
Subject: Re: FFT accuracy

Posted by [dale](#) on Thu, 23 Apr 1992 13:59:27 GMT

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

In article <1992Apr20.172439.11546@colorado.edu>, ali@anchor.cs.colorado.edu (Ali Bahrami) writes...

> ...A double-precision FFT would, of course, provide better accuracy, but
> is not provided because there is no double precision complex data
> type.

Since IMSL/IDL has just been released a few of us have been looking at this discussion with interest. Finally Mike Pulverenti came up with the following observations that give a workaround and hope for the future:

IMSL/IDL is developing a scheme to compute double precision complex FFT's of complex arrays internally, but return the results in the current complex data type of IMSL/IDL. If the IMSL/IDL routine FFTCOMP is called with a complex array, and the DOUBLE keyword is present and nonzero, then IMSL/IDL will internally promote your data to double complex, compute the FFT in double precision, and then demote the results back to single precision complex for return to the user.

If you can't wait for the next release of IMSL/IDL, you can alternatively compute a double precision complex FFT of a real array by computing a double precision real FFT, then permute the result to mimic the results of a complex FFT. This will, in effect, produce a double precision complex FFT of real data. The example below demonstrates the pattern of the permutation needed.

```
==> x = random(5)           ; Define an array of random numbers.
==> pm, fftcomp(x, /double)   ; Compute the double precision
    3.5419804                ; real FFT.
    0.17009673
    0.16646779
   -0.045797205
    0.11475440
==> pm, fftcomp(x, /complex)   ; Note that every element of the
(   3.54198,    0.00000)       ; complex FFT appears in the
(   0.170097,   0.166468)       ; double precision FFT.
(  -0.0457972,   0.114754)
(  -0.0457972,  -0.114754)
(   0.170097,  -0.166468)
```

Using a double precision FFT code should certainly help if you require more accuracy, but the examples given in article <9204171736.AA03199@dip.eecs.umich.edu>, pan@ZIP.EECS.UMICH.EDU do not imply poor accuracy of the FFT's.

One method to determine the accuracy of the FFT's is by first examining the 'best case' scenario of computing the DFT by means of a simple matrix-vector operation involving an NxN orthogonal matrix F, where $F(k,j)$ is defined to be $\exp((2*(PI)*k*j)i/N)$. (The 'i' in the last expression denotes the imaginary part of a complex number.) Computing the DFT by this method is known to be stable numerical process. In order to determine the numerical stability of the FFT algorithm you are using, you can compare it's results with the original DFT.

Dale

Have a good day!

Dale L. Neaderhouser dale@imsl3.imsl.com FAX: 713-242-9799
Senior Software Engineer uunet!imsl!dale IMSL: 713-242-6776
Post Sales Technical Support: 800-324-4675 Sales: 800-222-4675
