

Two for the Price of One! The Hidden Capacity of the Complex Fast Fourier Transform

W. Smith

July 11, 1989

Introduction

The power and versatility of the complex fast Fourier transform is well known (and indeed, has been outlined in this newsletter previously [1, 2]). The purpose of this article is to point out yet another aspect of it, which leads to even greater efficiency in computational applications. It concerns the situation in which the function to be Fourier transformed is *real*; a situation that happily arises frequently in molecular dynamics applications. To see how this can be exploited however, we must do a little mathematics. Readers interested in a more thorough account should consult reference [3].

Basic Fourier Transform Properties

The standard form of the Fourier transform relates a function $h(t)$ with another function $H(f)$ through an integral:

$$H(f) = \int h(t) \exp(-2\pi i f t) dt \quad (1)$$

and under reasonable circumstances, this transform has an inverse :

$$h(t) = \int H(f) \exp(2\pi i f t) df \quad (2)$$

in which f and t (frequency and time - or some other suitable conjugate variables) define the *domains* of the two functions.

An important property possessed by many functions is that they may be described as *even* or *odd*. This simply means that if we change the sign of the argument, an even function

returns the same value, while an odd function returns the negative of the original value. i.e.

$$h(t) = h(-t)$$

for an even function, and

$$h(t) = -h(-t)$$

for an odd function. In general functions are neither even nor odd, but can be expressed as the sum of an even and an odd function i.e.

$$h(t) = e(t) + o(t)$$

with

$$e(t) = \frac{1}{2}(h(t) + h(-t))$$

and

$$o(t) = \frac{1}{2}(h(t) - h(-t)).$$

Products of functions can also be even or odd. The product of two even functions or of two odd functions is even, while the cross product is odd. These properties are largely self evident. Equally self evident is the fact that a definite integral of an odd function over an interval $[-\alpha, +\alpha]$ *must* be zero, while the corresponding integral of an even function may be nonzero.

These simple properties are very useful when employed in conjunction with the Fourier transform. Most importantly, for the purposes of this note, it can be shown that when $h(t)$ is a real function, the Fourier transform $H(f)$, which in general is *complex*, has a real part that is *even* in f and an imaginary part that is *odd* in f . This is seen most readily in the inverse Fourier transform (Eq. 2) of $H(f)$ back to $h(t)$. The $\exp(i2\pi ft)$ term may be split into an even function $\cos(2\pi ft)$ and an odd function $i \sin(2\pi ft)$ through Euler's relation. It follows simply, from what has been said above, that the real part of $H(f)$ has to be even and the imaginary part has to be odd, if the final expression of $h(t)$ is to be real.

This property becomes even more interesting if the function $h(t)$ is constructed to be a complex sum of two *real* functions (say $a(t)$ and $b(t)$) in the following way:

$$h(t) = a(t) + i b(t). \tag{3}$$

It then turns out that if this function is Fourier transformed, it is possible to extract *both* the Fourier transforms of $a(t)$ and $b(t)$ from the result with very little effort. Effectively this means that we can Fourier transform the two real functions $a(t)$ and $b(t)$ *at the same time*. This useful property merits a closer examination.

Fourier Transforming Real Functions

Let $h(t)$ be defined as in equation (3), and let its Fourier transform be $H(f)$. Let the functions $a(t)$ and $b(t)$ have Fourier transforms $A(f)$ and $B(f)$ respectively. We shall distinguish between the real and imaginary parts of the complex functions by the use of single and double dashes, *i.e.*:

$$H(f) = H'(f) + i H''(f)$$

where $H'(f)$ is the real part and $H''(f)$ the imaginary part.

Since equation (3) represents a linear combination of two functions, we can write directly:

$$H(f) = A(f) + i B(f).$$

(This simple equation is deceptive; $A(f)$ and $B(f)$ are complex!) Separating out the real and imaginary parts of all these functions gives:

$$H'(f) = A'(f) - B''(f) \tag{4}$$

$$H''(f) = A''(f) + B'(f). \tag{5}$$

```
PARAMETER (N=2**integer)
DIMENSION A(N),B(N),AI(N),BI(N)
COMPLEX H(N)
C
C LOAD COMPLEX ARRAY H(N) WITH REAL A(N) AND B(N)
DO 100 I=1,N
100 H(I)=CMPLX(A(I),B(I))
C
C CALL SYSTEM FAST FOURIER TRANSFORM
CALL FFT(N,H,plus system specific parameters)
C
C EXTRACT FOURIER TRANSFORMS OF A AND B
A(1)=REAL(H(1))
AI(1)=0.0
B(1)=AIMAG(H(1))
BI(1)=0.0
DO 200 I=2,N
A(I) = 0.5*( REAL(H(I)) + REAL(H(N+2-I)))
AI(I) = 0.5*(AIMAG(H(I)) - AIMAG(H(N+2-I)))
```

```

      B(I) = 0.5*(AIMAG(H(I)) + AIMAG(H(N+2-I)))
      BI(I) = 0.5*(-REAL(H(I)) + REAL(H(N+2-I)))
200 CONTINUE
      . . . . .
      etc.

```

Figure 1. FORTRAN Code for Fourier Transform of
Two Real Functions Simultaneously

Also, for negative values of frequency we can write:

$$H'(-f) = A'(-f) - B''(-f) \quad (6)$$

$$H''(-f) = A''(-f) + B'(-f) \quad (7)$$

We can now use what we know about the real and imaginary parts of the Fourier transforms $A(f)$ and $B(f)$, namely that the real parts are even and the imaginary parts are odd, to rewrite (6) and (7) as:

$$H'(-f) = A'(f) + B''(f) \quad (8)$$

$$H''(-f) = -A''(f) + B'(f) \quad (9)$$

Lastly, combining equations (4),(5),(8) and (9), we obtain

$$\begin{aligned}
 A'(f) &= \frac{1}{2}(H'(f) + H'(-f)) \\
 A''(f) &= \frac{1}{2}(H''(f) - H''(-f)) \\
 B'(f) &= \frac{1}{2}(H''(f) + H''(-f)) \\
 B''(f) &= \frac{1}{2}(-H'(f) + H'(-f))
 \end{aligned} \quad (10)$$

These equations clearly show that, once the components of $H(f)$ are obtained by Fourier transforming $h(t)$, we can easily calculate the components of $A(f)$ and $B(f)$, which constitute the Fourier transforms of $a(t)$ and $b(t)$.

The realisation of this method in rude FORTRAN is particularly simple, and is presented in Figure 1. The complex array $H(N)$ is constructed from the two real arrays $A(N)$ and $B(N)$. A standard fast Fourier transform (FFT) routine (every computing system has one!) calculates the discrete Fourier transform. The real parts of the Fourier transforms of A and B overwrite the original arrays. The imaginary parts are placed in arrays $AI(N)$ and $BI(N)$ respectively.

Applications of this trick are obviously many, but a particularly nice application might be the method of Osguthorpe *et al* [4], which Fourier transforms of the trajectories of atoms in MD simulations of complex molecules. The purpose is to ‘project out’ particular frequencies (for example, those corresponding to particular modes of vibration) prior to displaying the trajectories in moving graphics. The effect of this technique is visually stunning and greatly reduces the ‘visual chaos’ effect of conventional MD movies. It is apparent here that long Fourier transforms of many trajectories are required, and any trick that reduces the computational cost is beneficial. (Indeed, it is not even necessary in this application to extract the final Fourier transforms since the data can be filtered and inverse transformed without needing to do this.)

Convolution and Correlation

A well known application of the fast Fourier transform is to speed up the calculation of convolution and correlation integrals [2, 3]. Since in MD we are usually confronted with real data, we may ask if the above trick can be exploited here also, to permit the calculation of (say) two convolution integrals at the same time. This is indeed the case, though the algorithm is not so easily described. I shall attempt to outline the method for the calculation of a convolution integral, the corresponding treatment for correlation integrals is very similar.

The standard form of a convolution integral is

$$p(t) = \int a(\tau)b(t - \tau) d\tau \quad (11)$$

and the equivalent expression in the frequency domain is [3]

$$P(f) = A(f)B(f) \quad (12)$$

where $A(f)$, $B(f)$ and $P(f)$ are the Fourier transforms of $a(t)$, $b(t)$ and $p(t)$ respectively. The simplicity of this expression, coupled with the computational efficiency of the FFT is what makes the FFT method of computing convolution integrals so attractive.

Now if $a(t)$ and $b(t)$ are real functions, and we introduce two additional real functions $c(t)$ and $d(t)$, with convolution $q(t)$, we can calculate both $p(t)$ and $q(t)$ at the same time in the following manner.

Define two complex functions $g(t)$ and $h(t)$ as follows

$$g(t) = a(t) + i c(t)$$

$$h(t) = b(t) + i d(t).$$

If we Fourier transform $g(t)$ and $h(t)$ we get $G(f)$ and $H(f)$ respectively, whose real and imaginary parts we represent in the usual notation

$$G(f) = G'(f) + i G''(f)$$

$$H(f) = H'(f) + i H''(f).$$

Following the above account of the simultaneous Fourier transform of two real functions we can write immediately (c.f equation (10))

$$\begin{aligned} A'(f) &= \frac{1}{2}(G'(f) + G'(-f)) \\ A''(f) &= \frac{1}{2}(G''(f) - G''(-f)) \\ C'(f) &= \frac{1}{2}(G''(f) + G''(-f)) \\ C''(f) &= \frac{1}{2}(-G'(f) + G'(-f)) \\ B'(f) &= \frac{1}{2}(H'(f) + H'(-f)) \\ B''(f) &= \frac{1}{2}(H''(f) - H''(-f)) \\ D'(f) &= \frac{1}{2}(H''(f) + H''(-f)) \\ D''(f) &= \frac{1}{2}(-H'(f) + H'(-f)) \end{aligned} \tag{13}$$

The function products we require in the frequency domain are

$$P(f) = A(f)B(f)$$

and

$$Q(f) = C(f)D(f).$$

The products on the right of these equations can be expanded into

$$A(f)B(f) = (A'(f)B'(f) - A''(f)B''(f)) + i (A'(f)B''(f) + A''(f)B'(f))$$

and

$$C(f)D(f) = (C'(f)D'(f) - C''(f)D''(f)) + i (C'(f)D''(f) + C''(f)D'(f))$$

from which it is obvious that

$$\begin{aligned}
 P'(f) &= A'(f)B'(f) - A''(f)B''(f) \\
 P''(f) &= A'(f)B'''(f) + A''(f)B'(f) \\
 Q'(f) &= C'(f)D'(f) - C''(f)D''(f) \\
 Q''(f) &= C'(f)D'''(f) + C''(f)D'(f)
 \end{aligned}
 \tag{14}$$

Clearly, by way of the equations (13), all these components may be calculated from the components of $G(f)$ and $H(f)$.

```

PARAMETER (N=2**Integer)
DIMENSION A(N),B(N),C(N),D(N),P(N),Q(N)
COMPLEX G(N),H(N),Z(N)
RNORM=1.0/FLOAT(N)
C CONSTRUCT COMPLEX ARRAYS G(N) AND H(N)
DO 100 I=1,N
G(I)=CMPLX(A(I),C(I))
H(I)=CMPLX(B(I),D(I))
100 CONTINUE
C CALCULATE FOURIER TRANSFORMS OF G(N) AND H(N)
CALL FFT(N,G,plus system specific parameters)
CALL FFT(N,H,plus system specific parameters)
C CONSTRUCT COMPLEX Z(N) ARRAY
Z1=REAL(G(I))*REAL(H(I))
Z2=AIMAG(G(I))*AIMAG(H(I))
Z(1)=RNORM*CMPLX(Z1,Z2)
DO 200 I=1,N/2
AI1=0.5*(REAL(G(I+1))+REAL(G(N+1-I)))
AI2=0.5*(AIMAG(G(I+1))-AIMAG(G(N+1-I)))
CI1=0.5*(AIMAG(G(I+1))+AIMAG(G(N+1-I)))
CI2=0.5*(-REAL(G(I+1))+REAL(G(N+1-I)))
BI1=0.5*(REAL(H(I+1))+REAL(H(N+1-I)))
BI2=0.5*(AIMAG(H(I+1))-AIMAG(H(N+1-I)))
DI1=0.5*(AIMAG(H(I+1))+AIMAG(H(N+1-I)))
DI2=0.5*(-REAL(H(I+1))+REAL(H(N+1-I)))
PI1=RNORM*(AI1*BI1-AI2*BI2)
PI2=RNORM*(AI1*BI2+AI2*BI1)
QI1=RNORM*(CI1*DI1-CI2*DI2)
QI2=RNORM*(CI1*DI2+CI2*DI1)

```

```

      Z(I+1)=CMPLX(PI1-QI2,PI2+QI1)
      Z(N+1-I)=CMPLX(PI1+QI2,QI1-PI2)
200  CONTINUE
C    INVERSE FOURIER TRANSFORM ARRAY Z(N)
      CALL FFT(N,Z,plus system specific parameters)
C    EXTRACT CONVOLUTION ARRAYS
      DO 300 I=1,N
      P(I)= REAL(Z(I))
      Q(I)=AIMAG(Z(I))
300  CONTINUE
      .      .      .      .      .      .
      etc.

```

Figure 2. FORTRAN Code for Calculation of Two Real Convolution Integrals Simultaneously.

This however is only part of the story. The product obtained in the frequency domain must now be inverse Fourier transformed to obtain the final convolution. That is we must obtain $p(t)$ and $q(t)$ from $P(f)$ and $Q(f)$. Since we know that $p(t)$ and $q(t)$ are real functions, it would be nice to obtain our result in an economical and straightforward way, using the lessons we have learned already. We can do this as follows.

Define a complex function $z(t)$ such that

$$z(t) = p(t) + i q(t) \quad (15)$$

We now say this has a Fourier transform $Z(f)$ with components $Z'(f)$ and $Z''(f)$. As before, these components are related to the Fourier transforms of $p(t)$ and $q(t)$, namely $P(f)$ and $Q(f)$:

$$\begin{aligned}
 P'(f) &= \frac{1}{2}(Z'(f) + Z'(-f)) \\
 P''(f) &= \frac{1}{2}(Z''(f) - Z''(-f)) \\
 Q'(f) &= \frac{1}{2}(Z''(f) + Z''(-f)) \\
 Q''(f) &= \frac{1}{2}(-Z'(f) + Z'(-f)).
 \end{aligned} \quad (16)$$

We may rearrange these equations to give

$$Z'(f) = P'(f) - Q''(f)$$

$$\begin{aligned}
Z''(f) &= P''(f) + Q'(f) \\
Z'(-f) &= P'(f) + Q''(f) \\
Z''(-f) &= -P''(f) + Q'(f).
\end{aligned}
\tag{17}$$

These final equations tell us how to construct the function $Z(f)$ in the frequency domain so that on inverse Fourier transforming it to $z(t)$ we find the convolution functions $p(t)$ and $q(t)$ in the real and imaginary parts respectively, of the function $z(t)$.

This completes the description of the method. Its implementation in FORTRAN appears in Figure 2, where the variable names are the same as in the above text. The variable RNORM is a factor which correctly renormalises the functions after inverse Fourier transforming them. The reader may like to derive the corresponding algorithm for correlation functions, but beware the product of Fourier transforms in the frequency domain, since one of the functions must now be a complex conjugate.

This algorithm also has obvious applications in MD. Less well known is its application in quantum simulations [5], about which we hope to say more at a later date!

References

- [1] W. Smith, CCP5 Info. Quart. **5** 34 (1982).
- [2] W. Smith, CCP5 Info. Quart. **7** 12 (1984).
- [3] E.O. Brigham, *The Fast Fourier Transform and its Applications*, Prentice Hall, NJ 1988.
- [4] D.J. Osguthorpe, Bath NATO Summer School, September 1988, "*Fluids, Polymers and Solids*".
- [5] K. Singer and W. Smith, paper in preparation (1989).