

SIGNAL PROCESSING & SIMULATION NEWSLETTER

Discrete Fourier Transform (DFT) and the FFT

Let's take this continuous signal composed of three sinusoids.

$$g(t) = \cos(2\pi f_a t) - \frac{1}{3} \cos(2\pi 3 f_b t) + \frac{1}{5} \cos(2\pi 5 f_c t)$$

Assume that $f_a = 1$, $f_b = 3$, and $f_c = 5$. The waveform is plotted below.

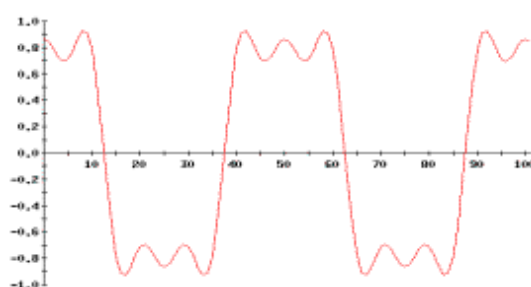


Figure 1 - A signal representing a square wave

We can draw the Fourier transform of this signal easily by examining the amplitudes of each of these frequencies and then putting one-half on each side of the y-axis as shown in Fig. 2 for a two sided spectrum. Real signals such as this one produce only one sided spectrums also shown below.



Figure 2 - The two-sided and single-sided Fourier Transform of $g(t)$

This is the theoretical Fourier Transform of the continuous waveform $g(t)$. The Fourier Transform tells us that there are just three frequencies in the signal and no others. There is no ambiguity in the results. This ideal Fourier Transform is what we want to see when we do the Fourier Transform on an analyzer or on a computer but in reality this is nearly impossible to obtain. All implementations of the Fourier Transform are attempts to achieve the theoretical results, however, digital signal processing introduces approximations and truncation effects which keep us from realizing the ideal.

Figure 3 shows the outline of the same signal along with dots that represent what we actually see of the signal on an oscilloscope. This is because most signals we capture are sampled versions of the real analog signal. We pulse the analog signal every so often and then plot these sampled values. We connect the samples and get a proxy to the signal.

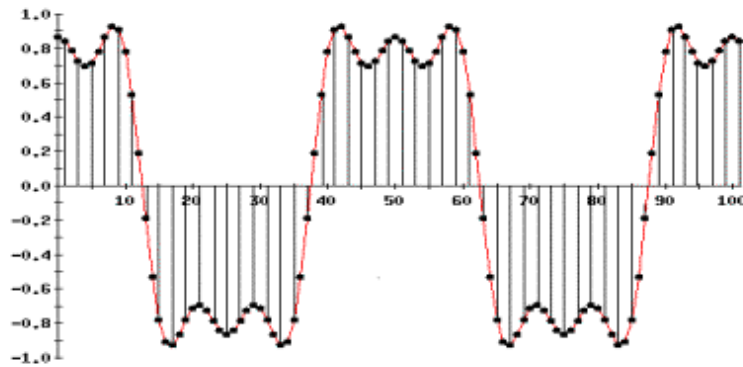


Figure 3 - The discrete samples of a real signal as shown by dots. We really do not know the underlying shape of the signal.

Mathematically, the sampled signal is obtained by multiplying the target signal with an impulse train of period t . Since we usually collect only a limited number of samples we limit the length of the impulse train to a certain time window. The discrete signal is expressed as

$$g(k\tau) = g(t)\delta(t - k\tau)$$

So before we can even look at a signal, two things have happened. 1. we have multiplied the target signal by an impulse train and made the continuous signal a discrete signal and 2. we have chosen to collect only a limited number of samples, in effect windowing the sequence with a rectangular window function.

Fig 4a shows the original signal and its Fourier Transform. In (b) we have the Fourier Transform of a pulse train which is used for sampling the original signal. The Fourier transform of the impulse train consists of just one frequency, the sampling frequency.

Next step is the rectangular window that limits the infinite impulse train. Its Fourier Transform is shown in c and is the well-known sinc function.

Now we have a signal which is a product of three signals.

$$g(k\tau) = \underset{1}{g(t)} \underset{2}{\delta(t - k\tau)} \underset{3}{u(t)} \quad \text{for } t < T$$

The first is the original signal, the second is the impulse train of period t and the third is a step function lasting for time T . What about the Fourier Transform of the product of these three signals? Mathematically we know that multiplication in time domain of two signals results in convolution of their spectrums in the frequency domain. So we have an inkling that the convolution of all three of the Fourier Transforms may not give us the spectrum in (a). But is it close enough to (a), and if not how different is it?

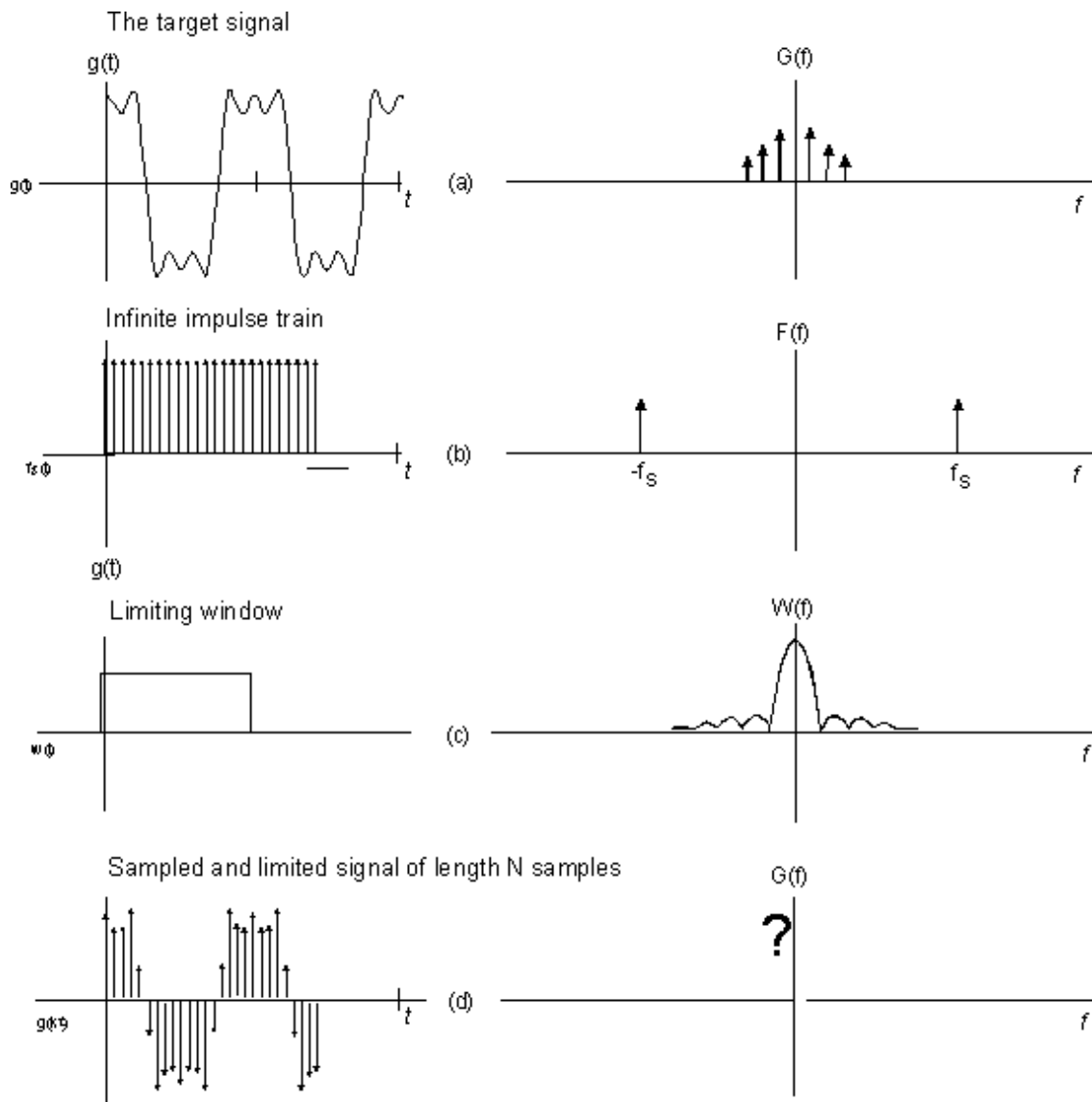


Figure 4 - The quandary of digital signal processing

a. The continuous target signal, **b.** is multiplied by an impulse train of frequency f_s , **c.** The limiting window of length T , **d.** The sampled and limited signal, what do we get here?

Let's define some terms.

Sampling frequency: f_s

Sampling frequency is a measure of how often we pulse the continuous signal to obtain the samples. The quantity sampled is the amplitude of the signal.

Sample Time: t

Time between each sample. It is also the inverse of the sampling frequency.

If sampling frequency, $f_s = 20$ samples/sec, then

$$t = 1/f_s = .05 \text{ sec}$$

Total Number of Samples: N

The total number of samples collected or available = N

Sequence Time Length: T

This the time length of the collected sequence and is equal to the product of the sample time and the total number of samples.

$$= N t$$

Sample index: k

The sequence time is no longer continuous so instead of t , we use a discrete time measure called k_{th} sample. This is an index of the samples. Its range extends from 0 to $N-1$, where N is the last sample. Each k_{th} sample of total N samples, is located at time k times t secs.

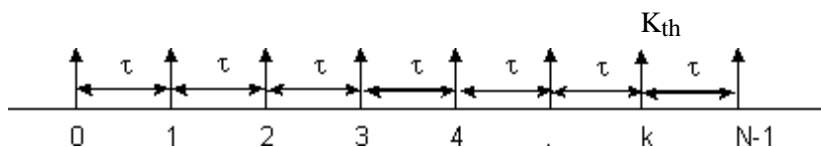


Figure 5 - Referring to individual samples

In a continuous signal we refer to a particular point at its instantaneous value of t . For discrete signals, We refer to any particular sample as $g(kt)$. So each sample differs in time from the previous one by t secs. For example the 3rd sample similarly is located at $(3 \times .05) = .15$ secs and the 10th sample is located at time $(10 \times .05) = .5$ secs..

Define the $g(t)$ in its discrete form as

$$g(t) = g(k t)$$

Harmonic index: n

Frequencies that are integer multiple of a fundamental frequency are referred to as harmonics of that fundamental frequency. In computing DFT, we use the concepts of harmonics in a special way.

From the sampling theorem, we know if we want to recreate a signal from its samples then we must collect at least twice its frequency number of samples per second. This also says that we can detect frequencies in a signal only up to one half of its sampling frequency. So the values of n ranges from

$$n \frac{f_s}{N} \leq \frac{f_s}{2}$$

or $n \leq \frac{N}{2}$

This says that we can only detect half as many harmonics as the total number of samples. However the index itself goes from $-(N-1)$ to $+(N-1)$ and spans both sides of the spectrum reflecting the positive and negative components of the frequency.

Relating the Discrete Fourier transform to the Continuous Fourier Transform

Recall that the Fourier transform is given by

$$G(f) = \int_{-\infty}^{+\infty} g(t) e^{-j\omega t} dt \quad -1-$$

The above equation says that if we multiply the target signal by a complex sinusoid of harmonic frequencies 1, 2, 3,

..n one at a time and then integrate the results, the integration yields the amplitude of the n_{th} harmonic. Why? Because multiplication by the sinusoid acts as filter for all other frequencies. (Refer to Fourier Transform Tutorials No. 1 and 2). The objective is to compute the complex coefficients c_n , which when plotted give us the frequency content of the signal.

$$a_n = \int_{-\infty}^{+\infty} g(t) \sin(n\omega t) dt = \text{amplitude of the } n_{th} \text{ harmonic sine wave}$$

$$b_n = \int_{-\infty}^{+\infty} g(t) \cos(n\omega t) dt = \text{amplitude of the } n_{th} \text{ harmonic cosine wave}$$

$$c_n = \sqrt{a_n^2 + b_n^2} \text{ the complex coefficients which produce the spectrum}$$

The integration limits in the Fourier transform formula of Eq. 1 go from $-\infty$ in to $+\infty$ in. What does that mean for our sampled sequence of N samples?

Fourier Transform also requires that the signal be periodic. But looking at only N samples we can not tell if the samples cover one exact period, more than one or less than one period. In order to do the Fourier Transform, we need at least one whole period or the result is suspect.

We already see some problems as we go from continuous to discrete processing, The problems are

1. We do not have an infinitely long series and
2. We do not know if the N samples we have observed cover a single period, less than a single period or more than one.

Let's continue despite the fact that we don't know if what we are about to do is right. We are going to assume that these two things will not cause us much trouble and the results will be acceptable.

Now let's change time from continuous to discrete by making the following substitutions for time and frequency.

$$g(t) = g(k t)$$

$$\omega_n = 2\pi f_n$$

Equation 1 becomes

$$G(f_n) = \sum_{k=0}^{N-1} g(k\tau) e^{-j(2\pi f_n)(k\tau)} \quad -2-$$

We have also changed the integral to summation to note the change from continuous to discrete as two processes are equivalent in the two domains.

Fundamental frequency of the signal

By applying the Fourier Transform algorithm on these N samples, we have made an implicit assumption. We have assumed that the signal is periodic over N samples, so we have assumed that the fundamental frequency of our signal is equal to the inverse of time T of the N samples. We express the fundamental frequency as

$$f_0 = \frac{1}{T}$$

We can rewrite T as a function of sample time, t and total number of samples chosen, N to alternately express the fundamental frequency in terms of f_s and N.

$$f_0 = \frac{1}{\tau N} = \frac{1}{\frac{\text{sec s}}{\text{sample}} \times \text{Total no. of sample}}$$

$$f_0 = \frac{f_s}{N}$$

This is a very important concept to understand. It says that you have artificially set the fundamental frequency to the sampling frequency divided by the total number of samples observed. It is a strange idea seemingly having nothing to do with the target signal and in fact this is true.

This frequency referred to as the *fundamental frequency* of the signal really is kind of a resolution frequency and has nothing to do with the target signal. It just means that we resolve the target signal components in integer multiples of this resolution frequency.

Let's say that we sampled the above signal at sample time of $1/20^{\text{th}}$ sec and observed 60 samples. Then the fundamental frequency is

$$f_0 = \frac{f_s}{N} = \frac{20}{60} = .333 \text{ Hz}$$

Now when we compute the Fourier Transform we will be stepping this fundamental frequency by integer multiples. With $f_0 = .333$, the next harmonic would be $f_1 = .666$ and so on. The harmonics used in the analysis are not integers but *integer multiple* of the fundamental frequency of the signal as determined by the sampling frequency and the N samples observed. An alternate way to see these harmonics is see them as bins which collect energy. In DFT they are also called cells.

Now the n^{th} harmonic can be expressed as n times f_0 .

$$f_n = n f_0$$

this is also equal to

$$f_n = \frac{n}{\tau N} = \frac{n f_s}{N}$$

Now we rewrite the Fourier Transform substituting above expression for f_n in Eq. 2.

$$G(f_n) = \sum_{k=0}^{N-1} g(k\tau) e^{-j(2\pi \frac{n}{\tau N})(k\tau)}$$

t's cancel and we get,

$$G\left(\frac{n}{\tau N}\right) = \sum_{k=0}^{N-1} g(k\tau) e^{-j2\pi \frac{n}{N} k}$$

$$G\left(\frac{n}{\tau N}\right) = \frac{1}{N} \sum_{k=0}^{N-1} g(k\tau) e^{-j2\pi \frac{n}{N} k}$$

-3-

The above form of the Fourier Transform is called the Discrete Fourier Transform (DFT). The Division by N is used to normalize the values.

DFT is a special case of the Fourier Transform and is actually an approximation of the real thing. The validity of

the approximation is effected by the type of waveform we are dealing with as well as the parameters f_s and N .

Computing the DFT

The process of computing the DFT is identical to computing the Fourier coefficients we did in Tutorial 1.

Here you need to know

1. What is the sampling frequency of the target signal? Is the sampling frequency large enough so that it covers all significant frequencies in the signal?
2. How many samples do we need?

First compute the fundamental frequency, and starting with the fundamental frequency we multiply the discrete signal by a complex exponential and perform summation on the result.

Do you recall what it means to multiply by a complex exponential? How do you interpret the following equation?

$$f(t) e^{-j2\pi ft}$$

The figure below shows what is happening in real-life.

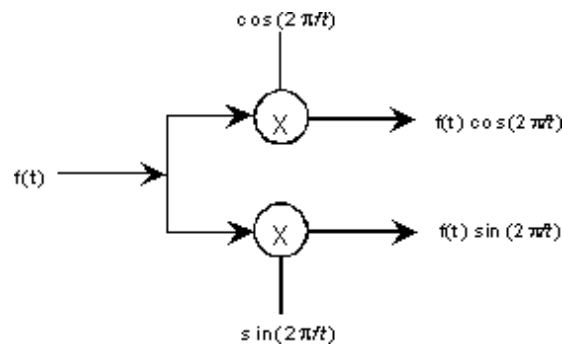


Figure 6 - What we are really doing when we multiply by a complex exponential

The signal in fact is being split into two parts, 1. multiplied by a sine wave and the other by a cosine of the same frequency. The resulting two signals are orthogonal and are the result of multiplication with the complex exponential or phasor.

DFT Step by step

Now we compute the DFT of the signal in Fig 1.

Step 1 - Multiply the target signal in 7a by a cosine wave in 7b of frequency f_0 . For this demonstration, we assume that f_0

= 1. (Although only cosines are shown, we do this for both sines and cosines and keep track of the results separately.)

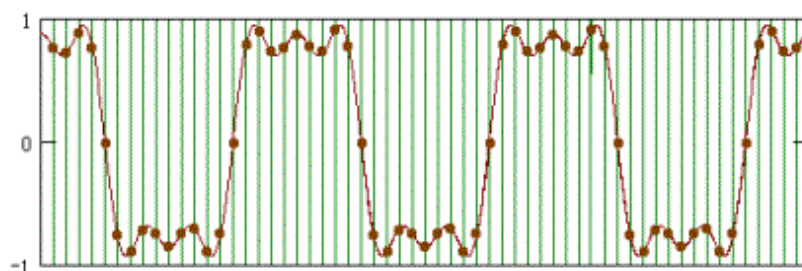


Figure 7a - The sampled target signal

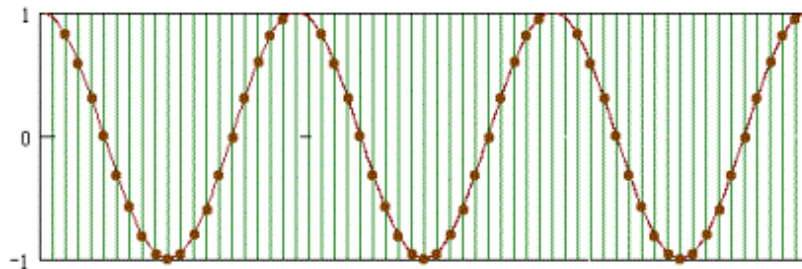


Figure 7b - First Harmonic $f_1 = 1 * f_0$

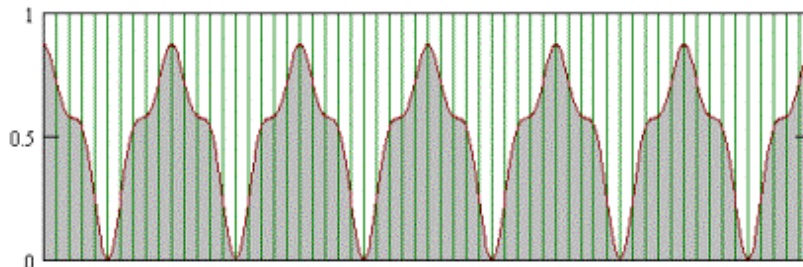


Figure 7c - result of multiplying the first harmonic with the target signal. The waveform has positive area.

The multiplication gives us the waveform in 7c. Now integrate this waveform over the N samples. In a discrete case, we integrate by multiplying the sample amplitude by the width of the base which is equal to t , the sample time, using the trapezoidal rule. We are in effect adding up the areas of all the small gray rectangles in Figure 7c. Each sample value is multiplied by t and these areas are summed.

The result of the multiplication tells us something interesting. We see that the resulting waveform is not even, so it has net area under it. This means that there is a signal hiding in this frequency. What is the amplitude of this frequency? That we know only when we complete the summation. The result of the summation gives us the amplitude of this harmonic in the target signal.

Step 2: Now multiply the Signal in 7a with the second harmonic as shown in Figure 7d. The multiplication gives the waveform in Figure 7e.

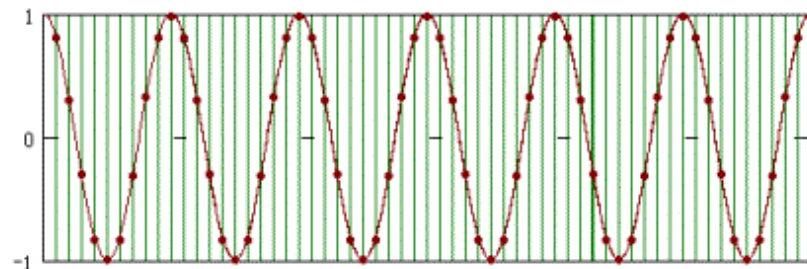


Figure 7d - 2nd Harmonic $f_2 = 2 * f_0$

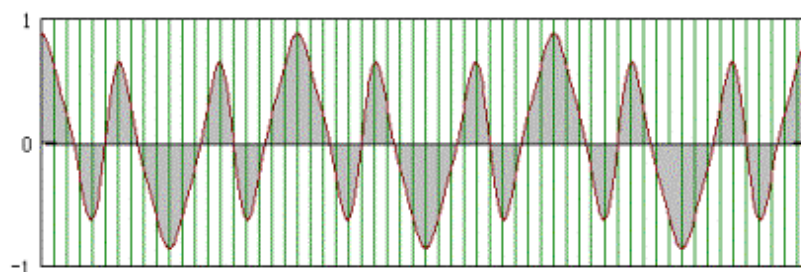


Figure 7e - Result of multiplying the 2nd harmonic with the target signal. The waveform has no area.

The waveform of 7e is even, which means that the summation of the little gray rectangles will give zero area. Since it has no net area means there is nothing of interest here.

Let's go to the next harmonic. Now multiply the target signal with the 3rd harmonic as in Fig 7f. The resulting waveform is shown in Fig 7g.

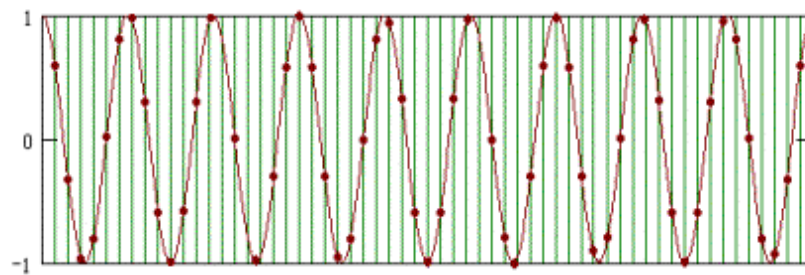


Figure 7f - 3rd Harmonic $f_2 = 3 * f_0$

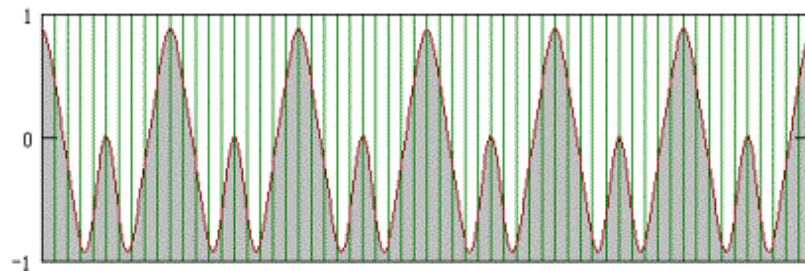


Figure 7g - Result of multiplying the 3rd harmonic with the target signal. The waveform has positive area.

Once again, we get a waveform that has a DC component, so we will do the summation to figure out the amplitude.

In Figure 7h and 7i we show the results of multiplication with harmonics 4 and 5. For 5th harmonic we once again get a positive DC term.

We will stop here because we already know that our signal only has three frequencies. And we have found them all.

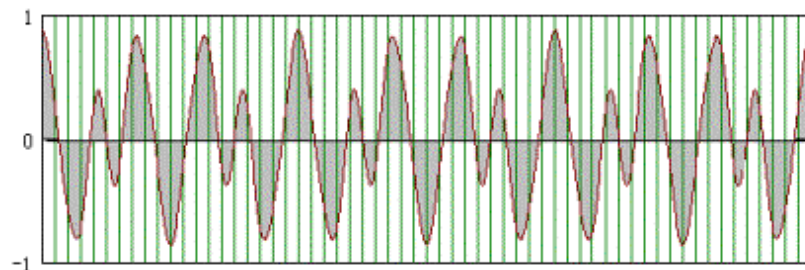


Figure 7h - Result of multiplying the 4th harmonic with the target signal. The waveform has no area.

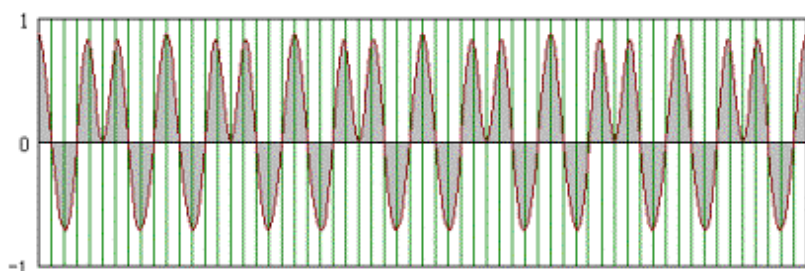


Figure 7i - Result of multiplying the 5th harmonic with the target signal. The waveform has positive area.

Fast Fourier Transform (FFT)

The DFT although clear and easy to compute requires a good many calculations. For each harmonic, we have $N+1$ multiplications and we do this N times, giving us $N^2 + N$ calculations. A 256 point DFT would require 65792 calculations. The algorithm for Fourier Transform existed for more than 200 years before it came into widespread use mostly because we could not cope with such large number of computations. The algorithm waited for the development of the microprocessor.

In 1948, Cooley and Tukey and came up with a computational breakthrough called the Fast Fourier Transform algorithm. This was an ingenious manipulation of the inherent symmetry of the calculations. It now allowed the computation of a N point DFT as a function only $2N$ instead of the N^2 . So a 256 point DFT would only require 512 calculations, a huge improvement from 65792 calculations doing it the laborious way. The algorithm was quickly and widely adopted and is the basis of all modern signal processing.

Most DSP books spend a lot of time going through the mechanism of the FFT and how it is computed. Huge butterfly figures in our books help to confuse and confound us as to what is actually going on. Most of us just give up on it. Let me alleviate your guilt. Although extremely important in itself, the understanding of the mechanism of the FFT is not all that important. So, I am skipping over the details of its implementation.

The method has been programmed in all sort of software and we can safely skip it without impacting our understanding of the *application* of the DFT and the FFT.

The main thing one needs to know about the FFT is that it works only with sample numbers that are powers of 2, such as 16, 32, 64 etc.. We cannot do a FFT on an arbitrary number of samples as we can with DFT which is a generic process. The FFT is a DFT with constraints on the number of samples.

The other thing about the FFT process to know is that it allows zero-padding. Let's say we have 28 samples and we wish to do a DFT via the FFT, we can do two things, 1. we can do a 16 point FFT and discard the remaining 12 points or we can insert four zeros at the end so we have 32 points. Now we can do a 32 point FFT. The zero-padding provides us better resolution but does not provide any extra information. The frequency detected is still a function of the original N samples and not the zero-padded length, although the FFT does *look* a lot better. And looks count.

FFT of a signal

Now we will do an example using the signal of Eq 1. This signal has three frequency components. I have sampled it at the rate of 16 samples per second. We will do several FFTs, from lengths 16 to 256 samples and examine the impact of the sampling frequency and the number of samples.

Before we do that let's review the process of doing a FFT as shown in Fig. 8

1. We sample our continuous signal with an infinite length impulse train of frequency f_s .
2. We then limit the result to a certain number of samples by multiplying the result of step 1 by a rectangular function of period T . All values outside this window are now zero.
3. We realize that the FT of this limited signal is continuous because the Fourier Transform of all periodic-discrete signals is continuous. We are after the discrete Fourier Transform, so to do that we sample the limited signal in the frequency domain.
4. This results in a replicated signal. We first windowed the signal to make it non-periodic and then after windowing it, we have once again made it periodic. This assumes that the windowed section is a true periodic representative of the whole signal. And that by making it periodic again, we get the original signal back. This is of course, just an assumption, which is rarely true for real signals.
5. The Fourier Transform of this new periodic signal contains the effects of the windowing function and as such is smeared. This smearing is most pronounced when the window size is not an integral number of the signal period.

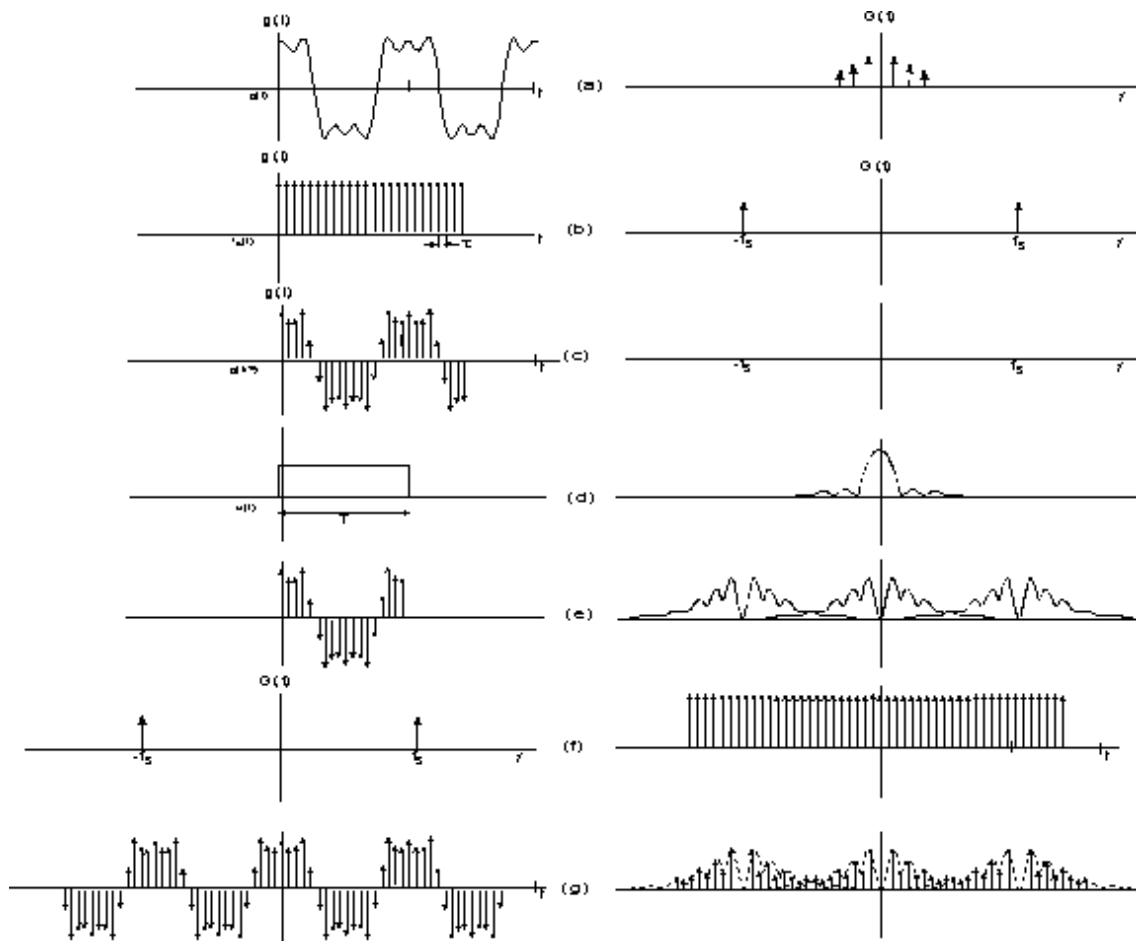


Figure 8 - DFT/FFT of a signal

Doing FFT of the signal g(t) of Fig. 1

$$g(t) = \cos(2\pi f_a t) - \frac{1}{3} \cos(2\pi 3 f_b t) + \frac{1}{5} \cos(2\pi 5 f_c t)$$

We sample g(t) at the rate of 16 samples per second. Each frequency component of g(t) is then also sampled at 16 samples per second. In each 16 samples, there are an integer number of cycles of each of the three frequencies as shown in Figure 9. (There is nothing magic about these numbers, all integers frequencies will give an integer number of cycles per one cycle of the sampling frequency. But had one of the frequencies been 3.5 instead of 3, we see that then we would get a non integer number of cycles per 16 samples.)

- f_a = 1, contains 1 complete cycle in 16 samples
- f_b = 3, contains 3 complete cycles in 16 samples
- f_c = 5, contains 5 complete cycles in 16 samples

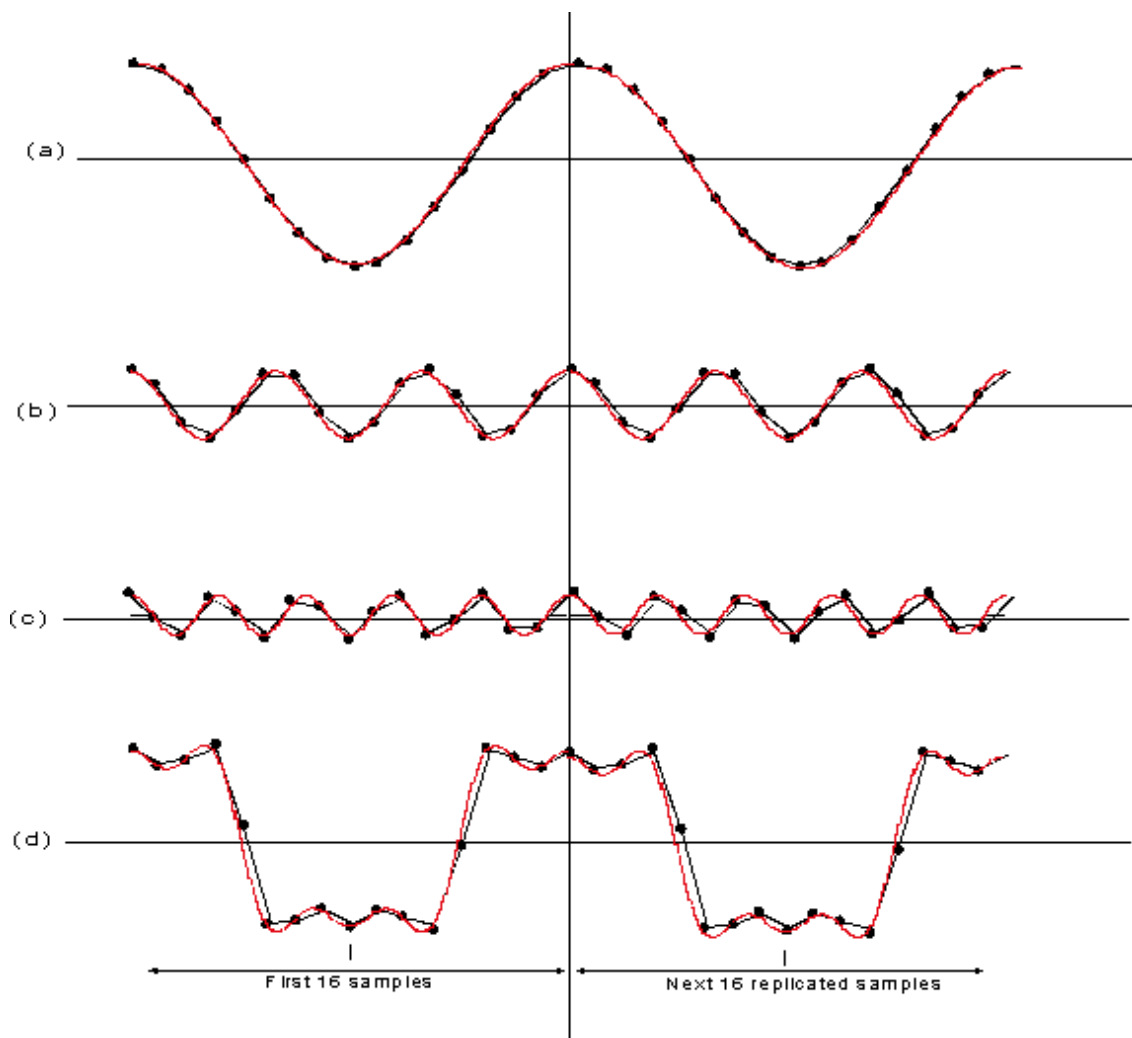


Figure 9 - Each component of $g(t)$ (f_a), (f_b) and (f_c) and the composite signal $g(t)$ which is the sum of (a), (b) and (c) shown in (d) are sampled at 16 times per second. When replicated the 16-sample pieces form the original signal without discontinuities.

N which is a multiple of 16 then always contains an integer number of cycles of each of these frequencies.

In the following examples, we show the effect of f_s and N on the results we obtain. Remember that we want the spectrum in Figure 2 based on theory.

Case 1. We decide to use only 16 of the collected samples. $F_s/N = 1$ This means that each frequency varies by 1 Hz. We see seven points. The values at frequencies 1, 3 and 5 are correct but in between this is a terrible looking representation of the real FT. All spectrums shown are one-sided and were done in SPW.

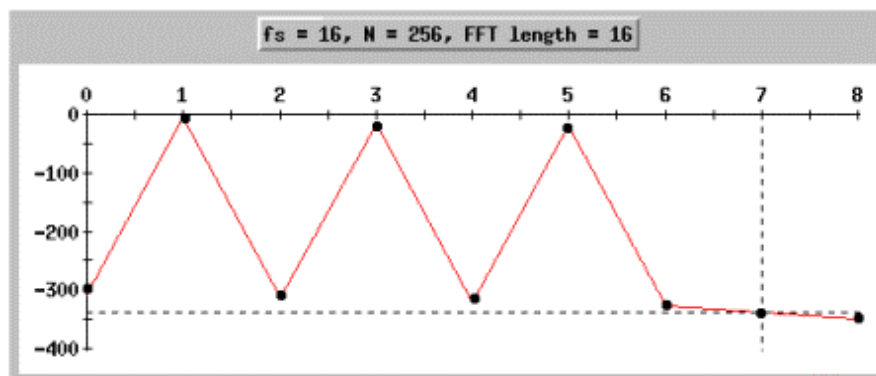


Figure 10a - FFT of a three-frequency signal with $N = 16$

Case 2. Here we used 32 samples instead of 16. Now we have increased the resolution to $f_s/N = 16/32 = .5$. Now we get 14 points, each .5 Hz away.

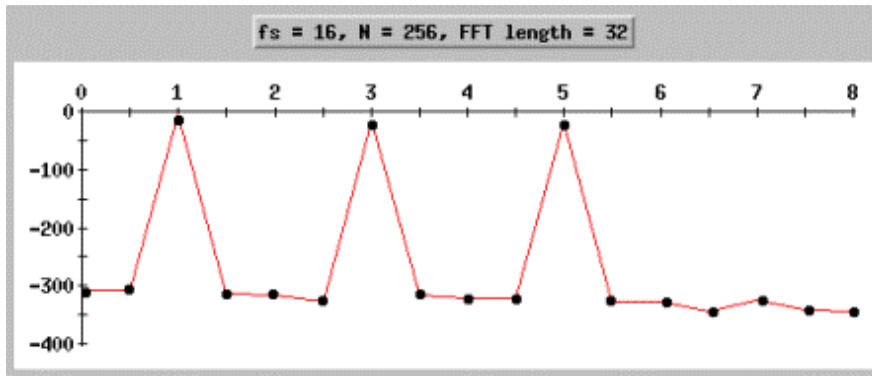


Figure 10b - FFT of a three-frequency signal with $N = 32$

Case 3: $N = 64$ samples Now $f_s/N = 16/64 = .25$. This Fourier Transform is looking much better.

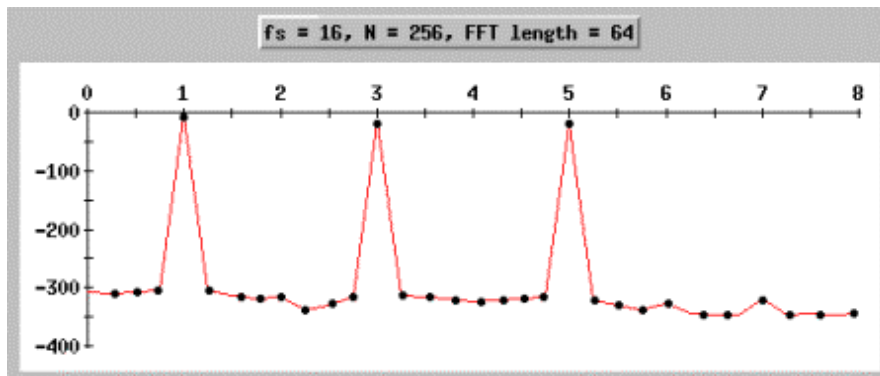


Figure 10c - FFT of a three-frequency signal with $N = 64$

Case 4: $N = 128$, The FFT looks nearly like the theory.

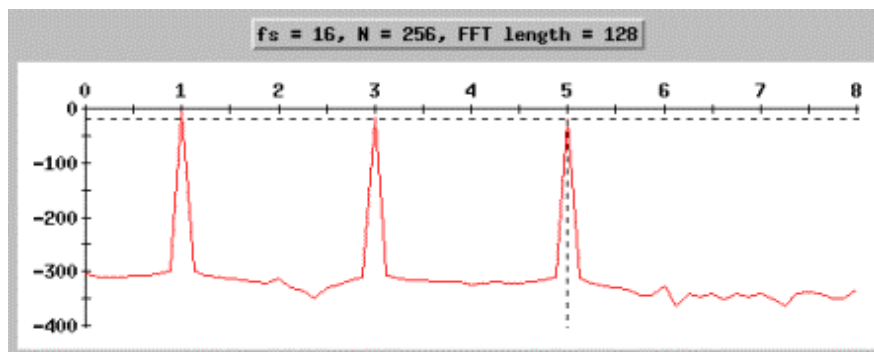


Figure 10d - FFT of a three-frequency signal with $N = 128$

Case 5: $N = 256$, This FFT looks quite satisfactory.

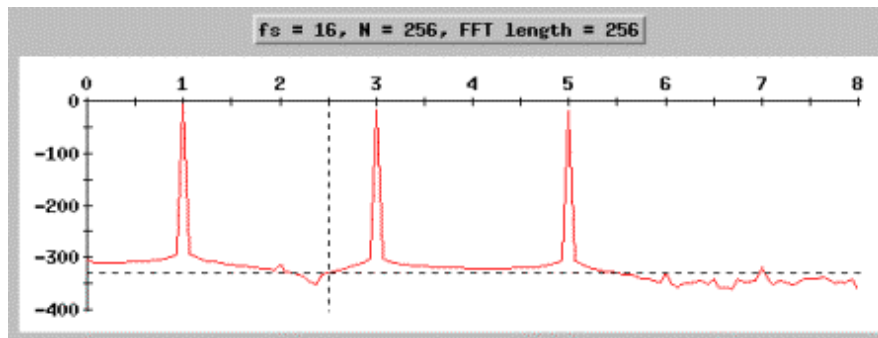


Figure 10e - FFT of a three-frequency signal with $N = 256$

What conclusion can we draw from these? It is clear that the factor f_s/N has the largest impact. So we can always improve the FFT by increasing the size of the FFT. The smaller the f_s/N parameter, the better the resolution of our spectrum.

However, in this example all things were perfect. The window size contained exactly an integer cycles of the target signal.

What happens when the window size does not cover an integer multiple number of cycles?

Case 2. Sampling frequency not an integer multiple of the FFT length

In this case, I have sampled the signal with $f_s = 20$ instead of 16. We will use 32 as the smallest FFT size. Let's look at what portion of the signal we capture in these 32 samples.

$f_a = 1$, contains 1 complete cycles in 20 samples or 1.6 cycles per 32 samples

$f_b = 3$, contains 3 complete cycles in 20 samples or 4.8 cycles per 32 samples

$f_c = 5$, contains 5 complete cycles in 20 samples or 8 cycles per 32 samples

(please note that the 3rd frequency does have an integral number of cycles per 32 samples.)

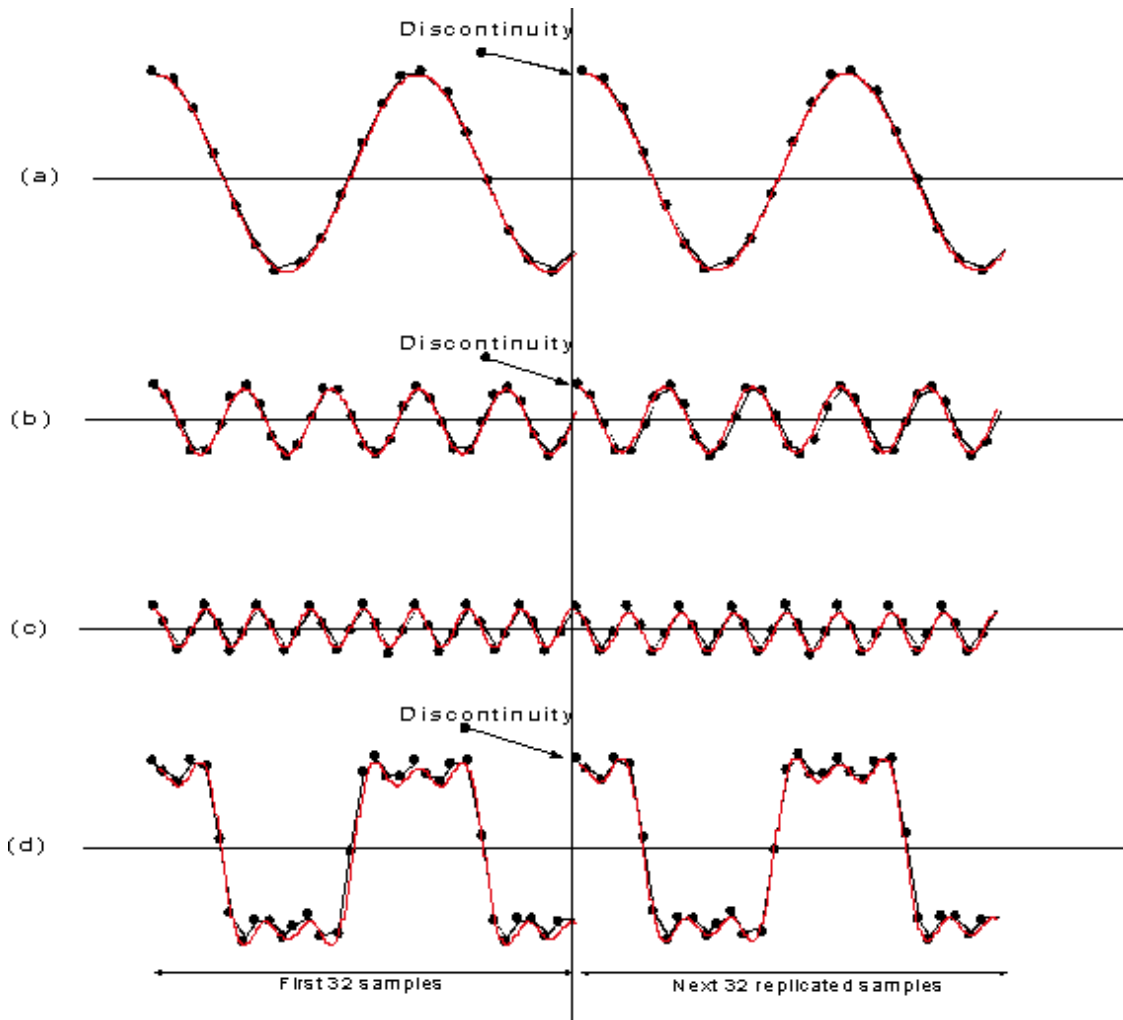


Figure 11 - Each component of $g(t)$ is sampled 20 times per second. When replicated the 20-sample pieces do not form the original signal. The replicated signal has discontinuities.

Now we do FFT of this signal. N which is always a multiple of 16 contains a non-integer number of cycles of the 1st and the 2nd frequencies. We will see smearing in these two frequencies caused by the lack of a truly periodic signal.

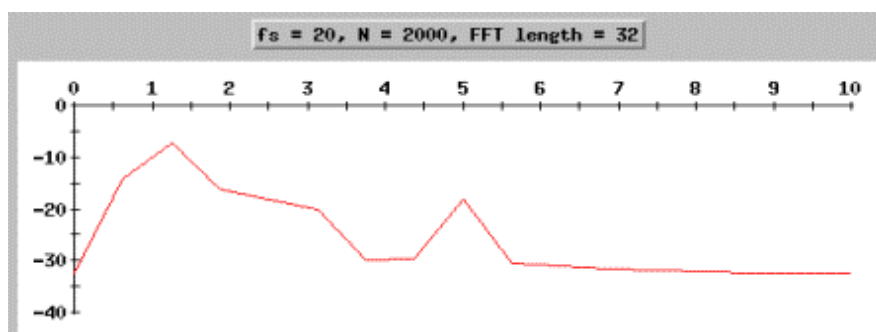


Figure 12a - FFT of a three-frequency signal with $N = 32$

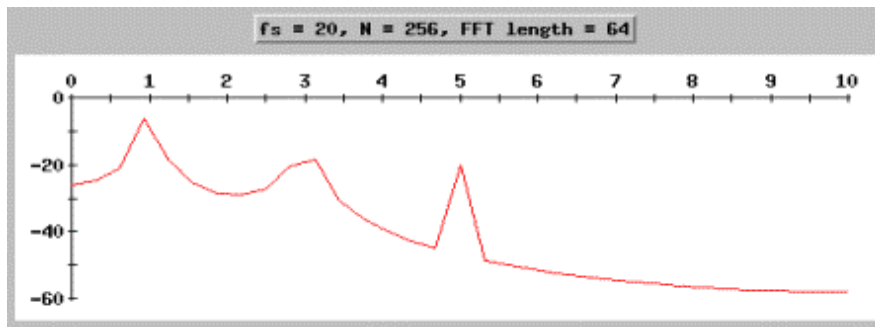


Figure 12b - FFT of a three-frequency signal with $N = 64$

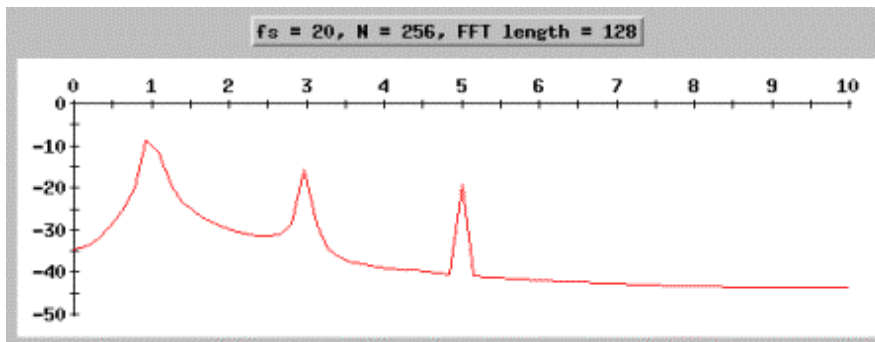


Figure 12c - FFT of a three-frequency signal with $N = 128$

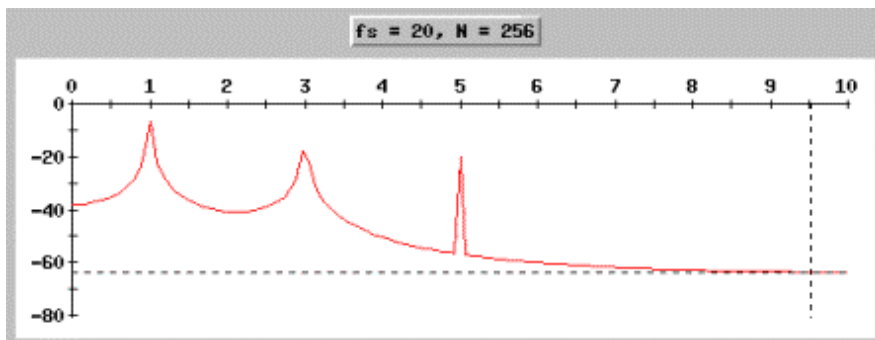


Figure 12d - FFT of a three-frequency signal with $N = 256$

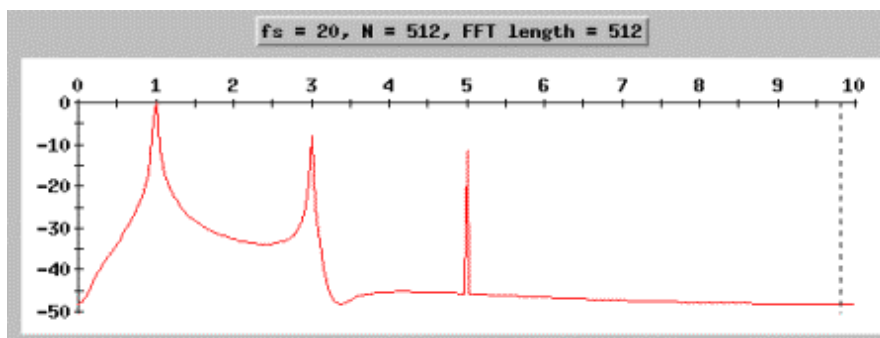


Figure 12e - FFT of a three-frequency signal with $N = 512$

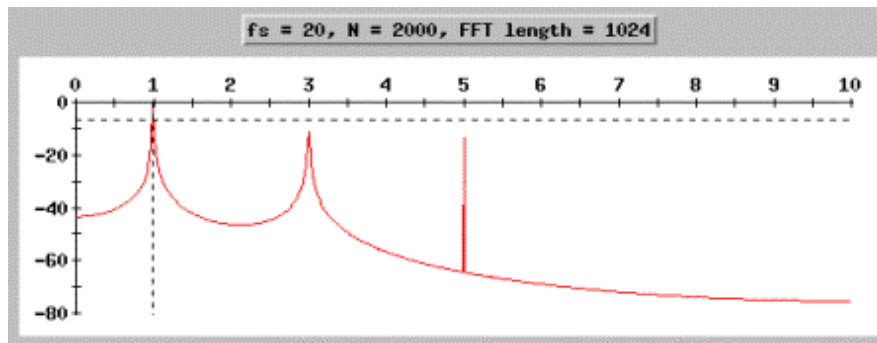


Figure 12f - FFT of a three-frequency signal with N = 1024

Compare this FFT to the 256 point FFT for the first case. Both signals are the same, the only difference is the sampling frequency. In the second case the FFT length no longer covers an integer number of cycles and so discontinuities are present in the signal.

However, it is interesting to note that the representation is not useless, and is in fact quite good. The last frequency which we know is 5, gives us quite good results. Why? Because frequency of 5 divides evenly into the FFT lengths of 32, 64, 128, 256, 512 and 1024 so it is represented faithfully.

A general rule can be stated as:

$$\frac{f_s f_{sig}}{2^n} = \text{integer}$$

where f_s = sampling frequency, f_{sig} = signal frequency, 2^n are all numbers that are powers of 2 such as 16, 32, 64, 128, etc.

Example: $f_s = 20$

$$f_a = 1$$

No solution to above equation for N.

$$f_b = 3$$

No solution to above equation for N.

$$f_c = 5$$

for $n = 5$, we can solve the above equation. The minimum FFT size is 2^5 or 32 and for this signal frequency and sampling frequency combination, we will get a good FFT as we saw in the example FFT.

We can also draw this conclusion: if we do a large enough FFT, we can to a large extent overcome the problems associated with unknown truncations and can get acceptable results. A concept called windowing the function with something other than the rectangular function we used also helps to reduce these effects. We will discuss this in the next issue.

Zero padding

In most case, the FFT length is always less than the number of samples available on hand. But sometimes we do an FFT that is larger than the number of samples. In this case, the algorithm appends a bunch of zeros to the end of the signal to artificially make it longer. Doing so sometimes gives better resolution for the signal space and also gives us “good looking” spectrums. Whether or not to zero pad is a subjective thing and depends on how much data you have and what type of signal it is.

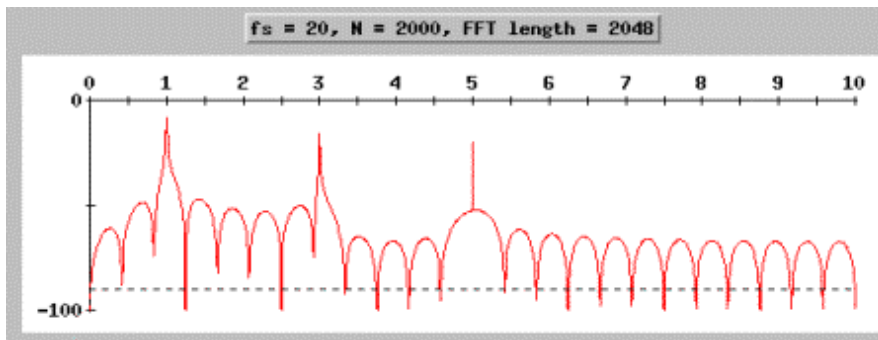


Figure 13a - FFT of a three-frequency signal with actual samples = 2000,
Zero-padded to N = 2048

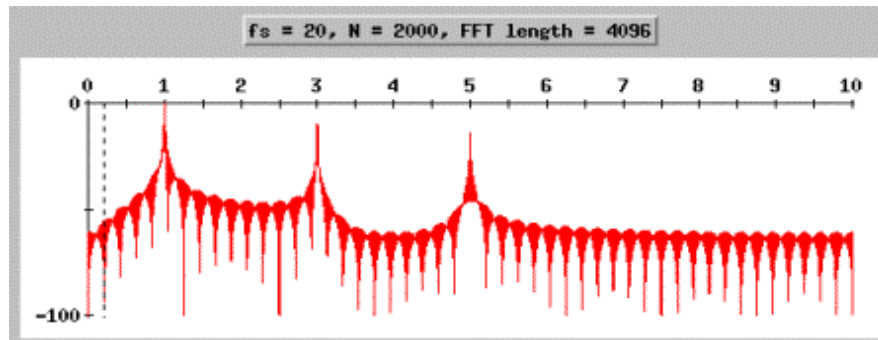


Figure 13b - FFT of a three-frequency signal with actual samples = 2000,
Zero-padded to N = 4096

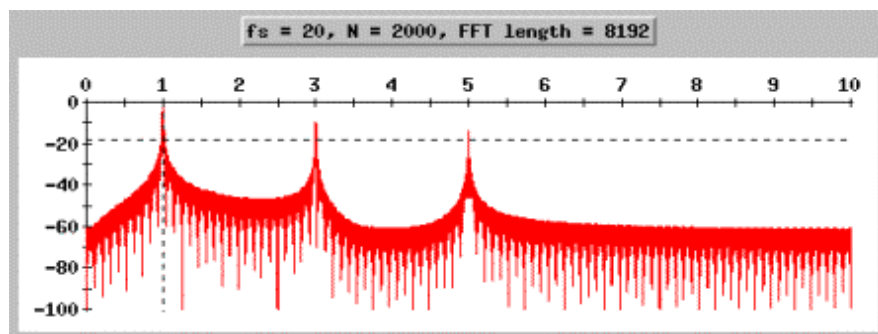


Figure 13c - FFT of a three-frequency signal with actual samples = 2000,
Zero-padded to N = 8192

This is a good looking spectrum and demonstrates that the resolution, f_s/N is an important parameter of how good of a FFT we will get. The above FFT created from 2000 points and then zero padded to 8192 is really very good. The fact that the input signal was not a faithful reproduction of the actual signal is nearly inconsequential.

How wonderfully forgiving and idiot proof DFT/FFT is!

Next: Periodogram, aliasing, leakage and windowing and other approximations/modifications of the Fourier Transform

By: Charan Langton
Complexoreal.com
mntcastle@earthlink.net

