1985

# A general pattern recognition technique for open curves /

Charles Constantine Gumas
*Lehigh University*

# A GENERAL PATTERN RECOGNITION TECHNIQUE FOR OPEN CURVES

by

Charles Constantine Gumas

A Thesis
Presented to the Graduate Committee
of Lehigh University
in Candidacy for the Degree of
Master of Science
in
Computer Science and Electrical Engineering

Lehigh University
1985

This thesis is accepted and approved in partial fulfillment of the requirements for the degree of Master of Science in Computer Science and Electrical Engineering.

May 21, 1985

(date)

Bruce D Fritchman

Professor in Charge

Eric D. Thompson

Chairman of Department

# Acknowledgements

I would like to thank Professor Donald Rockwell of the Mechanical Engineering Department for our many thought provoking discussions, for his support, and for providing me with access to his department's VAX 11/780. I would also like to thank my thesis advisor Professor Bruce D. Fritchman for his helpful and willing advice. In addition, I thank the CSEE Department secretaries, Mary Morrison and Terri Ziebarth, for their kindness and willingness in helping me. Finally, I thank Donna Reiss for patiently typing this thesis during the mad rush of finals.

# Table of Contents

## Table of Contents (continued)

# List of Notations

| Symbol | Description |
|---|---|
| $F$ | Fourier Transform |
| $F^{-1}$ | Inverse Fourier Transform |
| $f(t)$ | Continuous time signal |
| $f_T(t)$ | Continuous time, periodic signal with period T |
| $f(n)$ | Sequence (possibly representing samples of a continuous time signal) |
| $f_N(n)$ | Periodic sequence with period N |
| $F(\omega)$ | Spectral Density as a function of $\omega$ (or $\tilde{\omega}$) |
| $F(k)$ | Fourier Coefficients |
| $F_M(k)$ | Fourier Coefficients of a sequence f(n) where the coefficients are calculated using the M-th root of unity. |
| $\omega$ | Frequency variable with units of radians/second |
| $\tilde{\omega}$ | An angle with units of radians. Usually related to $\omega$ by $\tilde{\omega} = \omega S$, where S is the sampling period. |
| $T$ | Total duration of a signal |
| $T_S$, $S$ | Sampling period. |
| $Sa(x)$ | Sampling function which equals sin(x)/x. |

ABSTRACT: This paper presents a complete approach to pattern recognition problems for discrete, digital, open curves. An assumption is made that the patterns are completely characterized by a curve connecting two points of the pattern. Algorithms (i.e. for curve tracking) are presented that create the curves needed for each pattern, and methods for normalizing, reconstructing, averaging, and comparing the similarity of any number of patterns are discussed.

Special topics are introduced that clarify the relationship between the pattern recognition problem and methods of signal processing. Basic signal theory, interpolation and decimation, parameterization of discrete signals, linear predictive coding, and an introduction to Fourier Descriptors are among the topics of review.

Finally, a specific application of these ideas to the recognition and distinction of discrete patterns of hydrogen bubble timelines is discussed. Actual patterns are tracked, parameterized, reconstructed, and compared. The data compression and reduction in calculations is shown to be significant in comparison with other methods.

## Introduction:

This paper discusses a pattern recognition scheme designed for the solution of a specific problem. The methods of analysis, however, are presented in a general context and are applicable to a wide range of problems (eg. image, speech, or other signal processing). In this section, we discuss a specific application of our methods to the study of timeline patterns. In the following sections, we first introduce general signal processing/pattern recognition concepts and then discuss our specific application.

In figure I.1, a digitized image from a fluid dynamics experiment [25 ] is shown. The image has dimension 211x165 pixels and the intensity of each pixel is quantized to eight bits (256 grey levels). To obtain I.1, the unprocessed digitized image (not shown) was processed by a number of (linear) convolutional filters and a (nonlinear) thresholding operation. The result, similar to I.1 but with 256 grey levels, is a bandpass filtered version of the original unprocessed image [see 7,8,9]. Because I.1 is a black and white version of the filtered image, the resulting quantization is 1 bit. Note, however, that all the images presented in this paper are actually quantized to eight bits and that generally the algorithms presented are applicable to the multi-level case.

Figure I.1 shows a typical pattern encountered in the study of fluid dynamics. The flow direction is from left to right, and the triangular region is an obstruction in the flow. The flow is seeded with hydrogen bubbles [ref. 25] that form the hydrogen bubble timelines. The timelines appear as dark regions in I.1, but they are actually regions of high intensity in the original images. As the timelines impinge upon the object, they are swept around it and form patterns that characterize the flow. It is desired to efficiently characterize the flow patterns from an analysis of the timelines.

Because the dimension of the image is 211x165, 211x165x8 (=278,520) bits are required to completely describe it in the form shown. To reduce this number, one might employ a region thinning algorithm to produce an image similar to Fig. I.2. Here, the algorithm reduces the thickness of each timeline to one pixel while (generally) maintaining the connectivity of the regions. During the reduction, it also chooses the maximum intensity center of each timeline. The image can then be totally described by 1 bit quantization with 211x165x1 (=34,815) bits, which is a reduction by a factor of eight. The resulting image, however, does not suggest a convenient way to quantitatively characterize the flow.

If the flow pattern is (almost) completely characterized by a few timelines, then the number of bits required to describe the flow is drastically reduced, and a method to quantitatively characterize the flow is suggested. When two timelines suffice, and if each timeline is of (average) length 100 pixels, then we find that approximately 2x100x0.1 (=20) Fourier coefficients need to be stored to sufficiently characterize the flow pattern. Of course, depending on the specific application, this number will change somewhat. Since the Fourier coefficients are complex, and if they are stored with eight bits of precision, then the total number of bits needed to characterize the flow is 20x2x8 (=320). The reduction is nearly three orders of magnitude from the original image. Furthermore, in comparing the spectra of two timelines, one need only compare coefficients of the same frequency. We find [sec. 3.2.2] that approximately 10% of the Fourier coefficients are needed to discern patterns of the same general shape.

If our assumption is valid that two (a few) timelines suffice to describe the flow pattern, what is the motivation for using Fourier techniques to compare timelines? This question is answered by consideration of the alternatives, eg. geometric techniques such as spline fitting, etc. To accurately describe a curve by a spline fit, a number of points on its arclength must be supplied for the algorithm. The variables are the number of points given to the algorithm and the order of the polynomial approximation. From the Nyquist criterion (sec. 1.2.1), the number of samples needed is a function of the bandwidth of the curve. If we assume that only 10% of the (unique) Fourier coefficients are required, then the number of samples needed to compute them is 20% of the total number of samples. Hence, the spline fit will need this many points for the approximation. Furthermore, the polynomial approximation should be at least third order (or more), and the number of computations necessarily becomes large. Finally, it is necessary to normalize the representation of each pattern so that it can be compared, averaged, and reconstructed with other patterns. For these reasons, Fourier description of patterns is appealing.

In the next section (section 1), fundamental signal concepts needed for the understanding of our pattern recognition scheme are reviewed. In section 2, the procedures for obtaining a parameterization of each timeline, i.e. the relative tangent angle function, are presented along with the fundamentals of linear prediction (sec. 2.3). In section 2.4, a short discussion on the distortion of discrete curves is presented. A distortion algorithm is presented and is used later in analyzing the usefulness of the similarity measure presented in sec. 3.2.2.

In Section 3, Fourier Descriptors are introduced, and our implementation of them is explained. Sections 3.2.2 and 3.2.3 present the preliminary results of our method.

## Section 1 Signal Analysis: Basic Principles

We begin with a review of some of the basic principles of signal analysis as they pertain to this paper [references 1 through 6].

### Section 1.1: Transforms of Signals and Sequences

Let us begin with an aperiodic signal f(t) and assume that is nonzero for some duration T and zero elsewhere [Fig. 1.1a]. By using the Inverse Fourier Transform (IFT), we can express this function as an integral of an eternal exponential times the spectral density F(ω).

$$f(t) \overset{\Delta}{=} F^{-1}[F(\omega)] = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{j\omega t}d\omega \tag{1.1}$$

The spectral density F(ω) is given by the Fourier Transform (FT) of f(t).

$$F(\omega) \overset{\Delta}{=} F[f(t)] = \int_{-\infty}^{\infty} f(t)e^{-j\omega t}\,dt \tag{1.2}$$

When f(t) is a real signal, F(ω) will be a complex function with even magnitude and odd phase. Furthermore, we may note that the sign of the exponent, $\pm j\omega t$, in each of these definitions is purely a matter of convention (as long as they have opposite values in 1.1 and 1.2). An example of a real signal is plotted in Fig. 1.1abc along with the magnitude and phase of its spectral density.

Suppose that we construct a periodic function with period T that is simply a repetition of functions f(t), [Fig. 1.2a].

$$f_T(t+mT) = f(t) \qquad t_0 \leq t \leq t_0+T$$
$$m=0, \pm 1, \pm 2, \ldots \qquad (1.3)$$

By the Fourier Series (FS) expansion of $f_T(t)$, we can express $f_T(t)$ as an infinite sum of eternal exponentials.

$$f_T(t) \triangleq \sum_{k=-\infty}^{\infty} F(k)e^{j\omega_0 kt} \; ; \; \omega_0 = 2\pi/T \qquad (1.4)$$

Here, $\omega_0$ is the fundamental frequency, and the kth harmonic is sometimes written $\omega_k=\omega_0 k$. F(k) is the k-th Fourier coefficient and is given by

$$F(k) = \frac{1}{T}\int_0^T f_T(t)e^{-j\omega_0 kt}\, dt \qquad (1.5)$$

The interval of integration can be taken as any interval of length T, and for reasons of symmetry, the interval [-T/2, T/2] is a common choice. Again, for real signals $f_T(t)$, the Fourier coefficients are complex, have magnitudes that are even functions of k, and have phases that are odd functions of K. This requires F(k)=F*(-k), which expresses the k-th coefficient as the complex conjugate of the -(k-th) coefficient. The spectrum of $f_T(t)$ is a plot of F(k) versus k, or equivalently, $\omega_k$ versus $\omega$. We must note that the spectrum only exists at integer multiples of $\omega=\omega_0$, and it is undefined at other values of $\omega$, [Fig. 1.2bc].

We can imagine, now, that we take samples of F(t) at fixed intervals spaced $T_S$ seconds apart [Fig. 1.3]. In this paper, we ignore the issues of how the sampling is done [see 6, chap. 2], and we procede to analyze the resulting discrete time sequence $f_s(n)$ [Fig. 1.3b], where

$$f_s(n) \triangleq f(nT_s) \quad n=0, \pm 1, \pm 2 \; \ldots$$

By definition, $f_s(n)$ will be undefined for noninteger values of n. For the purposes of this discussion, we assume that our sampling will yield $N=T/T_s$ samples and that our sampler is somehow designed so that it is synchronized with the signal such that T is exactly equal to $NT_s$. We say that the sampling frequency is $\omega_s=2\pi/T_s$. Furthermore, we drop the subscript s and simply recognize that f(n) is a sequence of numbers that are nonzero only for $0 \le n \le N-1$. We can then express f(n) as an integral of an eternal exponential times a spectral density function $F(\tilde{\omega})$. The relation is given by the Inverse Fourier Transform (IFT) for sequences,

$$f(n) \triangleq \frac{1}{2\pi} \int_{-\pi}^{\pi} F(\tilde{\omega}) e^{j\tilde{\omega}n} \, d\tilde{\omega} \tag{1.6}$$

Note that the limits of integration are $[-\pi,\pi]$, but any range of length $2\pi$ suffices. The spectral density is given by the Fourier Transform [FT] for sequences,

$$F(\tilde{\omega}) = \sum_{n=-\infty}^{\infty} F(n) e^{-j\tilde{\omega}n} \tag{1.7}$$

We can distinguish the spectral density of the FT for sequences (eqn. 1.7) from that of the FT for signals (eqn. 1.2) by recognizing that $\tilde{\omega}$ is purely an angle (radians), while $\omega$ is an angular frequency (radians/sec). They are related by $\omega=\tilde{\omega}/T_s$ where $T_s$ is a sampling period (seconds). Also, because only integer values of n are allowed,

$$e^{j(\tilde{\omega}n+2\pi n)} = e^{j\tilde{\omega}n}$$

so that

$$F(\tilde{\omega}+2\pi n) = F(\tilde{\omega})$$

for all integer n, and $F(\tilde{\omega})$ is seen to be periodic with period $2\pi$. Hence, in addition to being a complex function with even magnitude and odd phase, $F(\tilde{\omega})$ is periodic. Combining these symmetry relations, we note that if values of the spectral density are known for $\tilde{\omega}\varepsilon[0,\pi]$, then $F(\tilde{\omega})$ is known for all $\tilde{\omega}$.

Because $F(\tilde{\omega})$ is continuous and periodic, and because f(n) is discrete and aperiodic, we should expect that there is a Fourier Series relating the two. Examining equations 1.6 and 1.7, we see their duality with equations 1.5 and 1.4.

Before proceeding, a final comment about the FT (eqn. 1.7) is in store. Because f(n) is nonzero for only N terms, we may write

$$F(\tilde{\omega}) = \sum_{n=n_0}^{n_0+N-1} f(n)e^{-j\tilde{\omega}n}$$

$$= A \sum_{n=0}^{N-1} f'(n)e^{-j\tilde{\omega}n}$$

where $A = e^{-j\tilde{\omega}n_0}$, and $f'(n) = f(n+n_0)$. This is equivalent to a shifting of the n axis and the introduction of an appropriate phase shift. Usually we will assume $n_0=0$ so that A=1. Figs. 1.3b-f show examples of a sequence and a spectral density.

Just as the Fourier Transform of aperiodic signals was extended into a Fourier Series for periodic signals (eqns. 1.1,2 into 1.3,4) the Fourier Transform for aperiodic sequences is extended into the Discrete Fourier Series for periodic sequences (eqns. 1.6,7,8 into 1.9,10). First, we construct $f_N(n)$ by either sampling the signal $f_T(t)$ or by repeating the sequence f(n). Choosing the second method [Figs. 1.4a,b],

$$f_N(n+mN) = f(n) \qquad 0 \le n \le N-1$$
$$m=0, \pm 1, \pm 2, \ldots$$

The Inverse Discrete Fourier Series (IDFS) of the periodic sequence $f_N(n)$ is then given by

$$f_N(n) \triangleq \sum_{k=0}^{N-1} F_N(k)\, e^{j\tilde{\omega}_0 kn} \tag{1.9a}$$

where $\tilde{\omega}_0$ is the fundamental angular (radian) spacing $\tilde{\omega}_0 = 2\pi/N$. The exponential is recognized as a power of the N-th root of unity. This is commonly written,

$$W_N \triangleq e^{j\tilde{\omega}_0} = e^{j2\pi/N}$$

so that eqn. 1.9a becomes

$$f_N(n) \triangleq \sum_{k=0}^{N-1} F_N(k)\, W_N^{kn}$$

The Discrete Fourier Series (DFS) for periodic sequences is

$$\begin{aligned} F_N(k) &= \frac{1}{N} \sum_{n=0}^{N-1} f_N(n) e^{-j\tilde{\omega}_0 kn} \\ &= \frac{1}{N} \sum_{n=0}^{N-1} f_N(n)\, W_N^{-kn} \end{aligned} \tag{1.10}$$

Analogous to the Fourier Series coefficients of eqn. 1.5, these coefficients represent individual frequency components. The spectrum of a periodic sequence $f_N(n)$ is a line spectrum defined only at $\tilde{\omega}=\tilde{\omega}_0 k$ for integer k. In contrast to the coefficients of eqn. 1.5, and similar to the spectral densities of eqns. 1.7,8, the coefficients $F_N(k)$ are periodic with period N.

$$F_N(k) \equiv F_N(\tilde{\omega}_0 k) = F_N(\tilde{\omega}_0(k+mN)) \qquad \begin{array}{l} k=0, \pm 1, \pm 2, \ldots \\ m=0, \pm 1, \pm 2, \ldots \end{array}$$

They also retain the symmetry relation $F_N(k) = F_N^*(-k)$. Hence, the symmetry and periodicity relations reveal that, if N is even, then knowing (N/2+1) sequential values of $F_N(k)$ suffices to evaluate $F_N(k)$ for all k. If N is odd, then only (N-1/2+1) sequential coefficients need to be known. Figs. 4c,d show examples of DFS spectral coefficients.

## Section 1.2: Spectral Analysis

In this section, we discuss the Nyquist criterion, interpolation, and decimation. We conclude the section with a discussion on comparisons and averages of sequences.

### Section 1.2.1 Nyquist Criterion

Suppose that our original function f(t) is bandlimited so that $F(\omega)=0$ outside the range of $\omega\varepsilon[-W,W]$ as shown in Figure 1.5a [unless specified otherwise, the spectra given by $F(\omega)$ will be illustrated by their magnitudes only ]. By sampling $f(t)$ at a rate $\omega_1$, we generate a sequence $f_1(n)$ that has $N_1$ samples. We introduce a new notation for the sampling period, which we designate by S. The sampling period $S_1$ is given by $S_1=2\pi/\omega_1$, and since the signal has nonzero duration T, we write $T = N_1S_1$.

If the sampling rate $\omega_1$ satisfies the Nyquist Criterion,

$$\omega_1 \geq 2W \tag{1.11}$$

we are assured that the FT of $f_1(n)$ will be equal to the spectral density $F(\omega)$ of $f(t)$. This is true only in the range $\tilde{\omega}\varepsilon[-\pi,\pi]$, and the two spectra are alike except for a constant scaling factor. When the sampling rate satisfies the Nyquist criterion, it is possible to exactly reconstruct the original continuous time signal $f(t)$ from its samples, $f_1(n)$. This is shown in Appendix A.

On the other hand, when the sampling frequency is less than 2W, so that it does not satisfy the Nyquist criterion, the spectra $F(\omega)$ and $F_1(\tilde{\omega})$ may be highly dissimilar. Figures 1.5a,b,c, show the original signal, the spectrum of the "properly sampled" signal (or "oversampled"), and the spectrum of the "improperly sampled" signal (or "undersampled"). For the undersampled case, we see that $F(\tilde{\omega})$ is periodic but that each period does not look like $F(\omega)$. Instead, the spectrum is "aliased". That is, because $\omega_1<2W$ or equivalently, $\tilde{\omega}_1<2WS_1$, the periodic placement of $F(\omega)$ at multiples of $2\pi$ restuls in overlapping.

In the overlapping regions, the spectra add together, and the spectral density $F(\tilde{\omega})$ is distorted. As shown in appendix A, when aliasing is present in the spectrum, we cannot reconstruct the original continuous time signal from it. The reconstructed signal contains errors that depend on the amount of aliasing in the spectrum. We can, however, always obtain the exact values of $f_1(n)$ from its spectrum. This is simply a consequence that the IFT is unique and totally independent of the existence of aliasing.

Suppose that we have a sequence $f(n)$, regardless of where it comes from. As in eqn. 1.8, we write the FT as

$$F(\tilde{\omega}) = \sum_{n=0}^{N-1} f(n)e^{-j\tilde{\omega}n}$$

By letting $Z=e^{j\tilde{\omega}}$, we can write this as a Z-transform,

$$\begin{aligned} F(z) &\overset{\Delta}{=} F(e^{j\tilde{\omega}}) \\ &= F(\tilde{\omega}) = \sum_{n=-\infty}^{\infty} f(n)Z^{-n} \\ &= \sum_{n=0}^{N-1} f(n)Z^{-n} \end{aligned} \qquad (1.12)$$

where we assume $n_0=0$. Specifically, we see that the FT of an aperiodic sequence is exactly its Z-transform evaluated on the unit circle. Furthermore, we are free to evaluate $F(z)$ at as many points on the unit circle as we like, and this is true regardless of the length N of the sequence. If we choose M evenly spaced points on the unit circle, then $\tilde{\omega}=(2\pi/M)k$, and Z is the M-th root of unity raised to the K-th power. Denoting $Z_k$ by,

$$Z_k = e^{j(2\pi/M)k}$$

we see that $Z_k = Z_{K+M}$. Hence, there are only M unique values of $Z_k$ and likewise $F(Z_k)$.

Next, consider the following. Denote the M point Z-transform of an N point sequence by

$$F_M(Z_k) = \sum_{n=0}^{N-1} f(n) Z_k^{-n}$$

$$= \sum_{n=0}^{M-1} f(n) e^{-j \frac{2\pi}{M} kn} \qquad 0 \leq k \leq M-1 .$$

When f(n) is a periodic sequence with period M, denote it as $f_M(n)$, then the fundamental angle is $\tilde{\omega}_0 = 2\pi/M$, and

$$F_m(Z_k) = \sum_{n=0}^{M-1} f(n) e^{-j\tilde{\omega}_0 kn}$$

$$= F_m(k)$$

That is, the M samples of the Z-transform are equal to the coefficients of the Discrete Fourier Series, given in eqn. 1.10, but scaled by M. Let us now consider the following two cases.

Case 1. Suppose $N \leq M$. We can then define a periodic sequence of M points $f_m(n)$ by zero extending it,

$$\begin{aligned} f_m(n+\ell M) &= f(n) & 0 \leq n \leq N-1 \\ &= 0 & N \leq n \leq M-1 \end{aligned} \qquad (1.13)$$

From eqn. 1.10, we write

$$F_m(k) = \frac{1}{m} \sum_{n=0}^{m-1} f_m(n)e^{-j\tilde{\omega}_0 kn} \qquad 0 \le k \le m-1$$

and hence each $F_m(k)$ will be a DFS coefficient of a sequence that looks like f(n) zero extended to length M. Figure 1.6 shows an example of a zero extended sequence.

<u>Case 2.</u> Suppose N>M (for convenience we also assume that N<2M), then,

$$M\,F_m(Z_k) = \sum_{n=-\infty}^{\infty} f(n)\,Z_k^{-n}$$

$$= \sum_{n=0}^{N-1} f(n)Z_k^{-n}$$

$$= \sum_{n=0}^{m-1} f(n)Z_k^{-n} + \sum_{n=m}^{N-1} f(n)Z_k^{-n}$$

Because $Z_k^m = 1$, we can write this as

$$M\,F_m(Z_k) = \sum_{n=0}^{m-1} f(n)Z_k^{-n} + Z_k^m \sum_{n=0}^{N-M-1} f(n+M)Z_k^{-n}$$

$$= \sum_{n=0}^{N-m-1} [f(n)+f(n+m)]Z_k^{-n} + \sum_{n=N-m}^{m-1} f(n)Z_k^{-n} \qquad (1.14)$$

If we now define a periodic sequence $f_m(n)$ with period M by,

$$f_m(n) = f(n) + f(n+M) \qquad 0 \le n \le N-M-1$$

$$= f(n) \qquad N-M \le n \le M-1$$

then

$$F_m(k) = \frac{1}{M} \sum_{n=0}^{M-1} f_m(n) Z_k^{-n}$$

$$= \frac{1}{M} \sum_{n=0}^{m-1} f_m(n)\, e^{-j\tilde{\omega}_0 kn} \qquad 0\leq k\leq M-1$$

Again, this represents the M DFS coefficients of a periodic sequence $f_m(n)$, where $f_m(n)$ is a wrapped version of the N length sequence f(n). Fig. 1.11 shows an example of a wrapped sequence, where the above equation for $f_m(n)$ is used.

Finally, we comment that by "zero extending" a sequence we can calculate M>N DFS coefficients. By "wrapping" a sequence, we shorten it to calculate M<N DFS coefficients. Next, we use the duality of equations 1.9 and 1.10 to develop more interesting ideas.

## Section 1.2.2 Zero Extending and Interpolation

Consider a periodic sequence of length N. Then, the DFS coefficients are

$$F_N(k) = \frac{1}{N} \sum_{n=0}^{N-1} f_N(n) e^{-j\tilde{\omega}_1 kn} \qquad 0\leq k\leq N-1$$

where we use $\tilde{\omega}_1=2\pi/N$. Now consider the case, as in case 1 of the previous section,

$$f_m(n) = f_N(n) \qquad 0\leq n\leq N-1$$
$$= 0 \qquad N\leq n\leq M-1$$

Denoting $\tilde{\omega}_2=2\pi/M$, we can write

$$F_m(k) = \frac{1}{M} \sum_{n=0}^{m-1} f_m(n)e^{-j\tilde{\omega}_2 kn} \qquad 0\leq k\leq m-1$$

$$= \frac{1}{M} \sum_{n=0}^{N-1} f_N(n)e^{-j\tilde{\omega}_2 kn}$$

Whenever $\tilde{\omega}_2 k = \tilde{\omega}_1 \ell$, then

$$F_m(k) = \frac{N}{M} F_N(\ell) \;. \tag{1.16}$$

and this is true when,

$$\tilde{\omega}_2 k = \tilde{\omega}_1 \ell$$

$$= \frac{2\pi}{M} k = \frac{2\pi}{N} \ell$$

$$k = \frac{M}{N} \ell \;.$$

For example, when M=2N, every other coefficient in the $F_m(k)$ sequence equals one of the coefficients in the $F_N(k)$ sequence. Figure 1.6 shows this case. With the constraint that k and $\ell$ are integers, equation 1.16 is true for all ratios of M and N, M>N. The remaining coefficients of $F_m(k)$ lie amidst those that match with $F_N(k)$.

Presently, we assume that M/N is an integer and later we relax this constraint. For integer M/N, N coefficients match and M-N do not. We now determine that the M-N nonmatching coefficients are actually interpolations between the matching coefficients.

In Figure 1.7 we examine a portion of the spectra and concentrate on the region between $F_N(\ell)$ and $F_N(\ell-1)$. In general, the coefficient $F_N(\ell-1)$ will match the coefficient $F_M(\frac{M}{N}(\ell-1))$ and $F_N(\ell)$ will match

$F_M(\frac{M}{N}\ell)$. Hence, there are

$$\frac{M}{N}\ell - \frac{M}{N}(\ell-1) - 1 = \frac{M}{N} - 1 \qquad (1.17)$$

coefficients that are interpolation coefficients. We want to show that these interpolation coefficients lie on the same envelope as the coefficients that match.

When $\frac{M}{N} - 1$ is an integer, we wish to examine the coefficients given by $F_M(k+B)$ for B ranging from 1 to $\frac{M}{N} - 1$, where $K = \frac{M}{N}\ell$, for $\ell=0,1,\ldots,N-1$. We have,

$$F_M(\frac{M}{N}\ell + B) = \frac{1}{M}\sum_{n=0}^{M-1} f_M(n)e^{-j\frac{2\pi}{M}[\frac{M}{N}\ell + B]n}$$

$$= \frac{1}{M}\sum_{n=0}^{N-1} [f_m(n)e^{-j\frac{2\pi}{M}Bn}]e^{-j\frac{2\pi}{N}\ell.n} \qquad (1.18)$$

Denote the term in the brackets by f'(n), and denote $\gamma = \frac{2\pi B}{m}$, $\tilde{\omega}_o=2\pi/N$. Then, the above equation is recognized as the $\ell$-th coefficient of an N point DFS of the sequence f'(n). That is,

$$F_M(\frac{M}{N}\ell+B) = \frac{N}{M}F_N'(\ell\,\tilde{\omega}_o)$$

where

$$F_N'(\ell\,\tilde{\omega}_o) = \frac{1}{N}\sum_{n=0}^{N-1} f'(n)e^{-j\tilde{\omega}_o\ell n}$$

We know that the DFS coefficient above is related to the spectral density of the FT by

$$F_N'(\ell\,\tilde{\omega}_o) = \frac{1}{N}F'\,(\tilde{\omega}=\ell\,\tilde{\omega}_o)$$

Also, applying the frequency shifting property of the FT, $F'(\tilde{\omega}) = F(\tilde{\omega}-\gamma)$ since $f'(n) = f(n)e^{-j\gamma n}$. Combining the above relations, we have

$$Fm(\frac{M}{N}\ell + B) = \frac{1}{M}F(\tilde{\omega}_o\ell-\gamma) .$$

Since $k = \frac{M}{N}\ell$, this tells us that the K+B-th DFS coefficient lies on the same envelope as the DFS coefficients that match with $F_N(\ell)$.

These arguments are equally valid for all ratios of M and N. When M/N is not an integer, however, there will be fewer coefficients between $F_m(k)$ and $F_N(k)$ that match exactly. For example, if M=110 and N=100, then only those coefficients where $K=11/10\ \ell$ (integer $\ell$) will match. That is , k=11 matches with $\ell$=10, k=22 matches with $\ell$=20, etc. Now, however, between every two coefficients of $F_m(k)$ that match, there are ten (=N) that do not. So, equation 1.17 is still used, but B ranges from one to ten, and not all the values of $\ell$ will lead to integer values of K.

Consider the ramifications of what we have discovered. By simply appending zeroes to our (time) sequence, we have obtained more samples of the Z-transform, or equivalently, more DFS coefficients. When M is even, these coefficients are unique only in the range [0, M/2+1]. In this range, the angle (radians) $\tilde{\omega}$ ranges from $[0,\pi]$. In terms of absolute frequency, $\tilde{\omega}=\pi$ is equivalent to $\omega=\pi/S$ where S is the sampling period. Since the sampling frequency is $\omega_s = 2\pi/S$, $\tilde{\omega}=\pi$ corresponds to one half the sampling frequency. Hence, interpolating to obtain more coefficients in the range $\tilde{\omega}=[0,\pi]$ does not yield any more information about the signal $f(t)$. Instead, it reveals more of the detail of the spectrum of $f(n)$. So, we see that if $F(\tilde{\omega})$ is an aliased version of $F(\omega)$, this procedure does not help to "un-alias" it. This is made obvious by the fact that inverting any version of $F_M(\tilde{\omega})$ will simply produce the original sequence $f(n)$ with M-N zeroes appended to it. By adding zeroes we have not changed the sampling frequency nor its relation to the Nyquist criterion.

The utility of this procedure, however, is that we can now compare the DFT spectra of any two (or more) sequences, even if they have different lengths. Furthermore, we can do so on a coefficient by coefficient basis. The only assumption is that the sequences were obtained at the same sampling rate. We examine the case of different sampling rates in section 1.2.3. For the present, however, figure 1.8 introduces the notation that we use for interpolation in the frequency domain.

We have already mentioned the duality of the FT (or the FS, etc.), but we have not as yet reaped any benefits from it. Now , however, we examine the case where we calculate $F_N(k)$ from $f_N(n)$ and then append zeroes in the frequency domain, [Fig. 1.9].

Because the spectrum above $\tilde{\omega}=\pi$ is redundant, (i.e. loosely speaking) we append zeroes from $K=(N/2+1)$ to $K=M/2$. We then use the relations $F(k)=F(k)^*$ and $F(k+M)=F(k)$ to construct the rest of the spectrum. We denote this zero extended spectrum $F_M(k)$, and by performing an inverse DFT, we generate a sequence of length M, $f_M(n)$.

Now, by the duality of the DFT, we see that $f_M(n)$ matches with $f_N(\ell)$ whenever $n=\frac{M}{N}\ell$, and the rest of the values of $F_m(n)$ are interpolations. The case where M=2N is shown in Figure 1.9, but just as before (i.e. eqns. 1.16, 17, 18, etc.), M/N is not restricted to be an integer. Figure 1.10 shows the notation developed for interpolation in the (time) sequence domain.

Let's note the ramifications of this development. Given any set of DFS coefficients, we can zero extend the spectrum and obtain a sequence of length M, as in Figure 1.9. Furthermore, M can be any number greater than N. In effect, by making M large we can emulate a sequence that has been obtained from a sampler that has a sampling frequency as large as we would like. In fact, we can emulate a sampling frequency that approaches infinity. Hence, it would seem that the barrier of the Nyquist criterion has been broken!

But obviously, by zero extending our spectrum we are not adding any new information to the resulting sequence (or, as the number of zeroes approaches infinity, we get a continuous time signal). In fact, the only real information we are using comes from the original spectrum $F_N(k)$. If $F_N(k)$ is an aliased version of $F(\omega)$ (i.e., if $f_N(n)$ is not a properly sampled version of $f(t)$), then all versions of $F_M(k)$ will simply be obtained by separating the aliased portions of $F_N(k)$ by zeroes, and the aliasing persists.

Again, we recognize the duality of the situation with our previous case. For interpolation in the frequency domain, we thought of the sampling period remaining constant whereas the total sampling time $T_0$=NS or T=MS varied. For interpolation in the sequence domain, we can think of the total sampling time remaining constant, i.e. T=NS, whereas the sampling period S varies. Hence, for the interpolated sequence T=MS'=NS.

## Section 1.2.3 Wrapping and Decimation

At the end of section 1.2.1, we introduced a way to evaluate the Z-transform of a length N sequence at M points. When N>M, we defined a length M sequence by "wrapping" the coefficients $F_N(k)$ to obtain $F_M(k)$ (eqn. 1.15). We use the term wrapping because $F_M(k)$ is obtained by performing a modulo M operation on $F_N$,

$$F_M(k) = \sum_{(\ell=k)\text{mod } M} F_N(\ell) \tag{1.19}$$

That is, the sum is over all values of $\ell$ that equal K modulo M.

Figure 1.11 shows an example of the wrapping process and the resulting sequences obtained by performing the inverse transformations. It is clear from Figures 1.11a,b that the spectrum $F_M(k)$ is an aliased version of $F_N(k)$; hence, we may wonder if the sequence $f_M(n)$ will have elements equal to every other term in $F_N(n)$.

We now show that the decimated sequence, obtained by inverting the spectrum of eqn. 1.19 (which generalizes eqn. 1.14), will not have elements equal to those in the original sequence. Specifically, the sequence in figure 1.11d will not be made up of every other element in the sequence of figure 1.11c. We will find that the difference is due to the aliasing caused by the wrapping procedure of eqn. 1.19. First, we consider the case of decimation by an integer. Then we consider the general case N/M>1.

Suppose that $f_N(n)$ is decimated by a factor of two. Then, N=2M, and we would like $f_m(n)$ to have the property,

$$f_m(n) = f_N(2n) \qquad n=0,1,\ldots,m-1 \tag{1.20}$$

Note that the DFS expansion of $f_m(n)$ is

$$f_n(n) = \sum_{k=0}^{M-1} F_m(k)W_m^{kn}$$

and the DFS coefficients are given by

$$F_m(k) = \frac{1}{m}\sum_{n=0}^{m-1} f_m(n)W_m^{-kn} \qquad k = 0,1,\ldots,m-1$$

From eqn. 1.20, we substitute for $f_m(n)$,

$$F_m(k) = \frac{1}{m}\sum_{n=0}^{m-1} f_N(2n)W_m^{-kn}$$

and for $f_N(n)$ we substitute its DFS

$$F_m(k) = \frac{1}{m}\sum_{n=0}^{m-1} [\sum_{p=0}^{N-1} F_N(p)W_N^{2np}]W_m^{-kn}$$

$$= \frac{1}{M}\sum_{p=0}^{N-1} F_N(p) \sum_{n=0}^{m-1} W_N^{2pn}\, W_m^{-kn}$$

But now, since N=2M, $W_N^2=W_M$, and

$$F_m(k) = \sum_{p=0}^{N-1} F_N(p)\left[\frac{1}{m} \sum_{n=0}^{M-1} W_m^{(p-k)n}\right] \quad k=0,1,\ldots,M-1$$

The term in brackets is zero for all values of k except for k=p, when it has value one. Hence,

$$F_M(k) = F_N(k) \qquad k = 0,1,\ldots,M-1 \tag{1.21}$$

Comparing this equation with eqn. 1.14, we see that the terms that represent the aliasing do not appear in the above expression. As a result, inverting a wrapped spectrum does not yield a sequence of values taken from the N length sequence. To properly decimate, the aliased terms must be filtered out before the spectrum is wrapped and inverted. To finish the special case for decimation by two, simply use the relation N=2M in the above expressions.

We now examine the case of decimation by any ratio $\beta=N/M>1$. The arguments are nearly the same as for the case of integer $\beta$, except that it is necessary to interpolate in the frequency domain before decimating in the sequence domain.

Again, we begin with the sequence $f_N(n)$ and determine $f_M(n)$, where

$$f_M(n) = f_N(\beta n) \qquad n=0,1,\ldots,M-1 \tag{1.22}$$

Now, however, recognize that for noninteger $\beta$ there may be very few values of $\beta n$ that are integers, and so eqn. 1.22 does not completely describe $f_m(n)$. In fact, it is apparent that $f_m(n)$ will have to be a decimated version of some larger sequence than $f_N(n)$.

Let us now treat $f_m(n)$ and $f_N(n)$ as though they are aperiodic. We can express their spectral densities as

$$F_M(\tilde{\omega}) = \sum_{n=0}^{M-1} f_M(n)e^{-j\tilde{\omega}n} \tag{1.23}$$

$$F_N(\tilde{\omega}) = \sum_{n=0}^{N-1} f_N(n)e^{-j\tilde{\omega}n}$$

Though the sequences are aperiodic, their spectral densities are periodic with period $\tilde{\omega}=2\pi$, as always. In terms of absolute frequency, however, they have different periods. This is because we are treating $f_M(n)$ and $f_N(n)$ as if they are samples of the same original signal f(t), but obtained at different sampling rates $S_m$ and $S_N$. Hence, in terms of absolute frequency, they are periodic with periods $\omega_s$,($\omega_s$= sampling frequency)

$$\frac{2\pi}{S_M} = \frac{2\pi}{T} M \quad \text{and} \quad \frac{2\pi}{S_N} = \frac{2\pi}{T} N$$

respectively. Here, T is the duration of the original signal f(t). This is the same phenomenon described by figure 1.5, with the possible exception that the aliasing may not be as pronounced.

From equations 1.22 and 1.23, suppose we write,

$$F_M(\tilde{\omega}) = \sum_{n=0}^{M-1} f_N(\beta n)e^{-j\tilde{\omega}n}$$

$$= \sum_{n=-\infty}^{\infty} f_N(\beta n)e^{-j\tilde{\omega}n}$$

At this point, $f_n(\beta n)$ might not have an actual value (because $\beta n$

might not be an integer), but we can still understand the equation in a mathematical sense. Now, substitute for $f_N$ the expression in brackets below (its FT),

$$F_M(\tilde{\omega}) = \Sigma_{n=-\infty}^{\infty} [\frac{1}{2\pi} \int_{-\pi}^{\pi} F_N(\alpha) e^{j\alpha\beta n} d\alpha] e^{-j\tilde{\omega}n}$$

$$= \frac{1}{2\pi} \int_{-\pi}^{\pi} F_N(\alpha) [\Sigma_{n=-\infty}^{\infty} e^{j(\alpha\beta-\tilde{\omega})n}] d\alpha$$

Because the expression in these new brackets is the FT of a constant sequence (equal to one; the FT is evaluated at a frequency $\alpha\beta-\tilde{\omega}$), it is a weighted delta function.

$$2\pi\ \delta(\alpha\beta-\tilde{\omega}) = \Sigma_{n=-\infty}^{\infty} e^{j(\alpha\beta-\tilde{\omega})n} \quad .$$

As a result, the integral over $\alpha$ in the above reduces to an evaluation of the integrand at $\alpha=\tilde{\omega}/\beta$. And so,

$$F_M(\tilde{\omega}) = F_N(\tilde{\omega}/\beta) \qquad \tilde{\omega}\varepsilon[-\pi,\pi]$$

This means that if we are given a sequence of length N, we can find M samples of $F_N(\tilde{\omega})$ in the range $\tilde{\omega}\varepsilon[-\pi/\beta, \pi/\beta]$, and these samples will be the spectral coefficients of $F_M(\tilde{\omega}_0 k)$ for $\tilde{\omega}_0=2\pi/M$ and $k=0,.,...,M-1$.

The notation we have developed for decimation is shown in Figure 1.12.F. The procedure for decimation by any ratio $\beta=N/M>1$ follows, and the dual of this procedure can be used for decimation in the frequency domain.

Given a sequence of length N:

1. First zero extend the sequence to the nearest integer multiple of M, say $\gamma M$, such that $\gamma M \geq N$. See Fig. 1.12a.
2. Perform a DFT to find the $\gamma M$ DFS coefficients.
3. Filter the resulting spectrum at the radian angle of $\pi/\gamma$ (that is, retain only the first $M/2+1$ coefficients. See Fig. 1.12 b,c).
4. Construct the spectrum $F_M(k)$. (See Figure 1.12d. Here the number of coefficients is reduced from $\gamma M$ to only M. See note below.)
5. Invert $F_M(k)$ to obtain the decimated sequence $f_M(n)$. (Fig. 1.12e).

Note that if these operations are not implemented in software, steps 4 and 5 become:

4. Invert the filtered spectrum (still of length $\gamma M$) to obtain a sequence of length $\gamma M$, denote it $f'(n)$).
5. Obtain $f_M(n)$ from this last sequence by letting $f_M(n)$ be equal to

   $f_M(n) = f'(\gamma n)$ for $n=0$ to $M-1$.

## Section 1.2.4 Comparisons and Averages

In this section we examine a number of important concepts needed for the application of Fourier Descriptors to our work. We use the signal theory developed in the previous sections to develop methods for comparing any two curves and for averaging any number of curves together.

We begin by considering Fig. 1.13. Here, we have two sequences $f_1(n)$ and $f_2(n)$ of lengths $N_1$ and $N_2$. At this point, no assumptions are made about how these sequences are obtained, and we have purposely drawn them so that they appear to have been sampled with different sampling periods, $S_1$ and $S_2$, and to have different absolute duration, $T_1$ and $T_2$. However, we assume that all sequences have been obtained by sampling at a rate greater than or equal to the Nyquist rate. In Figure 1.13 we denote W as the bandwidth of a sequence in terms of absolute frequency.

As discussed previously, there are two useful ways to think about these sequences. The first is to treat them as having been sampled at the same sampling rate, $2\pi/S_1=2\pi/S_2$, but that they are different in duration, i.e. $T_2>T_1$ and hence $N_2>N_1$. This interpretation is useful to us, for example, if we are looking at a photograph (or a digitized version of it) and we see a number of patterns. We would probably like to distinguish between objects of different shape and size. Furthermore, even if two objects, buildings for example, are of the same general shape, we would probably like to distinguish between buildings of different sizes.

The second way to treat the sequences is as if they were sampled at different rates, i.e. $2\pi/S_1>2\pi/S_2$, and that the original signals were of the same duration, i.e. $T_2=T_1$ and hence $N_2>N_1$. This interpretation is useful, for example, if we have two photographs of the same object, but the second camera was closer to the object than the first. Hence, the same object appears in both photos, but in the second it appears larger (and perhaps rotated and translated).

Using the previous ideas as motivation, we now assume that the two sequences $f_1(n)$ and $f_2(n)$ describe the same shape. That is, $f_2(n)$ is in some sense an interpolated version of $f_1(n)$, but as yet, we make no assumptions about their generation. With this in mind, we examine the two ways to think about these sequences.

Case 1: $T_2>T_1$, $N_2>N_1$, $S_2=S_1$

If we assume that the samples $f_1(n)$ were taken at the Nyquist rate, so that $2\pi/S_1=2W_1$, then it is clear that $W_2$ must be less than $W_1$. This is because the sequences have the same shape but are of different lengths. Figures 1.14 a,b show the DFT spectra of $f_1(n)$ and $f_2(n)$.

Now suppose that we want to compare $M>N_2>N_1$ of the DFT coefficients of the two spectra on a one for one basis. Clearly, to compare coefficients that correspond to the same absolute frequency we will have to interpolate in the frequency domain. We do this by zero extending each of the sequences to length M and then performing M point DFTs on them. Figures 1.14c and d show the resulting spectra. With these spectra, we compare coefficient with coefficient and determine the similarity of the two spectra (and hence sequences). For the specific case illustrated in the figures, we conclude that the two sequences are not very similar (they have very different duration and Nyquist frequency as seen in Figures 14c and d).

Perhaps the question might arise about the value of computing the DFT spectra to make such a comparison. Though this will be explained more fully in another section [Fourier Descriptors], it is worthwhile to mention three important qualities of this method. First, the DFT coefficients are veritably independent of the picture orientation, translation, and rotation from where the sequences are obtained. Secondly, most of the characteristic shape information of a pattern (sequence) is contained in the low frequency coefficients; and thirdly the value of M can be chosen so that the DFT is computed very efficiently (eg $M=2^n$). In this way, the calculation and comparison of coefficients i slikely to be more efficient than many complicated geometrical techniques (eg, spline fitting, etc).

Another important question that arises concerns averaging. If we are given a number of sequences and are told to find an average sequence, we could simply average all the zero extended sequences together and compute an average DFT spectrum. Furthermore, we could calculate the average duration of the sequences and only retain the appropriate number of elements of the average zero extended sequence. This would yield the average sequence and it would be of an average length. We refer to this as a "direct average".

As a general pattern recognition technique, these results may not seem particularly useful. One might doubt the utility of a pattern recognition technique that compares all patterns as if they were assigned an absolute scale. There are cases where this is exactly what is needed, however, as in the case of range estimation problems. Furthermore, because we can choose M at our convenience, this method is simple and efficient.

For the more general case of recognizing patterns of the same shape, regardless of absolute size, position, and orientation, we turn to case 2.

Case 2: $T \triangleq T_1 = T_2$, $N_2 > N_1$, $S_2 < S_1$

Again, we refer to Figures 1.13 a,b. We assume that the original signals from which $f_1(n)$ and $f_2(n)$ are obtained are actually the same signal. But since $f_2(n)$ is obtained by a sampler with sampling frequency $2\pi/S_2 > 2\pi/S_1$, $f_2(n)$ is a longer sequence than $f_1(n)$. We assume that $2\pi/S_1$ is the Nyquist Frequency so that there is no aliasing in either of the spectra. Also, we note from Figures 1.15 a,b that the two spectra will repeat at different values of absolute frequency (i.e. the sampling frequency), but that the spectral coefficients of the two spectra occur at the same absolute frequency. The coefficients of both spectra are separated by an absolute frequency of $\omega = 2\pi/T$. Furthermore, the coefficients match one for one (except for a scaling constant) in the range $\omega \epsilon [-W,W]$, since the bandwidth of each sequence is the same as that of the original signal. If we wish to compare M coefficients from each spectrum, $M > N_2 > N_1$, we simply zero extend the DFT coefficients as shown in Fig. 1.15c. In this case, the two zero extended spectra are exactly alike (up to a constant) and comparing $\frac{M}{2}+1$ (or $\frac{M-1}{2}+1$ for M odd) coefficients between the two spectra is enough to determine their similarity.

For the specific case of comparing two sequences (and no more), we may remark that it is foolish to zero extend the spectra and, only then, compare them. We really need compare only the minimum of $[N_1/2+1, N_2/2+1]$ coefficients, where $N_1$ and $N_2$ are assumed even. In the general case where we compare many spectra, and where we are not sure that the sequences are properly sampled, we zero extend all the spectra to a length M, where M is greater than or equal to the largest length of all the sequences. In this way, we compare sequences of different lengths and bandwidths and can judge their similiarity.

Furthermore, we may want to average a number of sequences together. In this case, recall that zero extending the spectra is equivalent to interpolating in the sequence domain. Hence, if we average M coefficients of the zero extended spectra, we can obtain an average spectrum of length M. By inverting this spectrum, we obtain a sequence that is equivalent to a term by term average of all the interpolated sequences. This is equivalent to normalizing the lengths of the original sequences, and so we term this average as the "normalized average". Figure 1.16 shows two original sequences drawn as if they are of different duration. Also shown are the averages of cases one and two, i.e. direct and normalized averages.

## Section 2 Discretized Open Curves and Their Analysis

In this section we discuss discrete curves and methods by which they are described.

### Section 2.1: Introduction

In the analysis of vortical flows, a number of flow visualization techniques are available [ref. 25]. In developing our techniques we have assumed that hydrogen bubble illumination is the visualization method of predominant interest. With very minor alterations, however, our techniques are directly applicable to other methods as well. For example, the only change necessary to examine flow patterns from dye injection techniques is to search for regions of darkness, relative to the background, instead of the bright regions characteristic of hydrogen bubbles.

For pattern recognition problems associated with digital images, an acquaintance with image processing techniques is essential. Though a discussion of them is beyond the scope of this paper, a description of basic image processing techniques can be found in references [7,8, 9,10]. Rosenfeld and Kak [7] was particularly useful.

As shown in figure 2.1, the unprocessed patterns that we deal with are of two general types. The first type is shown connecting points a and b with a' and b', respectively. Moreover, these "timelines" are made up of two parts: the regions from $a_0,b_0$ to $a_1,b_1$, where the x component of fluid velocity varies with y position; and the remaining portions of each timeline lying in the "uniform flow field". Here, we note that the uniform flow field may be used as a reference in describing the timelines, but the actual "interesting" part of the patterns lies in the non-uniform velocity region. All the information about the flow is contained in the curve segments $(a_0,a_1)$ and$(b_0,b_1)$.

The second type of flow pattern is given by the curve segments $(c_0,c_1)$, $(d_0,d_1)$, $(c_0', c_1')$ and $(d_0', d_1')$. In each case, the entire portion of the curve is in the non-uniform region of the flow, and the entire curve carries information. For these curves, we choose the convention that the start point of each pattern will be the endpoint of each segment that is closest to the uniform flow region. For the first type of flow, we choose the lower endpoint as the start point. These conventions are necessary if we want to avoid the complicated computations that are required in the general case, where no conventions are followed. These conventions will not restrict the analysis because we are dealing with open curves. In the majority of cases where Fourier Descriptors are applied, however, the curves are closed. For closed curves, conventions of the sort that we use might not be possible to construct, and furthermore such conventions might not be desirable. As a result, our use of Fourier Descriptors is a specialization of a general and powerful technique.

## Section 2.2 Discrete Representations of Curves

Consider a binary image that is a discretized version of a picture. The term discrete is used to signify that the discrete image is only defined at specific points in the spatial domain. A binary image implies that it is two colored (eg black and white). This is a special case of a digital image, where the colors (or grey levels) are finite in number. Whenever images are processed by computer, they are both digital and discrete, and this is the case we consider. Furthermore, we will loosely use the term "digital" to mean both "discrete and digital".

Figure 2.2a shows a continuous curve, and 2.2b shows its digital version, where the digitization is binary (i.e. one bit digitization). If the curve in Fig. 2.2b is traced from point $a_0$ to $a_1$, this path can be recreated by remembering the direction of each step taken and the total number of steps. From any one picture element (pixel), we can proceed to one of its eight-neighbors (see [1] first edition chapter 9) by moving in one of eight possible directions (Figure 2.2c). Because our programs are written in FORTRAN where subscripts start at 1, we choose the numbering scheme as shown. Otherwise it is more convenient to number the directions zero through seven. Since there are twenty one dark pixels in fig. 2.2b, it will take twenty steps to reach $a_1$ from $a_0$. By keeping track of the directions, we generate the twenty element "chain code" of Fig. 2.2d. In traversing this discretized path, we note that it is much more jagged than might be expected. By using (what we have termed) Freeman's corner cutting matrix [11, Table III; 13, Table 1], the excessive jaggedness can be removed from this path. This is done by examining every adjacent pair of chain code elements and replacing them with the appropriate directions specified by the matrix. If the matrix element is zero (blank), we leave the pair of elements unchanged. By repeated application of this algorithm, we can remove the jaggedness without altering either the start or end positions of the curve. Freeman shows how this matrix may also be used to find the length of any given chain encoded curve. For Freeman's definition of length, a more detailed description of the matrix, and many other properties of discretized curves, see references [11, 12, 13].

## Section 2.3 Curve Tracking and Linear Predictive Coding

In this subsection, we introduce the specific methods that we have applied to extract curve information from digitized images.

### Section 2.3.1 Introduction

For the case of a binary image, as in Figure 2.2b, and where the image is uncluttered (i.e. all curves are non-intersecting or non-near intersecting) the tracking of the curves is trivial. By tracking, we mean the generation of a chain code that describes the progression of a path along a curve pattern on a digitized image. For the case of multilevel digital images (i.e. n>1 bit digitization), tracking is no longer trivial, and the complexity grows with n. For our case, we have eight bits (256 levels) of intensity information per pixel. In addition, the images have noise added to them from the digitization process, and the original image scene is unevenly illuminated. In spite of the difficulties, we have developed a simple curve tracking algorithm that successfully tracks most curves. In some cases, however, even if the curves seem easily intelligible by eye, the simple tracking algorithm fails to successfully follow the path. Most of these cases are caused by uneven illumination of the original scene or near intersecting objects, but whenever the simple algorithm fails, we employ a tracking algorithm based on linear predictive coding (LPC). We have combined the two algorithms so that when the simple algorithm fails, the LPC algorithm automatically is envoked.

It is interesting to note that even though we refer to the faster of the two algorithms as "simple", the code (FORTRAN) is quite complicated. This is probably a consequence of the ad-hoc nature of the algorithm, and in fact, the code is a number of times longer than the LPC algorithm. Nevertheless, the simple algorithm executes more quickly. This is consistent with the notion that short, structured code may be easy to read (and write), but as a result of the often complicated operations placed inside nested loops, such algorithms may be slow in implementation.

Since curve tracking in a multilevel, noisy environment presents many subtle problems, we present some of the fundamental ideas behind our general curve tracking algorithm (i.e. the "simple" one).

## Section 2.3.2 The General Curve Tracking Algorithm

Figures 2.3 through 2.7 describe and define the parameters we use. The figures offer specific values for some of the parameters (e.g. the number of shells used), and the only justification for their use is that we have had success with them in our applications. In fact, when a curve is successfully tracked, we have observed that the path chosen by the algorithm is nearly always the path that we would have chosen by a visual examination of the image. Furthermore, we can make some general comments about this algorithm's performance.

1. If the curve has a path that is relatively clear to a human observer, the algorithm is nearly always successful.

2. If the curve is relatively clear, except for a few small breaks, the algorithm will many times correctly negotiate through the breaks and result in a successful tracking.

3. Even when the path is not clear to humans, the algorithm will many times be successful. (This is a result of the eight bit grey level digitization, and the limited range of the video display that we used.)

4. Whenever the algorithm is not successful, examination of the image makes it clear why the algorithm failed. We have found that the primary causes of failure are uneven illumination, near intersecting curves, and faulty digitization.

5. There are a number of simple ways to improve the algorithm's performance. Time limitations have prevented their implementation.

We can ascribe the success of the algorithm to its short-observation, foresightedness and hindsightedness. Most of the curves we track have such short term predictability. On the other hand, its shortsightedness does not allow it to overcome some relatively minor obstacles. Below, we briefly summarize some of the properties of the algorithm.

We designate the "pixel of interest" (POI) as the current position of the tracking algorithm. In searching for the next pixel in the path, the algorithm only searches in a "wedge" of pixels that is defined by the "direction of arrival" (DOA) at the POI. As shown in Figure 2.3a, the DOA is actually a calculated average of the previous M values of the DOA. This average, however, is carried out in a "modified modulo 8" sense. This is because FORTRAN does not allow zero subscripts, and as a result, we define the possible directions of movement from the POI as in Fig. 2.3b. Furthermore, the average must be carried out such that averaging a "direction 8" with a "direction 2" yields a "direction 1". This technique is not shown. The wedge of interest (WOI) depends upon the DOA at the POI. Figures 2.4a and 2.5a show the two possible geometries of the wedge when the DOA is even or odd, respectively. We have used a wedge that defines a region of ninety degrees about the POA. Wedges of other shapes are certainly possible, and one may question the asymmetry of our choice. For our case, however, angular observation was deemed more important than depth.

Each wedge is divided into three shells. The pixels of each shell all have the same eight-neighbor distance from the POI, and that distance corresponds to the shell number. Furthermore, the pixels of each shell are numbered in a counterclockwise sense from the most clockwise edge of the wedge. If a wedge has a side of length N pixels, then it contains $N^2$-1 pixels other than the POI. Furthermore, it has N-1 shells, and each shell has $2\ell+1$ pixels, where $\ell$ is the shell number. We define each shell to have a weight

$$\mathrm{Wt}(k) = \frac{2\ell+1}{N^2-1} \qquad K=1,\ldots,N-1$$

where K is the shell number for the weight, and $\ell$=N-K is a shell number K shells away from the outer edge of the wedge. We choose this form of the shell weight because it is inversely related to both the distance of the shell and the number of points in the shell. Also, the sum of the shell weights is one. Other weighting functions could be used as well (if so, we recommend less weight to the low order shells).

In addition to shell weights, we define the direction weight of each pixel in a shell. Because the 45 degree marks are given as direction weights 1, 2, and 3, the other direction weights are defined by their percentage angular deviation from the 45 degree marks. These are shown in Figures 2.4,5. The direction weight matrix (for a given DOA) is shown in Figure 2.6.

Finally, the maximum intensity direction (MID) is found by the weighted average shown in figure 2.7. In each shell j, the pixel with the maximum intensity is found, and these values are stored in the array INT(j). Once the MID is calculated, NEWD is calculated. NEWD is the direction by which the path leaves the POI, and it is related to the DOA and MID by the simple relation

$$\begin{aligned} \text{NEWD} &= \text{DOA} + \text{MID} - 2 \\ &= \text{DOA} + \text{DDOA} \qquad \text{(modified modulo 8)} \end{aligned}$$

Here, DDOA=MID-2 is the deviation of NEWD from the DOA, so that the new direction is equal to the old direction plus a deviation. With the wedge geometries that we've defined, the DDOA takes on one of only three possible values, (-1,0,1).

### Section 2.3.3 Linear Prediction and Curve Tracking

We have seen that the general curve tracking algorithm is quite useful for most cases but that there are times when it fails. The failures are attributable to information that is a part of the digitized image, but with regard to tracking a specific path, the information is misleading. We seek to overcome such obstacles by recognizing that their informational content is inconsistent with the past evolution of the curve. The method of linear prediction is well suited for this purpose.

Linear predictive analysis has become one of the most widely used methods in digital speech processing [14, Chap. 8]. It is also used in many other fields and has even been applied to image processing [15]. Because of its predominance in the coding of speech signals, however, the technique has been given the acronym LPC meaning linear predictive coding.

The advantages of the LPC technique are in its accuracy of estimation and its speed of computation. In addition, it is flexible enough to allow various degrees of accuracy at the cost of further computation (or other tradeoffs). However, techniques exist where the cost of greater accuracy is small on the margin, eg. Durbin's recursive relations [14]. Furthermore, the system approximation can be guaranteed stable. While there are a number of ways to implement the LPC technique, we have chosen the covariance method. Our reasons for this are discussed later.

LPC methods are especially useful for the analysis of systems that can be modelled by an all pole linear system. That is, a parameter s(m) is to be approximated by a linear combination of its past values. The approximation is given by

$$\tilde{s}(m) = \Sigma_{k=1}^{p} \alpha_k \, s(m-k) \qquad (2.1)$$

where the $\alpha_k$ are the LPC coefficients. The error signal is given by

$$\begin{aligned} e(m) &= s(m) - \tilde{s}(m) \\ &= s(m) - \sum_{k=1}^{p} \alpha_k \, s(m-k) \qquad (2.2) \end{aligned}$$

and the "prediction error filter" is given by

$$A(z) = \frac{E(z)}{S(z)} = 1 - \sum_{k=1}^{p} \alpha_k \, z^{-k} \qquad (2.3)$$

When s(m) is the desired output of the system, for example, when a glottal excitation is the input and a segment of speech is the output, then the prediction error filter is an "inverse filter" for the system [Fig. 2.8]. If the excitation is given by u(m), then the transfer function of the system (with unity gain) is given by

$$H(z) = \frac{S(z)}{U(z)} = \frac{1}{A(z)} \qquad (2.4)$$

This is an all pole model of the system and is equivalent to a p tap filter, as shown in Figure 2.8. In some cases, the transfer function of a system has both poles and zeroes. By allowing p to be made large, however, it can be shown that any system characteristics can be well approximated by an all pole model.

Though there are a number of formulations of the linear predictive technique, only the autocorrelation method and the covariance method are discussed here. In [14], Rabiner and Schafer mention that all other formulations are equivalent to one of these two, or to a third method called the lattice method. Because the lattice method yields LPC coefficients that exactly equal those of the autocorrelation method, we will not discuss it.

The main advantage of the autocorrelation method is that it is guaranteed to yield a stable estimate of the system. In addition, this formulation yields a set of equations that have terms equal to autocorrelations of the original sequence (at a given shift). Because the autocorrelation of a sequence can be Fourier transformed to yield the power spectrum, this method is useful for spectral estimation (note, however, that spectral estimation can also be done by the covariance method. (ref. 14, p436)). Finally, the matrix that describes the linear equations that must be solved is both Toeplitz and symmetric. This allows an efficient solution of the set of equations, i.e. Durbin's recursion.

For our purposes, however, the autocorrelation method will not yield satisfactory results. The main disadvantage of this method is the necessity to window and smooth the input sequence, eg. by a Hamming window. This means that at the beginning of each analysis frame, i.e. the nonzero portion of the windowed sequence, a nonzero signal is to be predicted from the previous zero valued portions of the signal. This results in large errors at the beginning of each analysis frame. Likewise, at the end of each frame, when the windowed sequence approaches zero, a near zero valued signal is to be predicted by nonzero signal values. Again, large errors result. The saving grace for the autocorrelation method is that for large window frames, eg. on the order of one hundred samples or more, the mean squared error approaches that of the covariance method. These results are emphasized by the data of Chandra and Lin [16] and of Maragos, Schafer, and Mersereau [15]. Figures 2.9 and 10 show some of their results.

Because our data frames will be on the order of N=30 (or less), we have implemented the covariance method. The main disadvantage of the covariance method is that it cannot guarantee the stability of the system estimation. For large window frames, where the autocorrelation and covariance results converge, this is not a problem. For our case of short windows, however, we recognize this as a problem. In practice, we find that instabilities rarely occur at crucial points of the tracking procedure. We now present the fundamental notions that motivate and describe our covariance method.

As shown in Figure 2.11, we imagine that a curve is tracked up to the point m=0 and that the values of s(m) for $m \geq 0$ are to be predicted. As given by equation 2.1, we assume that s(m) is a linear combination of its past values, i.e. m<0. At this point in the tracking, we have not yet calculated any predictor coefficients. In theory, two things must first be decided: the order p of the predictor, and the analysis frame size N. In practice, we also decide how many predictions to calculate before re-evaluating the predictor coefficients. (We call this number LPUP representing the linear prediction update length.) It is clear that as LPUP gets large, the accuracy of the predictions decreases. As defined in equation 2.2, the error signal is a function of the LPC coefficient $\alpha$. We now decide to choose the values of $\alpha$ in such a way as to minimize the squared error of the predictions. Since this is done for each of the N predictions in the analysis frame, we really wish to minimize the mean squared error, which is given by

$$E_m \triangleq \overline{e^2(m)} \triangleq \frac{1}{N} \sum_{m=-N+1}^{0} [s(m) - \sum_{k=1}^{p} \alpha_K s(m-k)]^2 \qquad (2.5)$$

There are p different values of $\alpha$ to choose, so p equations must be solved. Each is of the form,

$$\frac{dEm}{d\alpha j} = \sum_{m=-N+1}^{0} \frac{1}{N} \frac{d}{d\alpha j} [s(m) - \sum_{k=1}^{p} \alpha_k s(m-k)]^2$$

$$= 0 \qquad \text{for } j = 1,2,\ldots,p \qquad (2.6)$$

Defining the term in the brackets as u, then

$$\frac{dEm}{d\alpha j} = \sum_{m=-N+1}^{0} \frac{2}{N} u \frac{du}{d\alpha j} = 0 \qquad j=1,p$$

We can drop the 2/N and proceed to write

$$\frac{dEm}{d\alpha j} = \sum_{m=-N+1}^{0} [s(m)-\sum_{k=1}^{p} \alpha_k \; s(m-k)][-s(m-j)] = 0$$
$$j=1,p$$

This implies that

$$\sum_{m=-N+1}^{0} \sum_{k=1}^{p} \alpha_k \; s(m-k)s(m-j) = \sum_{m=-N+1}^{0} s(m)s(m-j) \qquad j=1,p \qquad (2.7)$$

We purposely leave the limits of the first sum in this form to emphasize that we are calculating the LPC coefficients with values of m<0. We now define $\phi(i,j)$ as

$$\phi(i,j) \overset{\Delta}{=} \sum_{m=-N+1}^{0} s(m-i)s(m-j) \qquad (2.8)$$

If we interchange the order of summation on the left side of equation 2.7, we recognize that

$$\sum_{k=1}^{p} \alpha_k \sum_{m=-N+1}^{0} s(m-k)s(m-j) = \sum_{m=-N+1}^{0} s(m)s(m-j) \qquad (2.9)$$

$$= \sum_{k=1}^{p} \alpha_k \; \phi(k,j) = \phi(0,j) \qquad j=1,p \qquad (2.10)$$

For any given j, eqn. 2.10 represents the j-th row of the set of equations shown below.

$$\begin{bmatrix} \phi(1,1) & \phi(1,2) \ldots \phi(1,k) \ldots \phi(1,P) \\ \phi(2,1) & \phi(2,2) \ldots \phi(2,k) \ldots \phi(2,P) \\ \vdots & \vdots \qquad \vdots \qquad \vdots \\ \vdots & \vdots \qquad \vdots \qquad \vdots \\ \phi(P,1) & \phi(P,2) \ldots \phi(P,k) \ldots \phi(P,P) \end{bmatrix} \begin{bmatrix} \alpha_1 \\ \alpha_2 \\ \vdots \\ \vdots \\ \alpha_p \end{bmatrix} = \begin{bmatrix} \phi(0,1) \\ \phi(0,2) \\ \vdots \\ \vdots \\ \phi(0,P) \end{bmatrix} \qquad (2.11)$$

In matrix form, this is written

$$\overset{\rightarrow}{\vec{\Phi}}\,\vec{\alpha} = \vec{\Phi}_0 \qquad (2.12)$$

We recognize that since $\phi(j,k)=\phi(k,j)$ then $\overset{\rightarrow}{\vec{\Phi}}$ is a symmetric matrix. Also, $\phi(o,j)=\phi(j,0)$. Furthermore, the diagonal elements are related by

$$\phi(j+1,k+1) = \phi(j,k) + [s(-N-i)s(-N-j)] - [s(-i)s(-j)] \qquad (2.13)$$

and hence $\overset{\rightarrow}{\vec{\Phi}}$ is analogous to a covariance matrix.

Figure 2.12 presents a schematic layout of our LPC curve tracking algorithm. As a note, the present version of the algorithm performs the same operations in a slightly modified order.

## Section 2.4 Distortion of Discrete Curves

Given the chain encoded sequence of a discrete pattern (eg. obtained from a tracking algorithm), we discuss some properties and methods of their distortion. We also present a distortion algorithm that preserves the start and end points of the pattern and uniformly distorts the pattern.

Recall that a chain code element (i.e. a number from one to eight) represents the incremental direction of traversal from the pixel of interest (POI) to the next pixel in the pattern. From Fig. 2.2c, the relations of Fig. 2.13 are derived, and four pairs of pixel opposites exist. Clearly, if any one element is added to the chain code, the chain code is lengthened and the endpoint (or start-point) of the resulting pattern is altered. For the endpoint to be brought back to its original position, a number of chain code elements must be added to the sequence with the constraint that the total $\Delta X$ and the total $\Delta Y$ of all the inserted elements is zero. The smallest number of pixels that can be added to a sequence without altering either the start or end points is two: a chain code element and its opposite.

Distortions can be made in any one of the four principal directions defined by the four pairs of opposites. That is, directions 1 and 5 define an axis through the POI at an angle of zero radians. Therefore, inserting an equal number of 1's and 5's into the chain code will not alter the endpoints of the pattern and will distort it in angular direction zero. Likewise, distortions can be made in angular directions $\pi/4$, $\pi/2$, and $3\pi/4$ radians. Figure 2.14 shows a chain code, and its pattern, and a distortion of this chain code in direction $\pi/4$, and its pattern. By randomly inserting the "distortion elements" into the sequence, however, two undesirable results appear. First, the distorted pattern loses its qualitative similarity with the original, and second, protruding branches in the pattern may appear. These branches correspond to a retracing of the path that occurs when two opposite elements are sequentially placed in the chain code. Both of these results are due to the randomness of the insertions.

Hence, we condition the placement of distortion elements into the chain code by using (what we have termed as) the tetrahedron test. This test conditions the movement from the POI such that it differs in angular direction from the direction of arrival (DOA) by either $-\pi/4$, 0, or $\pi/4$ radians. This condition is consistent with the restrictions used in the general curve tracking algorithm (sec 2.3.2). Figures 2.15 a and b illustrate these ideas.

By distorting in this way, the components of the pattern are exaggerated or stretched in the direction of distortion. The evolution of the pattern in other directions, however, is unaltered. Because we will examine the spatial frequency characteristics of these distorted patterns (sec. 3.2), because the distortion only alters certain spatial frequencies, and because there is no justification for direction selective distortion, we have developed a distortion algorithm that is biased in direction only as much as the undistorted pattern is biased in direction. We discuss the algorithm below.

Previously, we showed that a distortion in any one angular direction can be accomplished by inserting a pair (or many pairs) of elements into the chain code. This distortion might be termed "linear" or "directional". If distortions are performed simultaneously in each of the four directions, then there is no directionality. Furthermore, because a minimum of four pairs (or eight elements) must be inserted into the sequence, and because these elements can be arranged to approximate a circle (actually an octogon), the distortion might be termed "circular" or "omni-directional". Obviously, the amount of distortion is dictated by the number of circles that are inserted. Figure 2.16a illustrates this.

Suppose that when inserting elements into a chain code it is agreed that no two distortion elements of the same value will be placed sequentially and that all insertions follow the tetrahedron rule. Then there is a maximum number of circular insertions that can be made, and furthermore, this number is the minimum of the number of insertions allowed for any single element.

For any single element, we use the tetrahedron rule to calculate the maximum number of possible insertions (MPI) without allowing for repetitions of elements. Then we find the minimum of the MPI (MINMPI) , which is the maximum number of possible circular insertions. To distort the pattern, a number less than MINMPI is chosen as the number of desired insertions (NUMDES). Clearly, these insertions can be made in a very large number of different ways, but for uniform distortion of the pattern, the insertions should be spaced throughout the length of the pattern as evenly as possible.

Because some patterns may yield a MINMPI equal to zero, there is an apparent problem. In this case, however, the original pattern is biased against certain directions, and so omitting distortions in these directions is not undesirable. In fact, using the minimum nonzero MPI allows the distortion to be as directional as the original pattern. This seems to be desirable in general. Below, the algorithm is presented in more detail.

## The Circular Distortion Algorithm

Given a chain code sequence:

1) Calculate the number of occurrences (NOC) of each element, i.e.

$$\mathrm{NOC}(j),\ j=1,8$$

2) Find the maximum possible insertions (MPI) per element by use of the tetrahedron rule, i.e.

$$\begin{array}{lclclcl}
\mathrm{MPI}(1) & = & \mathrm{NOC}(8) & + & \mathrm{NOC}(1) & + & \mathrm{NOC}(2) \\
\mathrm{MPI}(2) & = & \mathrm{NOC}(1) & + & \mathrm{NOC}(2) & + & \mathrm{NOC}(3) \\
\vdots & & \vdots & & \vdots & & \vdots \\
\mathrm{MPI}(7) & = & \mathrm{NOC}(6) & + & \mathrm{NOC}(7) & + & \mathrm{NOC}(8) \\
\mathrm{MPI}(8) & = & \mathrm{NOC}(7) & + & \mathrm{NOC}(8) & + & \mathrm{NOC}(1)
\end{array}$$

3) Find the minimum of the nonzero MPI. Denote this as MINMPI. This is also the maximum number of insertions without repetition.

3A) If some MPI are zero, then: for every element number j where MPI(j)=0, exclude j from the insertions, and also exclude its opposite element, i.e. j+4 (modified mod 8).

4) Denote by NUMDES ($\leq$MINMPI) the number of insertions desired.

5) For each of the distortion directions, calculate the fraction of the possible insertions that are desired. Denote these by FRACT(j), j=1,8 where

$$\mathrm{FRACT}(j) = \mathrm{NUMDES}/\mathrm{MPI}(j)$$

and where MPI(j) $\neq$ 0.

6) For every element in the chain code sequence, perform the following:

   a) Set the current element of the distorted sequence equal to the original element.

   b) Use the tetrahedron test to decide which elements could possibly be inserted after this element.[exclude directions from 3A]. Increment a counter for each of these elements that keeps track of how many times it appears as a possible insertion element.

   c) Compare FRACT(j) with the counter (for element j) and decide if an insertion is made.

   d) Insert the proper elements into the distorted sequence.

Figure 2.16b shows distortion patterns made by this algorithm. Note that some improvements can still be made, most notably, in step 6b the order of the elements tested and inserted should be made random. This would diminish the occurrance of the semicircular patterns evidenced for large NUMDES. As an example of other types of distortion, Figure 2.17 shows a directional distortion where the distortion is not uniformly spread throughout the length of the pattern, but the start and end points are preserved. Figure 2.18 shows a distortion where the distortion is uniformly spread throughout the pattern length, but the start and step points are not preserved.

# Section 3 Fourier Descriptors

## Section 3.1: Introduction

In the last section, we described how to obtain chain code descriptions for the curve patterns (timelines of a flow) from curve tracking techniques. We now employ the chaincodes to quantitatively describe each curve.

From the chaincode of a curve, we readily obtain the tangent angle of the tracked curve for each incremental step of the track. Because the chaincode describes the movement from pixel to pixel in one of only eight angular directions (recall figure 2.2c), the tangent angle of the curve at that step is one of only eight values. Figure 3.1 shows the relation between chaincode elements and the tangent angle of the curve, as we have chosen it. As shown, all the chaincode elements are associated with angles in $[0,2\pi]$ plus the anbiguity of $2\pi m$ radians, where m is any integer.

Clearly, what is meaningful is not the absolute tangent angle of the curve, but instead, the change in the tangent angle from the original direction of the curve. Indeed, it can be argued that even this function retains too much information and that the only significant information is where the tangent angle changes from step to step. Figure 3.2 shows the relationships between the three measures of the tangent angle as a function of length along the curve.

When dealing with discretized curves, the continuous absolute tangent angle function $\theta(\ell)$ is discretized into the sequence $\theta(n)$.

Additionally, one compares different sequences by comparing their DFT spectra, and hence one can compare different curves of an image. Furthermore, since $\phi(n) = \theta(n) - \theta(0)$, it is clear that the DFT coefficients of $\phi(n)$ are exactly the same as those of $\theta(n)$, except for the zero-th coefficient. (The zero-th coefficient contains the average value of the sequence.) Hence, when comparing DFT coefficients, using $\phi(n)$ in place of $\theta(n)$ is perfectly acceptable.

In contrast, the DFT coefficients of $\psi(n)$ are very dissimilar from those of $\theta(n)$. Consider the following equations.

$$\begin{aligned} \psi(n) &= \phi(n) - \phi(n-1) \\ &= \theta(n) - \theta(0) - \theta(n-1) + \theta(0) \\ &= \theta(n) - \theta(n-1) \end{aligned} \tag{3.1}$$

$$\begin{aligned} \Psi(z) &= \Theta(z)[1-z^{-1}] \\ \Psi(\tilde{\omega}) &= \Theta(\tilde{\omega})[1-e^{-j\tilde{\omega}}] \\ &= \Theta(\tilde{\omega})[2e^{j(\tilde{\omega}/2+\pi/2)}]\sin(\tilde{\omega}/2) \\ |\Psi(\tilde{\omega})| &= 2|\Theta(\tilde{\omega})|\sin(\tilde{\omega}/2) \qquad \tilde{\omega}\varepsilon[0,2\pi] \end{aligned} \tag{3.2}$$

(Note that $\sin(\tilde{\omega}/2)$ is always positive for $\tilde{\omega}$ in $[0,2\pi]$.) From eqn. 3.2 we see the problem of using the function $\psi(n)$ instead of $\theta(n)$ or $\phi(n)$; i.e. the DFT coefficients $\Psi(\tilde{\omega})$ are distorted versions of $\Theta(\tilde{\omega})$. For $\tilde{\omega}$ near zero, the $\Psi(\tilde{\omega})$ are always near zero. For $\tilde{\omega}$ near $\pi$, the $\Psi(\tilde{\omega})$ are amplified versions (by a factor of 2) of $\Theta(\tilde{\omega})$. Later, we will see that the basic shape of a curve depends only on its low frequency coefficients, and hence, when comparing the $\Psi(\tilde{\omega})$ of two similar looking curves, one may wrongly conclude that they are dissimilar. That is, by using the $\Psi(\tilde{\omega})$, low frequency similarity is obscured by high frequency dissimilarity. We conclude that, though $\psi(n)$ may be useful in other situations, it is not a useful function in determining basic shape similarity.

In our discussion so far, we have intimated that by performing a DFT on the tangent angle function, any two curves may be compared. Cosgriff [17] was the first to suggest this method, and many others have since pursued it. However, most of the work done in this area has been for the case of plane closed curves. In general, however, the DFT coefficients by themselves are not good measures of the similarity of two curves. This is because they contain information about size, orientation, and phase of the curves.

Granlund [18] was one of the first to define measures of similarity that are based on the Fourier coefficients, but his measures have the ability to recognize similar shapes at arbitrary translations, rotations, and dilations. Wallace and Mitchell [19] show the need to obtain properly normalized Fourier descriptors, and they develop the use of a library from which to match and identify shapes. In a more modern and more theoretical analysis of Fourier Descriptors, Crimmons [22] shows that measures using DFT magnitudes alone are not sufficient to uniquely describe closed curves. We also site a number of other references [20,21,23,24], and we remark that a useful list of references is supplied in [21,23]. In a manner similar to [18] and [20], yet with a number of differences, we describe our methods.

## Section 3.2 Fourier Descriptors and the Comparison of Plane Open Curves

Below we briefly discuss the mathematical relations between the DFT coefficients of a tracked curve and the actual curve. Then, we discuss our similarity measure.

### Section 3.2.1 Theoretical Development

Suppose that a continuous relative tangent angle function $\phi_o(\ell)$ is passed through a sample and hold device. We then obtain the continuous function $\phi(\ell)$ and the sequence $\phi(n)$, as shown in figure 3.3. Now, note that $\phi(\ell)$ is an approximation of $\phi_o(\ell)$, and the accuracy of the approximation depends upon the sampling rate $\omega_s = 2\pi/S$ and the bandwidth W of the signal $\phi_o(\ell)$, where S is the period. Treating $\phi(\ell)$ as a continuous aperiodic function of $\ell$, its Fourier transform is written

$$\Phi(\omega) = \int_0^L \phi(\ell)\, e^{-j\omega\ell} d\ell$$

$$= \sum_{n=0}^{N-1} \phi(nS) \int_{nS}^{(n+1)S} e^{-j\omega\ell} d\ell \qquad (3.3)$$

Note that $\phi(nS)$ comes out of the integral because it is constant over the limits of integration; also, $\phi(n)$ can be substituted for $\phi(\ell)$ (at values $\ell$=ns) to obtain,

$$\Phi(\omega) = \sum_{n=0}^{N-1} \phi(n) \cdot \frac{-1}{j\omega} [e^{-j\omega\, S(n+1)} - e^{-j\omega Sn}]$$

$$= \frac{1}{j\omega} \sum_{n=0}^{N-1} \phi(n) e^{-j\omega Sn} \cdot e^{-\frac{j\omega S}{2}} [e^{\frac{j\omega S}{2}} - e^{-\frac{j\omega S}{2}}]$$

$$= \frac{2}{\omega} e^{-\frac{j\omega S}{2}} \sin(\omega S/2) \sum_{n=0}^{N-1} \phi(n) e^{-j\omega Sn} \qquad (3.4)$$

It is important to note that $\Phi(\omega)$ is an aperiodic spectral density, and it is an approximation to the actual spectral density $\Phi_0(\omega)$ of the original curve $\phi_0(\ell)$.

By taking samples of this spectral density (eqn. 3.4) at values $\omega S=\tilde{\omega}_k=2\pi k/N$, we can write eqn. 3.4 as,

$$\Phi(\tilde{\omega}_k) = \frac{2S}{\tilde{\omega}_k} e^{-j\frac{\tilde{\omega}_k}{2}} \operatorname{Sin}(\frac{\tilde{\omega}_k}{2}) \sum_{n=0}^{N-1} \phi(n) e^{-j\tilde{\omega}_k n}$$

$$= \frac{2\pi}{\omega_s} e^{-j\frac{\tilde{\omega}_k}{2}} \operatorname{Sa}(\frac{\tilde{\omega}_k}{2})[\text{DFT of } \phi(n);\ \tilde{\omega}=\tilde{\omega}_k] \qquad (3.5)$$

Here, we use the definition of the absolute sampling frequency $\omega_s=2\pi/S$, the sampling function $Sa(x) = \sin(x)/x$, and a scaled version of the DFT (eqn. 1.10).

The interpretation of this result is very important. Note that $\Phi(\omega)$, the approximation of $\Phi_0(\omega)$, can be generated through use of the DFT coefficients of the sequence $\phi(n)$. Hence, if $\phi(n)$ is of length N (even), then the first (N/2+1) DFT coefficients uniquely specify the spectral density of the sampled function $\phi(\ell)$. As a result, $\Phi_0(\omega)$ can only be estimated up to the absolute frequency $\omega=\pi/S$, i.e. half the sampling frequency. Figure 3.4 shows how $\Phi(\omega)$ is related to $\Phi(\tilde{\omega}_k)$ and the DFT of $\phi(n)$.

In defining a measure of similarity between two curves, it is important to note that the spectra $\Phi(\tilde{\omega}_k)$ gives less weight to the coefficients as $\tilde{\omega}_k$ grows than does the DFT of $\phi(n)$. For two steplike functions (as $\phi(\ell)$ is steplike) then, the high frequency components contribute less to their similarity than do the high frequency components of their samples (e.g. $\phi(n)$). This is a result of the sampling function in eqn. 3.5. These results also re-enforce the intuitive notions that high frequency components are more important in the reconstruction of sequences (impulse-like) than they are in the reconstruction of (smooth) continuous time or space bandlimited functions.

With the above results, we now discuss what Bennett and MacDonald [21] have reported. They measured the correlation between the DFT coefficients (used in the approximation of $\Phi(\tilde{\omega}_k)$ in 3.5) and the original coefficients used in their test sequences. They found that the correlation for the coefficients for $\tilde{\omega}>\pi/2$ was essentially random and of low magnitude. Their conclusion is that there is little or no dependence between the high frequency components of their control data and the empirical data. From figure 3.4, we would expect that there be correlation for $\tilde{\omega}$ greater than $\pi/2$ and even up to $\pi$. This is because, at $\tilde{\omega}=\pi/2$, the sampling function $Sa(\tilde{\omega}/2)$ still equals 0.900, and even at $\tilde{\omega}=\pi$ the sampling function equals 0.657. This indicates that there must be other factors that are heavy determinants of the correlation of two sampled curves. Specifically, these factors must have large components for values of $\tilde{\omega}$ greater than $\pi/2$. The most obvious cause of this phenomenon is the direction quantization to eight levels (3 bits). A second cause is the tracking algorithm.

## Section 3.2.2: Fourier Coefficients and Reconstruction

Figure 3.5a shows a relative tangent angle function quantized to multiples of $\pi/4$. The sequences shown in 3.5b,c represent the phase and magnitude of the DFT coefficients, and 3.5d shows the actual curve tracked from a digitized picture. This is the same curve as the original from Fig. 2.16b. Figures 3.6 and 3.7 show three of the distorted curves from Fig. 2.16b along with their magnitude and phase coefficients. Figure 3.8 shows two extremely dissimilar curves and their DFT magnitude and phase coefficients.

It is desirable to understand the significance of the Fourier coefficients to the shape of the curves. Figure 3.9 shows the original curve and its relative tangent angle function. Adjacent to these are the corresponding functions obtained from an IDFT of a truncated version of the original DFT spectrum. Here, the reconstruction of the curve was accomplished by retaining 2% of the lowest order unique DFT coefficients. For Fig. 3.9, there are 75 unique original coefficients, and the reconstruction only used one of these.. Clearly, neither the curve nor the tangent angle function resembles the originals very closely, and yet, the amount of similarity present is quite surprising. Figure 3.10 shows the reconstruction using 3.5%.(equal to two) of the original coefficients. Figure 3.11 shows the reconstruction using 5% (equals three) of the coefficients, and 3.12 shows it using 10% (equals seven). Figures 3.13 through 3.18 show reconstructions using 20%, 40%, 60%, 80%, 90% and 100% of the lowest order coefficients. These figures dramatically indicate that if the patterns of interest are similar to those in the figures, then their general shapes are determined predominantly by their low order DFT coefficients. This conclusion, however, depends entirely upon the shapes of the patterns of interest. It is entirely possible that in experiments with completely different patterns of interest, a much larger percentage of coefficients is needed to sufficiently describe the patterns. For most digital images, however, the patterns of interest are likely to be made up of a large number of pixels, and in such cases it is likely that a small percentage of DFT coefficients carry the predominant amount of shape information.

It is important to note that this result is generally independent of what Bennet and MacDonald have reported (as discussed in sec. 3.2.1). Their results indicate that even when the original patterns have significant high spatial frequency components, there is very low correlation between these and the high order DFT coefficients of the tracked curves.

In the next section, the DFT coefficients are normalized to facilitate the comparison of various spectra. It is important to note, however, that for the proper reconstruction of the tangent angle sequence, and hence the chain code, the original DFT coefficients must be used. If a normalized version is used in the reconstruction, the tangent angle "envelope" will be the same, but there will be an improper scaling of the sequence values. This results in an improper chain code representation of the curve and the inability to recreate the actual pattern. These arguments are equally applicable to the case of a "normalized" average, and care must be taken to reconstruct a tangent angle function that leads to a meaningful chain code.

### Section 3.2.3 Similarity Measures

In this section, we present a simple measure from which to quantitatively compare the similarity of any two patterns. First, however, the DFT coefficients must be normalized so that two patterns with the same spectral envelopes, but with different scalings of the envelopes (i.e., the signals have different energy: Re Parseval's Thm.), can be recognized as exactly similar.

In addition, it is necessary to discuss the significance of the zero-th coefficient. The zero-th coefficient of the DFT (eqn. 1.10) represents the average value of the relative tangent angle function. For the case of closed curves [20], generally there is no convention that uniquely defines the start and end points of the closed curve. That is, any point on the curve can be chosen as the start point. In this case, the average value of the relative tangent angle is meaningless; it varies depending on the start point. Hence, for closed curves, the zero-th DFT coefficient is not used in the similarity measure.

In the case of open curves, conventions for choosing the start and end points of the curve are possible (sec. 2.1). Additionally, the relative tangent angle is independent of the starting direction, and the zero-th DFT coefficient represents the average deviation of the tangent angle function from its starting value. This is a useful quantity in determining the similarity of two curves, and so the zero-th coefficient is included in the similarity measure of open curves. It is important to note, however, that the DFT coefficients should be calculated as in eqn. 1.10, that is, the factor of 1/N should be included in the calculation. Many library DFT subroutines (eg. IMSL) do not include this factor, and care should be taken to note this. In this case, the zero-th coefficient contains information about the absolute size of the pattern, and it should not be used in the similarity measure.

For the similarity measure, only the magnitude of the DFT coefficients need to be normalized. The phases are unaltered. For the normalization, the largest magnitude coefficient of each spectrum is assigned to the same value (eg. 1000), and the remaining coefficients are scaled appropriately. As noted earlier (sec. 3.2.2), the scale factor for each spectrum should be recorded to facilitate the averaging and/or reconstruction of the patterns.

For two properly chosen and normalized sets of coefficients, X(k) and Y(k) the similarity of the magnitude coefficients can be determined by

$$S_M(N) = \frac{\Sigma_{j=1}^{N} X(j)Y(j)}{[\Sigma_{j=1}^{N} X^2(j)\Sigma_{j=1}^{N} Y^2(j)]^{\frac{1}{2}}} \tag{3.6}$$

where X(j) and Y(j) denote the magnitude of the j-th coefficient from each spectrum. N is the number of coefficients that are compared (eg. see sec. 1.2.4). Note that eqn. 3.6 really describes the similarity in terms of auto- and cross-correlations of X and Y. The numerator is the cross-correlation with zero shift, commonly written as $R_{XY}(0)$. The denominator is the square root of the product of the individual auto-correlations $R_X(0)$ and $R_Y(0)$. For the similarity of the phase coefficients, the same expression is used except X(j) and Y(j) denote the phase (radians) of the jth coefficient. We distinguish the similarity of the magnitude coefficients from that of the phase coefficients by using the notation $S_M(N)$ and $S_\Phi(N)$, respectively.

From the well known Schwartz inequality, it is clear that the similarity measures will have absolute value less than one, $S_m$ will always be greater than or equal to zero, but $S_\Phi$ may be either positive or negative. This is because the phase of the coefficients has been defined on $[-\pi,\pi]$. By converting all the phases to values in $[0, 2\pi]$, $S_\Phi$ will also be positive or zero. We now define the overall similarity of curves X and Y by

$$S(N) = S_M(N)S_\Phi(N) \tag{3.7}$$

where N coefficients are used in the calculation.

Figure 3.19 presents a table of values for the similarity of the curves in Figure 2.16b. The similarity measures are for the distorted curves in comparison with the original curve.

A number of comments should be made about the similarity measure and the distortion patterns. The table of values in Fig. 3.19 and the qualitative similarity of the curves in Fig. 2.16b indicate that the low order coefficients are quite adequate for determining the general similarity of two patterns. For similar curves, however, it is also clear that more coefficients need to be included in the calculation if a finer measure of similarity is required.

From the results, it is also possible to remark on the distortion algorithm (sec. 2.4) used to create the curves of Fig. 2.16b. From Fig. 2.16b, the distortion algorithm apparently preserves the qualitative nature of the original pattern. The table of Figure 3.19 shows a similar result; the distortion algorithm preserves the frequency content of the original pattern, even when the distorted pattern is 50% larger (NUMDES=12) than the original.

Perhaps, for the purposes of this presentation, the distortion algorithm is too "good". It seems desirable to distort a pattern to the point where the similarity is appreciably diminished. One way to accomplish this would be to appreciably distort the original DFT coefficients and then reconstruct the distorted patterns. This is the logical alternative (or dual) of distorting in the spatial domain and then calculating the DFT coefficients.

The final thing to note about the values in Fig. 3.19 is the trend of the $S_\Phi(N)$ for a given N. For small N, the trend coincides with the intuitive notion that more distortion leads to less similarity. As N grows, however, this trend is not followed. In fact, when N reaches its maximum value (as defined in sec. 1.2.4), the $S_\Phi$ values for highly distorted curves are sometimes greater than those of the less distorted curves. This result, however, is consistent with the conclusion that, even for similar patterns, the high order coefficients are not well correlated.

In addition, because the phase coefficients for high frequencies may have values equal to (or even larger than) the low frequency coefficients, the resulting $S_\Phi(N)$ will fluctuate in value for increasing N. This is not true for the magnitude coefficients, however, where the high order coefficients have small magnitude in comparison with the low order coefficients. An interesting study could be made by comparing similarity values for (highly) similar curves, but where $S_M$ and $S_\Phi$ are calculated from coefficients selected from various ranges of the spectra.

## Conclusion

In this paper, we have dealt with a pattern recognition problem. We first developed an efficient way of characterizing a discretized pattern through curve tracking and the relative tangent angle function. We then applied our knowledge of signal and spectral analysis to obtain the Fourier description of patterns (open curves). Finally, we have described a method for calculating the similarity of the original patterns.

Fig. I.1 Digitized image of a fluid flow pattern (Hydrogen Bubble Visualization).

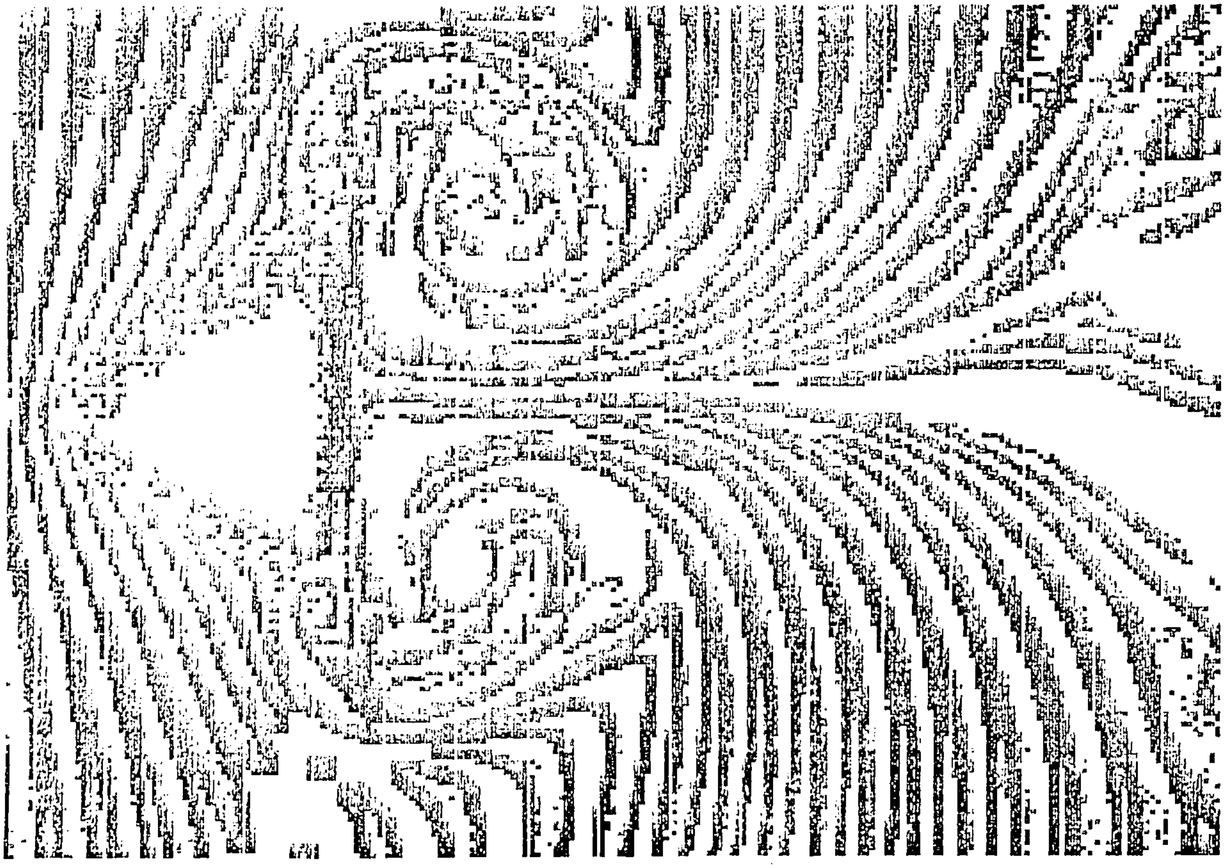
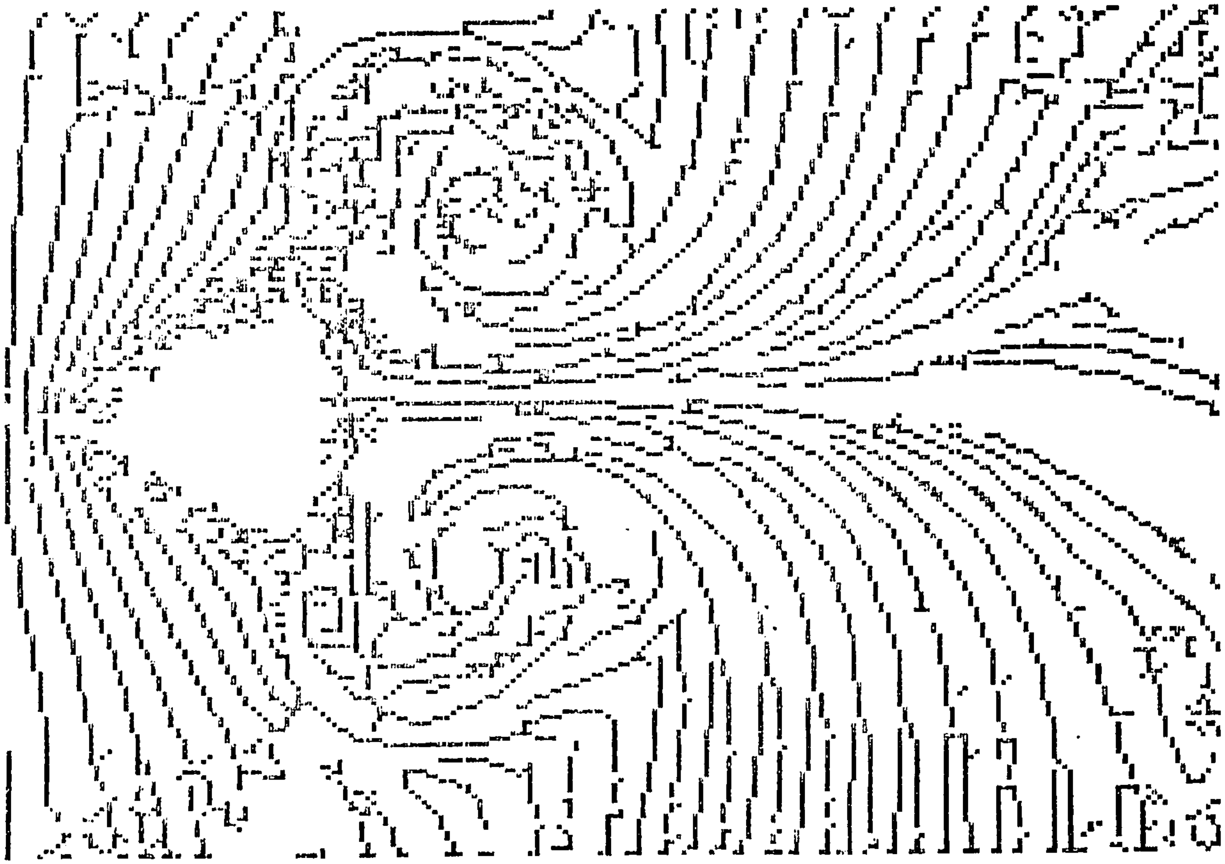


Fig. I.2 The image of Fig. I.1 processed by a region thinning algorithm.



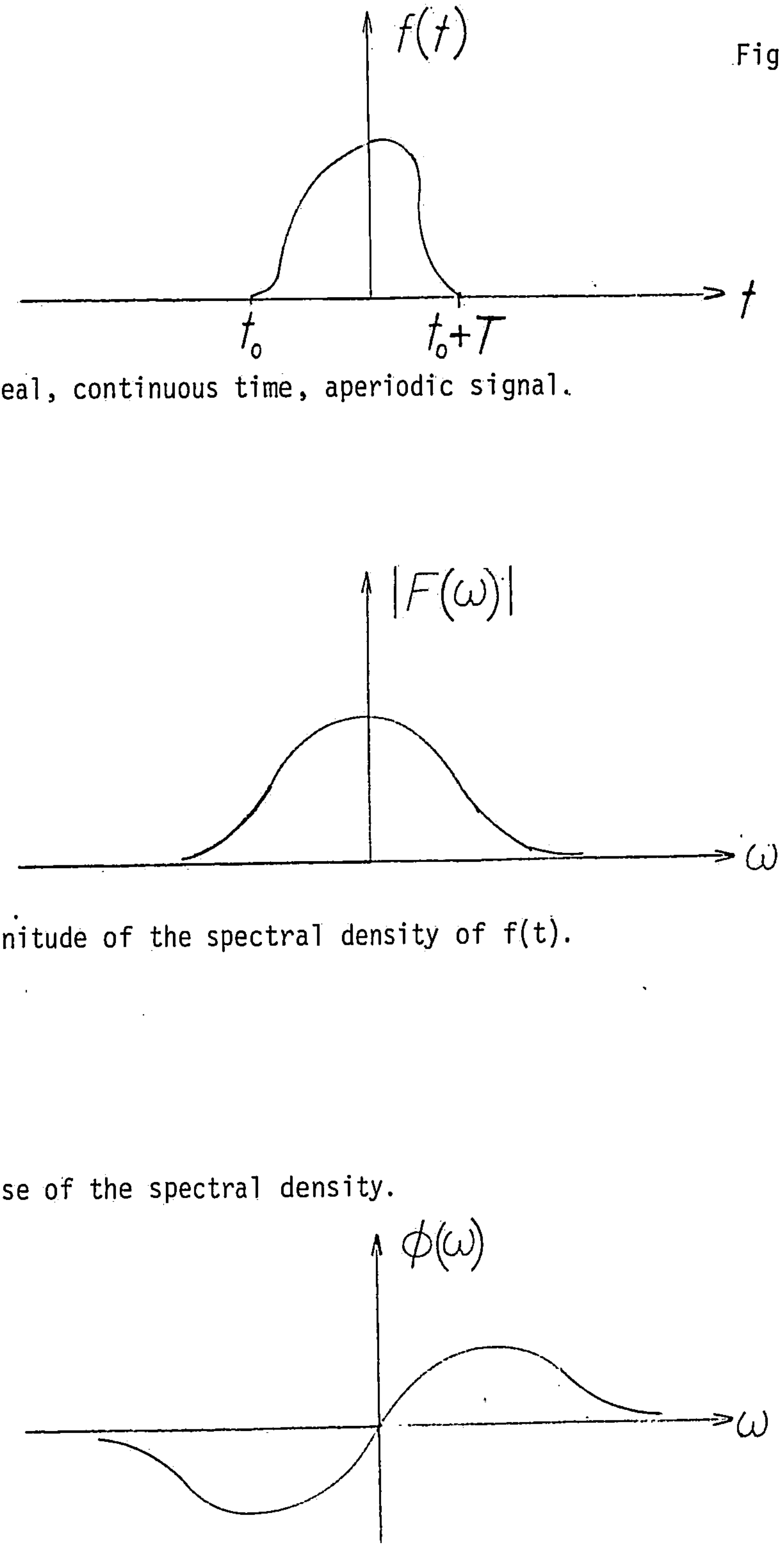


Fig. 1.1

a) A real, continuous time, aperiodic signal.

b) Magnitude of the spectral density of f(t).

c) Phase of the spectral density.

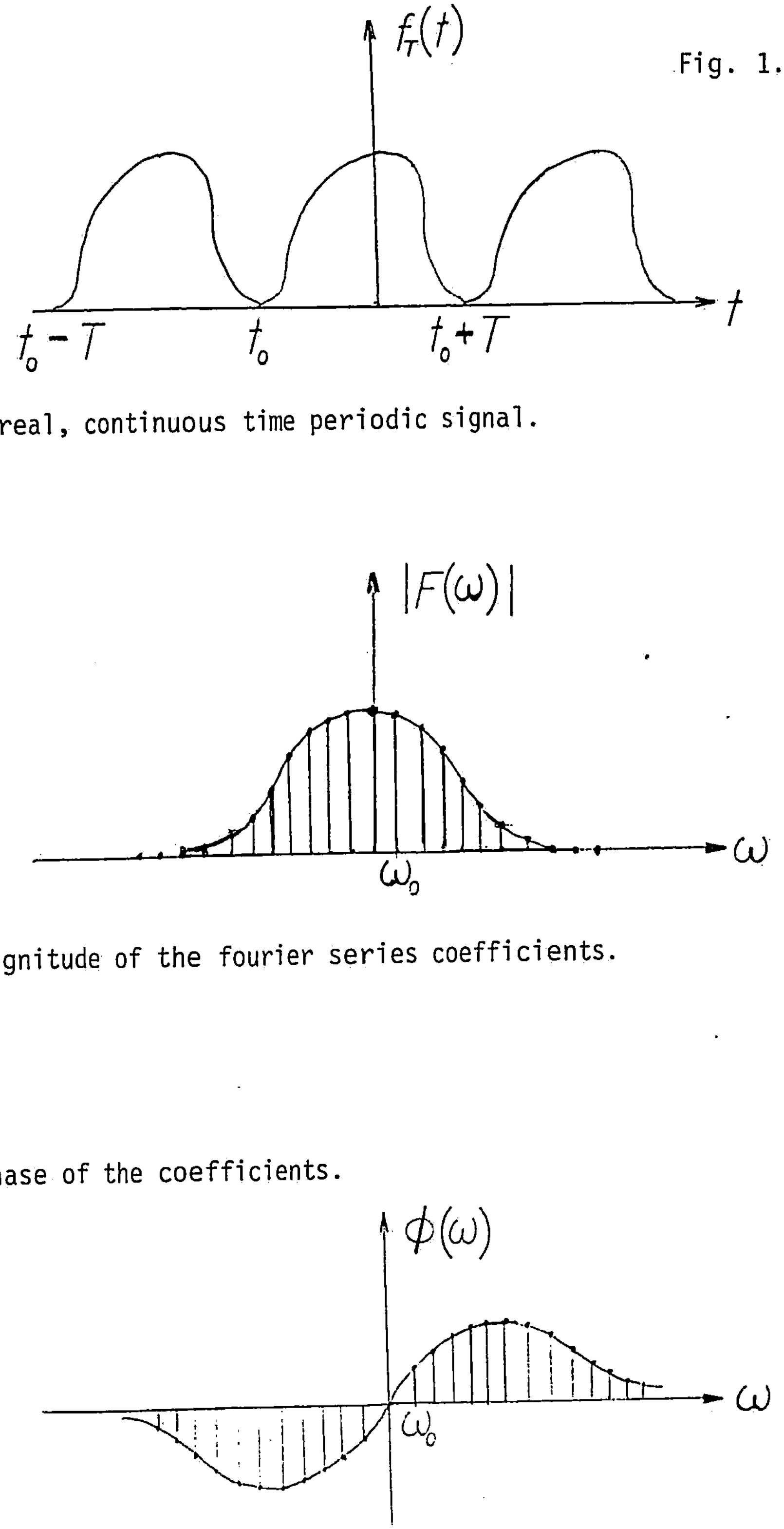
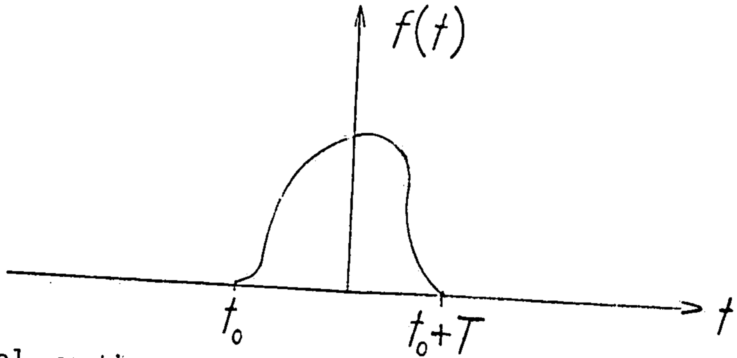


Fig. 1.2

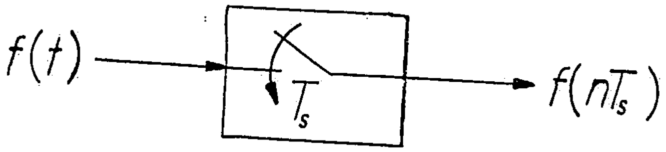
a) A real, continuous time periodic signal.

b) Magnitude of the fourier series coefficients.

c) Phase of the coefficients.



A real, continuous time, aperiodic signal.



$T_s$ = Sampling Period

$\omega_s$ = Sampling Frequency (radians/sec.)

$= 2\pi/T_s$

$N$ = Number of Samples in Duration T

$= T/T_s$

Fig. 1.3a) The original signal, and the sampler.

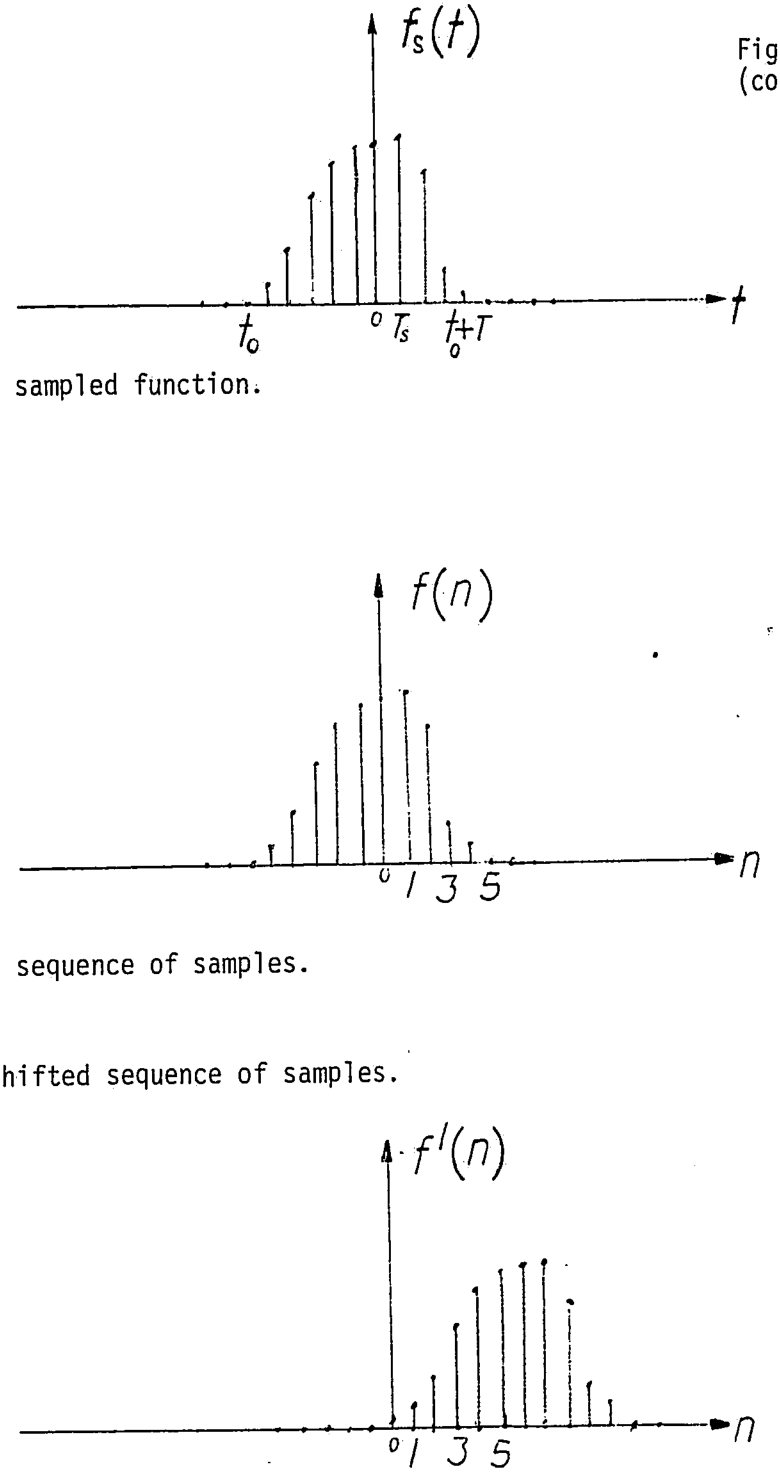


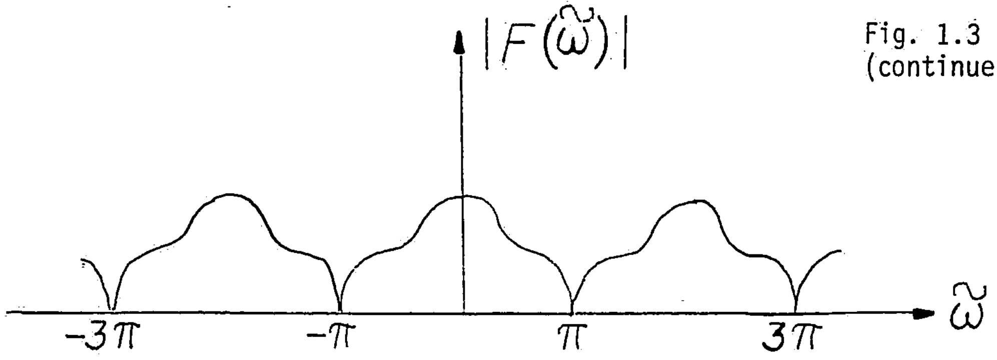
Fig. 1.3
(continued)

b) The sampled function.

c) The sequence of samples.

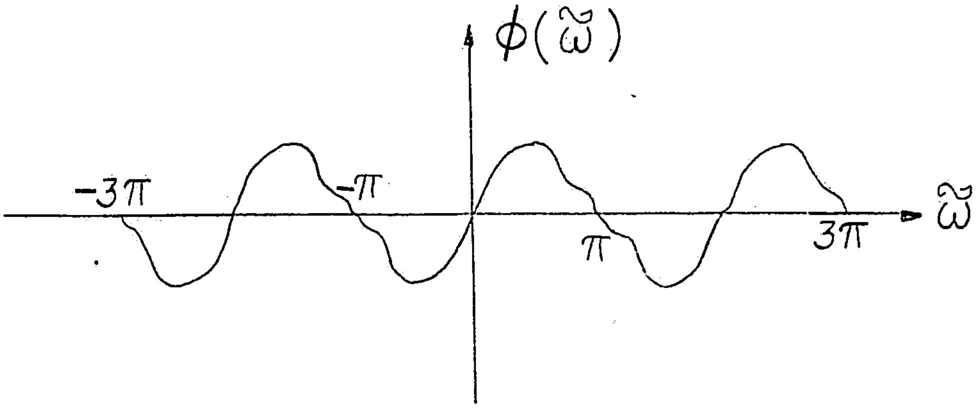
d) A shifted sequence of samples.

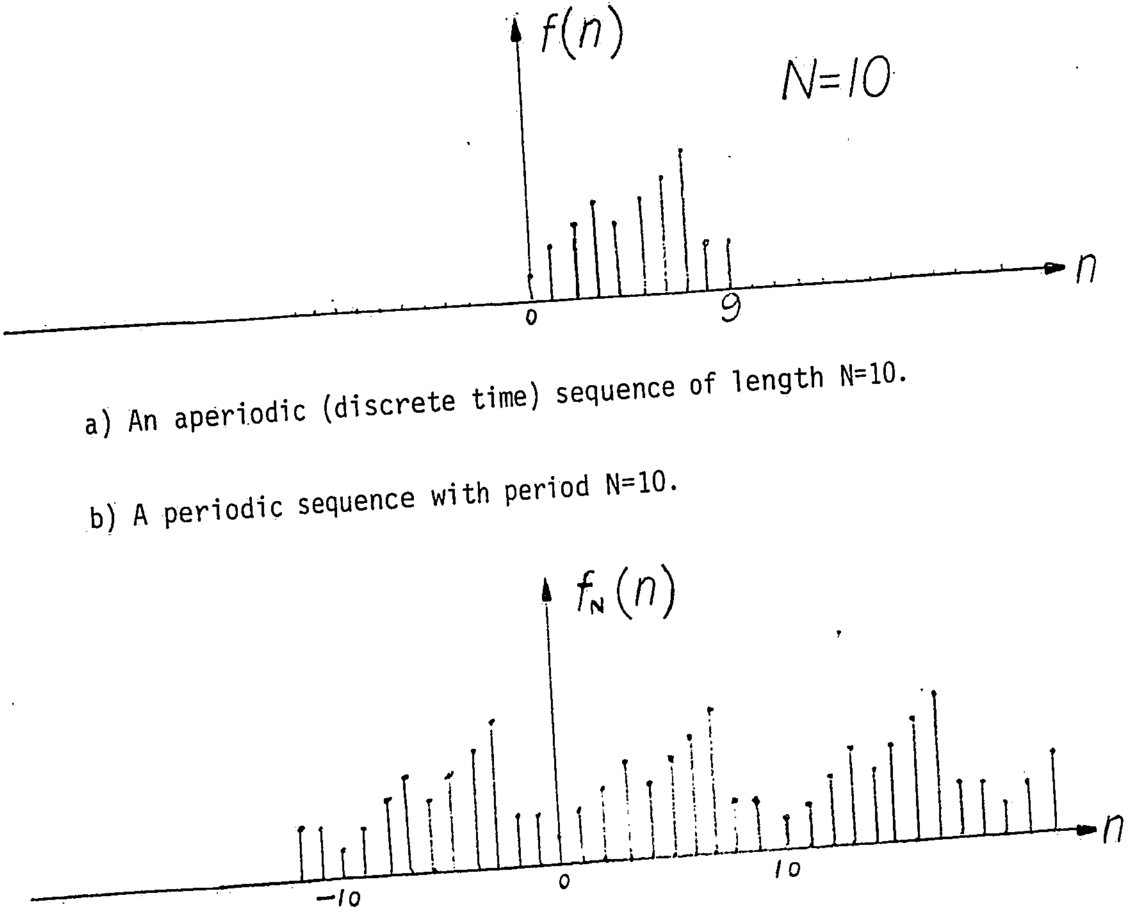
Fig. 1.3
(continued)



e) Magnitude of the spectral density of the FT, and

f) its phase.

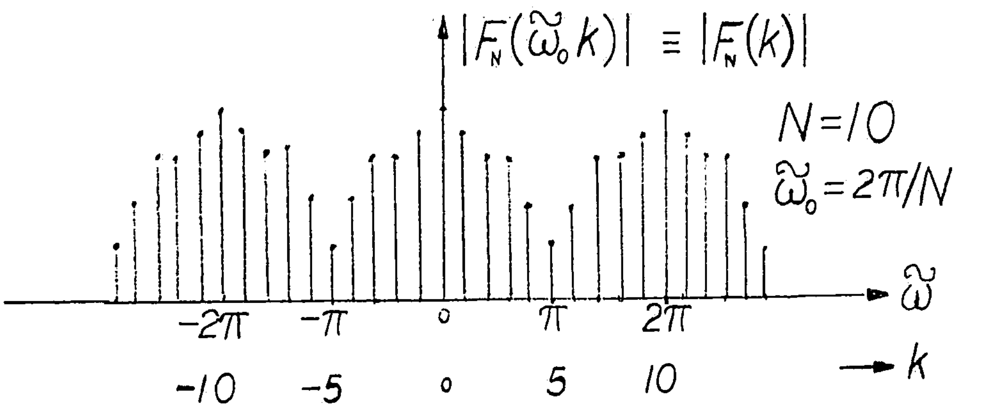




a) An aperiodic (discrete time) sequence of length N=10.

b) A periodic sequence with period N=10.

Fig. 1.4



c) Magnitude of the coefficients,

d) and their phase.

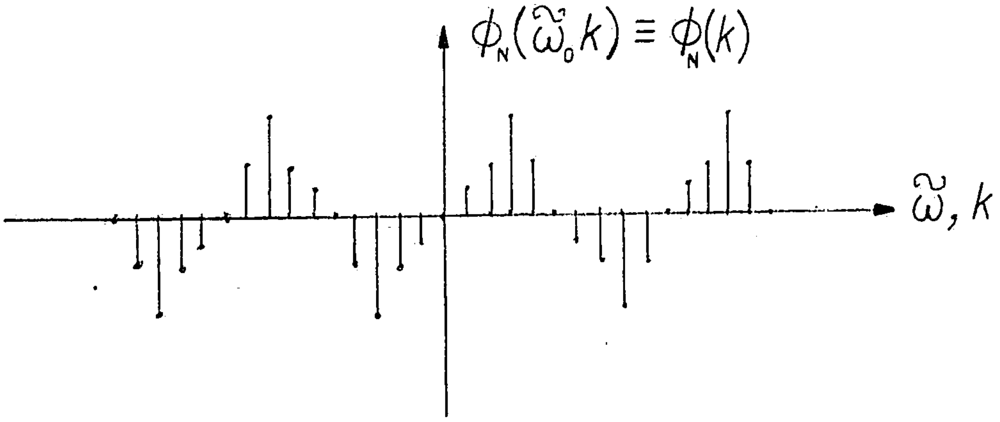


Fig. 1.4 (continued) Discrete-time Fourier Series coefficients.

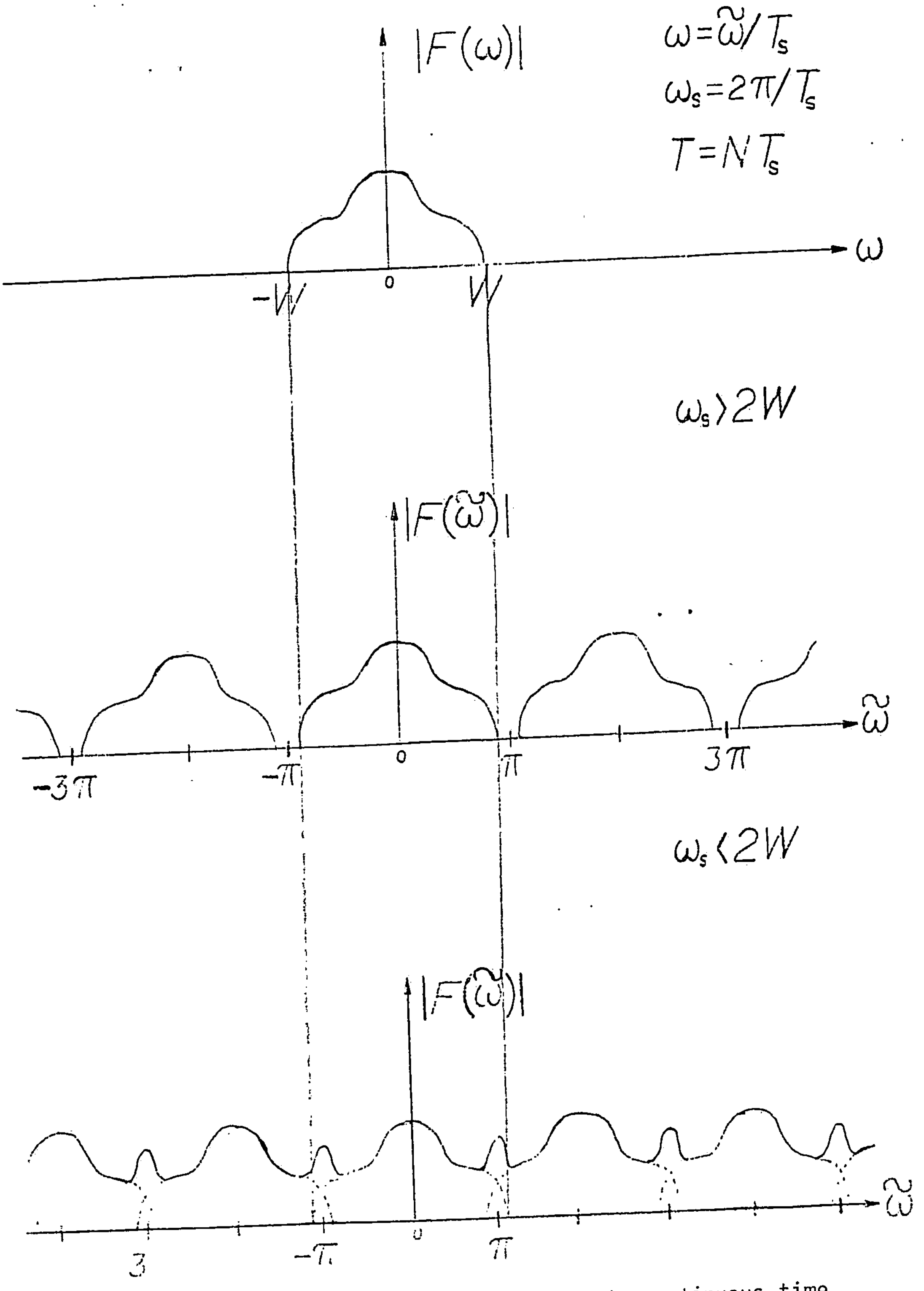
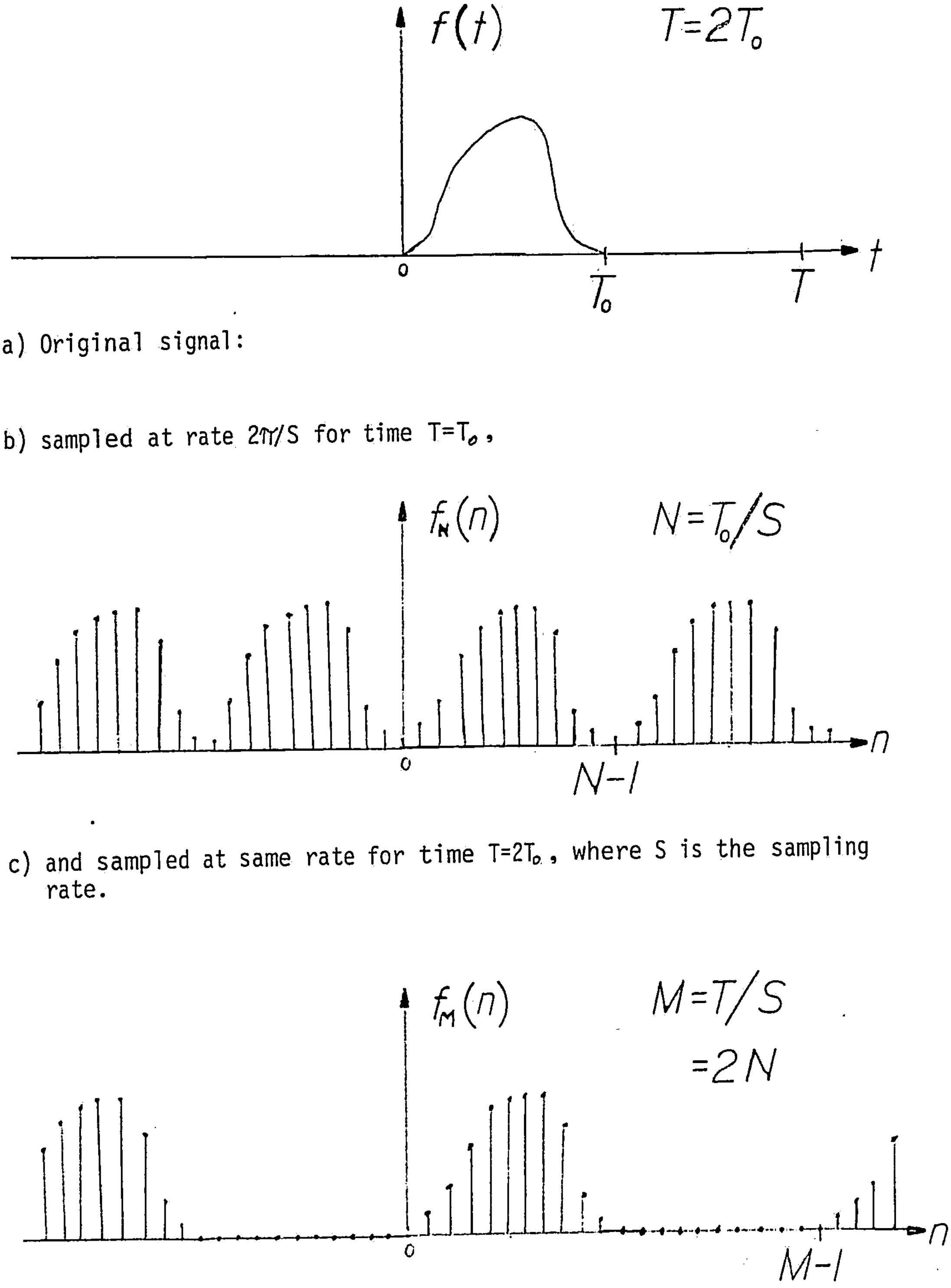


Fig. 1.5 a) Spectral density of a bandlimited, continuous time, aperiodic signal.
b) Spectral density for case of oversampling, and
c) undersampling.

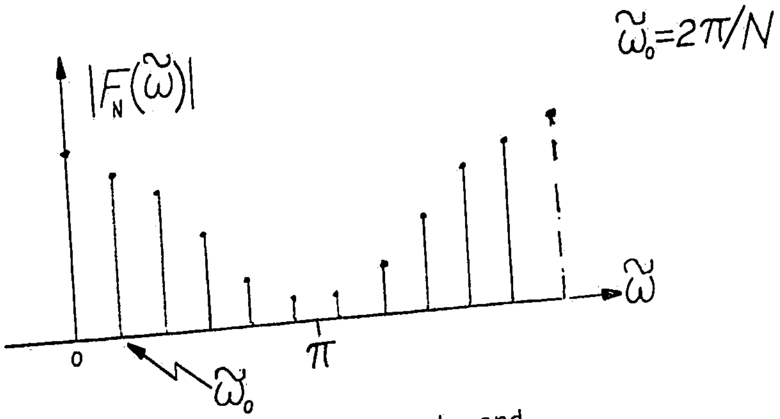


a) Original signal:

b) sampled at rate $2\pi/S$ for time $T=T_0$,

c) and sampled at same rate for time $T=2T_0$, where S is the sampling rate.

Fig. 1.6



d) DFS coefficients for sequence from b, and

e) DFS coefficients for sequence from c. Bold coefficients in e exactly match the frequencies of coefficients in d.

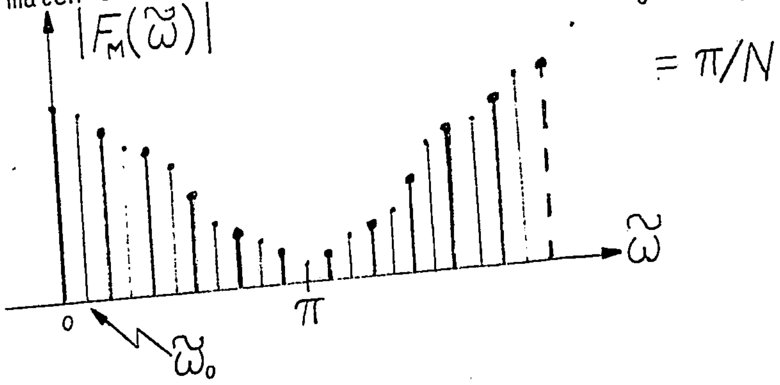


Fig. 1.6 (continued)

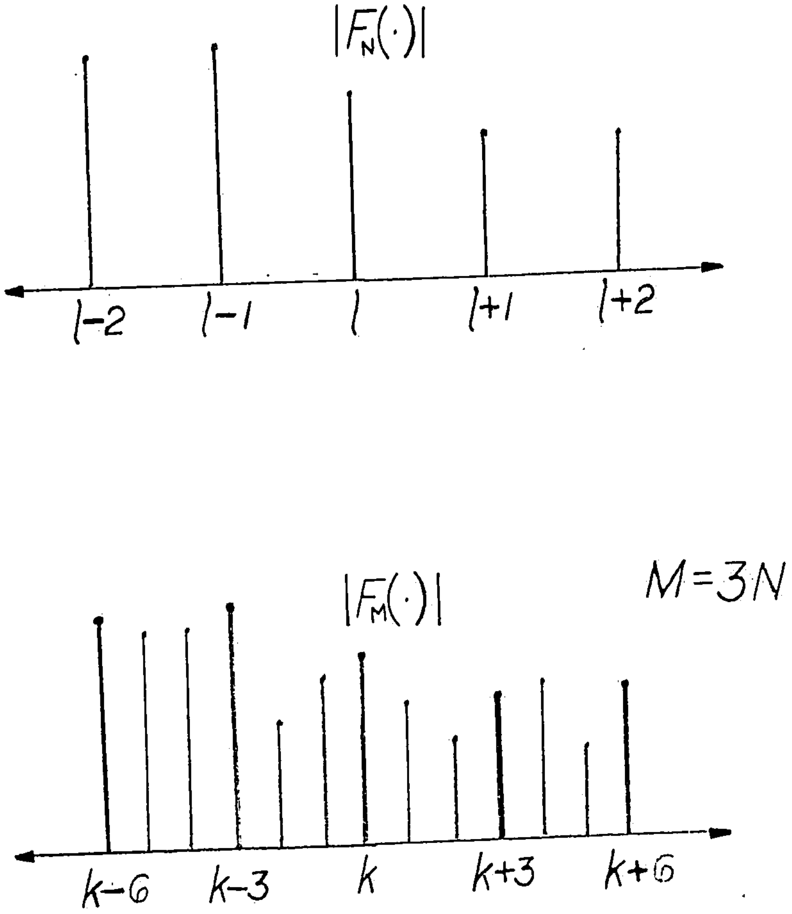


Fig. 1.7 Part of the DFS spectra of a) $F_N(l)$ and b) $F_M(k)$, both for the special case of M=3N.

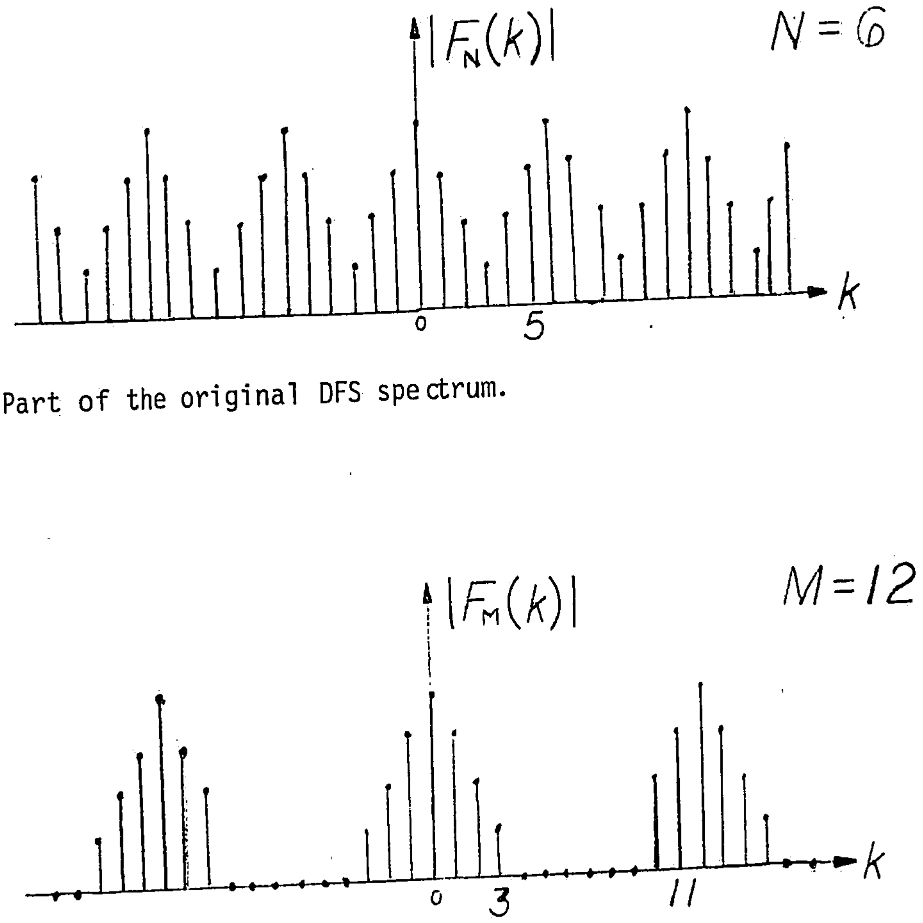
## Interploation in the Frequency Domain

| | |
|---|---|
| $f_N(n) \xrightarrow{M>N} f_M(n)$ | Zero extend $f_N(n)$ by appending M-N zeros to it. |
| $f_N(n) \xrightarrow{\mathcal{F}_N} F_N(\tilde{\omega})$ | N point DFT. |
| $f_M(n) \xrightarrow{\mathcal{F}_m} F_M(\tilde{\omega})$ | M point DFT. |

$F_M(\tilde{\omega})$ is the interpolated version of $F_N(\tilde{\omega})$.

$$\left\{\begin{array}{l}\text{Appending zeroes in}\\ \text{the sequence (time)}\\ \text{domain}\end{array}\right\} \dashrightarrow \left\{\begin{array}{l}\text{Interpolating in the}\\ \text{frequency domain}\end{array}\right\}$$

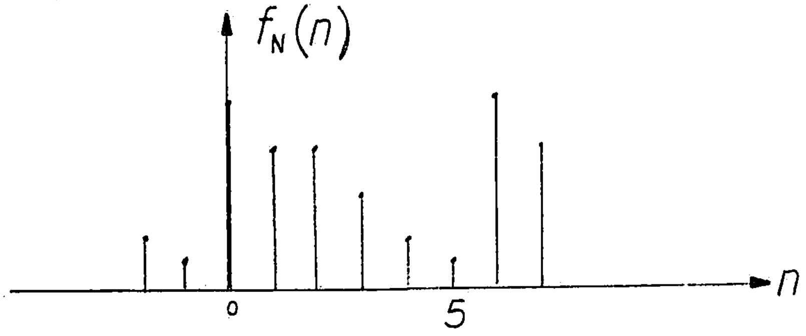
Fig. 1.8



a) Part of the original DFS spectrum.

b) Zero extended spectrum. Note that the spectra are the same for coefficients (-J,J), where for N even, J=N/2. For N odd, J=(N-1)/2.

Fig. 1.9



c) Part of the original DFS spectrum.

d) Interpolated sequence. from the IFT of the spectra of b).

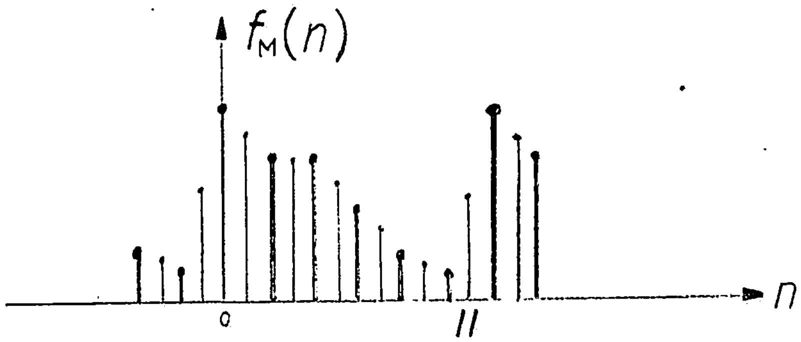


Fig. 1.9 (continued)

## Interpolation in the (time) Sequence Domain

$F_N(\tilde{\omega}) \xrightarrow{M > N} F_M(\tilde{\omega})$ Append zeroes to $F_N(\tilde{\omega})$.

$F_N(\tilde{\omega}) \xrightarrow{\tilde{\mathcal{F}}_N^{-1}} f_N(n)$ N point IDFT

$F_M(\tilde{\omega}) \xrightarrow{\tilde{\mathcal{F}}_M^{-1}} f_M(n)$ M point IDFT

Here, $f_M(n)$ is the interpolated version of $f_N(n)$.

$$\left\{\begin{array}{l}\text{Zero extend in the}\\ \text{frequency domain}\end{array}\right\} \dashrightarrow \left\{\begin{array}{l}\text{Interpolation in the}\\ \text{sequence domain}\end{array}\right\}$$

Fig. 1.10

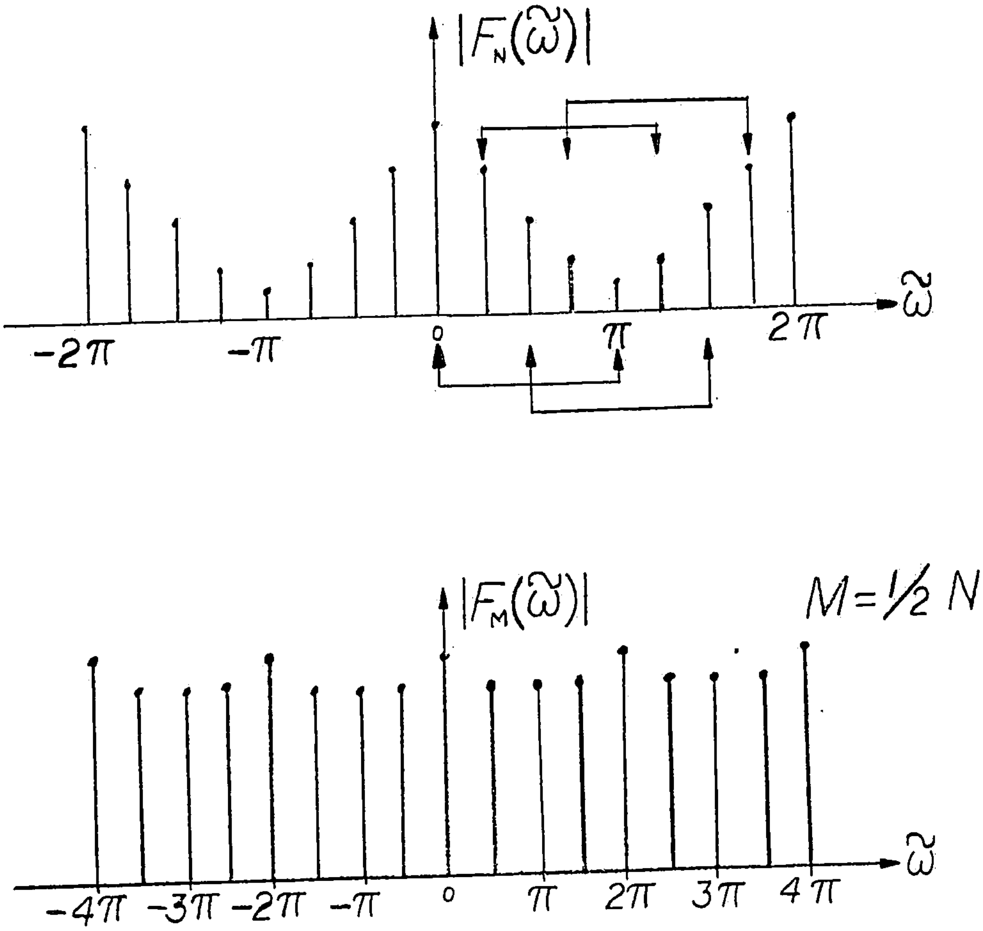


Fig. 1.11 a) Original DFS coefficients for N=8.
b) Coefficients of the "wrapped" spectrum, where M=(1/2)N=4. Note that wrapping may lead to an aliased spectrum.

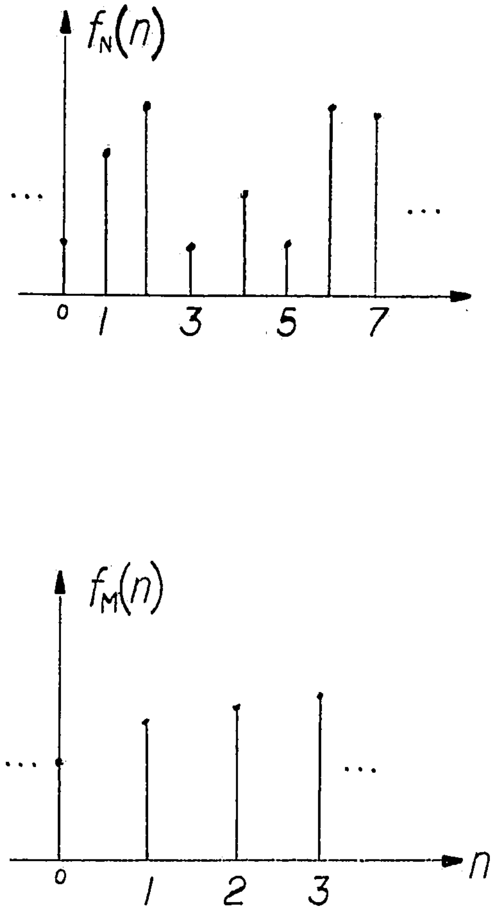
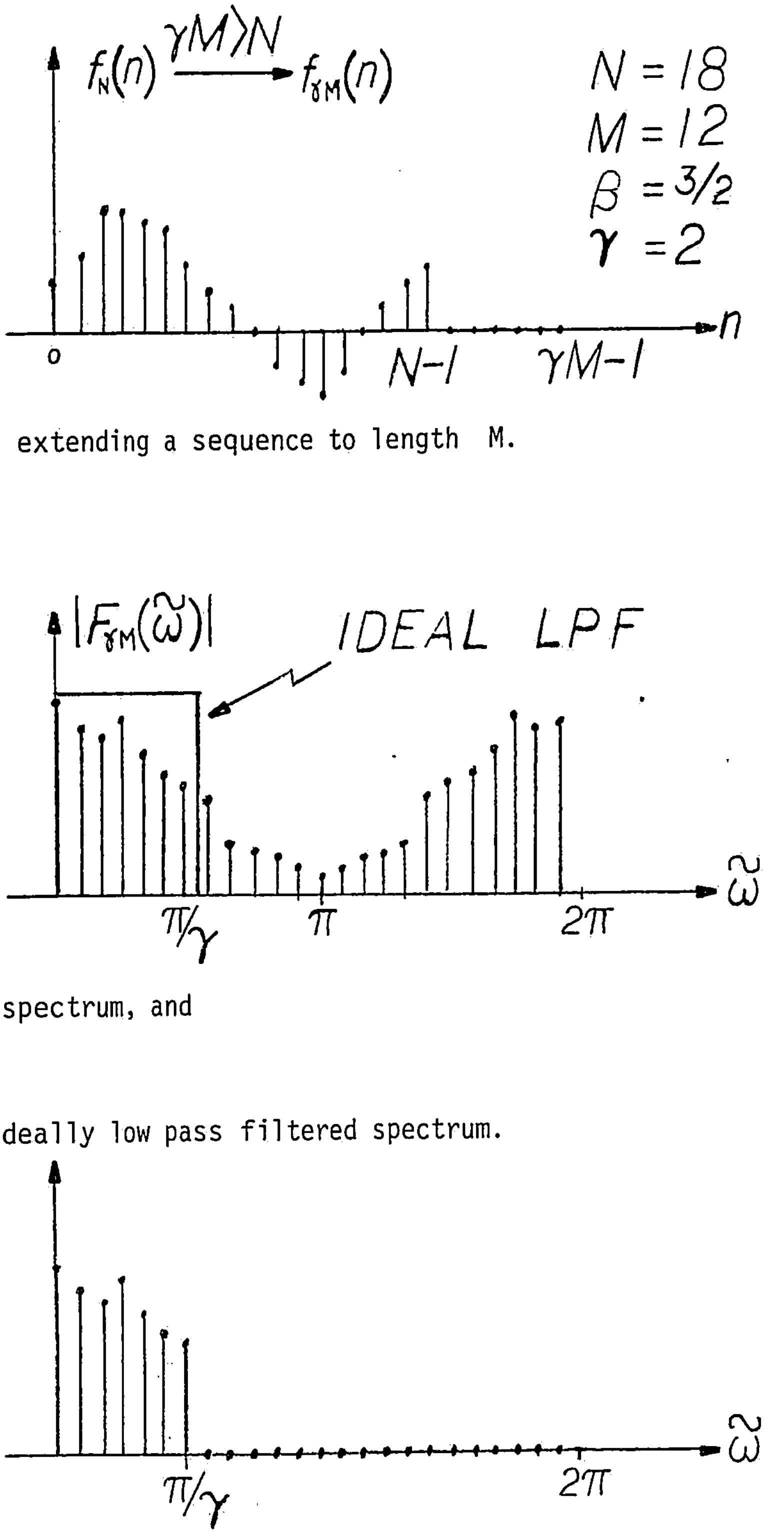


Fig. 1.11 (continued)

c) Original sequence.
d) Sequence from wrapped spectrum. Note that in general this sequence may not look similar to the original. Because the high frequencies may have been obscured, it will tend to be smoother than the original.

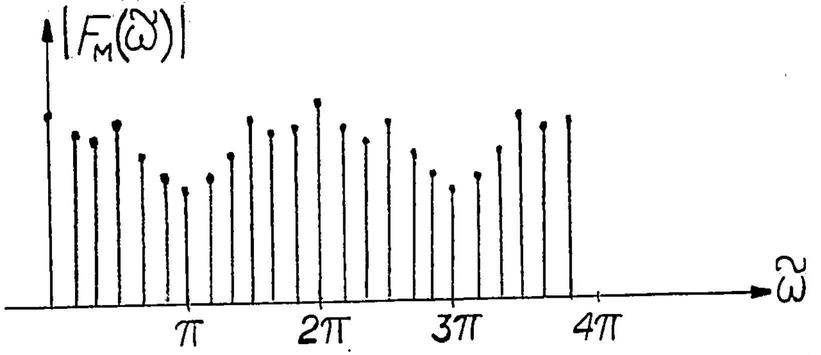


a) Zero extending a sequence to length M.

b) DFT spectrum, and

c) an ideally low pass filtered spectrum.

Fig. 1.12



d) Wrapping of the spectrum in (c) yields this spectrum and
e) this sequence. This sequence is a decimated version of (a). Every other element of (e) equals every third element of (a).

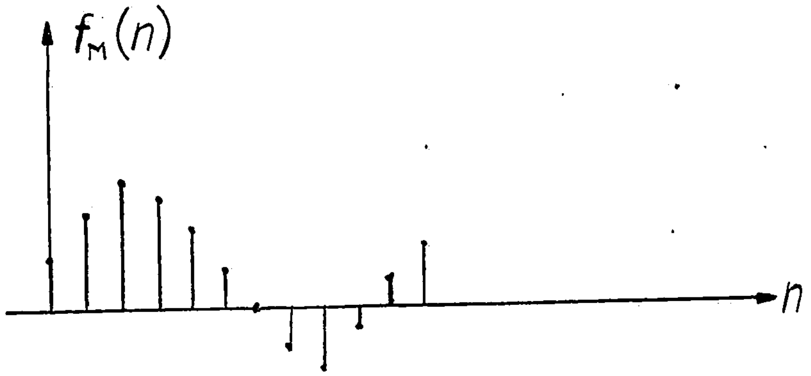


Fig. 1.12 (continued)

1.12 f) Decimation in the Time Domain

$M > N$
$\beta = N/M$

If $\beta$ is an integer, then $\gamma = \beta$
If $\beta$ is not an integer, then $\gamma$ = the next integer larger than $\beta$

| | |
|---|---|
| $f_N(n) \xrightarrow{\gamma M > N} f_{\gamma M}(n)$ | Zero extend the sequence |
| $f_{\gamma M}(n) \xrightarrow{\mathcal{F}_{\gamma M}} F_{\gamma M}(k)$ | DFT of zero extended sequence ($\gamma M$ point) |
| $F_{\gamma M}(k) \xrightarrow{\gamma M > M} F_M(k)$ | Filter and wrap the resultant spectrum |
| $F_M(k) \xrightarrow{\mathcal{F}^{-1}} f_M(n)$ | IDFT to obtain the decimated sequence |

Fig. 1.12 (continued)

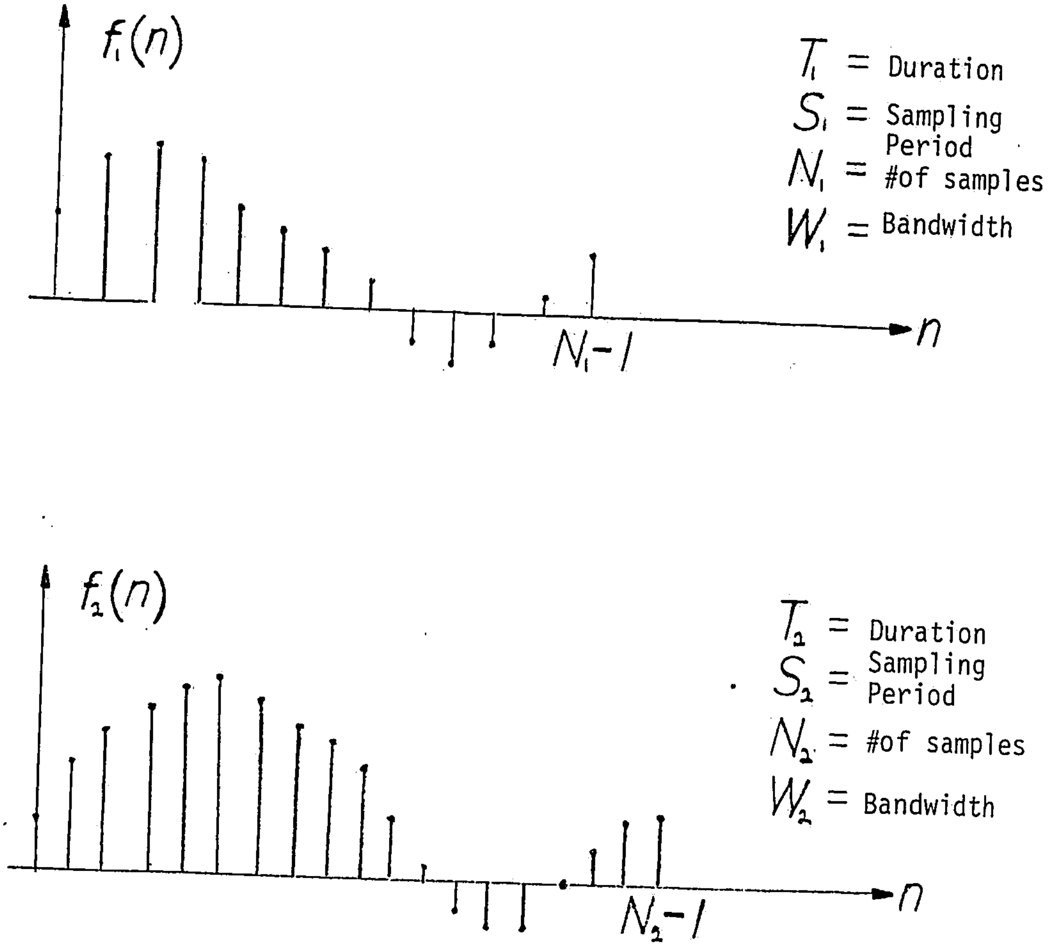


Fig. 1.13 Two sequences (a) and (b). The relation T=NS holds in general where Tis the duration of the original signal. W is its bandwidth.

Case 1:

$T_2 > T_1$

$N_2 > N_1$

$S_2 = S_1$

$W_2 < W_1$

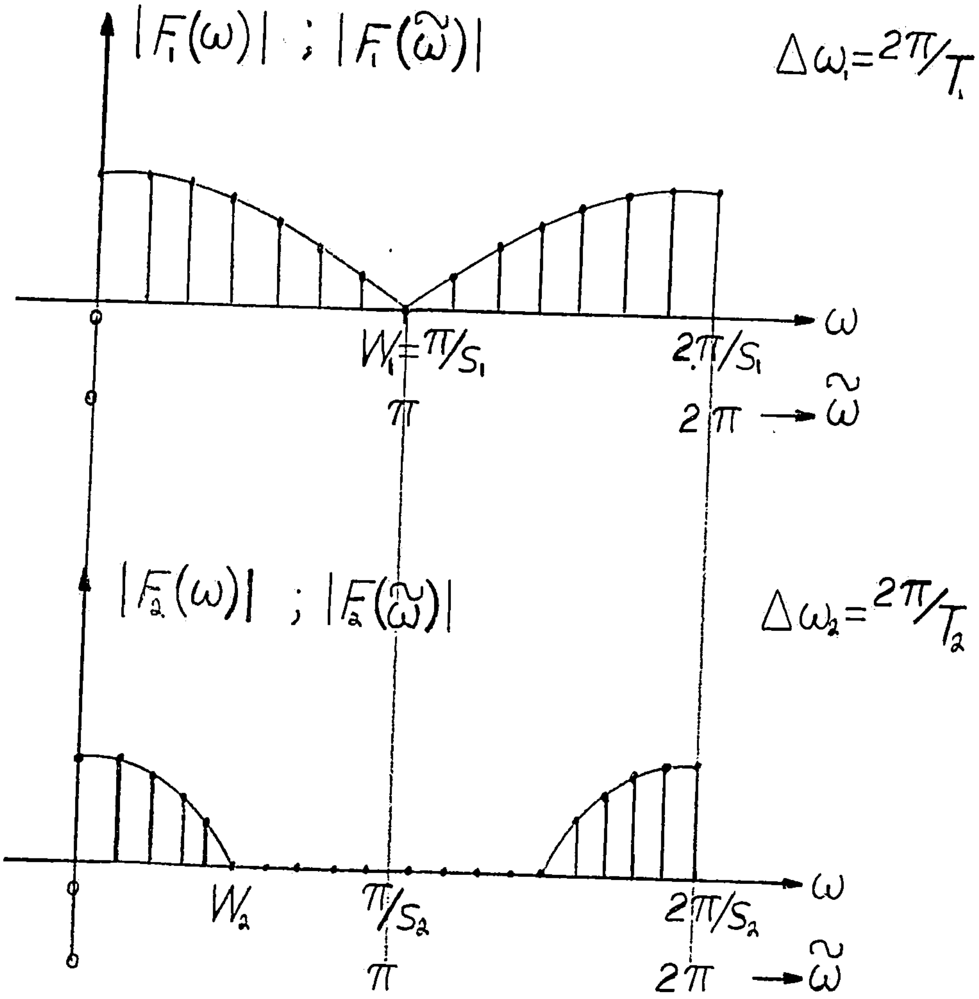


Fig. 1.14 a) and b) Spectra of the two sequences of fig. 1.13 for case 1. $\Delta\omega$ is the spacing between coefficients.

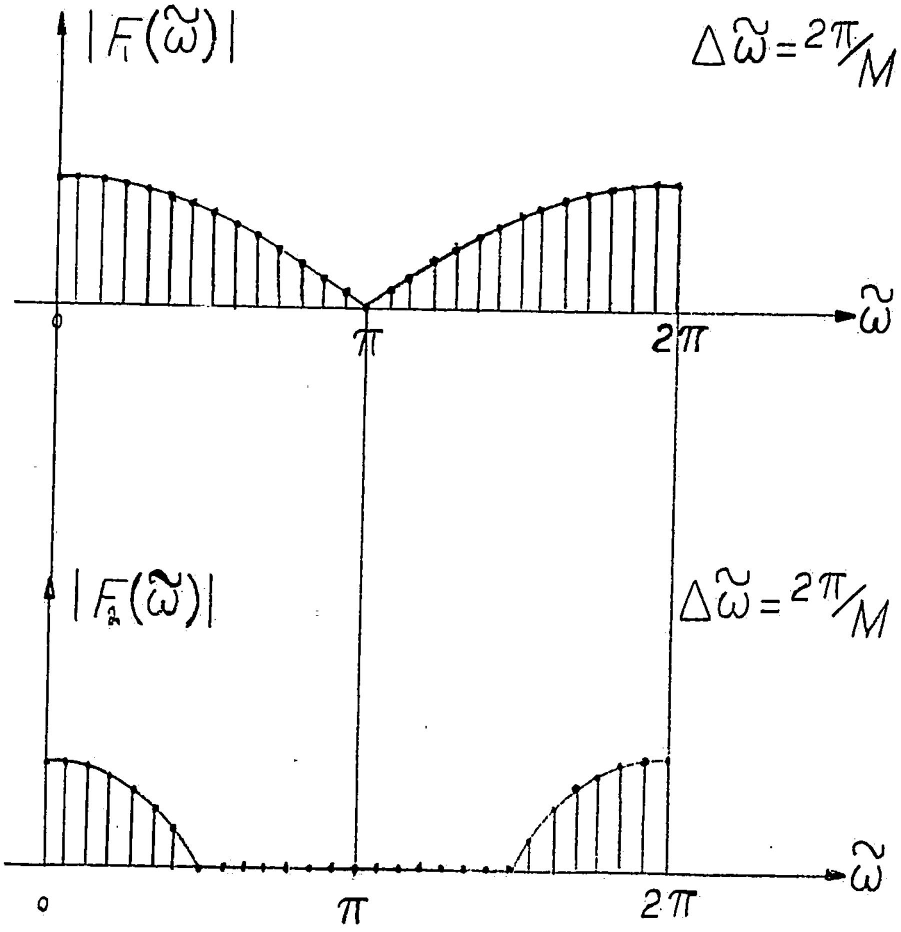


Fig. 1.14 (continued)

c) and d) Spectra of the zero extended sequences. Each spectrum contains M coefficients where the spacing is the same for both spectra.

Case 2: $T_2 = T_1$

$N_2 > N_1$

$S_2 < S_1 \Longrightarrow \Delta\omega_2 = \Delta\omega_1$

$W_2 = W_1$ But $\Delta\tilde{\omega}_2 \neq \Delta\tilde{\omega}_1$

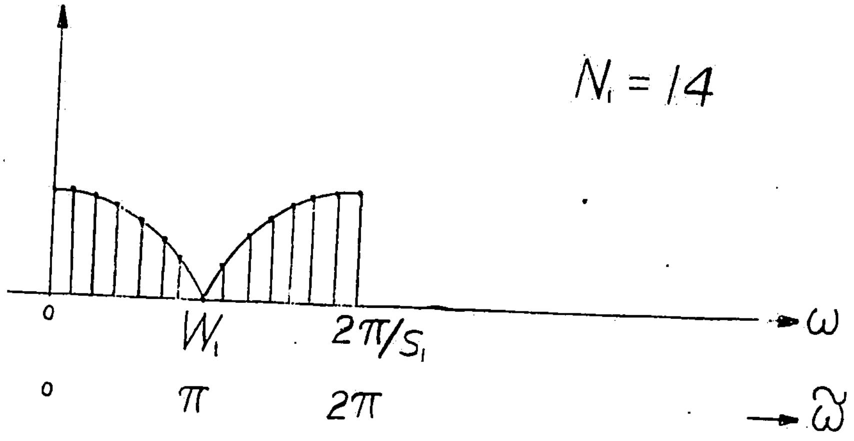


Fig. 1.15 Case 2 a)Spectrum of the first sequence

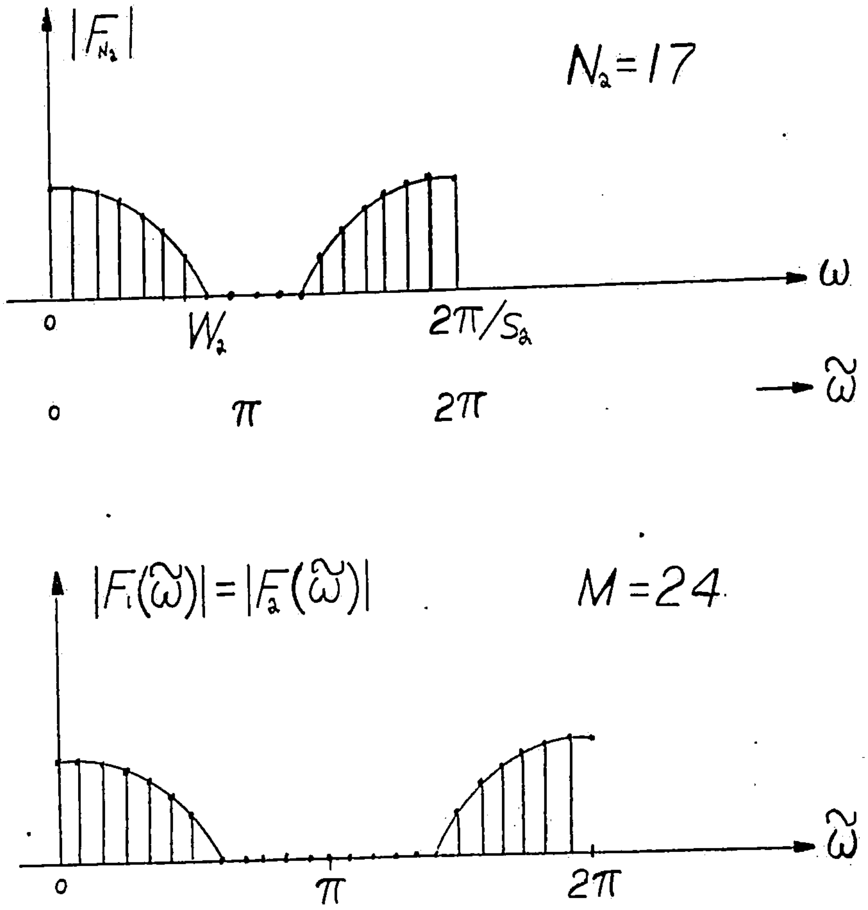


Fig. 1.15 (continued) b)Spectrum of the second sequence.
c)Spectra of (a) and (b) zero extended. In this case they are equal.

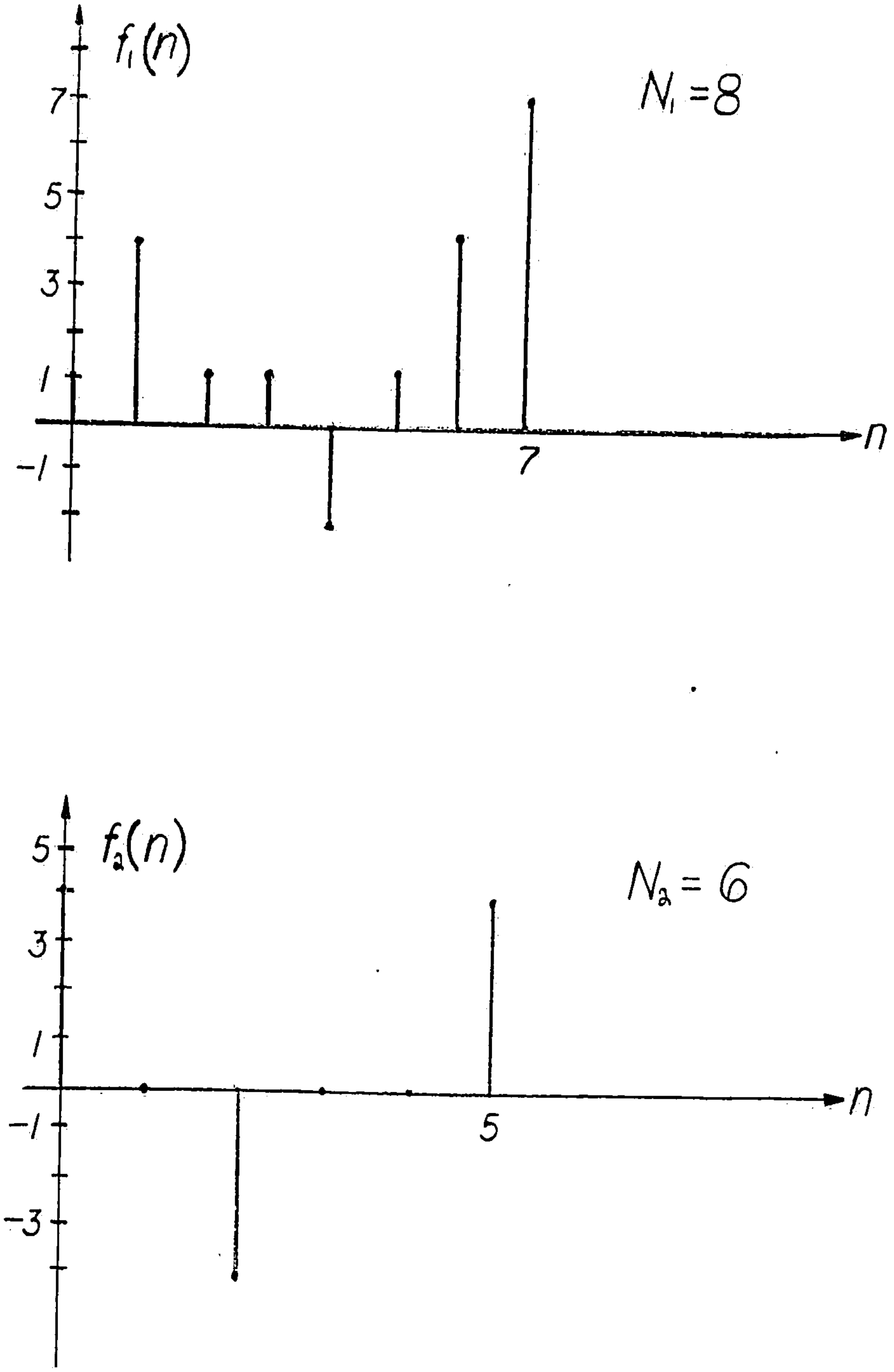


Fig. 1.16 Sequences of lengths 8 and 6. The sequences are:
a) 1,4,1,1,-2,1,4,7 and
b) 4,0,-4,0,0,4.

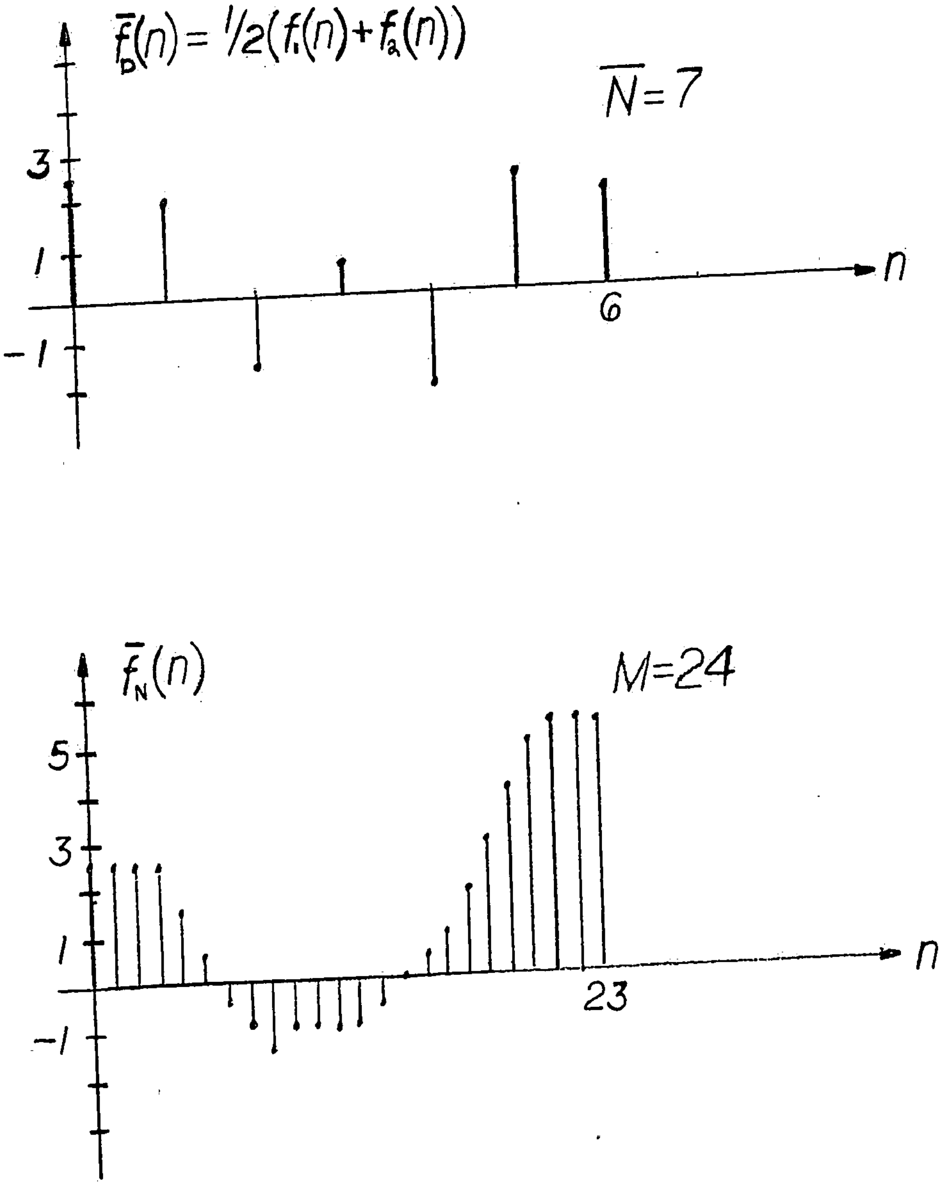


Fig. 1.16 (continued)

c) Case 1 average, ie "Direct Average," N=7.
d) Case 2 average, ie "Normalized Average", where M=7.
Note that the averages are quite disimilar.

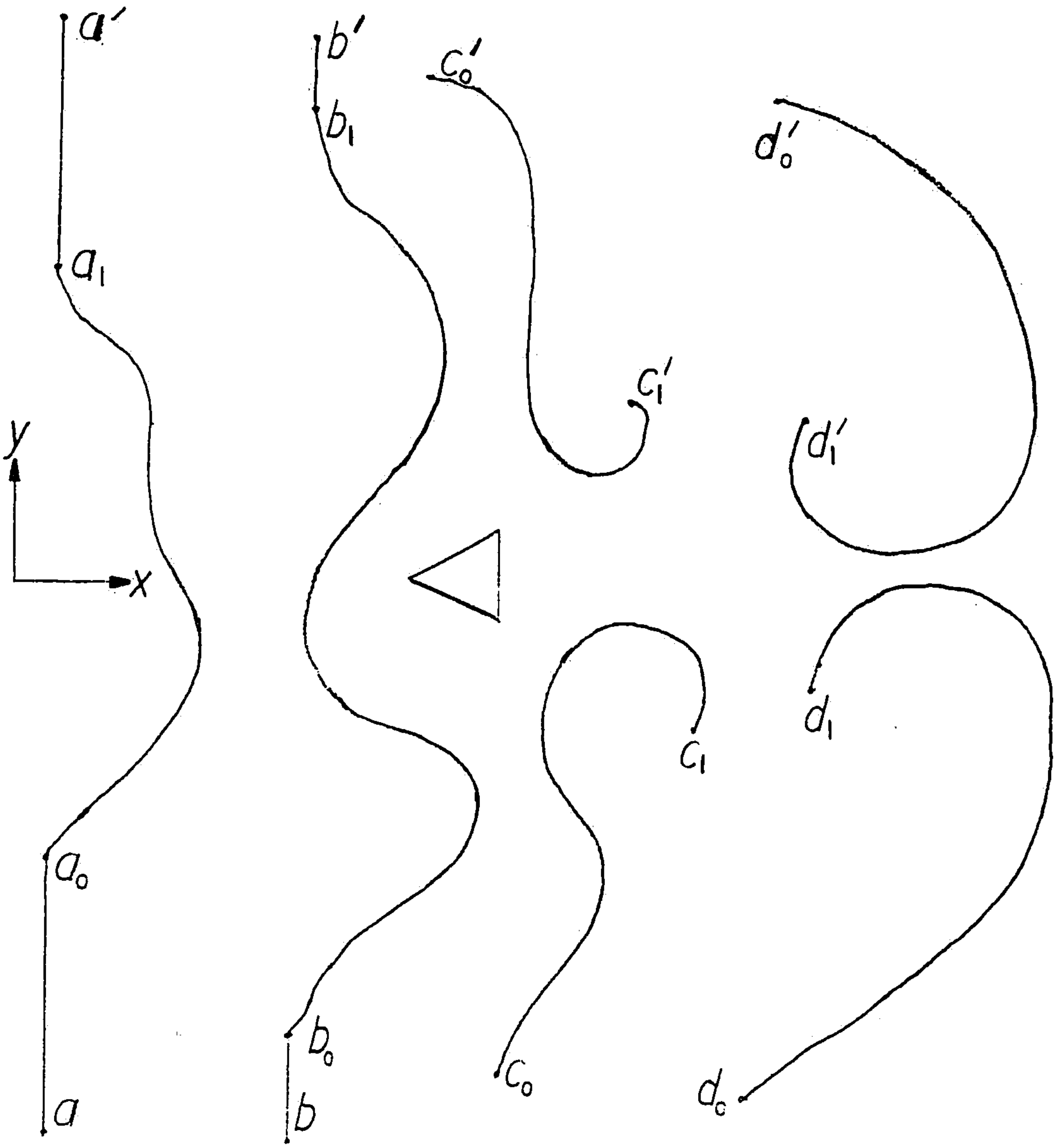


Fig. 2.1 Flow patterns. The triangular object is an obstruction in the flow (not necessarily stationary). As the water (fluid) is swept around the object, the hydrogen bubble timelines create different patterns. The direction of the flow is from left to right.

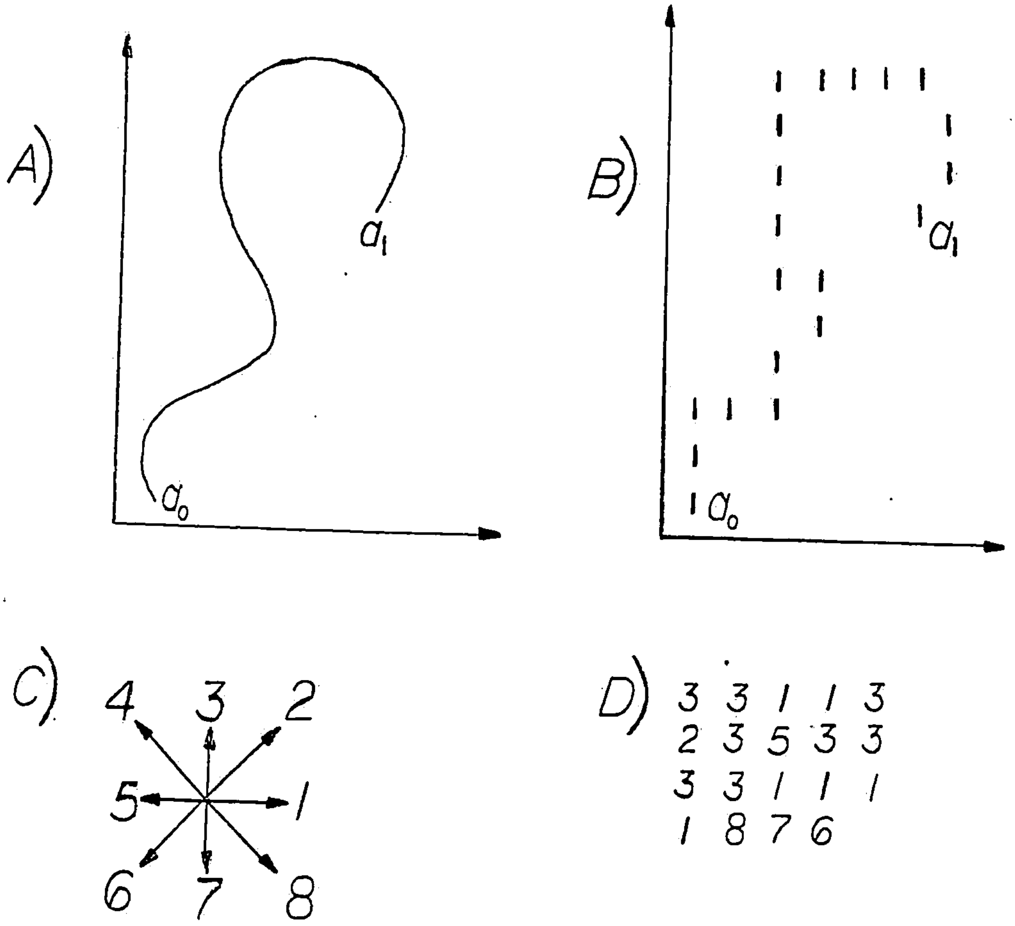


Fig. 2.2 a)Curve in continuous spatial domain
b)Original discretized version
c)Possible movements from any one pixel, ie the pixel (or point) of interest, POI
d)19 element chain code describing 20 element pattern

E) Freeman's Corner Cutting Matrix

| | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 |
|---|---|---|---|---|---|---|---|---|
| 1 | | | 2 | 3 | | 7 | 8 | |
| 2 | | | | 3,3 | 3 | | 1 | 1,1 |
| 3 | 2 | | | | 4 | 5 | | 1 |
| 4 | 3 | 3,3 | | | | 5,5 | 5 | |
| 5 | | 3 | 4 | | | | 6 | 7 |
| 6 | 7 | | 5 | 5,5 | | | | 7,7 |
| 7 | 8 | 1 | | 5 | 6 | | | |
| 8 | | 1,1 | 1 | | 7 | 7,7 | | |

F) 3 2 2 3
3 3 3 3
3 2 1 1
1 8 7 6

Shortened
Chain Code
(16 elements)

G) The processed pattern. Note that the start and end points are unaltered.

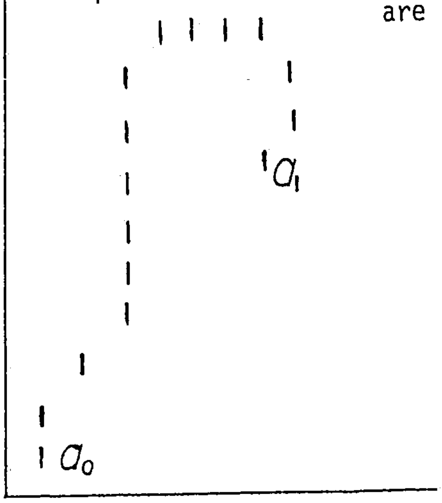


Fig. 2.2 (continued)

Direction of Arrival (DOA) at the Point of Interest (POI) is given by the average of the last M DOA's. That is, by summing from j=1 to M,

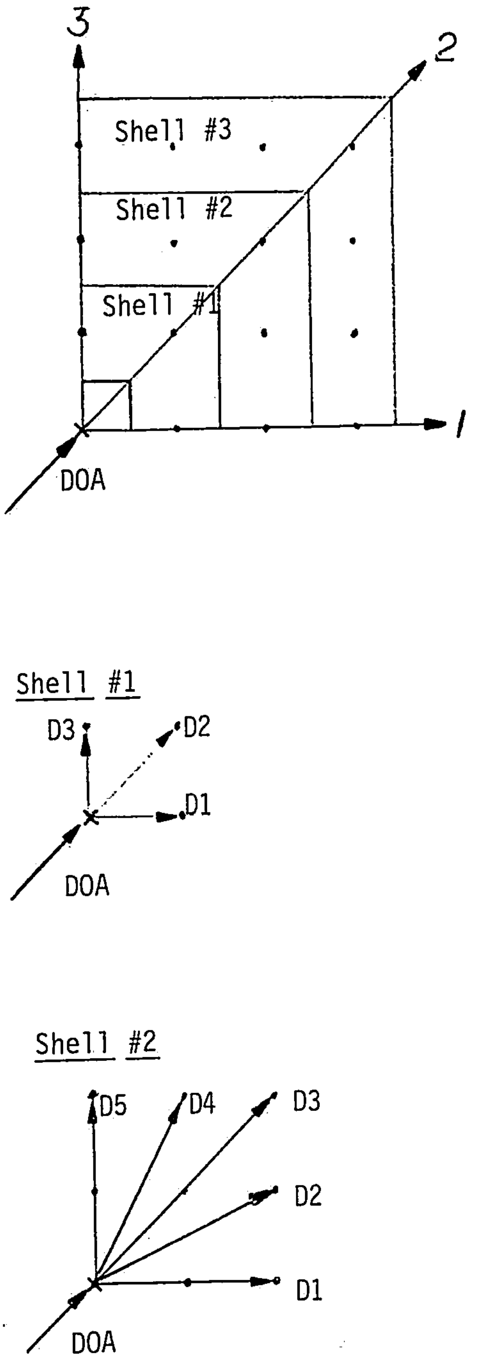
$$DOA(k) = 1/M \ \Sigma \ DOA(k-j) \quad \text{modified modulo 8}$$

The modified modulo 8 value of a number N is calculated as

$$\begin{aligned} N \text{ (mmod 8)} &= N \text{ (modulo 8)} \quad \text{if this is not zero.} \\ &= 8 \text{ if the above is zero.} \end{aligned}$$

Fig. 2.3a Modified Modulo Eight

Fig. 2.3b See Fig. 2.2c.



Shell #1

Shell #2

a) Wedge of interest where the direction of arrival (DOA) at the point of interest (POI) is even. Here, DOA=2 and the number of shells N=3.

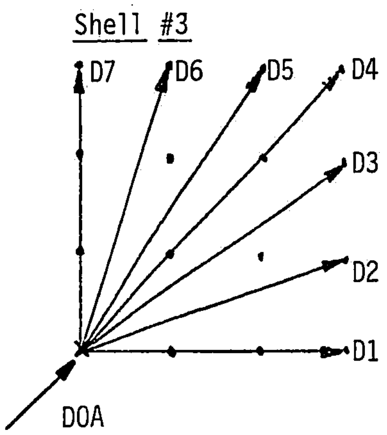
b) Direction Weights for Shell #1:

D1=1.00
D2=2.00
D3=3.00

c) Direction Weights for Shell #2:

D1=1.00
D2=1.59
D3=2.00
D4=2.41
D5=3.00

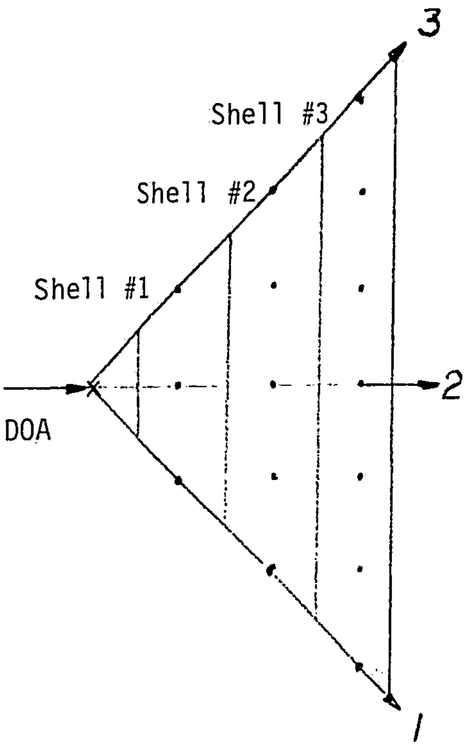
Fig. 2.4



d) Direction Weights for Shell 3:

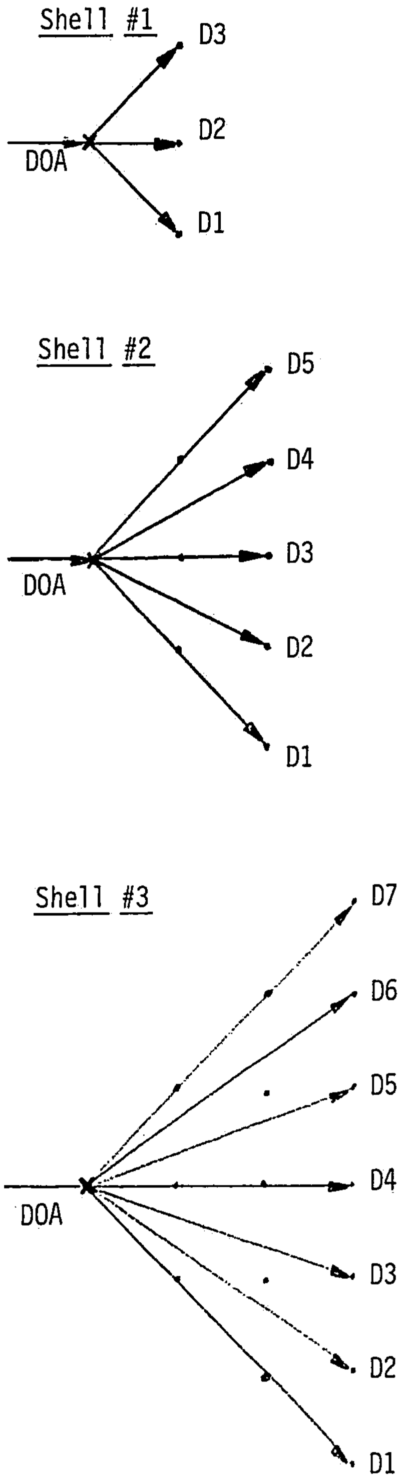
D1=1.00
D2=1.41
D3=1.75
D4=2.00
D5=2.25
D6=2.59
D7=3.00

Fig. 2.4 (continued)



a) Wedge of interest for odd DOA. Here, DOA=1 and number of shells is N=3.

Fig. 2.5



b) Direction Weights for Shell 1:

D1=1.00
D2=2.00
D3=3.00

c) Direction Weights for Shell 2:

D1=1.00
D2=1.41
D3=2.00
D4=2.59
D5=3.00

d) Direction Weights for Shell 3:

D1=1.00
D2=1.25
D3=1.59
D4=2.00
D5=2.41
D6=2.75
D7=3.00

Fig. 2.5 (continued)

a) Shell Weights: Wt(i), where i=shell #'s 1 through N.

eg. Wt(1)=7/15
Wt(2)=5/15
Wt(3)=3/15

b) Direction Weights Matrix: Parameters of DW(i,j,k)

ie. i=1 if DOA is odd
i=2 if DOA is even

j=Shell Number (1 through N)

k=Point in Shell (eg. 1 through 7)

c) Direction Weight matrix when DOA is odd, ie. i=1 in DW(1,j,k)

| k = | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| j= 1 | 1.00 | 2.00 | 3.00 | - | - | - | - |
| 2 | 1.00 | 1.41 | 2.00 | 2.59 | 3.00 | - | - |
| 3 | 1.00 | 1.25 | 1.59 | 2.00 | 2.41 | 2.75 | 3.00 |

d) Direction Weight matrix when DOA is even, ie. i=2 in DW(2,j,k)

| k= | 1 | 2 | 3 | 4 | 5 | 6 | 7 |
|---|---|---|---|---|---|---|---|
| j=1 | 1.00 | 2.00 | 3.00 | - | - | - | - |
| 2 | 1.00 | 1.59 | 2.00 | 2.41 | 3.00 | - | - |
| 3 | 1.00 | 1.41 | 1.75 | 2.00 | 2.25 | 2.59 | 3.00 |

Fig. 2.6

Maxk (j) = Point # (k) of the maximum intensity pixel in shell number j, where j=1 to 3. Maxk will range 1 through 7.

Int (j) = Intensity of the pixel at point Maxk(j) in shell j, ie. the maximum intensity of all pixels in shell j.

Wt (j) = Shell weight of the j-th shell (as in fig. 2.6a).

DW (i,j,maxk(j)) = Direction weight of the maximum intensity pixel in shell j, given that the DOA is odd or even (i=1 or 2) and that the point is in position Maxk(j).

The Maximum Intensity Direction (MID) is defined as a normalized weighted average, given as below:

$$\text{MID} = \text{Nearest Integer of } \frac{\text{Sum of } \begin{bmatrix}\text{Direction}\\ \text{Weights}\end{bmatrix} \cdot \begin{bmatrix}\text{Intensity}\\ \text{Values}\end{bmatrix}\begin{bmatrix}\text{Shell}\\ \text{Weights}\end{bmatrix}}{\text{Sum of } \begin{matrix}\text{Intensity}\\ \text{Values}\end{matrix} \ \begin{matrix}\text{Shell}\\ \text{Weights}\end{matrix}}$$

$$= \text{Nearest Integer of } \frac{\sum_{j=1}^{N} DW(i,j,Maxk(j))*Wt(j)*Int(j)}{\sum_{j=1}^{N} Wt(j)*Int(j)}$$

Deviation from DOA (DDOA) and the New Direction (NEWD) of departure from the POI are given by:

$$\text{DDOA} = \text{MID} - 2$$

$$\begin{aligned}\text{NEWD} &= \text{DOA} + \text{MID} - 2\\ &= \text{DOA} + \text{DDOA}\end{aligned}$$

Fig. 2.7 Maximum Intensity Direction (and its parameters) and the New Direction of departure.

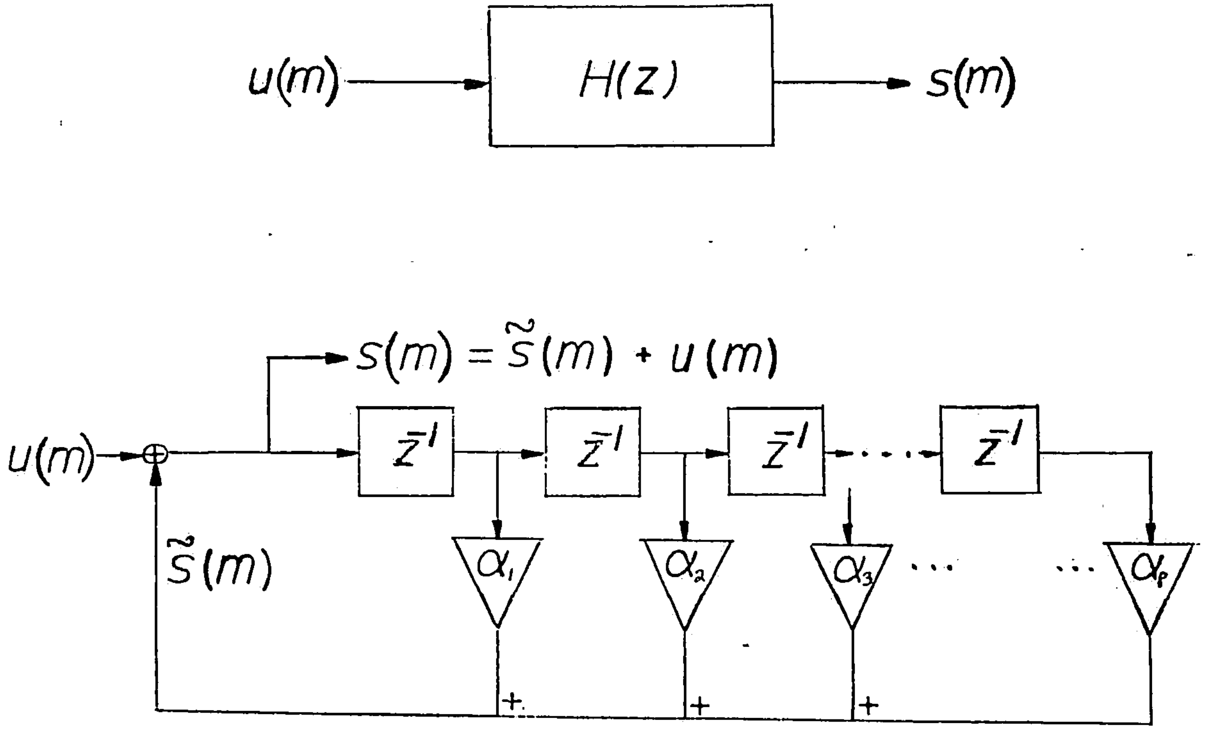
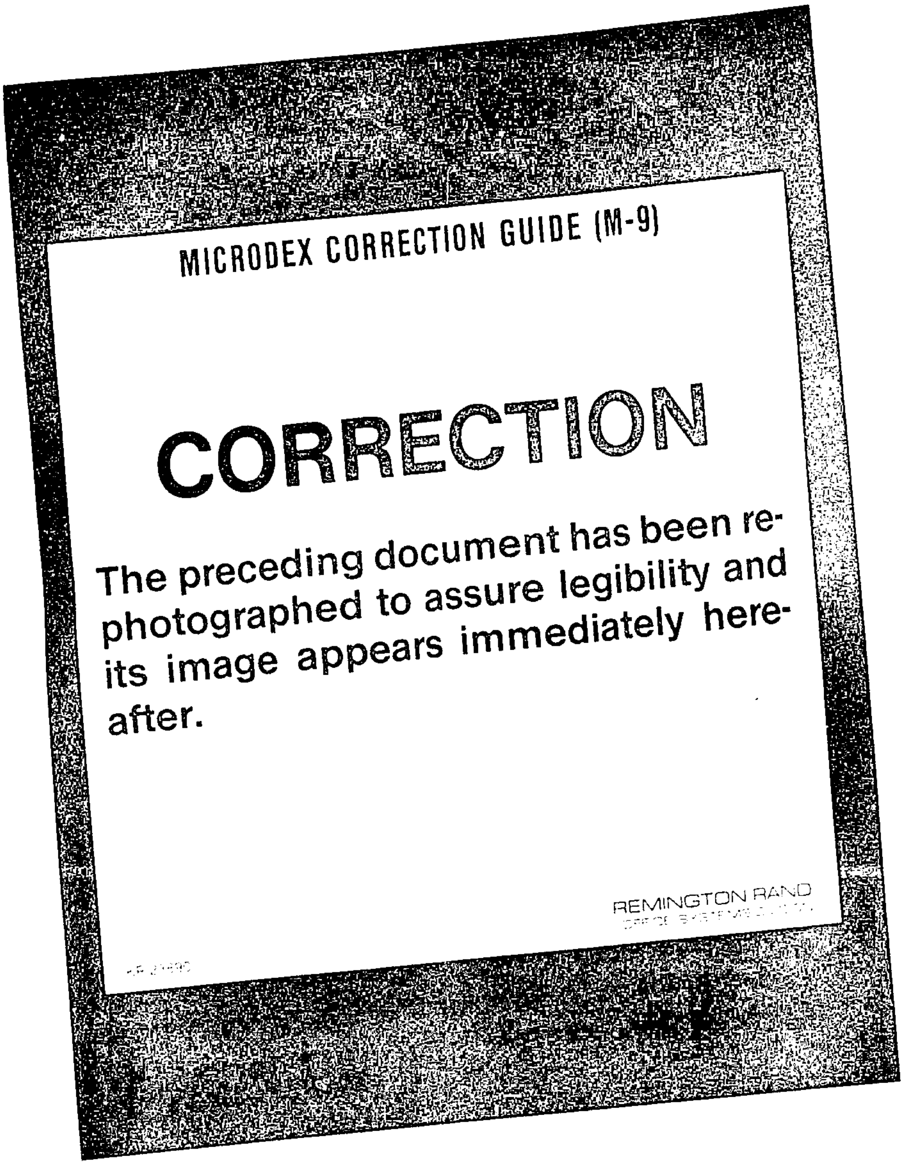


Fig. 2.8 a) Transfer function between source and output.
b) All pole model: a p tap filter.



MICRODEX CORRECTION GUIDE (M-9)

# CORRECTION

The preceding document has been rephotographed to assure legibility and its image appears immediately hereafter.

REMINGTON RAND

Fig. 2.8 a) Transfer function between source and output.
b) All pole model: a p tap filter.

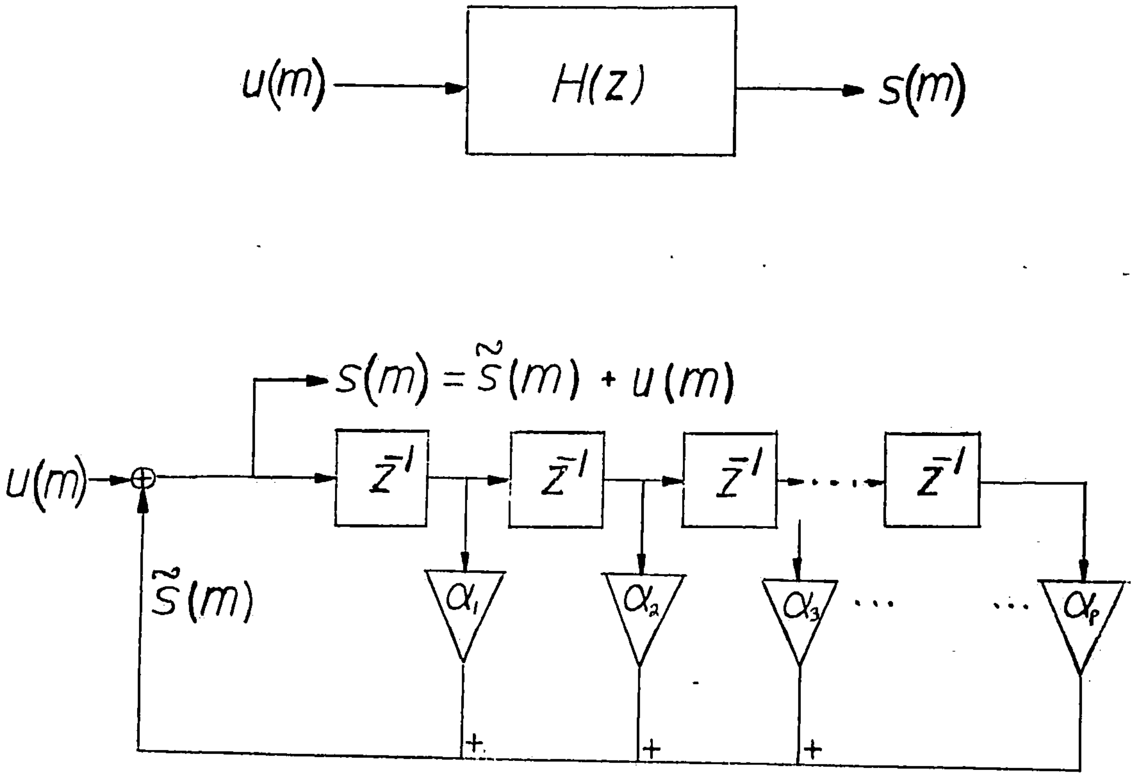


Fig. 2.8 a) Transfer function between source and output.
b) All pole model: a p tap filter.

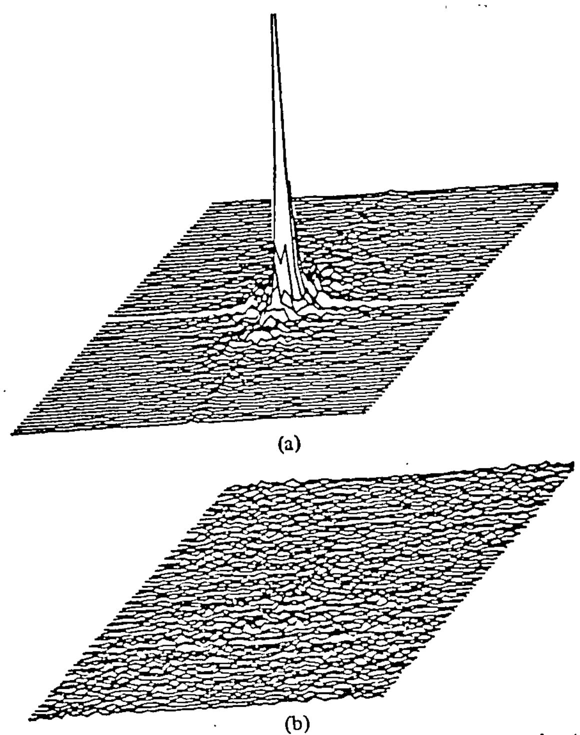


Fig. 3. Perspective plots of the magnitude of the 2-D Fourier transform (a) of the original image, and (b) of the prediction error signal ($P = 8$, $M = 32$) (the prediction error is magnified three times relative to the original image).

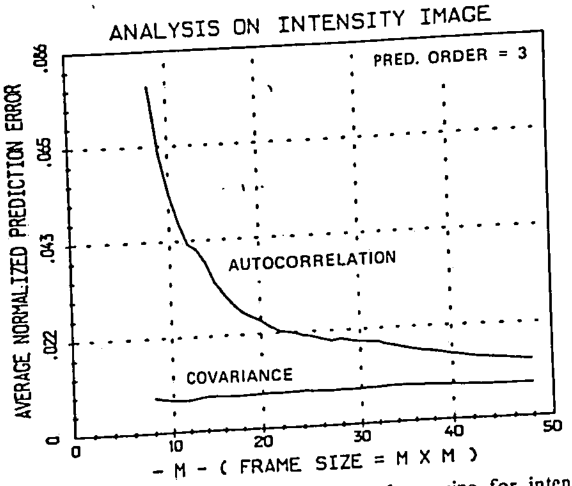


Fig. 4. Variation of prediction error versus frame size for intensity images ($P = 3$).

Fig. 2.9 Prediction error results, and comparison of autocorrelation and covariance errors (after Maragos et al, ref. 15).

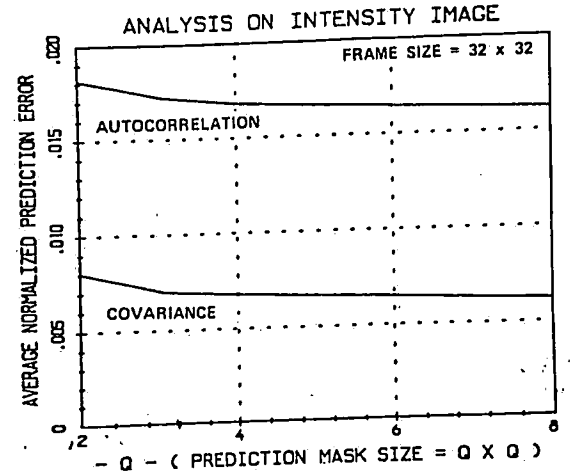


Fig. 5. Variation of prediction error versus size of prediction mask for intensity images ($M = 32$).

Fig. 2.9 (continued) (After Maragos et. al. (ref. 15))

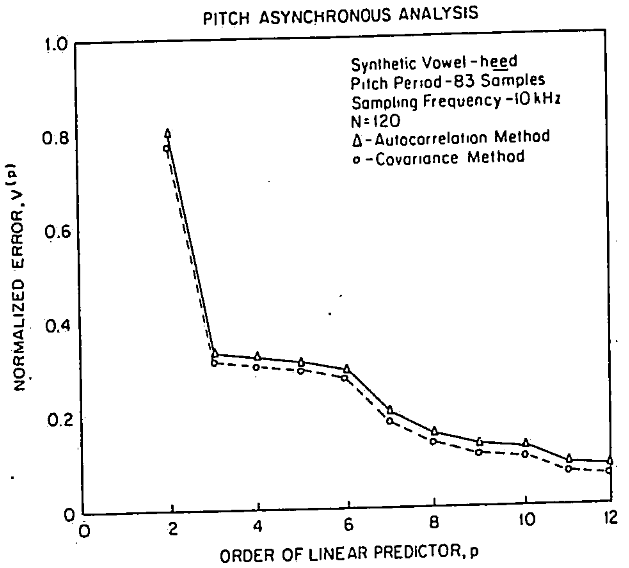


Fig. 8.11 Variation of prediction error with predictor order, $p$, for voiced section of a synthetic vowel—pitch asynchronous analysis. (After Chandra and Lin [15].)

Fig. 2.10 Comparison of autocorrelation and covariance error for asynchronous prediction of speech (from ref. 14, but after Chandra and Lin (ref. 16)).

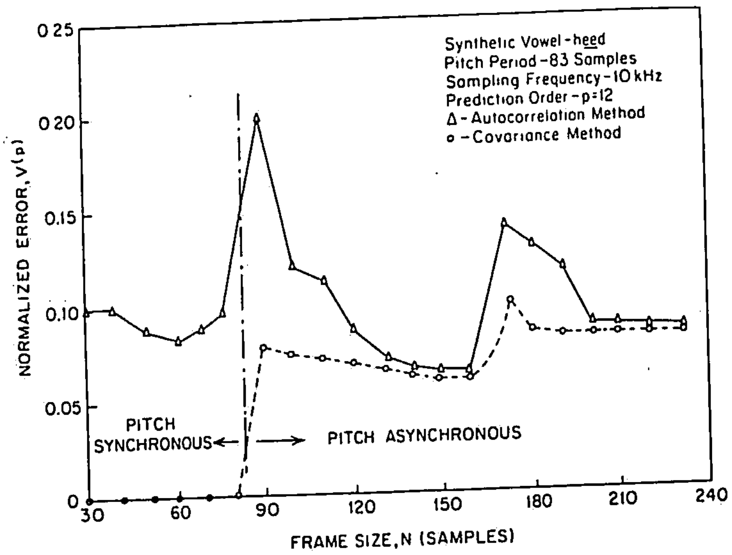


Fig. 8.12 Variation of prediction error with section length, *N*, for a voiced section of synthetic speech. (After Chandra and Lin [15].)

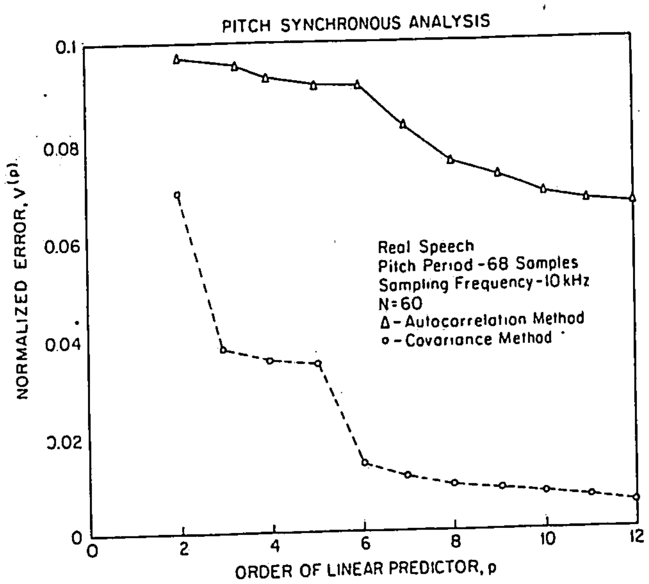
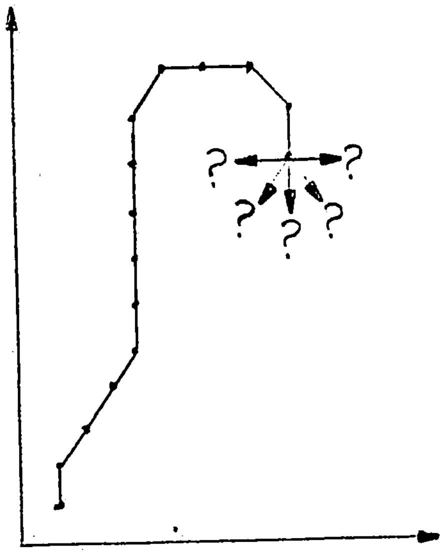


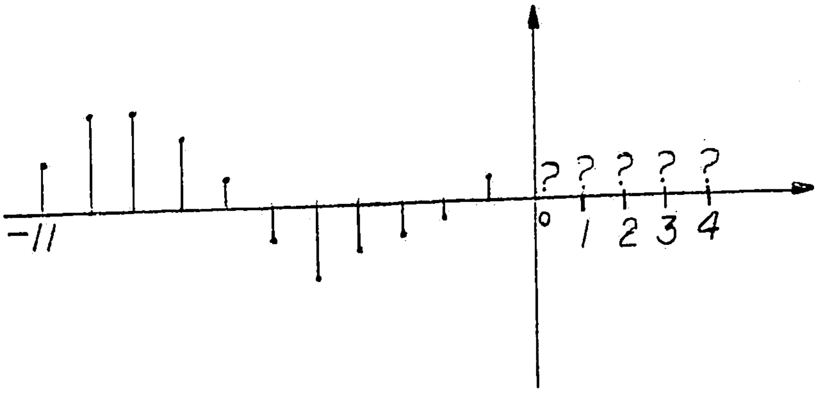
Fig. 8.13 Variation of prediction error with predictor order, *p*, for a voiced section of a natural vowel—pitch synchronous analysis. (After Chandra and Lin [15].)

Fig. 2.10 (continued) (From ref. 14, but after Chandra and Lin (ref. 16))

Fig. 2.11

a) A partially tracked curve, and
b) a partially completed sequence (that might represent the tangent angle of a curve as a function of its arc length. How should its continuation be predicted?)





Input Parameters

P = # of taps for predictor

N = Window length (for averaging the past values of the path)

LPUP = Update length, ie the number of values predicted with each set of prediction coefficients.

CERR = Cumulative error threshold.

SQERR = Cumulative square error threshold.

Variables

CUMER = Value of the cumulative error for the current predictions

CSQER = Value of the cumulative square error for the current predictions

OLDP = "Old Path," ie, chain code values of the unsuccessfully tracked path. This is a one dimensional array.

NEWP = "New Path," ie, this is a buffer that holds the chain code values of the predicted path. Only upon successful completion of the path are these saved.

ICNT = Counter that marks the number of points from OLDP that are used in NEWP. This also marks the beginning of the current prediction window, ie, the last P + N values of OLDP are used to predict LPUP values for NEWP.

Fig. 2.12 The Linear Predictive Tracking Algorithm
a) The input parameters and the variables.

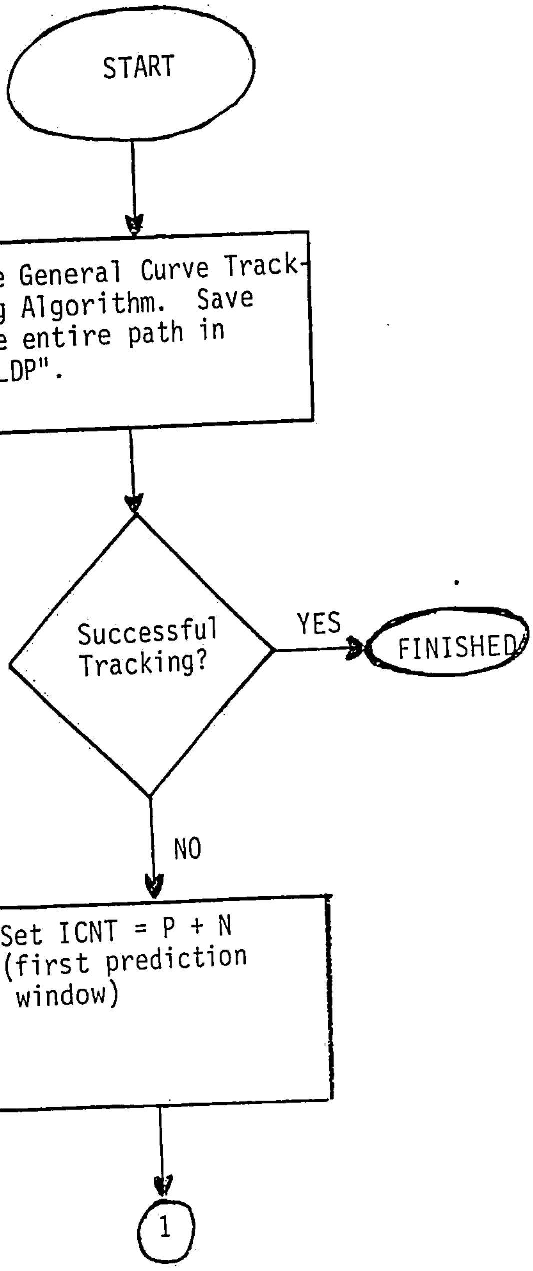
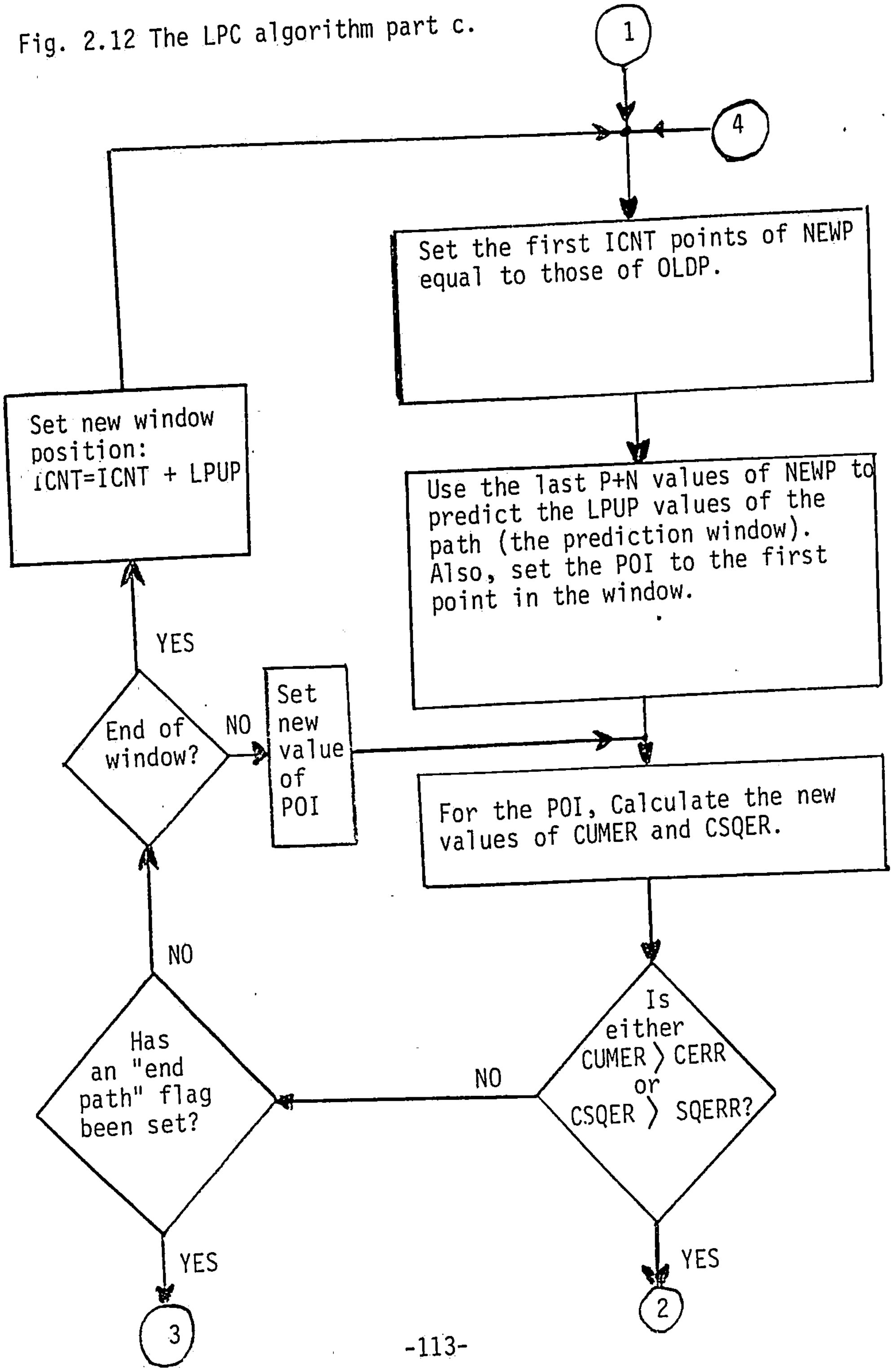

START
Use General Curve Tracking Algorithm. Save the entire path in "OLDP".
Successful Tracking?
YES
FINISHED
NO
Set ICNT = P + N (first prediction window)
1

Fig. 2.12 The LPC algorithm part b.

Fig. 2.12 The LPC algorithm part c.


1
4
Set the first ICNT points of NEWP equal to those of OLDP.
Use the last P+N values of NEWP to predict the LPUP values of the path (the prediction window). Also, set the POI to the first point in the window.
For the POI, Calculate the new values of CUMER and CSQER.
Is either CUMER 〉 CERR or CSQER 〉 SQERR?
YES
2
NO
Has an "end path" flag been set?
YES
3
NO
End of window?
NO
Set new value of POI
YES
Set new window position: ICNT=ICNT + LPUP

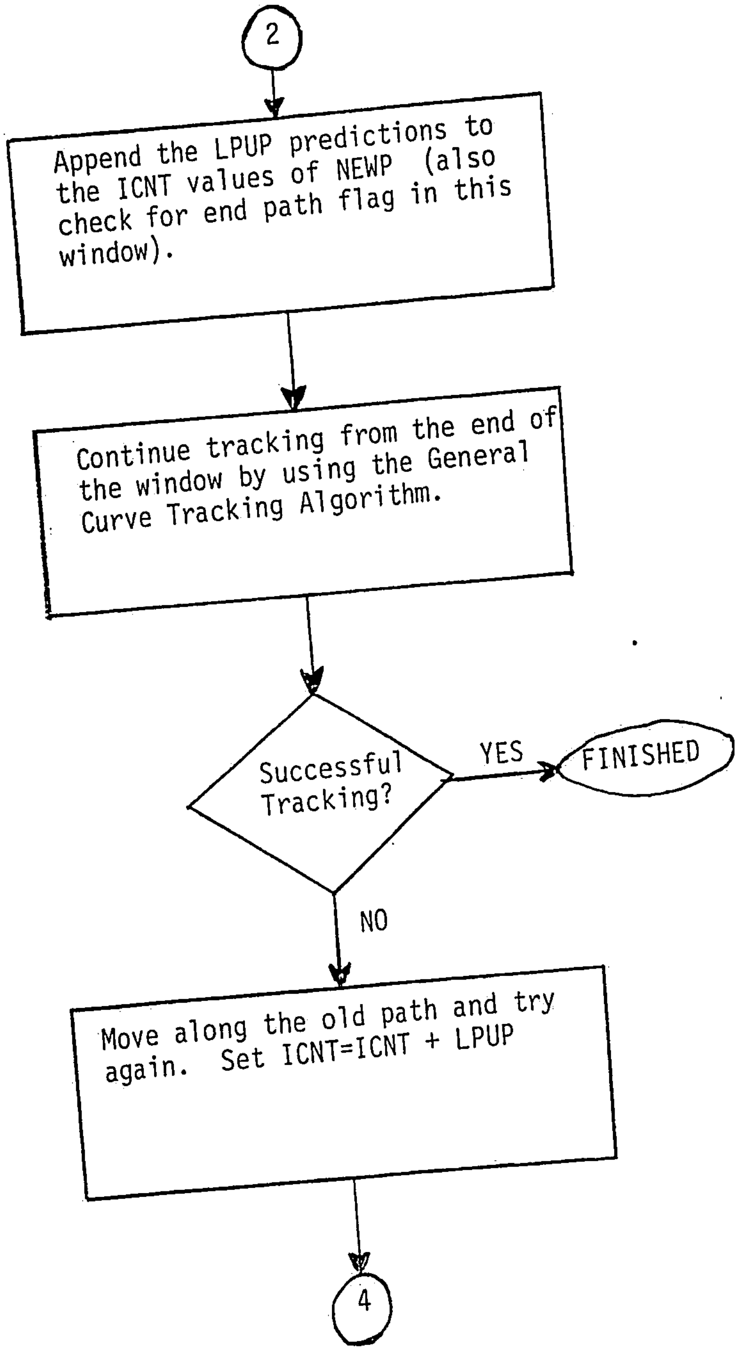

2
Append the LPUP predictions to the ICNT values of NEWP (also check for end path flag in this window).
Continue tracking from the end of the window by using the General Curve Tracking Algorithm.
Successful Tracking?
YES
FINISHED
NO
Move along the old path and try again. Set ICNT=ICNT + LPUP
4

Fig. 2.12 The LPC algorithm part d.

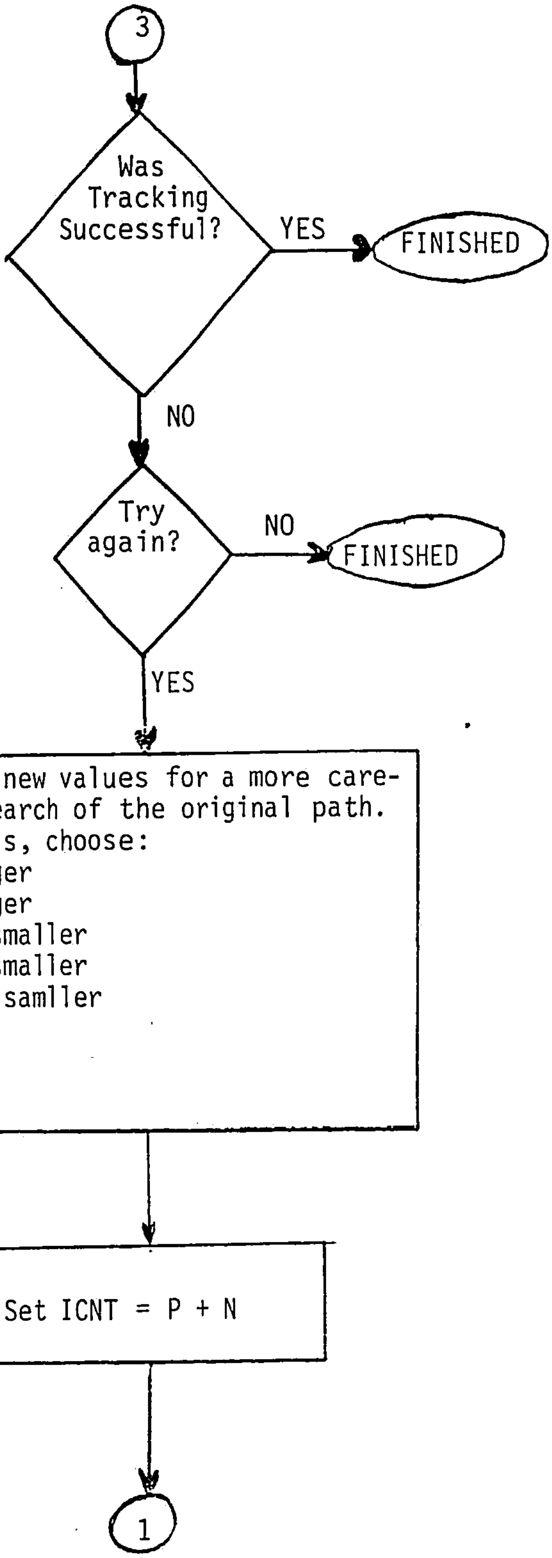


Fig. 2.12 The LPC algorithm part e.

Fig. 2.13 Chain code elements and their opposites.

c.c = chain code element

| c.c | $\Delta x$ | $\Delta y$ | c.c | $\Delta x$ | $\Delta y$ |
|---|---|---|---|---|---|
| 1 | 1 | 0 | 5 | -1 | 0 |
| 2 | 1 | 1 | 6 | -1 | -1 |
| 3 | 0 | 1 | 7 | 0 | -1 |
| 4 | -1 | 1 | 8 | 1 | -1 |

a) The original chain code (i), and the chain code (ii) with elements added randomly for distortion in direction $\pi/4$. Elements with underlines are the insertions.

i) 4,4,5,4,3,3,4,5,5,6,7,6,7,8,1,8,8,1,1,8

ii) 4,4,5,$\underline{2}$,4,3,$\underline{6}$,3,$\underline{6}$,4,5,5,$\underline{2}$,6,7,6,7,$\underline{2}$,8,1,$\underline{6}$,8,8,$\underline{2}$,1,1,$\underline{6}$,8

b) The corresponding patterns.

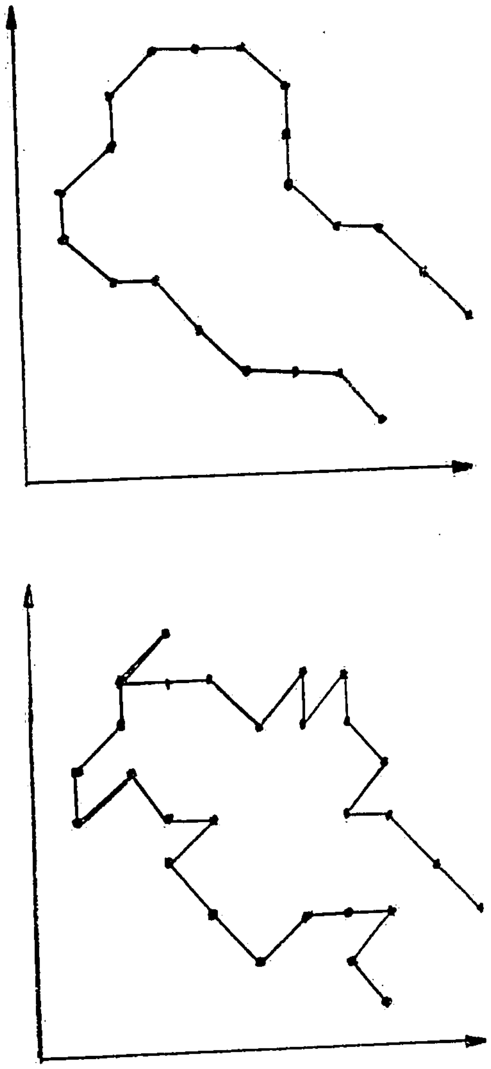


Fig. 2.14 Chain code and random distortion in direction $\pi/4$.

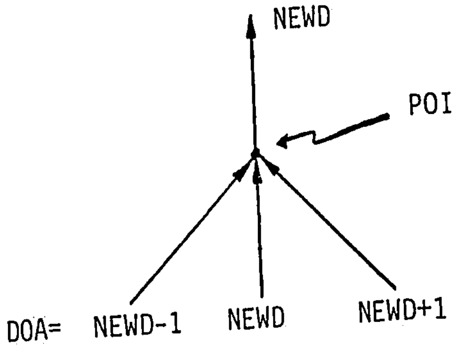
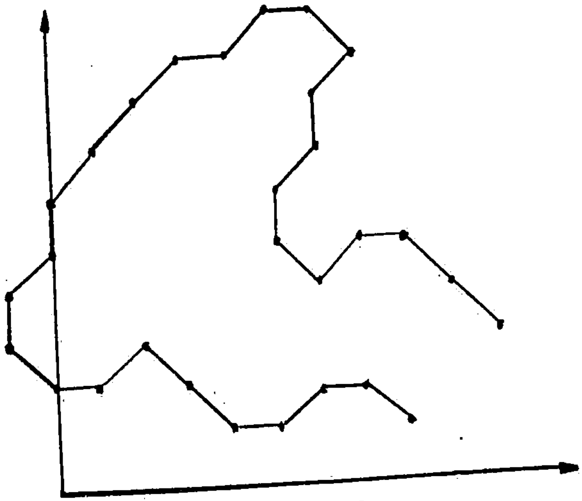


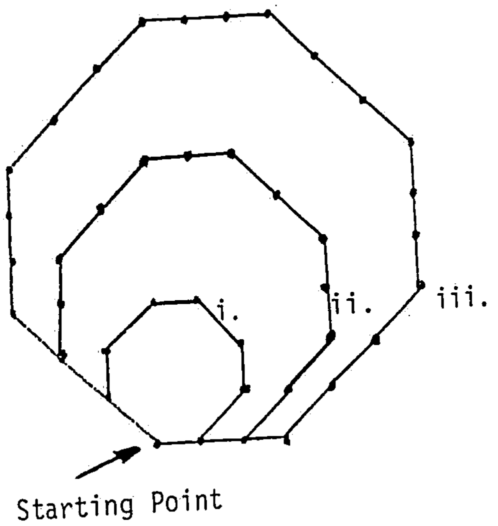
Fig. 2.15a) The Tetrahedron Test.

Fig. 2.15b) Distortion of chain code from Fig. 2.14a using the tetrahedron test to qualify the insertions. Underlined elements are the insertions.

4,4,5,6,4,3,2,3,2,4,5,6,5,6,6,6,7,6,7,8,1,2,8,8,1,2,1,8

Fig. 2.15c) The corresponding pattern with distorrion in direction $\pi/4$.





Chain Codes

i. 1,2,3,4,5,6,7,8
ii. 1,1,2,2,3,3,4,4,5,5,6,6,7,7,8,8
iii.1,1,1,2,2,2,3,3,3,4,4,4,5,5,5,6,6,6,7,7,7,8,8,8

Fig. 2.16a) Principle of circular distortion (omnidirectional).

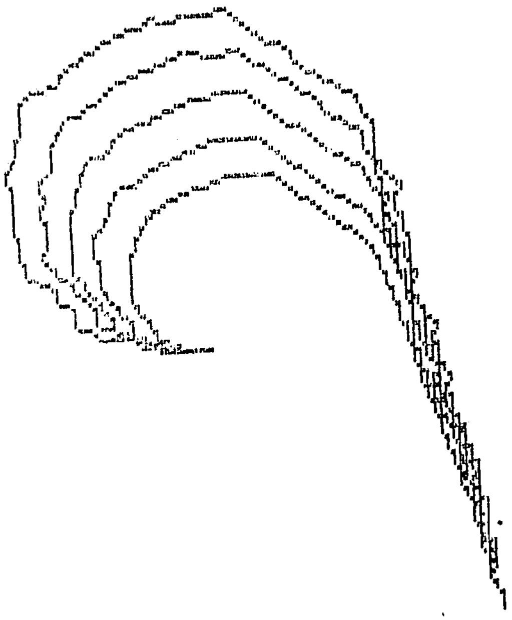


Fig. 2.16b) Circular or omnidirectional distortion. The original curve appears in the center.

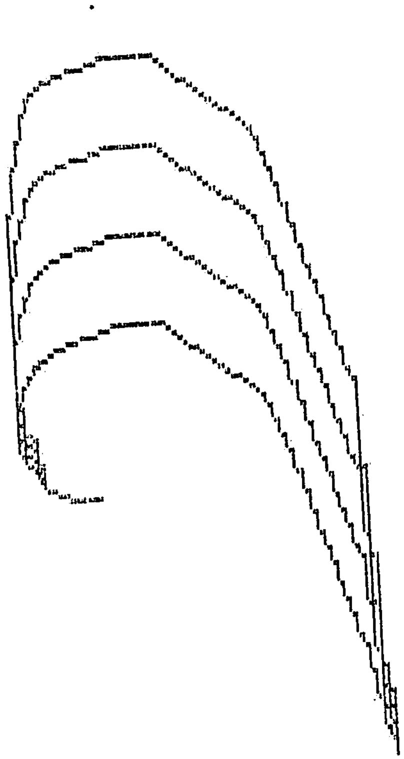


Fig. 2.17 Linear or directional distortion. The original curve appears at the bottom.

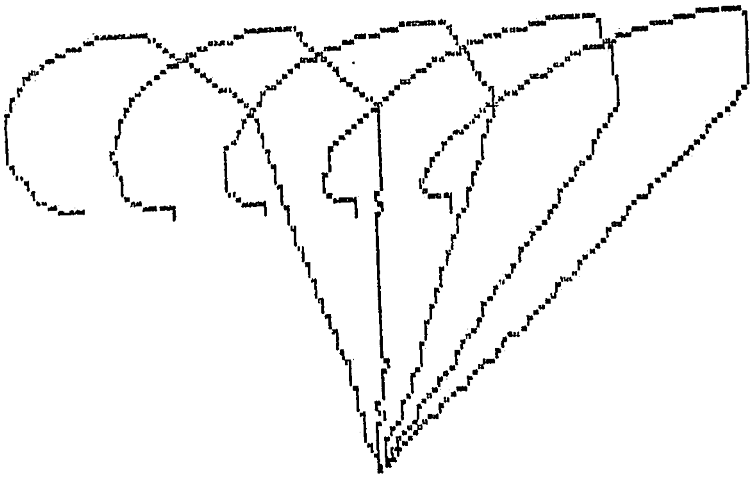


Fig. 2.18 Another type of distortion. The original curve appears at the left.

| Chain Code Element | Absolute Tangent Angle (radians) |
|---|---|
| 1 | $0$ |
| 2 | $\pi/4$ |
| 3 | $\pi/2$ |
| 4 | $3\pi/4$ |
| 5 | $\pi$ |
| 6 | $5\pi/4$ |
| 7 | $3\pi/2$ |
| 8 | $7\pi/4$ |

Fig. 3.1 Tangent angle relations and chain code elements.

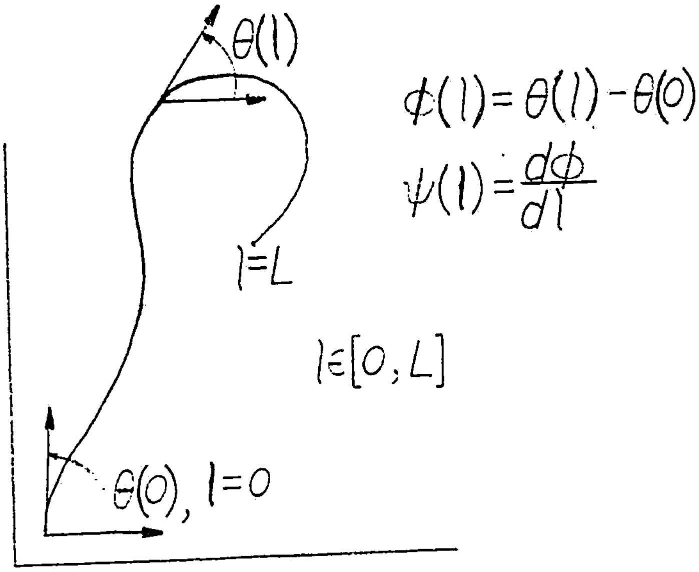


Fig. 3.2a $\theta(l)$=absolute tangent angle, $\phi(l)$=relative tangent angle, and $\psi(l)$=change in tangent angle.

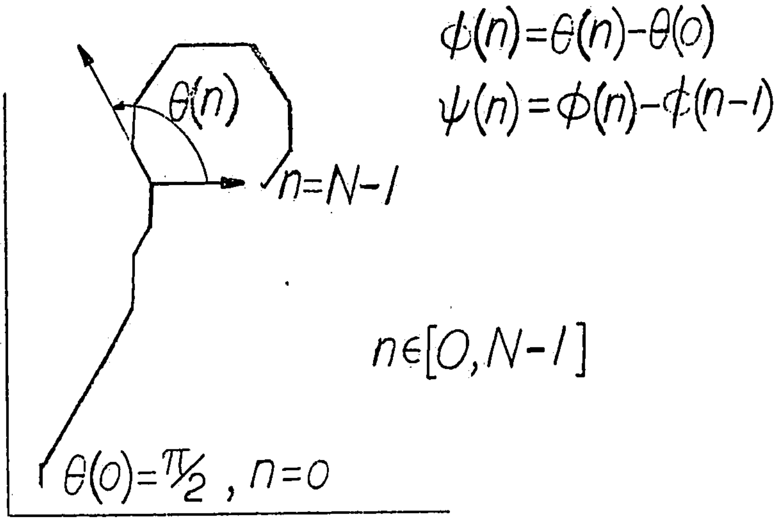
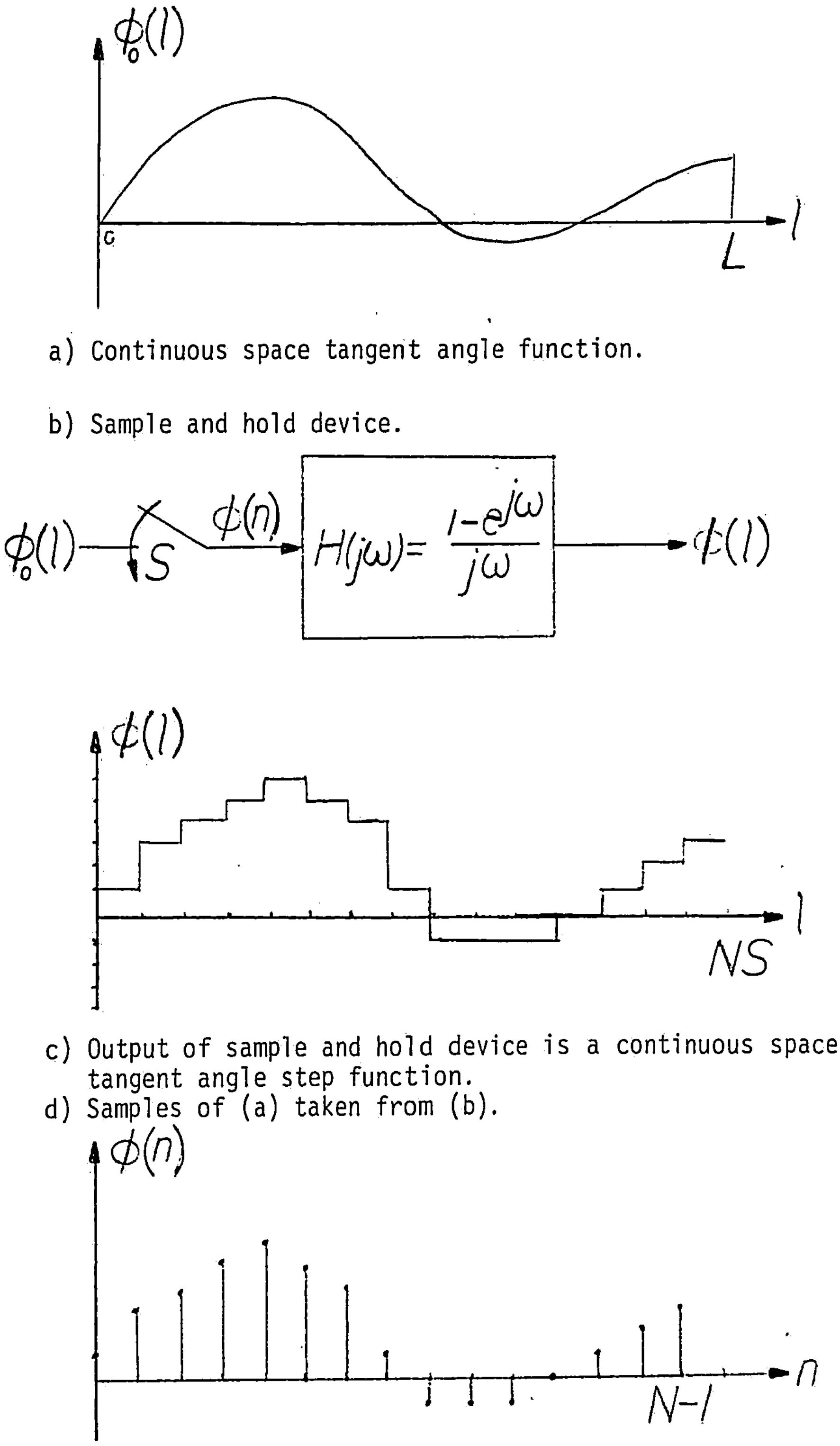

$\phi(n)=\theta(n)-\theta(0)$
$\psi(n)=\phi(n)-\phi(n-1)$
$\theta(n)$
$n=N-1$
$n\in[0,N-1]$
$\theta(0)=\pi/2\ ,\ n=0$

Fig. 3.2b Tangent angle quantization. $\theta(n)$=absolute tangent angle, $\phi(n)$=relative tangent angle,



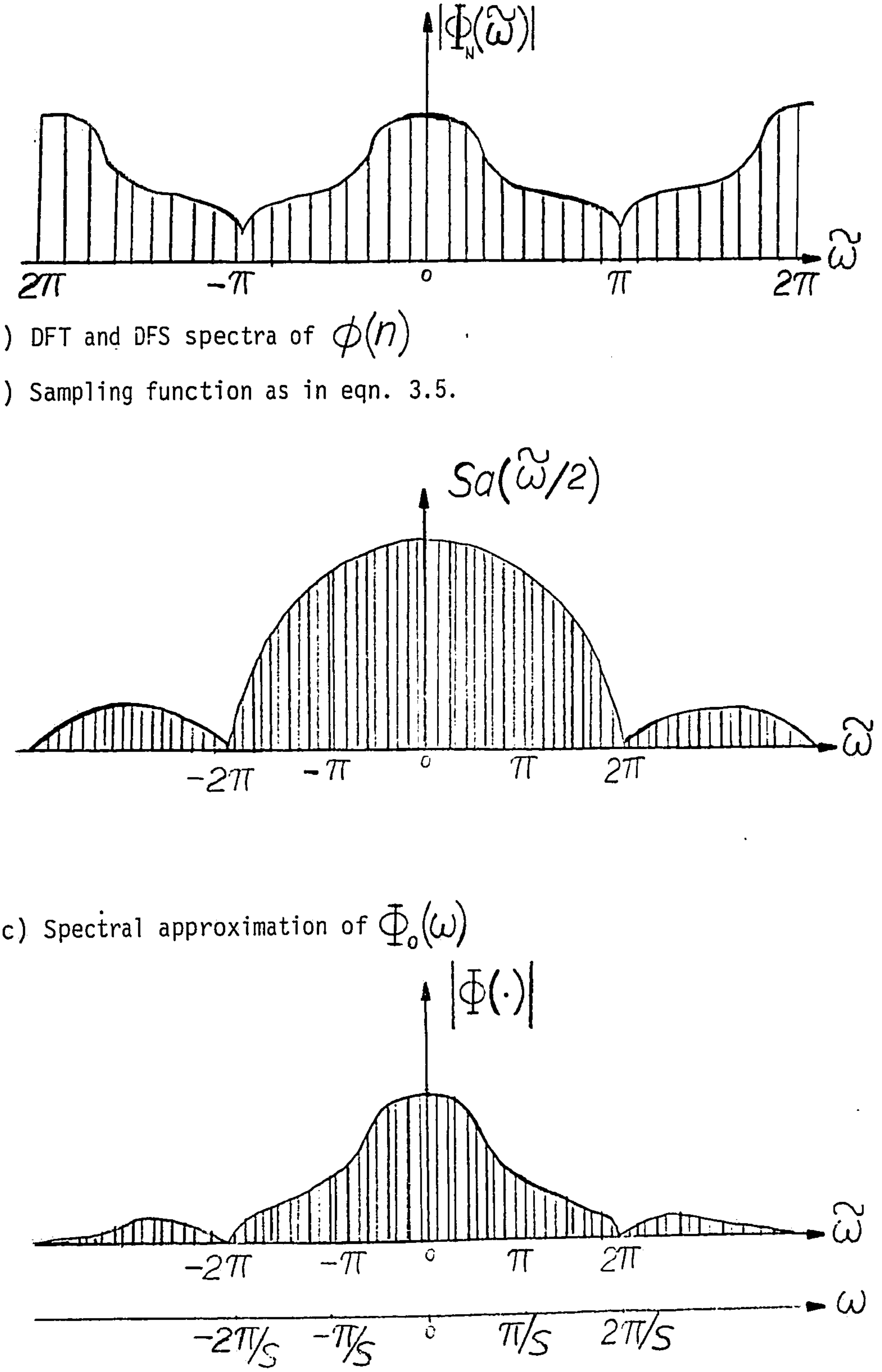
a) Continuous space tangent angle function.

b) Sample and hold device.

c) Output of sample and hold device is a continuous space tangent angle step function.

d) Samples of (a) taken from (b).

Fig. 3.3



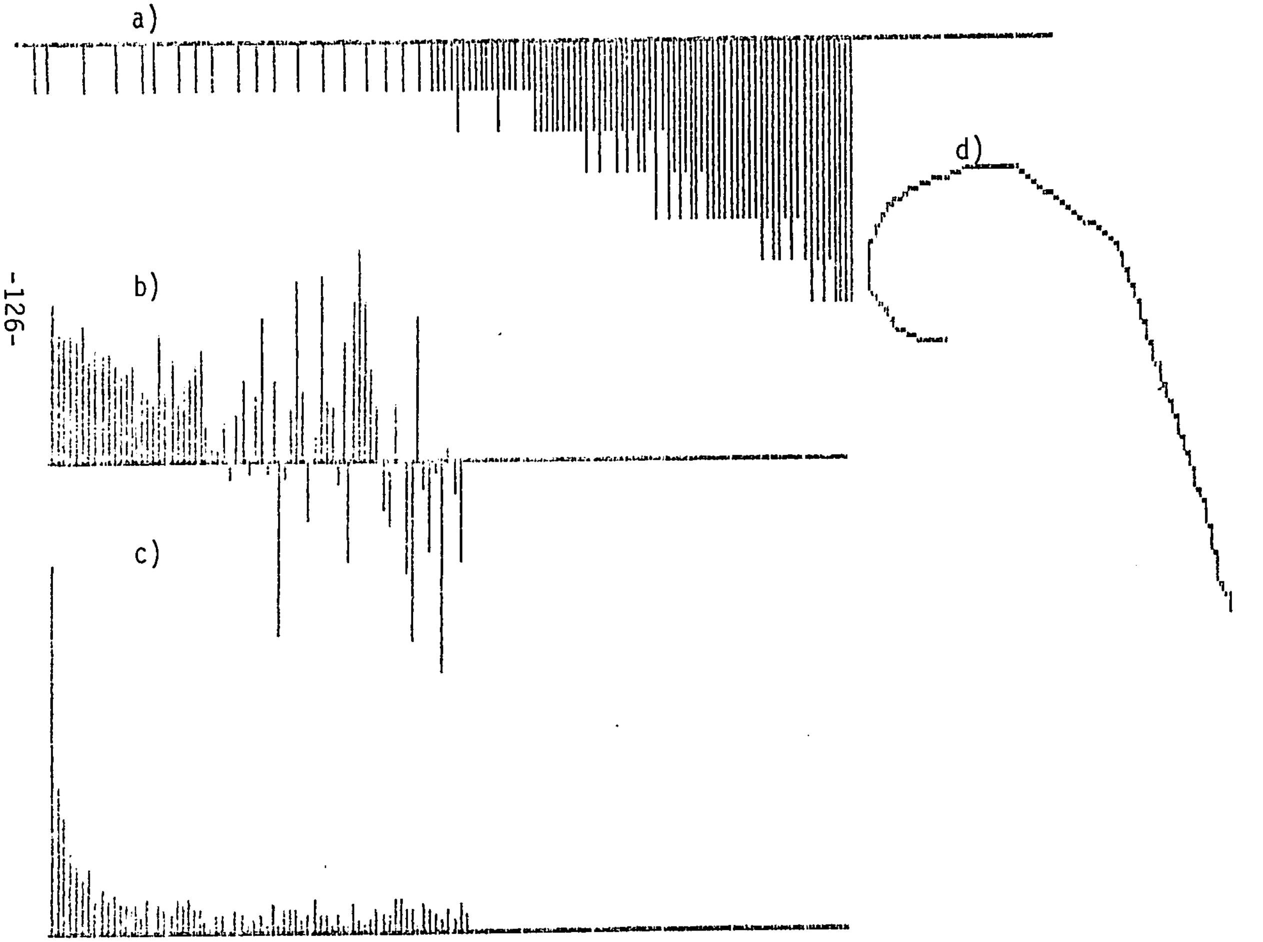
a) DFT and DFS spectra of $\phi(n)$

b) Sampling function as in eqn. 3.5.

c) Spectral approximation of $\Phi_0(\omega)$

Fig. 3.4

Fig. 3.5 A tracked curve a) its relative tangent angle function, 128 values
b) Fourier coefficients (phase), 75 values.
c) Fourier coefficients (magnitude), 75 values.
d) the tracked curve, 129 points.



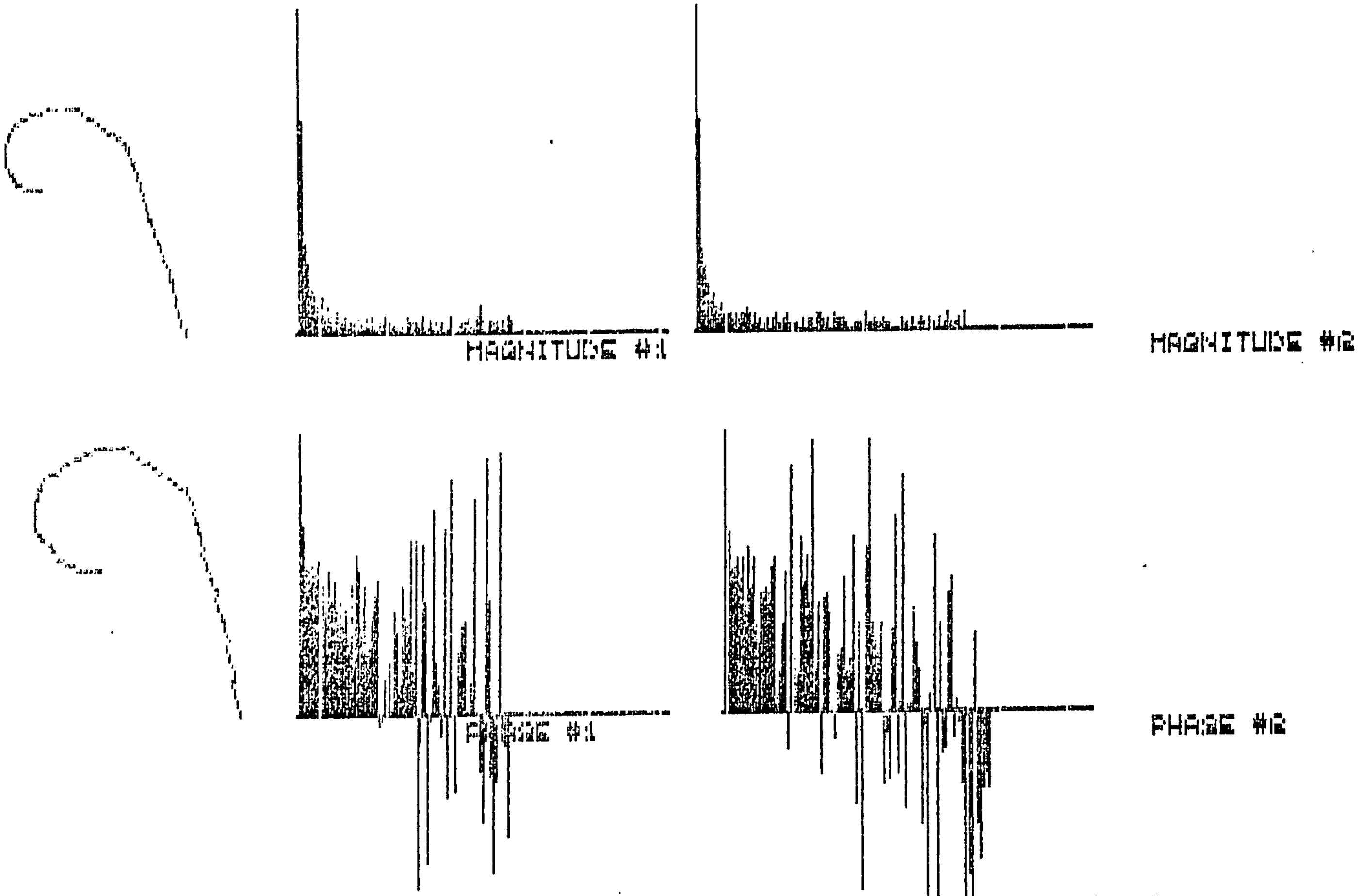


Fig. 3.6 Original curve and distorion (as in sec. 2.4) The ratio of insertions/possible insertions is 6/13, ie. NUMDES/MINMPI.

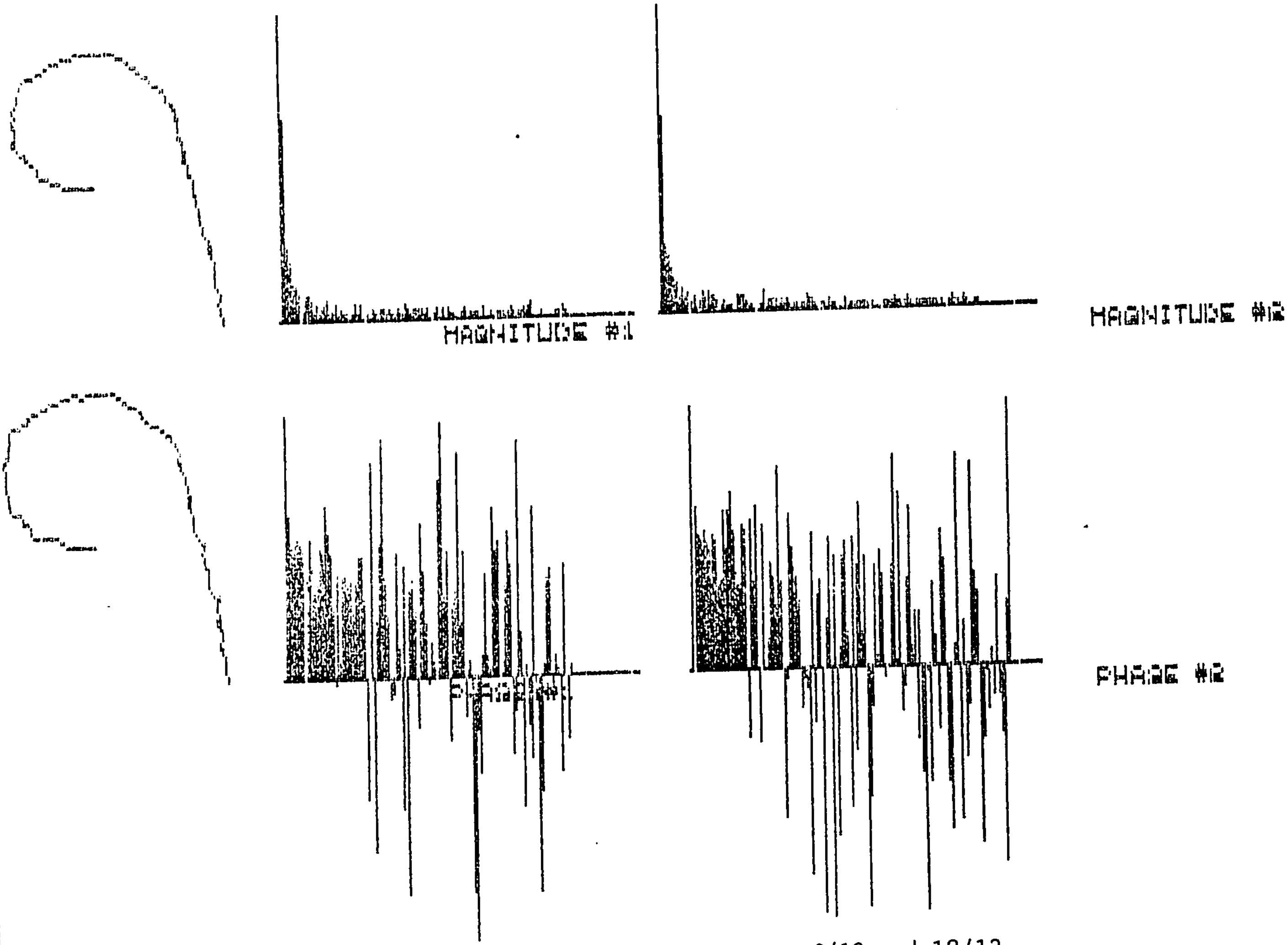


Fig. 3.7 Two further distortions, NUMDES/MINMPI = 9/13 and 12/13.

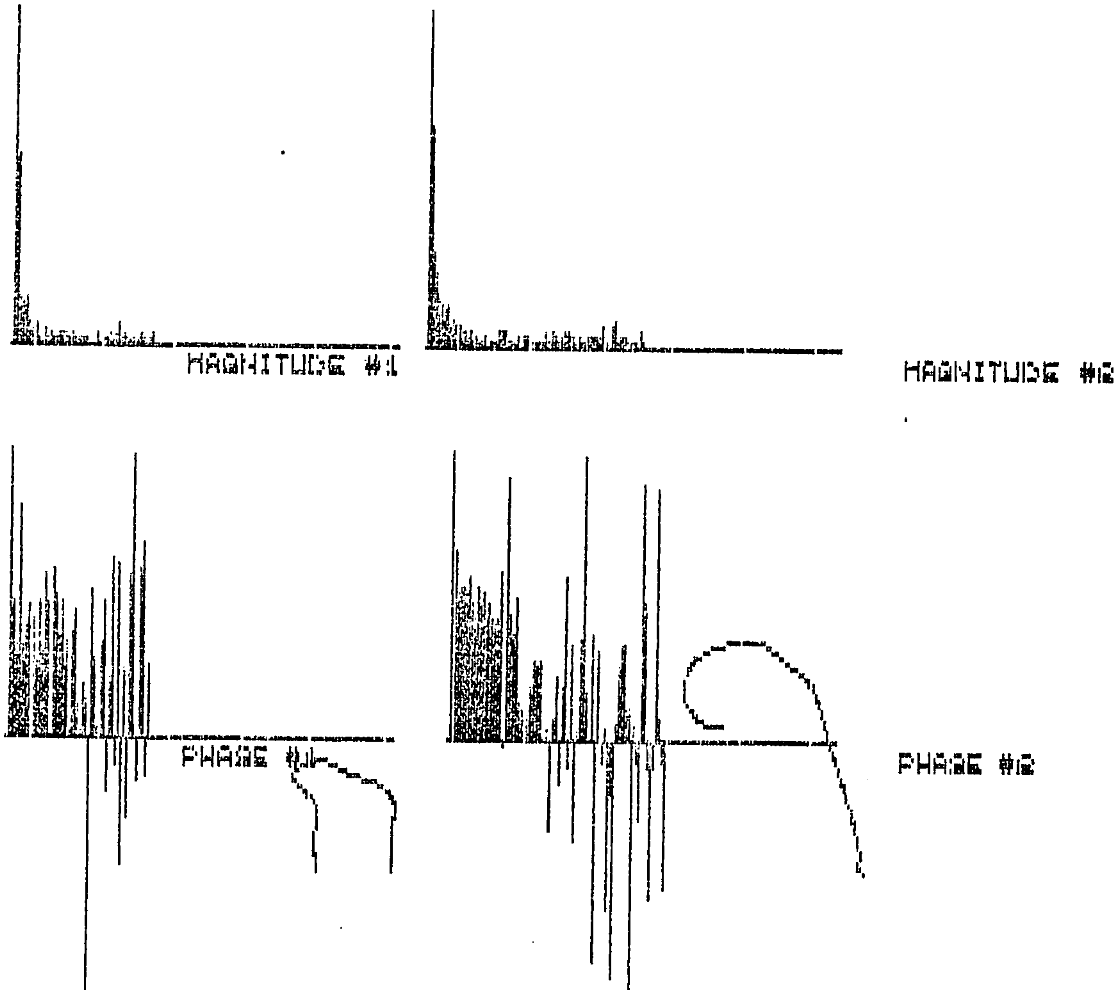


Fig. 3.8 Two dissimilar curves and their spectra.

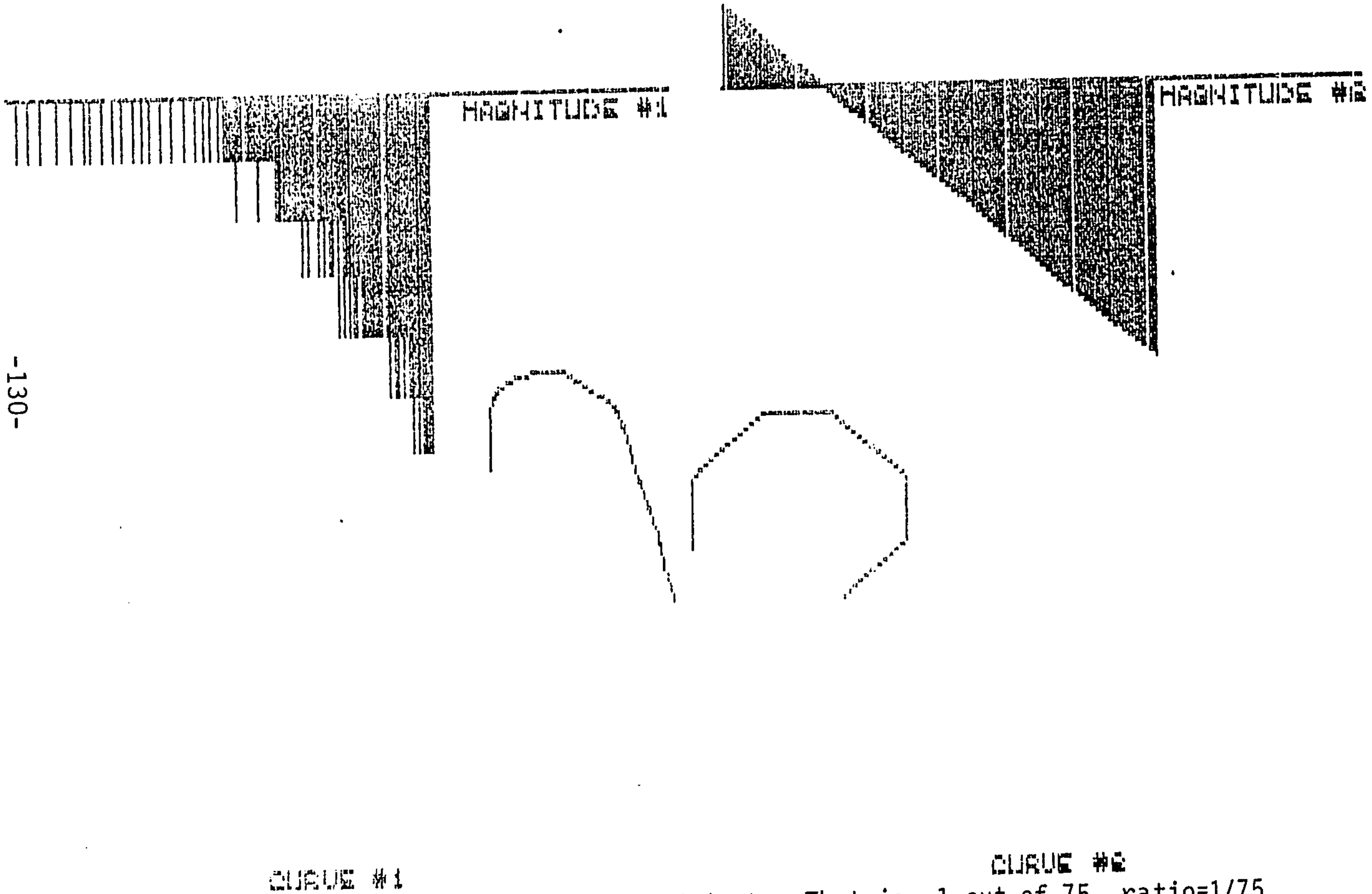


Fig. 3.9 Reconstruction with 2% of coefficients. That is, 1 out of 75, ratio=1/75.

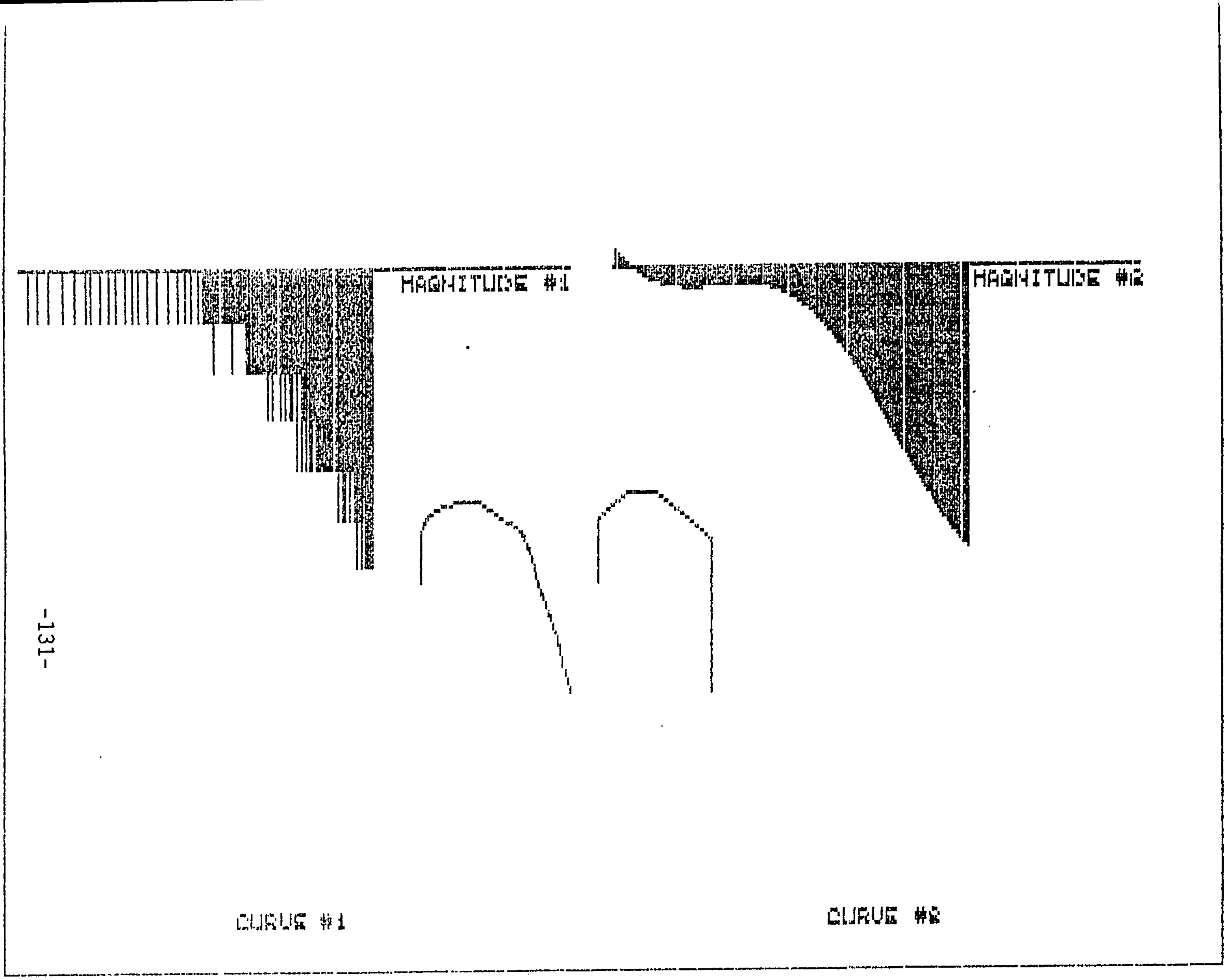


Fig. 3.10 Reconstruction with 3.5%, ratio=2/75.

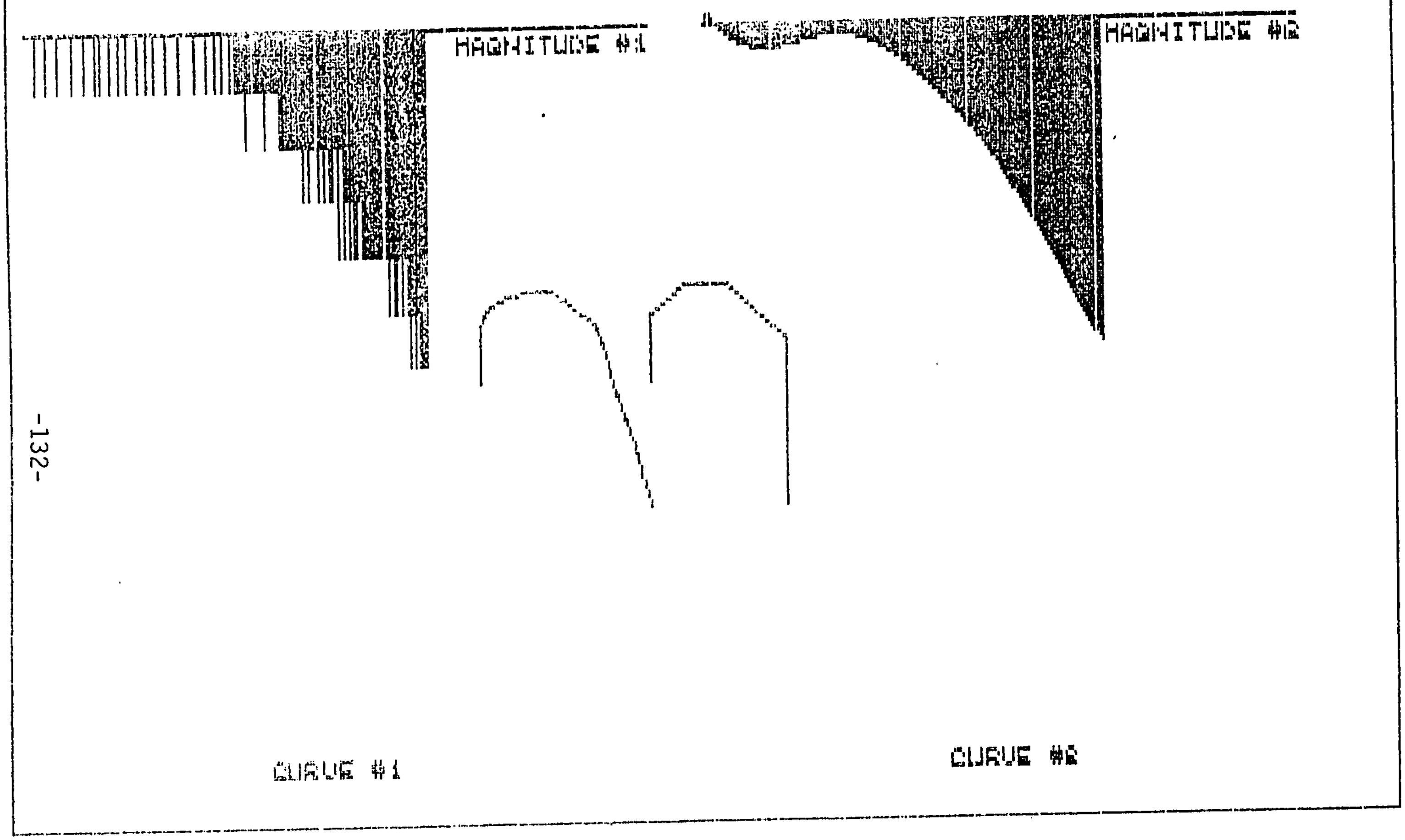


Fig. 3.11 Reconstruction with 5%, ratio=3/75.

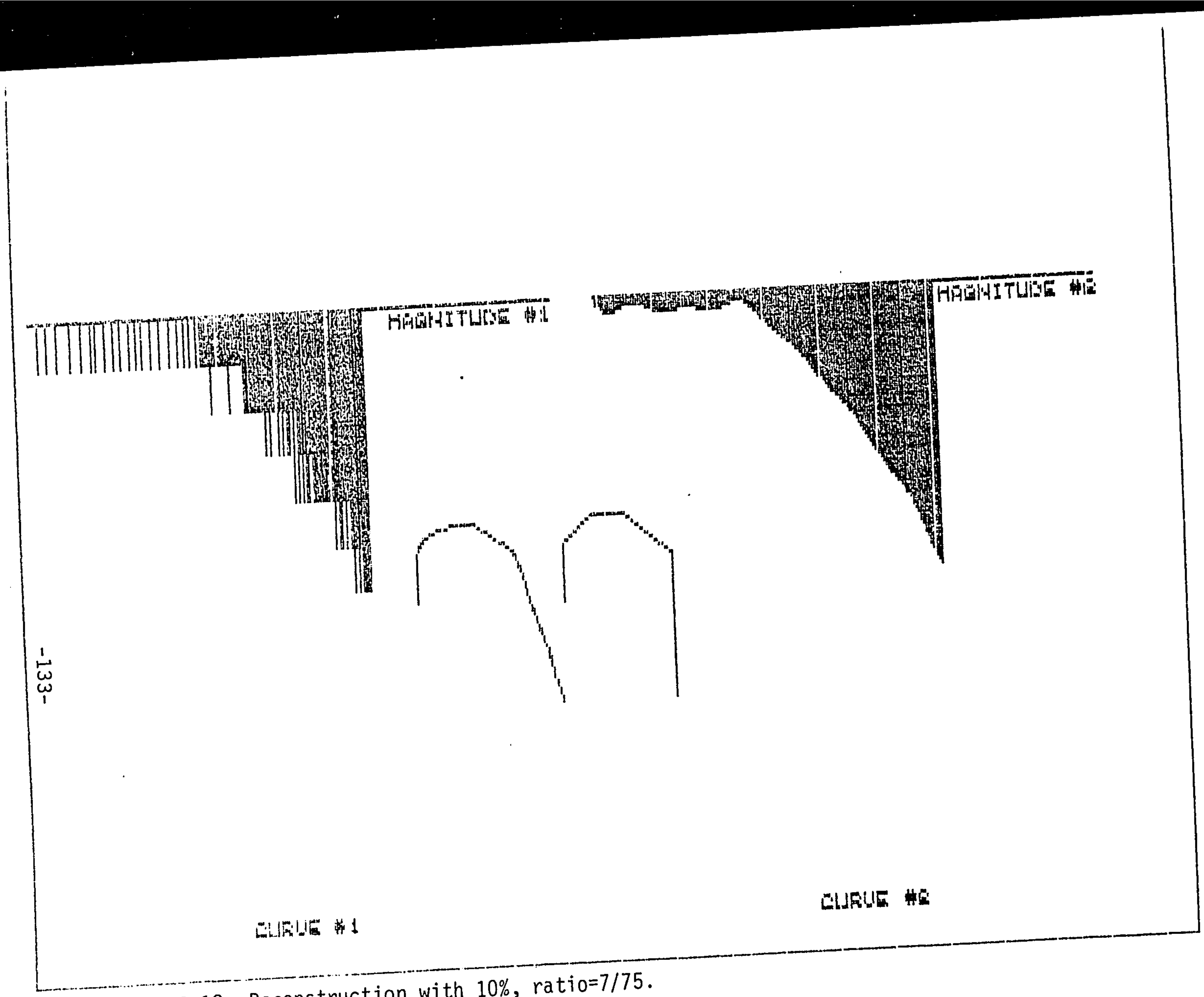


Fig. 3.12  Reconstruction with 10%, ratio=7/75.

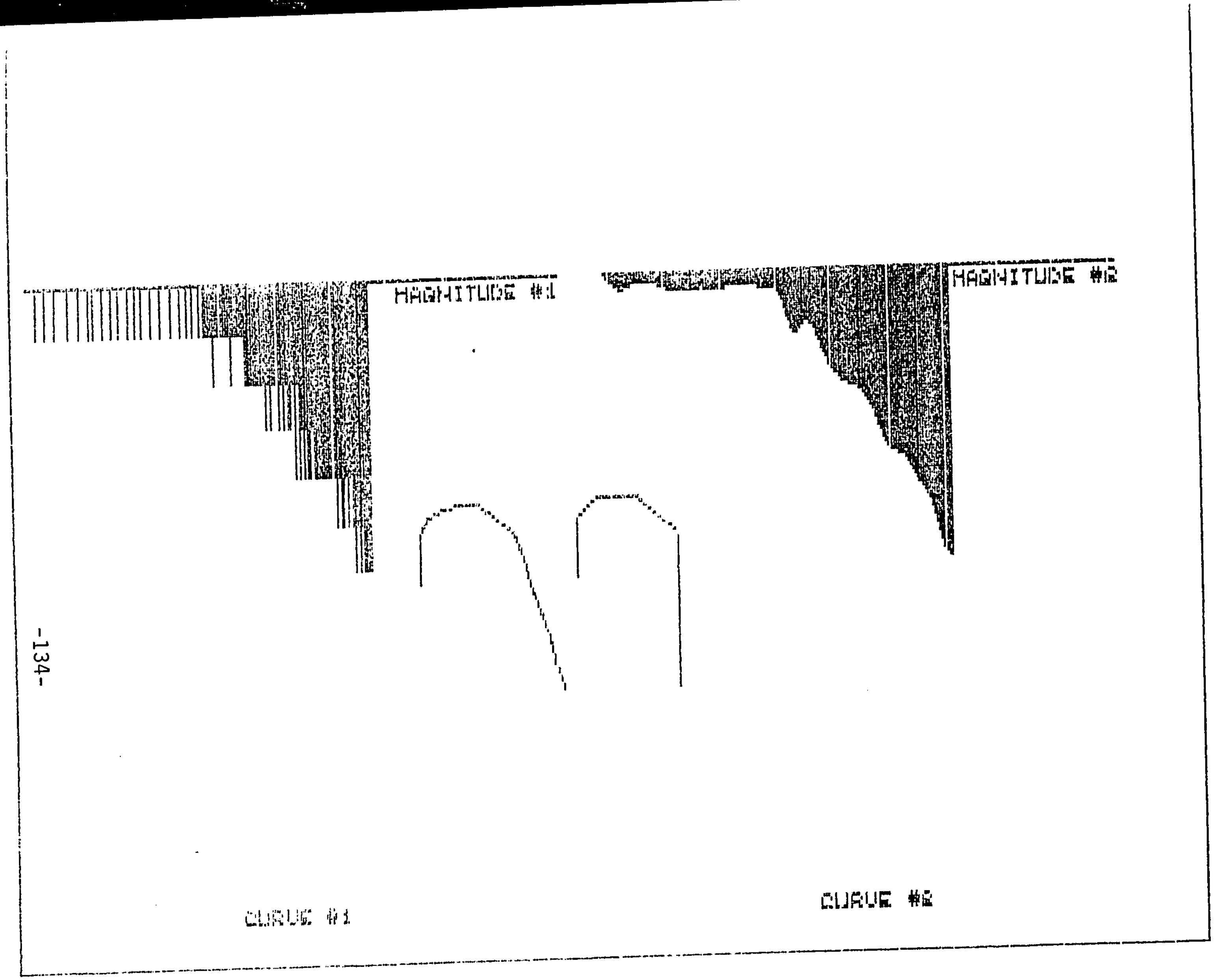


Fig. 3.13 Reconstruction with 20%, ratio=15/75.

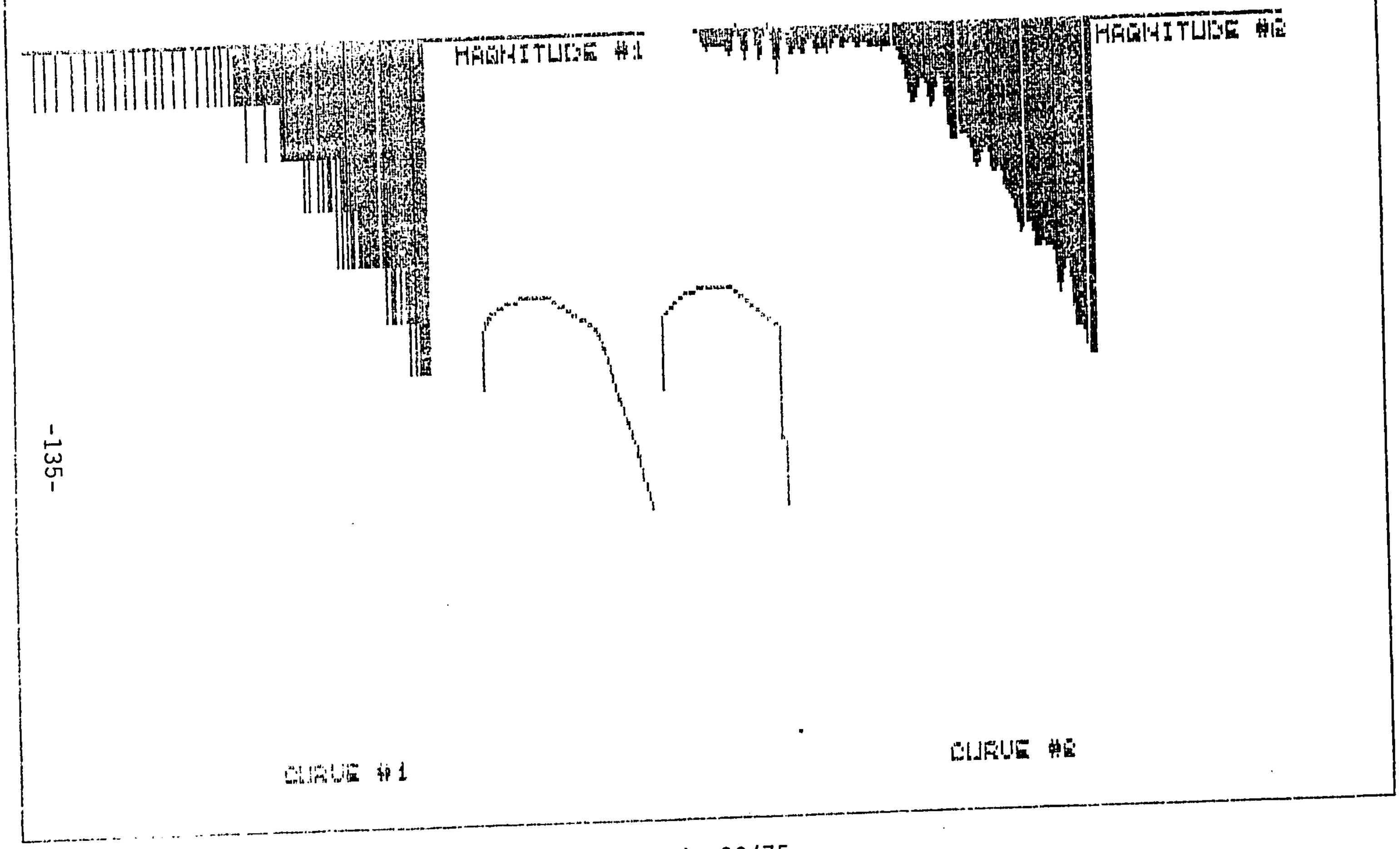


Fig. 3.14 Reconstruction with 40%, ratio=30/75.

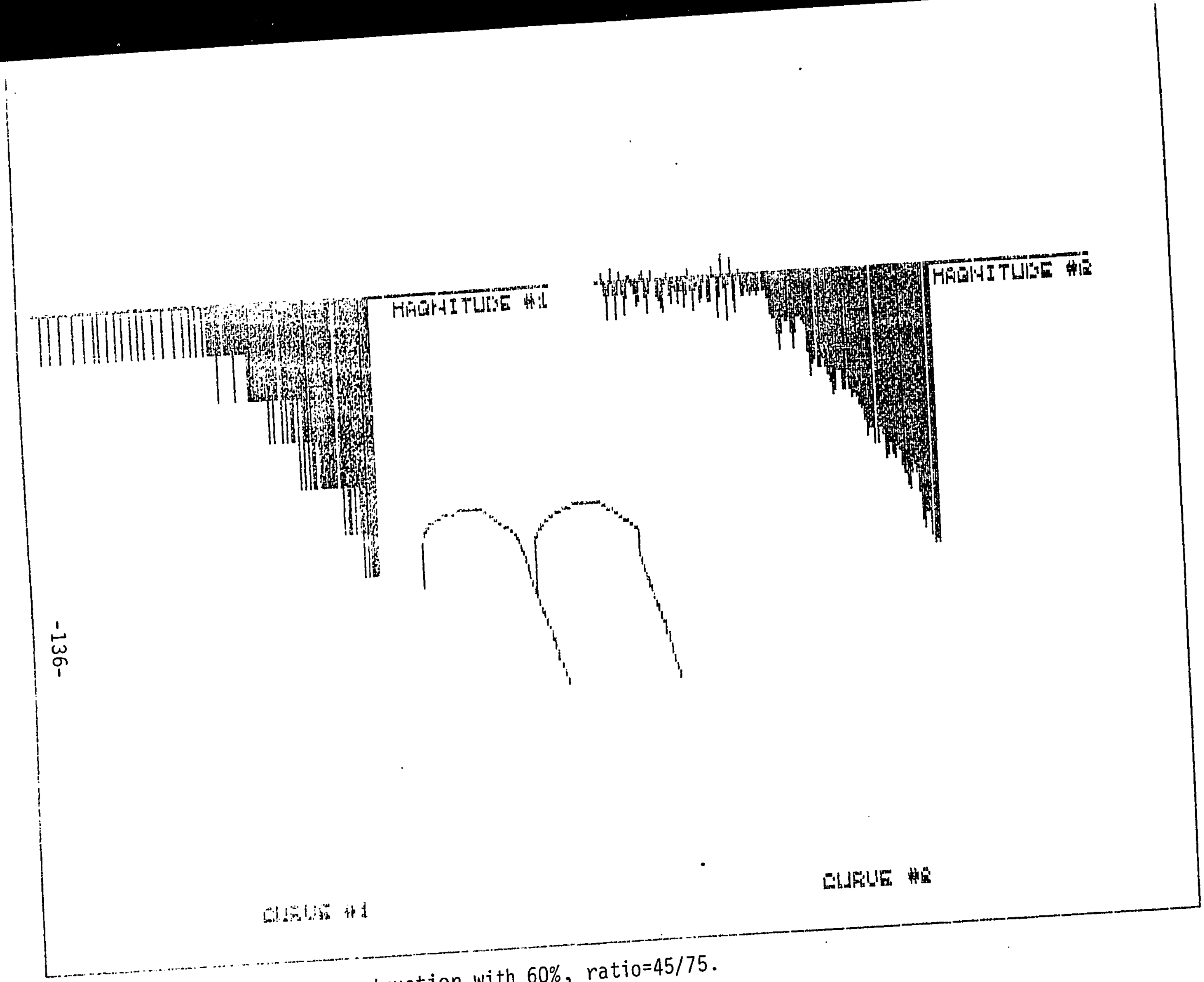


Fig. 3.15  Reconstruction with 60%, ratio=45/75.

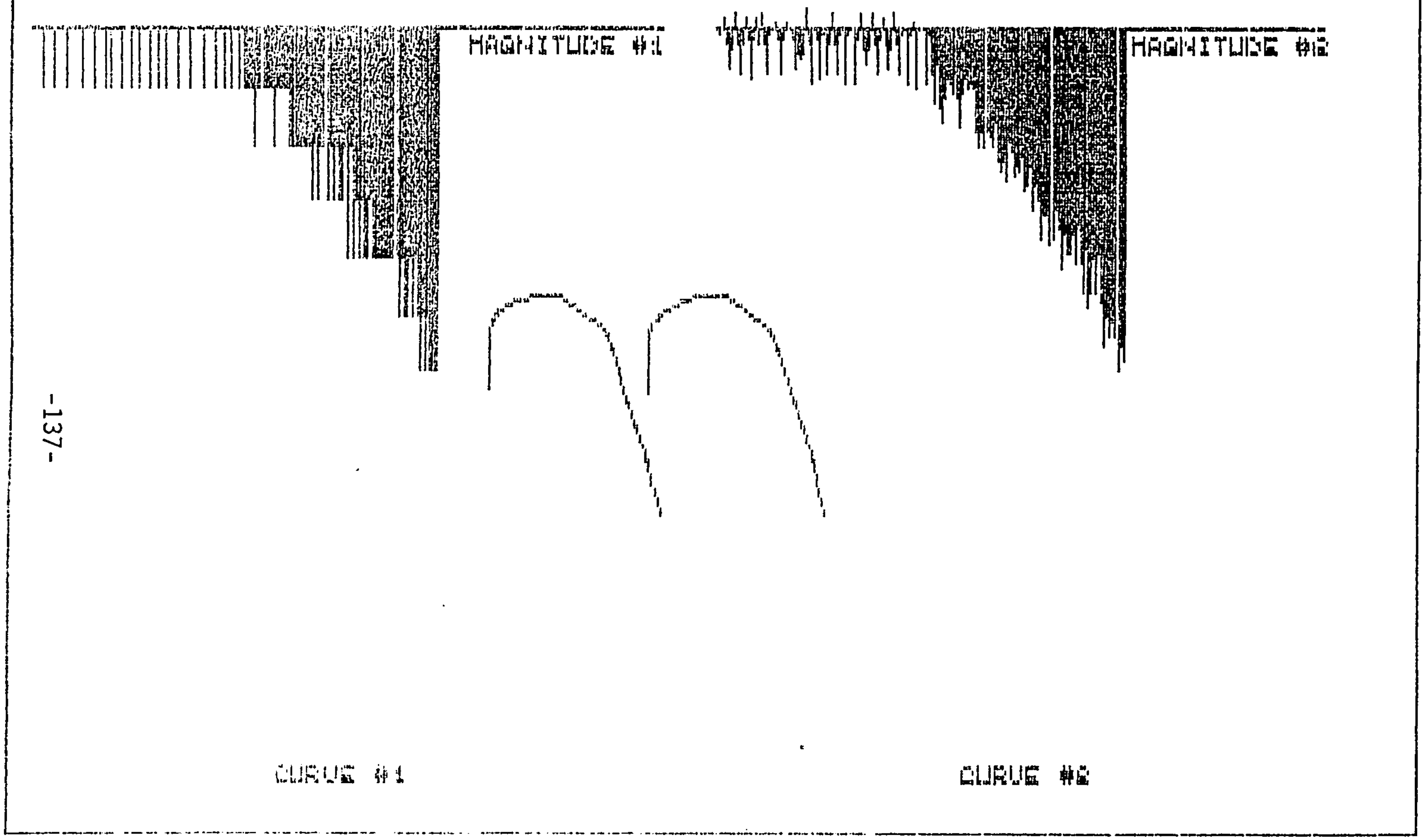


Fig. 3.16  Reconstruction with 80%, ratio=60/75.

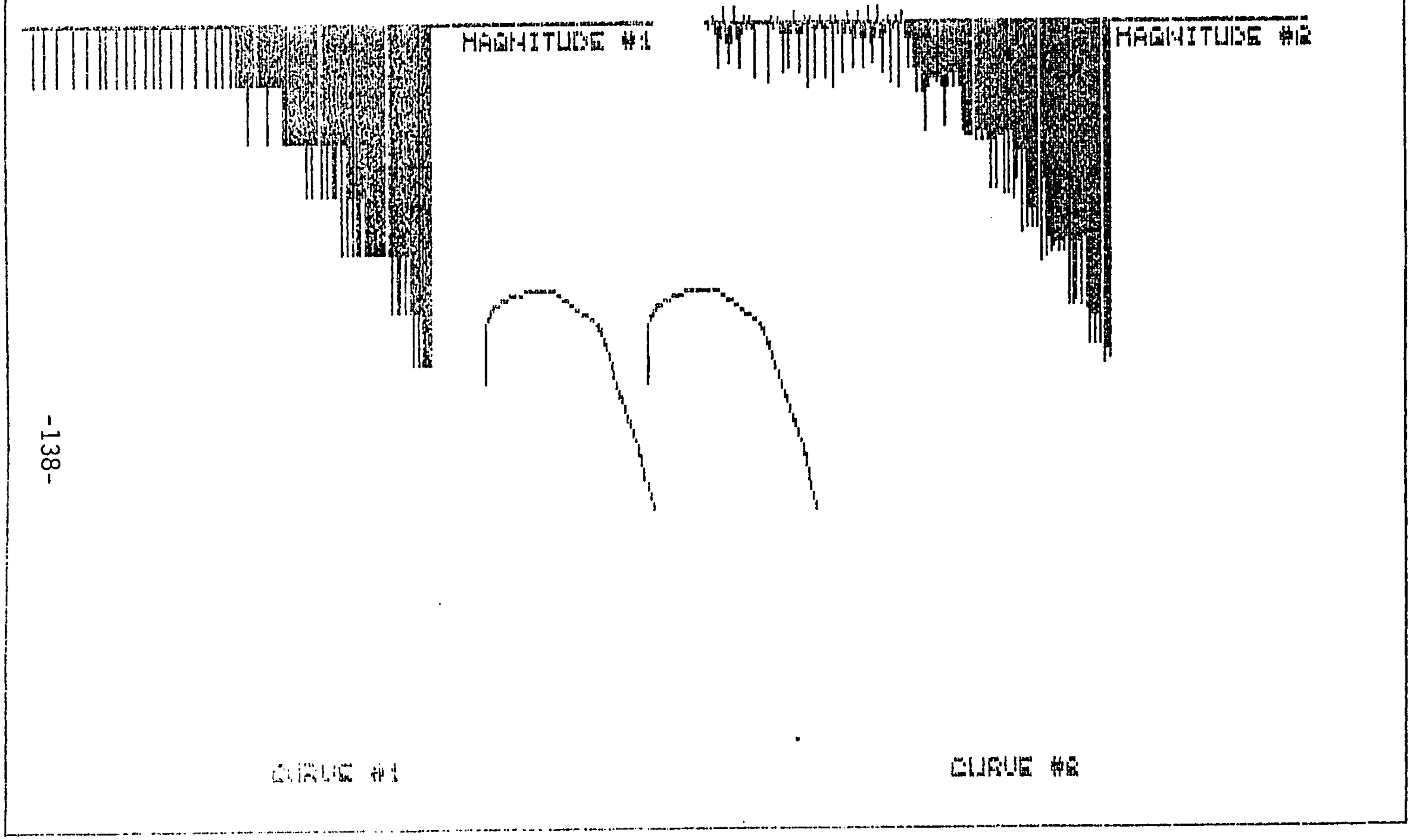


Fig. 3.17 Reconstruction with 90%, ratio=67/75.

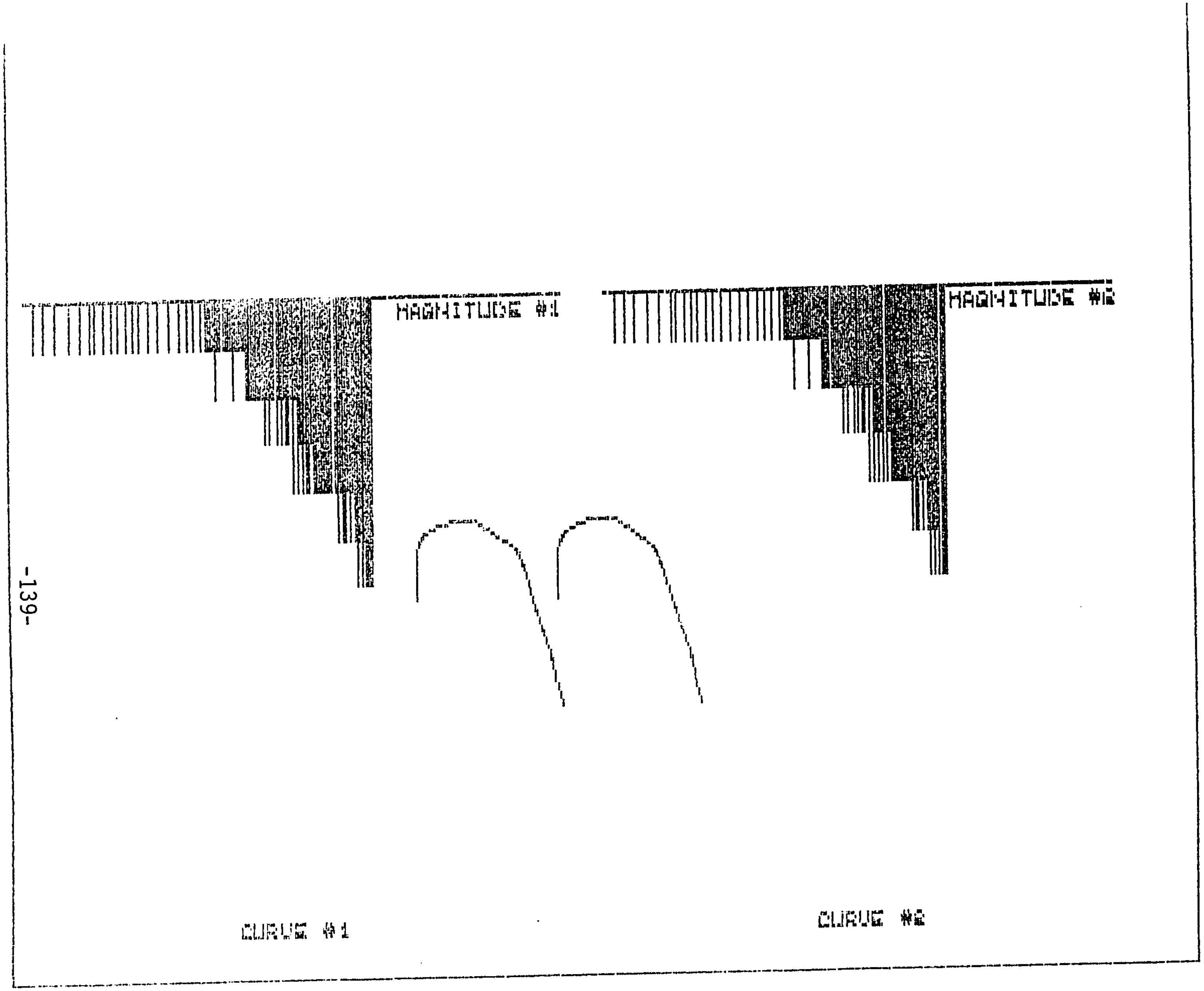


Fig. 3.18 Reconstruction with 100%, ratio=75/75.

| Recon. Ratio | $\frac{\text{NUMDES}}{\text{MINMPI}}$ | $S_M$ | $S_\Phi$ | $S=S_M$ x $S_\Phi$ |
|---|---|---|---|---|
| | | .9___ | .9__ | |
| 10/75 | 2 | 993 | 989 | .9982 |
| | 4 | 985 | 960 | .9945 |
| | 6 | 980 | 888 | .9868 |
| | 8 | 961 | 889 | .9850 |
| | 10 | 950 | 931 | .9263 |
| | 12 | 929 | 929 | .9859 |
| 25/75 | 2 | 947 | 660 | .9609 |
| | 4 | 925 | 190 | .9121 |
| | 6 | 918 | 060 | .8986 |
| | 8 | 912 | 580 | .9496 |
| | 10 | 913 | .8850 | .8773 |
| | 12 | 871 | .8840 | .8726 |
| 40/75 | 2 | 913 | .7961 | .7543 |
| | 4 | 896 | .8129 | .8036 |
| | 6 | 899 | .7940 | .7860 |
| | 8 | 900 | .8710 | .8623 |
| | 10 | 893 | .7820 | .7736 |
| | 12 | 836 | .7660 | .7534 |
| 55/75 | 2 | 874 | .5450 | .5381 |
| | 4 | 860 | .6730 | .6635 |
| | 6 | 867 | .6170 | .6088 |
| | 8 | 850 | .5720 | .5634 |
| | 10 | 833 | .5730 | .5634 |
| | 12 | 806 | .5390 | .5285 |
| 75/75 | 2 | 830 | .4200 | .4129 |
| | 4 | 793 | .5630 | .5513 |
| | 6 | 812 | .4730 | .4641 |
| | 8 | 799 | .3620 | .3547 |
| | 10 | 761 | .4590 | .4480 |
| | 12 | 743 | .3740 | .3643 |

Fig. 3.19 Similarity measures for various distortion levels (ie, NUMDES/MINMPI) and various reconstruction ratios.

# References

## Signal Processing

1. B.D. Fritchman, Discrete and Continuous Analysis of Signals and Systems, (to be published).

2. B.P. Lathi, Signals, Systems and Communications, J. Wiley and Sons, Inc., New York, 1965.

3. Oppenheim and Schafer, Digital Signal Processing, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1975.

4. L.R. Rabiner and B. Gold, Theory and Application of Digital Signal Processing, John Wiley and Sons, New York, 1976.

5. R.E. Crochiere and L.R. Rabiner, "Interpolation and Decimation of Digital Signals - a Tutorial Review", Proceedings of the IEEE, Vol. 69, No. 3, March 1981, pp. 300-331.

6. B.C. Kuo, Digital Control Systems, Holt, Rinehart, and Winston, Inc., 1980.

## Discrete Curves and Image Processing

7. A. Rosenfeld and A.C. Kak, Digital Picture Processing, Academic Press, 1st edition, 1976, (2nd edition is in 2 volumes, 1982).

8. T. Pavlidis, Algorithms for Graphics and Image Processing, Rockville, MD.: Computer Science Press, 1982.

9. W.K. Pratt, Digital Image Processing, New York: Wiley, 1978.

10. A. Rosenfeld, "Image Pattern Recognition", Proceedings of the IEEE, May 1981.

11. H. Freeman, "On the Encoding of Arbitrary Geometric Configurations," IRE Transactions on Electronic Computers. Vol. EC-10, pp. 260-268, June 1961.

12. H. Freeman, "On the Digital Computer Classification of Geometric Line Patterns", Proceedings of the National Electronics Conference, Vol. 18, 1962, pp. 312-324.

13. H. Freeman, "Techniques for the Digital Computer Analysis of Chain Encoded Arbitrary Plane Curves", Proceedings of the National Electronics Conference, Vol. 17, 1961, pp. 421-432.

### LPC Coding

14. L.R. Rabiner and R.W. Schafer, *Digital Processing of Speech Signals*, Prentice Hall, Inc., Englewood Cliffs, New Jersey, 1978.

15. P.A. Maragos, R.W. Schafer, and R.M. Mersereau, "Two-Dimensional Linear Prediction and Its Application to Adaptive Predictive Coding of Images", IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 22, pp. 403-415, 1974.

16. S. Chandra and W.C. Lin, "Experimental Comparison Between Stationary and Non-Stationary Formulations of Linear Prediction Applied to Speech," IEEE Transactions on Acoustics, Speech, and Signal Processing, Vol. 22, pp. 403-415, 1974.

### Fourier Descriptors

17. R.L. Cosgriff, "Identification of Shape", Ohio State Univ. Research Foundation, Columbus, Rep 820-11, ASTIA AD 254 792, Dec. 1960.

18. G.H. Granlund, "Fourier Processing for Hand Print Character Recognition", IEEE Transactions on Computers, Vol. C-21, pp. 195-201, Feb. 1972.

19. T.P. Wallace and O.R. Mitchell, "Analysis of Three-Dimensional Movement Using Fourier Descriptors", IEEE Transactions on Pattern Analysis and Machine Intelligence", Vol. PAMI-2, No. 6, Nov. 1980.

20. C.T. Zahn and R.Z. Roskies, "Fourier Descriptors for Plane Closed Curves", IEEE Transactions on Computers, Vol. C-21, No. 3, March 1972.

21. J.R. Bennett and J.S. MacDonald, "On the Measurement of Curvature in a Quantized Environment," IEEE Transactions on Computers, August 1975.

22. T.R. Crimmins, "A Complete Set of Fourier Descriptors for Two-Dimensional Shapes", IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-12, No. 6, Nov./Dec. 1982.

23. C.W. Richards, Jr., and H. Hemami, "Identification of Three-Dimensional Objects Using Fourier Descriptors of the Boundary Curve", IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-4, No. 4, July 1974.

24. E. Persoon and K.-S. Fu, "Shape Discrimination Using Fourier Descriptors", IEEE Transactions on Systems, Man, and Cybernetics, Vol. SMC-7, No. 3, March 1977.

25. S.Ziada and D. Rockwell, "Vortex-Leading-Edge Interaction", Journal of Fluid Mechanics, Vol. 118, pp. 79-107, Great Britain, 1982.

## Appendix A: Interpolation Formula for Reconstruction of Signals

Here, we discuss the Nyquist criterion and the conditions under which a continuous time signal may be exactly reconstructed from its samples.

Consider the Fourier transform of a bandlimited signal f(t), where the bandlimit is $\omega=W$,

$$f(t) = \frac{1}{2\pi}\int_{-\infty}^{\infty} F(\omega)e^{j\omega t}\, d\omega$$

$$= \frac{1}{2\pi}\int_{-W}^{W} F(\omega)e^{j\omega t} d\omega \tag{A.1}$$

By making a simple change of variable, i.e. $\tilde{\omega}=\omega S$ and $d\omega = d\tilde{\omega}/S$, eqn. A.1 can be written as,

$$f(t) = \frac{1}{2\pi S}\int_{-WS}^{WS} F(\tilde{\omega}/S)e^{j\tilde{\omega}t/s}\, d\tilde{\omega} \tag{A.2}$$

Recalling that $W=2\pi/S_N$, where $S_N$ is the Nyquist sampling rate, then $WS=2\pi S/S_N$. Denoting $2S/S_N$ by $\alpha$, we have

$$f(t) = \frac{1}{2\pi S}\int_{-\pi\alpha}^{\pi\alpha} F(\tilde{\omega}/S)e^{j\tilde{\omega}t/s}\, d\tilde{\omega} \tag{A.3}$$

Equation A.3 is simply a mathematically manipulated version of eqn. A.1, and it is still an integral of an aperiodic spectral density times an eternal exponential; that is, it is still a Fourier transform.

Suppose, now, that f(t) is sampled with period S to generate a sequence g(n) given by

$$g(n) = f(nS) = f(t)\Big|_{t=nS} \tag{A.4}$$

for integer n. Then, the discrete-time FT is given by,

$$G(\tilde{\omega}) = \sum_{n=-\infty}^{\infty} g(n)e^{-j\tilde{\omega}n} \tag{A.5}$$

We recall that the spectral density of eqn. A.5, $G(\tilde{\omega})$, can be compared with the spectral density of eqn. A.1, $F(\omega)$, by using the relation $\omega=\tilde{\omega}/S$. Hence, we define a function h(t) by an equation similar to eqn. A.3.

$$h(t) \overset{\Delta}{=} \frac{1}{2\pi S}\int_{-\pi\alpha}^{\pi\alpha} G(\tilde{\omega})e^{j\tilde{\omega}t/s}\, d\tilde{\omega} \tag{A.6}$$

Using equation A.5, interchanging the order of integration and summation, and then using eqn. A.4, we rewrite eqn. A.6 as,

$$h(t) = \frac{1}{2\pi S}\int_{-\pi\alpha}^{\pi\alpha} [\sum_{n=-\infty}^{\infty} g(n)e^{-j\tilde{\omega}n}]e^{j\tilde{\omega}t/s}\, d\tilde{\omega}$$

$$= 1/2\pi\alpha S_N \sum_{n=-\infty}^{\infty} g(n)\int_{-\pi\alpha}^{\pi\alpha} e^{j\tilde{\omega}(t/s-n)}\, d\tilde{\omega}$$

$$= \frac{1}{S_N} \sum_{n=-\infty}^{\infty} f(nS)[(\frac{1}{2\pi\alpha})(\frac{1}{j(t/s-n)})(e^{j\pi\alpha(t/s-n)} - e^{-j\pi\alpha(t/s-n)})]$$

Finally, we recognize that this can be written as,

$$h(t) = \frac{1}{S_N} \Sigma^{\infty}_{n=-\infty} f(nS)[Sa(\pi\alpha(t/s-n))] \qquad (A.7)$$

This is a complicated expression, but we recognize it as the well known sampling theorem interpolation formula [1,2,3,6], where the Nyquist criterion is satisfied when $\alpha \leq 1$. More precisely, it is also a convolution of the samples of f(t) and a sampling function, and hence the convolution in the time domain corresponds to a multiplication in the frequency domain. In fact, since the FT of a sampling function in the time domain is a gate function in the frequency domain, eqn. A.7 is equivalent to a low pass filtering operation on the spectrum of g(n)=f(nS). It is straightforward to show that

$$G(\omega) = \frac{1}{S} \sum_{n=-\infty}^{\infty} F(\omega-n\omega_0)$$

where $\omega_0=2\pi/S$, so that when the Nyquist criterion is satisfied, $G(\omega)$ is the same as the original spectrum, and eqn. A.7 will yield h(t)=f(t) for all values of time.

## Appendix B: Author's Biography

The author's mother, Maria K. Gumas, emmigated to the United States where she met and married an electrical engineer named Constantine Gumas. A year later, on the 15th of April 1961, they had their first child, Charles Constantine Gumas. Eighteen years later, the author attended the University of Pennsylvania as an undergraduate. In August 1983, he graduated Cum Laude with a Bachelor of Arts degree and with honors in his major, Physics. He then attended Lehigh University in pursuit of a Master's Degree in electrical engineering. His special interests are in signal processing. After graduation, he plans to work for SPARTA Corporation in McLean, Virginia where speech recognition will be among the problems of study. His future goals include a Ph.D. The author and his fiance Doreen M. Scalco plan to be married in June of 1986.