The art of signal processing

Alex Townsend

April 28, 2015

Signal processing relies on the FFT. That top-ten algorithm again. The task is to sample a signal:

$$f(t) = \sum_{n=0}^{\infty} \underbrace{A_n}_{\pm\text{amplitude}} \cos(2\pi \underbrace{n}_{\text{frequency}} t + \underbrace{a_n}_{\text{phase}}),$$

and recover the amplitude, frequency, and phase. We know from Nyquist's sampling law that we must sample at least 2 times per wavelength.

Nyquist's sampling law:
"To recover a signal it must be sampled at least two times per wavelength."

So if we take N samples, then we cannot hope to recover signals with frequency more than $N/2-1$. Thus, we determine/approximate A_n and a_n such that

$$f(k/N) = \sum_{n=0}^{N/2-1} A_n \cos(2\pi nk/N + a_n).$$

Since $\cos(z) = (e^{iz} + e^{-iz})/2$, this is the discrete Fourier transform in disguise. (You may wish to skip the derivation below, like many signal processing engineers.)

$$\begin{aligned}
\sum_{n=0}^{N/2-1} A_n \cos(2\pi nk/N + a_n) &= \sum_{n=0}^{N/2-1} \frac{1}{2} A_n e^{a_n i} e^{2\pi ink/N} + \sum_{n=0}^{N/2-1} \frac{1}{2} A_n e^{-a_n i} e^{-2\pi ink/N} \\
&= \sum_{n=0}^{N/2-1} \frac{1}{2} A_n e^{a_n i} e^{2\pi ink/N} + \sum_{n=N/2}^{N-1} \frac{1}{2} A_{n-N/2} e^{-a_{n-N/2} i} e^{-2\pi i(n-N/2)k/N} \\
&= \sum_{n=0}^{N/2-1} \frac{1}{2} A_n e^{a_n i} e^{2\pi ink/N} + \sum_{n=N/2}^{N-1} \frac{1}{2} A_{N-n-1} e^{-a_{N-n-1} i} e^{2\pi ik/N} e^{2\pi ikn/N} \\
&= \sum_{n=0}^{N/2-1} B_n e^{2\pi ink/N},
\end{aligned}$$

where

$$B_n = \begin{cases} \frac{1}{2}A_n e^{a_n i}, & 0 \le n \le N/2 - 1, \\ \frac{1}{2}A_n e^{a_n i} e^{2\pi i k/N}, & N/2 \le n \le N - 1. \end{cases} \tag{1}$$

Our task is to compute B_n from samples $f(k/N)$ and then to work out A_n and a_n. You will notice that

$$f(k/N) = \sum_{n=0}^{N/2-1} B_n e^{2\pi i n k/N}, \qquad 0 \le k \le N - 1$$

is not quite the DFT. There is a minus sign missing, and we want the inverse process: B_n from $f(k/N)$ not $f(k/N)$ from B_n. These two cancel each other out. It really is the FFT. I told you the FFT is like superman.

So we take samples of our signal, $f(k/N)$, compute the FFT of them to get B_n, and then undo (1). We have

$$\text{amplitude} = |A_n| = |2B_n e^{-a_n i}| = 2|B_n|.$$

The frequencies are the non-zero values of B_n. Lastly, we have

$$\text{phase} = a_n = \tan^{-1}\left(\frac{\text{Im}(B_n)}{\text{Re}(B_n)}\right).$$

Do not apply the formula for a_n, unless B_n is nonzero.

Warning: In MATLAB there is a scaling to take care of. Divide B_n by N after the FFT.

What happens if the signal is composed of non-integer frequencies? For example,

$$f(k/N) = \sum_{n=0}^{N/2-1} A_n \cos(2\pi\omega_n k/N + a_n).$$

Then, you will need extra ideas such as zero padding and windowing. (See class notes for brief introduction.)