EE3015 & EE3025 Presentation

Koidala SURYA Prakash

[1]ee18btech11026@iith.ac.in

October 3, 2021

Assignment 1

Problem 6.4

$$\text{Given } x(n) = \{\underset{\uparrow}{1}, 2, 3, 4, 2, 1\} \tag{1}$$

$$y(n) + \frac{1}{2}y(n-1) = x(n) + x(n-2) \tag{2}$$

$$\implies h(n) = \left(\frac{-1}{2}\right)^{n} u(n) + \left(\frac{-1}{2}\right)^{n-2} u(n-2) \tag{3}$$

Question

Compute $X(k)$, $H(k)$ and $y(n)$ using FFT and IFFT methods.

Solution

Computing y (for N samples) using FFT and IFFT :

$$X = FFT(x) \tag{4}$$

$$H = FFT(h) \tag{5}$$

$$Y = X.H \tag{6}$$

$$y = IFFT(Y) = \frac{1}{N} * FFT(Y^*)\{\texttt{only if y is real}\} \tag{7}$$

where Y^* = complex conjugate of Y

Recursive N-point FFT Algorithm

An N-point DFT can be written as :

$$
\begin{aligned}
X_k &= \sum_{n=0}^{N-1} x(n) e^{-j2\pi kn/N}, \quad k = 0, 1, \ldots, N-1 \\
&= \underbrace{\sum_{m=0}^{N/2-1} x_{2m} e^{-\frac{j2\pi km}{N/2}}}_{\text{N/2 DFT with even inputs}} + W_N^k \underbrace{\sum_{m=0}^{N/2-1} x_{2m+1} e^{-\frac{j2\pi k(m)}{N/2}}}_{\text{N/2 DFT with odd inputs}}
\end{aligned}
$$

where $W_N = e^{\frac{-j2\pi}{N}}$

Recursive N-point FFT Algorithm

While exploiting symmetry of W_N as :

$$W_N^{k+N/2} = -W_N^k \tag{8}$$

We transformed the iterative DFT problem to a Divide-Conquer algorithm as :

$$X_{0\rightarrow\frac{N}{2}-1} = X_{even} + \overline{W}_{N/2} * X_{odd} \tag{9}$$

$$X_{\frac{N}{2}\rightarrow N-1} = X_{even} - \overline{W}_{N/2} * X_{odd} \tag{10}$$

$$\overline{W}_{N/2}(i) = W_N^i$$

; for i = 0,1,2 ... (N/2) -1

Matrix representation of the FFT algorithm

An 8-point DFT can be represented as a Matrix product as follows:

$$\overline{X} = \overline{W}\,\overline{x}$$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \\ X(4) \\ X(5) \\ X(6) \\ X(7) \end{bmatrix} = \begin{bmatrix} W^0 & W^0 & W^0 & W^0 & W^0 & W^0 & W^0 & W^0 \\ W^0 & W^1 & W^2 & W^3 & W^4 & W^5 & W^6 & W^7 \\ W^0 & W^2 & W^4 & W^6 & W^0 & W^2 & W^4 & W^6 \\ W^0 & W^3 & W^6 & W^1 & W^4 & W^7 & W^2 & W^5 \\ W^0 & W^4 & W^0 & W^4 & W^0 & W^4 & W^0 & W^4 \\ W^0 & W^5 & W^2 & W^7 & W^4 & W^1 & W^6 & W^3 \\ W^0 & W^6 & W^4 & W^2 & W^0 & W^6 & W^4 & W^2 \\ W^0 & W^7 & W^6 & W^5 & W^4 & W^3 & W^2 & W^1 \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \\ x(4) \\ x(5) \\ x(6) \\ x(7) \end{bmatrix} \tag{11}$$

The FFT algorithm decompose $\overline{W}$ into sparse matrices by permuting $\overline{x}$ in a bit-reversed fashion resulting in $\overline{x_p}$

$$\overline{X} = \overline{W3}\ \overline{W2}\ \overline{W1}\ \overline{x_p} \tag{12}$$

$$\begin{bmatrix} X(0) \\ X(1) \\ X(2) \\ X(3) \\ X(4) \\ X(5) \\ X(6) \\ X(7) \end{bmatrix} = \begin{bmatrix} 1 & . & . & . & W^0 & . & . & . \\ . & 1 & . & . & . & W^1 & . & . \\ . & . & 1 & . & . & . & W^1 & . \\ . & . & . & 1 & . & . & . & W^1 \\ 1 & . & . & . & -W^0 & . & . & . \\ . & 1 & . & . & . & -W^1 & . & . \\ . & . & 1 & . & . & . & -W^1 & . \\ . & . & . & 1 & . & . & . & -W^1 \end{bmatrix} \begin{bmatrix} 1 & . & W^0 & . & . & . & . & . \\ . & 1 & . & W^2 & . & . & . & . \\ 1 & . & -W^0 & . & . & . & . & . \\ . & 1 & . & -W^2 & . & . & . & . \\ . & . & . & . & 1 & . & W^0 & . \\ . & . & . & . & . & 1 & . & W^2 \\ . & . & . & . & 1 & . & -W^0 & . \\ . & . & . & . & . & 1 & . & -W^2 \end{bmatrix} \begin{bmatrix} 1 & W^0 & . & . & . & . & . & . \\ 1 & -W^0 & . & . & . & . & . & . \\ . & . & 1 & W^0 & . & . & . & . \\ . & . & 1 & -W^0 & . & . & . & . \\ . & . & . & . & 1 & W^0 & . & . \\ . & . & . & . & 1 & -W^0 & . & . \\ . & . & . & . & . & . & 1 & W^0 \\ . & . & . & . & . & . & 1 & -W^0 \end{bmatrix} \begin{bmatrix} x(0) \\ x(4) \\ x(2) \\ x(6) \\ x(1) \\ x(5) \\ x(3) \\ x(7) \end{bmatrix} \tag{13}$$

Relation with the butterfly diagram

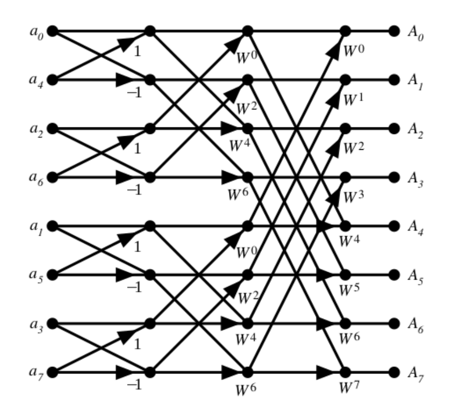


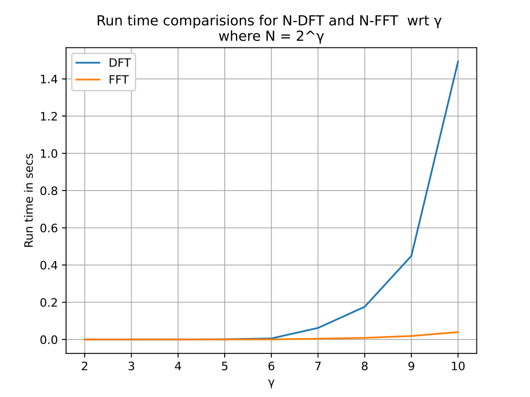
Figure: 8-point FFT butterfly diagram

Ref : Heckbert, P., 1995. Fourier transforms and the fast fourier transform (fft) algorithm. Computer Graphics, 2, pp.15-463.

Time complexity

Convolution/DFT requires N^2 operations $\approx \boldsymbol{O(N^2)}$.
While the same output can be achieved using FFT and IFFT within :

$$\underbrace{O(NlogN)}_{\substack{x \to X \\ h \to H}} + \underbrace{O(N)}_{Y = X * H} + \underbrace{O(NlogN)}_{Y \to y} \approx \boldsymbol{O(NlogN)} \tag{14}$$



C vs Python Implementaion

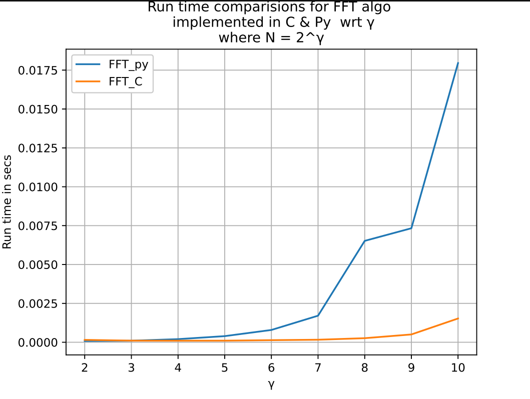


Figure: Comparing C vs Py implementation

Assignment 02

Design equivalent FIR realizations for filter number 114 with the following specifications.

Pass band specifications

$$F_s = 48kHz$$

$$F_{p1} = 7.2kHz \; ; F_{p2} = 6kHz$$

$$\omega_{p1} = 0.3\pi \; ; \omega_{p2} = 0.25\pi$$

$$\omega_c = 0.275\pi$$

$$\omega_l = \frac{\omega_{p1} - \omega_{p2}}{2} = 0.025\pi$$

Stop band specifications

$$\texttt{Tolerance:} \delta_1 = \delta_2 = \delta = 0.15$$

$$\Delta F = 0.3kHz \; ; \Delta\omega = 0.0125\pi$$

$$F_{s1} = 7.5Hz \; ; F_{s2} = 5.7kHz$$

$$\omega_{s1} = 0.3125\pi \; ; \omega_{s2} = 0.2375\pi$$

Designing a low pass equivalent

$$h_l(n) = \frac{\sin(n\omega_l)}{n\pi} w(n) \tag{15}$$

The Kaiser window is defined and computed as :

$$\begin{aligned} w(n) &= \frac{I_0\left[\beta N\sqrt{1-\left(\frac{n}{N}\right)^2}\right]}{I_0(\beta N)}, \; -N \le n \le N, \beta > 0 \\ &= 0 \qquad\qquad\qquad\qquad \text{otherwise,} \end{aligned}$$

Parameters of Kaiser window are :

$$A = -20\log_{10}\delta = 16.478(<21) \implies \beta N = 1$$

$$N \ge \frac{A-8}{4.57\Delta\omega} = 47.2 \implies \mathit{chosen}\ N = 100$$

$$w(n) = 1, \; -100 \leq n \leq 100$$
$$= 0 \qquad \text{otherwise}$$

`Desired lowpass filter impulse response is`

$$h_{lp}(n) = \frac{\sin(\frac{n\pi}{40})}{n\pi} \; ; -100 \leq n \leq 100$$
$$= 0, \qquad \text{otherwise}$$

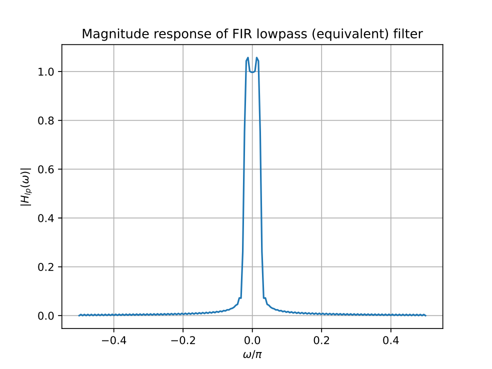


Figure: Magnitude response of the FIR lowpass digital filter

Converting into a causal FIR Bandpass Filter

A band pass filter can be obtained from a low pass equivalent through the following transformation ...

$$H_{bp}(j\omega) = H_{lp}(j(\omega - \omega_c)) + H_{lp}(j(\omega + \omega_c)) \tag{16}$$

$$\implies h_{bp}(n) = h_{lp}(n) * 2cos(n\omega_c) \tag{17}$$

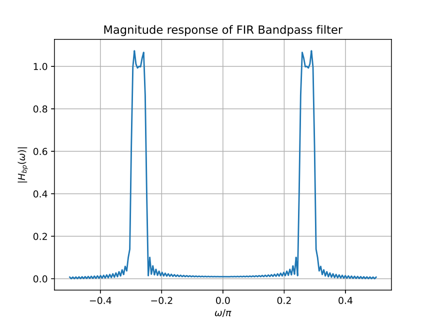


Figure: Magnitude response of the FIR bandpass digital filter

References

Heckbert, P., 1995. Fourier transforms and the fast fourier transform (fft) algorithm. Computer Graphics, 2, pp.15-463

Code & Reports

- Assignment-01 Link
- Assignment-02 Link
- Beamer template Link

Thank You !!