COMPUTING THE FFT OF TWO REAL SIGNALS USING A SINGLE FFT

J. Shima

4/15/2000

This paper demonstrates how to compute the FFT of two real-valued signals, x1(n) and x2(n), using a single FFT.

To start out we can compute the FFT of a signal vector, denoted x(n), using the 2 following cases.

CASE A:

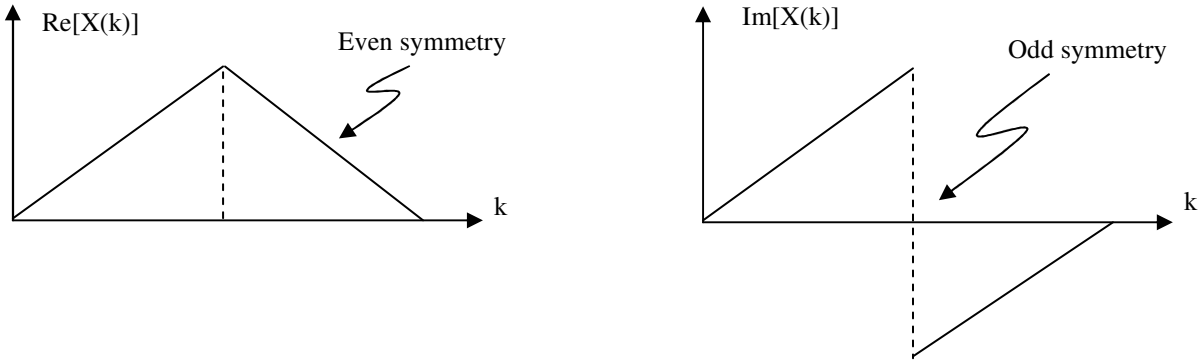
If x(n) is a real-valued signal, with n = 0 to N-1, we can perform an N-pt FFT by filling x(n) into the real part of the FFT array, and zeroing out the imaginary part of the FFT array (since x(n) is strictly real).

FFT array → real part holds [x(0) x(1) x(2) … x(N-1)]
imag part holds [0 0 0 0 … 0]

The FFT of x(n), denoted X(k), ends up being symmetric:

$$X(k) = X^*(N-k) \text{ for } k = 0 \text{ to } N-1$$

Here is a diagram showing the symmetry of X(k) for CASE A.



You can see that the real part of X(k) has even symmetry about the N/2 pt., and the imag part of X(k) has odd symmetry about the N/2 pt.

CASE B:

If x(n) is a real-valued signal, with n = 0 to N-1, we can perform an N-pt FFT by filling x(n) into the imaginary part of the FFT array, and zeroing out the real part of the FFT array (since x(n) is strictly real).

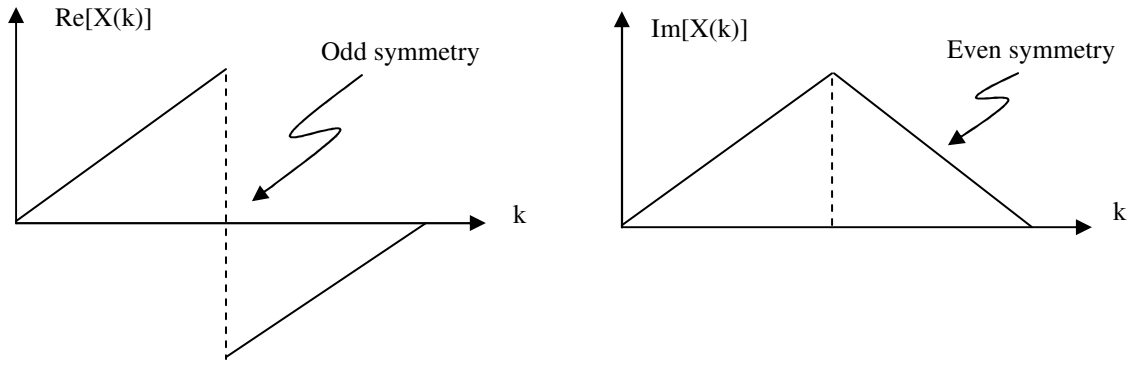
FFT array → real part holds [0 0 0 0 … 0]
imag part holds [x(0) x(1) x(2) … x(N-1)]

This is equivalent of taking the FFT of a signal y(n) = j*x(n), where j = sqrt(-1)

The FFT of x(n), denoted X(k), ends up being anti-symmetric:

$$X(k) = -X^*(N-k) \text{ for } k = 0 \text{ to } N-1$$

Here is a diagram showing the symmetry of X(k) for CASE B.



So, if we now pack the real-valued x(n) points into the imaginary part of the FFT array, the real part of X(k) is odd symmetric, and the imaginary part of X(k) is even symmetric. This is the exact opposite symmetry as seen in CASE A.

Using CASE A and CASE B, we can now formulate taking the FFT of 2 discrete time real-valued signals using a single FFT. We can take two N-pt signals, x1(n) and x2(n), and pack them into a single FFT buffer and then attempt to retrieve X1(k) and X2(k) from the results.

Let:

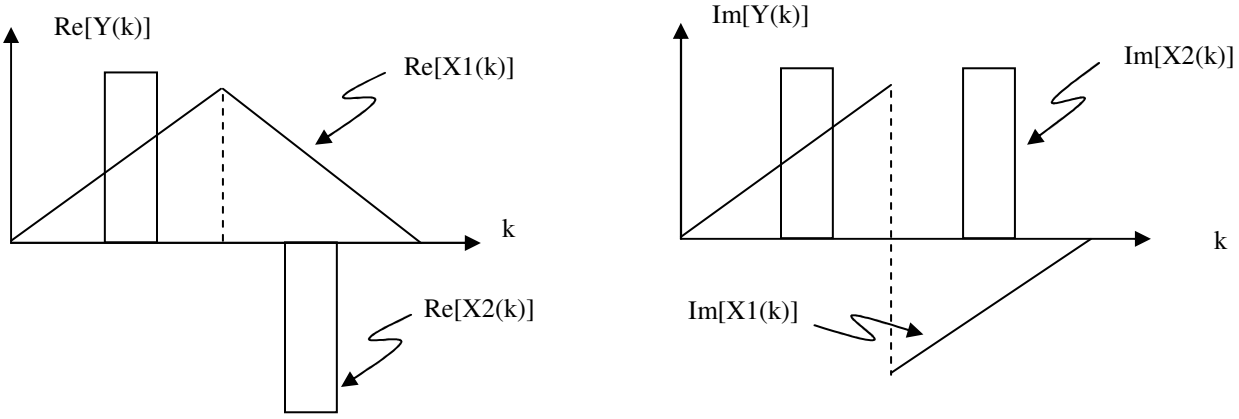
$$y(n) = x1(n) + jx2(n), \quad \text{for } n = 0 \text{ to } N-1$$

So now the FFT array hold the following.

FFT array → real part holds [x1(0) x1(1) x1(2) … x1(N-1)]
imag part holds [x2(0) x2(1) x2(2) … x2(N-1)]

When we take the FFT of the composite signal, we get Y(k), which is an addition of the original X1(k) and X2(k). But, because of the symmetry properties presented above, we can excise X1(k) and X2(k) from Y(k) without any loss of data.

Here is a diagram showing the symmetry of Y(k) = X1(k) + jX2(k).



In the above left-hand plot, we can see that Re[Y(k)] consists of Re[X1(k) + jX2(k)]. But since the jX2(k) spectrum is anti-symmetric by definition (see CASE B), we can totally recover the individual spectrums by adding the symmetric parts of the FFT and dividing by two.

That is, if we add together Re[Y(k)] + Re[Y(N-k)] in the above left-hand plot, then the signal Re[X2(k)] will cancel itself out. Thus we can separate Re[X1(k)] from Re[Y(k)].

We can write this as:

$$\text{Re}[X1(k)] = \tfrac{1}{2}(\text{Re}[Y(k)] + \text{Re}[Y(N-k)]), \quad k = 0 \text{ to } N-1$$

Or writing this in a more readable form (substituting subscript r = Re[.]):

$$X1r(k) = \tfrac{1}{2}[Yr(k) + Yr(N-k)]$$

Also looking at the left-hand plot of Re[Y(k)], we can recover Re[X2(k)] by subtracting the symmetric parts:

$$X2r(k) = \tfrac{1}{2}[Yr(k) - Yr(N-k)]$$

From the above righ-hand plot of Im[Y(k)], we can get back Im[X1(k)] and Im[X2(k)] using similar formulas:

$$X1i(k) = \tfrac{1}{2}[Yi(k) - Yi(N-k)]$$
$$X2i(k) = \tfrac{1}{2}[Yi(k) + Yi(N-k)]$$

Combining the terms for X1(k) gives:

$$X1(k) = X1r(k) + jX1i(k)$$

$$= \tfrac{1}{2}[Yr(k) + Yr(N-k)] + j/2*[Yi(k) - Yi(N-k)]$$
$$= \tfrac{1}{2}[Yr(k) + jYi(k)] + \tfrac{1}{2}[Yr(N-k) - Yi(N-k)]$$

or,

$$X1(k) = \tfrac{1}{2}[Y(k) + Y^*(N-k)]$$

Combining terms for X2(k) gives:

$$X2(k) = X2r(k) + jX2i(k)$$

$$= \tfrac{1}{2}[Yr(k) - Yr(N-k)] + j/2*[Yi(k) + Yi(N-k)]$$
$$= \tfrac{1}{2}[Yr(k) + jYi(k)] + \tfrac{1}{2}[-Yr(N-k) + Yi(N-k)]$$

or,

$$X2(k) = \tfrac{1}{2}[Y(k) - Y^*(N-k)], \quad k=0 \text{ to } N-1$$

We are not done yet. Since x2(n) was put into the imaginary buffer of the FFT, we really get:

$$X2(k) = \text{FFT}\{ j\, x2(n)\}$$

This allowed us to recover X2(k) due to symmetry, but we want the actual transform of x2(n).

$$X2(k) = \text{FFT}\{ x2(n) \}$$

So after we break out X2(k) from Y(k), we must divide the resultant X2(k) by "j" to get the actual transform of the x2(n) signal.

$$X2\_actual(k) = -jX2(k)$$

We can summarize this routine with the following equations for k = 0 to N-1:

$$\mathbf{X1(k) = \tfrac{1}{2}\,[Y(k) + Y^*(N-k)]}$$
$$\mathbf{X2(k) = -j/2\,[Y(k) - Y^*(N-k)]}$$

Where,

$$X1(k) = FFT\{\ x1(n)\ \}$$
$$X2(k) = FFT\{\ x2(n)\ \}$$

Thus, by packing x1(n) and x2(n), n=0 to N-1, into one N-pt. complex FFT buffer, we can perform a single N-pt. FFT and then extract the transforms X1(k) and X2(k) by using the above "unscrambling" equations. This saves us doing two discrete FFTs to obtain X1(k) and X2(k).

NOTE: Since we are effectively doing two N-pt. FFTs of two separate signals x1(n) and x2(n), the frequency resolution of each FFT is still fs/N.