CASPER Memo 41
The Polyphase Filter Bank Technique

Jayanth Chennamangalam[*]

Original: 2011.08.06
Modified: 2014.04.24

Introduction to the PFB

In digital signal processing, an instrument or software that needs to do Fourier analysis of some input signal performs a Discrete Fourier Transform (DFT). The straightforward application of the DFT on an input signal suffers from two significant drawbacks, namely, leakage and scalloping loss.

DFT leakage is the phenomenon in which, depending on the sampling frequency and the number of points in the transform, an input tone appears in more than one output frequency bin.[1] If this tone is not strong enough, this effect can go unnoticed. But in the case of, say, a strong radio frequency interference (RFI) signal, the leakage can drown out astronomical signals of interest in the nearby bins. This effect is shown in Figure 1 and is described in more detail in the following section.

DFT scalloping loss is the loss in energy between frequency bin centres due to the non-flat nature of the single-bin frequency response.

The polyphase filter bank (PFB) technique is a mechanism for alleviating the aforementioned drawbacks of the straightforward DFT. The PFB not only produces a relatively flat response across the channel, but also provides excellent suppression of out-of-band signals, as shown in Figure 2. A system that implements a PFB in addition to a DFT typically consumes about 1.5 times more resources than one that does not. In many cases, the data quality advantages outweigh this increase in cost. Spectrometers and correlators are typical beneficiaries of the PFB technique.

[*] Email: jayanthc@gmail.com

[1] For the purpose of this memo, the input signal is a time series and the output is in the frequency domain.

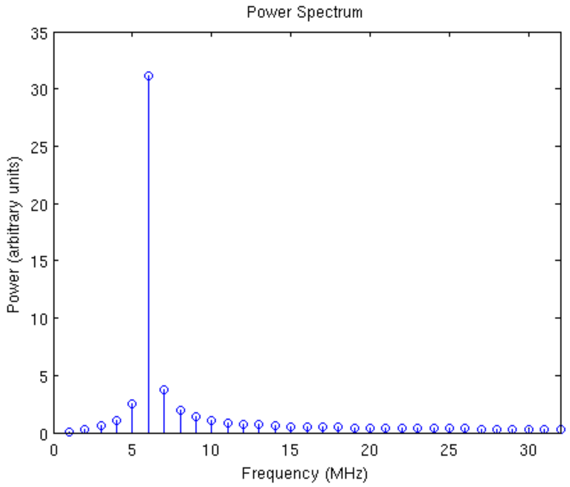

Figure 1: Demonstration of DFT leakage - a tone at 5.1MHz, sampled at 128MHz, and Fourier-transformed with 64 points, appears to varying levels in all the output frequency bins.

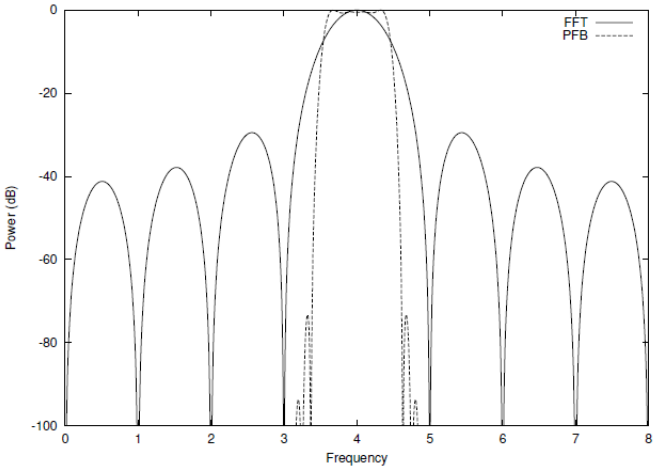

Figure 2: Comparison of the single-bin frequency response of a PFB with a direct FFT. Here, the length of the polyphase window is 8 times the length of the FFT. (Image courtesy: Dan Werthimer)

The Math Behind the PFB

The DFT of a sequence of values $x(n)$, sampled at a rate f_s, is defined as

$$X(k) = \sum_{n=0}^{N-1} x(n) e^{-i2\pi nk/N}, \tag{1}$$

where, for the purpose of this memo, $x(n)$ is a time series and $X(k)$ is in the frequency domain. From this definition it is clear that the DFT operates on a *finite* length N of time samples. Stated another way, the input to the DFT is equivalent to the product of an infinitely long time series and a rectangular window that fits over our time interval of interest. This implies that the frequency domain response of a complex sinusoidal waveform using the DFT would be the convolution of the Fourier Transform of the sinusoid and that of the rectangular window.[2] Since the Fourier Transform of the complex sinusoid is a shifted delta function, the result of the convolution is the Fourier Transform of the window – a sinc function – centred at the location of the delta function. Fortuitous combinations of N, f_s, and the input frequency can be such that the zeroes of the sinc function coincide with the bin centres of all other frequencies, in which case the problem is non-existent. But in general, the frequency domain bin centres lie at non-zero locations on the sinc function. That is, a single tone appears to some level in *all* the frequency bins of the DFT output. Since the energy contained in the input frequency bin 'leaks' into other frequency bins, this effect is called DFT leakage.

The solution to DFT leakage involves suppressing the side-lobes of the aforementioned sinc function by changing the single-bin frequency response of the DFT to approximate a rectangular function. This is achieved as follows.

Instead of taking an N-point transform directly, a block of data of size $N \times P = M$ is read, and multiplied point-by-point with a window function (in other words, the data is 'weighted'). As mentioned before, the shape of the window function determines the shape of the single-bin frequency response. Since we wish the single-bin frequency response to resemble a rectangular function as much as possible, we choose its Fourier Transform pair, the sinc function, as our window function. Once the multiplication is done, the block of data is split into P subsets of length N each, and added point-by-point. This array is then passed to a regular DFT routine to get an N-point transform that exhibits less leakage. This method is presented graphically in Figure 3.

The weighting/windowing can be thought of as a filtering process in which the elements of the window function are the filter coefficients. Mathematically, this process is given by

$$y(n) = \sum_{p=0}^{P-1} x(n + pN) h(n + pN) \tag{2}$$

where the sub-filter coefficients $h(n + pN)$ correspond to what are called P-tap 'polyphase sub-filters'. The N such polyphase sub-filters that make up this operation, together with

[2] The response would be the *sampled* convolution of the Discrete Time Fourier Transform (DTFT) of the sinusoid and that of the rectangular window, to be precise.

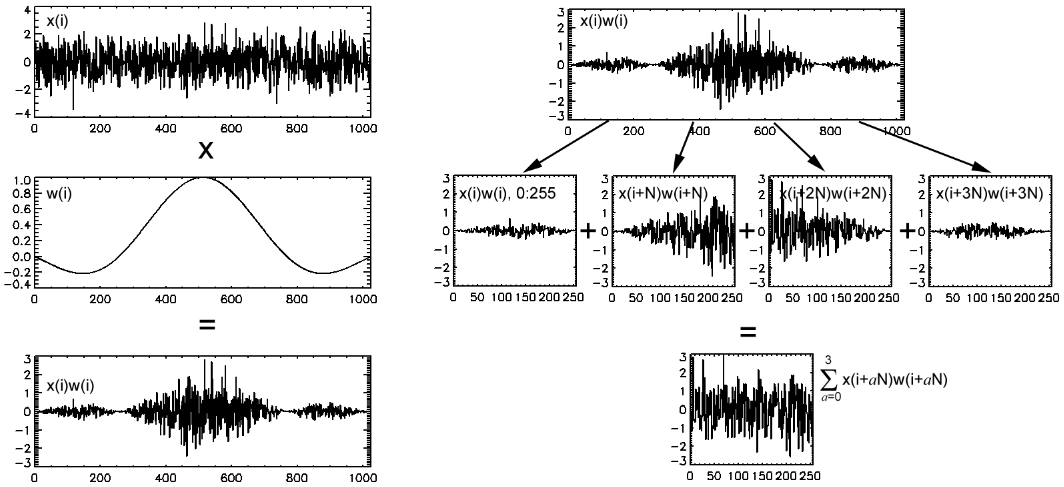

Figure 3: Graphical depiction of polyphase filtering. $x(i)$ is a time series of length $M = 1024$ samples, multiplied point-by-point with the window function $w(i)$ (a sinc function), also of the same length. The product is split into $P = 4$ blocks of length $N = 256$ samples each, and summed. This summed array of length $N = 256$ samples, shown at the bottom, on the right, is then input to a routine that takes a 256-point Fourier Transform. (Image courtesy: Dale Gary, [1])

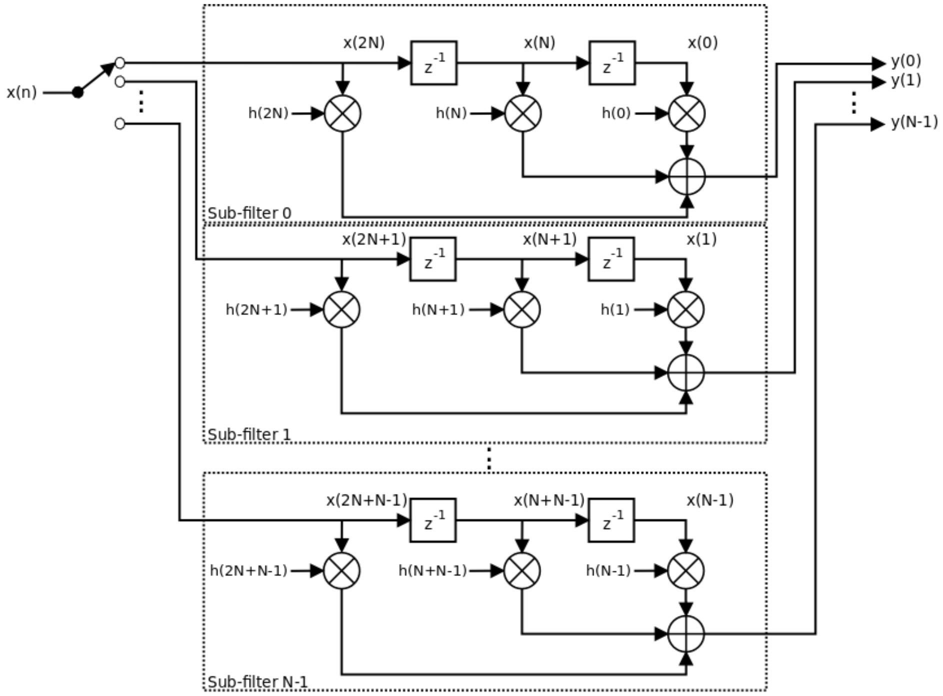

Figure 4: The FIR filter structure realization of a polyphase filter bank with $P = 3$ taps and N sub-filters. The commutator at the left rotates in the clockwise direction, and makes one complete rotation in the duration of one unit delay. The output of this structure is $y(n)$ given in Equation 2, which is the input to an N-point DFT.

the FFT filterbank stage that follows this, are collectively called a 'polyphase filter bank' ('PFB'). A realization of this filter bank is shown in Figure 4.

Since $h(n+pN)$ is a decimated-by-N version of $h(n)$, if the original filter has a pass-band width of f_s/N, each sub-filter has a pass-band width of f_s. (In other words, the original filter $h(n)$ is designed such that it has a pass-band width of f_s/N.) Since complex input data has a bandwidth of f_s, each sub-filter is essentially an all-pass filter. The only difference between the sub-filters is their phase response, which is why this structure is called a 'polyphase' filter bank.

This method is also known as 'weighted overlap-add' ('WOLA'), or 'window pre-sum-FFT'.

To suppress the sidelobes of the single-bin frequency response further, the sinc function that makes up the filter coefficients can be weighed with a smooth function, such as the Hanning window.

Implementations

Various implementations of the PFB are available online. The CASPER library comes with the `pfb_fir` (`http://casper.berkeley.edu/wiki/Pfb_fir`) and `pfb_fir_real` (`http://casper.berkeley.edu/wiki/Pfb_fir_real`) blocks that can be used with an FFT block. A stand-alone spectrometer program written in C, that reads 8-bit, complex, dual-polarisation data from a file and performs the PFB technique, and a CUDA equivalent, are available for download from the VEGAS git repository[3].

Useful Links and References

[1] Gary, D. E., `http://web.njit.edu/~gary/728/Lecture8.html`, Figures 6 & 7, and associated captions

[2] Lyons, R. P., *Understanding Digital Signal Processing*, §10.4 (Polyphase Filters) & §13.20 (A Practical Spectrum Analyzer)

[3] Proakis, J. G., & Manolakis, D. G., *Digital Signal Processing*, §10.5.2 (Polyphase Filter Structures)

[3] `https://github.com/casper-astro/vegas_devel/tree/master/src/gpu_dev`