

---

Subject: Re: Secrets FFTs revealed!!

Posted by [Paul van Delst](#) on Fri, 05 May 2000 07:00:00 GMT

[View Forum Message](#) <> [Reply to Message](#)

---

Peter Brooker wrote:

```
>
> <snip>
>
> Divide the interval into N sections.  $t \sim t_i = i \cdot T/N$ 
> Then,
>  $f(t_i) = \sum_n (A_n \cdot \exp(j \cdot 2 \cdot \pi \cdot n \cdot t_i / T))$ 
>  $= \sum_n (A_n \cdot \exp(j \cdot 2 \cdot \pi \cdot n \cdot i \cdot (T/N) / T))$ 
>  $= \sum_n (A_n \cdot \exp(j \cdot 2 \cdot \pi \cdot n \cdot i / N))$ ,  $n=0,1,\dots$ 
>
> This is exactly what is found in the IDL manual under the section for
> FFT. The only difference is that  $t$  has been replaced by  $u$  and  $A_n$  has
> been replaced by  $F(u)$ . Note that the period  $T$  has dropped out. Also note
> that  $t$  has been replaced by  $t_i = i \cdot T/N$ . In order for this to happen,
> the interval over which  $t$  is defined must be from  $[0, T]$ . This is
> different from the definition of  $t$  being defined over the interval
>  $[-T/2, T/2]$ . Perhaps this is why  $b_n = -bb_n$ .
>
> *****UNFORTUNATELY IT IS WRONG*****
>
> What is wrong is the values of  $n$  in the sum. IDL does not use the values
> of  $n=0,1,2,\dots$  IDL actually uses  $n = -N/2+1, -N/2+2, \dots, -1, 0, 1, \dots, N/2$ 
> The reason for doing this must have to do with FFT theory. Note also
> that the number of values of  $n$  is  $N$ .
```

Great job on the ref but I have always found it hard to read ASCII equations. :o)

The input to the FFT should include BOTH the positive and negative frequencies. So if you have a function of  $N+1$  points, then you want to supply the FFT with  $2N$  points (or if you have a function of  $N/2 + 1$  points you supply it with  $N$  points. The "+1"th point is the Nyquist point.

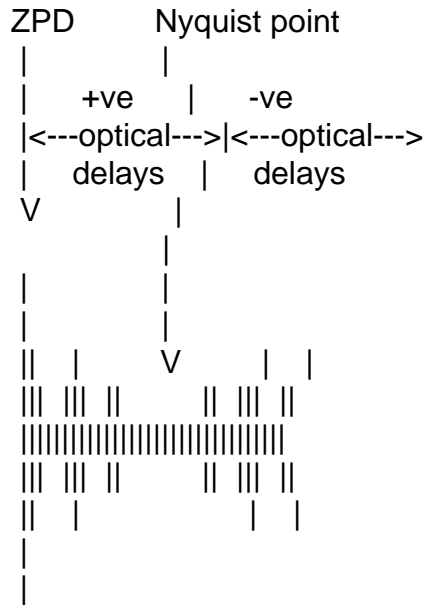
Say you have the following real function (e.g. an interferogram):

```

ZPD      Nyquist point
|         |
|   +ve   |
|<---optical--->|
|   delays   |
V           |
|           |
```



where ZPD == zero path difference. If you want to FFT this function to a spectrum, the input to the FFT should be:



where the function values at negative optical delays are simply the reflected positive ones. Note that neither the Nyquist point nor the ZPD point is reflected - the reflection occurs *around* the Nyquist point. The ZPD (or zero frequency point) is unique and the Nyquist point is ambiguous, i.e. it contains both +ve and -ve frequency info.

The IDL FFT documentation mentions the storage of the negative frequencies. It's just that given a real function (e.g. something measured), one simply repeats the +ve frequency values into -ve frequencies.

So, I don't think the documentation is wrong but the most I would be willing to bet is a beer or two (due to my lack of understanding, not RSI's).

Here a section of the header to my `fft_to_interferogram.pro` mentioning the reflection. Check out the input/output array sizes in the example section:

```
; PROCEDURE:
;   The input spectrum is reflected about it highest frequency
;   (largest wavenumber). The spectrum is FFT'd and the
;   resultant interval is scaled by the input spectrum wavenumber
;   interval.
;
;   The interferogram is then shifted so that the ZPD occurs at the
;   centre (like a measured IFG) and the redundant most negative
;   Nyquist point is placed at the end of the interferogram array.
```

```
; EXAMPLE:
```

```
;   Given a spectrum:
```

```
;   IDL> HELP, spc, v
;   SPC          FLOAT    = Array[12445]
;   V           FLOAT    = Array[12445]
```

```
;   The Fourier transform can be found by typing:
```

```
;   IDL> PRINT, fft_to_interferogram( spc, v, ifg, opd )
;   1
```

```
;   resulting in an interferogram and optical delay grid:
```

```
;   IDL> HELP, ifg, opd
;   IFG          DOUBLE   = Array[24889]
;   OPD          DOUBLE   = Array[24889]
```

and here's the code snippet that actually does the reflection, FFT'ing.  
Take note of the bounds of the REVERSE'd portion of "spectrum" - no zero  
frequency (point 0) and no Nyquist frequency (point n\_spectrum\_pts-1)  
reflection:

```
;-----
;   -- Fourier transform the input spectrum --
;-----

;-----
; Reflect the spectrum
;-----

    spectrum_to_fft = TEMPORARY( [ spectrum, REVERSE( spectrum[ 1:
n_spectrum_pts - 2 ] ) ] )

;-----
```

; FFT the spectrum

; -----

```
interferogram = FFT( TEMPORARY( spectrum_to_fft ), /DOUBLE, /INVERSE )
```

cheers,

paulv

--

Paul van Delst            Ph: (301) 763-8000 x7274

CIMSS @ NOAA/NCEP       Fax: (301) 763-8545

Rm.202, 5200 Auth Rd.   Email: [pvandelst@ncep.noaa.gov](mailto:pvandelst@ncep.noaa.gov)

Camp Springs MD 20746

---