Digital Filtering via DFT/IDFT

Prof. L. Gelman

Digital Filtering via DFT/IDFT

- **Suppose that we have a finite-duration sequence $x(n)$ of length L which excites an FIR filter of length M**

- **The output sequence $y(n)$ can be expressed in the time domain as the convolution of $x(n)$ and $h(n)$, where $h(n)$ is the unit impulse response of the filter**

- **The duration of $y(n)$ is $L+M-1$**
- **According to the convolution property of the Fourier transform, the frequency domain equivalent is**

$$Y(\omega) = X(\omega)H(\omega)$$

Use of the DFT in Linear Filtering

where $H(\omega)$ is the Fourier transform of the impulse response of the filter

- **Obviously, a DFT of size $N \geq L + M - 1$ is required to represent the output $y(n)$ in the frequency domain**

- **Thus we employ**

$$Y(k) = Y(\omega)\Big|_{\omega = 2\pi k / N} = X(\omega)H(\omega)\Big|_{\omega = 2\pi k / N} \qquad k = 0, 1, \ldots, N-1$$

Use of the DFT in Linear Filtering

- **Then we employ**

$$Y(k) = X(k)H(k) \qquad k = 0, 1, \ldots, N-1$$

where $X(k)$ **and** $H(k)$ **are the** N **point DFTs of the corresponding sequences**

- **Since the sequences** $x(n)$ **and** $h(n)$ **have a duration less than** N **we simply pad these sequences with zeros to increase their length to** N

- **This increase does not alter their Fourier transforms** $X(\omega)$ **and** $H(\omega)$

Use of the DFT in Linear Filtering

- **Since the N point DFT of the output sequence is sufficient to represent it in the frequency domain, it follows that the multiplication of the N point DFTs and followed by the computation of the N point inverse DFT, must yield the output sequence**

- **Thus with zero padding, the DFT can be used to perform linear filtering**

Computational Complexity: DFT in Linear Filtering

- **Let us assess the computational complexity of the FFT for fast convolution, e.g. when convolution is implemented by taking the inverse DFT of the product of the DFTs of two time series**

- **This method only yields sensible results when the sampling period and length of both time series are identical**

- **Zero-padding can be used to equalise the time series lengths if necessary**

Computational Complexity

- **Each FFT requires** $(N/2)\log_2 N$ **complex multiplications and** $N\log_2 N$ **additions (if *N* is a power of 2)**

- **Since the FFT is performed three times, twice for the DFT and once for the IDFT, the computational burden is** $(3N/2)\log_2 N$ **complex multiplications and** $3N\log_2 N$ **additions**

- **There are also** N **complex multiplications and** $N-1$ **complex additions required to compute** $Y(k)$

Computational Complexity

- **It is interesting to compare the efficiency of the FFT algorithm with the direct convolution, which requires the**

  N^2 **complex multiplications**

- **Let us focus on the gain G in the number of multiplications**

$$G = \frac{N}{(3/2)\log_2 N + 1}$$

- **For example, even for signal lengths as low as 1024 samples, this gain is 64**