UNIVERSITY OF SOUTHERN CALIFORNIA

DEPARTMENT OF CIVIL ENGINEERING

A NOTE ON THE CALCULATION OF FOURIER AMPLITUDE TRANSFORMS

by

F. E. Udwadia and M. D. Trifunac

Report No. CE 77-01

A report on research conducted under a Grant
from the National Science Foundation

April, 1977

ABSTRACT

This paper investigates the determination of the Fourier Amplitude Transform of a continuous, piecewise differentiable function using the computational efficiency of the Fast Fourier Transform. Interpolation between adjacent and decimated data sample points by means of splines and polynomials is studied. The linear spline which has wide application in earthquake engineering and seismological studies has been shown to provide better Fourier Amplitude estimates than the FFT through the examination of a numerical example. It is found that such a spline representation may be used to obtain improved estimates up to and often beyond the Nyquist frequency characterized by the spacing, Δt, of digital data points. To achieve a given level of accuracy, it is shown that the linear spline needs only about half the number of sample points per unit time. Some of the limitations of the technique are discussed and some applications cited.

INTRODUCTION

Before the widespread usage of the Fast Fourier Transform algorithm, a continuous time function (such as an accelerogram) encountered in earthquake engineering studies was modeled for digital computation by a series of polynomials passing through discrete digitized data points. A commonly used assumption was that the time function varied linearly between any two adjacent digitized data points.

The advent of the FFT algorithm[1,2,3] (which in essence calculates the Discrete Fourier Transform) with its large computational economy has not only made Fourier analysis an important aspect of data processing, but has led the engineer to adopt the discrete time series model for representing continuous time signals. Such a representation has been commonly used to facilitate the direct application of the FFT algorithm.

This discrete time data approach, which essentially assumes no knowledge of the time function between the consecutive digital data points, has been found to be useful in some time series analyses problems. However, in several earthquake engineering applications, in particular those dealing with semi-automatically digitized acceleration traces, additional information on the time function between digitial data points may be available. In ignoring such information, not only is there a loss of data but the problem of Fourier analysis becomes further compounded by difficulties that rise characteristically from the use of discrete time data. One such problem that would be

eliminated is that of aliasing, which arises purely because the function is not continuously defined for all time in the domain of study.

It is the purpose of this paper to use this additional information on the nature of the time function between consecutive digital data points[3] and to apply the FFT algorithm so that a computationally efficient method of Fourier analysis of a continuous, piecewise differentiable time function may be obtained.

THEORY

Consider the continuous function g(t) which one typically meets with in the analysis of strong ground motion data defined as follows

$$g(t) = \begin{cases} f(t) & 0 < t < T \\ 0 & \text{otherwise} . \end{cases}$$

The Fourier transform $F(\omega)$ of this function is defined as

$$F(\omega) = \int_{-\infty}^{\infty} g(t)e^{-i\omega t}dt$$

$$= \int_{0}^{T} f(t)e^{-i\omega t}dt . \qquad (1)$$

Dividing the time axis into N equal intervals, which correspond to the N+1 time coordinates t_i, $i = 0, 1, \ldots, N$, such that $t_{i+1} - t_i = \Delta t$ for all $i \in (0, N-1)$, equation (1) can be expressed as a summation over N subintervals so that

$$F(\omega) = \sum_{k=0}^{k=N-1} \int_{t_k}^{t_{k+1}} f_k(t) e^{-i\omega t} dt \qquad (2)$$

where $f_k(t)$ is the portion of $f(t)$ lying between points t_k and t_{k+1}. If further the function $f_k(t)$ is a known polynomial of order n, $p_k^{(n)}(t)$, between these two points then the integration in equation (2) can be performed directly. Equation (2) then reduces to

$$F(\omega) = \sum_{k=0}^{k=N-1} \int_{t_k}^{t_{k+1}} p_k^{(n)}(t) e^{-i\omega t} dt \ . \qquad (3)$$

In actual practice, in the digitization process, the values of the function $f(t)$ are determined only at a discrete set of points t_i, $i = 0, 1, \ldots, N$, so that obtaining such a polynomial representation for each interval along the time axis involves some degree of approximation. For a given order of polynomial, the extent to which the approximation is a good one will depend on the actual method of digitization used, which will in turn govern the spacing (which could be unequal) along the t-axis between various digitized points.

For the optical-mechanical digitization process,[4] for example, the first order (linear polynomial approximation may perhaps be the best. The reason for this is that in this semiautomatic process the operator picks out (and digitizes) points at such a spacing that the value of the function between one digitized point and its neighbor (on either side) can be approximated, in his judgment, by a straight line. Though this sort of a digitization process

is largely judgmental, trained operators have been found to perform very well.[4] This digitization practice in general leads to a non-uniform spacing of points on the t-axis. For ease of digital computations, several equispaced points are then interpolated between these non-equispaced digital data points. Such an interpolation would affect the accuracy of our approximation because the actual function between any two equispaced points may not be close to a straight line (linear) even though the function between two of the original non-equispaced points may have been closely representable by a straight line. Such inaccuracies would obviously arise when two consecutive equispaced data points straddle a point belonging to the original digitized (unequally spaced) data. However, if the spacing of the equidistant points along the t-axis is small in relation to the smallest spacing of the unequally spaced digital data, it has been found that the straight line approximation between the equidistant data points may be a good one.[4] From now on, the case of equispaced points will therefore be considered in this paper.

Though the method considered here is valid for any order polynomial, the main points of this discussion will be amply established by considering the linear and quadratic cases.

Case 1: The Linear Case

When

$$p_k^{(1)}(t) = y_k + \frac{\Delta y_k}{\Delta t}(t - t_k) \tag{4}$$

where

$$y_k = f(t_k),$$

$$\Delta y_k = y_{k+1} - y_k ,$$

$$\Delta t = t_{k+1} - t_k \qquad \text{for all } k\epsilon(0,N-1)$$

and $$N\Delta t = T ,$$

using equation (3) we have

$$F(\omega) = \sum_{k=0}^{k=N-1} A_k \tag{5}$$

where

$$A_k = \int_{t_k}^{t_{k+1}} [y_k + \frac{\Delta y_k}{\Delta t}(t - t_k)]e^{-i\omega t}dt . \tag{6}$$

Integrating, we get

$$A_k = y_k[\frac{1}{i\omega} - \{\frac{e^{-i\omega\Delta t} - 1}{\omega^2\Delta t}\}]e^{-i\omega k\Delta t}$$

$$+ y_{k+1}[-\frac{e^{-i\omega\Delta t}}{i\omega} + \frac{e^{-i\omega\Delta t} - 1}{\omega^2\Delta t}]e^{-i\omega k\Delta t} .$$

By equation (5) then,

$$\sum_{k=0}^{k=N-1} A_k = \sum_{k=0}^{k=N-1} y_k[\frac{1}{i\omega} - \{\frac{e^{-i\omega\Delta t} - 1}{\omega^2\Delta t}\}]e^{-i\omega k\Delta t}$$

$$+ \sum_{k=1}^{k=N} y_k[-\frac{1}{i\omega} + \frac{1 - e^{+i\omega\Delta t}}{\omega^2\Delta t}]e^{-i\omega k\Delta t} \tag{7}$$

so that

$$F(\omega) = \sum_{k=1}^{N-1} y_k e^{-i\omega k\Delta t}\,\Delta t\left[\frac{\sin \omega\Delta t/2}{\omega\Delta t/2}\right]^2$$

$$+ y_o\left[\frac{1}{i\omega} + \frac{1 - e^{-i\omega\Delta t}}{\omega^2\Delta t}\right]$$

$$+ y_N\left[-\frac{1}{i\omega} + \frac{1 - e^{+i\omega\Delta t}}{\omega^2\Delta t}\right]e^{-i\omega N\Delta t} \quad . \tag{8}$$

Adding and subtracting $y_o\left[\frac{\sin \omega\Delta t/2}{\omega\Delta t/2}\right]^2$ on the right hand side, we get

$$F(\omega) = \sum_{k=0}^{N-1} y_k e^{-i\omega k\Delta t}\,\Delta t\left[\frac{\sin \omega\Delta t/2}{\omega\Delta t/2}\right]^2$$

$$- y_o\left[-\frac{1}{i\omega} + \frac{(1 - e^{i\omega\Delta t})}{\omega^2\Delta t}\right]$$

$$+ y_N\left[-\frac{1}{i\omega} + \frac{(1 - e^{i\omega\Delta t})}{\omega^2\Delta t}\right]e^{-i\omega N\Delta t} \quad . \tag{9}$$

If $y_o = y_N = 0$, then

$$F(\omega) = \Delta t\left[\frac{\sin \omega\Delta t/2}{\omega\Delta t/2}\right]^2 \sum_{k=0}^{N-1} y_k e^{-i\omega k\Delta t} \quad . \tag{10}$$

The above expression is valid for all frequencies, ω. The summation in equations (9) and (10) can be done very efficiently if we restrict ourselves to the frequencies $\omega_j = \frac{2\pi j}{N\Delta t}$; $j = 0, 1, \ldots, N-1$, for then the summation will be simply the Discrete Fourier Transform of the digitized function f(t). This transform can be economically computed

using the Fast Fourier Transform[2] algorithm. It may be noted that at these specific frequencies ω_j, $j = 0, 1, \ldots, N-1$, the contributions to $F(\omega)$ from the second and third terms of equation (9) will cancel out if $y_0 = y_N$ because of the periodicity of the exponential term. We will then have

$$F(\omega) = \Delta t\left[\frac{\sin \omega\Delta t/2}{\omega\Delta t/2}\right]^2 \sum_{k=0}^{N-1} y_k \exp\left(-\frac{i2\pi jk}{N}\right)$$

$$= \left[\frac{\sin \omega\Delta t/2}{\omega\Delta t/2}\right]^2 \{\text{Discrete Fourier Transform of } f(t)\} . \qquad (11)$$

Had no information on the function been provided between the various digital points, the transform at the frequencies ω_j would have been computed by first analytically setting up the Discrete Transform and numerically computing it using the FFT. As seen from equation (11), the Fourier Transform of a piecewise linear continuous function can be expressed by the discrete transform at the frequencies ω_j provided a multiplication by the envelope function $\left(\frac{\sin \omega\Delta t/2}{\omega\Delta t/2}\right)^2$ is performed.[3] The economy afforded by the FFT algorithm can therefore still be utilized to obtain the transform, at the frequencies ω_j, of the continuous function. It may be noted that the condition $y_0 = y_N = 0$ mentioned above does not usually lead to a loss of generality, for this condition can always be guaranteed if f(t) is continuous over the real line t and is nonzero only over a finite domain. If, however, f(t) is discontinuous at the end points of the domain (0, T), the expression given by equation (9) would need to be used, again evaluating the transforms at the

specific frequencies ω_j so as to make use of the computational efficiency of the FFT.

It is instructive at this point to study the structure of equation (9) in more detail. The first term on the right hand side is the main contribution to the Fourier integral, whereas the second and third terms represent the effect of the two boundary points. Noting this, equation (9) can be derived in an alternate and perhaps more illustrative manner. Let the function f(t) be defined inbetween various digitized points by a series of spline functions. For example, as indicated in Figure 1, the piecewise linear function f(t) $t\epsilon(0, t_N)$ can be approximated by using a series of linear splines. Figure 1 (top) shows the linear spline denoted by $S^{(1)}$ and its negative half denoted by $S^{(1)}_{\frac{1}{2}}$. We see then (Figure 1) that the Fourier integral of f(t), as approximated by $p^{(1)}(t)$, can be expressed as the transform of a series of weighted spline functions as follows:

$$F(\omega) = \sum_{k=0}^{N-1} \int_{t_k}^{t_{k+1}} p_k^{(1)}(t) e^{-i\omega t} dt = \sum_{k=0}^{N-1} \int_{t_{k-1}}^{t_{k+1}} y_k S^{(1)}(t-k\Delta t) e^{-i\omega t} dt$$

$$- y_o \int_{-\Delta t}^{o} S^{(1)}_{\frac{1}{2}}(t) e^{-i\omega t} dt + y_N \int_{t_N-\Delta t}^{t_N} S^{(1)}_{\frac{1}{2}}(t-N\Delta t) e^{-i\omega t} dt \qquad (12)$$

But the transform of $S^{(1)}(t) = \overline{S^{(1)}}(\omega) = \Delta t[\frac{\sin \omega\Delta t/2}{\omega\Delta t/2}]^2$ so that

$$F(\omega) = \overline{S^{(1)}}(\omega) \sum_{k=0}^{N-1} y_k e^{-i\omega k\Delta t} - y_o \overline{S^{(1)}_{\frac{1}{2}}}(\omega) + y_N \overline{S^{(1)}_{\frac{1}{2}}}(\omega) e^{-i\omega N\Delta t} \qquad (13)$$

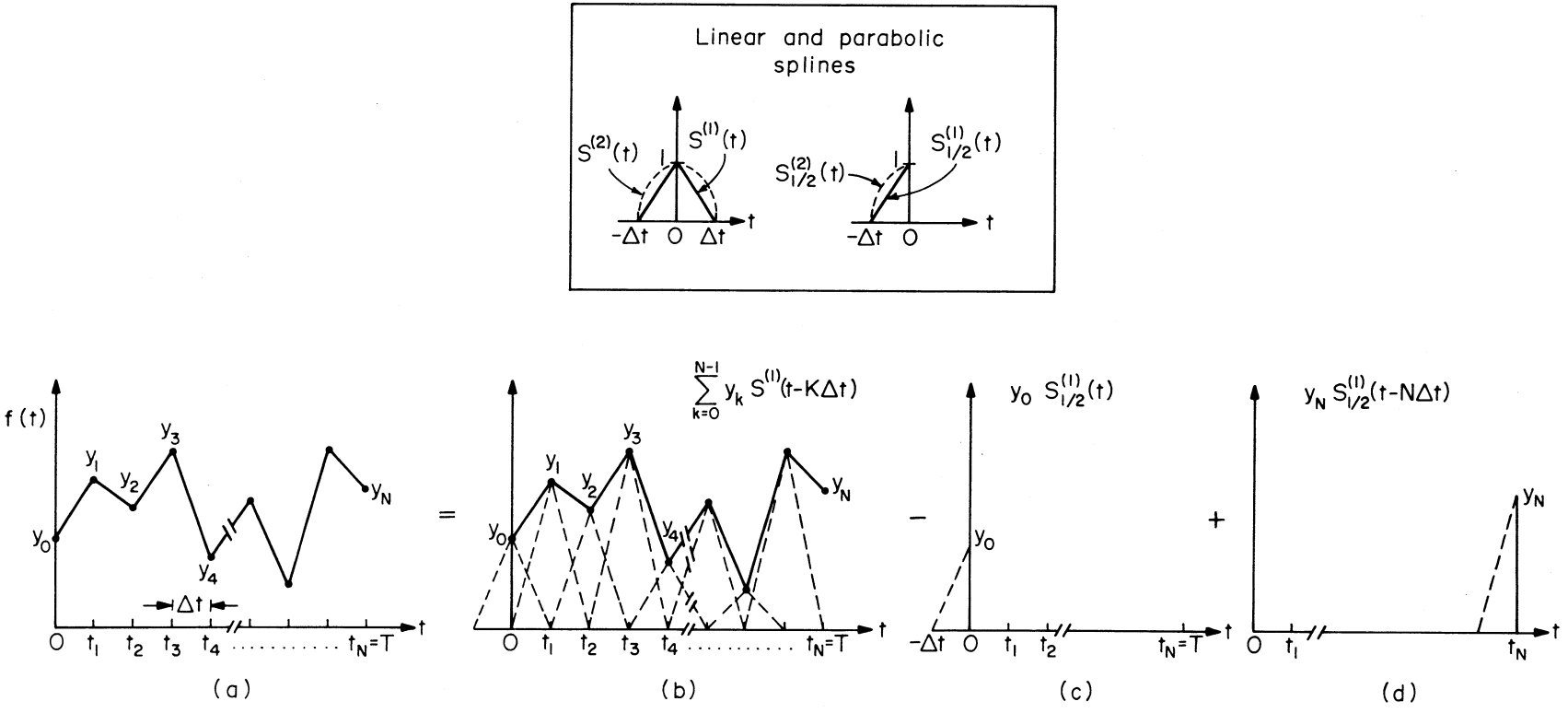

Figure 1. (a) Time series y_i, $i = 0, 1, \ldots, N$, with linear interpolation between any two adjacent points. (b, c, d) Illustration of the use of splines to obtain interpolation.

where $\overline{S_{\frac{1}{2}}^{(1)}(\omega)}$ is the transform of $S_{\frac{1}{2}}^{(1)}(t)$ and is given by

$$\overline{S_{\frac{1}{2}}^{(1)}(\omega)} = -\frac{1}{i\omega} + \frac{1}{\omega^2\Delta t}\left[1 - e^{i\omega\Delta t}\right] .$$

Equations (9) and (13) are identical, as of course they should be. However, equation (13) can now be used to generalize the concepts to higher order splines which could represent the curve between two adjacent points in terms of say a second or third order polynomial. We note that the resultant curve obtained by using such a series of splines will be continuous but only piecewise differentiable. In a manner similar to the above equation (13) may then be generalized to an n^{th} order spline as follows:

$$F(\omega) = \overline{S^n(\omega)} \sum_{k=0}^{N-1} y_k e^{-i\omega k\Delta t} - y_o\overline{S_{\frac{1}{2}}^{(n)}(\omega)} + y_N\overline{S_{\frac{1}{2}}^{n}(\omega)}e^{-i\omega N\Delta t} \qquad (14)$$

where $\overline{S^{(n)}(\omega)}$ is the Fourier transform of $S^{(n)}(t)$. Further, for the frequencies $\omega_j = \frac{2\pi j}{N\Delta t}$ assuming $y_o = y_N$ we have

$$F(\omega) = \overline{S^n(\omega)} \sum_{k=0}^{N-1} y_k e^{-i\omega t} \qquad (15)$$

Equation (15) is the generalization then of equation (10). In particular, for example, the parabolic spline leads to

$$\overline{S^{(2)}(\omega)} = \Delta t \frac{1}{(\omega\Delta t/2)^2}\left[\frac{\sin\omega\Delta t}{\omega\Delta t} - \cos\omega\Delta t\right] . \qquad (16)$$

The normalized functions $\overline{S^{(1)}(\omega)}/\Delta t$ and $\overline{S^{(2)}(\omega)}/\Delta t$ are shown in Figure 2.

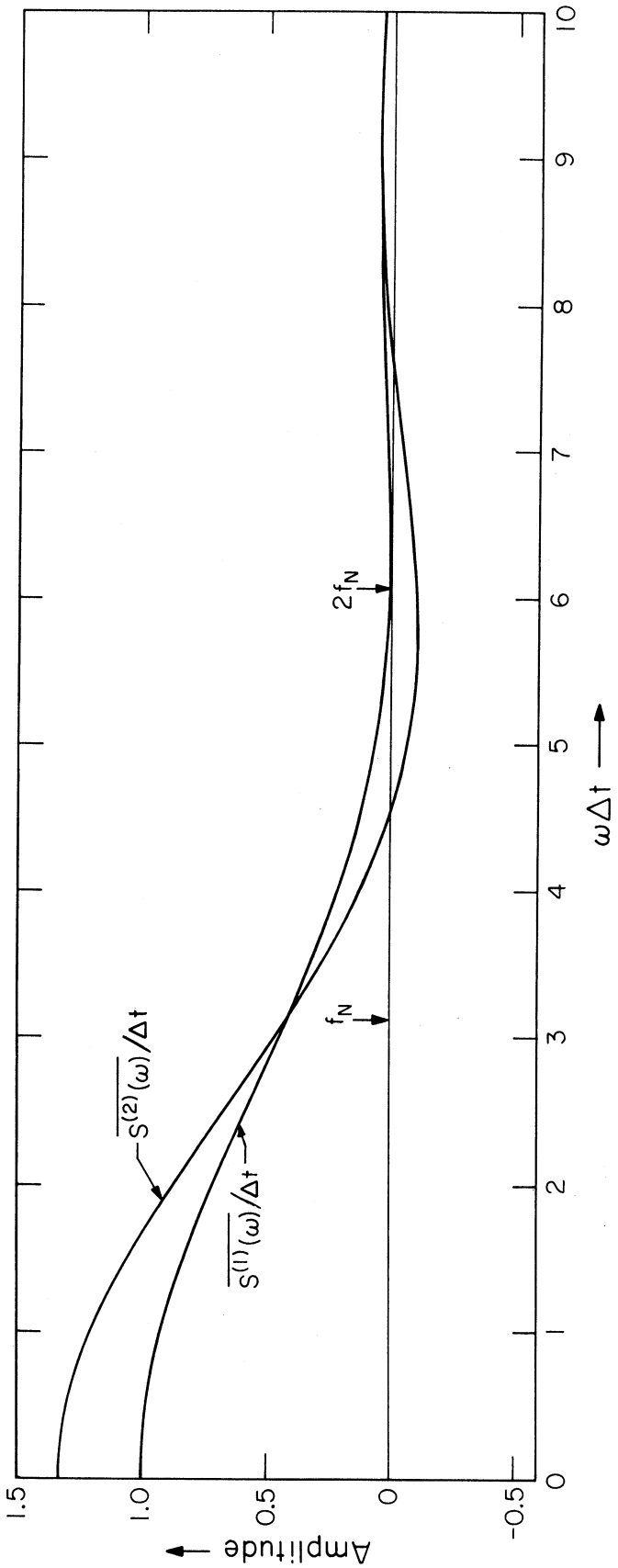

Figure 2. Envelope functions $\overline{S^{(1)}(\omega)}/\Delta t$ and $\overline{S^{(2)}(\omega)}/\Delta t$.

The envelope function $\overline{S^{(1)}(\omega)}/\Delta t$ is seen to be always positive except at the frequencies given by $\omega\Delta t = 2n\pi$, $n = 1, 2, \ldots$ at which it is zero. Its first zero occurs at twice the Nyquist frequency of the sampled data ($\frac{1}{2\Delta t}$). The function is less than unity for $\omega\Delta t > 0$ so that it tends to reduce the Fourier spectral amplitudes all over the spectrum.

The function $\overline{S^{(2)}(\omega)}/\Delta t$ is an oscillating function having a value of 4/3 at $\omega = 0$. The parabolic spline therefore concentrates more energy in the low frequency zone reaching the same amplitude at the Nyquist frequency as does the function $\overline{S^{(1)}(\omega)}$.

The functions $\overline{S^{(n)}(\omega)}$ may also be thought of as representing low pass filters, filtering out the periodic spectrum of the Discrete Fourier Transform.

Case 2: The Quadratic Case

Expressing $p_k^{(2)}(t)$ by the collocation polynomial

$$p_k^{(2)}(t) = y_k + \frac{\Delta y_k}{\Delta t}(t - t_k) + \frac{\Delta^2 y_k}{2(\Delta t)^2}(t - t_k)(t - t_{k+1}) \tag{17}$$

where

$$\Delta^2 y_k = \Delta[\Delta y_k] = y_{k+2} - 2y_{k+1} + y_k ,$$

and noting that $t_{k+1} = t_k + \Delta t$, $F(\omega)$ can be expressed as

$$F(\omega) = \sum_{k=0}^{N-1} A_k + \sum_{k=0}^{k=N-1} \frac{\Delta^2 y_k}{2(\Delta t)^2}[t^2 - (2t_k + \Delta t)t + t_k t_{k+1}]e^{-i\omega t}$$

$$= \sum_{k=0}^{N-1} A_k + \sum_{k=0}^{N-1} B_k \tag{18}$$

where A(k) is defined by equation (6),

$$B_k = T(\omega)\Delta^2 y_k e^{-i\omega t_k}$$

and

$$T(\omega) = \frac{1}{2(\Delta t)^2}\{\Delta t(\frac{1+e^{-i\omega\Delta t}}{\omega^2}) + \frac{1}{i\omega^3}(e^{-i\omega\Delta t} - 1)\} \quad (19)$$

The second term on the right hand side of equation (18) can be now written as

$$\sum_{k=0}^{N-1} B_k = \overline{T}(\omega) \sum_{k=0}^{N-1} [y_{k+2} - 2y_{k+1} + y_k] e^{-i\omega k\Delta t} . \quad (20)$$

Assuming that $y_{N+\ell}$, $y_{-\ell}$, $\ell = 1, 2, \ldots$ are zero

$$\sum_{k=0}^{N-1} B_k = \overline{T}(\omega)[e^{i\ \Delta t} - 1]^2 \sum_{k=0}^{N-1} y_k e^{-i\omega k\Delta t} + y_N e^{-i\omega(N-1)\Delta t}[e^{i\omega\Delta t} - 2] . \quad (21)$$

If further $y_o = y_N = 0$,

$$F(\omega) = [\overline{S^{(1)}}(\omega) + \overline{T}(\omega)(e^{i\omega\Delta t} - 1)^2] \sum_{k=0}^{k=N-1} y_k e^{-i\omega k\Delta t} . \quad (22)$$

The value of the summation as before may be obtained at the frequencies $\omega_j = \frac{2\pi j}{N\Delta t}$ by the FFT algorithm. It may be noted that this mode of expressing $p^{(2)}(t)$ differs from the second order spline function discussed earlier, for in this case the values of the three ordinates, y_{k+2}, y_{k+1} and y_k, are all utilized to specify the curve between t_k and t_{k+1}.

Case 3. Other Representations of f(t)

The results for the linear and higher order spline representations of f(t) can be used as a basis for further generalizations of the method and may also lead to a substantial reduction of computer time. As an example, one such representation of f(t) is shown in Figure 3. By decimating the original sequence $y_0, y_1, y_2, \ldots, y_{N-1}, y_N$ into $y_0, y_2, \ldots, y_{N-2}, y_N$, where the samples $y_0, y_1, y_2, \ldots$ are connected by straight lines, we obtain the new decimated function $f_1(t_i)$, $i = 0, 2, 4, \ldots$ which can be represented by linear splines over $4\Delta t$ intervals and with $N/2 + 1$ data points. The difference $f_2(t_i)$, $i = 1, 3, 5, \ldots$ between the functions $f_1(t_i)$, $i = 0, 2, 4, \ldots$ and the original sequence $y_0, y_1, y_2, y_3, \ldots$ can next be represented by $N/2$ amplitudes $x_1, x_3, x_5, \ldots$ (see Figure 3), which are also equally spaced at intervals of $2\Delta t$. These points $0, x_1, 0, 0, x_3, 0, x_5, 0, \ldots$, etc., can now be connected with straight lines, parabolas, or, for example, half-sine waves. Since this representation of

$$f(t_i) = \underset{i=0,2,4\ldots}{f_1(t_i)} + \underset{i=1,3,5\ldots}{f_2(t_i)}$$

is linear, the Fourier transform of f(t) becomes a linear combination of the Fourier transforms of $f_1(t_i)$ and $f_2(t_i)$. Assuming, for example, linear splines for both $f_1(t_i)$ and $f_2(t_i)$, we get (assuming that $y_0 = y_N = 0$)

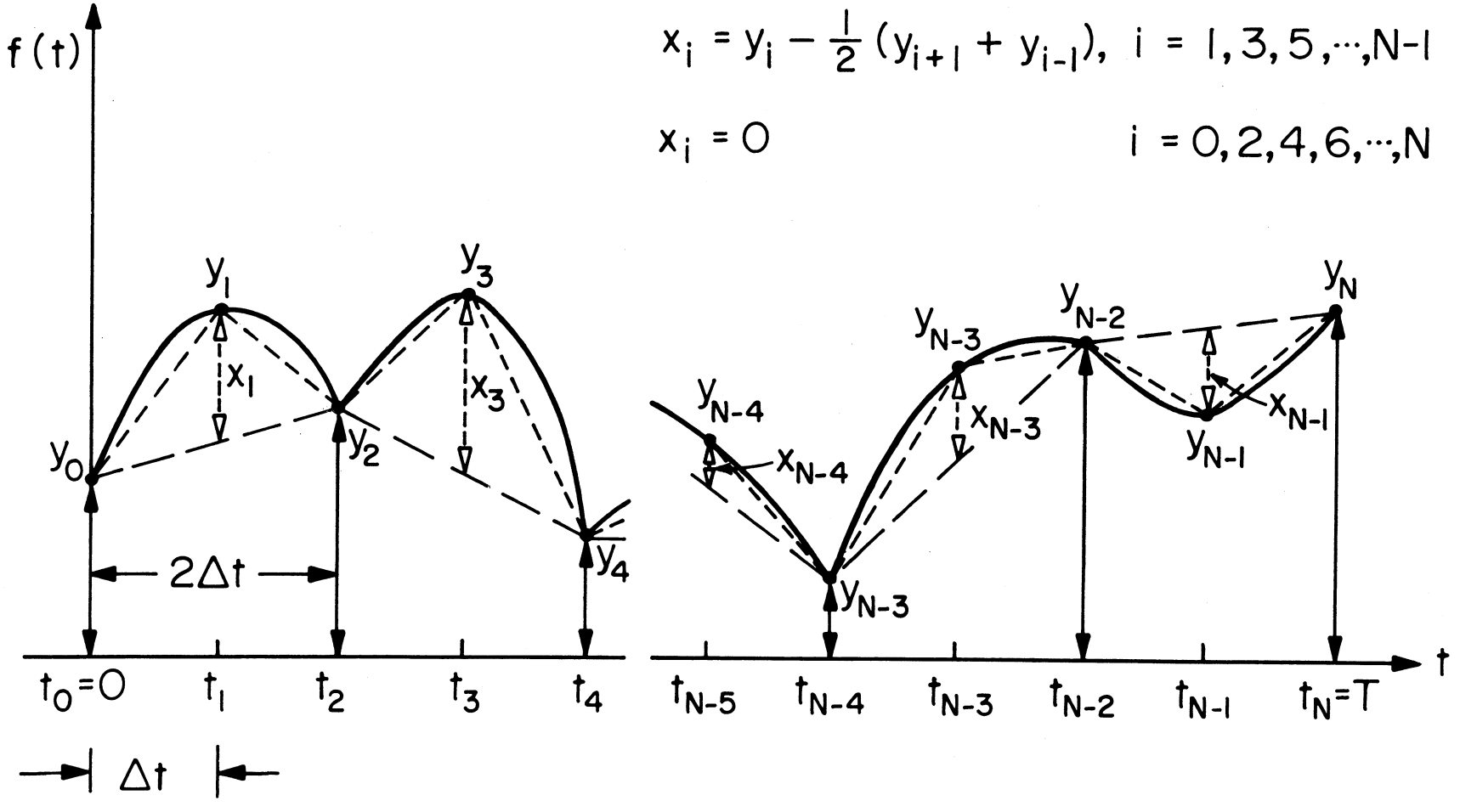

Figure 3. Representation of $f(t_i)$ using two series, each containing every alternate point of the original digital data of $f(t_i)$.

$$F(\omega) = \left(\frac{\sin \omega\Delta t}{\omega\Delta t}\right)^2 \{\text{D.F.T. of } f_1(t_i)_{i=0,2,4,\ldots,N}\} +$$

$$+ \left(\frac{\sin \omega\Delta t/2}{\omega\Delta t/2}\right)^2 \{\text{D.F.T. of } f_2(t_i)_{i=1,3,5,\ldots,N-1}\} \qquad (23)$$

The advantage in using this representation is that instead of approximately $N \log_2 N$ operations required to calculate the D.F.T. for the original data, only $N \log_2 (\frac{N}{2})$ operations are now required. The disadvantage of this method, however, is that only N/4 D.F.T. amplitudes are now available instead of N/2. Whether the benefits of shorter computer time may be more useful than the larger number of estimates of the Fourier amplitude spectrum will, of course, have to be determined by the requirements of each particular application.

The above representation of $f(t_i)$ in terms of $f_1(t_i)$, $i = 0, 2, 4, \ldots$, and $f_2(t_i)$, $i = 1, 3, 5, \ldots$, suggests that $f_1(t_i)$ would contain the predominant low-frequency part of $f(t_i)$, while $f_2(t_i)$, which is zero at $t_0, t_2, t_4, \ldots$, would contribute smaller amplitudes but higher frequencies to the final D.F.T. This is clearly the case, as may be seen from $(\frac{\sin \omega\Delta t}{\omega\Delta t})^2$, which has its first zero at Nyquist frequency $\omega_N = \pi/\Delta t$ and from $(\frac{\sin \omega\Delta t/2}{\omega\Delta t/2})^2$, whose first zero is at $2\omega_N$.

Other generalizations along these lines are, of course, possible. Their form and possible advantages, depend on the nature of f(t) between the digitized points and the number of Fourier amplitude estimates that are required.

NUMERICAL EXAMPLE

As an example to illustrate some of the ideas presented earlier, consider the function, g(t), to be a damped sine wave. Such a function defined by

$$g(t) = \begin{cases} e^{-\alpha t} \sin \frac{2\pi t}{T_0} & 0 \le t \le kT_0 \\ 0 & \text{otherwise} , \end{cases}$$

where α represents the damping, T_0 the period of the sine wave, and k the number of cycles, is commonly encountered both in earthquake engineering and in seismological applications.

The Fourier Transform of g(t) is

$$F[g(t)] = -\omega_0 \left[\frac{a + ib}{\omega^2 - \alpha^2 - \omega_0^2 - 2i\alpha} \right] \tag{24}$$

where

$$\omega_0 = 2\pi / T_0 ,$$

$$a = 1 - \exp(-k\alpha T_0) \cos (k\omega T_0)$$

and

$$b = \exp(-k\alpha T_0) \sin (k\omega T_0) .$$

Figures 4 and 5 show the function g(t) for $\alpha = 3$, $T_0 = 1$, and $k = 2$, and the discrete time signal sampled at the time intervals $\Delta t = T_0/8$, $T_0/10$, $T_0/16$, and $T_0/32$. The Fourier amplitude transforms corresponding to these Δt values are also shown.

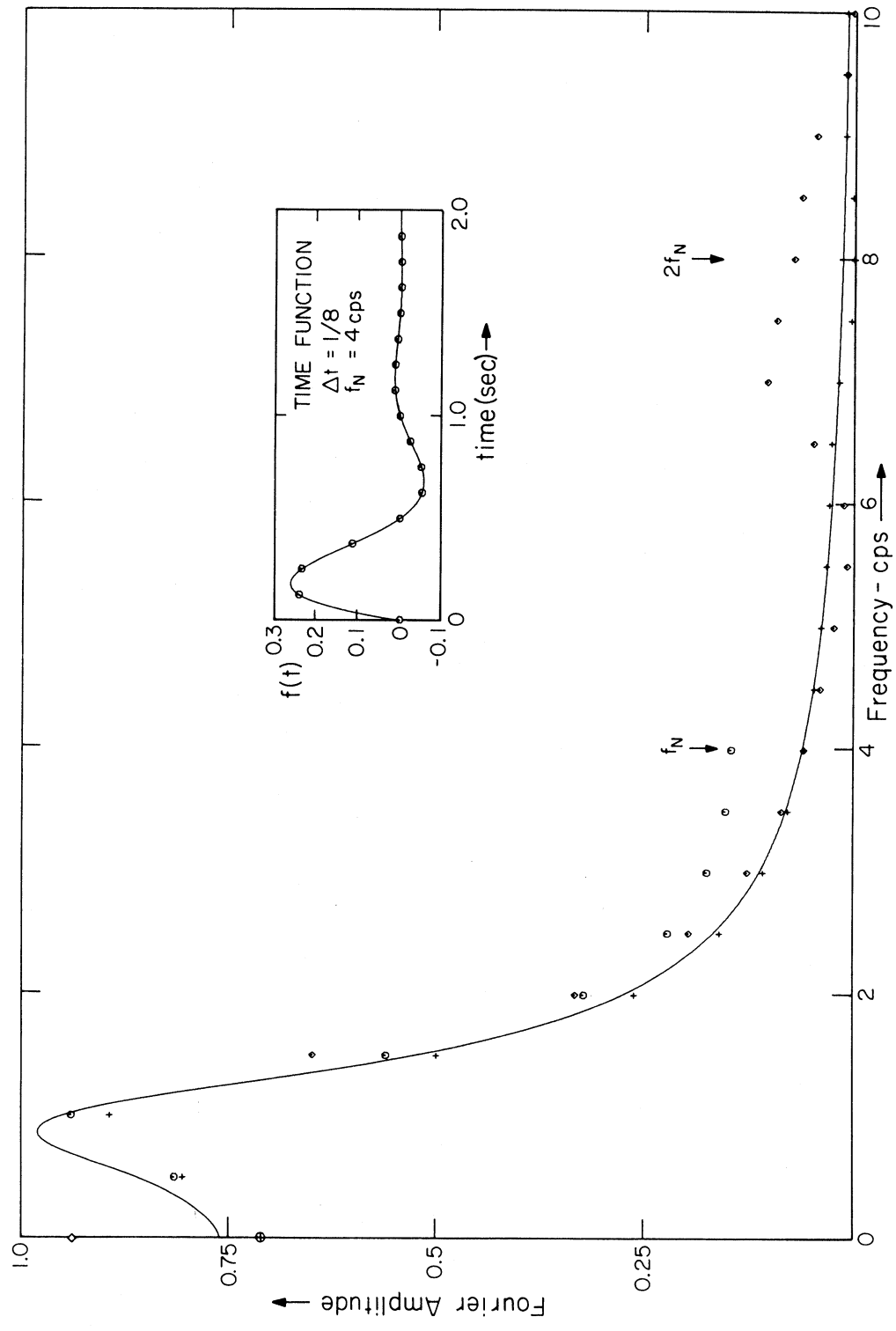

Figure 4a. The sampled time function and the Fourier amplitude spectra computed using the FFT (open circles), the linear interpolation technique (crosses), parabolic splines (diamonds), and the exact analytical method (solid line). All Fourier Amplitude Spectra have been normalized to unity.

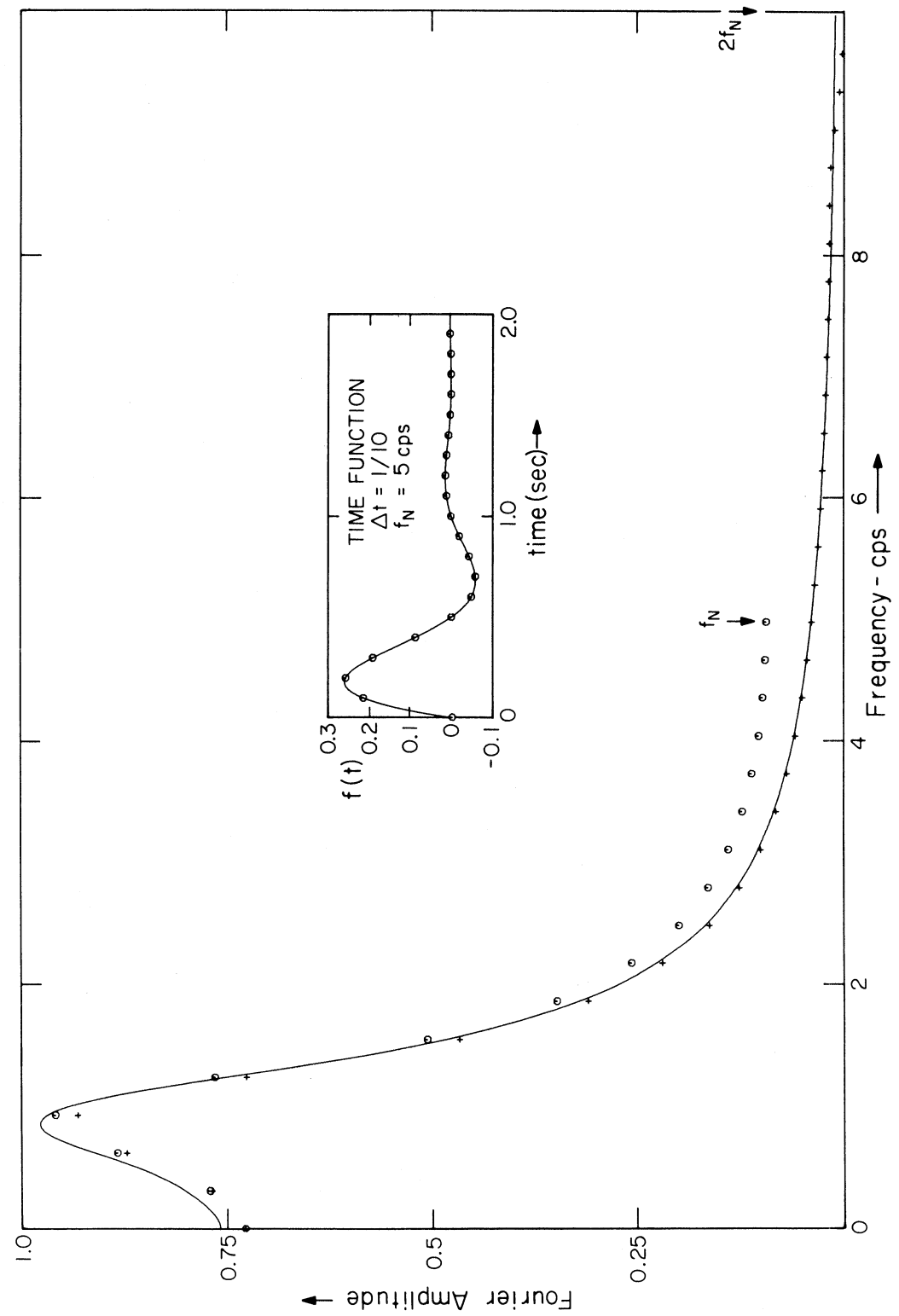

Figure 4b. The sampled time function and the Fourier amplitude spectra computed using the FFT (open circles), the linear interpolation technique (crosses), parabolic splines (diamonds), and the exact analytical method (solid line). All Fourier Amplitude Spectra have been normalized to unity.

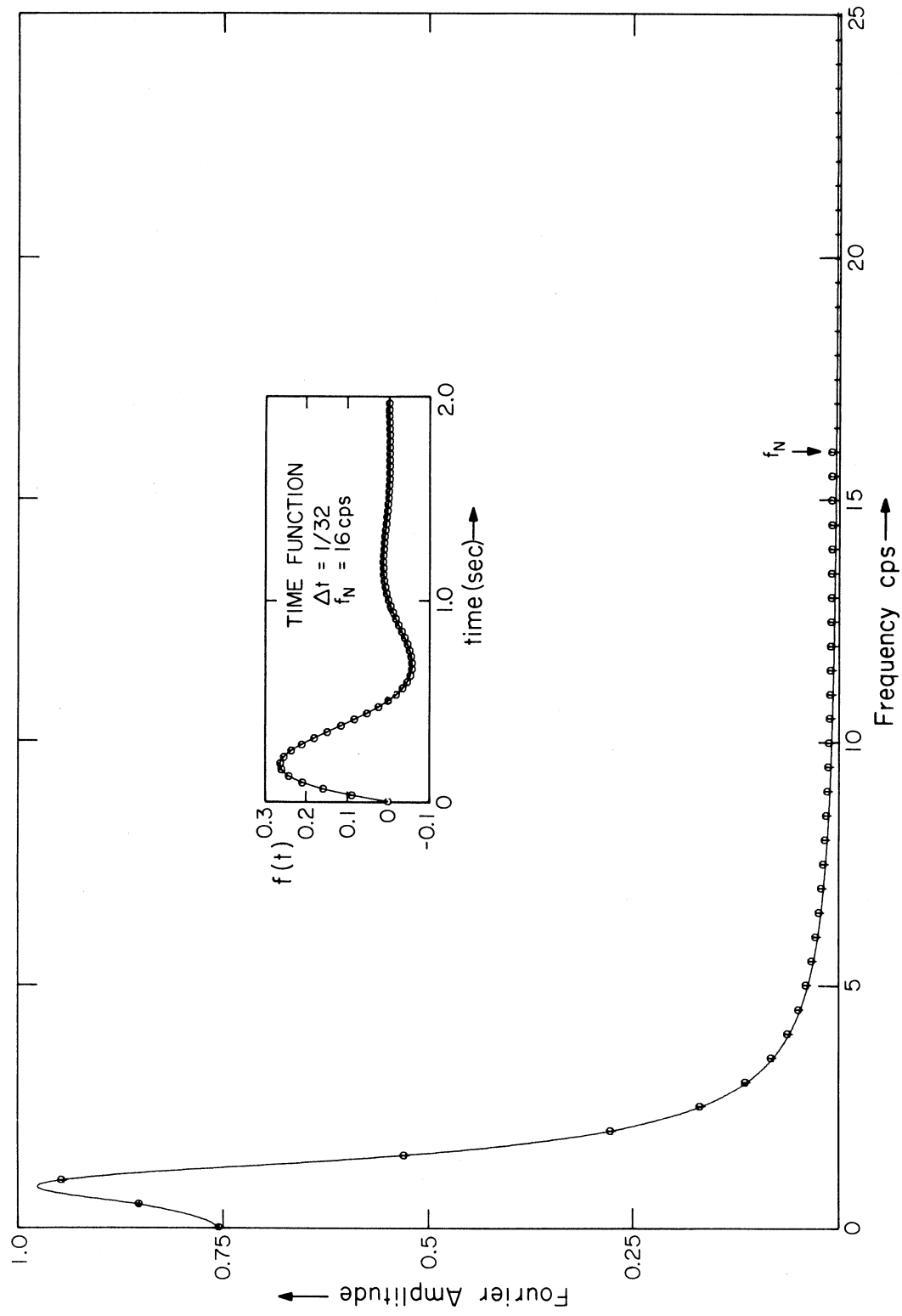

Figure 5a. The sampled time function and the Fourier amplitude spectra computed using the FFT (open circles), the linear interpolation technique (crosses), parabolic splines (diamonds), and the exact analytical method (solid line). All Fourier Amplitude Spectra have been normalized to unity.

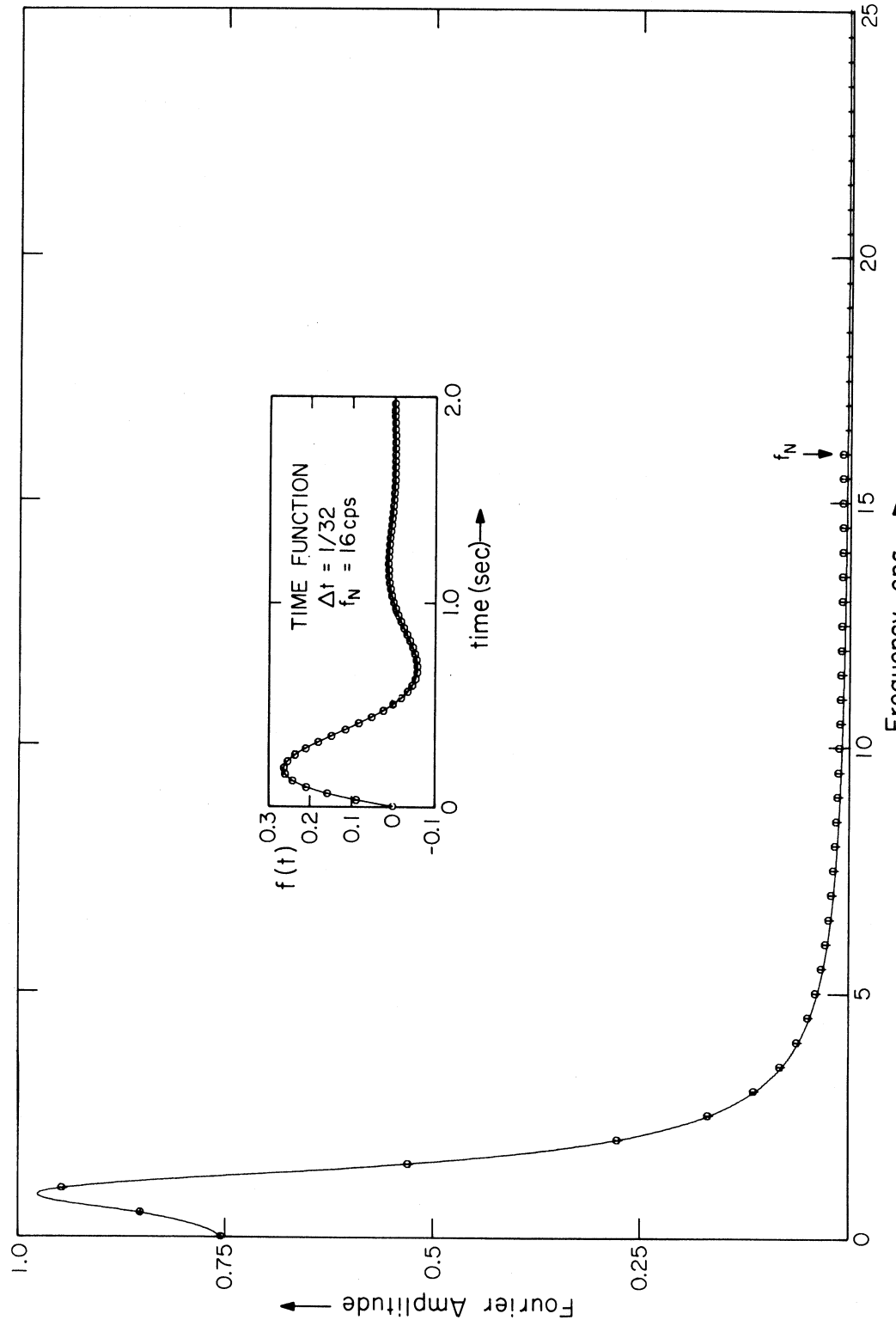

Figure 5b. The sampled time function and the Fourier amplitude spectra computed using the FFT (open circles), the linear interpolation technique (crosses), parabolic splines (diamonds), and the exact analytical method (solid line). All Fourier Amplitude Spectra have been normalized to unity.

The figures show a comparison between the exact transform (the solid line), calculated by using equation (24), the discrete Fourier transform (the open circles), computed by using the FFT algorithm, and the Fourier transform of the function obtained by a linear interpolation between the digital samples (the crosses), computed by using equation (11). Figure 4(a) also includes the transform (diamonds) obtained of the parabolic spline function [equations (15) and (16)]. As seen from the figure, the Fourier transform of the parabolic spline function does not give a close approximation to the exact transform. On the other hand, the linear spline transform and the exact curve are in close agreement, especially at the higher frequencies. The errors caused in the low frequency region of Figure 4(a) are a consequence of the fact that the number of sample points used to represent the function was "insufficient." This illustrates the necessity of using a dense spacing of points to represent the time function.

As the sampling interval Δt is reduced, the difference between the exact and the computed values reduces. Tables 1 and 2 show the errors obtained using the FFT and the linear spline (and for the case of $\Delta t = \frac{1}{8}$, the parabolic spline as well). We observe that the FFT error oscillates about the exact value being a minimum at 1 cps where most of the energy of the wave is concentrated. However, the error in using the FFT increases very rapidly rising to about 140% at the Nyquist frequency. The errors in using the FFT become significant beyond about half the Nyquist frequency, thus making the FFT approach accurate up to frequencies about $\frac{1}{2}$ to $\frac{1}{3}$ of the Nyquist frequency. The linear spline on the other hand shows a gradually decreasing

error up to Nyquist frequency. Beyond that, the error rises, reaching a value of 100% at twice the Nyquist frequency. This corresponds to the first zero of the envelope function $\overline{S^{(1)}(\omega)}$.

Also, as seen from the tables, the linear spline interpolation requires a smaller number of data points for an accurate definition of the function. To get the same degree of accuracy, the FFT requires the sample spacing Δt to be roughly a third to a half of that required by the linear spline. Though this result would depend on the actual character of the function g(t), this is physically understandable in view of the fact that in the linear spline case more information on the nature of function is being injected into the time function model. Such a linear interpolation will undoubtedly be good when the points are spaced sufficiently close to one another.

Whereas the Discrete Fourier Transform (which is computed by the FFT algorithm) prescribes an upper frequency limit beyond which no further information can be recovered, the linear spline time function, being continuous, has a nonperiodic frequency spectrum. The FFT algorithm can, however, still be used to calculate these higher frequency components, making use of the period nature of the Discrete Fourier Transform. It is improbable that an upper frequency limit for the linear spline case can be found independent of the function g(t). However, at the first zero of the envelope function $S^{(1)}(t)$ corresponding to twice the Nyquist frequency, the error becomes 100%. The tables indicate, that at least up to the Nyquist frequency and perhaps out to frequencies as high as 1.25 to 1.5 times the Nyquist

TABLE 1

$\Delta t = 1/8$, $f_N = 4$ cps					$\Delta t = 1/10$, $f_N = 5$ cps			
		% Error*					% Error*	
f_{cps}	f/f_N	FFT	Linear Spline	Parabolic Spline	f_{cps}	f/f_N	FFT	Linear Spline
0.0	0.0	-6.33	-6.33	24.6	0.0	0.0	-4.04	-4.04
0.5	0.125	-5.08	-6.30	24.6	0.625	0.125	-2.7	-4.02
1.0	0.25	-1.24	-6.2	23.7	1.25	0.25	1.12	-3.97
2.0	0.5	16.1	-5.82	19.9	1.875	0.375	8.06	-3.87
3.0	0.75	54.4	-5.03	12.06	2.5	0.5	18.78	-3.71
4.0	1.0	138.25	-3.43	-3.4	3.125	0.625	34.5	-3.49
5.0	1.25	--	0.07	-38.2	3.75	0.75	57.41	-3.19
6.0	1.5	--	9.5	-53.5	5.0	1.0	141.45	-2.14
7.5	1.875	--	-78.7	451.6	5.625	1.125	--	-1.23
8.0	2.0	--	-100.0	388.7	6.25	1.25	--	0.181
					7.5	1.5	--	6.7
					8.75	1.75	--	19.8
					9.375	1.875	--	-63.8
					10.0	2.0	--	-100.0

* $\text{Error} = \dfrac{\text{Computed Value - Exact Value}}{\text{Exact Value}}$

TABLE 2

$\Delta t = 1/16$ $f_N = 8$ cps				$\Delta t = 1/32$ $f_N = 16$ cps			
		% Error*				% Error*	
f_{cps}	f/f_N	FFT	Linear Spline	f_{cps}	f/f_N	FFT	Linear Spline
0	0	-1.57	-1.57	0.0	0.0	-0.39	-0.39
0.5	0.0625	-1.26	-1.57	1.0	0.0625	-0.07	-0.39
1.0	0.125	-0.297	-1.57	2.0	0.125	0.89	-0.39
2.0	0.25	3.67	-1.54	2.5	0.15625	1.63	-0.39
				3.0	0.1875	2.54	-0.39
3.0	0.375	10.719	-1.50	4.0	0.25	4.8	-0.38
4.0	0.5	21.585	-1.44	5.0	0.3125	8.0	-0.38
5.0	0.625	37.51	-1.35	6.0	0.375	11.9	-0.37
6.0	0.75	60.59	-1.23	7.0	0.4375	16.9	-0.368
7.0	0.875	94.30	-1.06	8.0	0.50	22.9	-0.36
8.0	1.0	144.7	-0.814	9.0	0.5625	30.19	-0.35
9.0	1.125	--	-0.447	10.0	0.625	38.94	-0.33
10.0	1.25	--	0.117	12.0	0.75	62.1	-0.308
12.0	1.5	--	2.79	14.0	0.875	95.8	-0.268
14.0	1.75	--	16.52	16.0	1.0	146.2	-0.2
15.0	1.875	--	-0.08	20.0	1.25	--	0.03
16.0	2.0	--	-100.0	24.0	1.531	--	0.877
				28.0	1.75	--	4.35
				30.0	1.875	--	18.26
				32.00	2.0	--	-100.0

* $\text{Error} = \dfrac{\text{Computed Value - Exact Value}}{\text{Exact Value}}$

frequency, the accuracy of the linear spline approximation is very high. However, serious distortion sets in beyond about 1.75 times the Nyquist frequency.

DISCUSSION AND CONCLUSIONS

In this paper an attempt has been made to utilize the FFT algorithm to calculate the Fourier Transforms of continuous, piecewise differentiable functions. Particular attention has been devoted to linear interpolation between digital data samples. This has been done for two reasons: firstly, in the optical mechanical digitization process,[3] the operator picks out points such that the function between them can be well described by a straight line; and secondly, it corresponds to the simplest polynomial whose study leads to a better understanding of Fourier Transforms involving higher order polynomial interpolations. Some of the characteristics of such interpolative schemes as applicable to the study of seismic signals can be summarized as follows:

(1) The Fourier transform of a discrete time series has a limiting frequency $\omega_N (= \frac{\pi}{\Delta t})$ beyond which the transform amplitudes do not remain independent. The interpolated function is a continuous one and does not have, in general, a periodic spectrum as does the discrete time series. Unlike the DFT, therefore, there appears to be no theoretically determinable limiting frequency, ω_N, independent of the nature of the function, g(t).

(2) The numerical example studied indicates two features. (1) The transform of the linearly interpolated function agrees well with the exact transform out to frequencies beyond the Nyquist frequency (about 1.25 to 1.5 times the Nyquist limit), whereas the FFT shows serious distortion beyond about half the Nyquist frequency. (2) To achieve a given level of overall amplitude transform accuracy, the number of samples required to be used by the FFT is about two or three times that required to be used if the linear interpolation scheme represented by equation (11) is utilized.

(3) The Fourier transform of the discrete time signal differs from that of the spline interpolated time signal by only a frequency dependent multiplicative factor. The algorithm for determining this transform is therefore simple and essentially employs the FFT. Since the number of required computer operations grows from $N\log_2 N$ to only $N\log_2 N+\frac{N}{2}$, higher accuracy is achieved with only minor increase of computer time.

The transfer function of a system calculated at the frequencies $\omega_j = \frac{2\pi j}{N\Delta t}$ by the FFT approach remains unaffected by the nature of the interpolative scheme used provided that the same scheme is used for both input and output functions and provided both records are sampled at equal time intervals Δt. For example, if the D. F. T. of the input function $x(t)$ is $X(\omega_j)$ and the D. F. T. of the output function $y(t)$ is $Y(\omega_j)$ then

$$\text{Transfer Function } (\omega_j) = \frac{A(\omega_j)Y(\omega_j)}{A(\omega_j)X(\omega_j)}$$

$$= \frac{Y(\omega_j)}{X(\omega_j)}$$

where $A(\omega_j)$ depends on the type of interpolation used.

Thus, linear interpolation, though it improves the actual accuracy of the transform estimate, does not improve the estimate of the transfer function.

(5) The response of an oscillator to strong ground shaking is nowadays usually computed by using an algorithm which is based on the direct integration of the Duhamel integral.[5] This algorithm assumes that the time function can be well represented between the digital data points by straight line segments. A comparable Fourier transform approach would then necessitate the use of equation (11) in the computation of the transform. Often the zero damping response spectrum computed by the Duhamel integral approach is compared with results from the FFT. Here again a meaningful comparison would require the transform of the linearly interpolated time function [as given by equation (11)] to be used.

ACKNOWLEDGMENTS

This research was supported by a grant from the National Science Foundation.

REFERENCES

1. Cooley, J.W., and J. W. Tukey, "An algorithm for the machine calculation of complex Fourier series," Math. Comput., 19, pp. 297-301, 1965.

2. Bergland, G. D., "A guided tour of the Fast Fourier Transform," IEEE Spectrum, 6, pp. 41-52, 1969.

3. Bracewell, R., The Fourier Transform and its Applications, McGraw Hill, Inc., New York, 1965.

4. Trifunac, M. D., F. E. Udwadia and A. G. Brady, "Analysis of errors in digitized strong-motion accelerograms," Bull. Seism. Soc. Amer., 63, pp. 157-187, 1973.

5. Nigam, N. C., and P. C. Jennings, "Digital calculation of response spectra from strong-motion earthquake records," Earthquake Engineering Research Laboratory, California Institute of Technology, Pasadena, 1968.