Efficient Implementation of Uniform Filter Banks

A digital filter bank in general can be depicted in figure 1. Here the input $x[n]$ is a broadband signal.

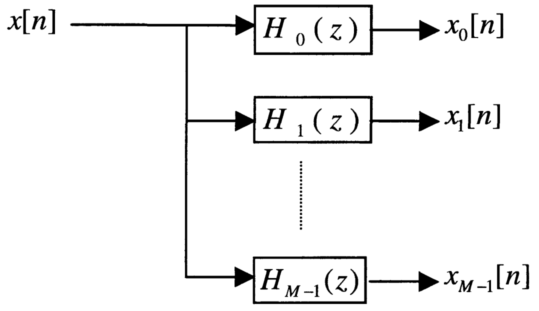

Figure 1: General digital filter bank

Suppose the filters $H_k(z)$ are frequency-shifted versions of a prototype filter $H_0(z)$, i.e.

$$H_k(z) = H_0(e^{-j2\pi k/M} z)$$

where the amount of frequency shift is $2\pi k/M$, then this is called a uniform filter bank, and its frequency response is illustrated in figure 2.

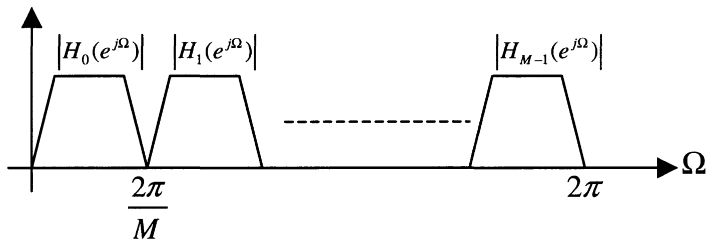

Figure 2: Typical response of a uniform filter bank

A uniform filter bank can be implemented efficiently using polyphase representation and FFT. Suppose $H_0(z)$ is FIR:

$$H_0(z) = h[0] + h[1]z^{-1} + h[2]z^{-2} + \cdots + h[N-1]z^{-N+1} = \sum_{n=0}^{N-1} h[n]z^{-n} \qquad (1)$$

Then the polyphase representation of $H_0(z)$ is (by definition)

$$H_0(z) = \underbrace{\left(h[n] + h[M]z^{-M} + h[2M]z^{-2M} + \cdots\right)}_{E_0(z^M)} + z^{-1}\underbrace{\left(h[1] + h[M+1]z^{-M} + h[2M]z^{-2M} + \cdots\right)}_{E_1(z^M)} + \cdots$$
$$= E_0(z^M) + z^{-1}E_1(z^M) + \cdots + z^{-M+1}E_{M-1}(z^M)$$
$$= \sum_{m=0}^{M-1} z^{-m}E_m(z^M)$$

The transfer function for the filter $H_k(z)$ is then

$$H_k(z) = H_0(e^{-j2\pi k/M}z) = \sum_{m=0}^{M-1}(e^{-j2\pi k/M}z)^{-m}E_m\left((e^{-j2\pi k/M}z)^M\right)$$
$$= \sum_{m=0}^{M-1} e^{-j\frac{2\pi km}{M}} z^{-m}E_m\left(z^M\right) \tag{2}$$
$$= \mathrm{DFT}_M\left\{z^{-m}E_m(z^M), M = 0,1,\ldots,M-1\right\}$$

Therefore the filter bank $H_k(z)$ can be implemented as shown in figure 3.

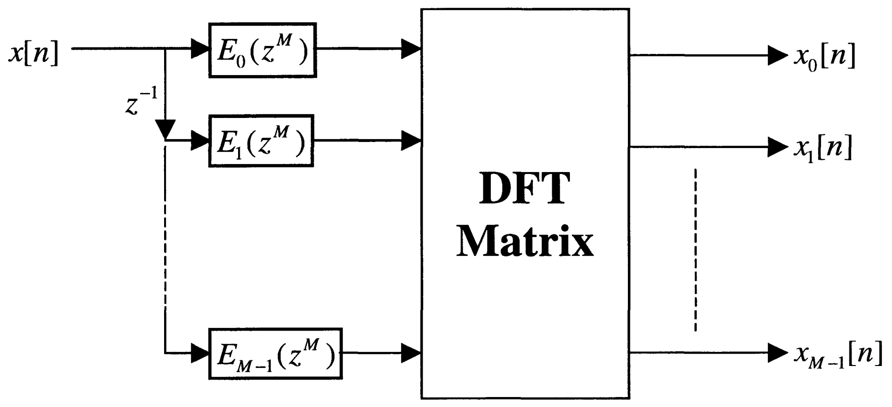

Figure 3: Efficient implementation of uniform filter bank

This implementation requires roughly $N + M\log M$ multiplications for each input/output sample, compared with NM multiplications for direct implementation of M filters. The design of uniform filter bank reduces to the design of a good prototype filter $H_0(z)$. If $H_k(z)$ are good band-pass filters with bandwidth $2\pi/M$, then it is natural to decimate $x_k[n]$ by a factor of M to reduce the output data rate, though small aliasing will occur and it is not guaranteed that $x[n]$ can be perfectly reconstructed from the decimated $x_k[n]$. The Noble identities say that the decimation can be done *before* the polyphase filters, as shown in figure 4. Here the output data rate is $1/M$ of the input, and $E_k(z)$ are FIR filters decimated from $H_0(z)$ by $e_k[n] = h[k+nM]$. The combination of time shift and decimation can be implemented by a de-multiplexer.

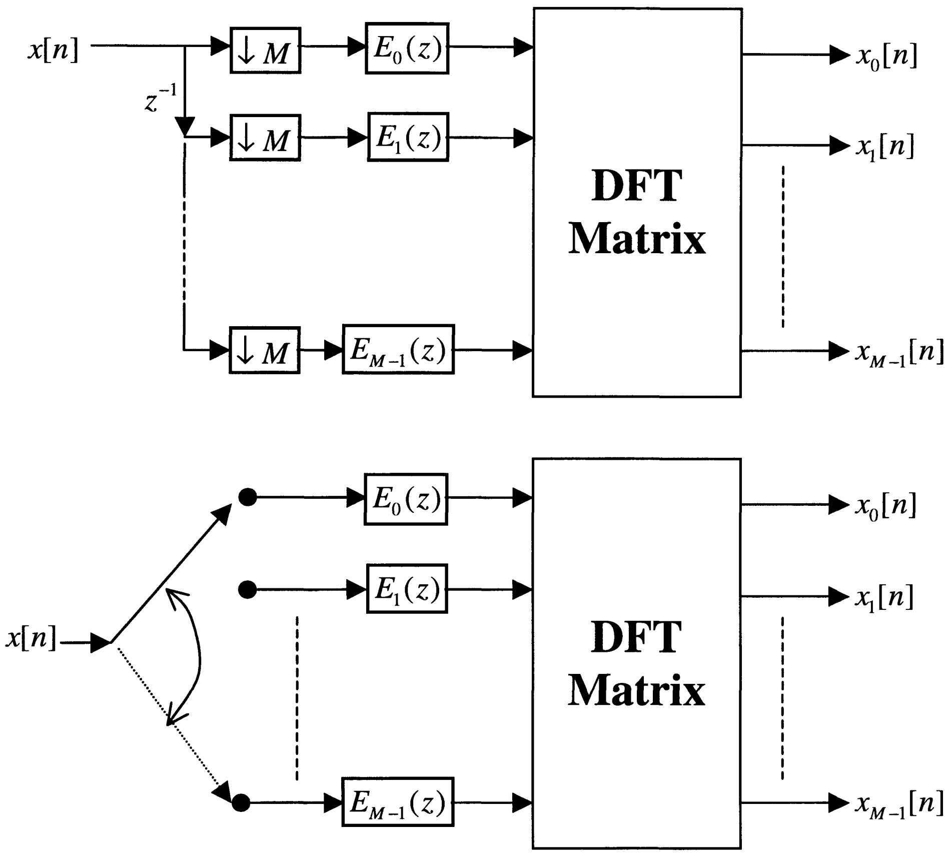

Figure 4: Decimated uniform filter bank, and its equivalent MUX form

Cosine modulated filter banks: For processing real-valued data, perhaps we want the output of the filter bank to be also real valued, i.e., we want the magnitude response of the filters to be like figure 5.

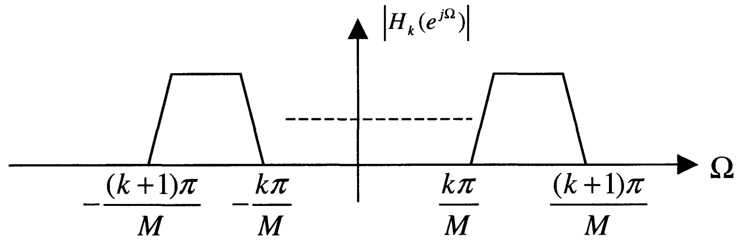

Figure 5: Frequency response of a channel in a cosine modulated filter bank

Then we want the so-called cosine modulated filter bank rather than the exponential modulated filter bank. It can be shown that equation (2) is replaced by

$$H_k(z) = \sum_{n=0}^{2M-1} T_{kn} z^{-n} G_n(-z^{2M}), \quad k = 0,1,\ldots,M-1$$

where $G_n(z^{2M})$ are polyphase components of the prototype filter $H_0(z)$, and

$$T_{kn} = 2\cos\left(\frac{\pi}{M}(k+0.5)(n-\frac{N}{2})+(-1)^k\frac{\pi}{4}\right)$$

Figure 4 is also replaced by figure 6. The implementation cost for this system is also the prototype filter plus the modulation. The modulation could be implemented by the fast discrete cosine transform (more to be elaborated).

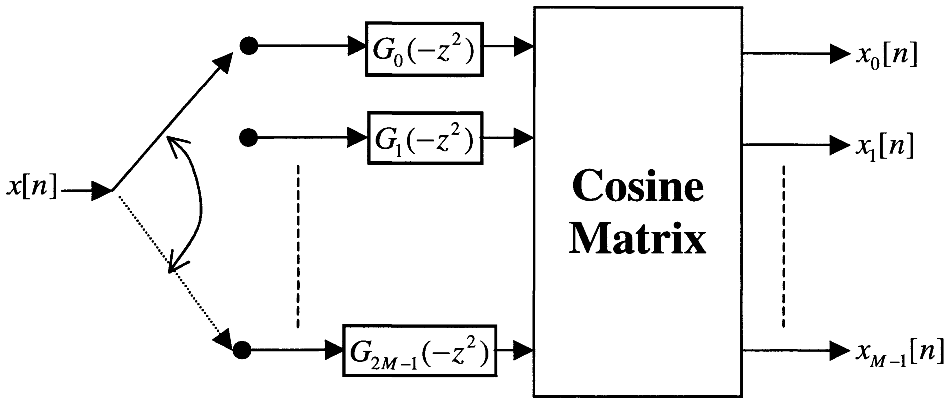

Figure 6: Cosine modulated filter bank

Other ideas:

- If the power complementary property is required, i.e.,
$$\sum_{k=0}^{M}\left|H_k(e^{j\Omega})\right|^2 = 1$$
then an additional constraint called paraunitariness can be imposed on the filter bank. It turns out that such filter bank also satisfies the perfect reconstruction property. I also understand that it is possible to design paraunitary cosine modulated filter banks.
- If we only want to do high-resolution spectral analysis on a small frequency range, there is a technique called "zoomed FFT". If we only need K frequency points for a length N record, this technique requires $N\log K$ multiplications rather than $N\log N$ for the regular FFT. Furthermore, this technique does several K-point FFT's rather than one long N-point FFT. I suppose this is easier in terms of managing the data and memory.
- From a pure mathematical standpoint, the most efficient FIR filter bank implementation would be in the frequency domain using overlap-add or overlap-save. Assume the filter length is N and we process N input samples at a time. Then the multiplication counts are roughly $2N\log(2N)$ for the forward FFT, $2MN\log(2N)$ for inverse FFT altogether, plus $2MN$ multiplications in the frequency domain. This works out to be $2M(\log(2N)+1)+\log(2N)$ multiplications per input sample, which could be less than N.

Reference:

1. P.P. Vaidyanathan, “Multirate Systems and Filter Banks”, Prentice Hall, 1993