MAT 320 Supplementary Notes Fall 2016

Discrete Fourier Transform.

Given input signal $\mathbf{x} = (x_0, x_1, \ldots, x_N)$, we define the Fourier Transform output signal $\mathbf{X} = (X_0, X_1, \ldots, X_N)$ by:

$$X_k = \mathrm{DFT}(\mathbf{x}, N, k) = \sum_{t=0}^{N-1} x_t e^{-i\frac{2\pi}{N}kt}, \quad k = 0, \ldots, N-1.$$

Fast Fourier Transform – Recursive version.

Assume $N = 2^m$ is a power of 2. Given input signal $\mathbf{x} = (x_0, x_1, \ldots, x_N)$, we define the Fourier Transform output signal $\mathbf{X} = (X_0, X_1, \ldots, X_N)$ by:

$$X_k = \mathrm{DFT}(\mathbf{x}, N, k)$$

in two pieces recursively, with each piece based on the half-size Fourier transforms as follows:

$$X_k = \mathrm{DFT}(\mathbf{x}_{ev}, N/2, k) + W_N^k \cdot \mathrm{DFT}(\mathbf{x}_{od}, N/2, k), \qquad k = 0, \ldots, \frac{N}{2} - 1$$

and

$$X_{\frac{N}{2}+k} = \mathrm{DFT}(\mathbf{x}_{ev}, N/2, k) - W_N^k \cdot \mathrm{DFT}(\mathbf{x}_{od}, N/2, k), \qquad k = 0, \ldots, \frac{N}{2} - 1,$$

where $\mathbf{x}_{ev} = (x_0, x_2, \ldots, x_{N-2})$ is the $N/2$ length signal of even index terms from $\mathbf{x}$, and $\mathbf{x}_{od} = (x_1, x_3, \ldots, x_{N-1})$ is the $N/2$ length signal of odd index terms from $\mathbf{x}$, and

$$W_N = e^{-i\frac{2\pi}{N}}.$$

Fast Fourier Transform – Non-Recursive version with bit reversal.

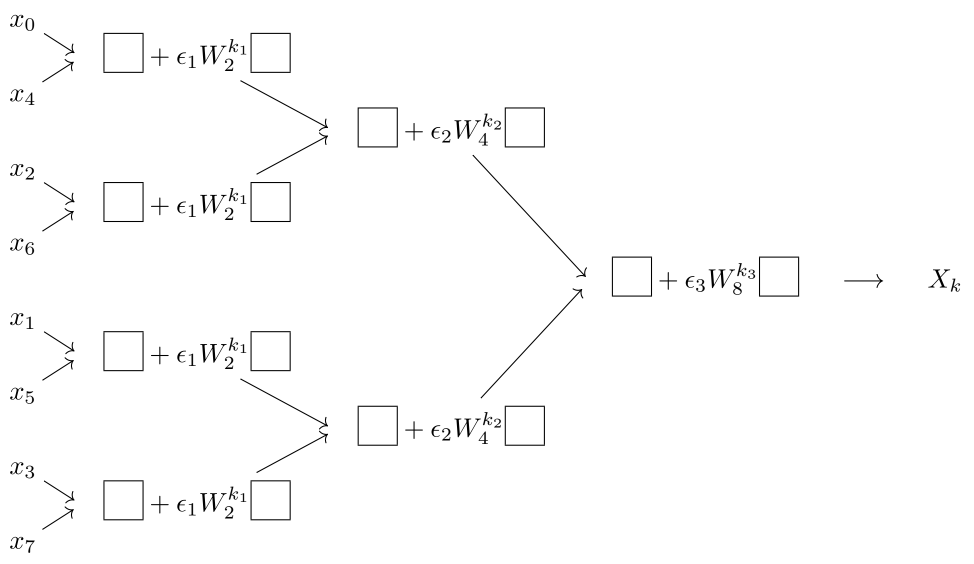
Next we implement the Cooley-Tukey Fast Fourier Transform, which is the same computation as in the recursive version but without the recursive function call. This is accomplished by first reordering the data indices according to bit reversal, then combining successive pairs using the same step as in the recursion formula above. This summation step needs to be done with correct W_M power and the correct sign, depending on whether the k value is in the first or second half of the values from 0 to $M - 1$. This is illustrated below, for $N = 8$.

The table below is for fixed k in the range $0 \le k \le 7$. Note: the values k_1, k_2, k_3 and ϵ_1, ϵ_2, and ϵ_3 depend on k and are discussed in more detail below.

The input values on the left are in bit-reversed index order. Call this column 0. In each successive column i, for $1 \leq i \leq 3$ the expression $\epsilon_i W_{2^i}^{k_i}$ connects the boxes together with the two inputs coming from the left, top to bottom feeding in order left to right into the two boxes.

So we always mean to imply 'top' becomes 'left' and 'bottom' becomes 'right', as in the diagram:

$$\begin{tikzcd}
A \arrow[dr] & \\
& \boxed{\text{A}} + \epsilon_1 W_2^{k_1} \boxed{\text{B}} \\
B \arrow[ur] &
\end{tikzcd}$$


x_0
x_4
x_2
x_6
x_1
x_5
x_3
x_7
$\square + \epsilon_1 W_2^{k_1} \square$
$\square + \epsilon_2 W_4^{k_2} \square$
$\square + \epsilon_3 W_8^{k_3} \square$
$\longrightarrow$
X_k

In the rightmost column we have the single sum, the recursive DFT for $N = 8$, which can be written:

$$\square + \epsilon_3 W_8^{k_3} \square \quad = \quad X_k \quad = \quad \text{DFT}(\mathbf{x}_{ev}, 4, k) + W_8^k \cdot \text{DFT}(\mathbf{x}_{od}, 4, k)$$

or

$$\square + \epsilon_3 W_8^{k_3} \square \quad = \quad X_{4+k} \quad = \quad \text{DFT}(\mathbf{x}_{ev}, 4, k) - W_8^k \cdot \text{DFT}(\mathbf{x}_{od}, 4, k)$$

for the appropriate $k = 0, 1, 2, 3$. The boxes are just placeholders to represent lower order DFT's. These two equations can be written as one with the appropriate k_3 and ϵ_3, which can be defined as:

$$k_3 \equiv k \pmod 4, \quad 0 \leq k_3 \leq 3, \qquad \text{and} \qquad \epsilon_3 = (-1)^{\lceil k/4 \rceil}.$$

This is simply saying that:

$$k_3 = \begin{cases} k, & 0 \leq k \leq 3 \\ k-4, & 4 \leq k \leq 7 \end{cases} \qquad \text{and} \qquad \epsilon_3 = \begin{cases} 1, & 0 \leq k \leq 3 \\ -1, & 4 \leq k \leq 7 \end{cases}$$

Continuing with the next level down in recursion, we have:

$$k_2 \equiv k_3 \pmod 2 \qquad \text{and} \qquad \epsilon_2 = (-1)^{\lceil k_3/2 \rceil},$$

and

$$k_1 = 0 \qquad \text{and} \qquad \epsilon_2 = (-1)^{\lceil k_2/2 \rceil}.$$

Computing these values for $k = 0, \ldots, 7$ we have the following table:

k	k_1	k_2	k_3	ϵ_1	ϵ_2	ϵ_3
0	0	0	0	1	1	1
1	0	1	1	-1	1	1
2	0	0	2	1	-1	1
3	0	1	3	-1	-1	1
4	0	0	0	1	1	-1
5	0	1	1	-1	1	-1
6	0	0	2	1	-1	-1
7	0	1	3	-1	-1	-1

Next, we see how the values k_i and ϵ_i can also be extracted from the bit reversal process.

Bit reversal for $N = 8$:

$0 = 000 \longleftrightarrow 000 = 0$

$1 = 001 \longleftrightarrow 100 = 4$

$2 = 010 \longleftrightarrow 010 = 2$

$3 = 011 \longleftrightarrow 110 = 6$

$4 = 100 \longleftrightarrow 001 = 1$

$5 = 101 \longleftrightarrow 101 = 5$

$6 = 110 \longleftrightarrow 011 = 3$

$7 = 111 \longleftrightarrow 111 = 7$

Notice that the values for $\epsilon_1, \epsilon_2, \epsilon_3$ can be obtained from the reversed bit strings. Simply replace the bits in the following way:

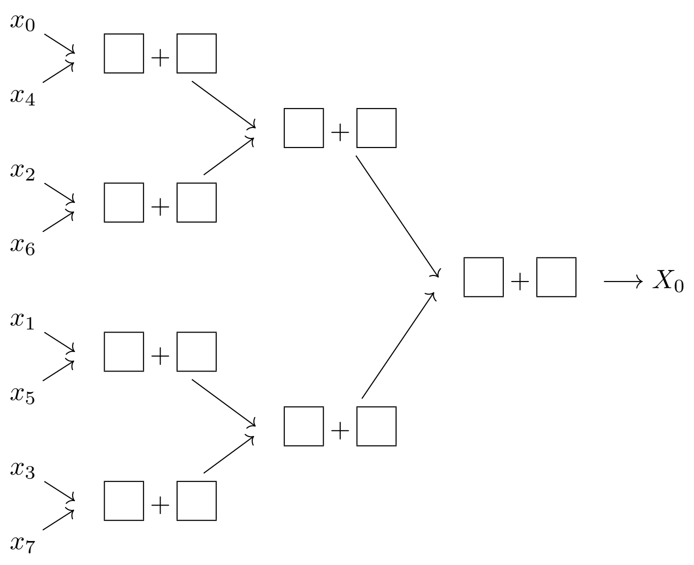
$$0 \quad \mapsto \quad 1 \qquad \text{and} \qquad 1 \quad \mapsto \quad -1$$

Also, the values for k_1, k_2, k_3 can be obtained from the reversed bit strings in the following way: Consider the three columns formed by the reversed bit strings. In each column replace any sequence of consecutive zero's or one's by a string of values starting at zero and increasing by one to fill the previous length sequence exactly. So the sequences map like:

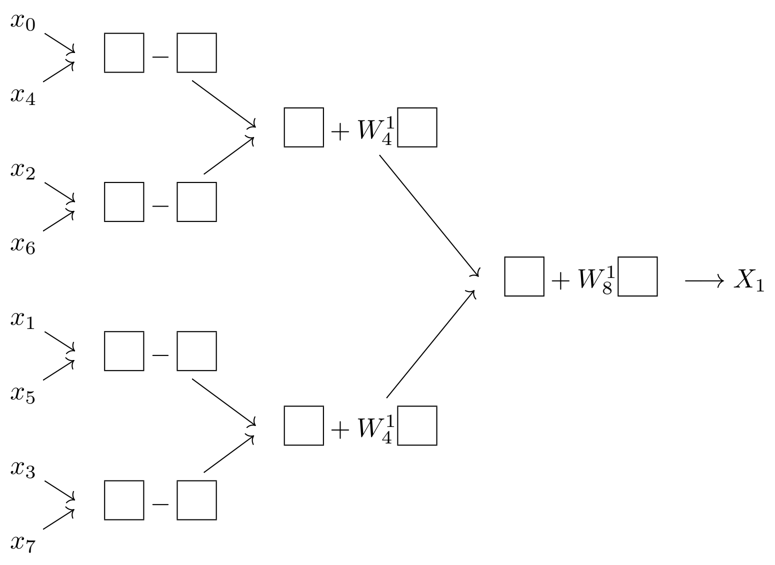
$$\begin{matrix}0\\0\\0\\0\end{matrix} \mapsto \begin{matrix}0\\1\\2\\3\end{matrix}, \quad \begin{matrix}1\\1\\1\\1\end{matrix} \mapsto \begin{matrix}0\\1\\2\\3\end{matrix}, \quad \begin{matrix}0\\0\\0\end{matrix} \mapsto \begin{matrix}0\\1\\2\end{matrix}, \quad \begin{matrix}1\\1\\1\end{matrix} \mapsto \begin{matrix}0\\1\\2\end{matrix}, \quad \begin{matrix}0\\0\end{matrix} \mapsto \begin{matrix}0\\1\end{matrix}, \quad \begin{matrix}1\\1\end{matrix} \mapsto \begin{matrix}0\\1\end{matrix}, \quad 0 \mapsto 0, \quad 1 \mapsto 0$$

Some values of the array for $N = 8$:

$k = 0$: $(k_1 = k_2 = k_3 = 0, \epsilon_1 = \epsilon_2 = \epsilon_3 = 1)$

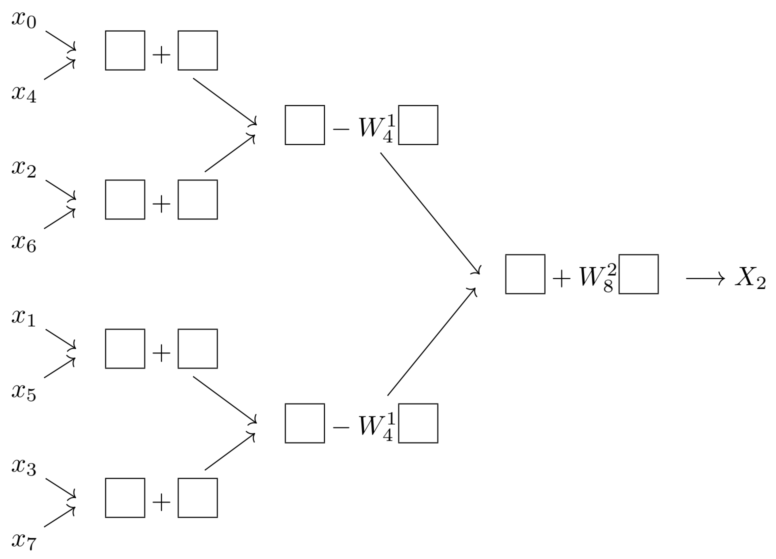


$k = 1$: $(k_1 = 0, k_2 = k_3 = 1, \epsilon_1 = -1, \epsilon_2 = \epsilon_3 = 1)$

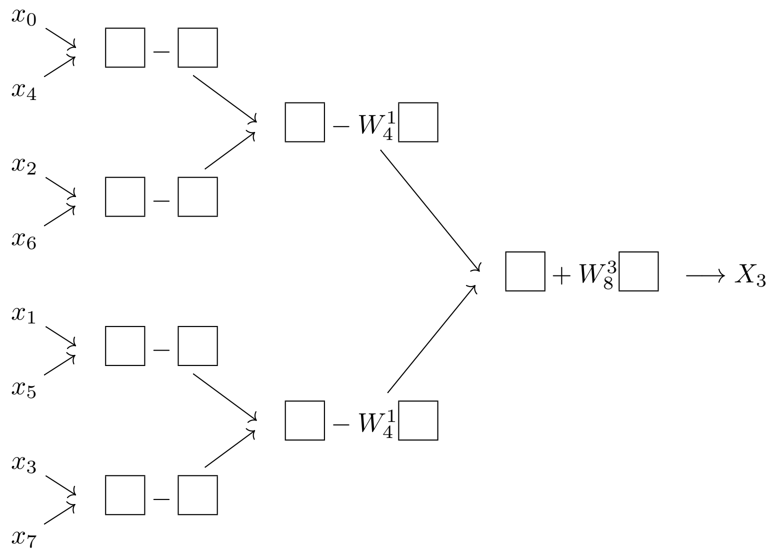


More values of the array for $N = 8$:

$k = 2$: $(k_1 = k_2 = 0, k_3 = 2, \epsilon_1 = 1, \epsilon_2 = -1, \epsilon_3 = 1)$

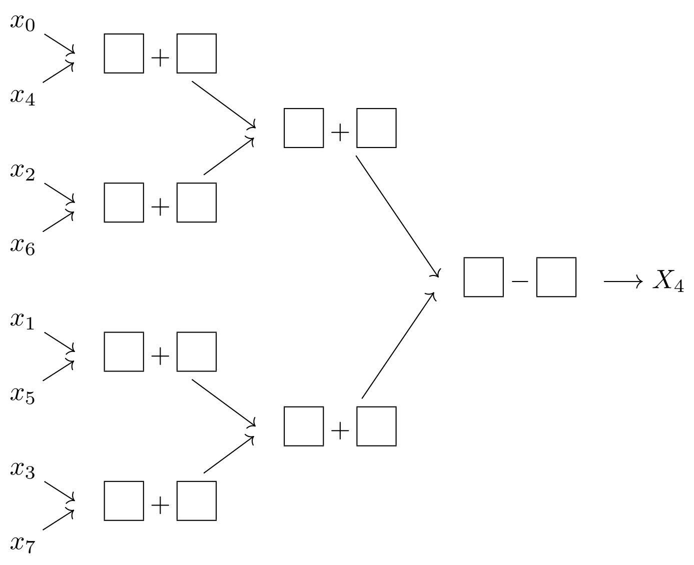


$k = 3$: $(k_1 = 0, k_2 = 1, k_3 = 3, \epsilon_1 = -1, \epsilon_1 = -1, \epsilon_2 = 1)$

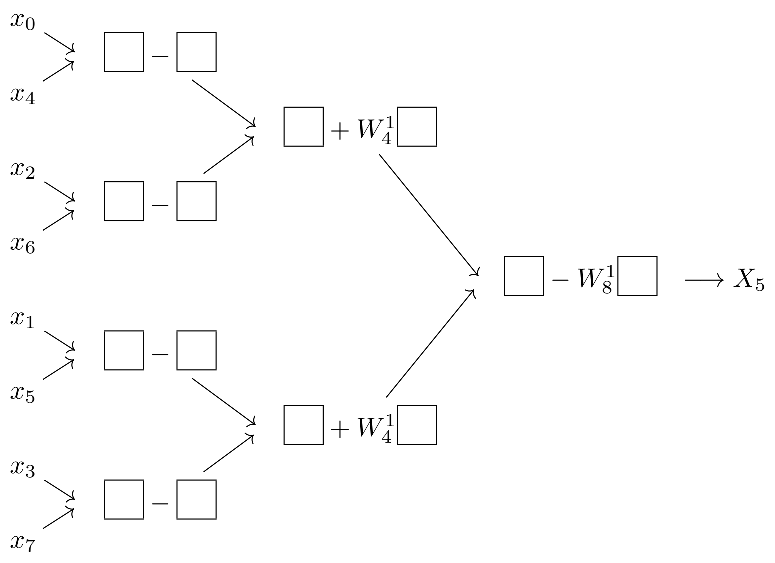


More values of the array for $N = 8$:

$k = 4$: $(k_1 = k_2 = k_3 = 0, \epsilon_1 = 1, \epsilon_2 = 1, \epsilon_3 = -1)$

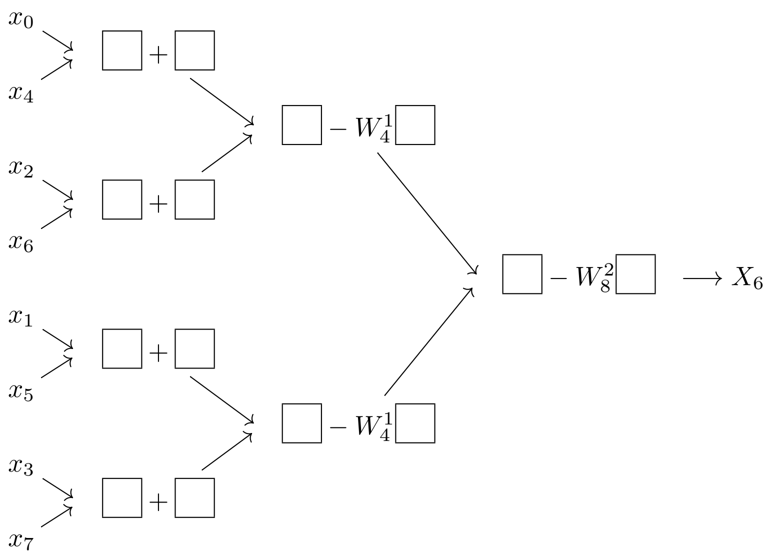


$k = 5$: $(k_1 = 0, k_2 = 1, k_3 = 1, \epsilon_1 = -1, \epsilon_1 = 1, \epsilon_2 = -1)$

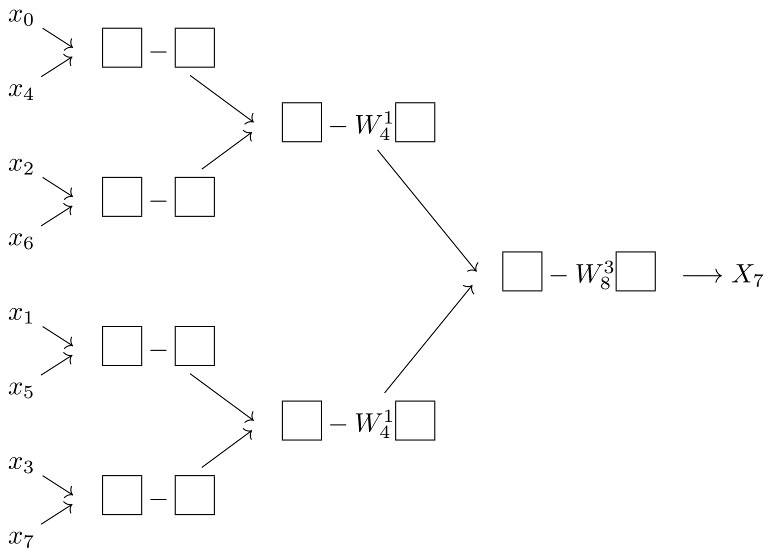


More values of the array for $N = 8$:

$k = 6$: $(k_1 = k_2 = 0, k_3 = 2, \epsilon_1 = 1, \epsilon_2 = -1, \epsilon_3 = -1)$

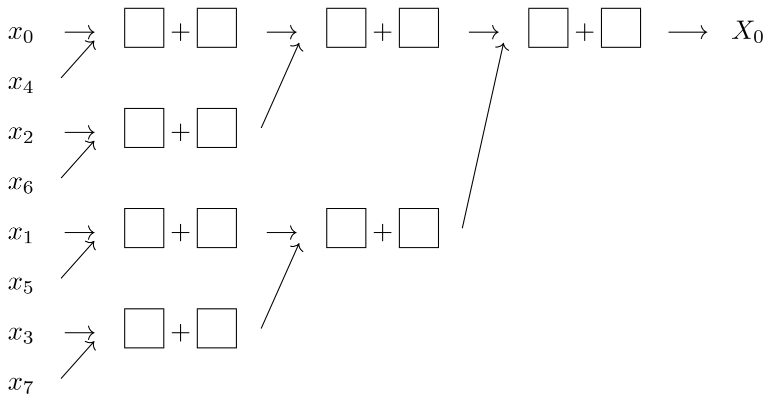


$k = 7$: $(k_1 = 0, k_2 = 1, k_3 = 3, \epsilon_1 = -1, \epsilon_1 = -1, \epsilon_2 = -1)$

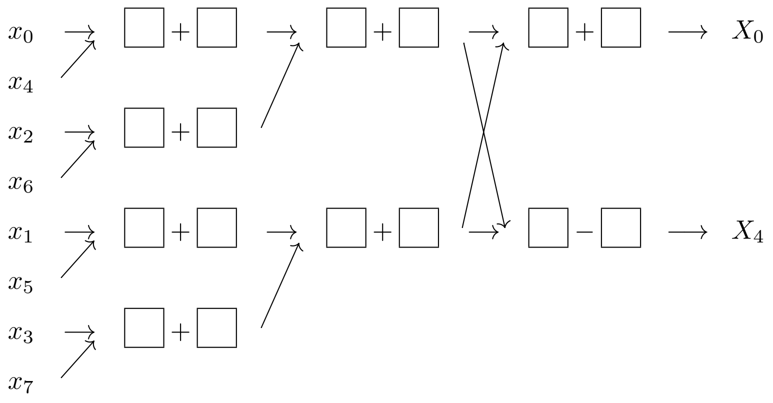


Next, we combine the arrays for each k into one array in order to save computation. The order of the outputs will be the usual order $X_0, \ldots, X_{N-1}$.

Below is the array for $k = 0$ rearranged so that the summations of entries in each column are output across from the uppermost entry:

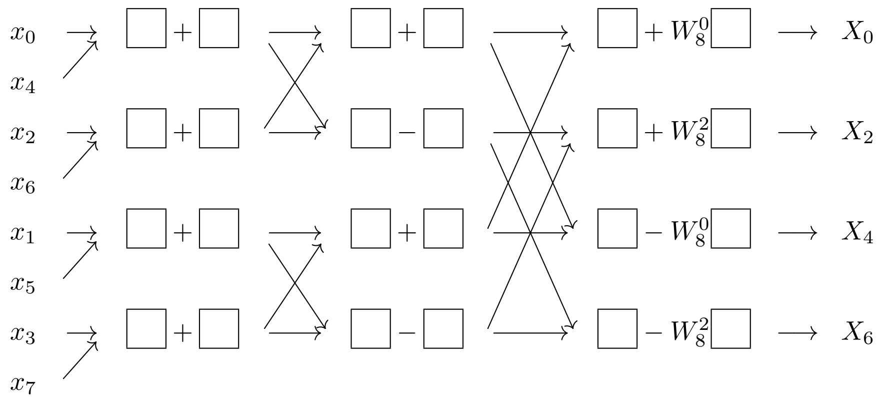


Since the calculation of X_4 differs only in the last step from X_0, we put that next into the diagram:

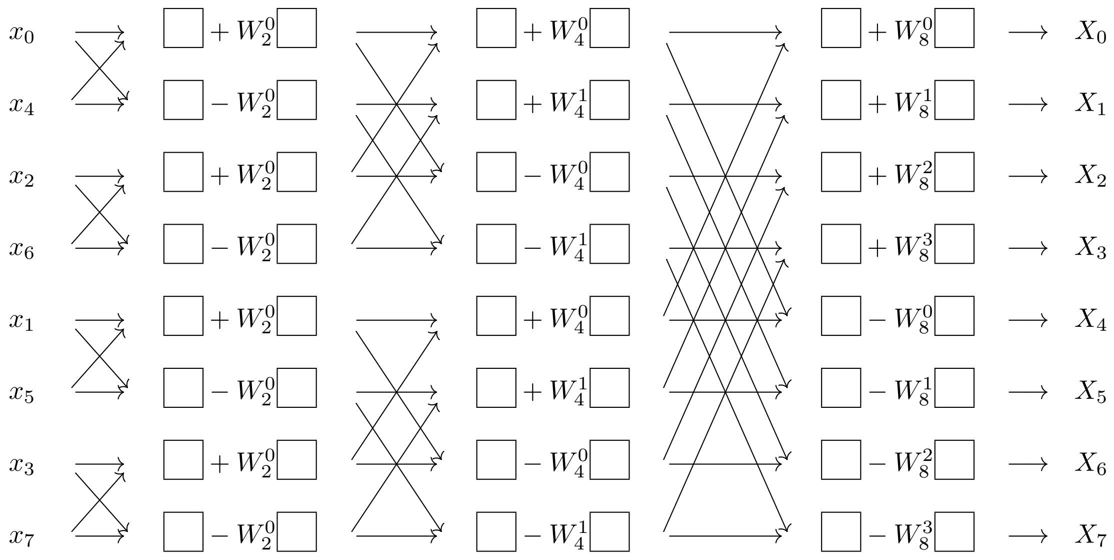


At this point we note that the inclusion of a difference is done symmetrically in a column in order to give the last step of a DFT for $N = 8$. We can also do the same for the lower order DFT's in the previous columns. This would mean that in column one (recall: column zero is just the input data), which is computing DFT's of order 2, the differences should be inserted directly below the sums. Then in column two the differences are spread out by two rows, and in column three they are spread out by four rows.

Continuing, we add the values X_2 and X_6 since they also use the same first column. These are all the values generated by using the sums of boxes in the first column. Notice that the second column now contains two types: sums or differences of boxes, and the third column contains four types: sums and differences with powers of W_8.



Next, we put all the powers of W_M back into the diagram, including the zero powers, and also fill in the array for the odd X_k to get the final diagram. This type of array is called a “butterfly" diagram.



Twiddle Factors:

With $N = 8 = 2^3$ we have three stages in the FFT computation. We can index the stages by $J = 1, 2, 3$ and note that the subscript of W_M for each stage is given by $M = 2^J$.

The first stage uses W_2. The second stage uses $W_4^i = W_{N/2}^i$ for $i = 0, 1$. The last stage uses $W_8^i = W_N^i$ for $i = 0, 1, 2, 3,$.

Since

$$W_N = e^{-i2\pi/N}$$

we also have

$$W_{N/2} = (e^{-i2\pi/N})^2 = W_N^2.$$

So all of the powers, or twiddle factors, can be written in terms of W_N.

To simplify computation, it is most efficient to compute only the needed W_M for each stage, then create the consecutive powers as needed. So, we need an array:

$$(W_2, W_4, W_8, \ldots, W_N) = (-1, -i, e^{-i\frac{\pi}{4}}, \ldots, e^{-i\frac{2\pi}{N}}).$$

We then use this to compute each stage, where we also need the powers

$$W_M^0, W_M^1, \ldots, W_M^{\frac{M}{2}-1}.$$

To compute these successive powers we can simply do multiplications like:

$$W_M^{p+1} = W_M \cdot W_M^p$$

which is more efficient than computing each power separately.

Next, each stage creates the butterfly patterns of size $M = 2, 4, 8$, etc. The butterfly pattern of size M breaks into two pieces, the top and bottom halves. The top half elements feed across and down by $M/2$ indices. The bottom half entries feed across and up by $M/2$ indices. Upper feeds into left and lower feeds into right entries in the boxes, as always.

At each stage there are N/M butterflies, each of size M. The size indicates the number of inputs and outputs to the butterfly pattern. Each butterfly can be computed in two pieces.

Suppose that the entries used as inputs into the butterfly computation are labelled $a_0, \ldots, a_{M-1}$. Suppose also that the outputs are called $b_0, \ldots, b_{M-1}$. Then the first half of the entries are computed as

$$b_j = a_j + W_M^j a_{j'},\ j = 0, \ldots, \frac{M}{2} - 1, \quad \text{where } j' = j + M/2$$

and the second half of the entries are computed as:

$$b_j = a_{j'} - W_M^{j-\frac{M}{2}} a_j,\ j = \frac{M}{2}, \ldots, M-1 \quad \text{where } j' = j - M/2.$$