

Digital Signal Processing

- Finite Impulse Response (FIR) filters
- Sampled analog filter response
- Design from frequency domain using IDFT
- Design using Fourier design method
- Other design methods
- Windowing

Digital Filtering

- The act of filtering **IS** the convolution of the input signal with the impulse response of the filter. Thus, we can multiply the DFT of the input signal by the filter values at each frequency, then take the inverse DFT (this is known as “Fast Convolution” if the FFT is used), or we can explicitly convolve the impulse response values with input signal values.
- The former case involves taking blocks of the signal, windowing, zero-padding, etc, but may be faster for large blocks.
- The latter case is simplest, and may more easily operate on a “live” stream of input samples. (Note that because convolution requires a reversal of one of the signals we are limited in the number of impulse response values that are practicable to store or compute).

Digital Convolution

- Recall convolution requires reversing one of the signals, shifting it, and calculating the overlapping area of the two signals, JUST to get one value of the output....

$$g(i) = \sum_{k=0}^{N-1} f(k)h(i-k) \quad \forall i = -\infty.. \infty$$

- or in the case of circular convolution...

$$g(i) = \sum_{k=0}^{N-1} f(k)h([i-k] \text{ MOD } N) \quad \forall i = -\infty.. \infty$$

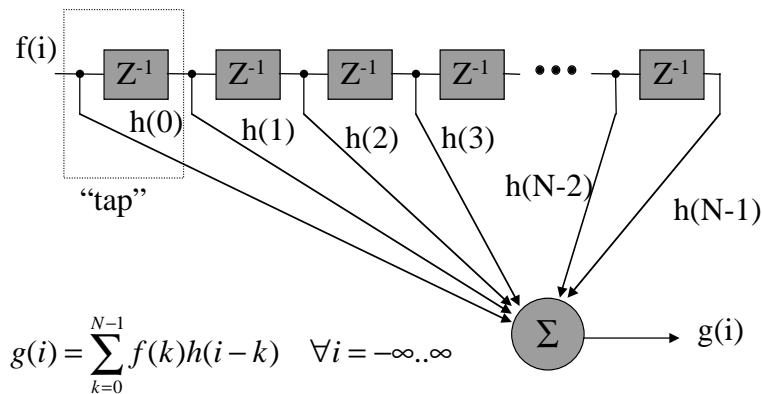
- Strictly, we only start i at zero, otherwise we would have an output before we placed any input. This requirement is known as “causality”.

Finite Impulse Response Filters

- A Digital Signal Processor is designed to implement the convolution operation with speed and ease. It is specialised to perform the multiplication and addition (MAC) required of each step of the convolution in single instructions.
- The limit on the convolution of N samples is why a digital filter implemented in this way is known as a “Finite Impulse Response” (FIR) filter.
- Each output sample of our FIR digital filter requires N MAC operations. Therefore, if the sampling rate is given, and the time taken to perform a MAC is known, we can determine the maximum impulse response length, N .
- (Another limit on N is the storage space of the DSP required to store the samples of the impulse response).

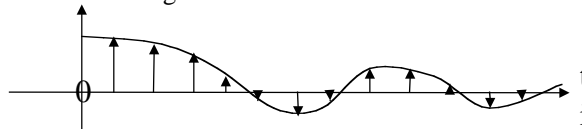
Block Diagram

- A graphical way to visualise the FIR filter operation is with a transversal filter diagram....



Impulse Response Values

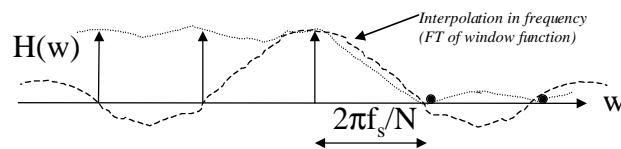
- The implementation of an FIR filter could not be simpler. So how do we determine the impulse response values, $h(i)$?
- 1/ We can sample the (real or mathematically generated) impulse response of an analog filter in the time domain...



- 2/ We can decide on the discrete frequency components of the filter, and employ the DFT to get the N tap weights... (sample the frequency domain)
- 3/ Or we can use a mathematical process to hand calculate or, if we are sensible, machine calculate, the $h(i)$... (Machine calculation by formula lends itself to scaling of the or number of taps at run-time!)

Considerations...

- Note that unlike the DFT relationship requiring samples in both time and frequency domain, the frequency domain specifying an FIR filter is continuous!
- The number of taps, N , chosen is usually a tradeoff between sharpness of frequency response, or of computation speed/storage in the DSP...
- By limiting the impulse response to N taps we must be aware of spectral leakage that may occur, and indeed the frequency resolution possible...



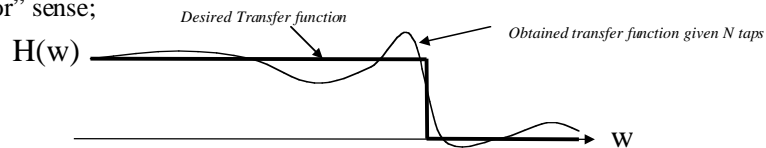
- The rate of cutoff of a filter (frequency spacing between the samples of the DFT), is specified by N , and the type of interpolation the frequency domain (the type of window applied in the sampled time domain)...

Causality and Stability...

- A “physically realisable” filter cannot produce an output before an input is applied! However, when designing an FIR filter we have the luxury that the filter is a “linear phase” filter. This means that moving the impulse response in time simply alters the phase of the transfer function in a way that introduces delay of the output, but no distortion. We therefore ignore the positioning of the impulse response values when designing...
- WE CANNOT REALISE THIS FILTER WITHOUT IMPLICITLY MAKING IT CAUSAL!
- We have no feedback system in our implementation of the filter, so the filter is stable for a stable input (provided the word length in the DSP is sufficient to handle the output swings).

Fourier Design method

- By truncating the impulse response $h(i)$ to N taps we only achieve an approximation to the desired frequency response of our filter. The Fourier or Frequency Design method for calculating $h(i)$, provides a the closest match to the desired transfer function, in a “mean square error” sense;



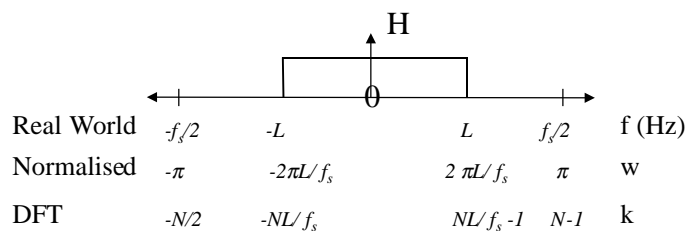
- Other methods for determining $h(i)$ adopt other error metrics, e.g. a least squares fit with more emphasis on reducing the error in the pass band, than in the stop band etc.
- An error metric in the design based on “mean absolute error” produces an “equi-ripple” transfer function.

Fourier Design Method contd.

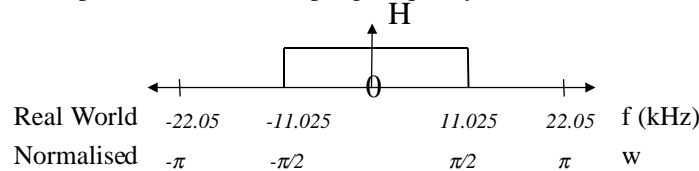
- The Fourier Design method provides a formula for calculation of $h(i)$

$$h(i) = \frac{1}{2\pi} \int_{-\pi}^{\pi} D(\omega) e^{j\omega i} d\omega$$

- Notice the desired frequency response $D(w)$ runs from normalised angular frequency coordinates, $-\pi$ to π . These extremities correspond to the folding frequency (half the sampling frequency, f_s)



- EXAMPLE: Consider a low pass filter that cuts off at 11.025 kHz and is implemented with a sampling frequency of 44.1 kHz.



$$h(i) = \frac{1}{2\pi} \int_{-\pi/2}^{\pi/2} 1e^{j\omega i} d\omega + zero$$

$$= \frac{1}{\pi i} \sin\left(\frac{\pi}{2} i\right)$$

- Note something special happens when i is zero! The result is indeterminate, so we must apply L'Hopital's rule to get $h(0)=1/\pi$
- We now have a formula for $h(i)$, from which, upon deciding on the number of taps, N , we can calculate the tap weights.
- Look at the sinc function that this formula describes; it is symmetrical about zero, with most of its energy in the main lobe; we should use $i=-N/2$ to $N/2-1$ to make use of this, (but shift it to make it causal!)

Final Touches

- The rippling that the mean square error approach of Fourier Design introduces when the number of taps is decided upon, may be unsatisfactory for the application you had in mind. The alternative to increasing the number taps (which may not be possible), is to "fudge" the $h(i)$ values, by weighting these by a window $w(i)$.
- This is entirely equivalent to the windowing we required to reduce spectral leakage previously. We are changing the frequency domain interpolation function from a sinc (corresponding to the rectangular window truncation) to something more suitable.
- Unfortunately windowing is a trade-off. By reducing the ripples, you suffer an increase in the width of the transition region. Which is best? That is up to you, the designer, and the application you intend your filter for.

Least Squares Fitting

- It is possible to assign a weighting to the fit at particular frequencies of the desired filter, and use a least squares algorithm to get the best fit. For example the ripple in the stop band might be acceptably larger than that in the pass-band, etc. This is still minimising the MSE of the fit. (see `firls` routine in Matlab).
- Alternatively we can use another metric for the fit to the desired transfer function. The equiripple filter arises when we use the MAE (Mean absolute error) to minimise. An algorithm that generates such a filter is the Remez Exchange or Parks-McClellan algorithms. (See `remez` in Matlab)

Remez Exchange Algorithm

- For all specified frequencies, k , with weighted vector $V(k)$ and desired response, D , with actual response, H .
- Choosing the order of the filter (e.g., Kaiser):

$$N = \frac{-20 \log_{10} \sqrt{\delta_p \delta_s} - 13}{2.32 |\omega_p - \omega_s|}$$
- Solve the $K+2$ linear equations, for $H(k)$ and $\varepsilon = \max |E(k)|$:

$$E(k) = V(k)[D(k) - H(k)] = (-1)^k \varepsilon$$

- or equivalently,

$$\sum_{i=0}^K g(i) \cos(ki) + \frac{(-1)^k \varepsilon}{V(k)} = D(k) \quad \forall k = 0..K+1$$
- Look for the largest $K+2$ extreme errors of $|E(k)|$, which are now a new set of K frequencies to work with.
- Finish if error is small enough, or repeat at step 2 above, or increase N and start again.
- (Actually the $h(i)$ are the one-sided impulse response, since we might assume conjugate symmetric $D(k)$ to start with)