T-1179

EVALUATION OF THE ALGORITHMS USED FOR THE FOURIER TRANSFORM

By

Douglas F. Miller

T-1179

A Thesis submitted to the Faculty and the Board of Trustees of the Colorado School of Mines in partial fulfillment of the requirements for the degree of Master of Science in Mathematics.

Signed: Douglas F. Miller
Douglas F. Miller

Golden, Colorado

Date: May 5, 1969

Approved: A. R. Brown, Jr.
A. R. Brown, Jr.
Thesis Advisor

Joseph R. Lee
Joseph Lee
Head of Department of Mathematics

Golden, Colorado

Date: May 5, 1969

ABSTRACT

An evaluation of algorithms for Fourier transformation was accomplished using a CDC 8090 and a CDC 6400 computer. Two input functions were used to help evaluate truncation errors.

Two computers with different lengths of computer word were used to show the effect of word length on accuracy of calculations.

Definite comparisons can be drawn between the algorithms to determine which algorithm is the optimum one to use in many specific cases.

ACKNOWLEDGMENTS

The author wishes to state his gratitude to Dr. A. R. Brown for his assistance in the preparation of this manuscript.

Special thanks go to Dr. R. L. Gray for stimulating interest in the numerical methods for calculating Fourier transforms.

Special considerations of acknowledgment go to the following for their financial support in the forms of teaching assistantships and graduate research assistantships:

Colorado School of Mines

Colorado School of Mines State Fund for Organized Research

Colorado School of Mines Foundation, Inc./ Faculty Improvement Fund

CONTENTS

ILLUSTRATIONS

INTRODUCTION

In many fields of science and engineering the Fourier transform is a very powerful tool in analysis. It has become possible with the modern digital computers of today to calculate the finite Fourier transform for much of the physical data to be analyzed.

It is the purpose of this thesis to make a time-and-error comparison of the algorithms for the finite transform. To the knowledge of the author there are basically seven different algorithms with which to calculate a finite Fourier transform. All but the seventh algorithm were run on a CDC 8090 and a CDC 6400. The seventh algorithm (FT7) was run on a SDS-9300 computer with the results being comparable to the previous six.

In the first chapter a brief derivation of the finite Fourier transform is given. In chapter two a description of the seven algorithms is given. Chapter three describes the input functions and shows some selected output of the algorithms.

An evaluation of the errors between the algorithms, computers, and functions is given in chapter four.

Chapter five gives a summary of the results and lists the conclusions arrived at by the evaluations.

T-1179

DISCRETE FOURIER TRANSFORMS

The Fourier transform of a function x(t) is defined as:

$$F(\omega) = \int_{-\infty}^{\infty} x(t)e^{-i\omega t}dt \qquad (1\text{-}1)$$

where $i = \sqrt{-1}$

ω = angular frequency = $2\pi f$

f = frequency

The inverse of this transform is

$$x(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{i\omega t}d\omega \qquad (1\text{-}2)$$

In order to handle this integration on a digital computer, the finite Fourier transform is incorporated using sampled, truncated data. To make a discrete function out of a continuous function, the continuous data must be "sampled." An approach to sampling the data can be accomplished by using the impulse function (Gray, 1965; McNett, 1967).

The impulse or delta function is defined (Papoulis, 1962) as:

$$\int_{-\infty}^{\infty} \delta(t)dt = 1; \quad \delta(t) = 0 \text{ for } t \neq 0 \qquad (1\text{-}3)$$

or by the property

$$\int_{-\infty}^{\infty} \delta(t)\ x(t)dt = x(o)$$

where x(t) is a continuous function. This property states that the integration of x(t) with δ(t) assigns to the function x(t) the value at x(o). This function implies then that the delta function is a distribution or a generalized function. A more precise treatment of the concept of distributions can be found in (Lee, 1960; Papoulis, 1962; Lighthill, 1964; and Zemanian, 1965).

An extension of the delta function to include all values of t leads to the following definition

$$\int_{-\infty}^{\infty} \delta(t - t_n)\ x(t)dt = x(t_n) \qquad (1\text{-}4)$$

so that now the integral assigns to the continuous function x(t) the values at $x(t_n)$. Also, equation (1-4) can be defined over a finite range, say -M to M.

Let the function,

$$X_M(t) = \begin{cases} x(t) & -M<t<M \\ 0 & t<-M,\ t>M \end{cases}$$

then

$$\int_{-M}^{M} \delta(t - t_n)\ x(t)dt = \int_{-\infty}^{\infty} \delta(t - t_n)X_M(t)dt$$

or equation (1-4) over the finite range -M to M would be,

$$\int_{-M}^{M} \delta(t - t_n)\ x(t)dt = \begin{cases} X_M(t_n) = x(t_n) & -M<t_n<M \\ X_M(t_n) = 0 & t_n<-M,\ t_n>M \end{cases} \qquad (1\text{-}5)$$

By considering a finite sequence of delta functions (spikes) a sampling operation is represented and can be defined by

$$S(t) = \sum_{n=-M}^{M} \delta(t - t_n) \qquad (1\text{-}6)$$

such that there are equally weighted spikes located along the t-axis at positions t_n.

If the spikes are located at equal intervals of Δt, then $t_n = n\Delta t$, and equation (1-5) becomes

$$S_M(t) = \sum_{n=-M}^{M} \delta(t - n\Delta t). \tag{1-6}$$

Equation (1-6) can be used as a sampling operator. Multiplication of a continuous function x(t) by this sampling operator results in a sampled function $x_d(t)$ represented by a sequence of inpulses weighted by the values of the continuous function at $t_n = n\Delta t$. This sampled sequence is

$$S_M(t)\ x(t) = x(t) \sum_{n=-M}^{M} \delta(t - n\Delta t)$$

or

$$X_d(t) = \sum_{n=-M}^{M} x(n\Delta t)\ \delta(t - n\Delta t) \tag{1-7}$$

By substituting equation (1-7), the discrete operator, into (1-1) for the continuous function, the finite Fourier transform can be defined

$$F_d(\omega) = \int_{-\infty}^{\infty} X_d(t) e^{-i\omega t} dt \tag{1-8}$$

$$F_d(\omega) = \sum_{n=-M}^{M} x(n\Delta t) \int_{-\infty}^{\infty} \delta(t - n\Delta t) e^{-i\omega t} dt \tag{1-9}$$

From the definition of the delta function it can then be shown (Papoulis, page 43, 1962) that

$$F_d(\omega) = \sum_{n=-M}^{M} x(n\Delta t)e^{-i\omega n\Delta t} \qquad (1\text{-}10)$$

An important feature of equation (1-10) is that the transform $F_d(\omega)$ is periodic.

Let $\Omega = 2\pi/\Delta t$

$$F_d(\omega + k\Omega) = \sum_{n=-M}^{M} x(n\Delta t)e^{-i(\omega + k\Omega)n\Delta t}$$

$$= \sum_{n=-M}^{M} x(n\Delta t)e^{-i\omega n\Delta t}\, e^{-ik2\pi n}$$

Since n and k are integers, $e^{-ik2\pi n} = 1$,

$$F_d(\omega + k\Omega) = \sum_{n=-M}^{M} x(n\Delta t)e^{-i\omega n\Delta t}, \text{ and}$$

$$F_d(\omega + k\Omega) = F_d(\omega).$$

Hence, the period of the finite Fourier transform is $\Omega = 2\pi/\Delta t$, which is called the angular sampling frequency.

In equation (1-10) the problem of calculating a frequency function $F(\omega)$ from the continuous function $x(t)$, is replaced by the problem of calculating the periodic function $F_d(\omega)$ from the sequence $x(n\Delta t)$. If $x(t)$ is not truncated by sampling, the periodic function is constructed by adding the set of functions $F(\omega + k\Omega)$ for $k = 0, \pm 1, \pm 2$, etc. Therefore the error in approximating $F(\omega)$ by $F_d(\omega)$ in the range $-\frac{1}{2}\Omega<\omega<\frac{1}{2}\Omega$ is the sum of the $F(\omega+k\Omega)$'s for $k \neq 0$. Thus, by choosing Δt small enough, $\Omega = \frac{2\pi}{\Delta t}$ can be made sufficiently large so that this error is negligible in the frequency range of interest.

TRANSFORM ALGORITHMS

This chapter describes seven algorithms to be compared.

The first algorithm (FT1) is a direct application of the discrete Fourier transform

$$F_d(\omega) = \sum_{n=-M}^{M} x(n\Delta t)e^{-in\omega\Delta t} \qquad (1\text{-}10)$$

since $e^{-iy} = \cos y - i \sin y$, (1-10) becomes

$$F_d(\omega) = \sum_{n=-M}^{M} \left[x(n\Delta t) \left[\cos (\omega n\Delta t) - i \sin (\omega n\Delta t) \right] \right] \qquad (2\text{-}1)$$

or to determine the individual coefficients of the expansion, let $X(\omega)$ = real part of the transform, and $Y(\omega)$ = - imaginary part of the transform. Then (2-1) would become

$$X(\omega) = \sum_{n=-M}^{M} x(n\Delta t) \cos (\omega n\Delta t) \qquad (2\text{-}2)$$

and

$$Y(\omega) = \sum_{n=-M}^{M} x(n\Delta t) \sin (\omega n\Delta t) \qquad (2\text{-}3)$$

from which the Fourier spectrum $A(\omega)$ can be computed,

$$A(\omega) = \sqrt{X^2(\omega) + Y^2(\omega)} \qquad (2\text{-}4)$$

and the phase angle $\Phi(\omega)$.

$$\Phi(\omega) = \tan^{-1}\left[-\frac{Y(\omega)}{X(\omega)}\right] \tag{2-5}$$

If there are a total of 2M + 1 points to describe the function x(t), (n ranging from M to -M), then for the algorithm FT1 there will be 2M summations over 2M multiplications to describe each frequency component of the real and imaginary parts of the transform; that is, 8M computations for each value of the frequency.

The second algorithm (FT2) employs an iterative technique to determine the sine and cosine values of equations (2-2) and (2-3). An initial value for the sine and cosine is set; if $\omega_1 = \Delta\omega$, then $\sin\theta = \sin(\omega_1\Delta t)$ and $\cos\theta = \cos(\omega_1\Delta t)$. Using the recurrence relations

$$\sin(j+1)\theta = \sin j\theta\cos\theta + \cos j\theta\sin\theta \tag{2-6}$$

for $j = 1, 2, \ldots, M$,

$$\text{and } \cos(j+1)\theta = \cos j\theta\cos\theta - \sin j\theta\sin\theta \tag{2-7}$$

for $j = 0, 1, \ldots, M$,

one can compute

$$X(\omega) = \sum_{n=-M}^{M} x(n\Delta t)\cos(\omega n\Delta t) \tag{2-8}$$

and

$$Y(\omega) = \sum_{n=-M}^{M} x(n\Delta t)\sin(\omega n\Delta t) \tag{2-9}$$

using library sub-programs for $\sin\theta$ and $\cos\theta$ only.

Once again there will be 2M summations over 2M multiplications, but this time the recurrence formulas are used to speed up the calculation.

G. Goertzel (Goertzel, 1958) developed the third algorithm (FT3). In this algorithm the calculation of the coefficients of the discrete Fourier transform (2-1) is accomplished on a more economical basis than either of the two preceding algorithms (FT1) or (FT2).

Once again, starting with the sums in equations (2-2) and (2-3), Goertzel has pointed out that one multiplication for each summation can be eliminated by employing the following recursive technique.

Let $$U_{2M+2} = 0 \tag{2-10}$$

$$U_{2M+1} = 0 \tag{2-11}$$

and $$U_m = (2 \cos \omega\Delta t)U_{m+1} - U_{m+2} + x(m) \tag{2-12}$$

where $m = (2M, 2M - 1, . . . , 1)$.

It has been shown (Hamming, 1962) how a recursive technique for the trigonometric functions through the use of the above relations can reduce the number of calculations for the transformation, so that

$$X(\omega) = \sum_{n=-M}^{M} x(n\Delta t) \cos (\omega n\Delta t) \tag{2-2}$$

becomes

$$X(\omega) = x(0) + (2 \cos n\omega\Delta t)U_1 - U_2; \tag{2-13}$$

and

$$Y(\omega) = \sum_{n=-M}^{M} x(n\Delta t) \sin (\omega n\Delta t) \tag{2-3}$$

becomes

$$Y(\omega) = U_1 \sin (\omega n\Delta t). \tag{2-14}$$

A proof of the above can be found in (Goertzel, 1958).

The same recursion relation for the trigonometric functions as used in FT2 can be employed here, so that if

$$C_n = \cos \omega n \Delta t$$

and

$$S_n = \sin \omega n \Delta t$$

then

$$\begin{bmatrix} C_{n+1} \\ S_{n+1} \end{bmatrix} = \begin{bmatrix} C_1 & -S_1 \\ S_1 & C_1 \end{bmatrix} \cdot \begin{bmatrix} C_n \\ S_n \end{bmatrix}$$

resulting in algorithm FT3 being

$$X(\omega) = x(0) + 2C_n U_1 - U_2 \qquad (2\text{-}15)$$

and

$$Y(\omega) = S_n U_1.$$

The number of calculations for this algorithm is less than 2M multiplications and 3M additions or 3M multiplications and 2M additions for each pair of complex parts per value of frequency.

The fourth algorithm (FT4) is basically the same algorithm as (FT1) with the exception that the values of the sine and cosine for any expected increments of the arguments, $n\omega\Delta t$, are calculated and placed in a table for future reference.

It is obvious that the speed of this algorithm after the initial calculation is dependent only on the access time of the computer being used. The major disadvantage of this algorithm lies in the large amount of computer memory that would be required to store the trigonometric tables.

Another recursive technique for determining sine and cosine is employed in the fifth algorithm (FT5). This recursive technique is as follows:

$$\cos (n + 1)\theta = 2 \cos\theta \cos n\theta - \cos (n - 1)\theta \qquad (2\text{-}17)$$

$$\sin (n + 1)\theta = 2 \cos\theta \sin n\theta - \sin (n - 1)\theta \qquad (2\text{-}18)$$

Therefore, the real and imaginary parts of the transform are as stated in equations (2-2) and (2-3)

$$X(\omega) = \sum_{n=-M}^{M} x(n\Delta t) \cos (n\omega\Delta t) \qquad (2\text{-}19)$$

and

$$Y(\omega) = \sum_{n=-M}^{M} x(n\Delta t) \sin (n\omega\Delta t) \qquad (2\text{-}20)$$

Once again the number of operations are 2M summations over 2M multiplications, but the speed comes in requiring only one calculation of sine and cosine.

The sixth algorithm is the z transform (FT6).

Let

$$z = e^{i\omega\Delta t} \quad (2\text{-}21)$$

then

$$F_d(\omega) = \sum_{n=-M}^{M} x(n\Delta t)e^{-in\omega\Delta t} \quad (1\text{-}11)$$

becomes

$$F_d(\omega) = \sum_{n=-M}^{M} x(n\Delta t)z^{-n} \quad (2\text{-}22)$$

and the algorithm would be as follows.

Let

$$z = \text{Complex value} = (x, y)$$

$$z = [\cos(\omega\Delta t); \sin(\omega\Delta t)] \quad (2\text{-}23)$$

then

$$X(\omega) = \sum_{n=-M}^{M} \text{Real } (Z)^n \; x(n\Delta t) \quad (2\text{-}24)$$

and

$$Y(\omega) = \sum_{n=-M}^{M} \text{Imaginary } (Z)^n \; x(n\Delta t)$$

The total calculations will be 2M summations over 2M multiplications, with an increase in speed of calculation coming from only one calculation of sine and cosine being necessary.

The final algorithm to be compared (FT7) is the Fast Fourier Transform developed by J. W. Cooley and J. W. Tukey (Cooley and Tukey, 1965). There has been a change in notation for this algorithm as compared to the previous six algorithms. The author felt this was necessary to develop the algorithm in such a manner as to show how the speedup is accomplished. More specifically, if a time sequence x(t) contains $N = 2^M$ samples, (0, 1, . . . , N), then the number of operations to accomplish the transform using this algorithm is about $2MN = 2N \log_2 N$. Let the time sequence x(t) be divided into two functions, a(t) and b(t), each having only half of the total N points. Also, let a(t) be composed of the even-numbered points, and b(t) be composed of the odd-numbered points.

$$a_n(t) = x(2n\Delta t) \qquad (2\text{-}27)$$

$$n = 0, 1, \ldots, \frac{N}{2} - 1$$

$$b_n(t) = x\left[(2n + 1)\Delta t\right] \qquad (2\text{-}28)$$

Corresponding to these two functions, define two functions that have Fourier transforms given by:

$$E(\omega) = \sum_{n=0}^{\frac{N}{2}-1} a_n(t)e^{-i2n\omega\Delta t} \qquad (2\text{-}29)$$

and

$$O(\omega) = \sum_{n=0}^{\frac{N}{2}-1} b_n(t)e^{-i2n\omega\Delta t} \qquad (2\text{-}30)$$

The transform for the entire N points in terms of the odd- and even-numbered points would be,

$$F_d(\omega) = \sum_{n=0}^{\frac{N}{2}-1} \left[a_n(t)e^{-i2n\omega\Delta t} + b_n(t)e^{-i\omega(2n + 1)\Delta t}\right]$$

or

$$F_d(\omega) = \sum_{n=0}^{\frac{N}{2}-1} x(2n\Delta t)e^{-i2n\omega\Delta t} + \sum_{n=0}^{\frac{N}{2}-1} x\left[(2n + 1)\Delta t\right]e^{-i\omega(2n+1)\Delta t}$$

resulting in

$$F_d(\omega) = \sum_{n=0}^{\frac{N}{2}-1} x(2n\Delta t)e^{-i2n\omega\Delta t} + e^{-i\omega} \sum_{n=0}^{\frac{N}{2}-1} x\left[(2n + 1)\Delta t\right]e^{-i2n\Delta t}$$

which in terms of equations (2-29) and (2-30) would be

$$F_d(\omega) = E(\omega) + e^{-i\omega}O(\omega).$$

Therefore it has been demonstrated that the transform of the time sequence of N points can be computed by reducing the sequence to two sequences of $\frac{N}{2}$ points. The computation of $E(\omega)$ and $O(\omega)$ can be reduced to the computation of sequences of $\frac{N}{4}$ points. These reductions are possible as long as the number of samples is divisible by 2. (Cochran, Cooley, and others, 1967). There are $\frac{N}{2}$ complex multiplications to go from two $\frac{N}{2}$ point analysis to an N point analysis. So that if N is the M^{th} power of two, $N = 2^M$, then M successive doublings would yield the N point analysis. There are two complex additions for each multiplication and the total calculations for the algorithm are less than $N \log_2 N$ operations, resulting in this algorithm having fewer multiplications than any other algorithm.

It should be pointed out that N does not necessarily have to be a power of two, but can be any composite number, such as $N = r_1 \cdot r_2 \, . \, . \, . \, r_m$, then m different subsequences of N are formed to make up the transform, yielding an algorithm with a total number of operations being $N(r_1 + r_2 + . . . + r_m)$. If all r_j $(j = 1, 2, . . , m)$ are equal, the total number of operations become $rN \log_r N$. Cooley and Tukey have determined that the optimum conditions exist when $r = 2, 3,$ or 4.

A simple example helps to clarify this algorithm. Such an example can be found in Cochran (Cochran, Cooley, and Others, 1967).

EXPERIMENTAL DATA

Test Functions

There were two different functions used to test the algorithms. The first function

Function 1 =

$$f_1(t) = kt^{T_o Q} e^{-Qt} \quad \text{for } t>0 \qquad (3\text{-}1)$$

$$f(t) = 0 \quad \text{for } t \le 0$$

where

k = scale factor which normalizes $f(t)$ to 1.0

T_o = 20.0 (the time (t) at which f(t) is maximum)

Q = .18203921 (a value used to insure convergence approaching zero near a time of 100.0)

were chosen because of the small effect of truncation in the range calculated. Figure 3.1 represents equation (3-1).

T-1179

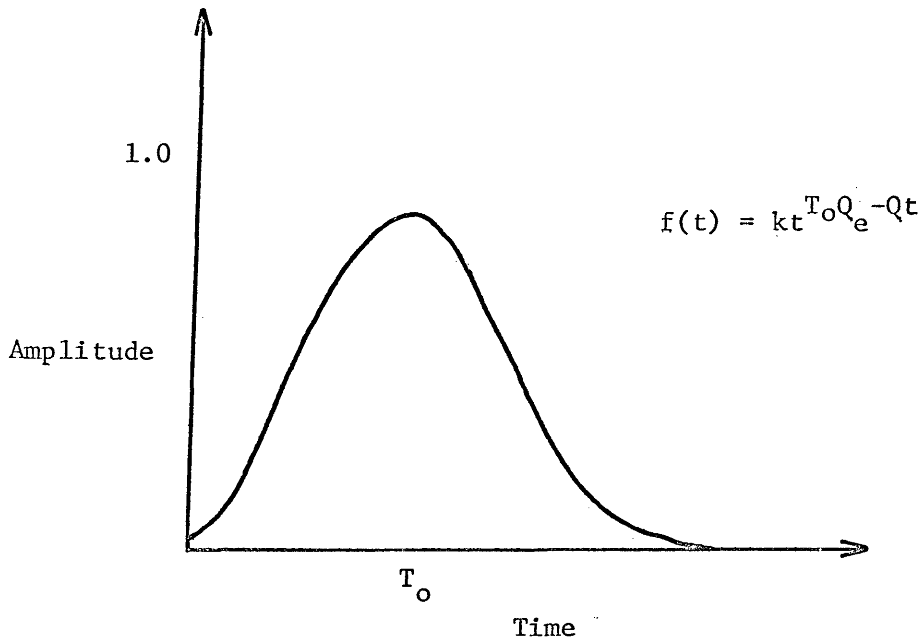

Figure 3.1

Function 2 =

$$f_2(t) = \text{Sinc}^2 t = \left[\frac{\sin \pi t}{\pi t}\right]^2 \qquad (3\text{-}2)$$

Since function 2, equation (3-2), is an even function, only values for $t \geq 0$ were used to calculate the transform. Figure 3.2 represents equation (3-2). The error between the discrete transform and the closed form transform as illustrated in Figure 3.4 represents the truncation effects of limiting the range of the function, from $t = 0$ to $t = M$. ($M = 129$ points as stated on page 22 and figure 3.4.)

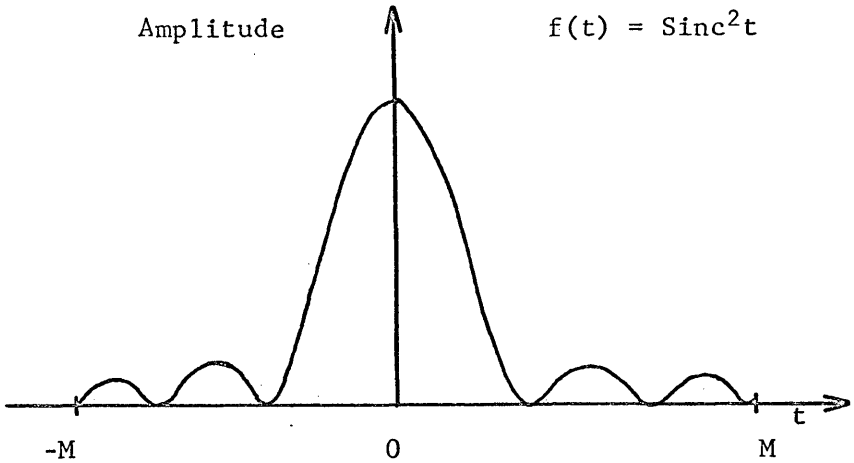

Figure 3.2

Data Display

Two samples of the test data computed that were plotted on a Calcomp Digital Incremental Plotter are displayed in figures 3.3 and 3.4. The printed results from which these plots were made appear in Appendix B, along with the input values for the two functions. Due to the overabundance of data gathered for all seven algorithms, and the redundancy of the plots, only two plots are displayed. The printed results for only three of the algorithms were chosen to be listed in this thesis: FT2, FT4, and FT6. These three were chosen because they represent the three basic groupings of the algorithms, excluding the fast Fourier transform (FT7).

T-1179

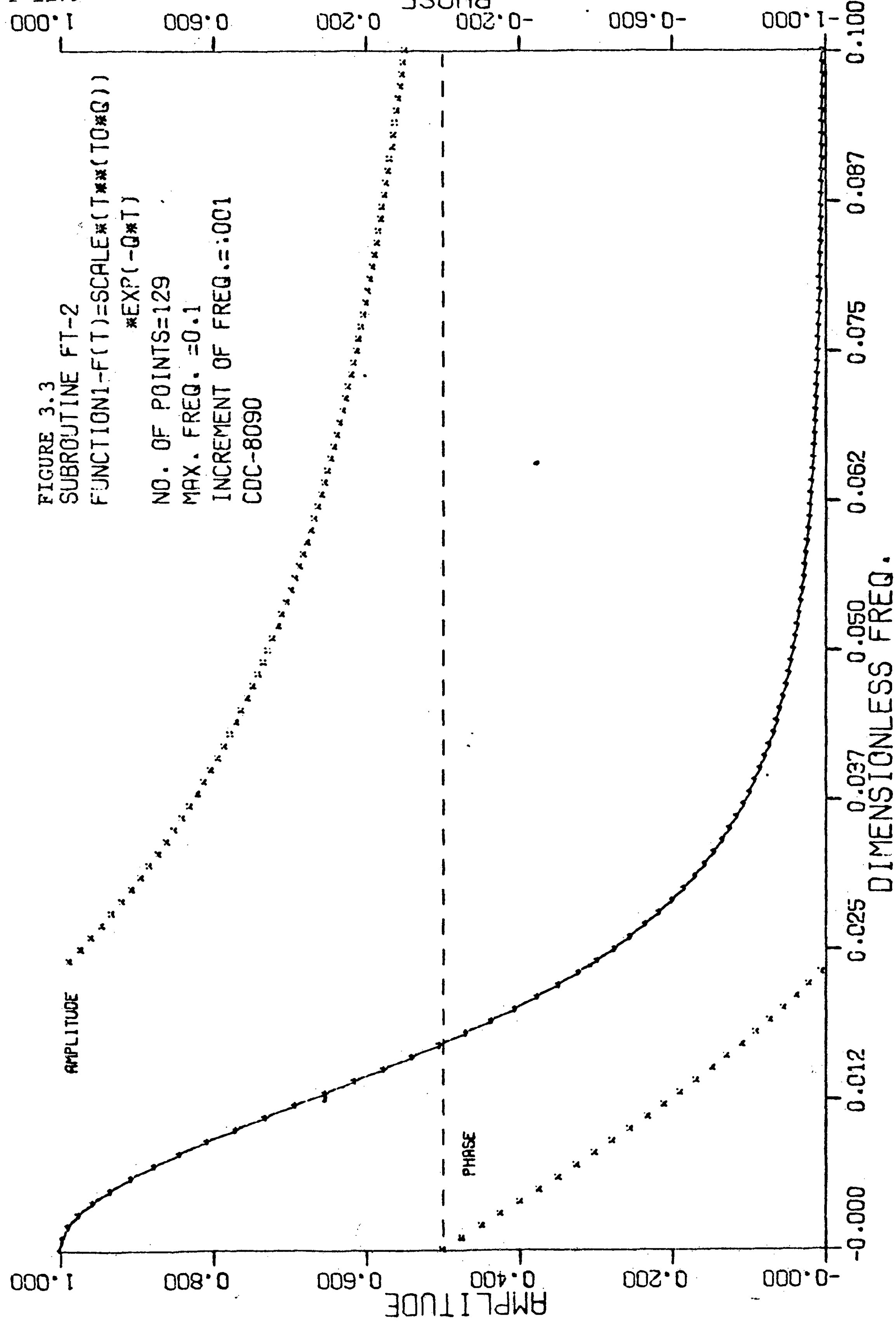
FIGURE 3.3
SUBROUTINE FT-2
FUNCTION1-F(T)=SCALE*(T**(TO*G))
*EXP(-Q*T)
NO. OF POINTS=129
MAX. FREQ. =0.1
INCREMENT OF FREQ.=.001
CDC-8090
PHASE
1.000
0.600
0.200
-0.200
-0.600
-1.000
AMPLITUDE
1.000
0.800
0.600
0.400
0.200
-0.000
-0.000
0.012
0.025
0.037
0.050
0.062
0.075
0.087
0.100
DIMENSIONLESS FREQ.

T-1179

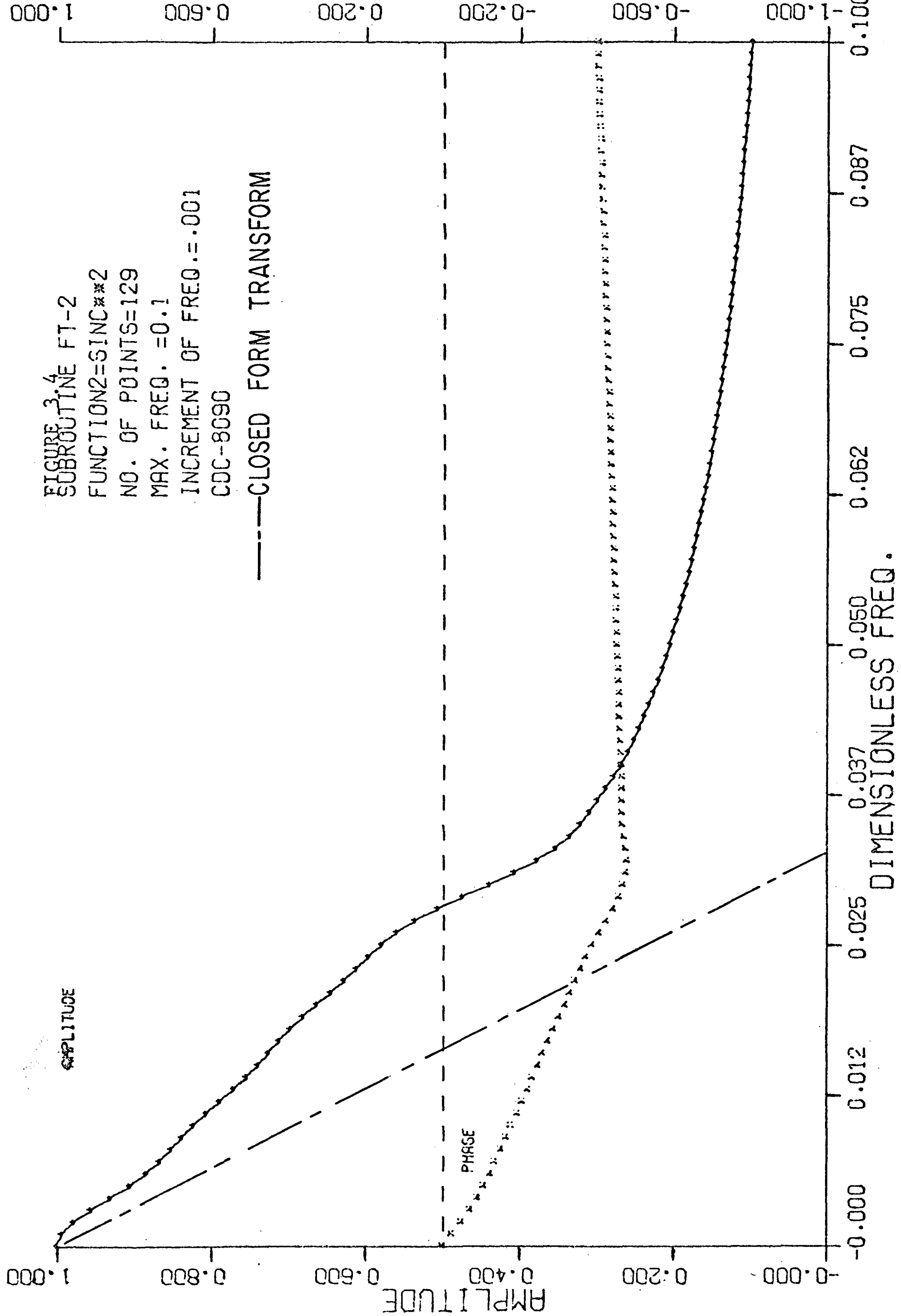
FIGURE 3.4
SUBROUTINE FT-2
FUNCTION2=SINC**2
NO. OF POINTS=129
MAX. FREQ. =0.1
INCREMENT OF FREQ.=.001
CDC-6090
CLOSED FORM TRANSFORM
AMPLITUDE
PHASE
DIMENSIONLESS FREQ.
-0.000
0.012
0.025
0.037
0.050
0.062
0.075
0.087
0.100
1.000
0.800
0.600
0.400
0.200
-0.000
0.600
0.200
-0.200
-0.600
-1.000

The amplitude axes of the figures have been normalized to one, while the phase axes have been plotted on a scale of $-\pi \leq \phi \leq \pi$ for the CDC-6400 computer and a normalized scale of $-1 \leq \phi \leq 1$ for the CDC-8090 computer.

The frequency axes of the figures were plotted in dimensionless frequency $f\Delta t$. This is a convenient unit for discrete transforms. To see this, use equation (2-1).

$$F_d(\omega) = \sum_{n=-M}^{M} \left[x(n\Delta t)\left[\cos(\omega n\Delta t) - i \sin(\omega n\Delta t)\right]\right] \quad (2\text{-}1)$$

To calculate this discrete transform on the computer, $F_d(\omega)$ must be calculated for discrete values of ω; therefore, (2-1) would become

$$F_d(k\Delta\omega) = \sum_{n=-M}^{M} \left[x(n\Delta t)\left[\cos(nk\Delta\omega\Delta t) - i \sin(nk\Delta\omega\Delta t)\right]\right] \quad (3\text{-}3)$$

$$k = 0, 1, \ldots, N.$$

Equation (3-3) is a function of the variable $k\Delta\omega\Delta t$. Upon doing the transform $\Delta\omega\Delta t$ is fixed and, by specifying either $\Delta\omega$ or Δt, the other is determined. The maximum N necessary is $N = 0.5/\Delta f\Delta t$, where $\Delta f = \frac{\Delta\omega}{2\pi}$, because the Nyquist frequency (ω_c) occurs at $N\Delta f\Delta t = 0.5$ on the dimensionless scale.

$$\omega_c = \frac{\pi}{\Delta t} = 2\pi f_c \quad (3\text{-}4)$$

$$f_c(\text{maximum } k\Delta f) = \frac{1}{2\Delta t} \quad (3\text{-}5)$$

therefore

$$N \Delta f \Delta t = 0.5 \qquad (3\text{-}6)$$

Truncation and Sampling

In order to show the effect that truncation can have on the result and so that this effect is not misinterpreted to be due strictly to approximating the transform, the two test functions were calculated for 81 and 129 points, for each of the seven algorithms.

It was not the intent of this thesis to study the effect of sampling, therefore, a sampling rate was used that would insure that the effects of aliasing would not be encountered.

The sampling rate for function 1 is $\Delta t = 1.0$. The sampling rate for function 2 is $\Delta t = .05$ for a total of 81 points, and $\Delta t = .03$ for a total of 129 points. The value of N in equation (3-6) for all examples run is 100.

For function 1

$$\Delta f = .001$$

$$\Delta t = 1.0.$$

Therefore the aliasing frequency would occur when

$$N = 0.5/\Delta f \Delta t = 5 \times 10^2.$$

For function 2

$$\Delta f = .001$$

$$\Delta t = .05$$

Therefore

$$N = 0.5/\Delta f \Delta t = 1 \times 10^4$$

and

$$\Delta f = .001$$

$$\Delta t = .03,$$

therefore

$$N = 0.5/\Delta f \Delta t = 1.67 \times 10^4.$$

It should be pointed out that for the transformations accomplished in this thesis, there would be two types of truncation because of the finite Fourier transform. The first would be truncation due to the approximation of the transform or the shrinking of the domain of the function. The second type of truncation would be due to the sampling of the function in the domain described. A more exacting treatment of sampling can be found in Bracewell (Bracewell, 1965) and Papoulis (Papoulis, 1962).

ERROR COMPARISON

This chapter is an evaluation of the error of the algorithm. Since all the algorithms are discrete calculations on a computer, there is not only a possible error due to the individual algorithm involved, but also there can be error due to the configuration of the computing machine used and the internal subroutines of the system that are necessarily used by the algorithms. These would include the trigonometric function and complex arithmetic that are available on most library systems of a computer.

Approximation Error

The discrete Fourier Transform (1-11) is a result of the Fourier Series.

$$x(t) = \frac{A_o}{2} + \sum_{k=1}^{\infty}(a_k \cos kt + b_k \sin kt) \qquad (4-1)$$

To make this series discrete, consider the following set of 2N functions, 1, cos t, . . . , cos Nt; sin t, sin 2t, . . . , sin (N-1)t. It can be shown (Hamming, 1962; Ralston, 1965, Lanczos, 1956) that these functions are orthogonal over the discrete set of points $t_i = \pi\frac{i}{N}$, $i = 0, \ldots, 2N - 1$.

The orthogonality relations are,

$$\sum_{t=0}^{2N-1} \sin \frac{\pi}{N} kt \sin \frac{\pi}{N} mt = \begin{cases} 0 \text{ if } k \neq m \\ N \text{ if } k = m \neq 0 \end{cases}$$

$$\sum_{t=0}^{2N-1} \sin \frac{\pi}{N} kt \cos \frac{\pi}{N} mt = 0 \qquad (4\text{-}2)$$

$$\sum_{t=0}^{2N-1} \cos \frac{\pi}{N} kt \cos \frac{\pi}{N} mt = \begin{cases} 0 \quad \text{if } k \neq m \\ N \quad \text{if } k = m \neq 0, N, \ldots \\ 2N \text{ if } k = m = 0, N, \ldots \end{cases}$$

By using 2N functions an exact fit can be made on the 2N points. But one reason why Fourier analysis is such a powerful tool is the fact that it accomplishes a smoothing in the whole region and by using all 2N functions this is not accomplished. Therefore, an approximation of the function x(t) is made by using $M<N-1$ terms. Then equation (4-1) becomes

$$X_M(t) = \frac{a_o}{2} + \sum_{k=1}^{M} (a_k \cos \frac{\pi}{N} kt + b_k \sin \frac{\pi}{N} kt) \qquad (4\text{-}3)$$

Then the a_k's and b_k's are determined so that the sum of the squares of the differences between the function and the approximated function is a minimum,

$$[x(t) - x_M(t)]^2 = \text{minimum} \qquad (4\text{-}4)$$

It is therefore evident that a general error for all the algorithms is a function of the number M of harmonics calculated. A point to be considered in determining M is that, due to sampling at even intervals, no frequency greater than the Nyquist frequency can be calculated. Therefore, the largest value of M needed to be considered can be determined as follows:

N = number of points of the function x(t) available

ω_c = Nyquist frequency

Δt = sampling interval

$$\frac{M\pi t}{N} = 2\pi\omega_c t \qquad (4\text{-}5)$$

$$M = 2\omega_c N = 2\omega_c n\Delta t \qquad (4\text{-}6)$$

$$t = 0, \Delta t, \ldots, n\Delta t$$

This value of M is not necessarily the one to use because $n \leq M$ is a function of the smoothing. That is, from (4-6)

$$\frac{M}{n} = 2\omega_c \Delta t \qquad (4\text{-}7)$$

Let the reciprocal of (4-7) be a smoothing parameter p.

$$p = 1/2\ \omega_c \Delta t \qquad (4\text{-}8)$$

and in order to smooth

$$p \geq 1. \qquad (4\text{-}9)$$

Algorithm Error

The algorithms can basically be put into four groups for error analysis. The first group, consisting of FT1 and FT4, will have an error that is a function of the inherent error in the computing system used plus the approximation error mentioned above. This error also includes error due to methods used to calculate the trigonometric functions. The computing system errors are covered in the next section of this chapter. Let this error be represented by E_1. Since there is a total of 8N calculations for each frequency for these two algorithms, the total error possible would be $8NE_1$.

The second group consists of the iterative algorithms, FT2, FT3, and FT5. Algorithms FT2 and FT5 have 8N calculations, but they have only one initial calculation of the trigonometric functions sine and cosine. Therefore, the error for this group E_2 is different from the preceding group, and the total error for FT2 and FT5 will be $8NE_2$ for each value frequency. FT3 has only 6N calculations with only one calculation for the trigonometric functions sine and cosine; therefore, the total error for FT3 would be $6NE_2$ for each value of frequency.

The third group includes the algorithm FT6 which has a total number of 2N calculations for the complex pair for each value of frequency. The error involved here is a function of the computer systems routine for complex arithmetic. The total error for the algorithm would then be $2NE_3$.

The last group contains the Fast Fourier Transform FT7 and will have an error E_4 log N.

Machine Error

The machines used to calculate data for this thesis were the Control Data Corporation 8090 and the Control Data Corporation 6400. The test data for the Cooley - Tukey algorithm, FT7, was run on a Scientific Data System 9300 computer. The three machines were used to get a comparison of the effect of word length on the calculations.

The CDC 8090 has a 27-bit mantissa that results in 8 significant figures. Accuracy should be placed at 7 figures because of the effect of round-off. The CDC 8090 does a truncation of least significant figures in all arithmetic operations therefore resulting in the last digit (8th) being questionable.

The CDC 6400 has an effective 48-bit mantissa for its 60-bit word resulting in about 14 significant digits. This computer has the capability of rounding off the least significant digit, thereby giving a more consistent accuracy at the 14th digit.

The accuracy of the library functions for calculating the sine, cosine, square root, etc., on the CDC 8090 is within the 7-digit accuracy of the computations.

The accuracy of the library functions on the CDC 6400, including the above-mentioned subroutines plus the subroutines to do complex arithmetic, is well within the 14-digit accuracy of the computations.

RESULTS AND CONCLUSIONS

In this chapter the results and conclusions of the experimental runs are evaluated. The speed of the algorithms involved is a function of the speed of the computer used. Therefore, a relative speed comparison between the algorithms which will apply generally to any machine is given here.

By comparing the results between the algorithms, the effects due to word lengths of the computers and the effects of truncation of the input data can be analyzed.

Finally, the evaluation of the comparisons of times and errors leads to conclusions to suggest which algorithm should be used when.

Time Results

The time results can be placed into groups just as the error results in the previous chapters were.

The first group which includes algorithms FT1 and FT4 is twice as slow as the second group which includes the algorithms FT2, FT3, FT5, and FT6. The seventh algorithm, FT7, is the fastest of all of the algorithms tested.

It should be pointed out here that relative speeds between any two algorithms can be increased by the proper use of computer "hardware." An example of this is one in which algorithm FT1, using a convolver apparatus (hardware designed for very rapid multiplication and addition), can approach the speed of algorithm FT7 for smaller numbers of total points evaluated.

A comparative table for the algorithms FT1 through FT6 on the CDC 8090 and CDC 6400 computers is given in the following table.

ALGORITHMAL TIME COMPARISONS
ON THE CDC 8090 AND CDC 6400

(Time in Minutes)

ALGORITHM	CDC 8090	CDC 6400
FT1	10.8	0.067
FT2	4.5	0.034
FT3	3.6	0.032
FT4	10.8	0.068
FT5	4.5	0.034
FT6	---	0.034

FT6 could not be run on the CDC 8090 due to the lack of complex arithmetic capabilities of that machine.

A time comparison of FT7 and FT1 has been run on the SDS 9300 by W. T. Prescott (Prescott, 1967) with the following results as shown. The algorithm FT1 was programmed in two different ways: 1) fixed point arithmetic in which a convolver unit was employed, and 2) floating point arithmetic, without the use of the convolver unit.

	Time in Seconds		
Total Number of Points Analyzed	Fixed Point (Convolver Used)	Floating Point (Convolver Not Used)	FT7
64	0.18	0.32	0.10
128	0.48	1.12	0.23
256	1.42	4.28	0.52
512	4.72	16.63	1.10
1,024	16.40	65.55	2.38
2,048	61.53	260.37	5.13

Error Results

The data calculated in all of the algorithms was output in real and imaginary values and amplitude and phase. The amplitude values were normalized to 1.0; therefore, an indication of accuracy is the comparison in decimal places between the algorithms.

On the CDC 8090 the error between algorithms is in the sixth decimal place. Error due to truncation of function 1 is in the third decimal place, while error due to truncation of function 2 is in the first decimal place.

On the CDC 6400 the error between algorithms is beyond the eight decimal places printed out. The error due to truncation of function 1 is in the third decimal place. The error due to truncation of function 2 is in the first decimal place.

The effect of word length of computers would be indicated by the discrepancy of output data between the two machines. It was observed that the error between the two machines was in the seventh significant figure due to the accuracy of the CDC 8090, which has a roundoff error in the eighth significant figure.

Conclusions

Because of the advances being made in the field of digital computers today and especially the ever-increasing size of the memories of these machines, the emphasis of this thesis has been on time of execution and errors between the stated algorithms.

The errors due to truncation have been pointed out and are independent of the individual algorithms. All algorithms agree to within 8 digits, therefore, the only basis for comparison is in relative time of execution with the following order indicating the preference of the algorithms: FT7, FT3, FT6, FT2 and FT5, FT1, and FT4.

APPENDIX I

FORTRAN PROGRAMS

The following is a list of the Fortran programs for the algorithms. The algorithms FT1 through FT5 were written in Fortran II which could be run on either the CDC 8090 or the CDC 6400. FT6 was necessarily required to be written in Fortran IV and, therefore, was run only on the CDC 6400. Algorithm FT7 was written in Fortran IV and is the program FORT as listed in the IBM SHARE program.

In each case only the subroutines for calculating the real and imaginary values of the transform are given. The amplitude and phase values were calculated in a subroutine labeled MOPH which is the last subroutine listed.

```
      SUBROUTINE FT1 (X, Y, NXY, AA, NA, DELFT, FTMAX)
      DIMENSION X(145), Y(145), AA(401)
      NXY = (FTMAX/DELFT)+1.0
      AN=NA
      DO 20 I=1, 145
      X(I) = 0
   20 Y(I) = 0
      ANUM = 6.28318530*DELFT
c
c     FREQ. LOOP
c
      DO 1 J=1, NXY
      FJ=J-1
      ANUM1=FJ*ANUM
c
c     TIME LOOP
c
      DO 1 K=1, NA
      FK=K-1
      X(J)=AA(K)*COS(ANUM1*FK)+X(J)
      Y(J)=-(AA(K)*SIN(ANUM1*FK))+Y(J)
    1 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FT2 (X, Y, NXY, AA, NA, DELFT, FTMAX)
      DIMENSION X(145), Y(145), AA(401)
      NXY = (FTMAX/DELFT)+1.0
      ANUM = 6.28318530*DELFT
      X(1)=AA(1)
      Y(1)=0
c
c     FREQ. LOOP
c
      DO 1 J=1, NXY
      X(J)=AA(1)
      Y(J)=0.0
      FJ=J-1
      S1=SINF(ANUM*FJ)
      C1=COSF(ANUM*FJ)
      C2=C1
      S2=S1
c
c     TIME LOOP
c
      DO 1 K=2, NA
      X(J)=X(J)+AA(K)*C2
      Y(J)=-AA(K)*S2+Y(J)
      S3=S2*C1+C2*S1
      C3=C2*C1-S2*S1
      S2=S3
      C2=C3
    1 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FT3 (X, Y, NXY, AA, NA, DELFT, FTMAX)
      DIMENSION X(145), Y(145), AA(401)
      DOUBLE PRECISION ARGI
      NXY=(FTMAX/DELFT)+1.0
      INC=-1
      A1=AA(1)
      DO 20 I=1, 145
      X(I)=0.0
   20 Y(I)=0.0
      ARGI=6.28318530717958647692528 7DO*DBLE(DELFT)
      C=COS(ARGI)
      S=SIN(ARGI)
      C1=1.0
      S1=0.0
c
c     FREQ.LOOP
c
      DO 2 N=1, NXY
      U1=0.0
      U2=0.0
      SCALE= 2.*C1
c
c     TIME LOOP
c
      DO 1 I=2, NA
      NA1=NA+2
      K=NA1-I
      BA=AA(K)
      UO=BA+SCALE*UI-U2
      U2=U1
    1 U1=UO
      X(N)=A1+C1*U1-U2
      Y(N)=-S1*U1
      Q=C*C1-S*S1
      S1=C*S1+S*C1
      C1=Q
    2 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FT4 (X, Y, C, S, NXY, AA, NA, DELFT, FTMAX)
      DIMENSION X(145), Y(145), C(145), S(145), AA(145)
      NXY=(FTMAX/DELFT)+1.0
      DO 20 I=1,145
      X(I)=0
      Y(I)=0
      C(I)=0.0
   20 S(I)=0.0
      ANUM=6.28318530*DELFT
      DO 1 J=1,NXY
      AJ=J-1
      ANUM1=ANUM*AJ
      DO 1 K=1,NA
      AK=K-1
      C(J)=COSF(ANUM1*AK)
      S(J)=SINF(ANUM1*AK)
      X(J)=AA(K)*C(J)+X(J)
      Y(J)=-(AA(K)*S(J))+Y(J)
    1 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FT5 (X, Y, NXY, AA, NA, DELFT, FTMAX)
      DIMENSION X(145), Y(145), AA(401)
      DOUBLE PRECISION ANUM, ARG
      NXY=(FTMAX/DELFT)+1.0
      X(1)=0.0
      DO 20 I=1,NA
   20 X(1)=X(1)+AA(I)
      Y(1)=0.0
      ANUM=6.28318530717958647692528 7DO*DBLE(DELFT)
C
C     FREQ.LOOP
C
      ARG=0.0
      DO 2 J=2,NXY
      X(J)=0.0
      Y(J)=0.0
      ARG=ARG+ANUM
      CSAVE=SNGL(DCOS(ARG))
      SSAVE=SNGL(DSIN(ARG))
      C1=2,*CSAVE
      C2-1.0
      S2=0.0
C
C     TIME LOOP
C
      DO I K=1,NA
      X(J)=AA(K)*C2+X(J)
      Y(J)=Y(J)+(AA(K)*S2)
      CAVE=C2
      SAVE=S2
      C2=C2*C1-CSAVE
      S2=S2*C1-SSAVE
      CSAVE=CAVE
      SSAVE=SAVE
    1 CONTINUE
    2 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE FT6 (X, Y, NXY, AA, NA, DELFT, FTMAX)
      DIMENSION X(145), Y(145), AA(145)
      TYPE COMPLEX Z, ZO, Z1, Z2, Z4, ZERO, ZREF
      DOUBLE PRECISION TWOPI, ARG
      DATA (ZERO=(0.,0.)), (ZO=(1.,0.))
      DATA (TWOPI=6.2831853071798647692528 7DO)
      NXY=(FTMAX/DELFT)+1.0
      ARG=-(TWOPI*DBLE(DELFT))
      Z4=CMPLX(SINGL(DCOS(ARG)),SNGL(DSIN(ARG)))
C
C     FREQUENCY LOOP
C
      Z2=ZO
      DO 1 IZ=1,NXY
      Z1=ZO
      Z=ZERO
C
C     TIME LOOP
C
      DO=2 IS=1,NA
      Z=Z+Z1*AA(IS)
    2 Z1=Z1*Z2
      Z2=Z2*Z4
      X(IZ)=REAL(Z)
      Y(IZ)=AIMAG(Z)
    1 CONTINUE
      RETURN
      END
```

```
      SUBROUTINE MOPH (X, Y, NXY, DELFT, POWMAX, NA)
      DIMENSION X(145), Y(145)
      POWMAX=0
      DO 2 N=1, NXY
      VALUE1=X(N)*X(N)+Y(N)*Y(N)
      IF(POWMAX-VALUE1)1,2,2
    1 POWMAX=VALUE1
    2 CONTINUE
c
c     LIST TRANSFORM VALUES
      VALUE1=SQRTF(POWMAX)
      VALUE2=1./VALUE1
      PRINT 4, NA
    4 FORMAT (1HI, 13X, 34HFOURIER TRANSFORM OF A SEQUENCE
      OF, I4, 1X,.19HEQUISPACED IMPULSES / 14X, 37HREAL
      AND IMAGINARY VALUES ARE LISTED./
      .1HO, 5X, 10HFREQ*DELFT, 8X, 4HREAL, 10X, 9HIMAGINARY,
      8X, 7HMODULUS, 10X, 5HPHASE, )
      FREQ=-DELFT
      DO 114 K=1,NXY
      POWER=X(K)*X(K)+Y(K)*Y(K)
      VALUE1=SQRTF(POWER)
      VALUE1=VALUE1*VALUE2
c
c     COMPUTATION OF PHASE
c
      IGO=1
      IF(X(K))216,210,214
  210 IF(Y(K))212,211,213
  211 PHASE=0
      GO TO 215
  212 PHASE=-.5
      GO TO 215
  213 PHASE=.5
      GO TO 215
  216 IGO=2
  214 ARG=Y(K)/X(K)
```

SUBROUTINE MOPH (CONTINUED)

```
      PHASE=.3183098*ATAN(ARG)
      GO TO (215,217),IGO
  217 IF(Y(K))218,219,220
  220 PHASE=1.+PHASE
      GO TO 215
  218 PHASE=PHASE-1.
      GO TO 215
  219 IF(X(K)225,211,211
  225 PHASE=1.
  215 REAL=X(K)*VALUE1
      AMAG=Y(K)*VALUE1
      FREQ=FREQ+DELFT
      PRINT 6, FREQ, REAL, AMAG, VALUE1, PHASE
    6 FORMAT (1HO, 5(F13.8, 3X,))
  114 CONTINUE
      RETURN
      END
```

APPENDIX II

DATA SAMPLES

As stated previously due to the abundance of repetitive values calculated for all the algorithms, only four of the algorithms are listed in table form here. The other algorithms not listed give identical values for the particular function transformed.

The tables listed in this appendix point out the results arrived at in the section, Results and Conclusions. Also listed in this section are the time values for the two input functions.

TIME VALUES - FUNCTION 1

81 AND 129 POINTS

	0	1	2	3	4
0	0	.00058	.00605	.02209	.05249
5	.09860	.15963	.23324	.31614	.40463
10	.49499	.58377	.66798	.74519	.81355
15	.87180	.91919	.95544	.98067	.99530
20	.99999	.99560	.98307	.96341	.93761
25	.90684	.87194	.83388	.79350	.75158
30	.70880	.66575	.62295	.58083	.53975
35	.49999	.46180	.42532	.39068	.35796
40	.32719	.29839	.27154	.24659	.22350
45	.20218	.18257	.16458	.14812	.13309
50	.11941	.10698	.09570	.08550	.07629
55	.06799	.06051	.05380	.04778	.04238
60	.03756	.03325	.02940	.02598	.02293
65	.02023	.01782	.01569	.01380	.01213
70	.01066	.00935	.00820	.00719	.00630
75	.00551	.00482	.00421	.00368	.00321
80	.00280	.00244	.00213	.00185	.00161
85	.00140	.00122	.00106	.00092	.00080
90	.00069	.00060	.00052	.00045	.00039
95	.00034	.00029	.00025	.00022	.00019
100	.00016	.00014	.00012	.00010	.00009
105	.00007	.00006	.00005	.00005	.00004
110	.00003	.00003	.00002	.00002	.00002
115	.00001	.00001.	.00001	.00001	0.0
120	0.0	0.0	0.0	0.0	0.0
125	0.0	0.0	0.0	0.0	0.0

TABLE II - 1

FUNCTION 1 = $kt^{T_o Q} e^{-QT}$

DELTA T = 1.0

T-1179

TIME VALUES - FUNCTION 2

81 POINTS

	0	.05	.10	.15	.20
0.0	1.000000	.991802	.967531	.928135	.875140
0.25	.810569	.736840	.656638	.572787	.488107
0.50	.405285	.326749	.254572	.190386	.135338
0.75	.090063	.054696	.028904	.011945	.001727
1.00	.000000	.002249	.007996	.015791	.024309
1.25	.032423	.039240	.044136	.046758	.047011
1.50	.045032	.041141	.035799	.029546	.022947
1.75	.016542	.010804	.006102	.002680	.000652
2.00	.000000	.000590	.002194	.004518	.007233
2.25	.010007	.012536	.014566	.015911	.016467
2.50	.016211	.015201	.013557	.011454	.009097
2.75	.006699	.004465	.002571	.001150	.000285
3.00	.000000	.000267	.001007	.002105	.003419
3.25	.004796	.006090	.007168	.007928	.008304
3.50	.008271	.007843	.007071	.006038	.004844
3.75	.003603	.002424	.001409	.000636	.000159

TABLE II - 2

FUNCTION 2 = $SINC^2$

DELTA T - .05

TIME VALUES - FUNCTION 2

129 POINTS

	0	.03	.06	.09	.12
0	1.000000	.997043	.988212	.97634	.953515
.15	.928135	.897851	.863080	.824298	.782031
.30	.736840	.689316	.640068	.589712	.538859
.45	.488107	.438028	.389163	.342008	.297013
.60	.254572	.215017	.178618	.145579	.116037
.75	.090063	.067666	.048790	.033327	.021114
.90	.011945	.005575	.001727	.000102	.000384
1.05	.002249	.005372	.009436	.014134	.019079
1.20	.024309	.029289	.033914	.038014	.041455
1.35	.044136	.045994	.046997	.047147	.046473
1.50	.045032	.042900	.040172	.036958	.033375
1.65	.029546	.025592	.021634	.017784	.014144
1.80	.010804	.007840	.005309	.003255	.001700
1.95	.000652	.000102	.000025	.000382	.001125
2.10	.002194	.003522	.005040	.008353	.010007
2.25	.010007	.011572	.012989	.014210	.015193
2.40	.015911	.016342	.016480	.016326	.015892
2.55	.015201	.014280	.013167	.011902	.010530
2.70	.009097	.007649	.006233	.004890	.003658
2.85	.002571	.001655	.000931	.000412	.000102
3.00	.000000	.000098	.000380	.000826	.001411
3.15	.002105	.002877	.003694	.004523	.005332
3.30	.006090	.006770	.007348	.007805	.008127
3.45	.008304	.008333	.008216	.007958	.007572
3.60	.007071	.005805	.005090	.004347	.003603
3.75	.003603	.002881	.002205	.001595	

TABLE II - 3

FUNCTION 2 = $SINC^2$

DELTA T = .03

T-1179

AMPLITUDE VALUES FUNCTION 1 81 POINTS

CDC 8090

DELTA FREQ.	FT2	FT4	FT5
0.000	.99999999	.99999999	.99999999
0.005	.95307677	.95307671	.95307827
0.010	.77114865	.77114865	.77114838
0.015	.57620717	.57620726	.57620626
0.020	.40523255	.40523250	.40523218
0.025	.27544226	.27544230	.27544260
0.030	.18411894	.18411893	.18411898
0.035	.12418553	.12418553	.12418566
0.040	.08430793	.08430792	.08430790
0.045	.05770939	.05770939	.05770942
0.050	.04088957	.04088957	.04088958
0.055	.02884608	.02884608	.02884608
0.060	.02090451	.02090450	.02090451
0.065	.01565712	.01565712	.01565712
0.070	.01135236	.01135236	.01135235
0.075	.00889723	.00889723	.00889722
0.080	.00675607	.00675607	.00675607
0.085	.00513657	.00513656	.00513657
0.090	.00430326	.00430326	.00430325
0.095	.00315050	.00315049	.00315050
0.100	.00270905	.00270906	.00270905

TABLE II - 4

FUNCTION 1 = $kt^{ToQ}e^{-QT}$

MAXIMUM FREQUENCY = 0.10

INCREMENT OF FREQUENCY = 0.005

AMPLITUDE VALUES FUNCTION 1 129 POINTS

CDC 8090

DELTA FREQ.	FT2	FT4	FT5
0.000	.99999999	.99999999	.99999999
0.005	.93419012	.93419006	.93419163
0.010	.77016979	.77016979	.77016953
0.015	.57636335	.57636344	.47636243
0.020	.40485689	.40485684	.40485651
0.025	.27491556	.27491561	.27491591
0.030	.18440353	.18440353	.18550357
0.035	.12392210	.12392210	.12392223
0.040	.08412949	.08412948	.08412946
0.045	.05795680	.05795680	.05795683
0.050	.04059855	.04059855	.04059857
0.055	.02892965	.02892966	.02892966
0.060	.02097001	.02097000	.02097001
0.065	.01544616	.01544616	.01544616
0.070	.01155379	.01155379	.01155379
0.075	.00876631	.00879932	.00876631
0.080	.00674294	.00674294	.00674294
0.085	.00524863	.00524863	.00524863
0.090	.00413340	.00413340	.00413339
0.095	.00328906	.00328905	.00328906
0.100	.00264272	.00264273	.00264272

TABLE II - 5

FUNCTION 1 = $kt^{T_o Q} e^{-QT}$

MAXIMUM FREQUENCY = 0.10

INCREMENT OF FREQUENCY = 0.005

AMPLITUDE VALUES FUNCTION 2 81 POINTS

CDC 8090

DELTA FREQ.	FT2	FT4	FT5
0.000	1.00000000	1.00000000	1.00000000
0.005	.95681238	.95681235	.95681266
0.010	.88582081	.88582081	.88582079
0.015	.83930908	.83930910	.83930887
0.020	.78873912	.78873911	.78873885
0.025	.73896874	.73896874	.73896909
0.030	.69560773	.69560770	.69560781
0.035	.64164650	.64164649	.64164666
0.040	.59122757	.59122757	.59122748
0.045	.52871854	.52871853	.52871860
0.050	.43167441	.43167441	.43167451
0.055	.34654176	.34654174	.34654180
0.060	.30384995	.30384995	.30384993
0.065	.27251323	.27251323	.27251323
0.070	.24684257	.24684257	.24684255
0.075	.22801608	.22801608	.22801609
0.080	.21059930	.21059930	.21059926
0.085	.19693648	.19693647	.19693645
0.090	.18475212	.18475213	.18475209
0.095	.17397784	.17397784	.17397780
0.100	.16493717	.16493717	.16493714

TABLE II - 6

FUNCTION 2 = $SINC^2$

MAXIMUM FREQUENCY = 0.10

INCREMENT OF FREQUENCY = 0.005

AMPLITUDE VALUES FUNCTION 2 129 POINTS

CDC 8090

DELTA FREQ.	FT2	FT4	FT5
0.000	1.00000000	1.00000000	1.00000000
0.005	.90615937	.90615936	.90615996
0.010	.82367167	.82367167	.83267167
0.015	.73955156	.73955159	.73955115
0.020	.66352948	.66352945	.66352898
0.025	.57923848	.57923853	.57923901
0.030	.43964796	.42964794	.43964813
0.035	.32063166	.32063167	.32063180
0.040	.26746351	.26746352	.26746346
0.045	.23156242	.23156243	.23156246
0.050	.20408390	.20408390	.20408393
0.055	.18283140	.18283139	.18283141
0.060	.16708726	.16708728	.16708726
0.065	.15258878	.15258880	.15258878
0.070	.14119292	.14119291	.14119292
0.075	.13140194	.13140194	.13140192
0.080	.12305525	.12305525	.12305524
0.085	.11531689	.11531688	.11531687
0.090	.10913360	.10913361	.10913358
0.095	.10300871	.10300871	.10300869
0.100	.09813238	.09813239	.09813236

TABLE II - 7

FUNCTION 2 = SINC^2

MAXIMUM FREQUENCY = 0.10

INCREMENT OF FREQUENCY = 0.005

AMPLITUDE VALUES FUNCTION 1 81 POINTS

CDC 6400

DELTA FREQ.	FT2	FT4	FT5	FT6
0.000	1.00000000	1.00000000	1.00000000	1.00000000
0.005	.93507674	.93507674	.93507674	.93507674
0.010	.77114867	.77114867	.77114867	.77114867
0.015	.57620724	.57620724	.57620724	.57620724
0.020	.40523251	.40523251	.40523251	.40523251
0.025	.27544320	.27544230	.27544230	.27544230
0.030	.18411893	.18411893	.18411893	.18411893
0.035	.12418552	.12418552	.12418552	.12418552
0.040	.08430792	.08430792	.08430792	.08430792
0.045	.05770939	.05770939	.05770939	.05770939
0.050	.04088957	.04088957	.04088957	.04088957
0.055	.02884607	.02884607	.02884607	.02884607
0.060	.02090451	.02090451	.02090451	.02090451
0.065	.01565712	.01565712	.01565712	.01565712
0.070	.01135235	.01135235	.01135235	.01135235
0.075	.00889723	.00889723	.00889723	.00889723
0.080	.00675607	.00675607	.00675607	.00675607
0.085	.00513657	.00513657	.00513657	.00513657
0.090	.00430326	.00430326	.00430326	.00430326
0.095	.00315050	.00315050	.00315050	.00315050
0.100	.00270906	.00260906	.00270906	.00270906

TABLE II - 8

FUNCTION 1 = $kt^{T_oQ}e^{-Qt}$

MAXIMUM FREQUENCY = 0.10

INCREMENT OF FREQUENCY = 0.005

AMPLITUDE VALUES FUNCTION 1 129 POINTS

CDC 6400

DELTA FREQ.	FT2	FT4	FT5	FT6
0.000	1.00000000	1.00000000	1.00000000	1.00000000
0.005	.93419010	.93419010	.93419010	.93419010
0.010	.77016983	.77016983	.77016983	.77016983
0.015	.57637343	.57636343	.57636343	.57636343
0.020	.40485686	.40485686	.40485686	.40485686
0.025	.27491561	.27491561	.27491561	.27491561
0.030	.18440352	.11440352	.18440352	.18440352
0.035	.12392209	.12392209	.12392209	.12392209
0.040	.08412948	.08412948	.08412948	.08412948
0.045	.05795680	.05795680	.05795680	.05795680
0.050	.04059856	.04059856	.04059856	.04059856
0.055	.02892965	.02892965	.02892965	.02802065
0.060	.02097001	.02097001	.02097001	.02097001
0.065	.01544616	.01544616	.01544616	.01544616
0.070	.01155379	.01155379	.01155379	.01155379
0.075	.00876631	.00876631	.00876631	.00876631
0.080	.00674294	.00674294	.00674294	.00674294
0.085	.00524863	.00524863	.00524863	.00524863
0.090	.00413340	.00413340	.00413340	.00413340
0.095	.00328906	.00328906	.00328906	.00328906
0.100	.00264272	.00264272	.00264272	.00264272

TABLE II - 9

FUNCTION 1 = $kt^{T_oQ}e^{-Qt}$

MAXIMUM FREQUENCY = 0.10

INCREMENT OF FREQUENCY = 0.005

AMPLITUDE VALUES FUNCTION 2 81 POINTS

CDC 6400

DELTA FREQ.	FT2	FT4	FT5	FT6
0.000	1.00000000	1.00000000	1.00000000	1.00000000
0.005	.95681237	.95681237	.95681237	.95681237
0.010	.88582083	.88582083	.88582083	.88582083
0.015	.83930913	.83930913	.83930913	.83930913
0.020	.78873913	.78873913	.78873913	.78873913
0.025	.73896879	.73896879	.73896879	.73896879
0.030	.69650770	.69560770	.69560770	.69560770
0.035	.64164647	.64164647	.64164647	.64164647
0.040	.59122755	.59122755	.59122755	.59122755
0.045	.52871850	.52871850	.52871850	.52871850
0.050	.43167441	.43167441	.43167441	.43167441
0.055	.34654174	.34654174	.34654174	.34654174
0.060	.30384995	.30384995	.30384995	.30384995
0.065	.27251324	.27251324	.27251324	.27251324
0.070	.24684257	.24684257	.24684257	.24684257
0.075	.22801608	.22801608	.22801608	.22801608
0.080	.21059929	.21059929	.21059929	.21059929
0.085	.19693648	.19693648	.19693648	.19693648
0.090	.18475213	.18475213	.18475213	.18475213
0.095	.17397784	.17397784	.17397784	.17397784
0.100	.16493717	.16493717	.16493717	.16493717

TABLE II - 10

FUNCTION 2 = $SINC^2$

MAXIMUM FREQUENCY = 0.10

INCREMENT OF FREQUENCY = 0.005

AMPLITUDE VALUES FUNCTION 2 129 POINTS

CDC 6400

DELTA FREQ.	FT2	FT4	FT5	FT6
0.000	1.00000000	1.00000000	1.00000000	1.00000000
0.005	.90615928	.90615928	.90615928	.90615928
0.010	.82367160	.82367160	.82367160	.82367160
0.015	.73955153	.73955153	.73955153	.73955153
0.020	.66352939	.66352939	.66352939	.66352939
0.025	.57923847	.57923847	.57923847	.57923847
0.030	.43964788	.43964788	.43964788	.43964788
0.035	.32063163	.32063163	.32063162	.32063162
0.040	.26746351	.26746351	.26746351	.26746351
0.045	.23156240	.23156240	.23156240	.23156240
0.050	.20408388	.20408388	.20408388	.20408388
0.055	.18283138	.18283138	.18283138	.18283138
0.060	.16708726	.16708726	.16708726	.16708726
0.065	.15258878	.15258878	.15258878	.15258878
0.070	.14119292	.14119292	.14119292	.14119292
0.075	.13140193	.13140193	.13140193	.13140193
0.080	.12305525	.12305525	.12305525	.12305525
0.085	.11531688	.11531688	.11531688	.11531688
0.090	.10913359	.10913359	.10913359	.10913359
0.095	.10300870	.10300870	.10300870	.10300870
0.100	.09813238	.09813238	.09813238	.09813238

TABLE II - 11

FUNCTION 2 = $SINC^2$

MAXIMUM FREQUENCY = 0.10

INCREMENT OF FREQUENCY = 0.005

SELECTED REFERENCES

Bracewell, Ron, 1965, The Fourier transform and its applications: New York, McGraw-Hill Book Co.

Brand, L., 1955, Advanced calculus: New York, John Wiley and Sons, Inc.

Churchill, R. V., 1960, Complex variables and applications: New York, McGraw-Hill Co.

Cochran, W. T., Cooley, J. W., and others, 1967, What is the Fast Fourier Transform: Proceedings of the IEEE, Vol. 55, No. 10, October, 1967.

Goertzel, G., 1958, An algorithm for the evaluation of finite trigonometric series: Amer. Math. Monthly, Vol. 65.

Gray, R. L., 1965, The elements of linear filter theory: Department of Geophysics, Colorado School of Mines.

Guillemin, E. A., 1959, The mathematics of circuit analysis: New York, John Wiley and Sons, Inc.

Hamming, R. W., 1962, Numerical methods for scientists and engineers: New York, McGraw-Hill Book Co.

Hildebrand, F. B., 1956, Introduction to numerical analysis: New York, McGraw-Hill Book Co.,

Javid, M., and Brenner, E., 1963, Analysis transmission and filtering of signals: New York, McGraw Hill Book Co.

Kaplan, W., 1959, Advanced Calculus: Reading, Mass., Addison-Wesley Publishing Co., Inc.

Lee, Y. W., 1960, Statistical theory of communication: New York, John Wiley and Sons, Inc.

Lighthill, M. J., 1962, Introduction to Fourier analysis and generalized functions: Cambridge, University Press.

McNett, J. L., 1967, Digital filter design by minimizing the weighted square error in the frequency domain: M. Sc. Thesis, Colorado School of Mines.

Papoulis, Athanasios, 1962, The Fourier integral and its applications: New York, McGraw-Hill Book Co.

Prescott, Wesley T., 1967, Analysis of Fast Fourier Transform, R & D Report No. 12-611-0-67 (Restricted), Continental Oil Company, Ponca City, Oklahoma.

Ralston, A., and Wilf, H. S., 1960, Mathematical methods for digital computers: New York, John Wiley and Sons, Inc.

Truxall, J. G., 1955, Automatic feedback control system synthesis: New York, McGraw-Hill Book Co.

Tukey, J. W., and Cooley, J. W., 1965, An algorithm for the machine calculation of complex Fourier series: Mathematics of Computation, Vol. 19, No. 90, April, 1965.

Zemanian, A. W., 1965, Distribution theory and transform analysis: New York, McGraw-Hill Book Co.