

Homework Assignment #6: Chap. 8-9

Due: June 4, 2020

I Program Assignment (100%)

1. (10%) Consider again the inverse DFT given in (8.2).

$$X[n] = \frac{1}{N} \sum_{k=0}^{N-1} X[k] W_N^{-kn}, \quad n = 0, 1, \dots, N-1 \quad (8.2)$$

- (a) (5%) Replace k by $\langle -k \rangle_N$ in (8.2) and show that the resulting summation is a DFT expression, that is, $\text{IDFT}\{X[k]\} = \frac{1}{N} \text{DFT}\{X[\langle -k \rangle_N]\}$.
- (b) (5%) Develop a MATLAB function `x = IDFT(X,N)` using the `fft` function that uses the above approach. Verify your function on signal $x[n] = \{1, 2, 3, 4, 5, 6, 7, 8\}$.
2. (15%) In this problem we will investigate differences in the speeds of DFT and FFT algorithms when stored twiddle factors are used.

$$\begin{bmatrix} X[0] \\ X[1] \\ X[2] \\ X[3] \\ X[4] \\ X[5] \\ X[6] \\ X[7] \end{bmatrix} = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & W_8 & W_8^2 & W_8^3 & W_8^4 & W_8^5 & W_8^6 & W_8^7 \\ 1 & W_8^2 & W_8^4 & W_8^6 & 1 & W_8^2 & W_8^4 & W_8^6 \\ 1 & W_8^3 & W_8^6 & W_8 & W_8^4 & W_8^7 & W_8^2 & W_8^5 \\ 1 & W_8^4 & 1 & W_8^4 & 1 & W_8^4 & 1 & W_8^4 \\ 1 & W_8^5 & W_8^2 & W_8^7 & W_8^4 & W_8 & W_8^6 & W_8^3 \\ 1 & W_8^6 & W_8^4 & W_8^2 & 1 & W_8^6 & W_8^4 & W_8^2 \\ 1 & W_8^7 & W_8^6 & W_8^5 & W_8^4 & W_8^3 & W_8^2 & W_8 \end{bmatrix} \begin{bmatrix} x[0] \\ x[1] \\ x[2] \\ x[3] \\ x[4] \\ x[5] \\ x[6] \\ x[7] \end{bmatrix} \quad (8.8)$$

- (a) (5%) Write a function `W = dft_matrix(N)` that computes the DFT matrix W_N given in (8.8).
- (b) (5%) Write a function `X = dftdirect_m(x,W)` that modifies the `dftdirect` function using the matrix `W` from (a). Using the `tic` and `toc` functions compare computation times for the `dftdirect` and `dftdirect_m` function for $N = 128, 256, 512, \text{ and } 1024$. For this purpose generate an N -point complex-valued signal as `x = randn(1,N) + 1j*randn(1,N)`. (verify your code with `fft` first)
- (c) (5%) Write a function `X = fftrecur_m(x,W)` that modifies the `fftrecur` function given on page 439 using the matrix `W` from (a). Using the `tic` and `toc` functions compare computation times for the `fftrecur` and `fftrecur_m` function for $N = 128, 256, 512, \text{ and } 1024$. For this purpose generate an N -point complex valued signal as `x = randn(1,N) +`

$1j*\text{randn}(1,N)$.

(verify your code with `fft` first)

3. (15%) Consider the flow graph in **Figure 8.10** which implements a DIT-FFT algorithm with both input and output in natural order. Let the nodes at each stage be labeled as $s_m[k]$, $0 \leq m \leq 3$ with $s_0[k] = x[k]$ and $s_3[k] = X[k]$, $0 \leq k \leq 7$.
 - (a) (5%) Express $s_m[k]$ in terms of $s_{m-1}[k]$ for $m = 1, 2, 3$.
 - (b) (5%) Write a MATLAB function `X = fftalt8(x)` that computes an 8-point DFT using the equations in part (a). Verify with sequence $x[n] = \{0,1,2,2,3,3,3,4\}$.
 - (c) (5%) Compare the coding complexity of the above function with that of MATLAB function `fftditr2` shown in **Figure 8.6**, and comment on its usefulness.

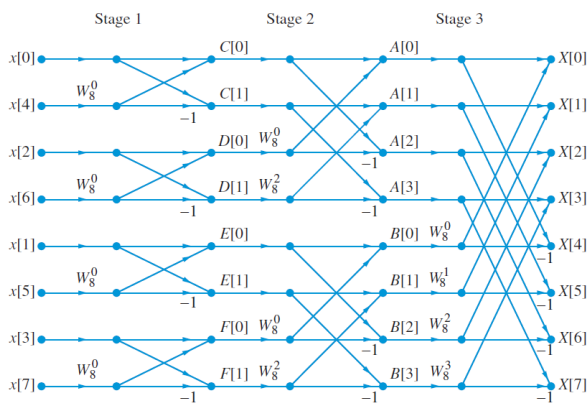


Figure 8.6 Flow graph of 8-point decimation-in-time FFT algorithm using the butterfly computation shown in Figure 8.4. The trivial twiddle factor $W_8^0 = 1$ is shown for the sake of generality.

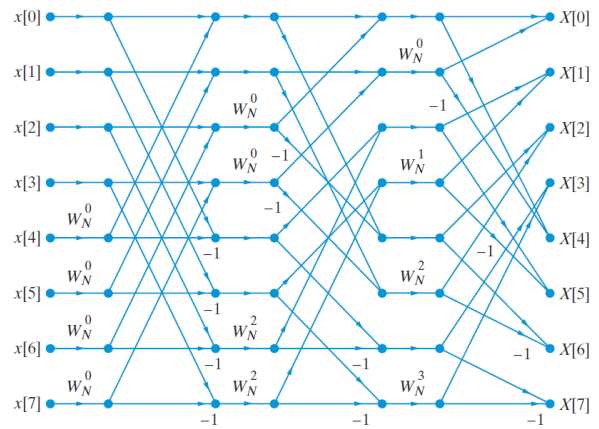


Figure 8.10 Decimation-in-time FFT algorithm with both input and output in natural order.

4. (10%) Using the flow graph of Figure 8.13 and following the approach used in developing the `fftditr2` function.
 - (a) (5%) Develop a radix-2 DIF-FFT function `X = fftdifr2(x)` for power-of-2 length N .
 - (b) (5%) Verify your function for $N = 2^v$, where $2 \leq v \leq 10$. For this purpose generate an N -point complex-valued signal as $x = \text{randn}(1,N) + 1j*\text{randn}(1,N)$.

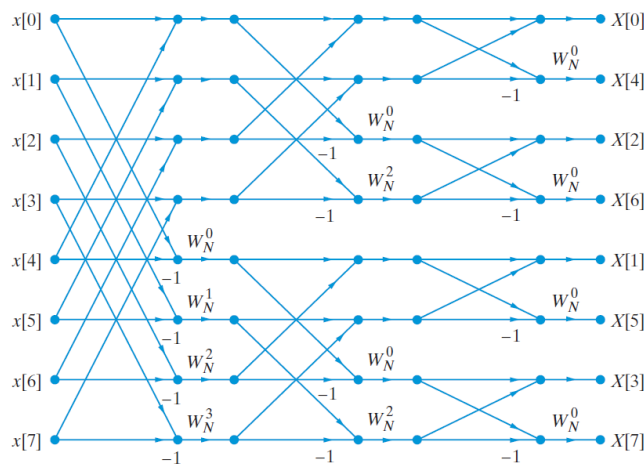


Figure 8.13 Flow graph for the decimation-in-frequency 8-point FFT algorithm. The input sequence is in natural order and the output sequence in bit-reversed order.

5. (10%)The `filterfirfd` implements the FIR direct form structure.
- (a) (5%)Develop a new MATLAB function `y=filterfir1p(h,x)` that implements the FIR linear-phase form given its impulse response in `h`. This function should first check if `h` is one of type-I through type-IV and then simulate the corresponding equations. If `h` does not correspond to one of the four types then the function should display an appropriate error message.

- (b) (5%)Verify your function on each of the following FIR systems:

$$h1[n] = \{1,2,3,2,1\},$$

↑

$$h2[n] = \{1,-2,3,3,-2,1\},$$

↑

$$h3[n] = \{1,-5,0,5,-1\},$$

↑

$$h4[n] = \{1,-3,-4,4,3,-1\},$$

↑

$$h5[n] = \{1,2,3,-2,-1\},$$

↑

For verification determine the first ten samples of the step responses using your function and compare them with those from the `filter` function.

6. (10%)Consider the IIR normal direct form II structure given in **Figure 9.6** and implemented by (9.18) and (9.20).

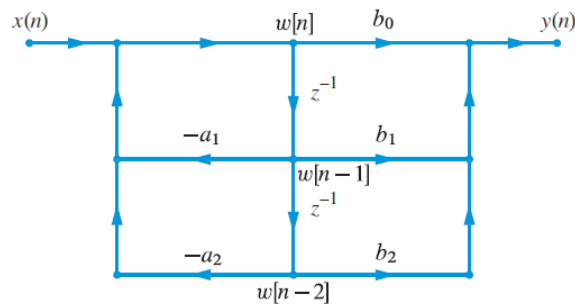


Figure 9.6 Direct form II structure for implementation of an N th order system. For convenience, we assume that $N = M = 2$. If $N \neq M$, some of the coefficients will be zero.

$$y[n] = \sum_{k=0}^M b_k w[n - k]. \quad (9.20)$$

$$w[n] = - \sum_{k=1}^N a_k w[n - k] + x[n]. \quad (9.18)$$

- (a) (5%)Using the MATLAB function `filterdf1` as a guide, develop a MATLAB function `y=filterdf2(b,a,x)` that implements the normal direct form II structure. Assume zero initial conditions.

- (b) (5%) Determine $y[n]$, $0 \leq n \leq 500$ using your function and `filterdf1` function with following inputs:

$$x[n] = \left(\frac{1}{4}\right)^n u[n], a = \left[1 \quad -\frac{3}{2} \quad \frac{1}{2}\right], b = 1$$

Compare your results to verify that your `filterdf2` function is correctly implemented.

7. (20%) The following numerator and denominator arrays in MATLAB represent the system function of a discrete-time system in direct form:

$$b = [1, -2.61, 2.75, -1.36, 0.27], a = [1, -1.05, 0.91, -0.8, 0.38].$$

Determine and draw each of the following structures:

- (5%) Cascade form with second-order sections in normal direct form I,
 - (5%) Cascade form with second-order sections in transposed direct form I,
 - (5%) Cascade form with second-order sections in normal direct form II,
 - (5%) Cascade form with second-order sections in transposed direct form II.
8. (10%) The frequency-sampling form is developed using (9.50) which uses complex arithmetic.

$$H(z) = \frac{1 - z^{-N}}{N} \sum_{k=0}^{N-1} \frac{H[k]}{1 - z^{-1} e^{j\frac{2\pi k}{N}}}, \quad H[k] = H(z) \Big|_{z=e^{j\frac{2\pi k}{N}}}, \quad (9.50)$$

- (a) (5%) Develop a MATLAB function `[G,sos]=firfd2fs(h)` that determines frequency sampling form parameters given in (9.51) and (9.52) given the impulse response in `h`. The matrix `sos` should contain second-order section coefficients in the form similar to the `tf2sos` function while `G` array should contain the respective gains of second-order sections. Incorporate the coefficients for the $H[0]$ and $H[N/2]$ terms in `sos` and `G` arrays.

$$H(z) = \frac{1 - z^{-N}}{N} \left\{ \frac{H[0]}{1 - z^{-1}} + \frac{H[\frac{N}{2}]}{1 + z^{-1}} + \sum_{k=1}^K 2|H[k]|H_k(z) \right\}, \quad (9.51)$$

$$H_k(z) = \frac{\cos(\angle H[k]) - z^{-1} \cos(\angle H[k] - \frac{2\pi k}{N})}{1 - 2 \cos(\frac{2\pi k}{N})z^{-1} + z^{-2}}. \quad (9.52)$$

- (b) (5%) Verify your function by input `h` with sampled frequency response (9.53) and compare with the system function (9.54) (see **example 9.6** in the textbook)

$$H[k] = H(e^{j\frac{2\pi}{33}k}) = e^{-j\frac{32\pi}{33}k} \times \begin{cases} 1, & k = 0, 1, 2, 31, 32 \\ 0.5, & k = 3, 30 \\ 0, & \text{otherwise} \end{cases} \quad (9.53)$$

$$H(z) = \frac{1 - z^{-33}}{33} \left[\frac{1}{1 - z^{-1}} + \frac{-1.99 + 1.99z^{-1}}{1 - 1.964z^{-1} + z^{-2}} + \frac{1.964 - 1.964z^{-1}}{1 - 1.857z^{-1} + z^{-2}} + \frac{-0.96 + 0.96z^{-1}}{1 - 1.683z^{-1} + z^{-2}} \right] \quad (9.54)$$