Mon. Not. R. astr. Soc. (1985) **213,** 773–776

An algorithm for significantly reducing the time necessary to compute a Discrete Fourier Transform periodogram of unequally spaced data

D. W. Kurtz *Department of Astronomy, University of Cape Town, Rondebosch 7700, South Africa*

Accepted 1984 November 9. Received 1984 September 19

Summary. A technique for reducing the computation time for a Discrete Fourier Transform by a factor of 4 to 6 is presented. There is no significant loss of accuracy. FORTRAN code which will directly replace Deeming's FORTRAN code is given.

1 Introduction

The Discrete Fourier Transform is a straightforward way to estimate the periodogram of unequally spaced data (Deeming 1975). Scargle (1982) has shown that, with slight modifications, Deeming's technique is equivalent to least-squares fitting. I find that in practice, for large data sets, Scargle's modifications of Deeming's definition of the Discrete Fourier Transform produce differences in the periodogram which are not significant. One can, therefore, think of the Deeming technique as equivalent to a least-squares estimate of the periodogram.

Deeming's technique is widely used in astronomy to derive oscillation frequencies from the light curves of pulsating stars. Readers who wish to see how widespread this use is can consult the Deeming (1975) reference in *Science Citation Index*. Others who may wish to see just an example of its use can look at my analysis of the rapidly oscillating Ap stars (Kurtz 1982) or O'Donoghue & Warner's (1982) analysis of the ZZ Ceti star, L19-2.

The Discrete Fourier Transform is defined by Deeming to be

$$F_N(\nu_j)=\sum_{k=1}^{N} f(t_k)\exp(i2\pi\nu_j t_k) \tag{1}$$

where $f(t_k)$ in astronomy normally represents the magnitudes or intensities measured at the times t_k, i.e. the points in the light curve of a star. Normally the real and imaginary parts of equation (1) are computed by putting it in the form

$$F_N(\nu_j)=\sum_{k=1}^{N} f(t_k)\cos(2\pi\nu_j t_k)+i\sum_{k=1}^{N} f(t_k)\sin(2\pi\nu_j t_k). \tag{2}$$

Since astronomical stellar light curves often consist of $\sim 10^4$ data points, and since it is also often necessary to calculate the Discrete Fourier Transform at $\sim 10^4$ frequencies in order to produce a fully resolved periodogram, these computations involve calculating $\sim 10^8$ sines and cosines. Most computers calculate trigonometric functions using one of several Tchebychev polynomials depending on the value of the operand. The University of Cape Town SPERRY 1100/81 takes just over $60\,\mu s$ to calculate a sine or cosine which is probably typical of many mainframe computers. This means that the computation of 10^8 sines and cosines will take over three hours of CPU time. This is a major computing limitation.

O'Donoghue (1981; O'Donoghue & Warner 1982) has shown how substantial computing time can be saved for large data sets by co-adding correctly phased, low-resolution Fourier transforms of pieces of the large data set interpolated to high resolution. That technique makes it possible to analyse data sets of any size at the expense of large amounts of storage space and considerable bookkeeping on the part of the person doing the computations.

Whether using O'Donoghue's modifications or not, many users of Deeming's technique use the FORTRAN code which he gives in the appendix of his 1975 paper. This involves the straightforward computation of equation (2) for a number of frequencies to produce a periodogram.

2 The algorithm

The periodograms calculated using Deeming's technique are always calculated for equally spaced frequencies. It is possible, therefore, to derive a recursion formula which avoids the necessity of calculating most of the sines and cosines inherent in multiple computations of equation (2). For equally spaced frequencies

$$\nu_j = \nu_{j-1} + \Delta\nu.$$

Substituting in equation (2) gives

$$F_N(\nu_j) = \sum_{k=1}^{N} f(t_k) \cos(2\pi\nu_{j-1}t_k + 2\pi\Delta\nu t_k) + i \sum_{N}^{N} f(t_k) \sin(2\pi\nu_{j-1}t_k + 2\pi\Delta\nu t_k). \tag{4}$$

Simple trigonometric identities allow us to rearrange equation (4) to

$$F_N(\nu_j) = \sum_{k=1}^{N} f(t_k) [\cos(2\pi\nu_{j-1}t_k) \cos(2\pi\Delta\nu t_k) - \sin(2\pi\nu_{j-1}t_k) \sin(2\pi\Delta\nu t_k)]$$
$$+ i \sum_{k=1}^{N} f(t_k) [\sin(2\pi\nu_{j-1}t_k) \cos(2\pi\Delta\nu t_k) + \cos(2\pi\nu_{j-1}t_k) \sin(2\pi\Delta\nu t_k)]. \tag{5}$$

By using equation (5) to calculate the real and imaginary parts of the Discrete Fourier Transform for all frequencies of interest for each $f(t_k)$, I find that the computation time is reduced by a factor of typically 4 to 6 depending on the number of frequencies searched and the number of data points. In the Appendix, I give FORTRAN code which can be substituted directly for Deeming's (1975) code.

The question of the comparative accuracy of Deeming's code and mine is important. Deeming's code calculates the sines and cosines independently so they are accurate to the error in the computer being used. For most computers this will be a binary truncation error rather than a round-off error. My code, on the other hand, because it is recursive, accumulates error

from frequency to frequency. The errors for the cosine should be essentially random, because of the subtraction of two truncated products; the errors for the sine will accumulate because of the addition of two truncated products.

I have compared the code in the Appendix with Deeming's code for large and small data sets and for periodograms of 2000 and 4000 frequencies. I can find no differences between the two codes larger than $\frac{1}{2}\sigma$ for the calculated amplitudes and phases. Generally the differences are substantially less than this. I recommend that any prospective user of the code presented in this paper should test it to his or her own satisfaction.

I also recommend that Deeming's technique should only be used as a search technique. When a frequency has been found in a data set, least-squares fitting should be used to find the optimum amplitude and phase associated with that frequency. When only one frequency is present in the data the Discrete Fourier Transform and least-squares fitting should give the same amplitudes and phases. One gains confidence seeing that this is true and the least-squares fitting also gives error estimates for the amplitudes and phases. When more than one frequency is present in the data and frequencies are sequentially removed from the data to search for further frequencies, it is imperative that the amplitudes and phases should be optimized by multivariate least-squares fitting. The amplitudes and phases so determined are *not* equivalent to the amplitudes and phases determined from sequential application of the Discrete Fourier Transform.

Acknowledgments

I would like to thank Dr Bill Whelau for suggesting this idea and Dr Darragh O'Donoghue for useful discussions.

References

Deeming, T. J., 1975. *Astrophys. Space Sci.*, **36,** 137.
Kurtz, D. W., 1982. *Mon. Not. R. astr. Soc.*, **200,** 807.
O'Donoghue, D., 1981. *PhD thesis,* University of Cape Town.
O'Donoghue, D. & Warner, B., 1982. *Mon. Not. R. astr. Soc.*, **200,** 563.
Scargle, J. 1982. *Astrophys. J.*, **263,** 835.

Appendix

The following FORTRAN code is meant to replace the FORTRAN code given by Deeming (1975). Note that the nesting of the DO-loops has been inverted. The AMOD statements are to avoid out-of-range errors in small machines. *FF* is defined here to be amplitude rather than power. $XN=N^2$. *XNU* is the array containing the frequencies. One can use this same code to calculate the spectral window by setting $F(I)=1.0$ for all I. Users with data sets covering long time spans with times recorded to high accuracy may need to define the array *T* and the variables *X*, *XFO*, and *XDF* as double precision. The AMOD in that case will need to be replaced with DMOD. As an example, in a machine with a 32-bit word length, floating point numbers may be represented with 1 bit for the sign, 7 bits for the exponent, 1 bit for the sign of the exponent, and 23 bits for the mantissa. This gives slightly less than 8 decimal digits of accuracy. If times are recorded as

Heliocentric Julian Dates to an accuracy of 1 s, then only two decimal places are available before the decimal point.

```
      DO 1 J=1, KD
      FR(J)=0.
1     FI(J)=0.
      DO 3 I=1, N
      F0=(KL-1)*DF
      X=TWOP I*T(I)
      XF0=X*F0
      XF0=AMOD(XF0, TWOPI)
      XDF=X*DF
      XDF=AMOD(XDF, TWOPI)
      S(0)=SIN(XF0)
      C(0)=COS(XF0)
      SDF=SIN(XDF)
      CDF=COS(XDF)
      DO 2 J=1, KD
      C=C0*CDF-S0*SDF
      S=S0*CDF+C0*SDF
      FR(J)=FR(J)+F(I)*C
      FI(J)=FI(J)+F(I)*S
      C0=C
2     S0=S
3     CONTINUE
      DO 4 J=1, KD
      XNU(J)=(KL+J-1)*DF
      FF(J)=2. *SQRT((FR(J)*FR(J)+FI(J)*FI(J))/XN)
      IF (FR(J). EQ. 0. )PHASE(J)=TWOP I/4. 0
4     IF(FR(J). NE. 0. )PHASE(J)=ATAN2(-FI(J), FR(J))
```