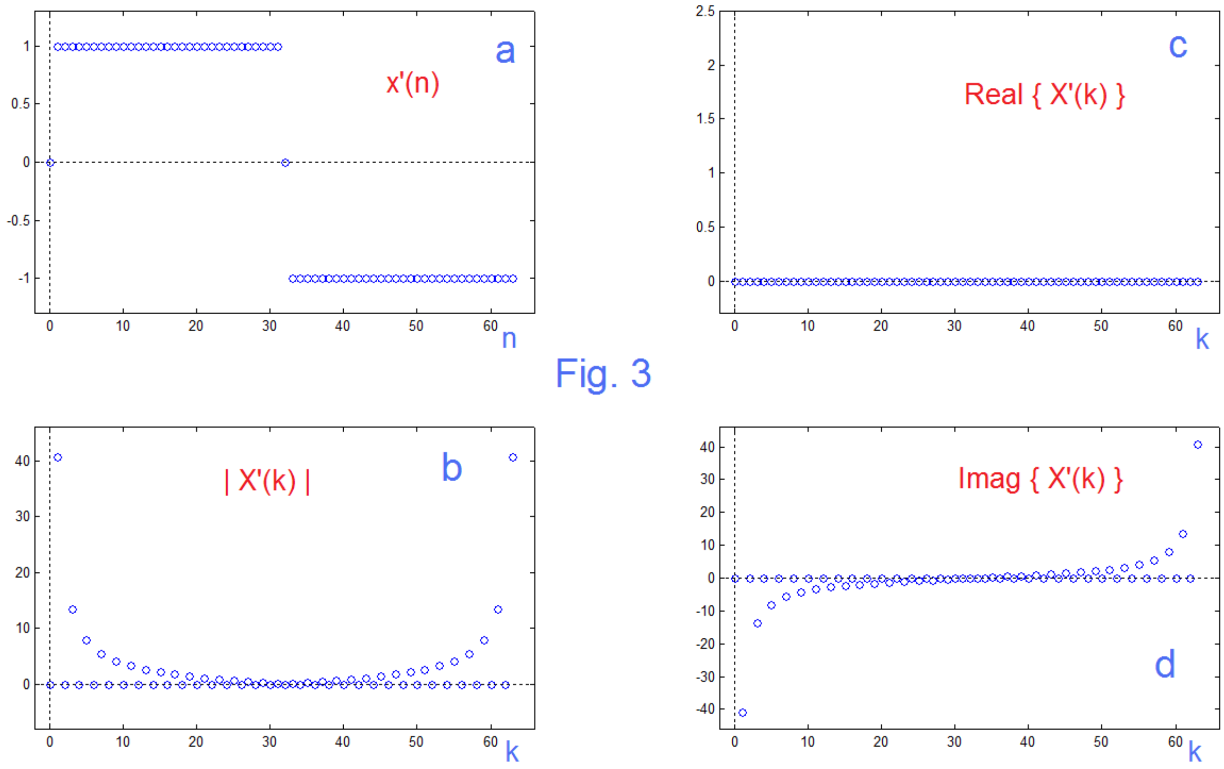
a
x'(n)
1
0.5
0
-0.5
-1
0
10
20
30
40
50
60
n
c
Real { X'(k) }
2.5
2
1.5
1
0.5
0
0
10
20
30
40
50
60
k
b
| X'(k) |
40
30
20
10
0
0
10
20
30
40
50
60
k
d
Imag { X'(k) }
40
30
20
10
0
-10
-20
-30
-40
0
10
20
30
40
50
60
k

Fig. 3

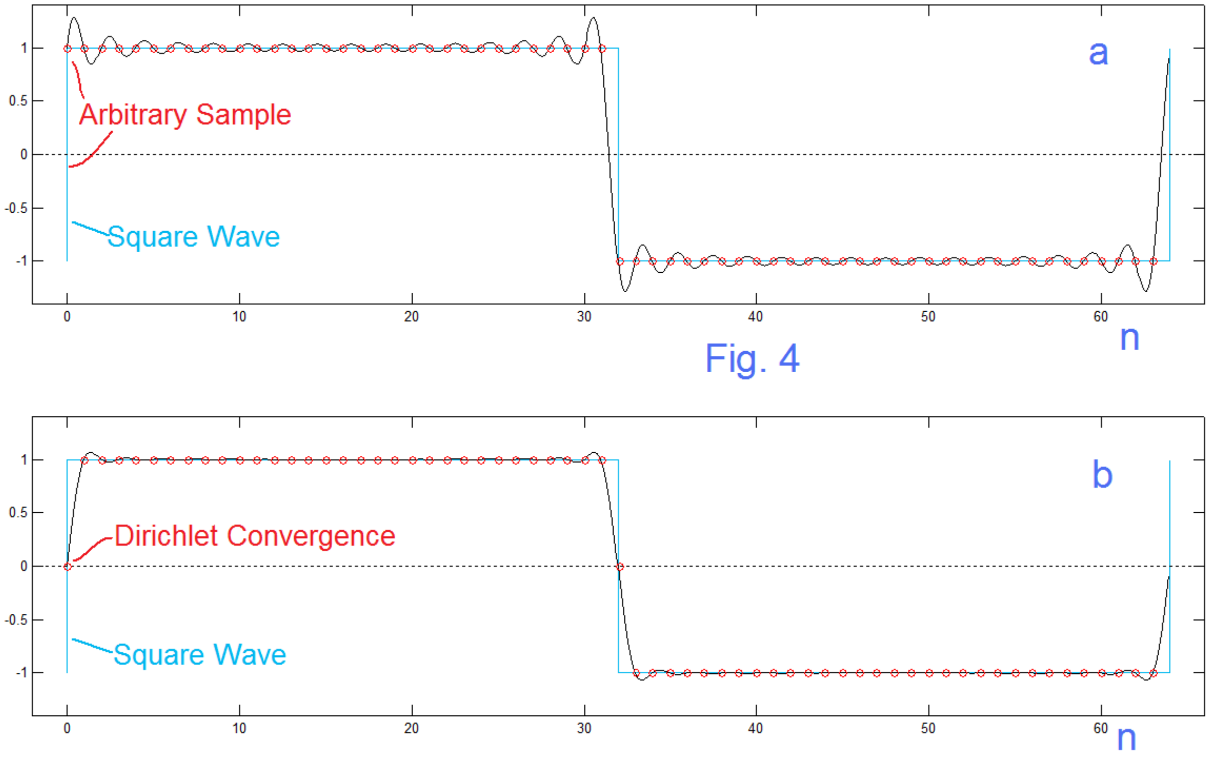

Fig. 4

One thing we have often urged our readers to remember is that the DFT "has in mind" a certain continuous-time waveform corresponding to the particular set of samples. This waveform will not in general be a function we might assume. If we want to find out what waveform this is, we can use DFT interpolation [3-5] to get additional samples that are between the given samples (usually plenty to get a virtually continuous plot). So what do these interpolations look like for the sequences x (presumed samples of square wave) and for x' (with 1st and 32nd samples removed)? And how do these interpolations relate to the presumed square wave? Fundamentally different!

A key to what is involved here is seen for time n=0 (and n=32) of Fig. 4a. What value does the continuous-time waveform (the blue square wave) have at these points? We have shown, in the first case, the sample taken to be +1 at n=0, although there is no reason to justify this, as n=0 is a point of discontinuity, as the square wave is there drawn.

There are several things to note: First, in Fig. 4a, we have shown samples of the square wave precisely where most people would first select them relative to the blue square wave. However, suppose we shifted the sampling time relative to the square wave by a small amount (say by the diameter of one of the red open circles). We would have the same result of 32 positive samples followed by 32 negative samples. Indeed, in a hardware acquisition (as opposed to just typing in values) we would have a hard time getting the perfect time alignment of Fig. 4a. So, exact timing aside, Fig. 4a is typical of what we would expect in a lab.

Secondly, in contrast to Fig. 4a, Fig. 4b is specially chosen. As hard as it might be to get a sample at time zero (the discontinuity of the square wave) we would find it quite impossible to grab a sample mid transition. Yet, when we talk about FS, the Dirichlet convergence says that the value of the FS reconstruction IS the midpoint of the transition. Thus Fig. 4b is the “Right Answer” when we are talking FS.

A third point relates to the interpolations, the black-line reconstructions in Fig. 4a and Fig. 4b. These are informative, and absolutely correct. Note that they go through the original data points exactly. The two zero samples of Fig. 4b serve like transition samples and greatly reduce ripple. Consider also the possibility that the blue square wave (our imagination!) might be a zero-cross of the black interpolated curves. This would be true for Fig. 4b, but not for Fig. 4a which is offset by half a sample.

Yet another clue of what is going on is afforded by considering the imagined square wave to be phase shifted, as in Fig. 5 where we have shifted the time samples (Fig. 5a) by 21 samples to the left. Here, we find that the magnitude DFT is exactly the same (compare Fig. 1b with Fig. 5b), and any other phase would be the same, except for the very special (unobtainable by sampling) case of Fig. 3. Here we have a real part, but not the same real part as in Fig. 1c.

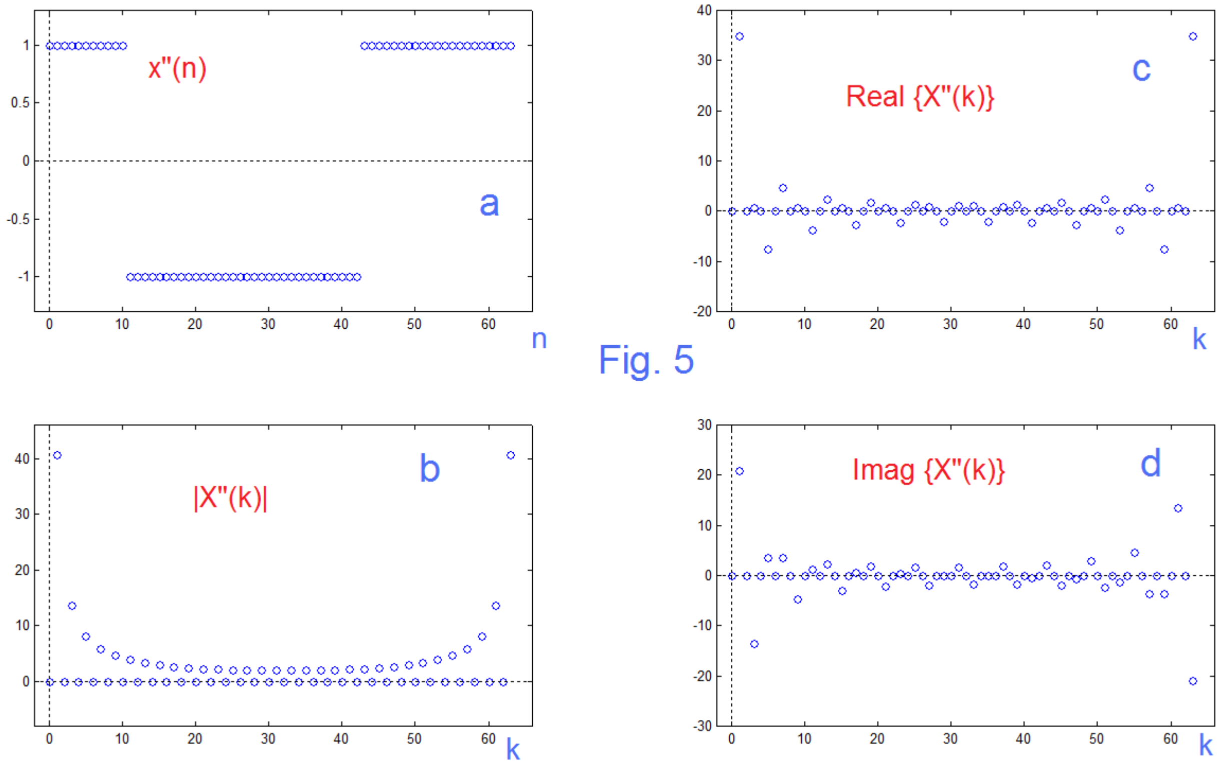

Fig. 5

Thus we make the case that both results (Fig. 1 and Fig. 3) are correct and are more or less an apples/oranges pairing (closely related, both fruits!, but different). As we emphasize, Fig. 1 is pretty much what one would first try when placing samples, and what one would likely aim for (and achieve, with a bit of care) in a lab. Fig. 3 is something that exists as a mathematical object.

Nonetheless neither is the correct FS result, although both approximate the result for low coefficient index. The difference of these from the FS is the bandlimited nature of the DFT. The spectral results are aliased and overlapped, making the components too large (making up for those that are truncated). The first three (odd) FS coefficients, as stated, should fall off as 1, 1/3 and 1/5. In Fig. 1b they fall off as 1, 0.3344, and 0.2019. In Fig. 3b, they fall off as 1, 0.3312, and 0.1961. Not bad. The results will not be as good for higher coefficients and will of course become unavailable about half way up. Getting more coefficients is a matter of using many more samples [2].

REFERENCES (All from EN Publications)

[1] "Fourier Map", Electronotes Application Note No. 410, May 6, 2014 http://electronotes.netfirms.com/AN410.pdf

[2] "Calculating the DFT and the Fourier Series: Each with the Other", ***Electronotes***, Vol. 19, No. 188, February 1997

[3] "Interpolating Samples with the DFT/FFT," Electronotes, Vol. 16, No. 172, July 1988.

[4] "A Short Presentation of FFT (DFT) Interpolation," Electronotes Application Note 398 July 20, 2013 http://electronotes.netfirms.com/AN398.pdf

[5] "FFT – Interpolation in Time and Frequency", ***Electronotes,*** Vol 23, Number 222, June 2014 http://electronotes.netfirms.com/EN222.pdf

[6] Matlab code used for figures above: http://electronotes.netfirms.com/AN426.txt