PATTERN RECOGNITION FOR A CLASS OF WARPED WAVEFORMS WITH APPLICATION TO WELL LOG SIGNATURE RECOGNITION

By

JOHN W. CARTINHOUR, JR.

Bachelor of Science
University of Arkansas at Little Rock
Little Rock, Arkansas
1982

Master of Science
Oklahoma State University
Stillwater, Oklahoma
1984

Submitted to the Faculty of the Graduate College
of the Oklahoma State University
in partial fulfillment of the requirements
for the Degree of
DOCTOR OF PHILOSOPHY
May, 1987

PATTERN RECOGNITION FOR A CLASS OF WARPED WAVEFORMS WITH APPLICATION TO WELL LOG SIGNATURE RECOGNITION

Thesis Approved:

Thesis Adviser

Dean of the Graduate College

PREFACE

The well log signature recognition problem considered in this study is essentially a pattern recognition problem involving waveform shapes which have been altered by a "warping" process. The possible solutions to this problem addressed in this research can be divided into two major categories: (1) methods based on dynamic programming, and (2) methods based on nonlinear prewarping filters.

The research presented herein was supported by the Oklahoma State University Center for Energy Research and by the Oklahoma State University Research Consortium on Well Log Data Enhancement via Signal Processing. The member companies of this consortium include Amoco Production Company, Acro Oil and Gas Company, Cities Service, Mobil Research and Development Corporation, Phillips, Sohio, and Texaco.

I would like to thank my major advisor, Dr. Rao Yarlagadda, for his guidance, patience, and encouragement. I would also like to thank Dr. James Baker, Dr. Keith Teague, Dr. Randy Reininger, and Dr. Zuhair Al-Shaieb for their willingness to serve on my committee.

I would also like to thank Mrs. Kelly Whitfield for her assistance in the final preparation of this manuscript, Dr. Hirak Patangia of the University of Arkansas at Little Rock for the important encouragement he gave me as an undergraduate, and Dr. Gary Stewart of the Oklahoma State University School of Geology for his help with real well log data.

TABLE OF CONTENTS

Chapter		Page
I.	INTRODUCTION	1
	The Well Log Signature Recognition Problem.	1
	Some Interesting Points About Warping	9
	Survey.	15
	Dissertation Overview	23
II.	COMPUTER SIMULATION OF THE WELL LOG SIGNATURE RECOGNITION PROBLEM.	25
	Introduction.	25
	Computer Generation of Random Search Problems	25
III.	SIGNATURE RECOGNITION BASED ON DYNAMIC PROGRAMMING WARPING.	35
	Introduction.	35
	Dynamic Programming Warping (DPW)	36
	Computational Considerations.	52
	Automatic Log Segmentation.	54
	Data Reduction with Sample Rate Adjustment (SRA).	72
IV.	ALTERNATIVES TO THE DYNAMIC PROGRAMMING WARPING APPROACH	83
	Introduction.	83
	Prewarping and Direct Template Matching	84
	Straight Line Prediction Filtering (SLPF) and Sample Rate Adjustment (SRA)	86
	Experimental Results with Direct Template Matching.	93
	On The Job Training (OJT) -- The Basic Idea	102
	The Use of Unitary Transformations for Data Reduction	106
	OJT and Statistical Pattern Recognition: Euclidean Distance and Clustering Transformations	108
	OJT and Statistical Pattern Recognition: Mahalanobis Distance and Probability Density Function Estimation.	114
	OJT in Conjunction with Singular Value Decomposition (SVD)	127
	Using OJT to Choose SLPF and SRA Parameters	129
	SLPF and SRA: More Experimental Results.	134
	Using DFT Magnitude Coefficients and Linear Prediction Coefficients as Features.	142

Chapter Page

V. CONCLUSIONS. 146

Suggestions for Future Research 149

REFERENCES. 158

LIST OF TABLES

Table		Page
I.	Experimental Results: DPW with Automatic Segmentation . . .	61
II.	Experimental Results: DPW with SRA Data Reduction	76
III.	Speedup Technique Summary.	76
IV.	Adjustment of SLPF Parameters (NSKIP and Threshold) with the Window Search Limits Fixed at [N/2, 2N].	94
V.	Adjustment of Window Search Limits with SLPF Parameters Fixed at NSKIP = 8 and Threshold = 1	94
VI.	Effect of Varying the Noise Level About the "Standard" Level ("A" is the Standard Level) with SLPF Parameters Fixed at NSKIP = 8 and Threshold = 1 and with the Window Search Limits Fixed at [4N/5 5N/4].	98
VII.	Adjustment of SRA Parameters (NSKIP and Threshold) with the Window Search Limits Fixed at [N/2, 2N].	99
VIII.	Adjustment of Window Search Limits with SRA Parameters Fixed at NSKIP = 8 and Threshold = 1	99
IX.	Effect of Varying the Model Noise Level While Fixing SRA Parameters at NSKIP = 8 and Threshold = 1 and Fixing the Window Search Limits at [4N/5, 5N/4]	100
X.	Experimental Results for the "Hybrid" Method	100
XI.	Results Using Walsh Transform Coefficients (64 Pt. Transform)	109
XII.	Some Statistical Pattern Recognition Results	109
XIII.	Mahalanobis Distance Results	123
XIV.	PDF Estimation Based on Equation (82) (45 Orthogonal Functions)	123
XV.	PDF Estimation: Sinusoidal Functions (20 Terms)	125
XVI.	PDF Estimation: Sinusoidal Functions (40 Terms)	125

Table Page

XVII. Results: Automatic Selection of SLPF Parameters with OJT. . 133

XVIII. Results: Automatic Selection of SRA Parameters with OJT . . 133

XIX. Experimental Results for the "Hybrid" Method 135

XX. SRA and SLPF Compared. 139

XXI. Summary of Results Based on 100 Random Problems. 147

LIST OF FIGURES

Figure		Page
1.	Signature Search.	3
2.	The Problem of Defining Shape	5
3.	Warping	5
4.	Gama Ray Logs from Two Different Boreholes.	7
5.	Average Value is Not Warp Invariant	10
6.	Discrete Warping.	13
7.	Explanation of Kemp's Method of Pairing and Weighting (After Kemp).	20
8.	(After Gordon and Reyment) Illustration of Slotting	20
9.	Blocky Log.	26
10.	Signature Location on Blocky Logs	26
11.	After Filtering	28
12.	After the Addition of Noise	28
13.	Calculation of the Fit Measurement.	30
14.	Examples of Fit = 0.7	30
15.	Signature Recognition Problems.	32
16.	(a) Path Through an NxM Grid of Points	38
16.	(b) Illustration of Warping Operations Associated with the Path Shown in Figure 16(a).	38
17.	Constraint Region	40
18.	Local Path Restrictions -- Itakura's Method	42
19.	Local Path Restrictions -- Sakoe and Chiba's Method	42
20.	Local Path Restriction for Example in Text.	42

Figure Page

21. Example of Dynamic Programming Algorithm. 46

22. Example of Warping y(n) to Fit x(n) 48

23. Warping Example 49

24. Example where DP Algorithm Fails to Locate the Minimum Distance Path 51

25. Signature Search Based on Segmentation. 56

26. Activity Curve Automatic Segmentation 58

27. Automatic Segmentation Adjustment 59

28. Average Value Subtraction and High Low Matching 65

29. Signature Recognition Using Dynamic Programming Warping and Automatic Segmentation on Gamma Ray Logs. 66

30. Effect of Sample Rate Adjustment. 74

31. Signature Recognition Using Dynamic Programming Warping and SRA for Data Reduction on Gamma Ray Logs. 78

32. Signature Recognition Using Dynamic Programming Warping and SRA for Data Reduction on Gamma Ray Logs. 80

33. Signature Recognition Using Dynamic Programming Warping and SRA for Data Reduction on Gamma Ray Logs. 81

34. SLPF. 88

35. SRA 88

36. Effect of SLPF and SRA. 90

37. Effect of SRA and SLPF on a Test Waveform 92

38. Signature Recognition Problems. 96

39. Standard Warping Functions. 104

40. Warping 105

41. Signature and 6 Warped Versions 126

42. "Class Centroid". 126

43. Algorithm Flowchart for Automatic Selection of SLPF Parameters. 131

Figure Page

44. Effect of Changing the Lowpass Filter Parameter MCUT. 137

45. Signature Recognition Using Direct Template Matching With SLPF on Gamma Ray Logs. 140

46. Signature Recognition Using Direct Template Matching With SRA on Gamma Ray Logs. 141

47. Different Classes, but same DFT Magnitude. 144

48. Point to Point Correlation 154

49. Point to Point Correlation Before "Cleanup". 156

CHAPTER I

INTRODUCTION

The Well Log Signature Recognition Problem

A problem that some geologists spend considerable time on is the correlation of logs from various boreholes in a region of interest. More often than not, this correlation is accomplished by the visual inspection of two well logs placed side by side, which is a time consuming process subject to inconsistencies due to the subjective nature of the comparisons. Naturally, the availability of digital computers has suggested to many researchers the possibility of automating the correlation of well logs. Of course, any realistic computer well log correlation package will have to be one with which professional log analysts can work interactively.

The subject of this work is "well log signature recognition," which is, so to speak, a "subset" of the overall problem. A well log signature is a short segment of a well log corresponding to a rock formation of interest. The correlation of logs can sometimes be broken down into a problem of correlating sections of the logs because of the presence of obvious "marker beds". Within a particular section of one log there may be a signature which a geologist has determined to be of particular interest. Instead of trying to correlate the entire section of this log with corresponding sections of nearby boreholes, it may be sufficient to search these corresponding sections for the signature of

interest. It should be emphasized that in this work it is assumed that the log section which has the original signature of interest may not correlate very well overall with the log section to be searched for the signature. It should also be pointed out that in this work a somewhat narrow view of the problem has been taken; it is treated as a pattern recognition problem involving the shape of the log waveforms under consideration. The inclusion of such information as core data has not been considered. With this caveat in mind, a more specific definition of the problem can next be considered.

In the context of this work, well log signature recognition is a pattern recognition problem which can be defined as follows. (The reader should consult Figure 1 as part of the explanation.) Given a signature sequence S(n), find a subsection of a log sequence Y(n), denoted X(n,K,M) in Figure 1, which best matches the signature S(n). (Or perhaps the goal could be to find several good choices, ranked in order starting with the best match, leaving the final decision to a professional geologist.) As indicated in Figure 1, X(n,K,M) is selected by a rectangular window which is slid along Y(n). In various parts of this dissertation, X(n,K,M) is referred to as a "candidate" sequence, and the notation is usually simplified to X(n) for convenience. The parameter M is the number of points in the rectangular window; parameter K denotes the window shift. In general, the correct values of K and M are both unknown.

It is probably worth pointing out early that this is not like the "traditional" pattern recognition problem involving a fixed number of previously defined classes. With the well log signature recognition problem, there is only one known class (the signature being searched

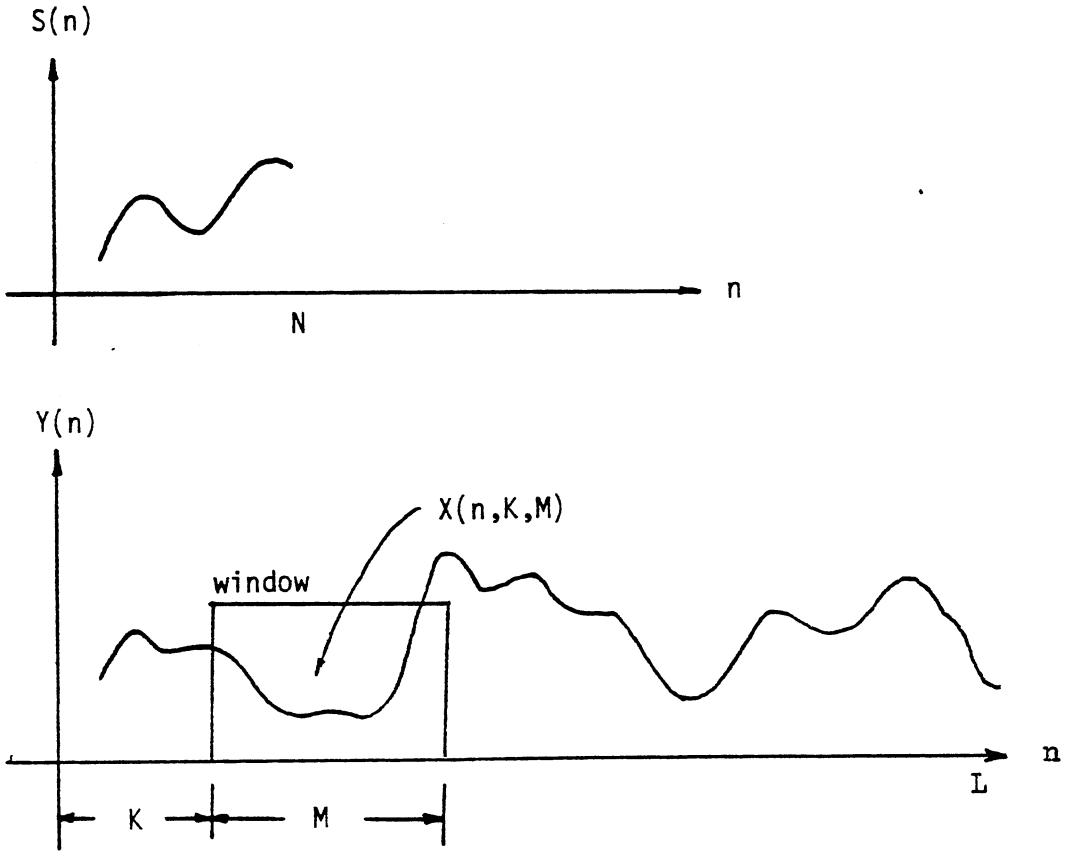


Figure 1. Signature Search

for), and the search algorithm must choose from a collection of candidates which come from previously undefined classes. Not only are the classes undefined, many of them may be difficult to distinguish from the actual signature class.

Variations in bed thickness, logging conditions, etc. cause the "shape" of the signature to change from one borehole to the next. "Shape" is something which can be difficult to define. For example, Figures 2(a) and 2(b) each have two "hills," with the hill on the left being the smaller of the two. If Figure 2(a) is squeezed in some places and stretched in others, a shape such as Figure 2(b) will result. For the purposes of this work, Figures 2(a) and 2(b) have the same "shape," and can be said to be related by a "warping" process. On the other hand, consider Figure 2(c): there are still two "hills," but the amplitudes have been drastically altered. For the purposes of this work, Figure 2(c) does not have the same "shape" as Figures 2(a) and 2(b). The definition of "shape" is clearly application dependent.

In this work the variation in well log signature shape from one borehole to the next is modeled as a warping process. In the continuous domain, "warping" means taking the signature waveform s(t) and replacing the argument t with a monotone increasing warping function w(t). That is,

$$s(t) \rightarrow s[w(t)] \tag{1}$$

It is also assumed that the endpoints of s(t) are mapped to the endpoints of s[w(t)]. Furthermore, to be realistic one should assume some constraints on the severity of the warping, which translates into assuming some upper and lower bounds on the slope of w(t). Figure 3 shows

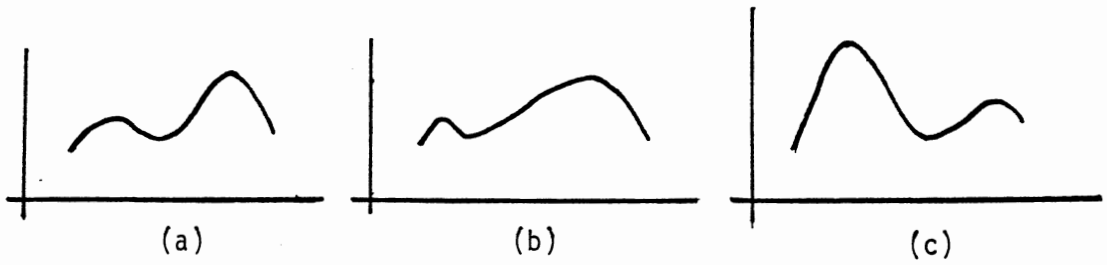


Figure 2. The Problem of Defining Shape

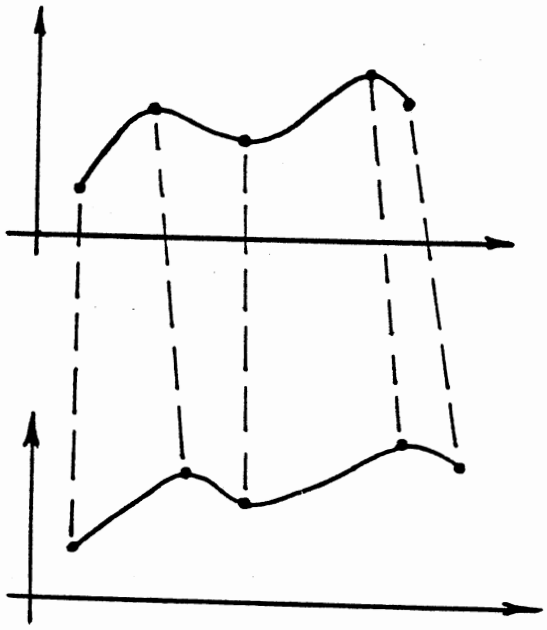


Figure 3. Warping

a hypothetical example of two waveforms, one of which is a warped version of the other.

With this simple model, the amplitudes of the local maxima and minima of a waveform do not change under warping. Of course, in the "real world" there will be small nonuniform amplitude changes in a signature from one log to the next, and the waveforms will be corrupted by noise. It should be remembered that the "true" log waveform has been convolved with the impulse response of the logging tool, and that therefore the shape of adjacent beds has an effect on signature amplitudes. These real world effects are included in the random simulated problems used in this work (to be discussed in Chapter II). It should also be noted that when working with real data, level shifts from one log to the next may also have to be dealt with. (Comments are made regarding level shift preprocessing in conjunction with real data examples considered later in this dissertation). In special cases it may even be necessary to resort to trend removal techniques. However, it should be noted with caution that log amplitudes have significant information which is altered by level shifting or trend removal schemes. These are options which should be provided with a signature recognition package, with the decision whether to use them left up to a professional log analyst. It should also be noted that there is a wide variety of other preprocessing techniques that can resorted to. As Robinson [1] has suggested, even simple transformations such as taking square roots can sometimes be useful. In this work the issue of preprocessing (other than for the level shifting problem) has not been addressed.

Figure 4 shows a real example of warping. The traces shown are gamma ray logs from two adjacent boreholes. The signatures marked off

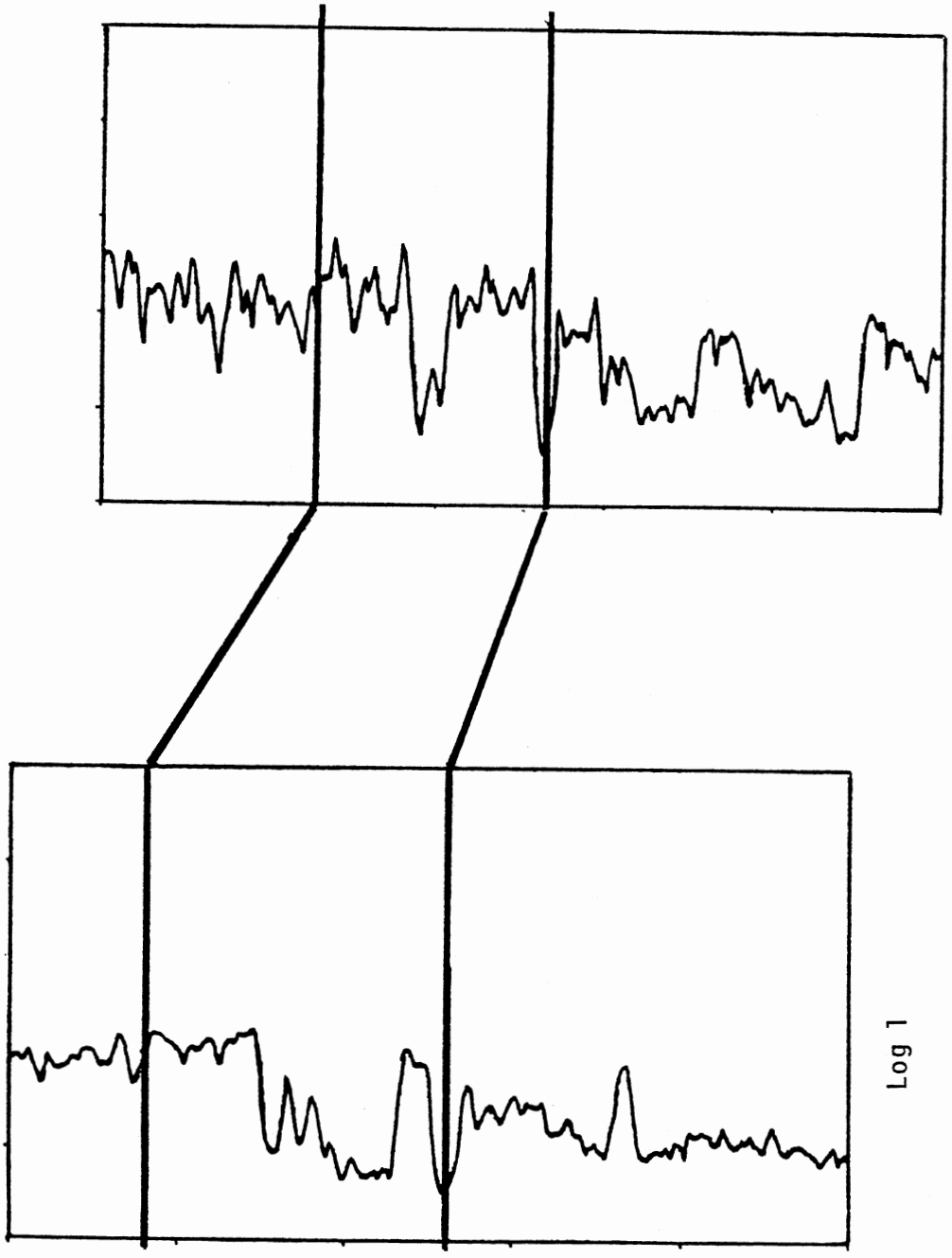


Figure 4. Gamma Ray Logs From Two Different Boreholes

by horizontal lines represent the same rock formation. (These signatures were picked out by Dr. Gary F. Stewart of the Department of Geology at Oklahoma State University based on more information than that shown by these two waveforms, such as other logs, cores, etc.)

With the warping model in mind, the signature recognition problem can now be described as searching the sequence Y(n) (again, see Figure 1) for a candidate sequence which is in the same class as the signature. The signature class consists of S(n) and all possible warped versions. The key question that must be addressed is how to determine a matching figure of merit for two sequences when warping is involved. The possible solutions to this question addressed in this work can be divided into two major categories: (1) methods based on dynamic programming (Chapter III), and (2) methods based on nonlinear prewarping filters (Chapter IV). The second category can be roughly divided into two subcategories: (a) direct template matching, and (b) statistical pattern recognition techniques. In regard to statistical pattern recognition, a method of artificially creating a training set for the signature class has been explored.

One of the major questions faced early in this work is how to objectively measure the "goodness" of a signature search algorithm. A search algorithm should be tested on many example problems to judge its performance. In this work, this difficulty is handled by generating artificial random signature search problems (Chapter II). However, real data has not been neglected.

It would be instructive at this point to take a look at some of the interesting aspects of the warping process. A relationship like $s(t) \rightarrow s[w(t)]$ is perhaps at a glance deceptively simple. The fact that

it is not can be illustrated by the observations contained in the next section.

Some Interesting Points About Warping

The fact that the warping phenomenon is not really simple can be illustrated by the following observations: (1) the average value of a waveform is not warp invariant, and (2) the Fourier series coefficients for a waveform and its warped version are not by any stretch of the imagination simply related. Another point which is important to make here is that a rectangular search window (as in Figure 1) is necessary since the search window must be warp invariant.

The fact that the average value of a waveform is not warp invariant is easily demonstrated by a simple example. Figure 5 shows two waveforms, each of which is a warped version of the other. The average values of the waveforms are indicated by dashed horizontal lines; they are not the same.

The relationship between the Fourier series coefficients of a waveform and its warped version can be derived as follows. Let f(t) be a waveform defined on the interval [0, T]. Then over this interval we can express f(t) as a Fourier series:

$$f(t) = \sum_{n=-\infty}^{\infty} c_n e^{jn\omega_o t} \tag{2}$$

where $\omega_o = 2\pi/T$, and where

$$c_n = \frac{1}{T}\int_0^T f(t)e^{-jn\omega_o t}dt \tag{3}$$

Now introduce a warping function w(t), and write

$$g(t) = f[w(t)] \tag{4}$$

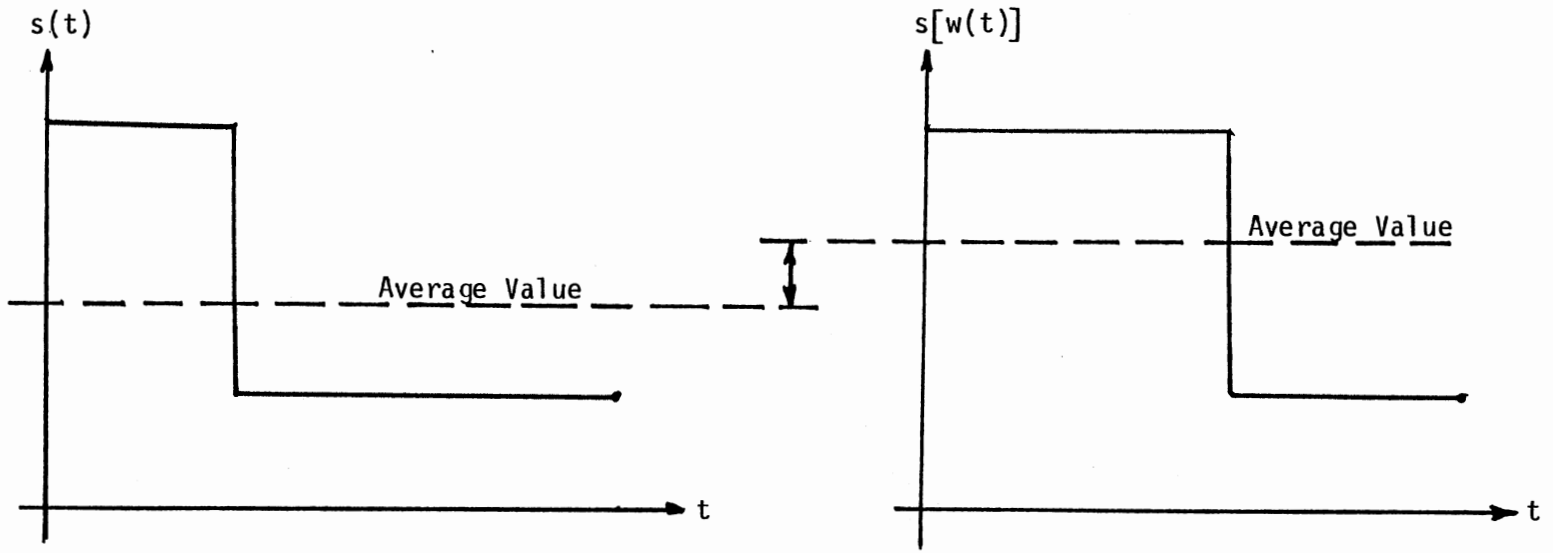


Figure 5. Average Value is Not Warp Invariant

i.e., g(t) is a warped version of f(t). For simplicity (and without much loss of generality) let it be assumed that f(t) and g(t) are both defined over the interval [0, T]. The warped version can also be expanded in a Fourier series:

$$f[w(t)] = g(t) = \sum_{n=-\infty}^{\infty} a_n e^{jn\omega_o t} \tag{5}$$

where

$$a_n = \frac{1}{T}\int_0^T g(t)e^{-jn\omega_o t}dt \tag{6}$$

Using Equations (2) and (4), the following is obtained:

$$g(t) = f[w(t)] = \sum_{n=-\infty}^{\infty} c_n e^{jn\omega_o[w(t)]} \tag{7}$$

Finally, combining Equations (6) and (7) results in a relationship between the Fourier series coefficients of the warped waveform and its original version:

$$a_n = \frac{1}{T}\int_0^T \left[\sum_{n=-\infty}^{\infty} c_n e^{jn\omega_o[w(t)]}\right] e^{-jn\omega_o t}dt \tag{8}$$

which can be expressed as

$$a_n = \frac{1}{T}\sum_{n=-\infty}^{\infty} c_n \left[\int_0^T e^{jn\omega_o[w(t)-t]}dt\right] \tag{9}$$

It is believed that there will be little dissention about the assertion that Equation (9) offers little hope of finding a set of warp invariant Fourier series based descriptors unless the warping function is restricted to special cases such as w(t) = at, where a is a constant -- a restriction which could not be justified in the well log signature recognition problem. (In image processing work it is possible to derive

a set of "Fourier descriptors" for a shape which are invariant under rotation, translation, and dilation [2,3]).

The argument about the difficulty of finding warp invariant parameters based on a Fourier series expansion can be expanded to point to the difficulty of finding warp invariant parameters based on any linear transformation of the form

$$\underline{y} = A\,\underline{x} \tag{10}$$

where $\underline{x}$ and $\underline{y}$ represent discrete sequences Y(n) and X(n), and A is a matrix (the Discrete Fourier Transform (DFT) matrix, for example). First of all, consider the idea of discrete warping functions for sequences as opposed to the continuous case. Figure 6 shows an example of how a discrete waveform is warped. The warping is accomplished by a mapping process depicted by the dashed lines. (As noted in the next section, the problem of nonuniform warping of sequences appears in areas as diverse as molecular biology, speech analysis, and geology). This mapping process can be represented as the multiplication of the original sequence (in vector form) by a warping matrix W consisting of ones and zeros. For example, warping S(n) to create X(n) can be represented as

$$\underline{x} = W\,\underline{s} \tag{11}$$

For example,

$$\begin{bmatrix} X(1) \\ X(2) \\ X(3) \\ X(4) \\ X(5) \\ X(6) \\ X(7) \\ X(8) \end{bmatrix} \quad \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad \begin{bmatrix} S(1) \\ S(2) \\ S(3) \\ S(4) \\ S(5) \\ S(6) \\ S(7) \\ S(8) \end{bmatrix} \quad \begin{bmatrix} S(1) \\ S(2) \\ S(2) \\ S(3) \\ S(5) \\ S(7) \\ S(7) \\ S(8) \end{bmatrix} \tag{12}$$

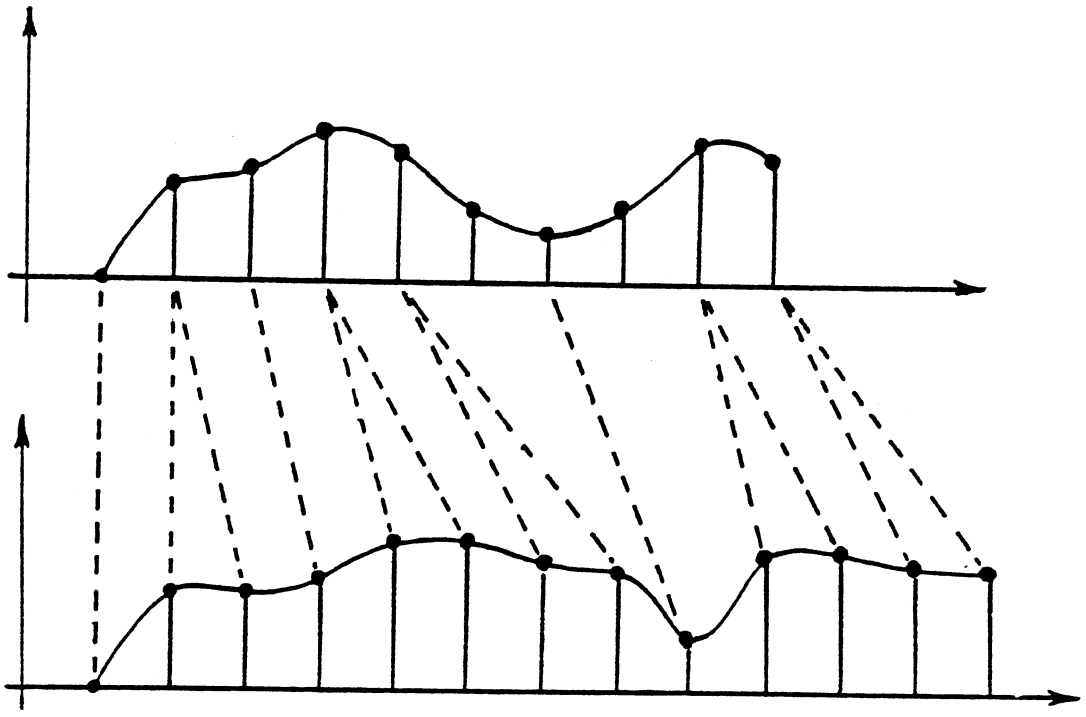


Figure 6. Discrete Warping

Again it is assumed for the sake of simplicity that the warping process under consideration does not change the length of the resulting sequence. Now suppose that both $\underline{s}$ and $W\underline{s}$ are multiplied by a transformation matrix A, i.e.,

$$\underline{y} = A\,\underline{s} \tag{13}$$

$$\underline{z} = A(W\underline{s}) \tag{14}$$

Assuming A is nonsingular, the relationship between the coefficients in the vectors $\underline{y}$ and $\underline{z}$ is given by

$$\underline{z} = A\,W\,A^{-1}\underline{y} \tag{15}$$

Given the nature of the warping matrix W, it seems reasonable to assert that there is no way to combine the coefficients in $\underline{z}$ to produce a set of warp invariant descriptors. Let it also be noted that AWA^{-1} is not in general a valid warping matrix. Therefore, even though $\underline{x}$ and $\underline{s}$ are related by a warping process, the corresponding vectors in the transform domain are not related in this manner.

Finally, consider the assertion that the search window must be warp invariant. Let a signature s(t) and and a warping function w(t) be given. Let x(t) be a warped version of the signature, i.e.,

$$x(t) = s[w(t)] \tag{16}$$

Let a(t) be a window function, and let s(t) and x(t) both be multiplied by this window. The critical question is: are the windowed waveforms a(t)s(t) and a(t)x(t) still related by the warping function? The answer is no, since for arbitrary a(t),

$$a(t)x(t) = a(t)s[w(t)] \neq a(t)s(t)\Big|_{t \to w(t)} = a[w(t)]s[w(t)] \tag{17}$$

However, note that if a(t) is a rectangular window, then a(t) = a[w(t)] and the windowed waveforms are still related by warping.

Having established the fact that the warping phenomenon is non-trivial, the next logical step is to consider how other researchers have approached the signature recognition problem. This is the subject of the next section.

Survey

The signature search problem depicted in Figure 1 was considered by Rudman and Blakely in 1976 [4]. However, their underlying assumption was that the warping process is uniform, i.e., the warping function has the form w(t) = at, where a is some constant. Their method, which is closely related to that of an earlier effort (1973) by Rudman and Lankston [5], involves iterative stretching of the signature S(n) and cross-correlation with the longer section Y(n) (see Figure 1). The section of Y(n) that best matches the given signature is defined to be the section that yields the largest correlation coefficient when compared with the signature.

The basic idea is as follows. Let $S_m(n)$ be the result obtained by stretching the original signature S(n) so that it has M points. (Stretching is accomplished by interpolation.) The cross-correlation of $S_m(n)$ and Y(n), denoted $C_{sy}(K,M)$, is given by

$$C_{sy(K,M)} = \sum_{n=0}^{M-1} S_m(n)y(n+K), \quad K = 0, 1, \ldots, L-M \tag{18}$$

where L is the number of points in Y(n). The cross-correlation is then

normalized to create a correlation coefficient $\hat{C}_{sy}(K,M)$, where $|\hat{C}_{sy}| \leq 1$. The correlation coefficient is given by

$$\hat{C}_{sy}(K,M) = \frac{(1/M)C_{sy}(K,M) - \bar{S}_m\bar{Y}_k}{\left[\left(\frac{1}{M}\sum_{i=0}^{M-1} S_m^2(i)\right) - \bar{S}_m^2\right]^{1/2}\left[\left(\frac{1}{M}\sum_{i=0}^{M-1} Y^2(i+K)\right) - \bar{Y}_k^2\right]^{1/2}} \tag{19}$$

where

$$\bar{S}_m = (1/M)\sum_{i=0}^{M-1} S_m(i) \quad ; \quad \bar{Y}_k = (1/M)\sum_{i=0}^{M-1} Y(i+K) \tag{20}$$

The largest value of $\hat{C}_{sy}(K,M)$ pinpoints the location of the subsection of Y(n) that best matches the given signature. The search is over a predetermined set of values for M.

In 1978 Kwon, Blakely, and Rudman [6] proposed a method of speeding up the algorithm of [4] by replacing the iterative search for the best linear warping factor with a novel frequency domain approach. Their method is described in greater detail in [7]. The basic idea is as follows. Suppose there is a signature s(t) which is transformed by shifting distance k and then warping the distance (t) axis by the warping factor a:

$$\hat{s}(t) = s[a(t-k)] \tag{21}$$

suppose further that y(t), the log being searched, can be expressed as

$$y(t) = n(t) + \hat{s}(t) \tag{22}$$

and that $\hat{s}(t)$ and n(t) are "uncorrelated", i.e.,

$$\int_{-\infty}^{\infty} n(t)\hat{s}(t+g)dt = 0 \text{ for all } g \tag{23}$$

Consider the magnitude squared spectrum of y(t):

$$Y(\omega)Y^*(\omega) = N(\omega)N^*(\omega) + \hat{S}(\omega)\hat{S}^*(\omega) + N(\omega)\hat{S}^*(\omega) + \hat{S}(\omega)N^*(\omega) \quad (24)$$

where $N(\omega)$ and $S(\omega)$ are the Fourier transforms of $n(t)$ and $s(t)$, respectively. The assumption that $\hat{s}(t)$ and $n(t)$ are uncorrelated causes the cross terms of Equation (24) to reduce to zero, resulting in

$$Y(\omega)Y^*(\omega) = N(\omega)N^*(\omega) + \hat{S}(\omega)\hat{S}^*(\omega) \quad (25)$$

The next step in the explanation is to consider how $\hat{S}(\omega)$ can be expressed in terms of $S(\omega)$, where $S(\omega)$ is the Fourier transform of the original signature. It turns out that [8]

$$\hat{S}(\omega) = (1/a)e^{-j\omega k}S(\omega/a) \quad (26)$$

which in turn leads to the expression

$$\hat{S}(\omega)\hat{S}^*(\omega) = (1/a)^2 S(\omega/a)S^*(\omega/a) \quad (27)$$

Substituting Equation (27) into Equation (25) results in an expression relating the magnitude squared spectrum of the log being searched to the magnitude squared spectra of the "noise" $n(t)$ and the original signature $s(t)$:

$$Y(\omega)Y^*(\omega) = N(\omega)N^*(\omega) + (1/a)^2 S(\omega/a)S^*(\omega/a) \quad (28)$$

Next introduce the notation $P_f(\omega) = F(\omega)F^*(\omega)$, and rewrite Equation (28) as

$$P_y(\omega) = P_n(\omega) + (1/a)^2 P_s(\omega/a) \quad (29)$$

Suppose the frequency scale is transformed to a logarithmic scale, i.e.,

$$\omega \rightarrow \log(\omega) \quad (30)$$

Then

$$P_y(\log(\omega)) = P_n(\log(\omega)) + (1/a)^2 P_s(\log(\omega) - \log(a)) \qquad (31)$$

The crucial observation is that it is not unreasonable to hope that the cross-correlation of $P_y[\log(\omega)]$ and $P_s[\log(\omega)]$ -- both of which can be estimated from the available data -- will have a peak at log(a). Therefore, once the location of this peak is determined, the linear warping factor a can be calculated. Once the actual warping factor is known, $\hat{s}(t)$ can be constructed from s(t), and the shift k can be determined from the cross-correlation of $\hat{s}(t)$ and y(t).

As seen in the above discussion, Kwon, Blakely, and Rudman [6] compare sequences of unequal length by "stretching" one of them so that both have the same length, and then calculating a correlation coefficient. However, other novel solutions have been suggested, such as that by Kemp in 1982 [9]. The method, which he classifies as "ad-hoc," can best be explained by a simple example. Suppose there are two sequences:

$$X(n), \quad n = 1, 2, 3, 4$$

$$Y(n), \quad n = 1, 2, 3$$

The "correlation coefficient" (r) defined by Kemp is then

$$\begin{aligned} r = {} & x^*(1)y^*(1)f_1 + x^*(2)y^*(1)f_2 + x^*(2)y^*(2)f_3 \\ & + x^*(3)y^*(2)f_4 + X^*(3)y^*(3)f_5 + x^*(4)y^*(3)f_6 \end{aligned} \qquad (32)$$

where

$$x^*(i) = [X(i) - \bar{x}]/S_x$$

$$y^*(i) = [Y(i) - \bar{y}]/S_y$$

$$\bar{x} = (1/4) \sum_{i=1}^{4} X(i)$$

$$\bar{y} = (1/3) \sum_{i=1}^{3} Y(i) \tag{33}$$

$$S_x = \sqrt{(1/4) \sum_{i=1}^{4} [X(i) - \bar{x}]^2}$$

$$S_y = \sqrt{(1/3) \sum_{i=1}^{3} [Y(i) - \bar{y}]^2}$$

The weights f_i and the pairings x(i), y(j) can be explained graphically by Figure 7 (after Kemp). The boxes intersected by the diagonal line define the pairings. The diagonal is subdivided into lengths d_1, d_2, ... by the intersections with the grid lines. The weights are given by:

$$f_i = d_i/d \tag{34}$$

where d is the length of the diagonal. In this example, the length is defined as d = 3x4 = 12. In general, d = MN, where M is the number of points in one sequence, and N is the number of points in the other. Note that if the two sequences have the same length, the "correlation coefficient" turns out to be the correlation coefficient defined in the usual manner.

The question of how to compare sequences has been the subject of many research papers. The problem arises in many diverse fields: molecular biology, geology, and speech analysis, to name a few [10]. The problem of nonuniform warping often appears, and the suggested solutions often involve the dynamic programming algorithm in some manner. Dynamic programming warping is described in detail by Anderson and Gaby [11], who prefer the term "dynamic waveform matching." They suggested several applications, including well to well correlation. Kerzner has made use of dynamic programming warping, which, he notes,

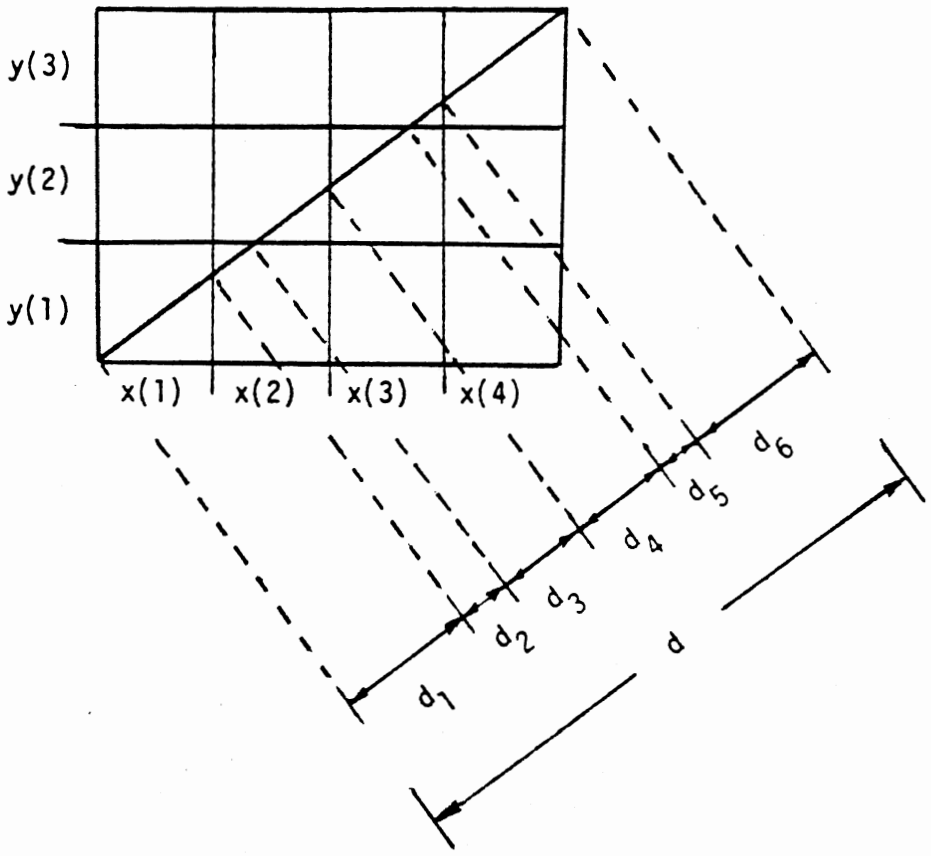


Figure 7. Explanation of Kemp's Method of Pairing and Weighting (After Kemp)

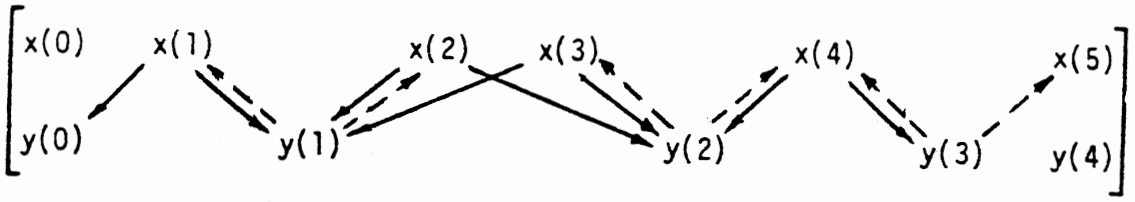


Figure 8. (After Gordon and Reyment) Illustration of Slotting

has been referred to as "spring loaded template matching," in research involving formation dip determination [12] and automatic depth matching of logs [13]. Dynamic programming warping has been used extensively in speech recognition algorithms; see, for example, the work of Itakura [14] and Sakoe and Chiba [15], which will be considered in some detail in Chapter III. Gordon and Reyment have suggested comparing two borehole sequences by the slotting method [16], which uses dynamic programming. The slotting method can be thought of as a form of sequence matching by nonuniform warping since the two sequences are, in effect, squeezed in some places and stretched in others (relative to each other) to obtain a better match. It is an interesting technique that is worth taking a closer look at.

The basic idea of the slotting method is as follows. Suppose we are given two sequences:

$$X(n),\ n = 1, 2, \ldots, N$$

$$Y(n),\ n = 1, 2, \ldots, M$$

The idea is to create a new sequence of length M+N by slotting X(n) and Y(n) together. That is, each element, or "object," of the new sequence is taken from either X(n) or Y(n), with the ordering of X(n) and Y(n) preserved. Each element of X(n) will therefore be located somewhere between two elements from Y(n), and vice-versa, except for the endpoints. For example, the slotted sequence might appear as

$$X(1),Y(1),X(2),X(3),Y(2),X(4),Y(3),\ldots$$

Y(1) is slotted between X(1) and X(2); X(2) is slotted between Y(1) and Y(2), as is X(3), and so on.

For any two sequences there are obviously many possible slottings. The objective is to find the slotting which places similar elements from X(n) and Y(n) together. To accomplish this, Gordon and Reyment first define a measure of "dissimilarity" between two elements X(i) and Y(j). A simple example of such a measure is

$$d[X(i),Y(j)] = |X(i) - Y(j)| \tag{35}$$

They then define a measure of "discordance" $\hat{d}(X,Y)$ for the overall slotting as "the sum (over all objects in each sequence) of the dissimilarities between an object and the two objects in the other sequence which bracket it." Figure 8 (after Gordon and Reyment) shows a possible slotting of X(1), ..., X(4) and Y(1), ..., Y(3). The solid arrows show the dissimilarity comparisons for each X; the broken line arrows show the dissimilarity comparisons for each Y. Thus the discordance for this slotting is:

$$\begin{aligned}\hat{d}(X,Y) = {} & d[X(1),Y(0)] + d[X(1),Y(1)] + d[X(2),Y(1)] \\ & + d[X(2),Y(2)] + d[X(3),Y(1)] + d[X(3),Y(2)] \\ & + d[X(4),Y(2)] + d[X(4),Y(3)] + d[Y(1),X(1)] \\ & + d[Y(1),X(2)] + d[Y(2),X(3)] + d[Y(2),X(4)] \\ & + d[Y(3),X(4)] + d[Y(3),X(5)]\end{aligned} \tag{36}$$

The dynamic programming algorithm is suggested by Gordon and Reyment as a means of finding the slotting that produces the smallest discordance. Dynamic programming is best explained by presenting a simple example; such an example is presented in Chapter III in conjunction with Itakura's technique.

Cheng and Lu have recently suggested the use of tree representation (a form of description language which basically describes a waveform in

terms of its peaks and valleys) as a means of comparing two waveforms [17]. The "distance" between two waveforms is defined as "the minimum number of operations needed to transform one tree to another." They believe that "the primary application for the tree matching method is on cross-well correlation." Another approach to waveform matching is the artificial intelligence technique using syntactic analysis based on a pattern "grammer." An example of a reference on this subject is the paper by Anderson [18] which describes a syntactic pattern recognition procedure for seismic waveforms. In this research, no work has been done with either tree representations or pattern grammers, but even a brief survey of waveform matching would be incomplete without mentioning them.

Dissertation Overview

Chapter II addresses the issue of computer simulation of the well log signature recognition problem. Such simulation has been resorted to in an attempt to find a way to objectively evaluate the performance of signature search algorithms. However, experimental results based on real well log data have not been slighted; real examples are discussed in various places throughout Chapters III and IV.

Chapter III considers well log signature recognition based on the dynamic programming warping algorithm. The basic idea is to compare a signature and a candidate by finding warping functions that optimize the matchup in some sense. Two different dynamic programming warping algorithms are described. Computational considerations are scrutinized, and speedup techniques based on (a) automatic log segmentation, and (b) data reduction via prewarping are presented.

Chapter IV considers alternatives to the dynamic programming warping approach. These alternatives are based on the use of nonlinear prewarping filters which tend to improve the matchup of waveforms in a signature class. Signature recognition based on these prewarping filters can roughly be divided into two categories: (a) direct template matching, and (b) statistical pattern recognition techniques. Statistical pattern recognition techniques depend on the existence of a training set for the signature class; a method of artificially creating such a training set, called "on the job training" (OJT), is presented. A method of using OJT to automatically select the parameters for the prewarping filters is also considered in this chapter.

Chapter V summarizes the results of this work and provides suggestions for future research.

CHAPTER II

COMPUTER SIMULATION OF THE WELL LOG SIGNATURE RECOGNITION PROBLEM

Introduction

A signature search algorithm should be tested on many example problems to judge its performance. It would be a formidable task to obtain hundreds of real examples where the true location of various signatures is known beforehand. In this work, the approach to the testing problem is to automatically generate random logs containing randomly warped signatures. The true location of the signature on the log to be searched is automatically defined during the generation process. The model includes the effects of filtering (tool response) and noise.

Computer Generation of Random Search Problems

The following is a step by step description of computer generation of random well log signature recognition problems:

1. A "blocky" log of 256 data points is created. The amplitude and width of each bed is determined by calling a uniformly distributed random number generator. The amplitude range is [0.0, 10.0]. The bedwidth range, in terms of data points, is [4, 24]; however, this range is easily adjustable. The minimum amplitude change from one bed to the next is 2.0 (also adjustable). See Figure 9.

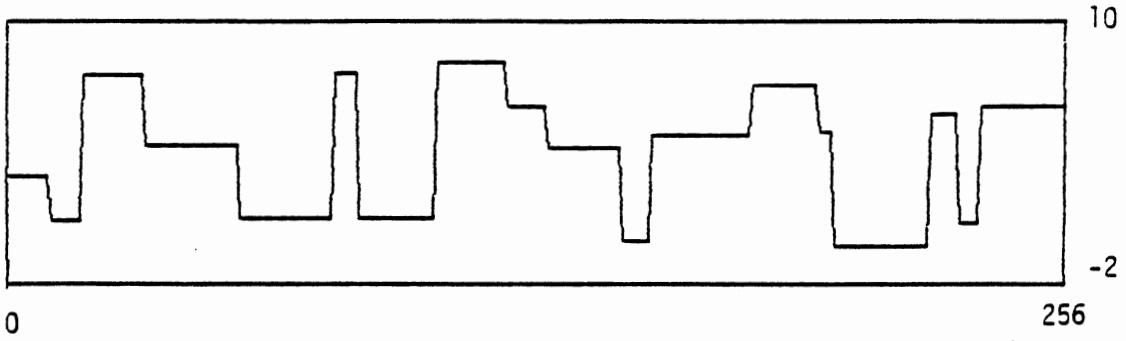


Figure 9. Blocky Log

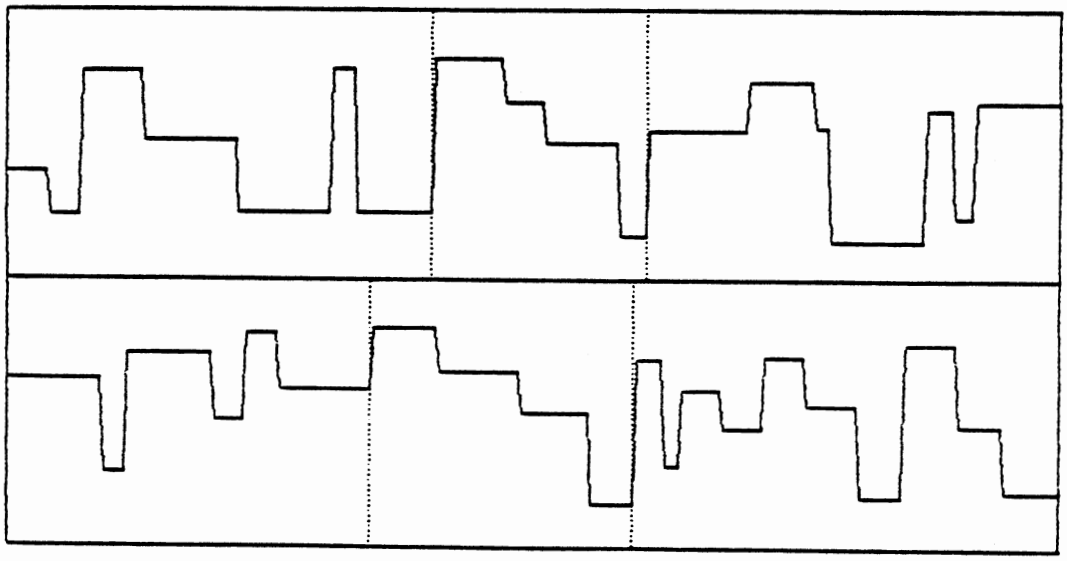


Figure 10. Signature Location on Blocky Logs

2. A four bed section in the middle of the log is chosen as the signature.

3. The signature is removed and randomly warped. This is accomplished by randomly varying each signature bed width, subject to the constraint that a bed cannot be lengthened or shortened by more than a factor of 2. The original amplitudes are unchanged.

4. Another random blocky log is constructed around the warped signature (the location of the signature on this second log is random). At this point, two logs have been constructed, as shown in Figure 10. The signature locations are shown with dotted lines. The "true location" of the signature on the second log (used to compare with the location found by the experimental signature search methods as a means of judging how well the algorithm has performed) is determined by the blocky log signature boundaries.

5. Each blocky log is lowpass filtered by taking the discrete Fourier transform, multiplying by a Butterworth shape lowpass spectrum, and then taking the inverse DFT. The result is as shown in Figure 11. Note that the beds adjacent to the signatures have an effect on the shape of the filtered signatures, which is what one would expect in reality. (The amplitudes are changed slightly.) The signature locations shown on Figure 11 are as determined by the blocky logs.

(A few comments about the lowpass filtering operation are in order. The DFT spectrum of the blocky log is symmetric about the midpoint N/2, which in this case is 128. The lowpass spectrum is also symmetric about this point. The "cutoff frequency" is roughly the point (denoted MCUT) where this lowpass spectrum starts to roll off. The "cutoff frequency" for filtering the blocky logs is MCUT = 60.)

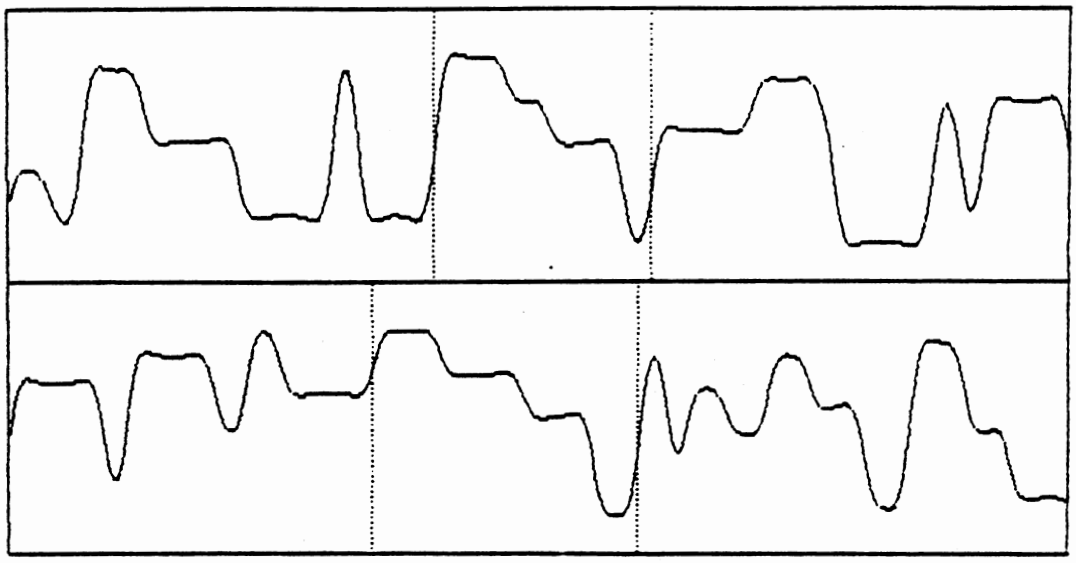


Figure 11. After Filtering

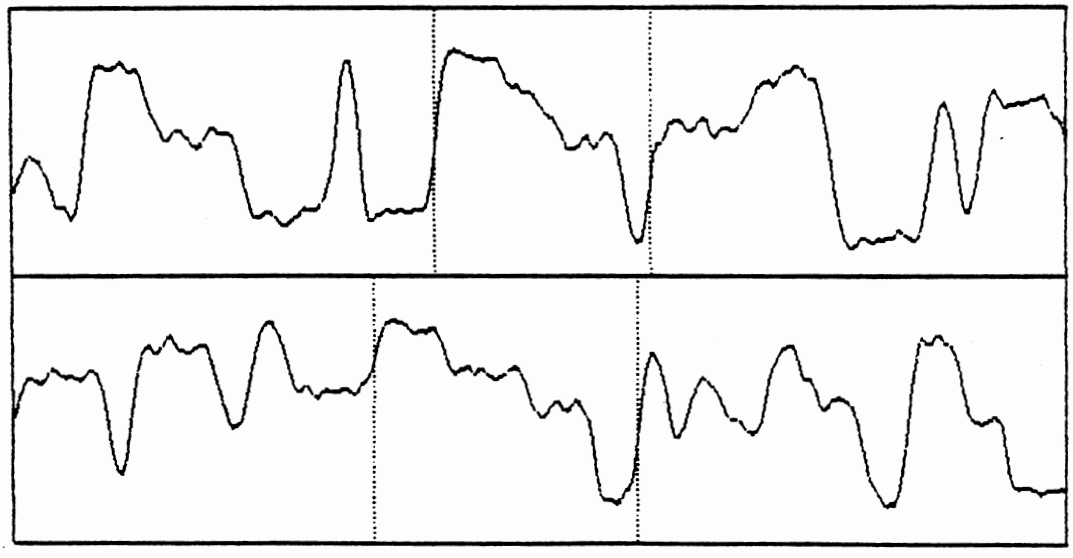
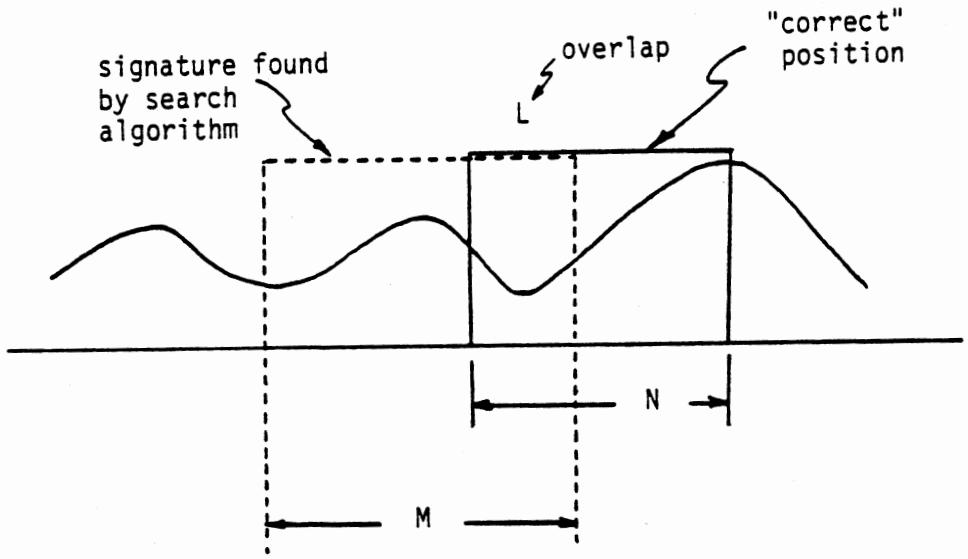


Figure 12. After the Addition of Noise

6. Filtered Gaussian noise is added to each log to create the result shown in Figure 12. The noise is created by Butterworth lowpass filtering a Gaussian white noise sequence. The lowpass cutoff frequency used for the noise is higher than that used to filter the blocky logs. (To be more specific, the "cutoff frequency" used here is MCUT = 110.)

To test a search algorithm, the original signature is extracted from the first filtered/noisy log (the location is determined by the blocky boundaries). The search algorithm then determines the location of this signature class on the second filtered/noisy log. The signature location, as determined by the search algorithm, is then compared with the "true" location. Figure 13 shows how a "fit measurement" is calculated. The fit will be a number between 0.0 (locations have no overlap) to 1.0 (search location equals the true location exactly). Figure 14 shows 3 examples of a fit = 0.7 case; the dark lines are the "true" window, and the dotted lines are the search window. Fit = 0.7 has been chosen as the dividing line for correct decisions and incorrect decisions. The "percent correct" figures shown in some of the tables of experimental results contained herein are the percentages where the fit was greater than 0.7.

It was desired to generate random search problems where simple signature search methods based on the assumption of uniform stretching (i.e., $w(t) = at$, where a is some constant) fail. With this in mind, a simple search method was programmed which calculates the distance between the signature and candidate after stretching the shorter of the two sequences using linear interpolation so that both have the same length. Automatic segmentation was not used. In principle, this is not much different than the method proposed by Rudman and Blakely in 1976



$$\text{Fit} = \frac{L}{\max(M,N)} \quad ; \; 0 \leq \text{Fit} \leq 1$$

Figure 13. Calculation of the Fit Measurement

Solid Line: "true" location; dotted line: search location

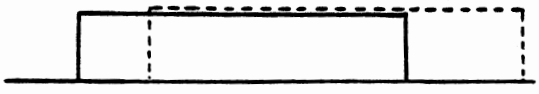
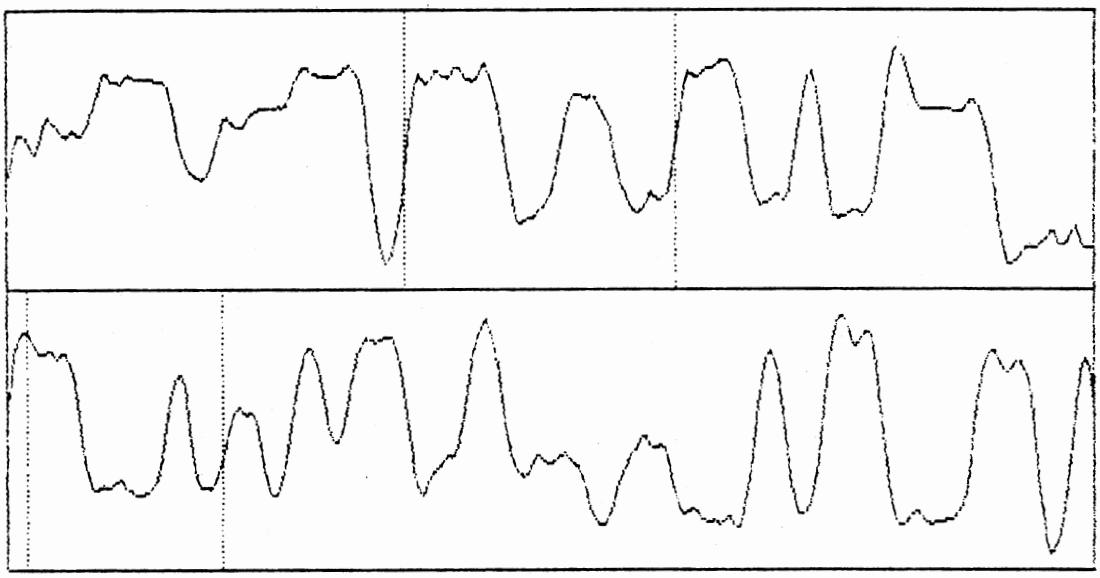


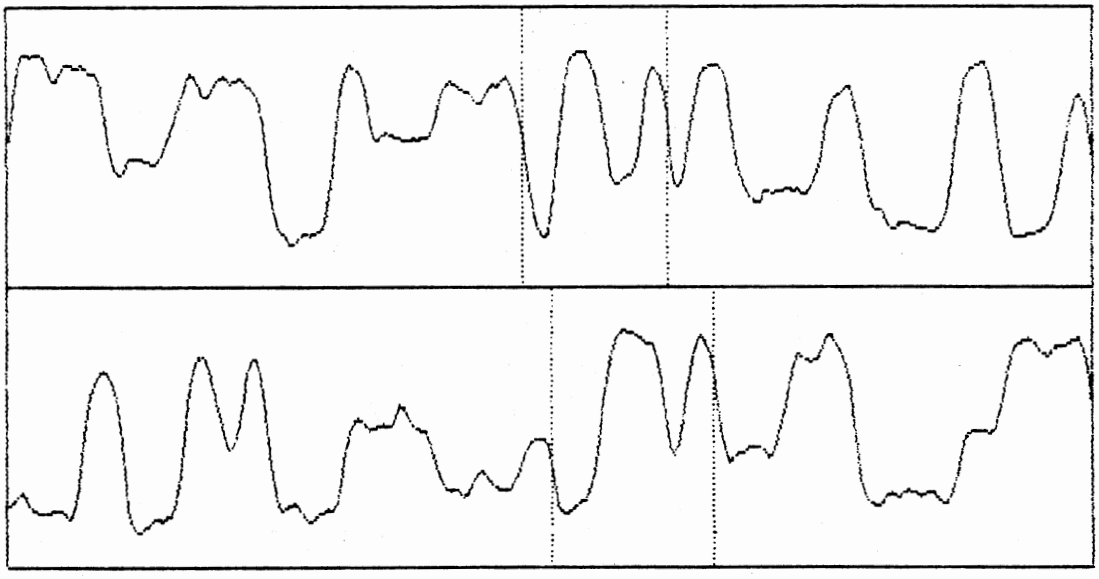
Figure 14. Examples of Fit = 0.7

[4]. The first 100 random problems where this method failed were recorded for future use. Six random search problems are shown in Figure 15.

In the next two chapters, various well log signature recognition algorithms are considered. All such algorithms were tested on the 100 random search problems discussed above. This provides a means of objectively comparing these algorithms.

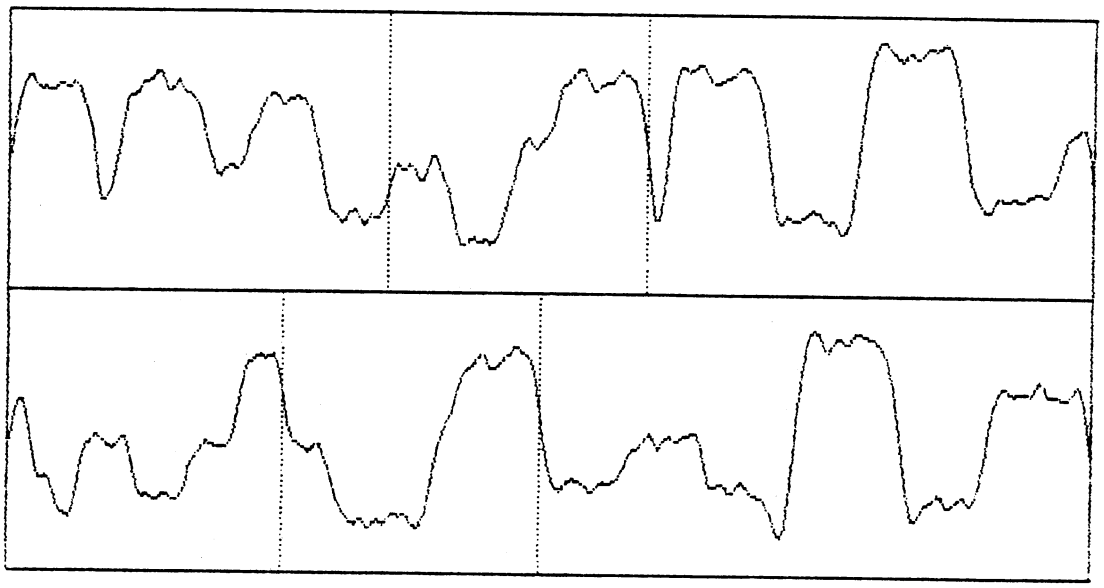


(a)

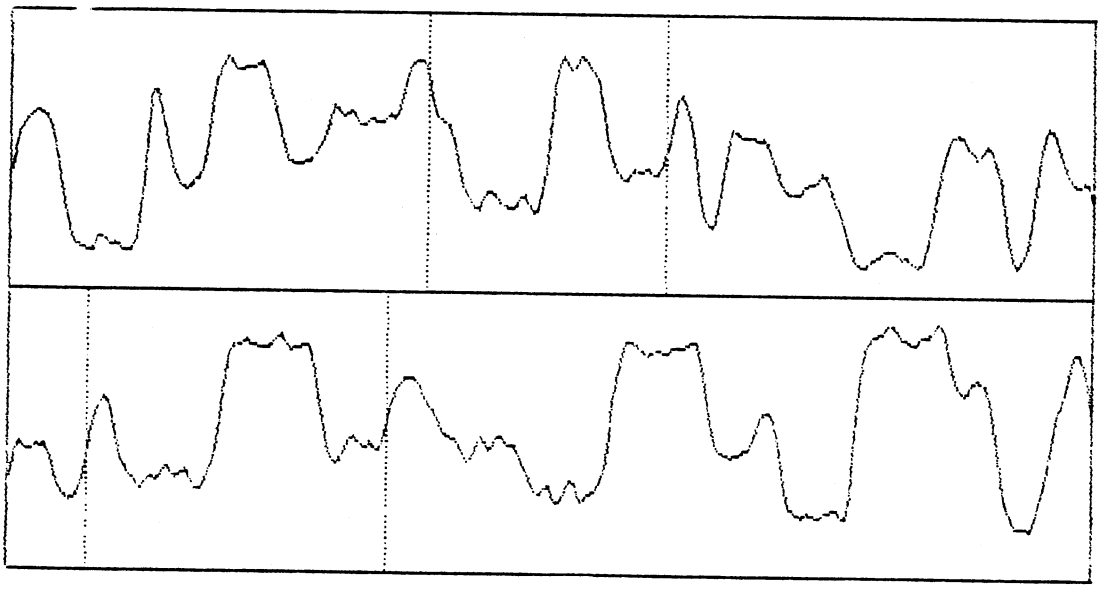


(b)

Figure 15. Signature Recognition Problems

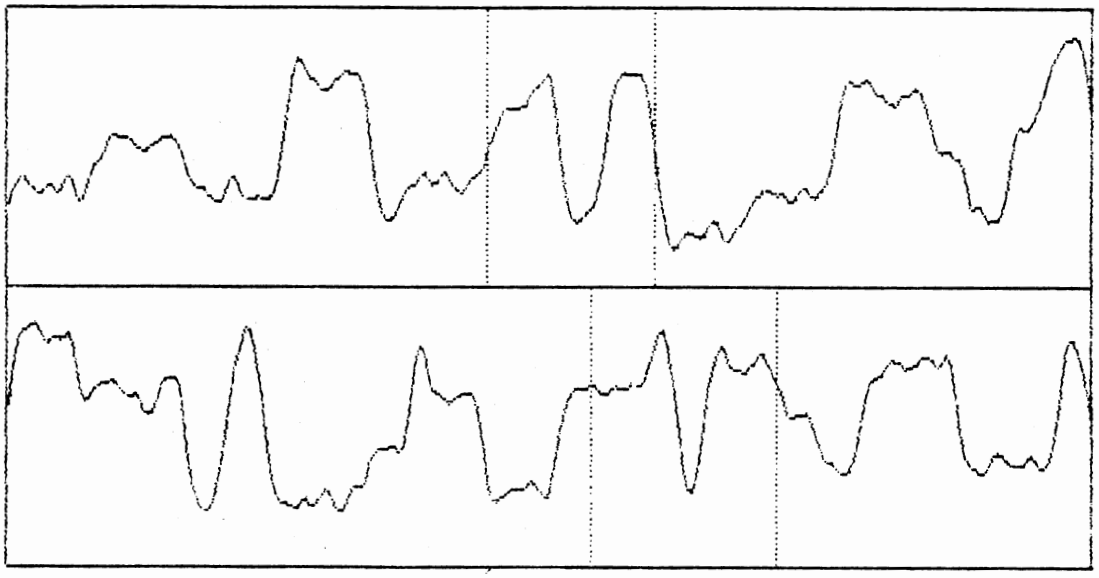


(c)

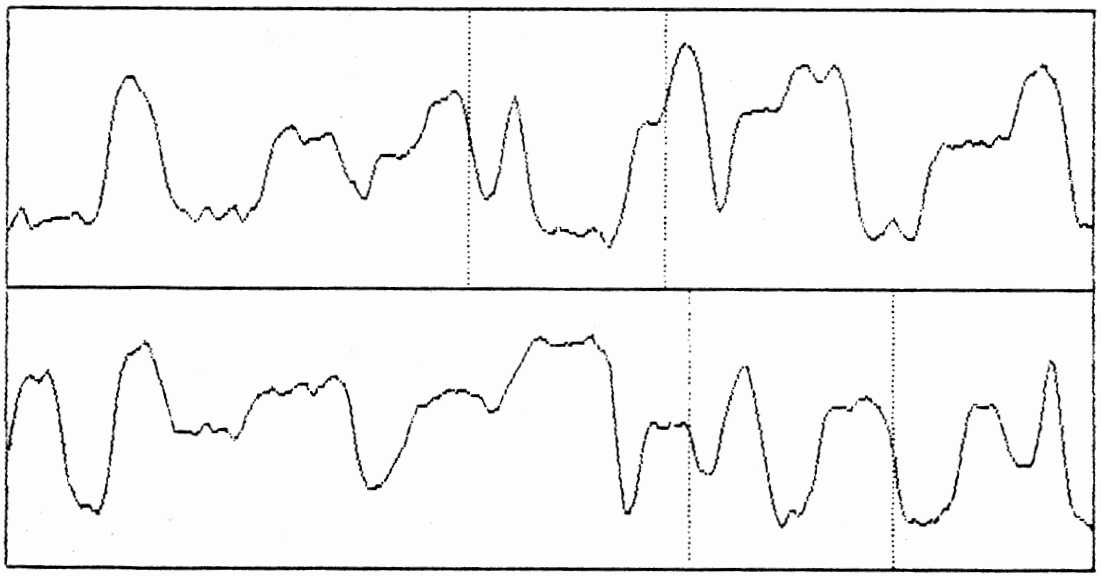


(d)

Figure 15. Continued



(e)



(f)

Figure 15. Continued

CHAPTER III

SIGNATURE RECOGNITION BASED ON DYNAMIC PROGRAMMING WARPING

Introduction

As described in Chapter I and shown in Figure 1, the well log signature recognition problem is a question of finding the candidate X(n,K,M) which comes closest to being in the same class as the given signature S(n). In this chapter, well log signature recognition based on the dynamic programming warping algorithm is considered. The basic idea is to compare a signature S(n) and a candidate X(n,K,M) by finding warping functions (for either or both) that optimize the matchup in some sense. Constraints on the severity of the warping are imposed. A normalized distance measure D(K,M), which has been optimized by the procedure, serves as the matching figure of merit for the candidate X(n,K,M). The candidate with the best matching figure of merit is selected as the best choice for the signature match. Two different dynamic programming warping algorithms are considered in detail in this chapter; the two methods differ in the details of the constraints imposed on the type of warping allowed.

It became obvious very early in the course of this research that in order to make this method viable the excessive requirement for CPU time would have to be reduced. Speedup techniques based on (a) automatic log segmentation, and (b) data reduction via sample rate adjustment (a

prewarping algorithm) are both described in this chapter. Experimental results based on computer simulated random problems and real well log data are presented.

Dynamic Programming Warping (DPW)

The dynamic programming warping techniques considered in this work are both set up in the same manner. Given two sequences to be matched:

$$S(n),\ n = 1, 2, \ldots, i, \ldots, N$$
$$X(n),\ n = 1, 2, \ldots, j, \ldots, M$$

the first step is to establish an NxM grid of points, as shown in Figure 16(a). To each grid point (i,j) a "local distance" d(i,j) is assigned:

$$d(i,j) = |S(i) - X(j)|^p \tag{37}$$

where p is in general a number between 1 and 2, inclusive. (When p = 1, d(i,j) is an "L1" distance; when p = 2, it is an "L2" distance.) The problem is to find the "path," i.e., a sequence of grid points from (1,1) to (N,M), which minimizes the sum of the local weighted distances along the path. For convenience, the points along the path can be numbered: grid point (1,1) is point number 1; the next point on the path is point number 2, etc. Consider a path connecting P points on the grid (grid point (N,M) will be the Pth point). Let d(n) be the distance associated with the nth point along the path. To be more precise, d(n) should perhaps be denoted d[i(n),j(n)], which is in keeping with the notation of Equation (37). Let g(n) be the "weight" assigned to the local path from point n-1 to point n. The problem is to find the path which minimizes the sum D given by

$$D = \sum_{n=1}^{P} d(n)g(n) \tag{38}$$

Note that after normalization the minimum value of D becomes the D(K,M) discussed earlier. Normalization is accomplished by dividing D by the total weight of the path, i.e.,

$$D(\text{normalized}) = D/\left(\sum_{n=1}^{P} g(n)\right) \tag{39}$$

(The simplest warping schemes use g(n) = 1 for all n.)

In general, the minimum distance path can be thought of as describing two discrete warping functions -- $W_S(n)$ for S(n), and $W_X(n)$ for X(n) -- which stretch and/or squeeze these two sequences so that they fit together in the best manner consistent with the restrictions on warping. That is, another general viewpoint is that the goal is to find the warping functions such that the distance given by

$$D = \sum_{n} |S[W_S(n)] - X[W_X(n)]|^P \tag{40}$$

is minimized.

The correspondence between a path through a grid of points and warping functions for two sequences is perhaps best illustrated by a simple example. Figure 16(a) shows an example of a path from (1,1) to (N,M) through a grid of points. The original sequence S(n) of N = 9 points is warped to form the sequence $S_W(n)$:

$$S_W(1) = S[W_S(1)] = S(1)$$
$$S_W(2) = S[W_S(2)] = S(2)$$
$$S_W(3) = S[W_S(3)] = S(3)$$
$$S_W(4) = S[W_S(4)] = S(3)$$
$$S_W(5) = S[W_S(5)] = S(4)$$

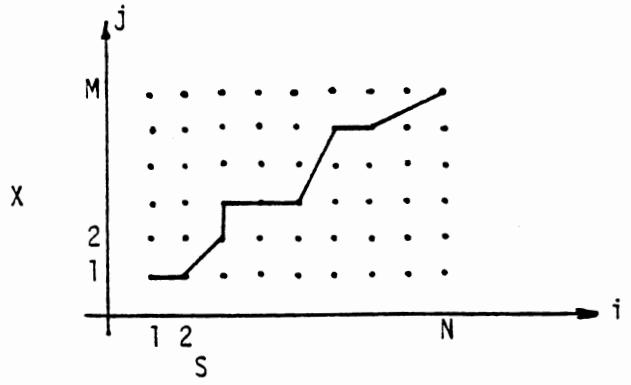


Figure 16(a). Path Through an NxM Grid of Points

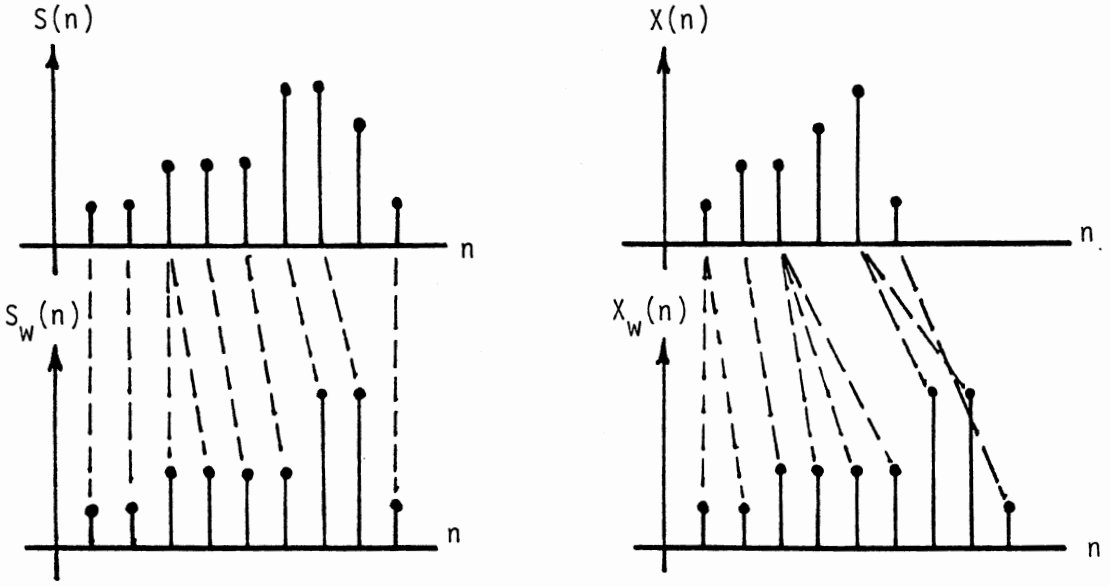


Figure 16(b). Illustration of Warping Operations Associated With the Path Shown in Figure 16(a)

$$S_W(6) = S[W_S(6)] = S(5)$$
$$S_W(7) = S[W_S(7)] = S(6)$$
$$S_W(8) = S[W_S(8)] = S(7)$$
$$S_W(9) = S[W_S(9)] = S(9)$$

Note that S(3) appears twice (stretching), and S(8) has been discarded (squeezing). Similarly, the original sequence X(n) of M = 6 points is warped to form the sequence $X_W(n)$:

$$X_W(1) = X[W_X(1)] = X(1)$$
$$X_W(2) = X[W_X(2)] = X(1)$$
$$X_W(3) = X[W_X(3)] = X(2)$$
$$X_W(4) = X[W_X(4)] = X(3)$$
$$X_W(5) = X[W_X(5)] = X(3)$$
$$X_W(6) = X[W_X(6)] = X(3)$$
$$X_W(7) = X[W_X(7)] = X(5)$$
$$X_W(8) = X[W_X(8)] = X(5)$$
$$X_W(9) = X[W_X(9)] = X(6)$$

Observe that in general a sequence is warped by deleting some samples (squeezing) and repeating others (stretching). Figure 16(b) illustrates the warping operations for this example. Note that for this example, a horizontal local path, i.e., a path from (i,j) to (i+1,j), means that X(n) is stretched at point j. Similarly, a vertical local path, i.e., a path from (i,j) to (i,j+1), means that S(n) is stretched at point i. A local path from (i,j) to (i+1, j+k), where k>1, means that points of X(n) have been deleted. Similarly, a local path from (i,j) to (i+k, j+1), where k>1, means that points of S(n) have been deleted. In practice one places restrictions on the "local paths" on the grid, i.e., on

the number of sequence points in a row that can be repeated or deleted. This is analogous to placing upper and lower bounds on the slope of a continuous monotone increasing warping function. Note also that it is required that the ordering of the points is preserved and that the endpoints of the original sequence become the endpoints of the warped sequence. These restrictions serve to define a constraint region in which the path must lie. The constraint region takes the shape of a parallelogram lying within the NxM grid, as shown in Figure 17. It should be noted that there must be some flexibility in these restrictions to insure that the constraint region does not vanish with some combinations of N and M.

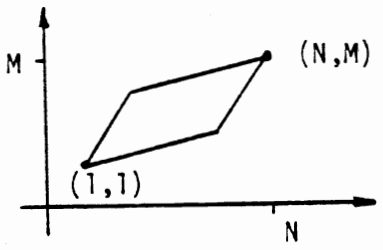


Figure 17. Constraint Region

The minimum distance path can be found by an exhaustive search of all possible paths, but a much better approach is to use the dynamic programming algorithm, which is based on Bellman's "principle of optimality" [19]. Dynamic programming is best explained by means of an example problem; such an example will be presented shortly. However, it would probably be helpful to present a rough outline of the algorithm at this point. For each i, i=1, 2, ..., N, in that order, the best local "backward" path for point (i,j) must be determined, for j=1, 2, ..., M

such that (i,j) is in the constraint region. A local backward path for (i,j) is a path from some point (i-m, j-n) to (i,j), where the range of possibilities for m and n is determined by the specific local path restrictions of the algorithm in use. As noted above, these restrictions are chosen to restrict the severity of the allowable warping. Usually there will be more than one local backward path to choose from; the one that minimizes the accumulated distance for the point (i,j) is chosen. The accumulated distance for (i,j) is the sum of all local distances of the points along the path from (1,1) to (i,j). The final step in the algorithm is to choose the best local backward path for point (N,M); this selection serves to define the final link in the selected path from (1,1) to (N,M), which is either the minimum distance path or a good approximation to it except for pathological cases. (The reason the final answer might not be the true minimum distance path will be explained later.)

Two different dynamic programming warping routines have been used in this research. The first is based on a method proposed by Itakura [14]; the second is based on a method proposed Sakoe and Chiba [15]. In terms of the path through the grid of points, these two methods have different local backward path restrictions, which can be explained as follows.

With Itakura's method, the path to point (i,j) (see Figure 18) can come from (i-1, j), (i-1,j-1), ..., (i-1, j-v), where v is some restriction imposed to lessen the severity of "squeezing." In addition, to lessen the severity of "stretching" points on the original X(n) sequence, the number of horizontal local paths in a row is limited.

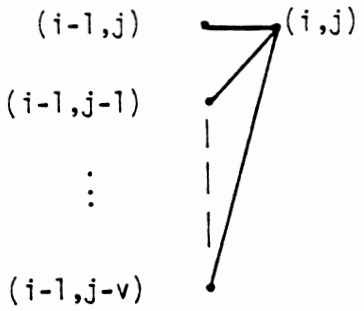


Figure 18. Local Path Restrictions (Itakura's Method)

Figure 19. Local Path Restrictions -- Sakoe and Chiba's Method

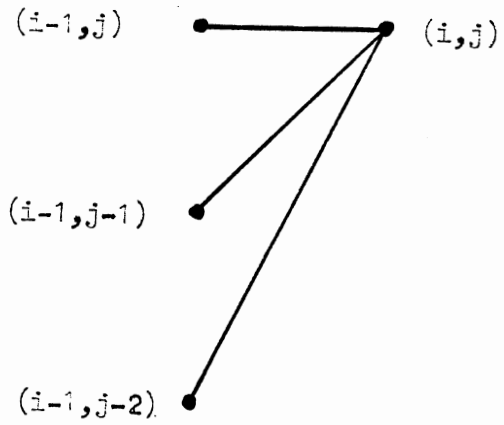


Figure 20. Local Path Restriction for Example in Text

In terms of the general description given earlier (Equation (40)), Itakura's method warps X(n) to fit S(n). In other words, only X(n) is actually warped. It is important to note that this method is not commutative. Suppose Dx is the minimum cumulative distance measure obtained by warping X(n) to fit S(n). Now suppose that the roles of X(n) and S(n) are reversed, i.e., suppose S(n) is warped to fit X(n), and a minimum distance Ds is obtained. In general, Dx and Ds are not necessarily equal.

It is also important to note that Itakura's method allows sample points of a sequence to be discarded in the process of warping it to fit another sequence. Discarding sample points in regions where the waveform is rapidly changing can be dangerous if the sampling is not very "fine" to begin with. The version of this algorithm used in this work always warps the shorter sequence to fit the longer of the two; this will cause the algorithm to tend to favor stretching over squeezing.

Normalization of distance D is accomplished by dividing by the number of points in the warped sequence. That is, with Itakura's method that weighting function g(n) is equal to 1 for all n.

With Sakoe and Chiba's method, the path to (i,j) (see Figure 19) can come from (i-1,j), (i-1, j-1) or (i, j-1). To lessen the severity of stretching of the original waveforms, both the number of horizontal and vertical local paths in a row are restricted. In addition, in order to simplify the algorithm, "90 degree turns" are not allowed. A "90 degree turn" is a horizontal local path followed by a vertical local path, or vice versa.

Unlike Itakura's method, with Sakoe and Chiba's method the number of points in a warped sequence is not known in advance. In other words (in reference to Figure 16), the local path constraints for Itakura's method dictate that there will always be N points on the path from (1,1) to (N,M), but the number of points on this path can be something other than N when Sakoe and Chiba's local path constraints are in place, and the exact number is unknown until the algorithm chooses the final path link. The algorithm includes a means of eliminating the bias in favor of shorter paths; this is accomplished by weighting some local paths more heavily than others.

Sakoe and Chiba's method warps both X(n) and S(n), and does not allow squeezing of either waveform. This method does commute -- that is, if the roles of X(n) and S(n) are reversed, the overall minimum distance measure will be the same.

Local path restrictions can cause the dynamic programming algorithm (as it is usually defined) to fail to find the true minimum distance path (although it will hopefully find a reasonably good path except for pathological cases). An example of this problem will be provided shortly. The problem can be overcome, but the resulting increased algorithm complexity is costly in terms of time since the solution involves checking distances of additional path possibilities. The implementation of Itakura's warping algorithm used in this work does not include compensation for this problem, but the implementation of Sakoe and Chiba's algorithm does. Experimental results (to be covered in detail later) show that Itakura's method is indeed faster.

The following is a simple example of dynamic programming warping based on Itakura's method. Suppose the two sequences are:

n	X(n)	Y(n)
1	.1	0
2	1.6	1.3
3	2.0	1.5
4	2.1	2.0
5	2.2	-

Suppose further that the local path restrictions are as shown in Figure 20, and that the number of horizontal local paths in a row is restricted to one. (These are the most severe restrictions that can be in force if both stretching and squeezing are allowed). The resulting constraint region is shown in Figure 21(a). The local distances $d = [X(i) - Y(j)]^2$ are shown under each dot. Starting at i = 2, note that each point has only one allowable backward path -- the path to (1,1). These paths are shown in Figure 21(b); the numbers under the dots in Figures 21(b) to 21(f) are the local accumulated distances. Moving to i = 3 and starting at the point (3,2), all the allowable backward paths from the point are drawn, as shown in Figure 21(c). The backward path to (2,1) results in an accumulated distance of 3.06, but the backward path to (2,1) yields an accumulated distance of only 0.59, so this path is chosen. Next, the same thing is done for the point (3,3): the backward path to (2,2) yields an accumulated distance of 0.35, but the backward path to (2,3) has a smaller accumulated distance of 0.27, so it is chosen. (The backward path to (2,1) has a large accumulated distance.) Figure 21(d) shows the backward paths and accumulated distances up to and including this step. Moving to i = 4 and starting with point (4,2), note that the one possible backward path within the constraint region (back to (3,2)) is disallowed because it would cause the occurrence of two local horizontal paths in a row. Thus there are no backward paths from (4,2). For the same reason, (4,3) has only one allowable backward path -- to

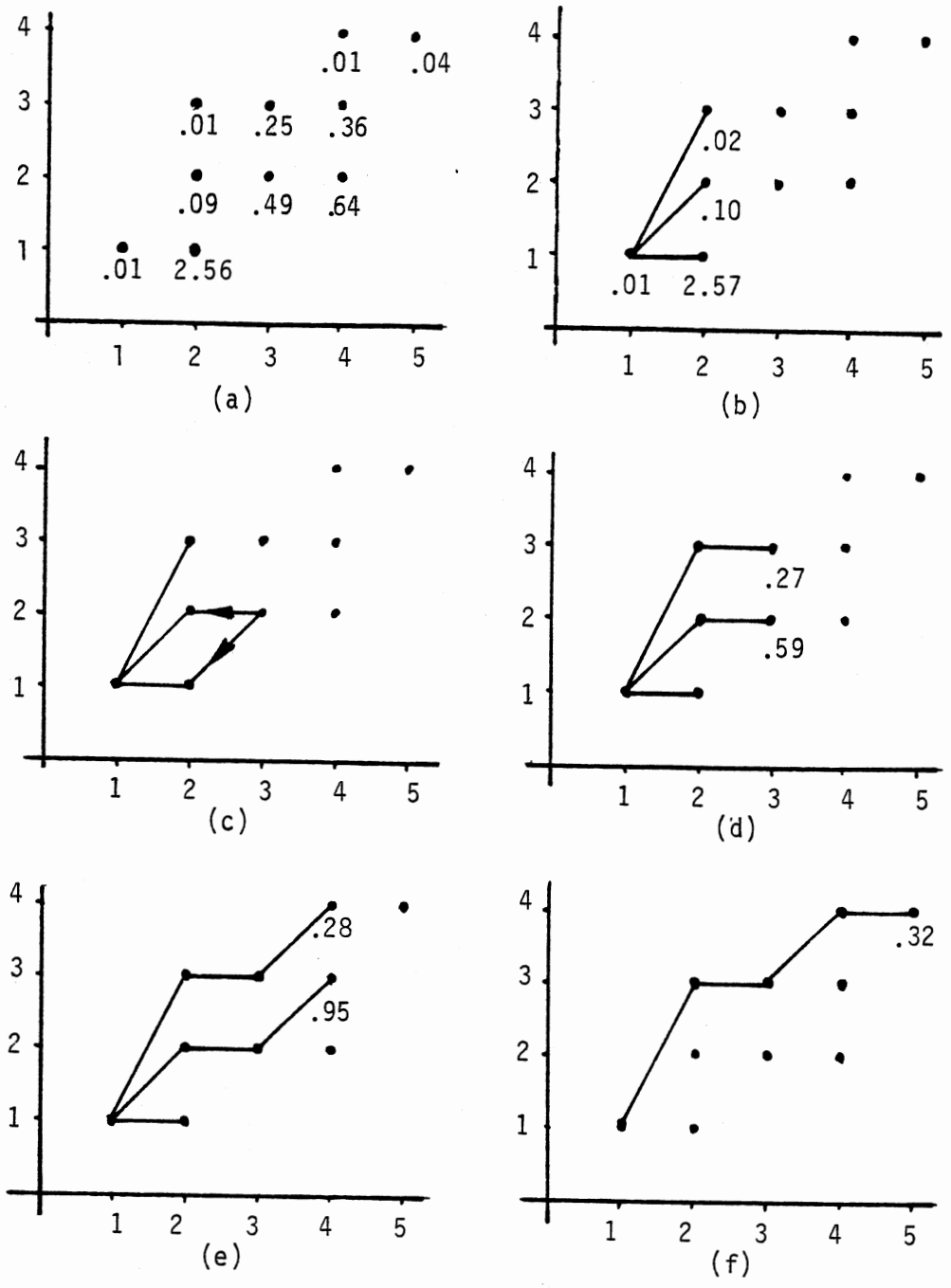


Figure 21. Example of Dynamic Programming Algorithm

(3,2). The point (4,4) has two allowable backward paths, but the one to (3,3) results in a smaller accumulated distance than the one to (3,2). Thus for i = 4, the situation is as shown in Figure 21(e). Finally, consider the backward paths from (N,M) = (5,4). The backward path to (4,2) is disallowed because that point has no backward path from itself. Of the remaining choices, the backward path to (4,4) yields the lowest accumulated distance (0.32). This last decision completes the process; the path is as shown in Figure 21(f). The result is that Y(n) has been warped as shown by the dotted lines in Figure 22; both squeezing and stretching have occurred.

Figures 23(a) to 23(e) are another illustration of how this version of the dynamic programming warping algorithm attempts to warp one sequence to fit the other. In this case, the reference sequence X(n) (shown in Figure 23(a)) has 98 points, while Y(n), the sequence to be warped, has 81 points (Figure 23(b)). The two sequences are shown plotted together in Figure 23(c). Although the sequences have similar shapes, they don't match very well when plotted together. Figure 23(d) shows the result produced by dynamic programming warping. The warped version of Y(n), denoted $Y_w(n)$, is plotted together with the reference sequence X(n); the match is clearly improved. Figure 23(e) shows the result when the roles of X(n) and Y(n) are reversed, i.e., with X(n) warped to match Y(n). Once again, the match is improved.

As noted earlier, the inclusion of a restriction on the number of times in a row a local path can be horizontal means that the dynamic programming algorithm may not locate the minimum distance path. The reason is that if at some point a backward horizontal path is selected, the availability of future path choices is affected. This problem is

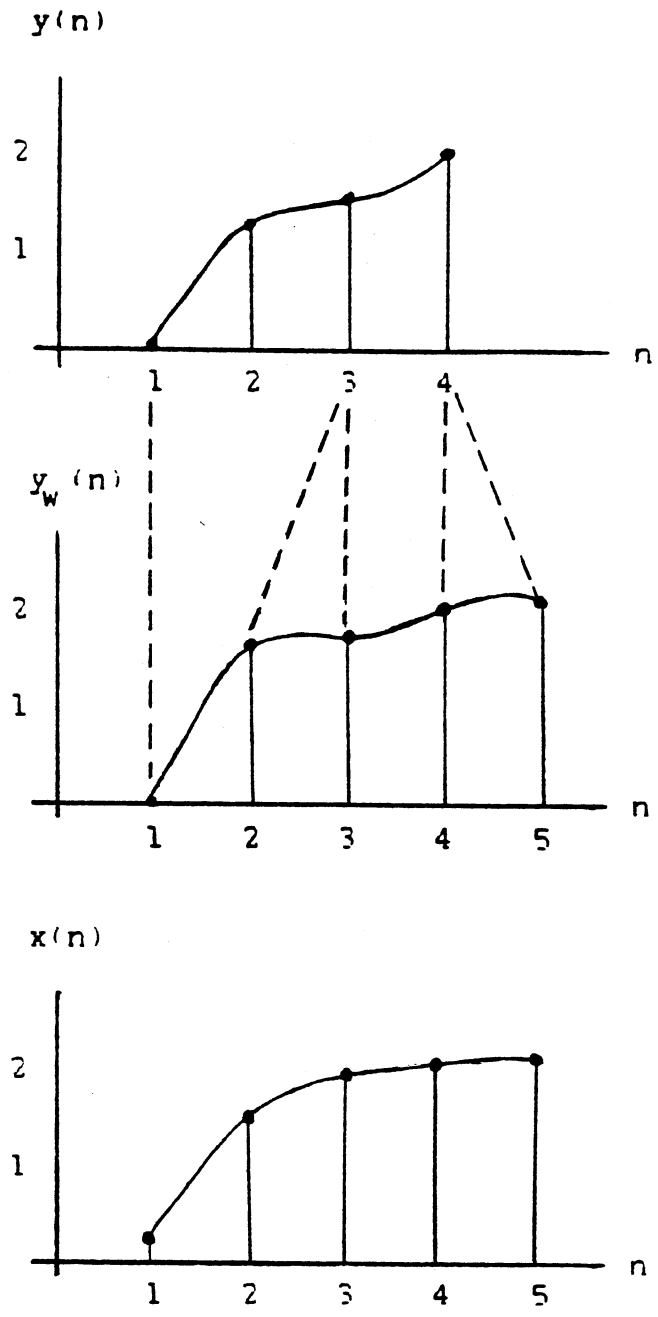
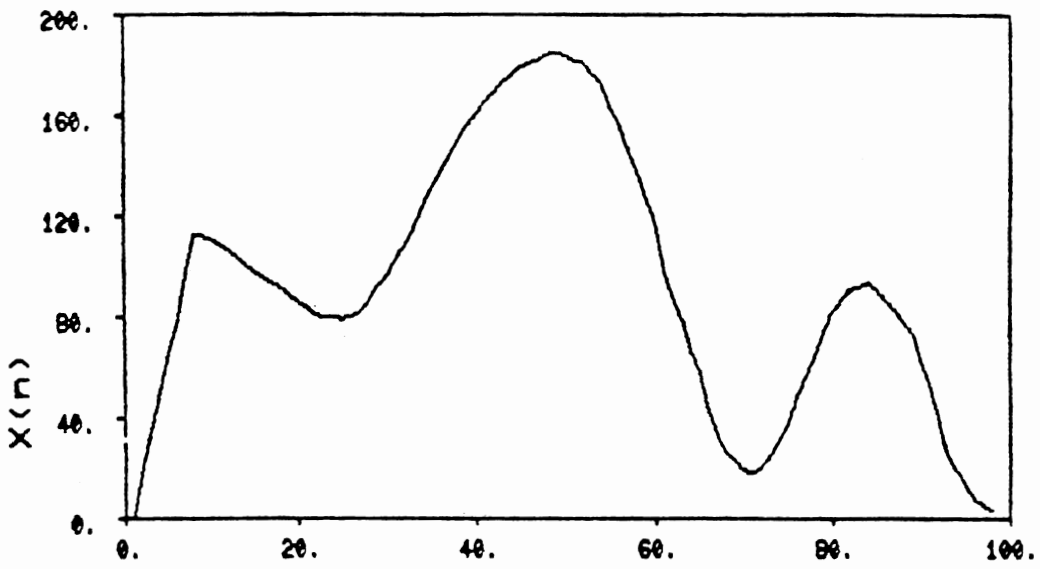
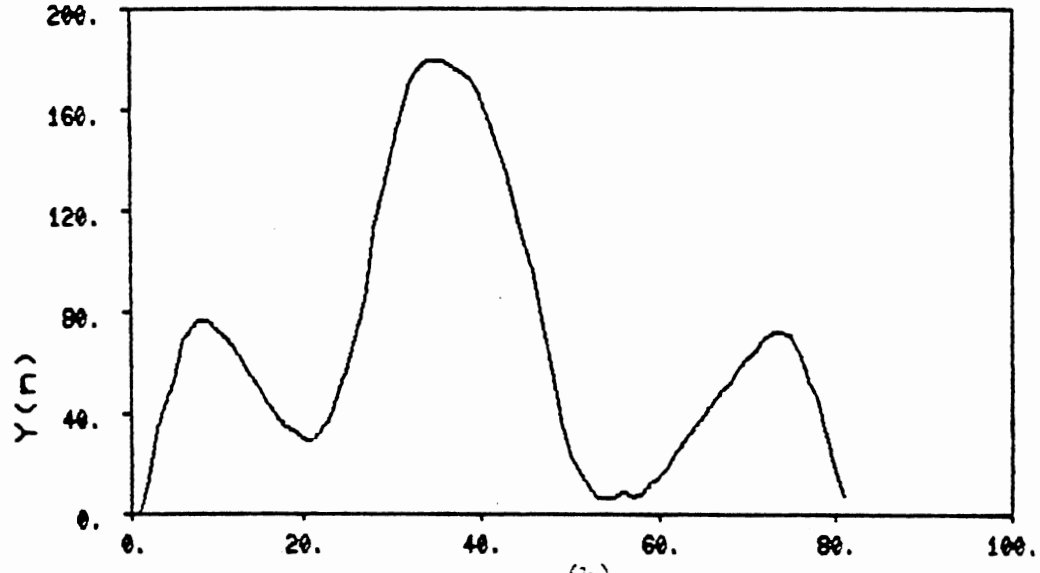


Figure 22. Example of Warping y(n) to Fit x(n)

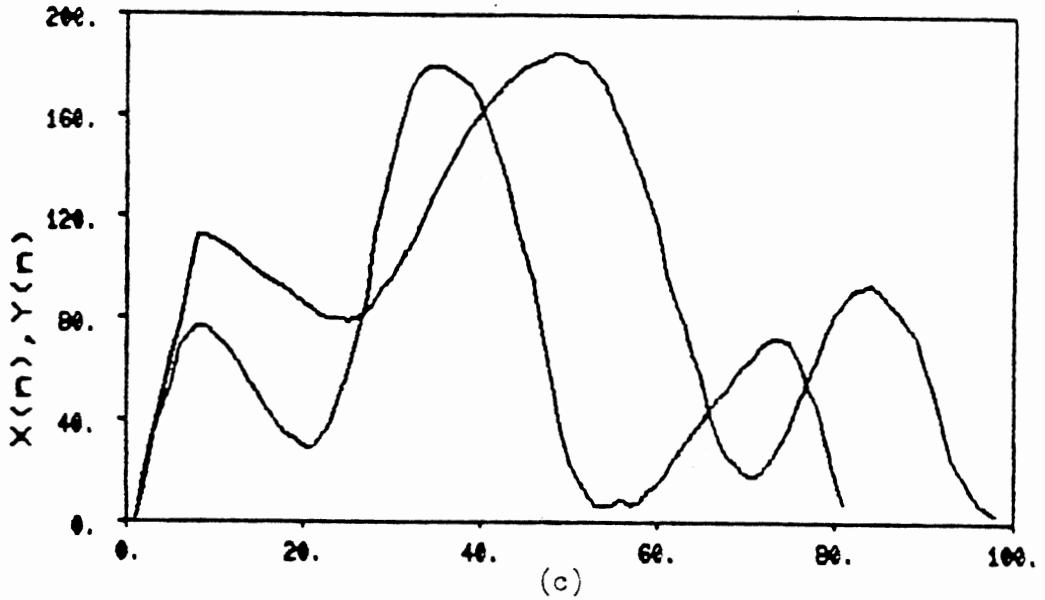


(a)

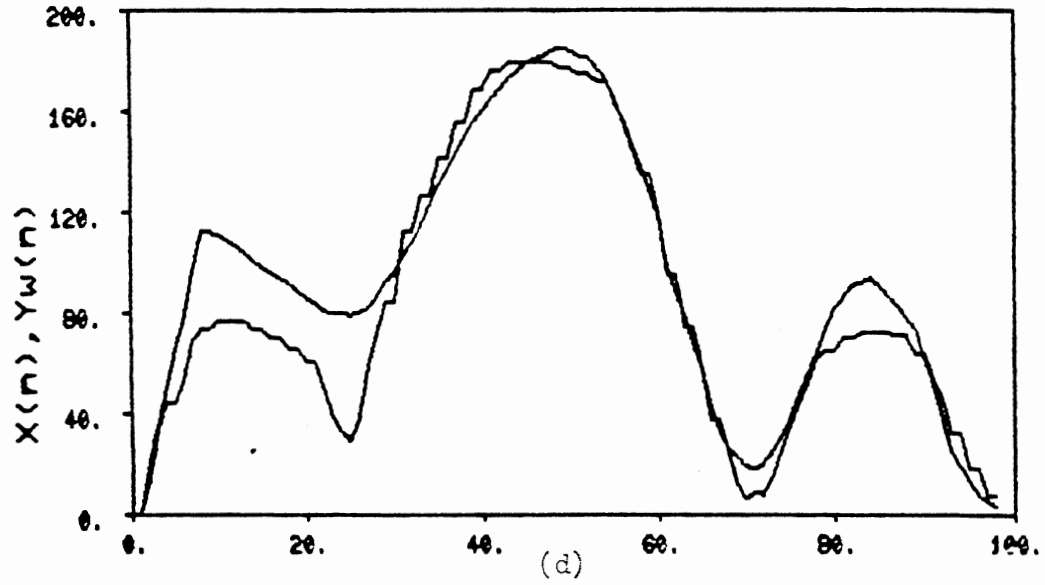


(b)

Figure 23. Warping Example

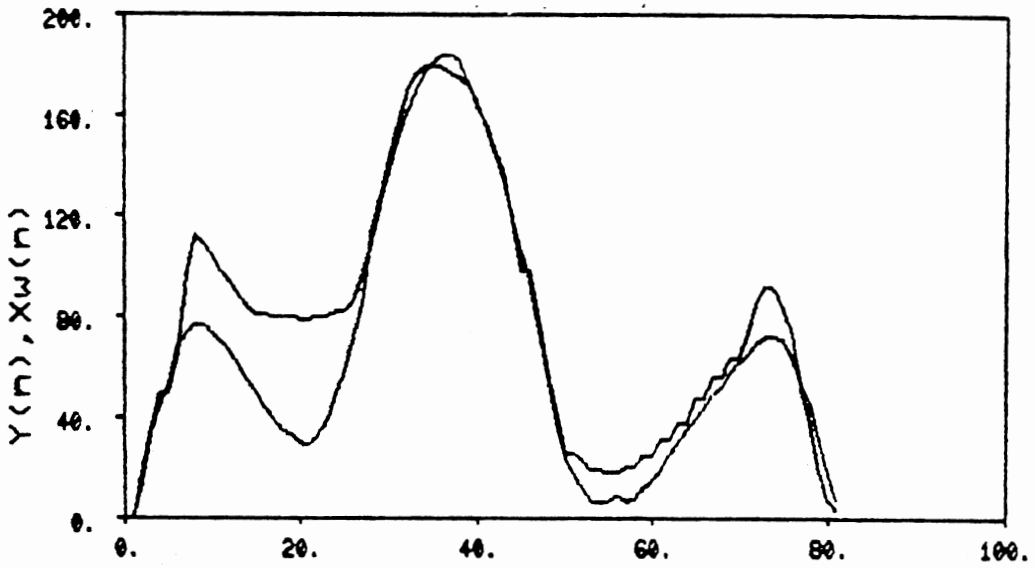


(c)



(d)

Figure 23. Continued



(e)

Figure 23. Continued

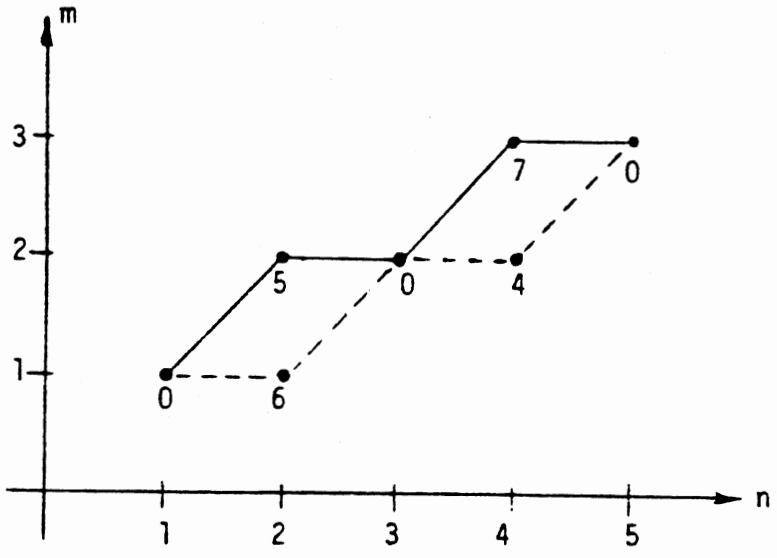


Figure 24. Example where DP Algorithm Fails to Locate the Minimum Distance Path

best clarified by a simple example. Figure 24 shows a constraint region for N=5, M=3 with the same local path constraints used in the earlier example. The dynamic programming algorithm will select the path shown by the solid lines, which results in an accumulated distance of D=12. Observe that at the point (3,2) the algorithm will choose the backward path to (2,2) because this selection will result in the minimum accumulated distance to the point (3,2). However, the selection eliminates the path section (3,2)→(4,2) from future consideration because of the constraint on horizontal paths. The actual minimum distance path is shown by the dotted lines, which has an accumulated distance of D = 10.

Now that the necessary details of dynamic programming warping have been covered, the question of the CPU time requirement for signature recognition based on this method can be considered. This is the subject of the next section.

Computational Considerations

Consider once again the search problem depicted in Figure 1. Since K and M are unknown, the number of candidates X(n,K,M) is very large. The dynamic programming algorithm, which must be invoked for every candidate, is much faster than an exhaustive search, but it is still computationally intense. Therefore, some restrictions on the scope of the search are necessary, i.e., it is desirable to cut down on the number of candidates. For openers, upper and lower bounds must be placed on M, denoted M_{max} and M_{min}. It can be shown that if all possible values of M and K are investigated, the number of candidates C is

$$C = \sum_{i=M_{min}}^{M_{max}} (L - i + 1)$$

$$C = (M_{max} - M_{min} + 1)(L + 1) - (M_{max}^2 + M_{max} - M_{min}^2 + M_{min})/2 \qquad (41)$$

For example, if the log has 256 points and candidates of lengths from 35 to 140 points are considered, there are 17,967 possibilities. However, the number of candidates can be reduced by increasing the length of the window by more than one point at each iteration for M, and by sliding the search window by more than one point at each iteration for K. If the iterative incremental changes for M and K are set to 3, the number of candidates is reduced by approximately an order of magnitude. Unfortunately, experiments with 256 point randomly generated logs have shown that even with this reduction, the process is unacceptably slow. With the incremental changes for M and K set to 3, the average CPU time requirement for a signature search based on Itakura's method is 321 seconds. (The window size was varied from N/2 to 2N, where N is the number of points in the given signature). If Sakoe and Chiba's method is used, the average time jumps to 1609 seconds. (These results are based on 5 search problems.) Therefore, two ways of significantly speeding up the process have been considered: automatic log segmentation and data reduction (discarding points with a prewarping filter). But before taking up these ideas in detail, it should be noted that there are opportunities for the employment of parallel processing schemes to significantly speed up the search. First of all, all searches involving different window sizes could be performed in parallel. Secondly, there are parallelisms in the dynamic programming algorithm itself which can be taken advantage of [20]. Parallel processing has not been used in this work. However, it is an important topic, and obvious opportunities in this direction need to be pointed out, even if only in passing.

Automatic Log Segmentation

Several sophisticated iterative segmentation procedures have been proposed in the literature. For example, see the papers by Pavlidis [21] and Hawkins and Merriam [22]. Other references of interest in this area include (but are by no means limited to) papers by Witkin [23], Blumenthal, Davis, and Rosenfeld [24], and Webster [25]. However, since in this work the motivation for investigating automatic segmentation is to speed up the signature search, relatively simple "one pass" segmenting techniques have been the focal point of attention. The one pass segmentation algorithm used in the signature search scheme presented here is based on the "activity curve" method proposed by Kerzner [12,26]. The governing philosophy here is that a fast one pass method should be used, and then the segment based signature search algorithm should be designed to be relatively insensitive to the inevitable segmenting errors. It is easier (and probably faster) to simply widen the scope of the search to lessen the impact of segmenting errors instead of spending a lot of time using recursive methods to fine tune the segmenting process. Since no segmenting technique could be expected to accurately detect bed boudaries every time, errors would have to be allowed for no matter what method was chosen. The principle difficulty is the presence of noise.

Before getting into the details of the one pass segmentation algorithm, it is instructive to consider a possible segment based signature search scheme:

(1). Apply the segmentation algorithm to the signature S(n). Let N_S be the number of segments in the signature, as determined by the algorithm.

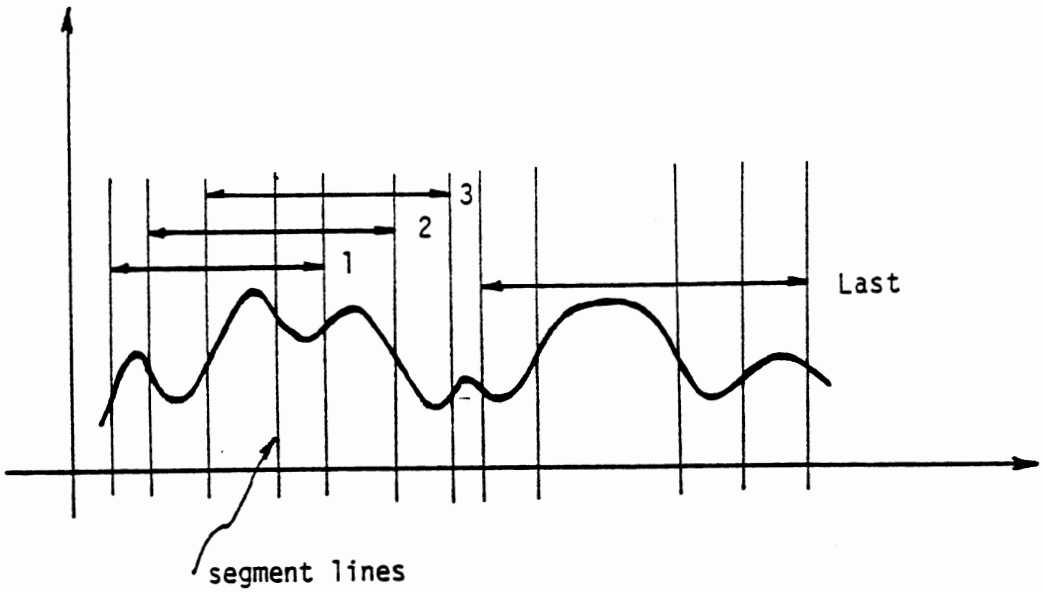
(2). Apply the segmentation algorithm to the log Y(n) to be searched for the signature.

(3). Define the search window in terms of the number of segments as well as by the number of points M and the shift K. The search will consider all candidates X(n,K,M) consisting of $N_s - N_a$, $N_s - N_a + 1$, ..., N_s, $N_s + 1$, ..., $N_s + N_u$ segments in a row. N_a and N_u are "fudge factors" which compensate for segmenting errors; it is quite possible that the true location of the signature on Y(n) will be judged to have a different number of segments by the segmentation algorithm than will S(n). Figure 25 shows how a 4 segment window is slid along a log. (One could include upper and lower bounds on the size of the search window such that candidates selected by this method that are either too long or too short would not be considered. This restriction was not imposed in this work, however.)

Activity curve analysis is one of a class of short-time analysis techniques which include such methods as short time energy, average magnitude, and zero-crossing rate. A good reference on this subject is the digital speech processing book by Rabiner and Schafer [27]. The activity curve segmentation algorithm is based on the idea that boundaries should be drawn through those areas which have the largest local variances. A local variance is the sample variance of a windowed sequence of data points, i.e.,

$$\text{Local variance} = (1/N) \sum_{\substack{\text{points in} \\ \text{window}}} [X(i) - \bar{x}]^2 \qquad (42)$$

where $\bar{x}$ is the mean value of the points in the window, and N is the number of points in the window. An activity curve describes the local variances along a sequence; the local maxima of the activity curve tend



1 first window position
2 second window position
etc.

Figure 25. Signature Search Based on Segmentation

to correspond to the steepest slopes of the curve being segmented. Consider Figure 26. The window is approximately centered at point k along some sequence X(n). The window is of width 2m+2. As the window is slid along the sequence, the local variance is calculated at each position. The end result is an activity curve given by

$$e(k) = \frac{1}{(2m+2)} \sum_{i=k-m}^{k+m+1} X^2(i) - \left[\frac{1}{(2m+2)} \sum_{i=k-m}^{k+m+1} X(i) \right]^2 \qquad (43)$$

The activity curve can be expected to have some small local maxima created by noise alone. Therefore, a threshold is established, as shown in Figure 26: a segment line is drawn on the original sequence at each spot corresponding to a local maximum on the activity curve exceeding the threshold.

Two parameters -- window length and threshold -- must be adjusted carefully to yield the best results. The window must be narrow enough to detect thin beds, but not so narrow as to be overly susceptible to noise. Adjustment of the threshold is also a compromise. The threshold must be high enough to prevent noise from dominating the process, but low enough to detect the gentle slopes associated with some bed boundaries. Figures 27(a), 27(b), and 27(c) show how sensitive the process is to parameter twiddling. Figure 27(a) shows the true bed boundaries as determined by the random log construction process (Chapter II). Figure 27(b) is the activity curve segmentation that resulted after trial and error adjustment of the threshold and window length. Figure 27(c) is an example of the segmentation that resulted before the adjustments were complete.

A signature search algorithm based on automatic log segmentation was tested on 100 randomly generated search problems. The activity

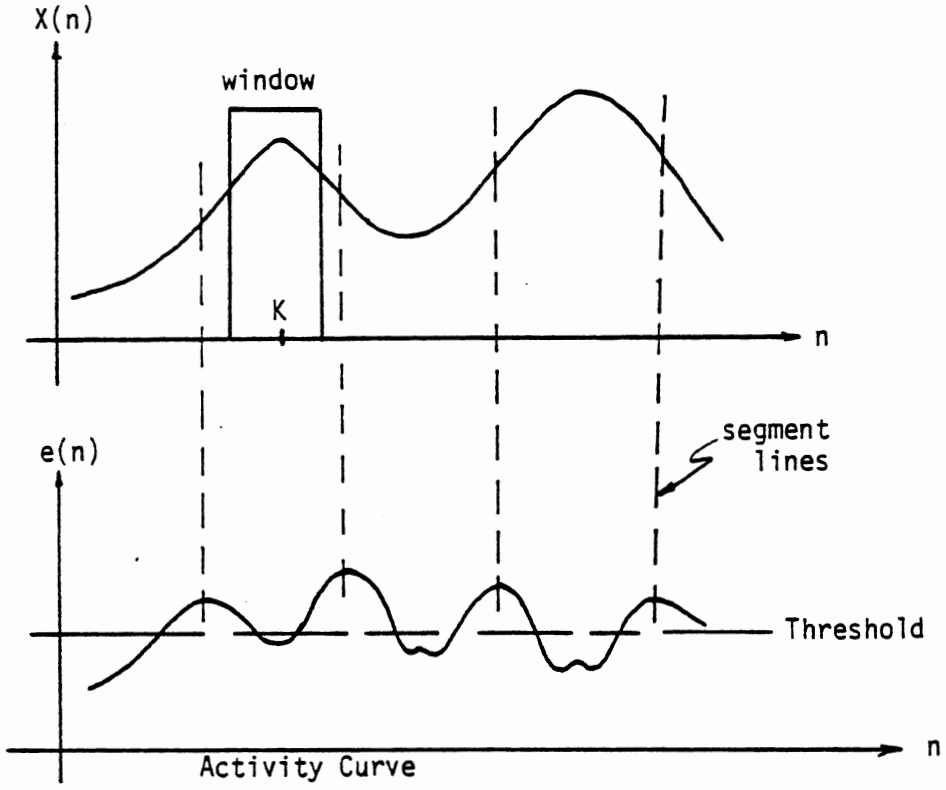
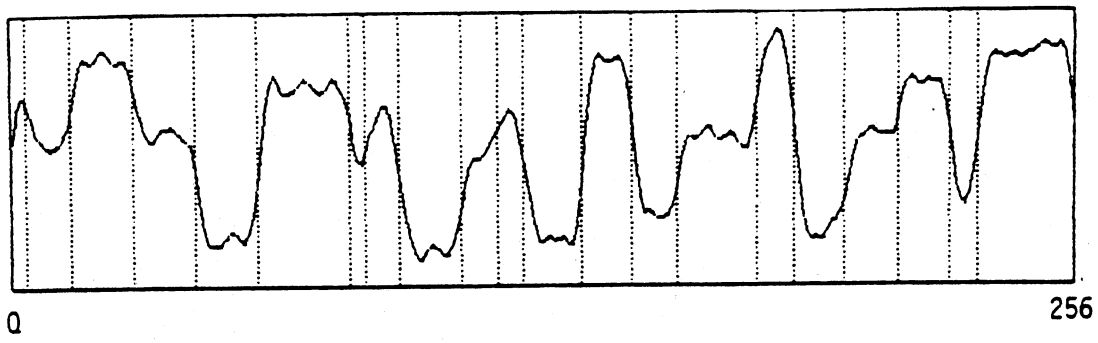
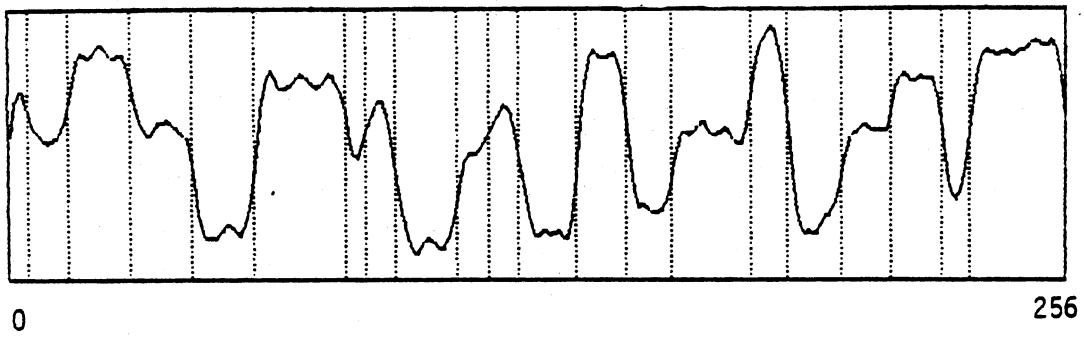


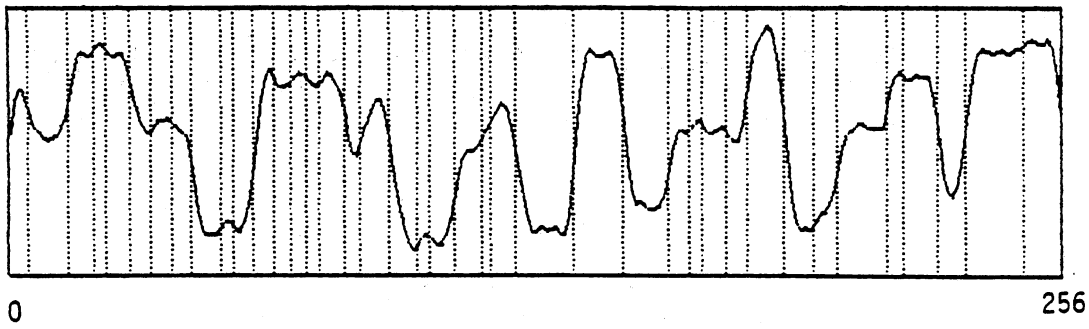
Figure 26. Activity Curve Automatic Segmentation



(a)



(b)



(c)

Figure 27. Automatic Segmentation Adjustment

curve parameters used were those used to obtain the segmentation shown in Figure 27(b). Candidates consisting of N_S - 2, ..., N_S + 2 beds were considered, where N_S is the number of signature beds as determined by activity curve segmentation. Both Itakura's and Sakoe and Chiba's dynamic programming warping methods were tried; for each method, both L1 (absolute value of differences) and L2 (square of differences) were used. The results are shown in Table I. These results suggest that as far as the "percent correct" figures are concerned, there is no advantage to using the L1 distance measure. (The L1 distance measure is known to be more robust in the presence of impulsive noise, which is not what is being dealt with here.) The results also show that although Sakoe and Chiba's dynamic programming warping method has some theoretical advantages, it did not perform significantly better than Itakura's method in terms of the precent correct figures. Furthermore, Itakura's method (as implemented here) has a clear advantage in terms of CPU time requirements. The most important observation is that the automatic segmentation scheme drastically reduced the CPU time requirement. Itakura's method without the segmentation speedup technique required an average of 321 seconds per search, as noted earlier. (This is for the case where the window size is varied from N/2 to 2N, where N is the number of points in the given signature; the incremental changes for window position and length are both set to 3.) When the segmentation method is in place, the average CPU time drops to 8 seconds.

These results are encouraging, but it must be emphasized that in the "real world" a method for which good results depend on parameter twiddling is not practical. A technique for automatically selecting the activity curve segmentation parameters is needed. One possibility that

TABLE I

EXPERIMENTAL RESULTS: DPW WITH AUTOMATIC SEGMENTATION

Speedup Technique/ Warping Method/ Distance Measure	Number of Problems	Average CPU Time (sec)	Percent Correct (Fit > 0.7)
none/It/L2	5	321.	100
none/SC/L2	5	1609.	100
seg/SC/L1	100	34.	92
seg/SC/L2	100	41.	94
seg/It/L1	100	8.	92
seg/It/L2	100	8.	89

note on abbreviations:

It -- Itakura's method
SC -- Sakoe and Chiba's method
seg -- automatic segmentation

has been investigated in this work is based on the idea that if the number of beds in the signature is known, the segmentation parameters can be selected by iteratively adjusting one or both of them until the algorithm segments the signature into the correct number of beds. (A restriction is also imposed on the minimum bed width.) The resulting parameters can then be used to segment the log being searched. There are two activity curve parameters to consider; it was decided to fix the window width and adjust the threshold. An initial guess for the threshold must be provided, along with an amount (dT) with which to initially increment or decrement the threshold. In the "real world" one would probably choose simple initial values; threshold = 1 and dT = 0.2 were selected here. A segmentation "fudge factor" is still allowed in the search: If NSB is the number of signature beds specified, the search is over NSB ± 2 segments. The experimental results based on 100 randomly generated signature search problems were as follows: percent correct = 72, and average CPU time = 8.7. (Itakura's warping method with L2 distance measure was used.) It is not surprising that the percent correct figure is lower than when parameter twiddling is allowed. It is interesting to note that the CPU time requirement for automatic parameter adjustment is not severe (the average CPU time for fixed parameters was 8 seconds).

At this point it should be interesting to consider some experimental results with real data. The logs selected for this purpose are the gamma ray logs shown in Chapter I (Figure 4). As noted there, the sections marked off by solid horizontal lines represent the same rock formation. For these experiments, the section on log 2 was chosen as the signature; log 1 was searched for this signature.

It was noted earlier that level shifts may present a problem with real log data. Average value subtraction from the signature and candidate sequences is a possible solution. However, as noted in Chapter I, average value is not warp invariant; consequently, there may be cases when average value subtraction does more harm than good. A possible alternative to average value subtraction is a process named "high low matching" in this work. High low matching adds a constant value to a candidate sequence X(n) based on the maximum and minimum values of the candidate sequence and the signature sequence S(n). Since these maximum and minimum values are ideally warp invariant, high low matching is approximately warp invariant. The constant C to be added to the candidate X(n) is determined as follows: let S_{max} and X_{max} denote the maximum values of S(n) and X(n), respectively. Let S_{min} and X_{min} denote the minimum values of S(n) and X(n), respectively. Then

$$C = (1/2)\ (S_{max} - X_{max}) + (1/2)(S_{min} - X_{min}) \tag{44}$$

Observe that if X(n) is a warped and level shifted version of S(n), i.e.,

$$X(n) = S[W(n)] + a \tag{45}$$

then $X_{max} = S_{max} + a$ and $X_{min} = S_{min} + a$. In this case, the constant C to be added to X(n) by high low matching is

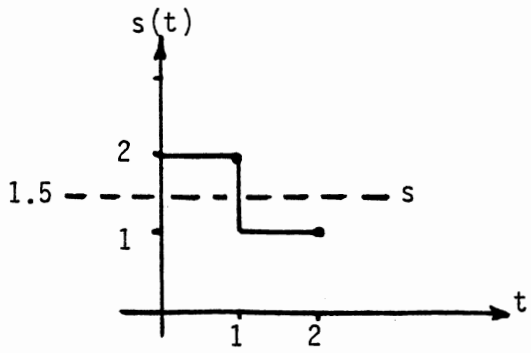
$$C = (1/2)(S_{max} - S_{max} - a) + (1/2)(S_{min} - S_{min} - a) = -a \tag{46}$$

In other words, if X(n) is a warped and level shifted version of S(n), high low matching does exactly what is desired: it removes the level shift.

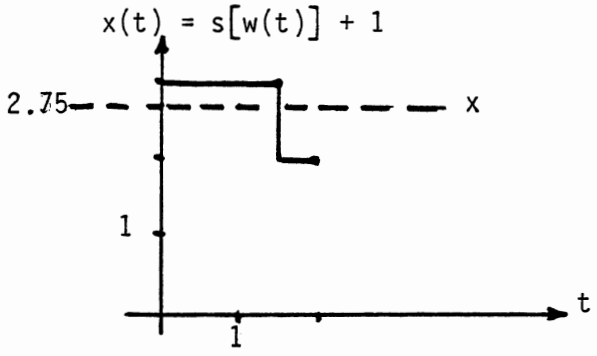
Figure 28 shows a simple example of where average value subtraction fails to accomplish what is needed. Figure 28(a) shows the original signature s(t); Figure 28(b) shows a warped and level shifted version x(t) = s[w(t)] + 1. The dotted lines show the average values. Figures 28(c) and 28(d) show the result of average value subtraction; note that the waveforms are still level shifted with respect to each other. Figures 28(e) and 28(f) show the result obtained with high low matching.

Figures 29(a) thru 29(f) show the results obtained on the real data using log segmentation (with the activity curve threshold automatically selected) in conjunction with dynamic programming warping (Itakura's method with an L2 distance measure). In each case, the dashed horizontal lines show the segmentation obtained by means of the activity curve. Recall that the procedure is first to adjust the threshold until the signature (on log 2) is segmented into the desired number of beds, and then to use the same threshold to segment the log to be searched (log 1). In each case, the answer chosen by the algorithm is indicated by the vertical arrow to the left of log 1, along with the resulting fit measurement.

It is assumed that the number of beds in the signature can be specified. Unfortunately, this is not necessarily an easy thing to do; the number of beds assigned to the signature on log 1 is not really obvious. (In the simulated random problems it is known that there are always 4 beds in the signature.) Therefore, two sets of experiments were run: one where the number of beds in the signature was set to 4, and another where the number of beds was set to 12. For each set, the algorithm was run (1) with average value subtraction; (2) with high low matching; and (3) with no level shift compensation attempted. High low



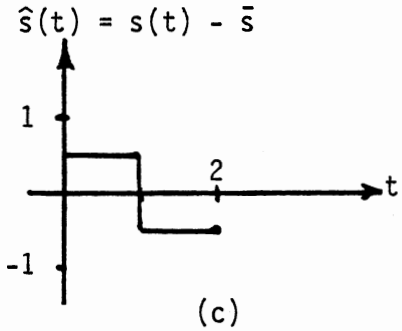
(a)



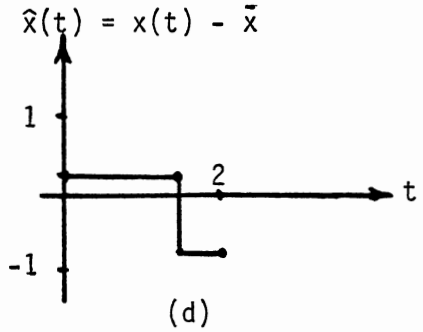
(b)

$$\bar{s} = \tfrac{1}{2}\int_0^2 s(t)dt = 1.5$$

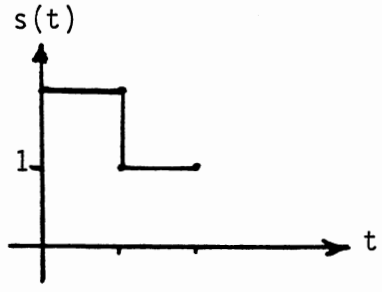
$$\bar{x} = \tfrac{1}{2}\int_0^2 x(t)dt = 2.75$$



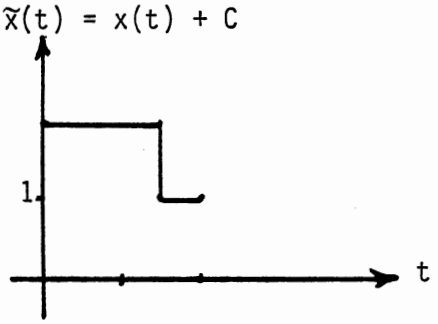
(c)



(d)



(e)



(f)

$$C = \tfrac{1}{2}(2 - 3) + \tfrac{1}{2}(1 - 2) = -1$$

Figure 28. Average Value Subtraction and High Low Matching

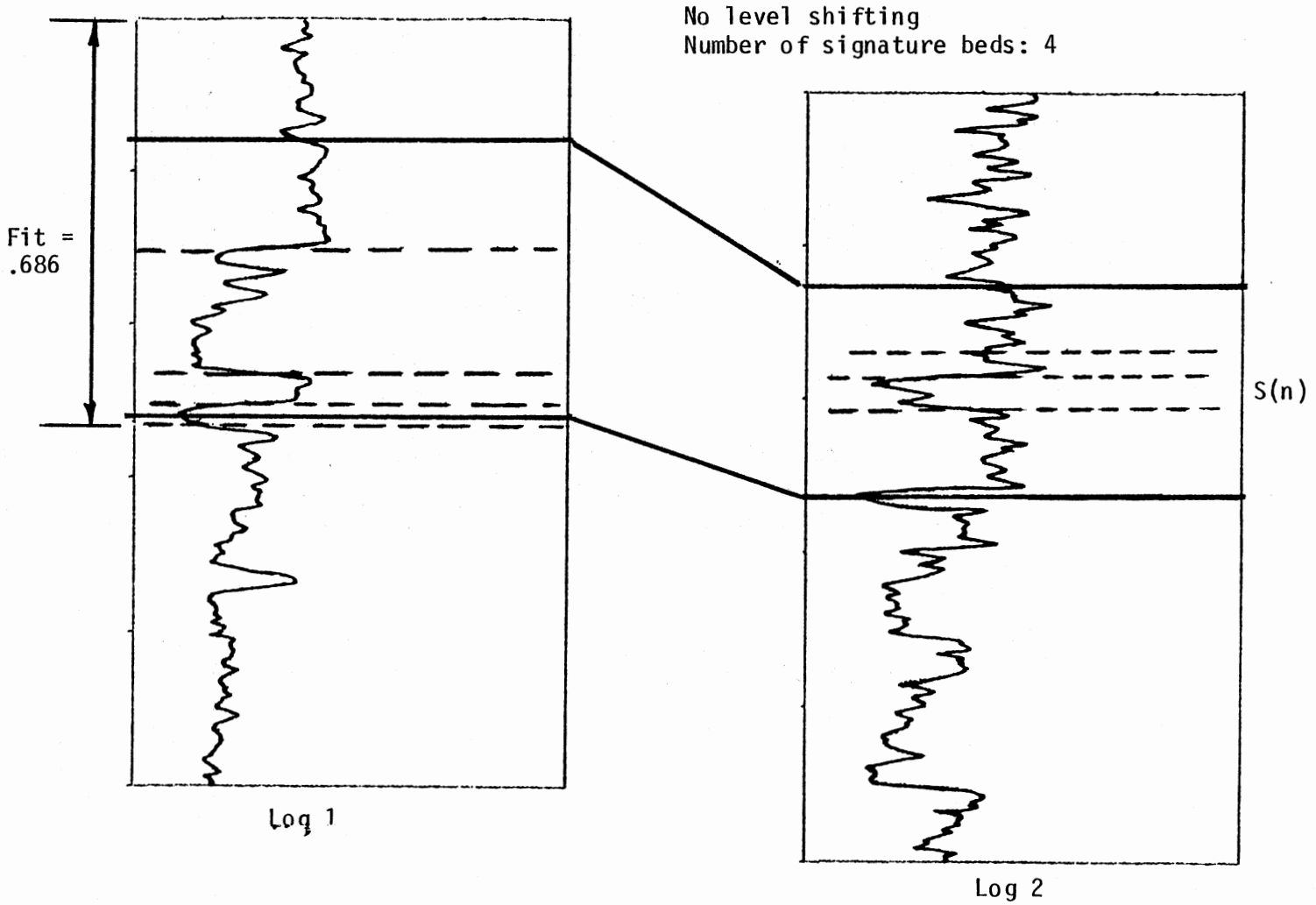


Figure 29(a). Signature Recognition Using Dynamic Programming Warping and Automatic Segmentation on Gamma Ray Logs

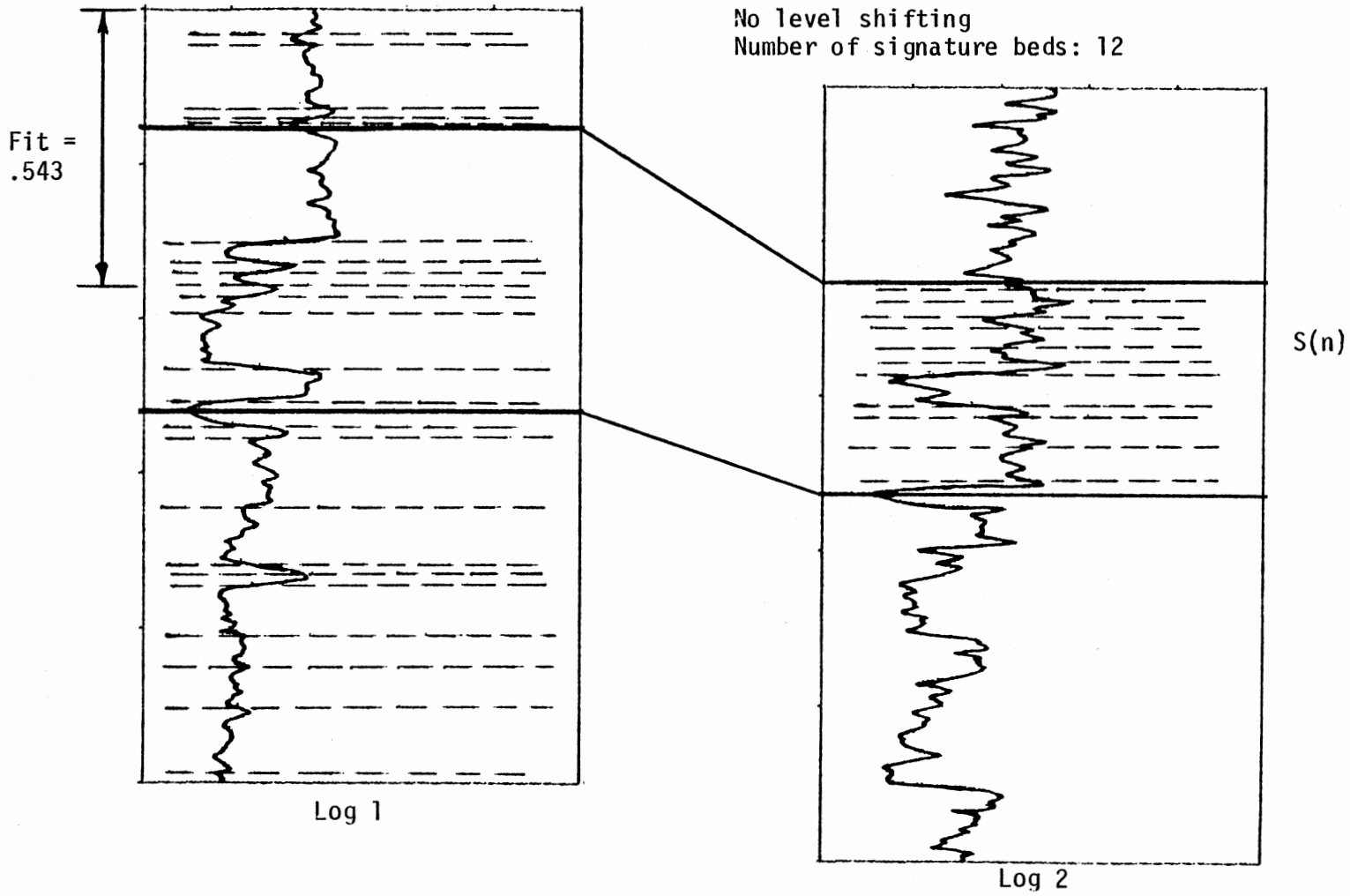


Figure 29(b). Continued

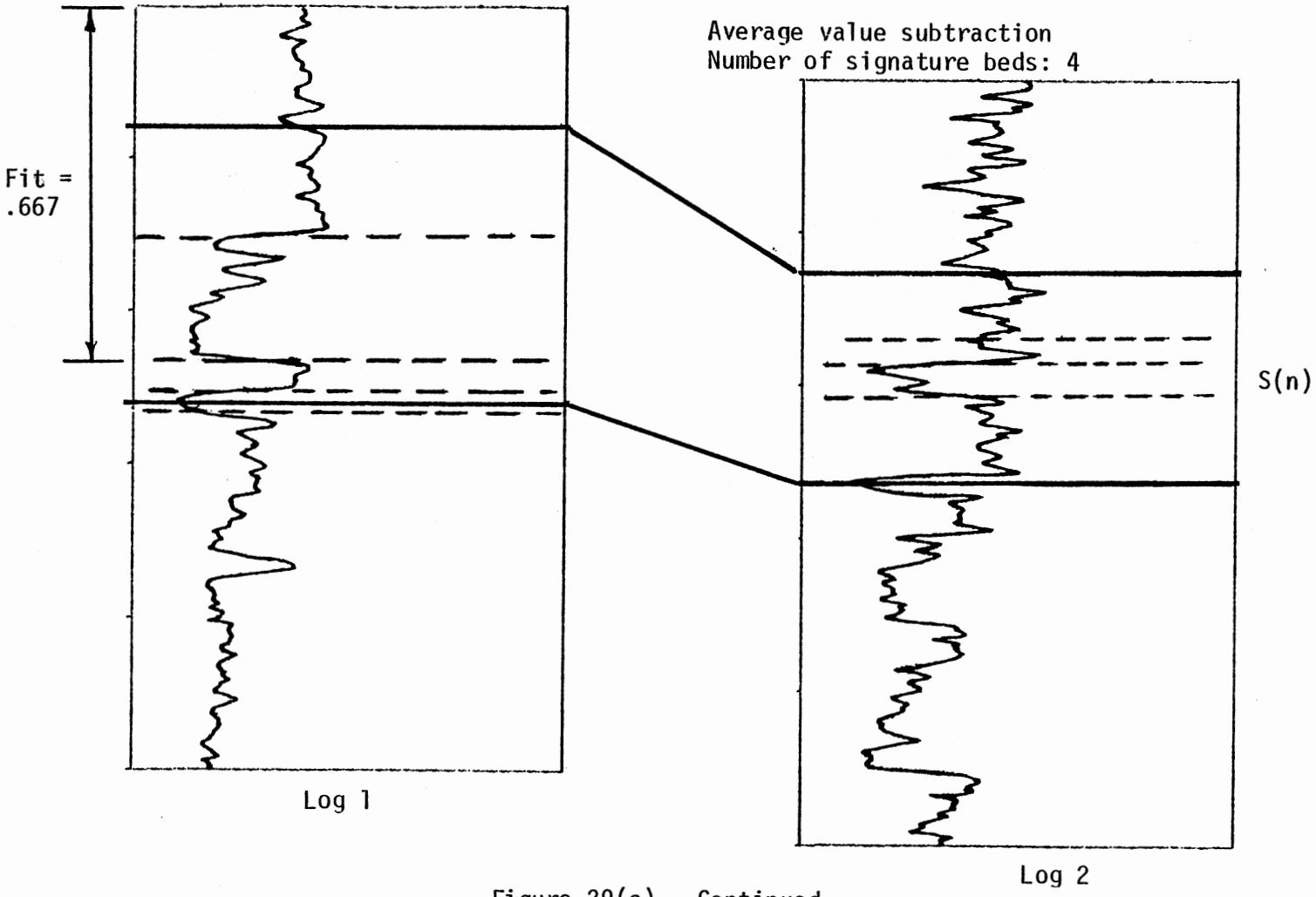


Figure 29(c). Continued

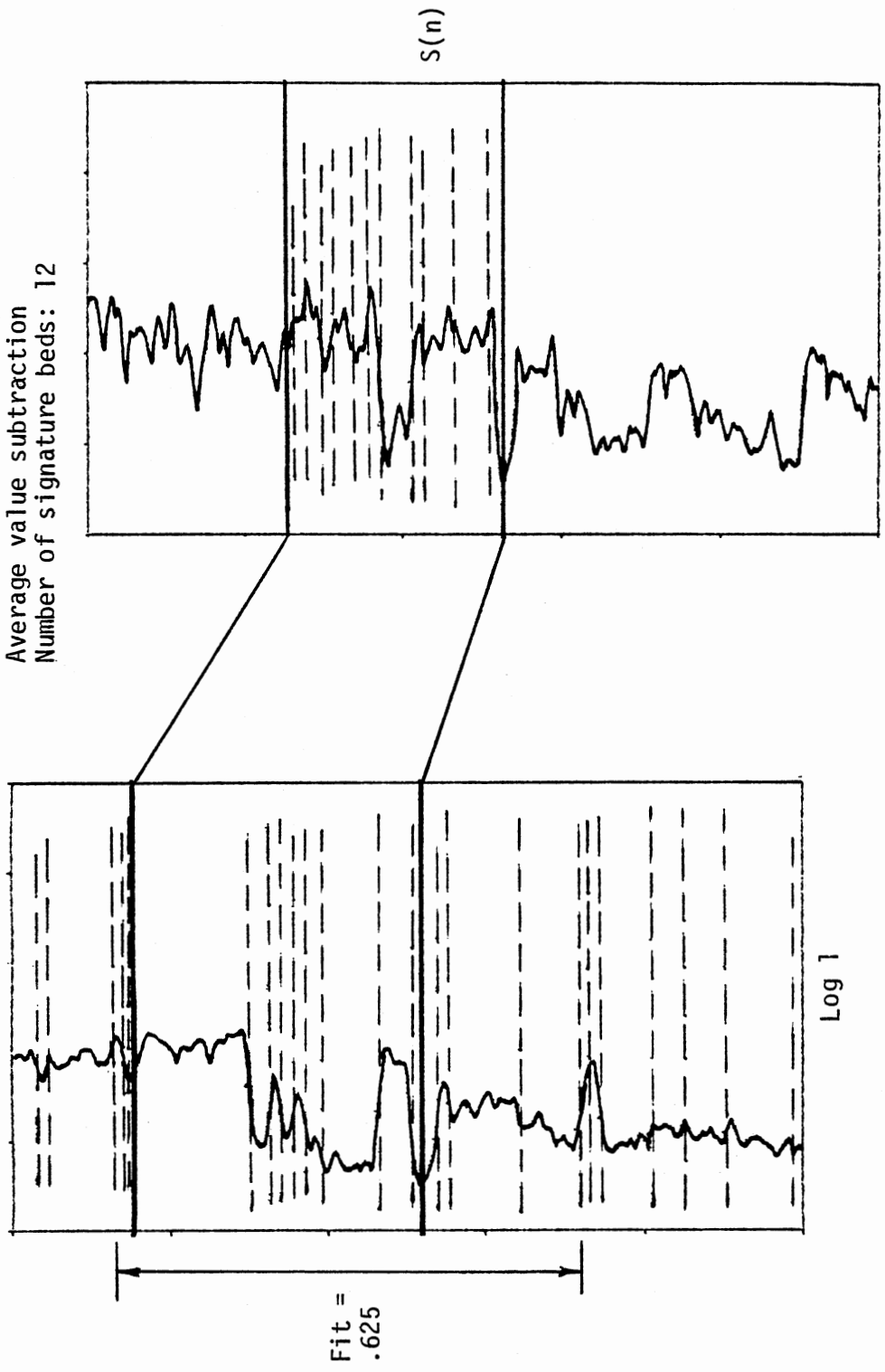


Figure 29(d) Continued

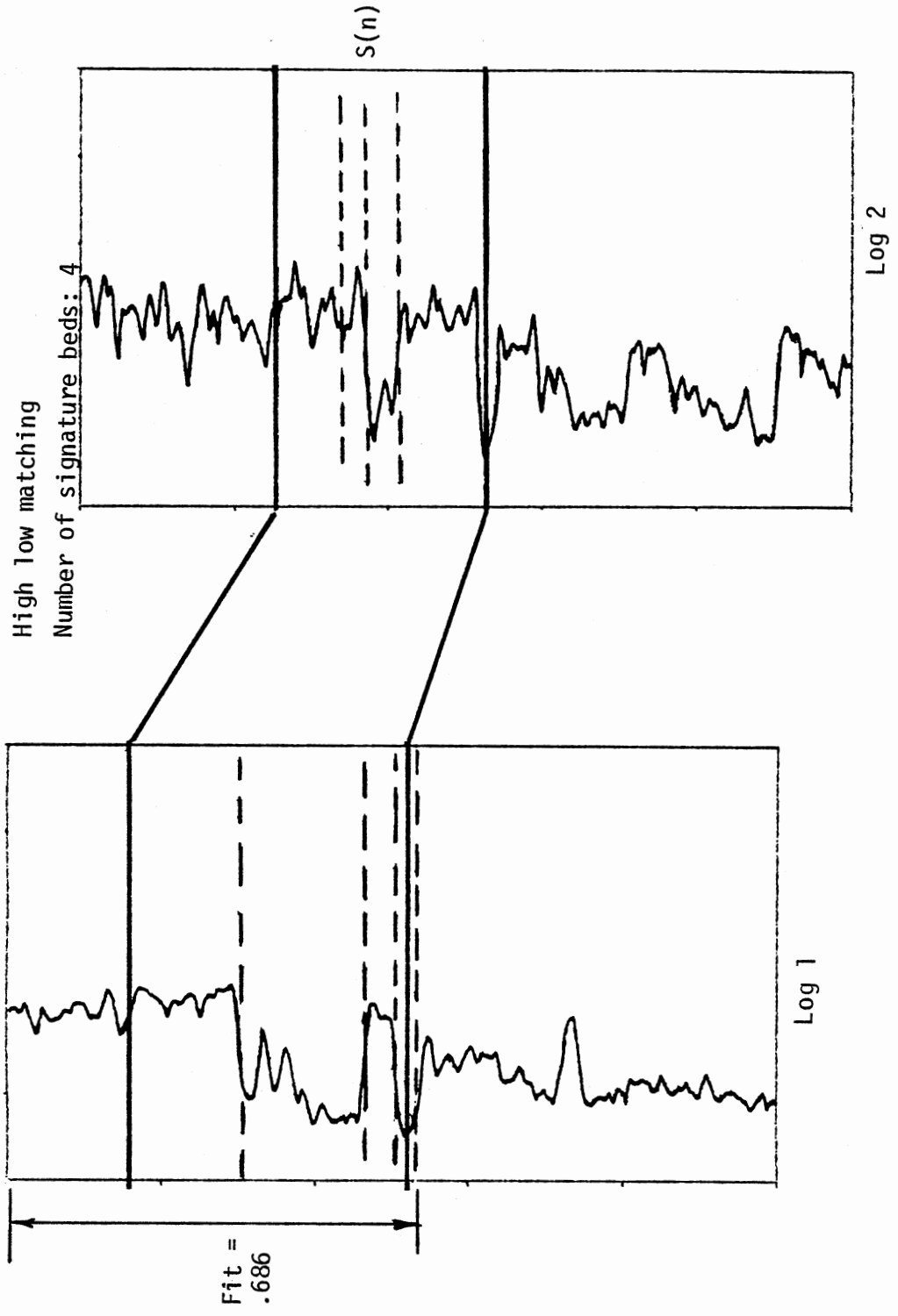


Figure 29(e). Continued

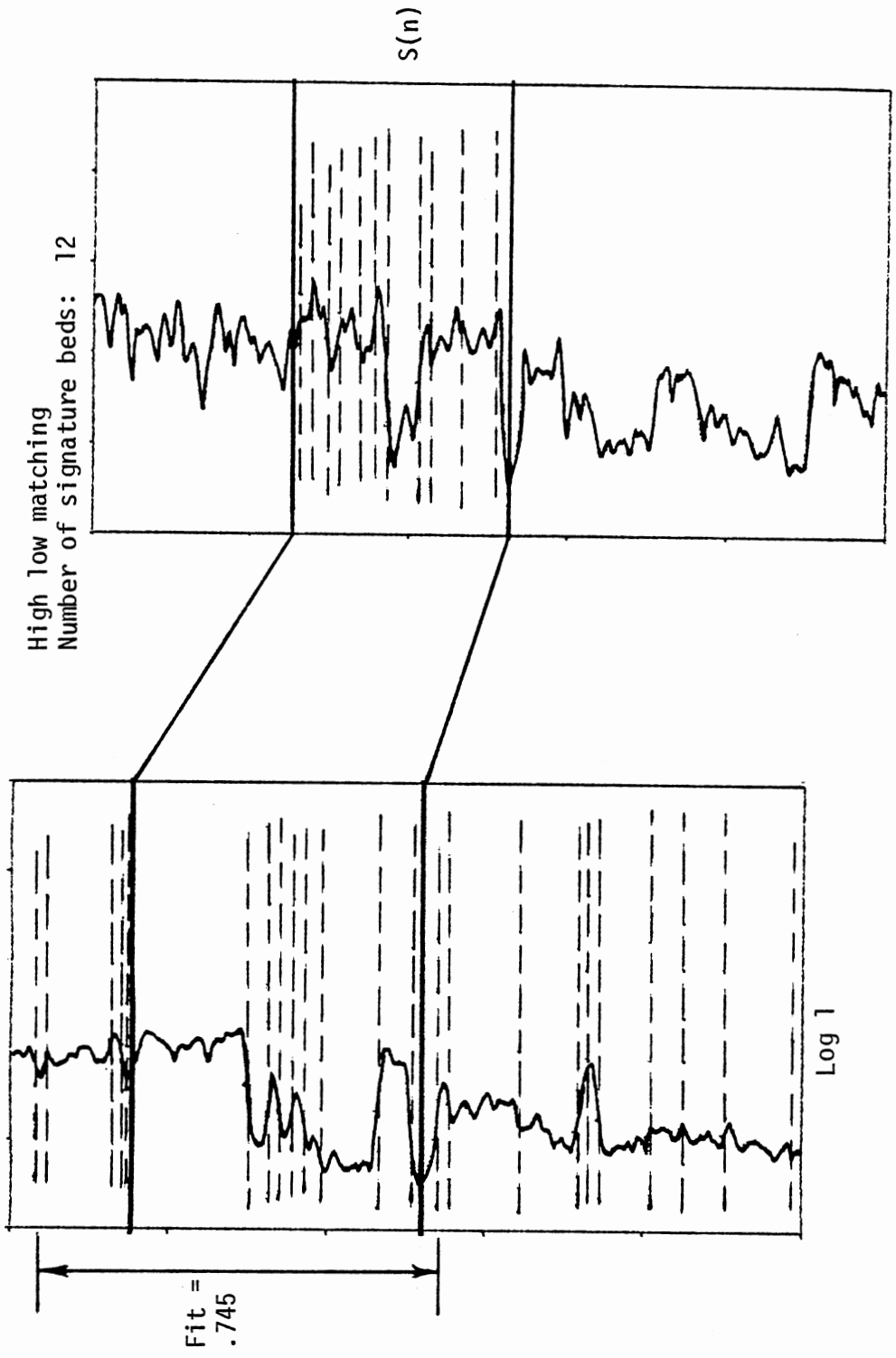


Figure 29(f). Continued

matching resulted in reasonable answers for both 4 and 12 signature bed assumptions. The fit measurements are not impressive, but it should be noted that the top boundary of the signature is not very well defined in this example.

The next section considers the second speedup technique mentioned in the introduction to this chapter: data reduction by means of the sample rate adjustment algorithm. Experimental results obtained using this method on simulated problems and real well log data are presented.

Data Reduction With Sample Rate Adjustment (SRA)

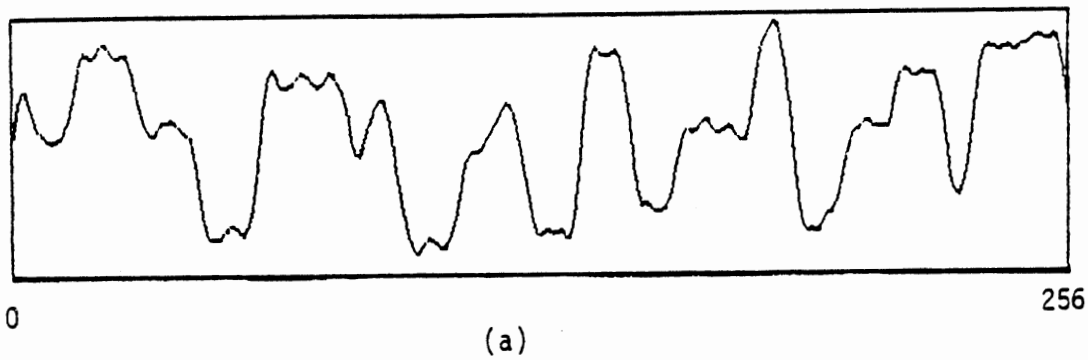
The sample rate adjustment algorithm is explained in detail in Chapter IV, where it is viewed as a nonlinear prewarping filter to be used in conjunction with direct template matching. However, it was originally intended to be a data reduction algorithm to be used in conjunction with signature recognition based on dynamic programming warping. The basic idea of the sample rate adjustment (SRA) algorithm is to reduce the number of sample points on sections of a waveform that are relatively "flat" in shape. Sample rate adjustment is a warping process that squeezes only. Since it is a warping process, there is ideally no loss of information in terms of a "class" of signature waveforms. By reducing the number of data points in both the signature and the log to be searched, the number of candidate sequences is reduced. (The mapping of points from the original log being searched to the SRA reduced version must be kept track of so that the solution found on the SRA reduced version can be translated to a solution on the original log.) It should also be observed that since the number of points in the dynamic programming constraint region is approximately proportional to

NxM (where N and M are the number of points in the sequences being matched), the two sequences can be warped to fit faster if the number of points they have is first reduced.

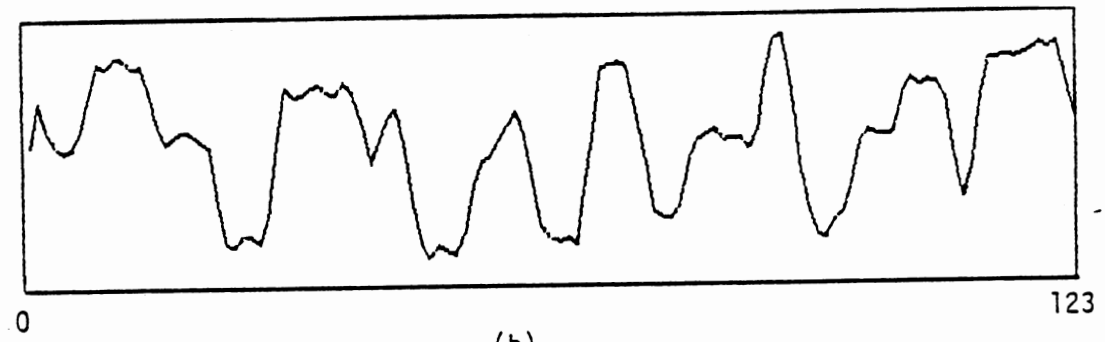
The decision whether or not to delete a point X(i) from a sequence is based on considering the change DX = |X(i) - X(j)|, where X(j) is the most recent point that was not deleted. If DX is less than the specified threshold, then X(i) is deleted. A restriction on the number of points in a row allowed to be deleted (denoted NSKIP) is also included.

Although the purpose of sample rate adjustment here is not data compression per se, it is interesting to briefly view it in that light. Let X(n), n=1,2,...,N be a data sequence. If each point in the sequence is represented by K bits, there are a total of NK bits required to represent the sequence. Suppose there exists a method to approximately reconstruct X(n) from its SRA-reduced version Y(n), n=1,2,...,M (M<N). Such a reconstruction method would require a mapping sequence Z(n), n=1,2,...,N, consisting of ones and zeros only (that is, each point in this sequence is represented by one bit). If X(i) is discarded by the SRA algorithm, Z(i) = 0; otherwise, Z(i) = 1. The total number of bits required to represent the SRA-reduced sequence and the mapping sequence is MK+N. Thus the ratio of the number of bits after compression to number of bits before compression is r = M/N + 1/K.

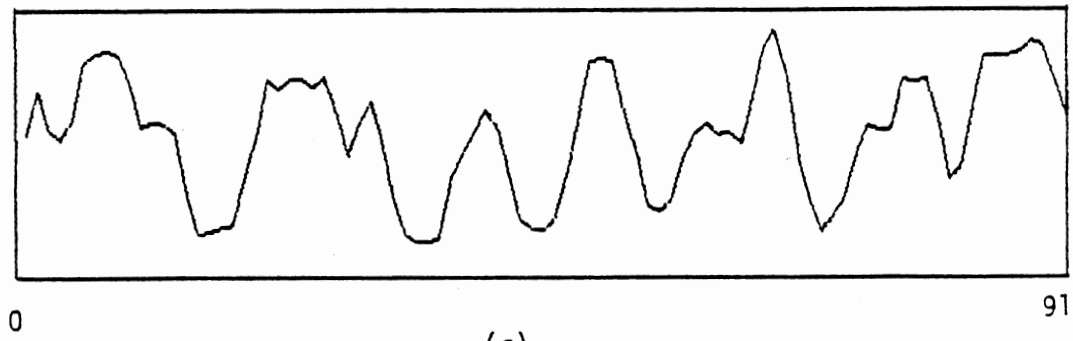
Figure 30 shows examples of how SRA affects the randomly generated logs. Figure 30(a) shows the original log of 256 points. Figure 30(b) shows how the original log is reduced to 123 points using SRA parameters threshold = 2 and NSKIP = 2. Figure 30(c) shows how the original log is reduced to 91 points with SRA parameters threshold = 2 and NSKIP = 3. Note the different horizontal scales on these figures.



(a)



(b)



(c)

Figure 30. Effect of Sample Rate Adjustment

Table II summarizes the experimental results based on 100 random signature search problems. Again it should be noted that a solution is judged to be correct if the fit measurement (Figure 14) is greater than 0.7. The sample rate adjustment parameters used were threshold = 2 and NSKIP = 2. The window width increment and slide increment were each 3 points, and the window width was varied from half the signature length to twice the signature length. (The length referred to is that of the signature after SRA was applied.) Both Itakura's and Sakoe and Chiba's dynamic programming warping algorithms were tried; for each method, both L1 and L2 distance measures were used. These results suggest that there may be an advantage to the L1 distance measure for these problems. However, this is in contrast to the results shown in Table I, which are inconclusive as far as this issue is concerned. Sakoe and Chiba's dynamic programming warping method gives slightly better percent correct figures than Itakura's method, but is significantly slower. The main result of interest is that the search using data reduction by means of SRA is significantly faster than the search without a speedup technique.

Table III compares the results using (1) no speedup technique, (2) segmentation with parameter "twiddling", (3) segmentation with the activity curve threshold automatically adjusted, and (4) sample rate adjustment. In Table III, all results are for Itakura's warping method with the L2 distance measure. The results show that either speedup technique turns an impractical signature search method into a practical one as far as CPU time requirements are concerned.

It is interesting to once again consider some experimental results with real well log data. Figure 31 shows gamma ray logs from two bore-

TABLE II

EXPERIMENTAL RESULTS: DPW WITH SRA DATA REDUCTION

Warping Method	Distance Measure	Average CPU Time (sec)	Percent Correct (Fit > 0.7)
Itakura	L1	16.	76
Itakura	L2	16.	66
Sakoe and Chiba	L1	58.	79
Sakoe and Chiba	L2	72.	74
		Results without SRA	
Itakura	L2	321.	100*
Sakoe and Chiba	L2	1609.	100*

*based on 5 problems

TABLE III

SPEEDUP TECHNIQUE SUMMARY

Speedup Technique	Average CPU Time (sec)	Percent Correct (Fit > 0.7)
none	321.	100
Segmentation (parameter twiddling)	8.0	92
Segmentation (auto. thresh. adjustment)	8.7	72
SRA	16.0	66

holes. The signature on the left log (as indicated on Figure 31) was used as the given signature. The right side log was searched for this signature; the correct location, as given by those who supplied this data, is also indicated on Figure 31.

The original data has approximately one sample every 0.15 meters. Thus the original signature has approximately 3300 points, and the log to be searched has over 10000 points. This is much more data than the search methods can handle. (The warping subroutine uses arrays of dimension (N,M), where N is the number of points in the signature and M is the number of points in the largest possible candidate. Even if arrays of this size could be used, the amount of CPU time required for the search would be enormous.) To overcome this problem, each log was processed by a digital lowpass filter with a sampling frequency to cutoff frequency ratio of 32 to 1, and then decimated by a factor of 16 (i.e., only every 16th point was saved). The logs shown in Figure 31 have been filtered and decimated. (The lowpass filter is an infinite impulse response (IIR) type based on a 3rd order Butterworth prototype.)

By inspection of Figure 31 it is clear that the signatures on the two logs are level shifted with respect to each other. Therefore, it should come as no surprise that the search algorithm required a level shifting technique to obtain the correct answer. For the result shown on Figure 31, the average value of the signature and of each candidate was subtracted before calling the warping subroutine. With this level shifting, the search algorithm chose the section indicated on Figure 31 by dotted lines, with an excellent fit of 0.92.

The search algorithm for the above example used sample rate adjustment with the same window width and slide increments and the same

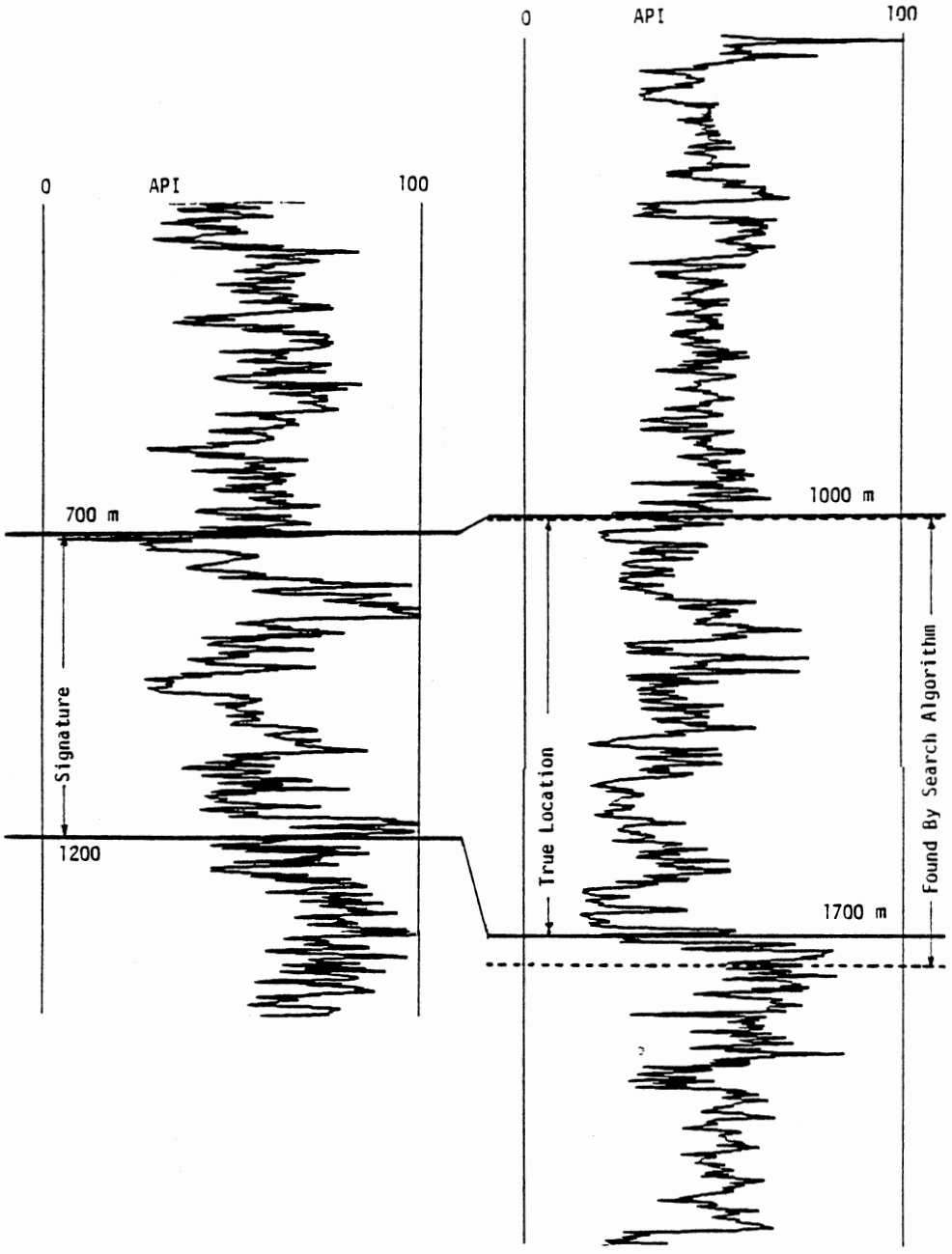


Figure 31. Signature Recognition Using Dynamic Programming Warping and SRA for Data Reduction on Gamma Ray Logs

SRA parameters as used in the random log experiments. Itakura's warping technique with the L2 distance measure was used.

Another real example using sample rate adjustment for data reduction is presented next. Figure 32 show the same gamma ray logs illustrated by Figure 4; the true location of the signatures is indicated by solid horizontal lines. Log 1 was searched for the signature indicated on log 2. Inspection of Figure 32 reveals that the signatures are level shifted with respect to each other; therefore, high-low matching was employed during the search.

The vertical arrow marked "A" to the left of log 1 shows the signature location determined by the search algorithm using SRA parameters threshold = 8 and NSKIP = 3. The vertical arrow marked "B" shows the location selected with SRA parameters threshold = 3 and NSKIP = 1. Both results are good, but it seems clear that good results can depend on careful selection of the SRA parameters. It is interesting to note that case A, where the SRA parameters allow more severe warping, is a better result than case B. Perhaps this is because a more severe version of SRA can make a signature and its warped versions "look more like" each other. This is an important idea in the next chapter. (Let it be noted in passing that if a prewarping algorithm makes warped versions of a sequence "look more like" each other, then average value subtraction becomes more suitable as a level shifting technique.)

Figure 33 serves as a sobering reminder of how things can go wrong. It was a matter of interest to see what would happen if the problem was reversed, i.e., what would happen if log 2 was searched for the signature indicated on log 1. The vertical arrow marked "A" shows a result obtained using average value subtraction; the fit is zero.

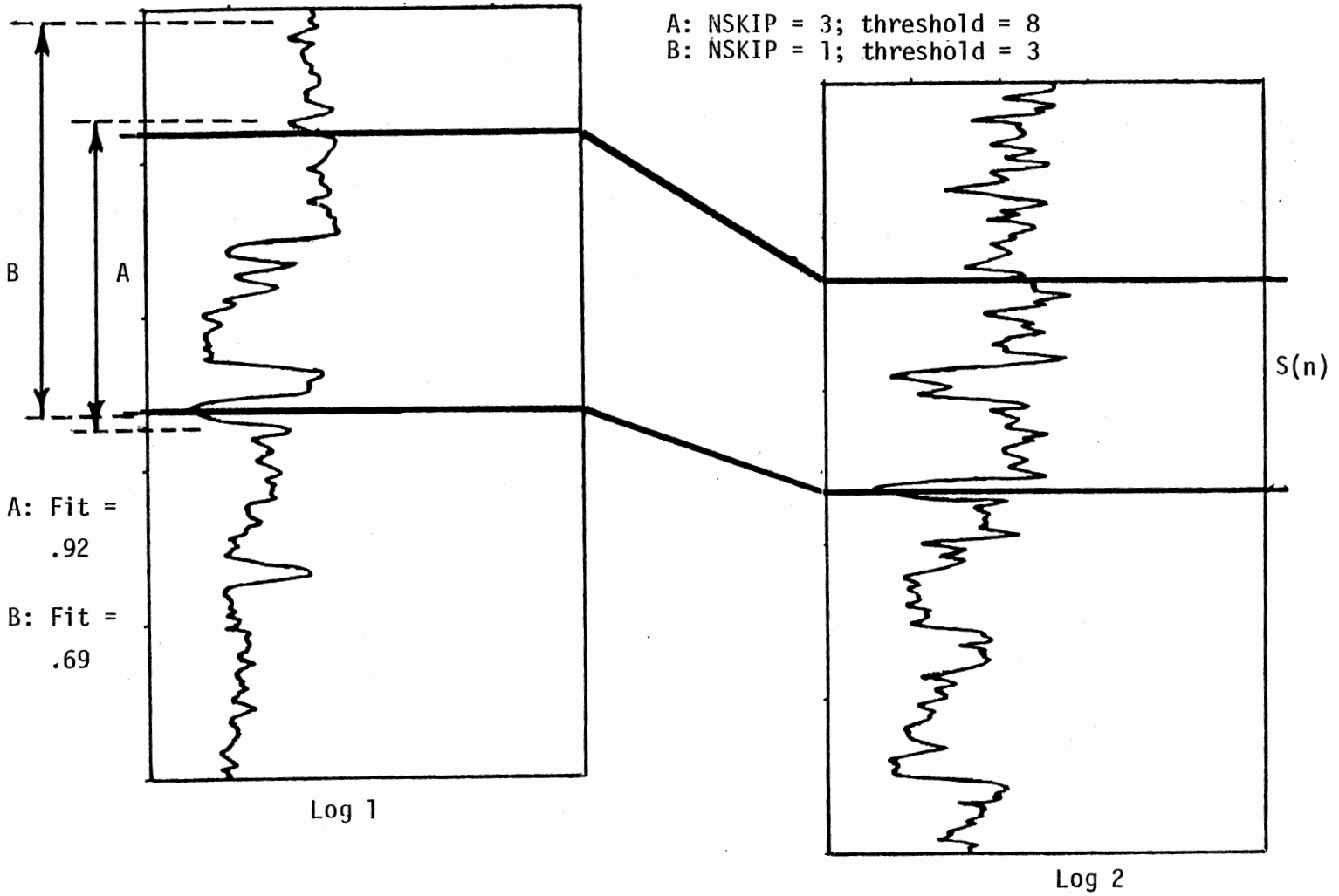


Figure 32. Signature Recognition Using Dynamic Programming Warping and SRA for Data Reduction on Gamma Ray Logs.

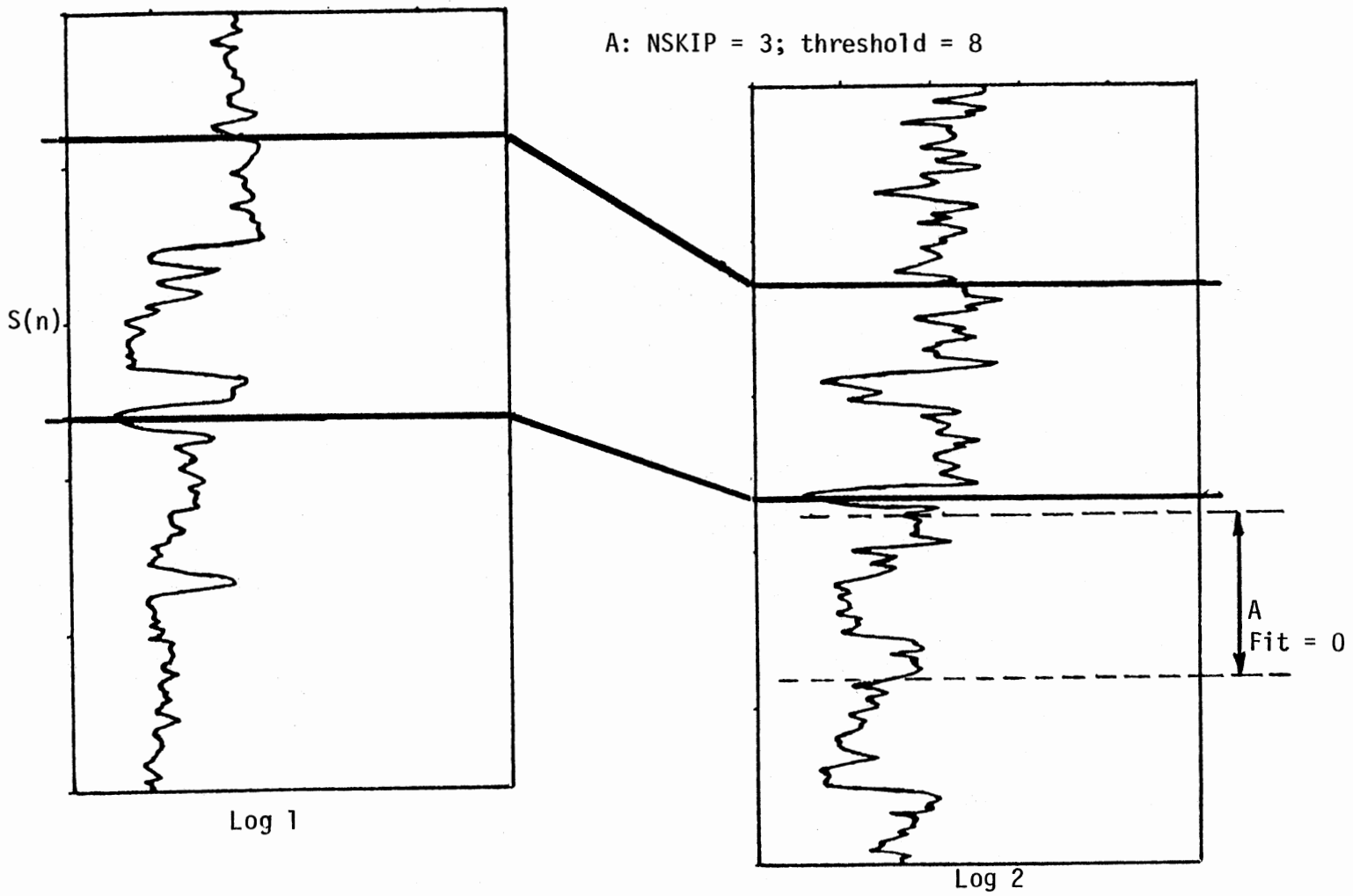


Figure 33. Signature Recognition Using Dynamic Programming Warping and SRA for Data Reduction on Gamma Ray Logs

(Note: high low matching also failed here.) The interesting thing about this case is that a careful inspection of the actual shapes involved shows that answer "A" is not really a bad answer from an abstract pattern recognition point of view. The random log generator used to simulate signature recognition problems will sometimes create similar situations. An example of this will be shown in the next chapter.

Dynamic programming warping in conjunction with speedup techniques (segmentation or data reduction) has been shown in this chapter to be a useful approach to the well log signature recognition problem. However, since the dynamic programming warping algorithm is still a computational bottleneck, there is a clear motive for considering alternate solutions. Alternatives to the dynamic programming warping approach are the subject of the next chapter.

CHAPTER IV

ALTERNATIVES TO THE DYNAMIC PROGRAMMING WARPING APPROACH

Introduction

This chapter considers alternatives to the dynamic programming warping based solution to the well log signature recognition problem. Perhaps the main motivation for considering such alternatives is to bypass the computational bottleneck created by the dynamic programming warping algorithm. These alternatives are based on the use of nonlinear prewarping filters which tend to improve the matchup of the waveforms in a signature class. Two such filters are considered: sample rate adjustment (SRA) and straight line prediction filtering (SLPF). Both algorithms can be thought of as ways to create a feature vector by selecting "significant" points from a sequence. The idea is that the set of "significant" points for sequences which are warped versions of each other should be similar.

Signature recognition based on these prewarping filters (which can also be described as nonuniform decimation algorithms) can be roughly divided into two categories: (a) direct template matching, which in this context means directly comparing the candidate with the signature after the prewarping is done, and (b) statistical pattern recognition techniques. Statistical pattern recognition methods depend on the existence of a training set for the signature class; a method of

artificially creating such a training set, called "on the job training" (OJT), is presented in this chapter.

Statistical pattern recognition techniques based on Euclidean distance from the class centroid, clustering transformations, Mahalanobis distance, and estimation of the class probability density function as a weighted sum of orthonormal functions are considered. An OJT based signature recognition scheme using singular value decomposition is also described. Statistical pattern recognition techniques are made more attractive by reducing the dimension of the pattern vectors; a method of accomplishing this with the Walsh transform is given.

A method of using on the job training to automatically select the parameters for the prewarping filters is also considered in this chapter. This is an important application of OJT because a requirement for subjective parameter "twiddling" would render prewarping filtering almost useless in the "real world."

Experimental results based on both computer simulations and real well log data are presented in this chapter.

Prewarping and Direct Template Matching

Ideally, it would be desirable to find a way to extract a warp invariant feature vector from a waveform. Such a feature vector, when extracted from a well log signature, could then be used as a template with which to directly compare the feature vectors extracted from the candidate sequences. Nonlinear prewarping filters can be thought of as a step in this direction. Both prewarping filters discussed herein can be thought of as ways to create a feature vector by selecting

"significant" points from a sequence. The idea is that the set of "significant" points from two warped versions of a sequence should be similar.

Suppose X(n) is a warped version of S(n). If so, then there ought to exist warping functions $W_X(s)$ and $W_S(n)$ such that

$$X[W_X(n)] = S[W_S(n)] \tag{47}$$

Both prewarping filters (straight line prediction filtering (SLPF) and sample rate adjustment (SRA)) represent attempts to find an operator that finds $W_X(n)$ by considering X(n) only, and attempts to find $W_S(n)$ by considering S(n) only. In other words, they represent attempts to find an operator which, when applied to both S(n) and X(n), causes the resulting sequences to match. Both SLPF and SRA could loosely be described as "adaptive" warping filters since the warping applied to a sequence depends on the sequence.

Before discussing the details of straight line prediction filtering and sample rate adjustment, it is important to consider how these prewarping filters are used in the signature recognition problem depicted in Figure 1. The procedure is as follows:

(1) Apply SLPF or SRA to the signature and to the log being searched. The mapping of points from the log being searched to the filtered version is kept track of in a correspondence array so that the solution found on the filtered log can be translated to a solution on the original log.

(2) Slide the search window across the filtered log and select candidates. It is desired to compare filtered signature and candidate using direct template matching, but to do this both sequences must have

the same length. This is accomplished by linear interpolation, which could be applied to one or both depending on the exact scheme being used. The "direct template matching" used here is the normalized sum of squared differences for the two sequences ("normalize" meaning in this case dividing the sum by the number of points in the sequence).

A brief explanation of what is meant by "linear interpolation" is in order. Suppose $X(n)$, $n = 0,1,\ldots,N-1$ is given, and a new sequence $\hat{X}(m)$, $m = 0,1,\ldots,M-1$ is to be created by linear interpolation. This operation is performed by the following relationship:

$$\hat{X}(m) = X(n) + [X(n+1) - X(n)] \left[\frac{m(N-1)}{M-1} - n\right] \tag{48}$$

where

$$n = \left\lfloor \frac{m(N-1)}{M-1} \right\rfloor \tag{49}$$

Equation (48) can be derived from the standard two point formula for a straight line. $\hat{X}(m)$ is estimated from a straight line fit to $X(n)$ and $X(N+1)$. The notation $\lfloor . \rfloor$ means "round down to the earnest integer." $\hat{X}(m)$ can be thought of as a "stretched" ($M>N$) or "squeezed" ($M<N$) version of $X(n)$.

Let us now turn to a discussion of the details of sample rate adjustment and straight line prediction filtering.

Straight Line Prediction Filtering (SLPF) and Sample Rate Adjustment (SRA)

The details of straight line prediction filtering (SLPF) will be considered first. The explanation that follows is for a SLPF of length 3, i.e., the prediction is based on a straight line fit to 3 data

points. In general, any length greater than or equal to 2 could be used, and in fact the SLPF subroutine used in this work has length as a variable parameter.

Let X(n), n = 0,1,... be the input to the filter. The output sequence is formed by deleting points from the input sequence. This is a form of warping that allows "squeezing" only. The basic idea is to squeeze parts of X(n) that are good fits to straight lines.

The first thing the SLPF algorithm does is fit a straight line to X(0), X(1), and X(2). See Figure 34. The straight line is then used to predict X(3). Let Z(3) be the prediction. The prediction error is

$$E = |X(3) - Z(3)| \tag{50}$$

If E is less than some specified threshold, then X(3) is deleted, i.e., it does not appear as part of the output. (The stored values in the input array are not changed.) Suppose E is less than the threshold. The next step is to predict X(4) using the same straight line, and calculate E = |X(4) - Z(4)|. The threshold test is applied again. If E is small enough, X(4) is deleted. This process is continued until one of two things happens:

(1) The maximum number of points in a row allowed to be deleted (denoted NSKIP) is attained.

(2) The threshold is exceeded.

Suppose one of these conditions occurs when X(i) is predicted. X(i) is then included in the filter output and a new straight line predictor is fitted to X(i-2), X(i-1), and X(i). The next point to be predicted is X(i+1). The counter for number of points in a row deleted is reset to zero.

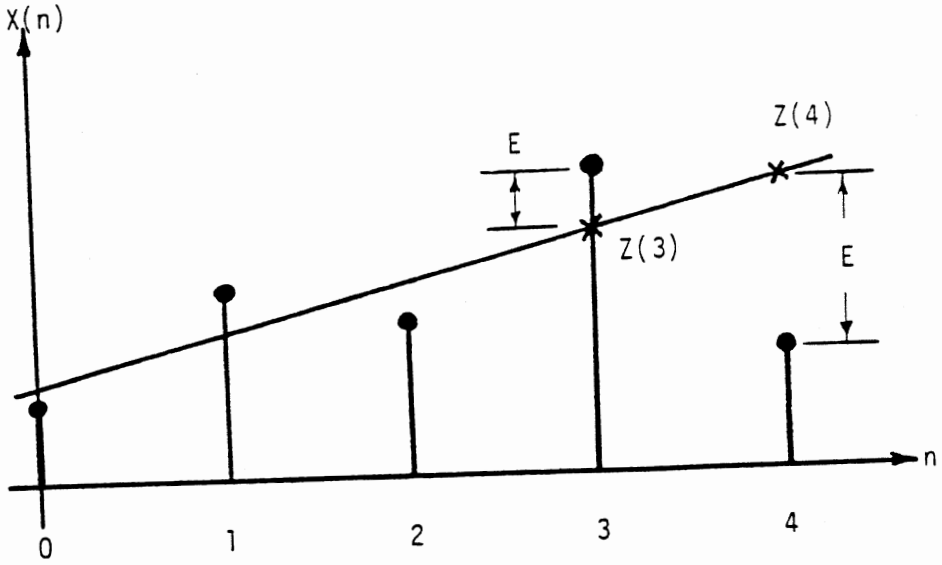


Figure 34. SLPF

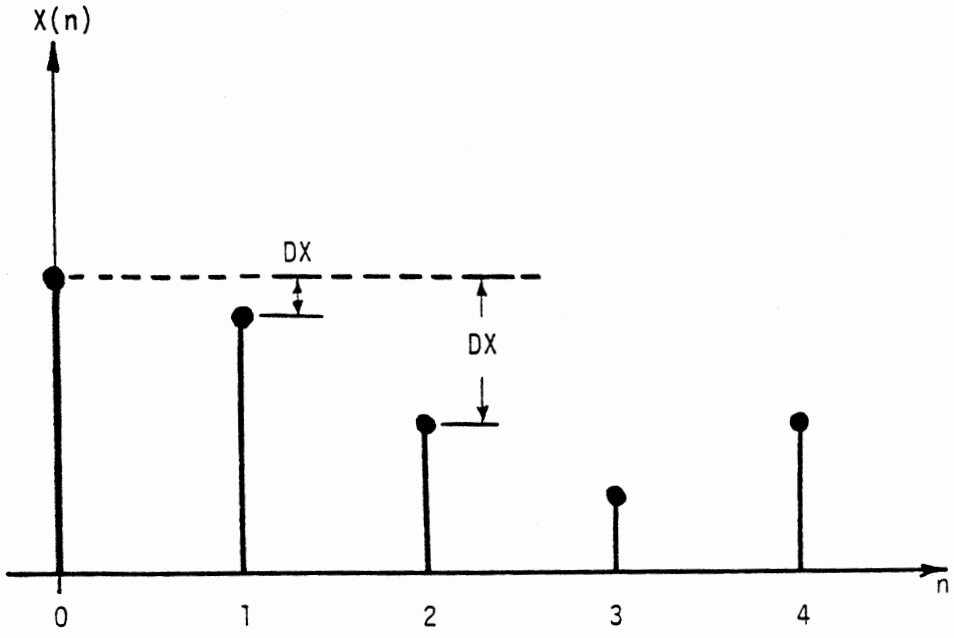


Figure 35. SRA

The details of sample rate adjustment are considered next. SRA is a warping process designed to reduce the number of sample points on sections of a log that are relatively "flat" in shape. Unlike SLPF, it does not discard points on sections of a waveform which have pronounced slopes, even if such sections are good fits to straight lines.

Let X(n), n=0,1,... be the input to the filter. As with SLPF, the output sequence is formed by deleting points from the input sequence. The SRA algorithm starts with X(0) as the reference point and calculates DX = |X(1) - X(0)|. See Figure 35. If DX is less than some specified threshold, then X(1) does not appear as part of the output. Suppose DX is less than the threshold. The next step is to calculate a new DX: DX = |X(2) - X(0)|. Note that X(0) is still the reference point. This process continues until one of two things happens:

(1) The maximum number of points in a row allowed to be deleted (NSKIP) is attained.

(2) The threshold is exceeded.

Suppose one of these conditions occurs for DX = |X(i) - X(0)|. Then X(i) is included in the filter output, and becomes the new reference point as well. The counter for number of points in a row skipped is reset to zero.

Figure 36(a) shows two noisy signatures of length 128 points which are warped versions of each other. The distance between these two waveforms (sum of squared differences) is 201.2. Both signatures were filtered by a SLPF of length 3 with a threshold = 1 and NSKIP = 8. After filtering, both were stretched (using linear interpolation) to a length of 128 points. The distance between these two waveforms, which are shown in Figure 36(b), is only 21.2. Figure 36(c) shows the results

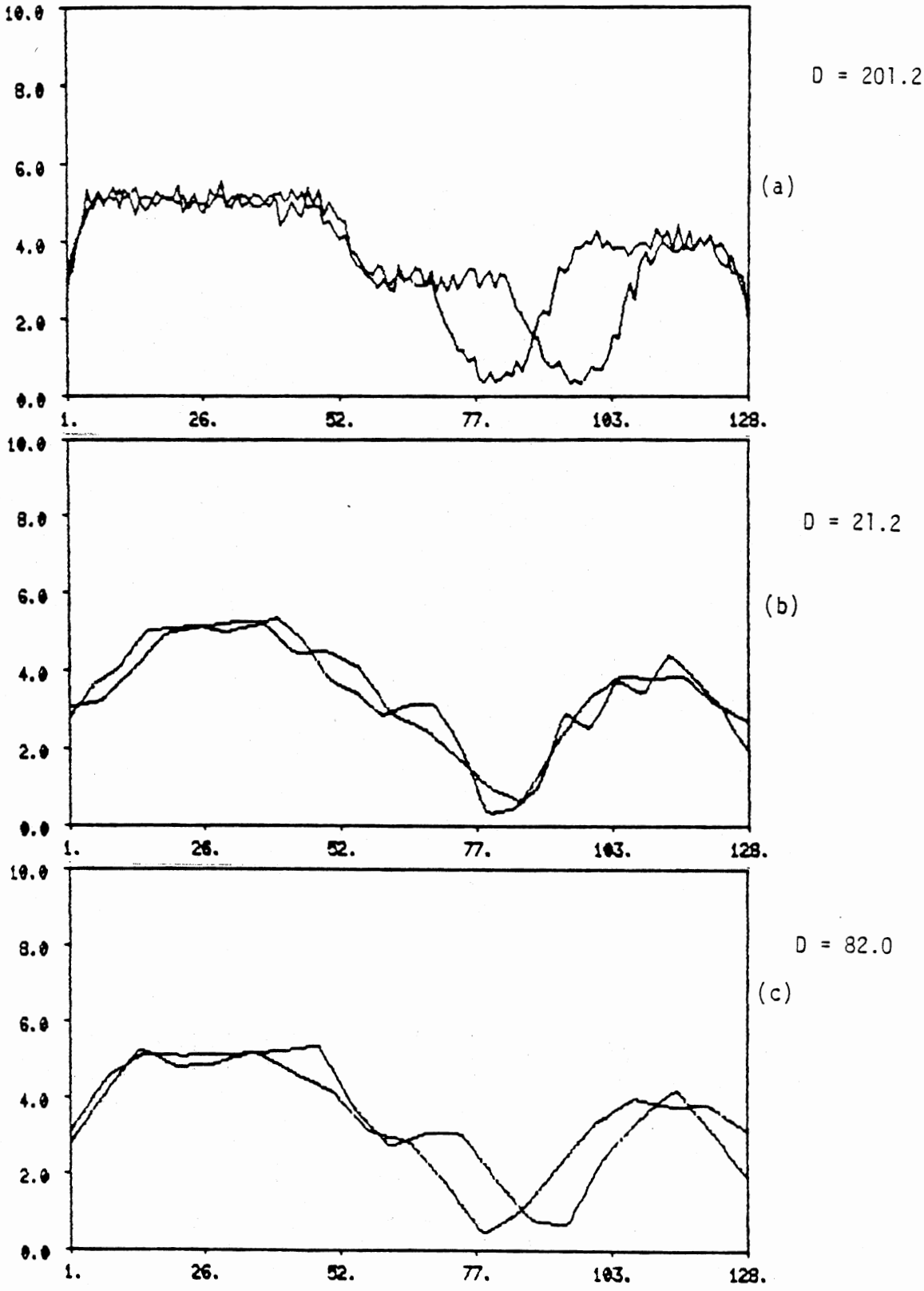


Figure 36. Effect of SLPF and SRA

using SRA instead of SLPF. In this case the distance is 82.0. (The SRA parameters used were threshold = 1 and NSKIP = 8).

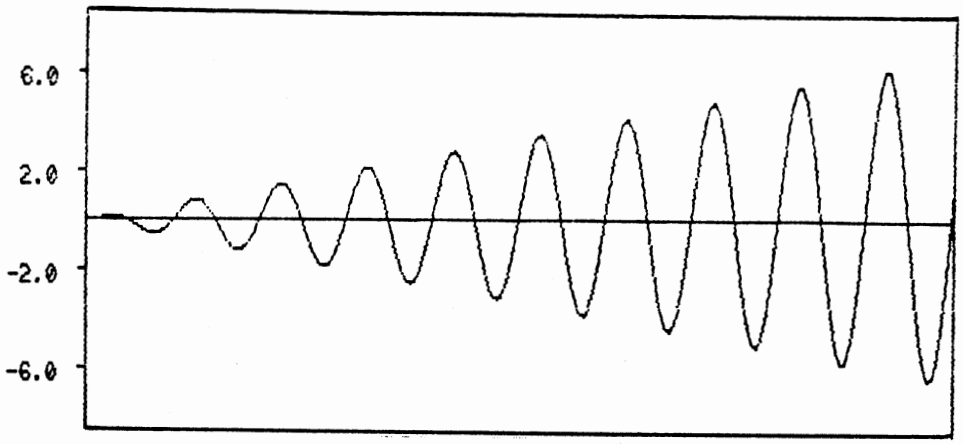
These results are highly selective and are presented to show that SLPF and SRA can indeed make warped versions of a waveform "look more like" each other. Other sets of waveforms were tried, and the results were not always as good.

It is interesting to observe the different effects that SRA and SLPF of length 3 (with the same threshold and NSKIP parameters) have on the sequence given by

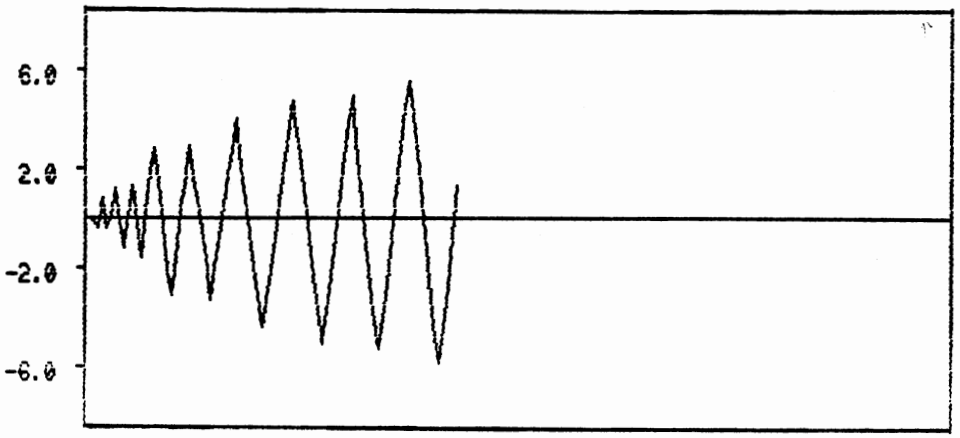
$$X(n) = (n/30)\sin(n\pi/10) \tag{51}$$

which is shown in Figure 37(a). Figure 37(b) shows the sequence that results from SRA; Figure 37(c) shows the sequence that results from SLPF. Inspection of these results shows that SLPF discards more points than SRA, which is to be expected since SRA affects the "flat" parts only. Inspection also shows that in both cases the filtering has a more pronounced effect on the low amplitude parts of the original sequence -- that is, more points are discarded there. This can be seen by observing the peak-to-peak distances. However, this effect is much more severe for SRA than for SLPF. It appears that the warping effect of SRA is more heavily dependent upon waveform amplitudes than is the case for SLPF.

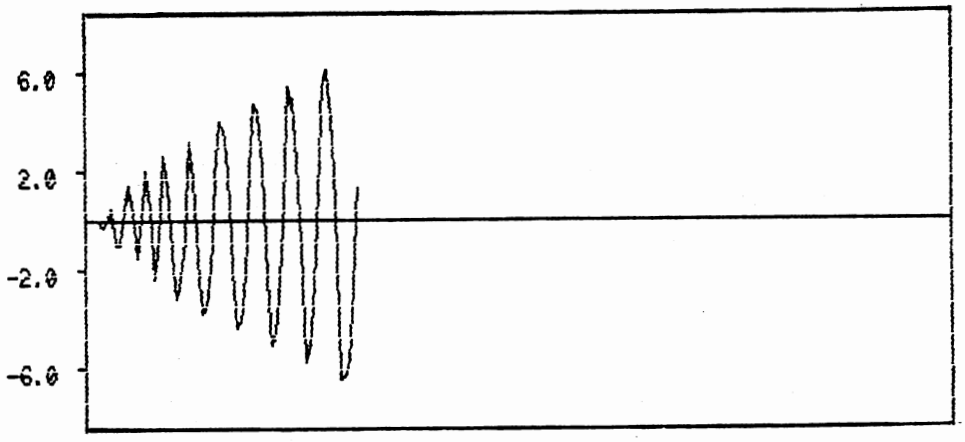
Now that the details of the SRA and SLPF algorithms have been covered, some experimental results with direct template matching can be considered.



(a) Before Filtering



(b) After SRA



(c) After SLPF

Figure 37. Effect of SRA and SLPF on a Test Waveform

Experimental Results with Direct Template Matching

The experimental results reported in this section are all based on the 100 random simulated signature search problems described in Chapter II. The search procedure using direct template matching is described in an earlier section. A good title for this section might well be "experimental results based on parameter twiddling." In a later section, the question of automatic selection of the filter parameters is considered. Results with real well log data are considered in that section.

The straight line prediction filter algorithm has three adjustable parameters: threshold, NSKIP, and filter length. The filter length was fixed at 3 in the following experiments. For the initial experiment the search window size was varied from N/2 to 2N, where N is the number of points in the signature (after filtering.) This range will hereinafter be denoted as the "window search limits," which in this case is [N/2,2N]. The window width and slide increments were both set to 3. (These increments were set to 3 in all experiments in this work unless otherwise noted.) In the initial experiment the SLPF threshold and NSKIP parameters were varied in an attempt to find good values. The results, shown in Table IV, point out the unfortunate but not unexpected fact that reasonably good results are highly dependent on the adjustment of these parameters.

It is interesting to observe that increasing NSKIP from 2 to 8 not only led to higher success rates, but also led to decreases in the CPU time requirement. However, the result for "no limit" on NSKIP suggests that one can have too much of a good thing. The choices NSKIP = 8 and threshold = 1 yielded the best results.

TABLE IV

ADJUSTMENT OF SLPF PARAMETERS (NSKIP AND THRESHOLD) WITH THE WINDOW SEARCH LIMITS FIXED AT [N/2,2N]

NSKIP	Threshold	Av. CPU Time	Percent Correct
2	2	.595	8
2	1	.625	18
5	2	.132	27
5	1	.231	40
5	.5	.568	43
8	2	.089	33
8	1	.188	45
8	.5	.521	45
(no limit)	1	.164	34

TABLE V

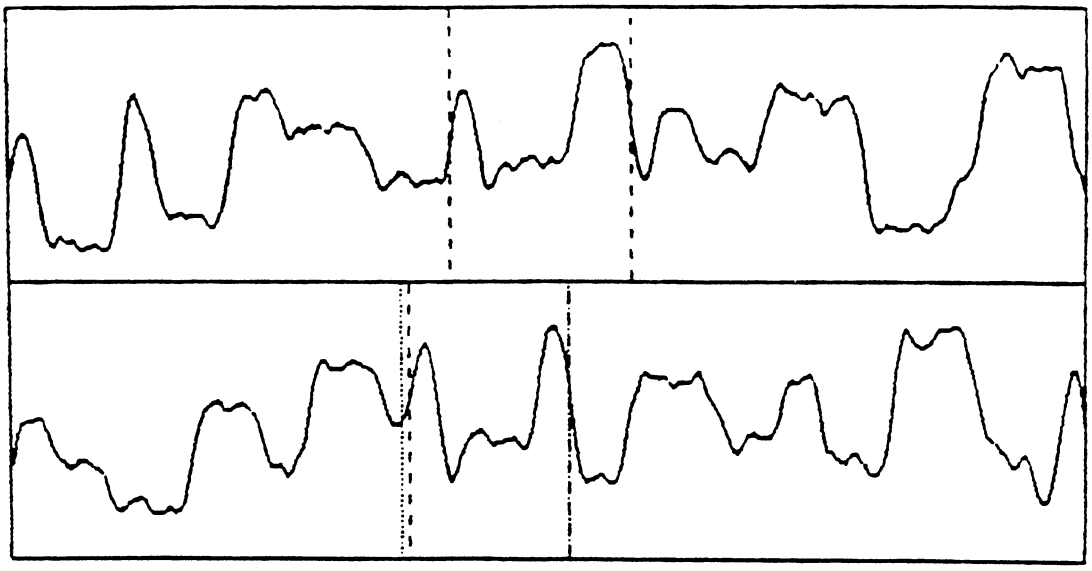
ADJUSTMENT OF WINDOW SEARCH LIMITS WITH SLPF PARAMETERS FIXED AT NSKIP = 8 AND THRESHOLD = 1

Window Search Limits	Av. CPU Time	Percent Correct
[N/2,2N]	.188	45
[3N/4,3N/2]	.113	50
[4N/5,5N/4]	.082	54
[N,N] (one size only)	.051	45

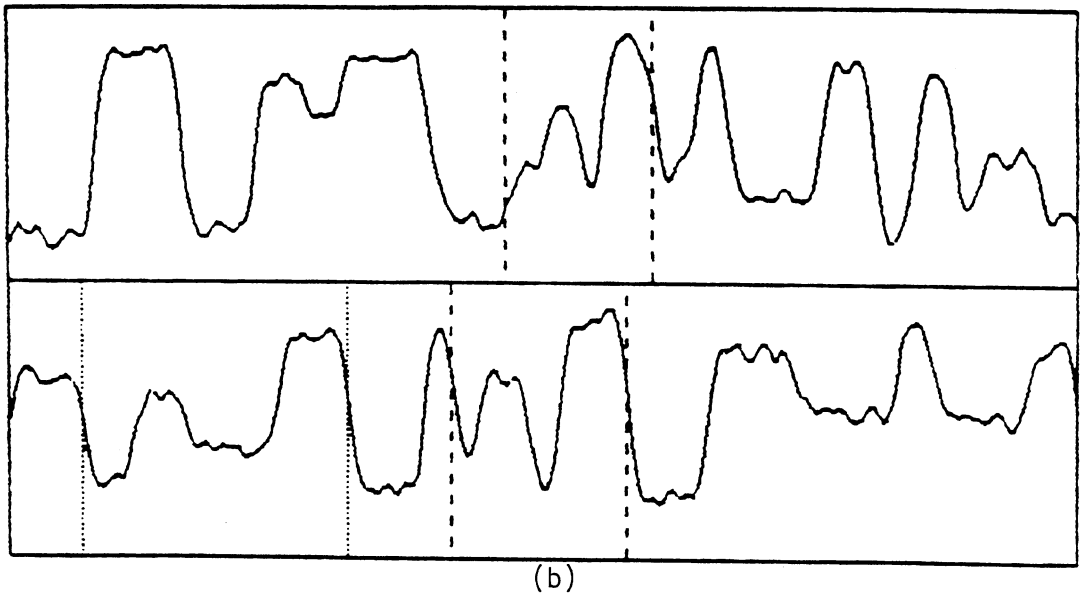
The next experiment involved varying the window search limits. The results (see Table V) show that window search limits of [4N/5, 5N/4] are a reasonably good choice. (It stands to reason that if prewarping makes a signature and its warped versions "look more like" each other, the window search limits could be narrowed.)

The best result obtained using SLPF and direct template matching was 54% correct, while higher success rates were observed using dynamic programming warping. However, direct template matching is significantly faster. For example, dynamic programming warping based on Itakura's method in conjunction with SRA used for data reduction, as reported in Chapter III, had a success rate of 66%, but the CPU time requirement was 15.88 seconds -- approximately 200 times the amount required to obtain the 54% correct figure using SLPF and direct template matching.

One might argue that none of these success rate figures are very impressive, but the reader is reminded that the simple technique described in Chapter II (based on an assumption of uniform stretching) scores zero percent correct on the same 100 problems. It should also be noted that the random problem generator can create some extremely difficult search problems. Consider Figure 38, which shows two examples of random search problems "solved" by direct template matching with SLPF. (In both cases, the window search limits were [N/2,2N]). Figure 38(a) shows a search problem that was solved correctly. Figure 38(b) shows an incorrect result. The dashed vertical lines show the actual signature location. The dotted vertical lines show the location chosen by the search algorithm. The chosen location (Figure 38(b)) is incorrect, but considering the shapes involved it is not an unreasonable choice. This



(a)



(b)

Figure 38. Signature Recognition Problems

is similar to the problem noted earlier with the real well log data (Figure 33, Chapter III).

A serious question that is bound to be raised is how sensitive are these results to changes in the noise level. An experiment was run where the noise level was first reduced by 50% over the standard level used in the random problem generator, and then increased by 50% over this level. The best SLPF parameters (NSKIP = 8, threshold = 1) and window search limits ([4N/5,5N/4]) were used. The results (Table VI) suggest that small changes in noise level won't necessarily lead to catastrophic decreases in the success rate. However, the fact that the success rate drops from 54% to 39% when the noise level is increased by 50% is more evidence that a method of automatically choosing the SLPF parameters for each individual problem is strongly desirable.

It is clear from the results shown so far in this section that the selection of SLPF parameters is a nontrivial matter. The same thing is true for the sample rate adjustment (SRA) parameters. Tables VII, VIII, and IX show the experimental results for direct template matching using SRA; these tables are set up in the same manner as those listing the results using SLPF. These results suggest that for the particular random problem generator parameters in use, the differences between SRA and SLPF are slight. The highest success rate observed is similar for both methods (49% for SRA and 54% for SLPF). However, a closer look at the results shows that many of the problems solved using SLPF were not solved using SRA, and vice-versa. The composite success rate (a success is scored if either SRA or SLPF is successful) is 74% (using the best parameters for each).

TABLE VI

EFFECT OF VARYING THE NOISE LEVEL ABOUT THE "STANDARD" LEVEL ("A" IS THE STANDARD LEVEL) WITH SLPF PARAMETERS FIXED AT NSKIP=8 AND THRESHOLD=1 AND WITH THE WINDOW SEARCH LIMITS FIXED AT [4N/5,5N/4]

Noise Level	Percent Correct
A - .5A	52
A	54
A + .5A	39

TABLE VII

ADJUSTMENT OF SRA PARAMETERS (NSKIP AND THRESHOLD) WITH THE WINDOW SEARCH LIMITS FIXED AT [N/2,2N]

NSKIP	Threshold	Av. CPU Time	Percent Correct
5	2	.092	24
5	1	.225	45
8	1	.150	47
12	1	.120	39

TABLE VIII

ADJUSTMENT OF WINDOW SEARCH LIMITS WITH SRA PARAMETERS FIXED AT NSKIP = 8 AND THRESHOLD = 1

Window Search Limits	Av. CPU Time	Percent Correct
[N/2,2N]	.150	47
[3N/4,3N/2]	.081	44
[4N/5,5N/4]	.054	49
[N,N] (one size only)	.024	40

TABLE IX

EFFECT OF VARYING THE MODEL NOISE LEVEL WHILE FIXING SRA PARAMETERS AT NSKIP=8 AND THRESHOLD=1 AND FIXING THE WINDOW SEARCH LIMITS AT [4N/5,5N/4]

Noise Level	Percent Correct
A - .5A	56
A	49
A + .5A	49

TABLE X

EXPERIMENTAL RESULTS FOR THE "HYBRID" METHOD

Number of preliminary candidates (M)	Av. CPU Time	Percent Correct
2	.33	56
5	.72	64
10	1.32	69
15	2.00	72
20	2.54	71
25	3.13	75
50	6.26	77
100	13.65	78

The best SRA parameters for direct template matching turned out to be threshold = 1 and NSKIP = 8. But it is worth noting that when these parameters were used in the case where SRA is employed for data reduction in conjunction with dynamic programming warping (see Chapter III), the success rate dropped from 66% to 52%. (However, this change did decrease the CPU time requirement from 15.88 seconds to .67 seconds.)

Sample rate adjustment is faster than straight line prediction filtering with length = 3 (.054 seconds vs. .082 seconds), but since SLPF outscored SRA in the percent correct category and since the warping effect of SLPF is less dependent upon waveform amplitude than is the case of SRA, it was decided to use SLPF in most of the work involving statistical pattern recognition and "on the job training." This work is the subject of the next several sections. But before considering statistical pattern recognition, it is interesting to consider the following question: is it possible to combine SLPF based direct template matching with the dynamic programming warping (DPW) method in such a way that the success rate is close to that obtained using DPW by itself, but the average CPU time requirement is still significantly less? In an attempt to partially answer this question, a "hybrid" signature recognition algorithm was designed which works as follows: A preliminary set of M candidates with the smallest distance from the signature is selected using SLPF based direct template matching in the manner described earlier. These M preliminary candidates (selected from the original log using the SLPF mapping array) are then compared with the original signature using dynamic programming warping; the candidate having the best match with the signature (in the DPW sense) is selected

as the final answer. Table X shows the result obtained using various sizes of M. (The SLPF based direct template matching algorithm uses the following parameters: threshold = 1, NSKIP = 8, and length = 3; the window search limits are [4N/5, 5N/4]. The DPW section uses Itakura's method with the L2 distance measure.) Table X shows that the success rate is actually improved over that obtained using DPW alone (Itakura's method with the L2 distance measure in conjunction with SRA based data reduction has a success rate of 66 percent) if M is set to 10 or greater. (For M = 10, the success rate is 69 percent). The required CPU time is of course greater than that required for the direct template matching scheme operating alone, but for the smaller values of M the CPU time requirement is significantly less than that for DPW operating alone. If M = 10, the CPU time requirement is reduced by an order of magnitude (from 15.88 down to 1.32).

On the Job Training (OJT) -- The Basic Idea

The well log signature recognition problem described herein cannot be cast in terms of the traditional pattern recognition problem involving M previously defined classes for which training sets are available. Here, it is assumed that there is only one known class (the signature being searched for), and the answer must be selected from a collection of candidates from previously undefined classes. There is not even a training set available for the signature class. However, it is known that if such a training set did exist, it would contain warped versions of the signature. Therefore, it is possible to create an artificial but useful training set for the signature class by creating a set of warped versions of the given signature. This process has been

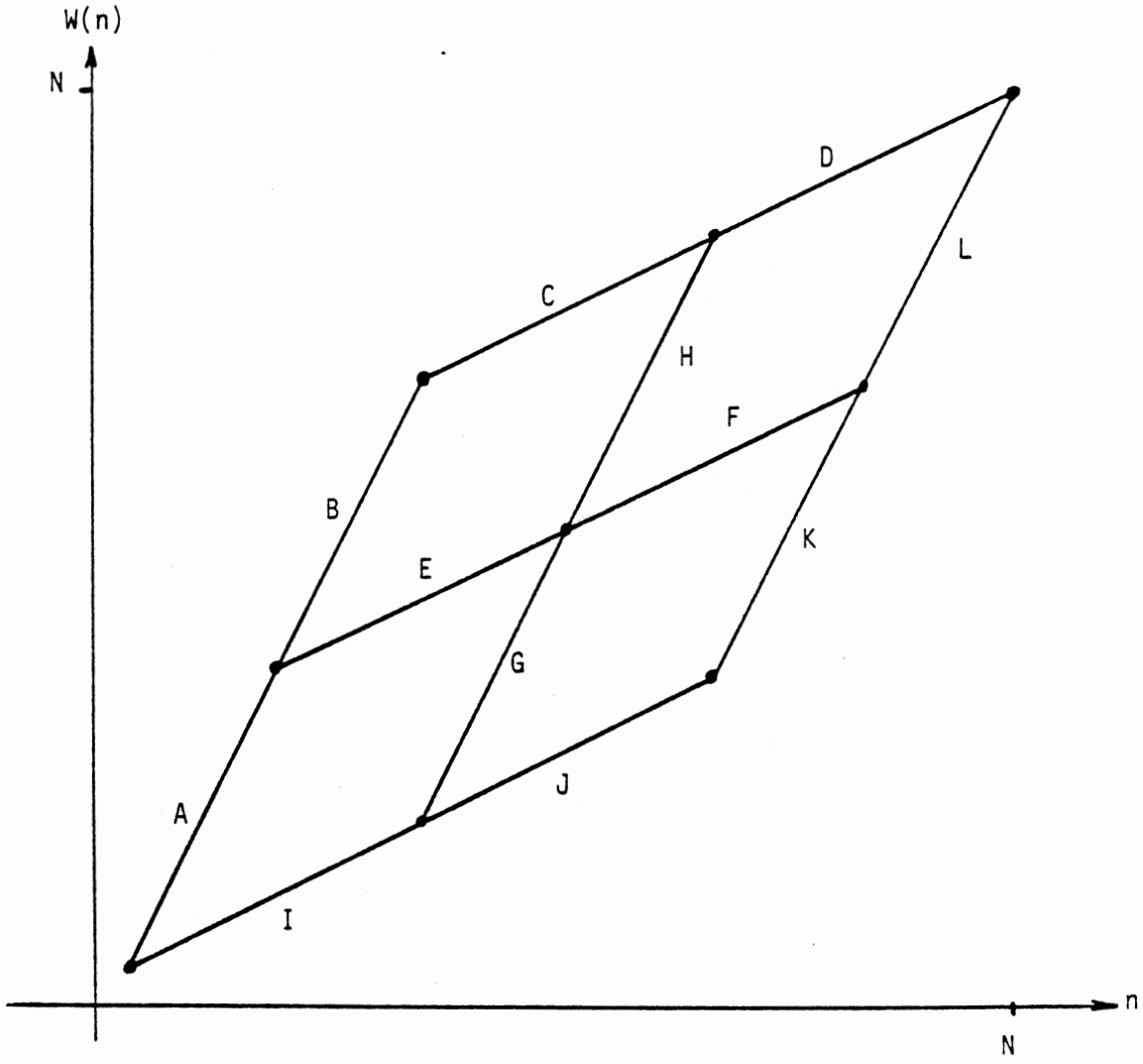
named "on the job training," or OJT for short. Two different methods of creating artificial training sets have been considered in this work.

The first method involves creating a standard set of warping functions which can be applied to a signature. Figure 39 shows six standard warping functions $W_1(n)$, $W_2(n)$, ..., $W_6(n)$ that have been used in the experiments. The artificial training set for the signature S(n) consists of the sequences S(n), $S[W_1(n)]$, $S[W_2(n)]$, ..., $S[W_6(n)]$. This set of warping functions was chosen because it seems more or less to cover the range of possibilities.

The second method involves random warping by (1) segmenting the signature into beds, and (2) applying a random uniform stretching factor to each bed. Figure 40 illustrates the idea. The segmentation method used is the optimal zonation algorithm proposed by Hawkins and Merriam [22]. An assumption about the number of beds in the signature is required. The criterion is to minimize the sum of within-segment variances. The algorithm that performs this task is based on dynamic programming. Once the segmenting is accomplished, as many randomly warped versions as desired can easily be generated. The uniform stretching of individual beds is done by linear interpolation.

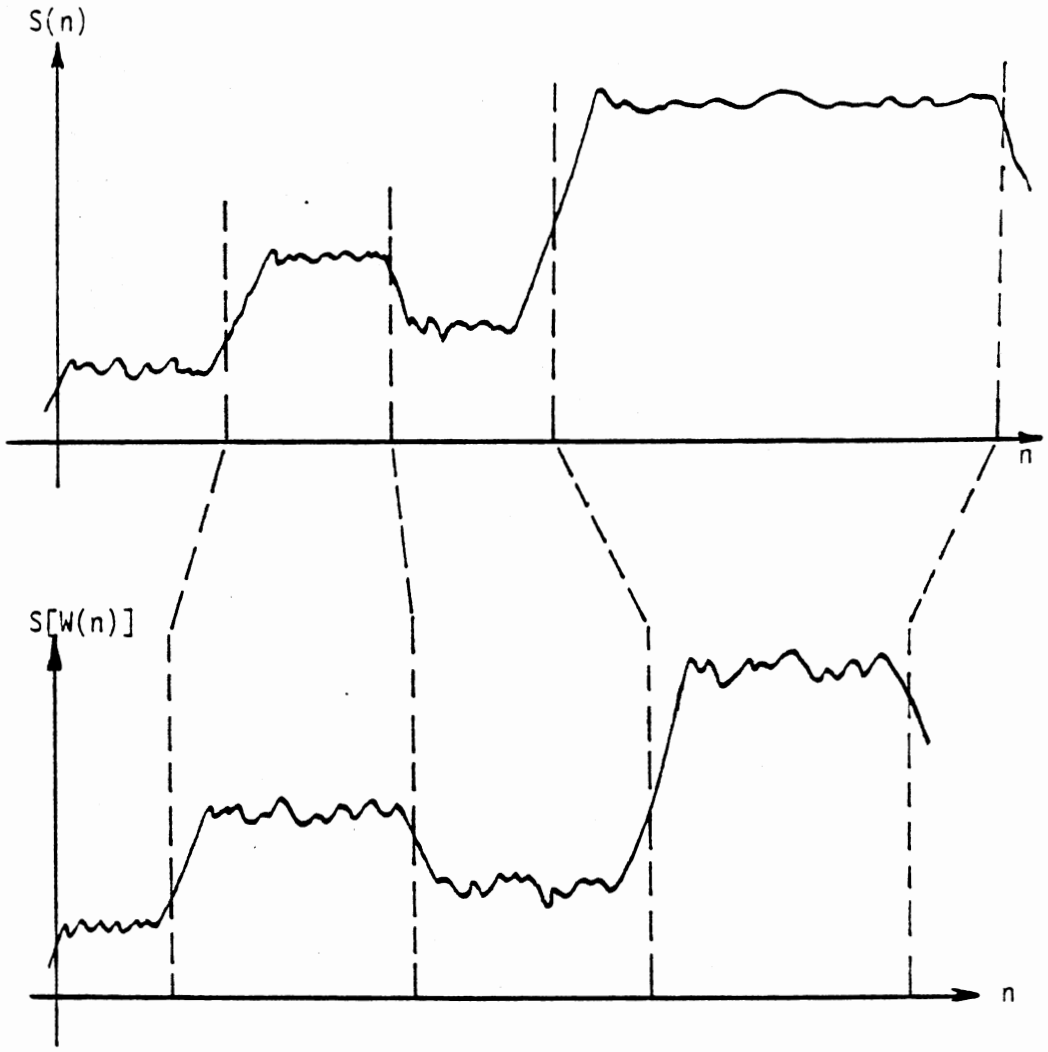
To be useful, all warped versions in the training set must have the same length. A standard length (such as 64 points) is chosen for convenience. In the search algorithm, all candidate sequences must also be converted to the standard length. Standard length is attained by linear interpolation. Experiments with OJT based signature recognition schemes have been run using both of the OJT methods discussed above.

As pointed out in the introduction to this chapter, statistical pattern recognition techniques are made more attractive if the dimension



Six standard warping functions for a sequence of length N are defined by the line segments: ABCD, AEFL, AEHD, IGFL, IGHD, IJKL

Figure 39. Standard Warping Functions



Creating a warped version of S(n) by segmenting into beds and then applying a random uniform stretching factor to each bed

Figure 40. Warping

of the pattern vector is reduced. A "standard length" of 64 points, as described above, can be interpreted to mean that the pattern vectors have dimension 64. The next section describes a way to reduce this dimension.

The Use of Unitary Transformations for Data Reduction

Let A be an nxn unitary matrix, i.e., let $A^* = A^{-1}$, where * denotes complex conjugate transpose. Examples of matricies which have this property (after using a normalization factor) include the discrete Fourier transform (DFT) matrix and the Walsh transform matrix. Let $\underline{x}$ be an nx1 column vector, the elements of which come from a sequence X(n), and consider the transformation

$$A\underline{x} = \underline{y} \tag{52}$$

Let $\underline{u}$ be an nx1 column vector, the elements of which come from a sequence U(n), and consider the transformation

$$A\underline{u} = \underline{v} \tag{53}$$

It is easy to show that the distance (sum of squared differences) between $\underline{x}$ and $\underline{u}$ is the same as the distance between $\underline{y}$ and $\underline{v}$:

$$\begin{aligned} D(\underline{y}, \underline{v}) &= (\underline{y} - \underline{v})^*(\underline{y} - \underline{v}) \\ &= (A\underline{x} - A\underline{u})^*(A\underline{x} - A\underline{u}) \\ &= (\underline{x} - \underline{u})^* A^* A(\underline{x} - \underline{u}) \\ &= D(\underline{x}, \underline{u}) \end{aligned} \tag{54}$$

Because of this property, no improvement in the signature recognition

rate can be expected by simply applying a unitary transformation to the signature and candidates before calculating the distance. (The question of whether such things as the DFT magnitude coefficients will lead to improvement will be considered later.) (It may also be worth pointing out once again, as first noted in Chapter I, that even if two vectors happen to be related by a warping process, in general the corresponding vectors in the transform domain are not related in this manner. The linear transformation and prewarping filter operations do not in general commute.) However, such transformations are often used for data reduction since they tend to "pack" the useful information into fewer coefficients. For example, it turns out that the first 10 coefficients of a 64 point Walsh transform do a good job of representing a 64 point signature, as will be shown shortly.

Data reduction can be important in certain applications of "on the job training." For instance, suppose the standard length of sequences in the training set is 64 points and it is desired to invert and/or find the eigenvalues and eigenvectors of the resulting 64x64 covariance matrix. It is, as they say, "well known" that in spite of the best planned algorithms, computers tend to choke on such problems. A reduction to, say, 10x10 makes life a little easier.

Experiments with data reduction using the Walsh transform and the DFT showed that they can be used to reduce the standard length from 64 to 10 with only a small change in the success rate. Straight line prediction filtering (threshold = 1, NSKIP = 8) was applied to the signature and to the log being searched in a direct template matching scheme. The window search limits were set to [4N/5, 5N/4]. The signature and candidates were stretched to a standard length of 64, and the

transform was then applied. The signature search was done in the transform domain using the L2 distance measure. The number of coefficients used in the distance calculation was varied from 64 down to 5. The results shown in Table XI are based on 100 random search problems. For comparison purposes, the results obtained without a transformation are included. The price to be paid for the use of these transformations is an approximate ten-fold increase in required CPU time. Results using the DFT were similar.

Now that the artificial generation of a set of pattern vectors for the signature class and a means of reducing the dimension of these vectors to something palatable has been covered, the focal point of attention can be turned to the details of the statistical pattern recognition techniques mentioned in the introduction to this chapter. The next two sections cover (1) Euclidean distance and clustering transformations, and (2) Mahalanobis distance and estimation of probability density functions.

OJT and Statistical Pattern Recognition: Euclidean Distance and Clustering Transformations

"On the job training" can be used to find statistical properties such as the class centroid and the class covariance matrix. These properties can then be used in conjunction with various distance measures.

Given a training set $\{\underline{s}_1, \underline{s}_2, \ldots, \underline{s}_M\}$, one can find the class centroid vector $\underline{m}$ given by

$$\underline{m} = (1/M) \sum_{i=1}^{M} \underline{s}_i \tag{55}$$

TABLE XI

RESULTS USING WALSH TRANSFORM COEFFICIENTS
(64 PT. TRANSFORM)

Number of Coefficients	Av. CPU Time	Percent Correct
(no transformation)	.189	55
64	1.582	55
50	1.571	55
40	1.566	55
32	1.574	55
10	1.510	50
5	1.540	44
(Results using DFT coefficients)		
10	1.640	49

TABLE XII

SOME STATISTICAL PATTERN RECOGNITION
RESULTS

Experiment	Average CPU Time	Percent Correct
A	.266	53
B	.293	38
C	1.80	54
D	1.78	52

and then hope that $\underline{m}$ is representative of the class. If $\underline{x}$ is a candidate vector, one could consider the distance of the candidate from the class centroid vector instead of the distance from the original signature:

$$D = (\underline{x} - \underline{m})^T(\underline{x} - \underline{m}) \tag{56}$$

A somewhat more sophisticated approach is to find a diagonal clustering transformation matrix W such that premultiplying each vector in the training set by W minimizes the intraclass distance [28]. The intraclass distance is given by

$$\bar{D} = (1/M) \sum_{j=1}^{M} \left[(1/(M-1)) \sum_{i=1}^{M} (\underline{s}_i - \underline{s}_j)^T(\underline{s}_i - \underline{s}_j)\right] \tag{57}$$

which can be interpreted as the average distance of one vector from another within the class. After some manipulation, this reduces to

$$\bar{D} = 2 \sum_{k=1}^{n} \sigma_k^2 \tag{58}$$

where σ_k^2 is the unbiased sample variance of the kth component of the pattern vectors. (n is the number of elements in each pattern vector.)

Let $W = \text{diag}(w_1, w_2, \ldots, w_n)$. If each pattern vector in the set is premultiplied by W, the new interclass distance is

$$\underline{D} = 2 \sum_{k=1}^{n} w_k^2 \sigma_k^2 \tag{59}$$

It is desired to find the w_k's that minimize this sum. To avoid the trivial solution $w_k = 0$, a constraint is needed. One possible constraint has the form

$$\prod_{k=1}^{n} w_k = K \neq 0 \tag{60}$$

where K is a constant chosen for convenience. It can be shown that if K is chosen as

$$K = \prod_{k=1}^{n} 1/\sigma_k \tag{61}$$

then the solution for w_k, obtained by using Lagrange multipliers [28], turns out to be

$$w_k = 1/\sigma_k \tag{62}$$

This result is intuitively pleasing since it means that each element in a pattern vector is weighted by the inverse of its standard deviation. That is, the smaller the standard deviation, the greater the importance assigned to the element.

It turns out that using this clustering transformation before calculating a Euclidean distance is equivalent to assuming that the covariance matrix C for the class is a diagonal matrix and then calculating the Mahalanobis distance. To show this, consider the following Euclidean distance equations:

$$\begin{aligned} D &= (W\underline{x} - W\underline{m})^T(W\underline{x} - W\underline{m}) \\ &= (\underline{x} - \underline{m})^T W^T W(\underline{x} - \underline{m}) \end{aligned} \tag{63}$$

Now observe that

$$W^T W = \text{diag}\left(\frac{1}{\sigma_1^2} \frac{1}{\sigma_2^2} \ldots \frac{1}{\sigma_n^2}\right) \tag{64}$$

which is the same thing as C^{-1} if the covariance matrix is diagonal. That is,

$$D = (\underline{x} - \underline{m})^T C^{-1}(\underline{x} - \underline{m}) \tag{65}$$

which is the Mahalanobis distance. (The connection between the Mahalanobis distance and the assumption of a normal probability density function for $\underline{x}$ will be pointed out in the next section.)

Before considering signature recognition based on estimates of the probability density function for the signature class, it would perhaps be instructive to consider some experimental results based on Euclidean distance from the class mean, both with and without the clustering transformation, and both with and without using the Walsh transform for data reduction. Straight line prediction filtering (SLPF) was used for prewarping in each of these experiments; the SLPF parameters were length = 3, NSKIP = 8, and threshold = 1. The window search limits were [4N/5, 5N/4]. The standard stretched length for sequences was 64. All results are for the same 100 random search problems. Experiment A is for the Euclidean distance from the class mean, without any transformations. Experiment B uses the clustering transformation described above, but does not use any data reduction technique. Experiment C is for the Euclidean distance from the class mean in the Walsh transform domain (keeping the first 10 coefficients only). Experiment D uses the clustering transformation and the Walsh transform for data reduction. A general outline of the signature search procedure for these four experiments is as follows:

Step 1. Stretch the signature to the standard length (64 points).

Step 2. Create six warped versions of the signature with the standard warping functions.

Step 3. Apply SLPF to the signature and its warped versions, then stretch them back to the standard length. This creates a preliminary training set.

Step 4. (OPTIONAL). Apply the Walsh transform to each sequence in the preliminary training set. Keep only the first 10 coefficients for each transformation. The resulting set of 10x1 vectors forms the training set.

Step 5. Find the centroid vector $\underline{m}$ for the training set.

Step 6. (OPTIONAL). Find the clustering transformation matrix W for the training set.

Step 7. Apply SLPF to the log being searched.

Step 8. Use the search window to extract candidates, each of which must be stretched to the standard length. (Optional) apply the Walsh transform to candidate sequences and keep the first 10 coefficients to form a candidate vector. Let $\underline{x}$ be the candidate vector. The distance measure is

$$D = (W\underline{x} - W\underline{m})^T(W\underline{x} - W\underline{m}) \tag{66}$$

if the clustering transformation is being used; otherwise, the distance measure is

$$D = (\underline{x} - \underline{m})^T(\underline{x} - \underline{m}) \tag{67}$$

Experimental results are summarized in Table XII.

In regard to the results for experiments A, B, C, and D, it is interesting to note that the simplest method (no Walsh transform and no clustering transformation) is as good as any of the others. In fact, the clustering transformation seems to make things worse, which shows that it cannot be viewed as a panacea. It should also be noted that experiments using the "random warping OJT" method with 20 warped versions of the signature revealed that for this particular application,

the random method has no advantage over the "standard warping function" method. Percent correct figures were lower, and the method is somewhat slower since the segmentation algorithm slows things down.

OJT and Statistical Pattern Recognition: Mahalanobis Distance and Probability Density Function Estimation

If the density function for the signature class is unknown but the mean vector $\underline{m}$ and covariance matrix C can be estimated (which can be accomplished by OJT), it can be shown that the choice of a normal density function is satisfactory from a maximum entropy point of view [28], where entropy is given by

$$H = -\int_{\underline{X}} p(\underline{x})\mathrm{Ln}(p(\underline{x}))\, d\underline{x} \tag{68}$$

The estimates for $\underline{m}$ and C are given by

$$\underline{m} = (1/M) \sum_{i=1}^{M} \underline{s}_i \tag{69}$$

$$C = (1/M) \sum_{i=1}^{M} (\underline{s}_i - \underline{m})(\underline{s}_i - \underline{m})^T \tag{70}$$

where M is the number of vectors in the training set. Finding the candidate vector $\underline{x}$ that minimizes the Mahalanobis distance is the same thing as finding the candidate $\underline{x}$ that maximizes the normal density function given by

$$p(\underline{x}) = \frac{1}{(2\pi)^{n/2}|C|^{1/2}} \exp[-1/2(\underline{x} - \underline{m})^T C^{-1}(\underline{x} - \underline{m})] \tag{71}$$

Of course, the actual covariance matrix C is not in general diagonal. In general, using the Mahalanobis distance requires the inversion of a

non-diagonal matrix. There apparently exists a "rule of thumb" that in order for the estimate of the covariance matrix to be nonsingular, it is necessary to have M>n, where n is the dimension of the sample vector $\underline{s}_i$ [29]. This rule of thumb is based on the fact that if M>n and the samples are from a class with a normal density function, the estimate of C has an inverse with probability 1 [30]. In any event, the usefulness of this rule of thumb has been demonstrated by experiments with the Mahalanobis distance, as will be shown later. To avoid inverting a 64x64 matrix, these experiments are based on vectors of dimension n=10, obtained by using the Walsh transform and keeping the first 10 coefficients. (There is a side benefit to using the Walsh transform here: the central limit theorem suggests that if the dimension of $\underline{x}$ is large, $\underline{y} = A\,\underline{x}$ will be approximately normally distributed even if $\underline{x}$ is not.) Let it be noted here that the best success rate observed using the Mahalanobis distance was only 37%.

It is also possible to use OJT to directly estimate the density function of the signature class. Let $\hat{p}(\underline{x})$ be the estimate. This estimate is expressed as a weighted sum of orthonormal functions, i.e., [28]

$$\hat{p}(\underline{x}) = \sum_{j=1}^{m} c_j F_j(\underline{x}) \tag{72}$$

where $F_1(\underline{x})$, $F_2(\underline{x})$, ..., $F_m(\underline{x})$ are a set of orthonormal functions. Signature recognition would be based on choosing the candidate that maximizes $\hat{p}(\underline{x})$. A mean-square error function is minimized in order to find the coefficients c_j. Let R be this function:

$$R = \int_{\underline{x}} [p(\underline{x}) - \hat{p}(\underline{x})]^2 \, d\underline{x} \tag{73}$$

where $p(\underline{x})$ is the true density function. Substituting (72 into 73) leads to

$$R = \int_{\underline{x}} [p(\underline{x}) - \sum_{j=1}^{m} c_j F_j(\underline{x})]^2 \, d\underline{x} \tag{74}$$

Taking partial derivatives with respect to c_k, $k = 1,2,\ldots,m$ leads to the set of equations

$$\int_{\underline{x}} \sum_{j=1}^{m} c_j F_j(\underline{x}) F_k(\underline{x}) \, d\underline{x} = \int_{\underline{x}} F_k(\underline{x}) p(\underline{x}) \, d\underline{x} \tag{75}$$

The quantity on the right side of (75) is the expected value of $F_k(\underline{x})$, which can be approximated by

$$E\,[F_k(\underline{x})] \cong (1/M) \sum_{i=1}^{M} F_k(\underline{x}_i) \tag{76}$$

where again, M is the number of training vectors $(\underline{x}_1, \underline{x}_2, \ldots, \underline{x}_M)$. (As with the work using the Mahalanobis distance, the training vectors used in the experiments with pdf estimation are of dimension 10). The quantity on the left side of Equation (75) can be considerably simplified because of the orthonormal property of $Fj(\underline{x})$, i.e.,

$$\int_{\underline{x}} F_j(\underline{x}) F_k(\underline{x}) \, d\underline{x} = \begin{cases} 1, & j = k \\ 0, & \text{otherwise} \end{cases} \tag{77}$$

Therefore, Equation (75) reduces to

$$c_j = (1/M) \sum_{i=1}^{M} F_j(\underline{x}_i) \;, \quad j = 1,2,\ldots,m \tag{78}$$

One set of single variable functions which are orthonormal in the interval $(-\infty, \infty)$ is given by

$$f_j(x) = \frac{\exp(-x^2/2) H_j(x)}{\sqrt{2^j \, j! \sqrt{\pi}}} \;, \quad j = 0,1,\ldots \tag{79}$$

where $H_j(x)$ are Hermite polynomials, given by

$$H_0(x) = 1$$

$$H_1(x) = 2x$$

$$H_{j+1}(x) - 2xH_j(x) + 2jH_{j-1}(x) = 0, \quad j = 1,2,... \tag{80}$$

Multivariate orthonormal functions can be constructed by multiplying together single variable orthonormal functions. For example, suppose $\underline{x}$ has dimension n = 4. Then orthonormal functions in 4 variables can be constructed using the following form:

$$F(\underline{x}) = f_i(x_1)f_j(x_2)f_k(x_3)f_1(x_4) \tag{81}$$

The indicies i, j, k, and 1 can take on any value greater than or equal to zero, out to the number of orthonormal functions in the set.

The construction of multivariate orthonormal functions based on Hermite polynomials for use in the density function estimation problem has been suggested by Tou and Gonzales [28]. However, it is easy to generate a simple set of 45 orthonormal functions in 10 space that do not depend on exponential weighting functions. These orthonormal functions are very similar to what one would obtain by concatenating Hermite polynomials. The difference is that instead of being orthonormal over all of 10-space, they are orthonormal over an arbitrarily large hypercube. (45 such functions should be enough to test the idea). Let

$$F_j(\underline{x}) = (1/\sqrt{A})x_i x_k = (1/\sqrt{A})G_j(\underline{x}), \quad \begin{matrix} i = 1,2,...,10 \\ k = 1,2,...,10 \\ i \neq k \end{matrix} \tag{82}$$

There are (10!)/(2!(10-2)!) = 45 such functions. Since the following integral is seperable, it is easy to show that the orthonormal property

is satisfied, i.e.,

$$\int_{\underline{x}} F_j(\underline{x}) F_k(\underline{x})\, d\underline{x} = \begin{cases} 1, & j=k \\ 0, & \text{otherwise} \end{cases} \tag{83}$$

where $\underline{x}$ is the region in 10-space given by

$$\begin{aligned} x_1 &\in (-T,T) \\ x_2 &\in (-T,T) \\ &\vdots \\ x_{10} &\in (-T,T) \end{aligned}$$

and where

$$A = \left(\int_{-T}^{T} dx\right)^8 \left(\int_{-T}^{T} x^2\, dx\right)^2 = \frac{2^{10} T^{14}}{9} \tag{84}$$

T is any constant; therefore, the region over which these functions are orthonormal is arbitrarily large. The constant A is in general much too large to handle on a computer. Fortunately, it isn't necessary to do so. Using these functions, the estimated density function has the form

$$\hat{p}(\underline{x}) = (1/\sqrt{A}) \sum_{j=1}^{45} c_j G_j(\underline{x}) \tag{85}$$

where

$$c_j = (1/M) \sum_{i=1}^{M} F_j(\underline{x}_i) = \frac{1}{M\sqrt{A}} \sum_{i=1}^{M} G_j(\underline{x}_i) = \frac{1}{M\sqrt{A}}\, \hat{c}_j \tag{86}$$

Therefore,

$$\hat{p}(\underline{x}) = \frac{1}{M\,A} \sum_{j=1}^{45} \hat{c}_j G_j(\underline{x}) \tag{87}$$

The constant 1/MA can be ignored since it has no effect on finding the maximum. A new decision function can therefore be defined:

$$Q(\underline{x}) = \sum_{j=1}^{45} \hat{c}_j G_j(\underline{x}) \tag{88}$$

where

$$G_j(\underline{x}) = x_i x_k, \; i \neq k; \; i,k = 1,2,\ldots,10 \tag{89}$$

and where

$$\hat{c}_j = \sum_{i=1}^{M} G_j(\underline{x}_i) \tag{90}$$

Another approach, recommended by Tou and Gonzales, is to treat the Hermite polynomials, without the exponential weighting function, as if they formed an orthonormal set. This is difficult to justify rigorously here, but it is claimed to be "in general a good practice which avoids computational difficulty" [28]. This method has also been tried in this work (using 11, 56, and 138 functions in the series); but in no case was the success rate any better than that obtained using the 45 functions described herein. 56 orthonormal Hermite polynomial based functions (with the exponential weighting function included) were also tried in the series pdf estimation, but the success rate for this method was even lower.

The experimental results with density function estimation based on the orthonormal functions of Equation (82) are disappointing, since the highest success rate observed was 31%. However, there is another set of orthonormal functions in 10-space which give much better results (although the 48% success rate was still far short of what was hoped for). These orthonormal functions are based on the sine and cosine functions:

$$F_1(\underline{x}) = A \sin(\omega x_1)$$

$$F_2(\underline{x}) = A \sin(\omega x_2)$$

$$\vdots$$

$$
\begin{aligned}
F_{10}(\underline{x}) &= A \sin(\omega x_{10}) \\
F_{11}(\underline{x}) &= A \cos(\omega x_1) \\
F_{12}(\underline{x}) &= A \cos(\omega x_2) \\
&\vdots \\
F_{20}(\underline{x}) &= A \cos(\omega x_{10})
\end{aligned}
\tag{91}
$$

where

$$\omega = (2\pi/T) \tag{92}$$

and where

$$A = \sqrt{2/T} \tag{93}$$

This set of functions is orthonormal over a hypercube in 10-space, each "side" of which has a length of KT, where K is some positive integer. The value of T must be chosen.

As was the case with the orthonormal functions of Equation (82), the constant multiplier terms in the density function estimate can be discarded, leading to the decision function

$$Q(\underline{x}) = \sum_{j=1}^{20} \hat{c}_j G_j(\underline{x}) \tag{94}$$

where

$$G_j(\underline{x}) = (1/A)F_j(\underline{x}) = \begin{cases} \sin(\omega x_j), & j = 1, \ldots, 10 \\ \cos(\omega x_{j-10}), & j = 11, \ldots, 20 \end{cases} \tag{95}$$

and where

$$\hat{c}_j = \sum_{j=1}^{20} G_j(\underline{x}_i) \tag{96}$$

Of course, one could generate more than 20 orthonormal functions of this form. In general, they have the form

$$F(\underline{x}) = A \sin(n\omega x) \tag{97}$$

or

$$F(\underline{x}) = A \cos(n\omega x) \tag{98}$$

The twenty-first through fortieth functions are

$$\begin{aligned} F_{21}(\underline{x}) &= A \sin(2\omega x) \\ &\vdots \\ F_{30}(\underline{x}) &= A \sin(2\omega x_{10}) \\ F_{31}(\underline{x}) &= A \cos(2\omega x_1) \\ &\vdots \\ F_{40}(\underline{x}) & A \cos(2\omega x_{10}) \end{aligned} \tag{99}$$

It must be noted, though, that increasing the number of orthonormal functions in the density function estimate from twenty to forty did not improve the signature recognition success rate.

Experimental results for the Mahalanobis distance and pdf estimation based on orthonormal functions will now be presented. Straight line prediction filtering was used for prewarping in these experiments; the SLPF parameters were length = 3, NSKIP = 8, and threshold = 1. The window search limits were [4N/5, 5N/4]. The standard stretched length was 64 points. The Walsh transform was used for data reduction; the pattern vectors have dimension n = 10. Except where indicated otherwise in the results tables, the OJT method was the segment and random warp technique; the number of training vectors is specified in these tables. All results are for the same 100 random search problems. A general outline of the signature search procedure for these experiments is as follows:

Step 1. Create an OJT training set by the segment and random warp method. Apply SLPF to each training vector, and then stretch each one back to the standard length. This creates the preliminary training set.

Step 2. Apply the Walsh transform to each sequence in the preliminary training set. Keep only the first 10 coefficients for each transformation. The resulting set of 10x1 vectors forms the training set.

Step 3. If the Mahalanobis distance is to be used, estimate the class mean and covariance matrix. Otherwise, estimate the pdf of the signature class using orthonormal functions.

Step 4. Apply SLPF to the log being searched.

Step 5. Use the search window to extract candidates, each of which must be stretched to the standard length. Apply the Walsh transform to candidate sequences, and keep the first 10 coefficients to form a candidate vector. Candidate vectors are evaluated as described above.

Table XIII shows the results for the Mahalanobis distance. As noted earlier, the highest success rate obtained was only 37%. The condition numbers (returned by LINPAC subroutine DGECO) are interesting because they demonstrate the validity of the "rule of thumb" concerning the number of training vectors needed to avoid badly conditioned covariance matricies. The smaller the condition number, the more ill conditioned the matrix is, i.e., the closer it is to being singular. These results show that the more badly conditioned the covariance matrix, the lower the success rate is.

Table XIV shows the results for pdf estimation based on the functions shown in Equation 82. The success rate was only 29% using 10 OJT

TABLE XIII

MAHALANOBIS DISTANCE RESULTS

Number of Training Vectors	Average Condition No.	Av. CPU Time	Percent Correct
7 *	7.4 E-9	2.01	10
10	4.9 E-7	2.55	13
12	8.2 E-5	2.63	32
20	4.7 E-4	2.89	36
50	8.7 E-4	4.05	37

* using "standard warping functions"

TABLE XIV

PDF ESTIMATION BASED ON EQUATION (82)
(45 ORTHOGONAL FUNCTIONS)

Number of training vectors	Average CPU time	Percent Correct
7 *	1.95	15
10	2.54	29
20	3.10	30
50	4.24	31

* using "standard warping functions"

vectors; increasing the number of training vectors to 50 resulted in a still disappointing success rate of 31%.

Table XV shows the results for pdf estimation based on the 20 sinusoidal functions shown in Equation 91. A success rate of 45% was observed using 20 training vectors; the rate increased to 48% when the number of training vectors was increased to 50. The best results were obtained by setting T (Equation 93) equal to twice the average value of the first Walsh transform coefficient of the OJT training set. Note that the first Walsh transform coefficient is the sum of all the values in the sequence being transformed.

Table XVI shows a result for pdf estimation based on 40 sinusoidal functions. Note that increasing the number of terms in the series estimation of the pdf from 20 to 40 did not improve the results.

In regard to all of the experimental results for statistical pattern recognition based on "on the job training," it is disappointing to observe that the percent correct figures obtained using these techniques show no real improvement over the results obtained using direct template matching without the benefit of OJT. Since this is the case, it seems reasonable to ask whether or not statistical pattern recognition methods are applicable to the problem under consideration. Most of the statistical methods discussed above make use of the "class centroid" vector. In most pattern recognition work there is a tacit assumption that the class centroid is also a member of the class; it may very well be that this is a poor assumption if the class is a set of warped sequences. Figures 41(a) thru 41(g) show a signature and 6 warped versions. Figure 42 shows the class centroid obtained by averaging these 7 waveforms; it is clearly not representative of the class. This example

TABLE XV

PDF ESTIMATION: SINUSOIDAL FUNCTIONS
(20 TERMS)

Number of Training Vectors	Value of T	Average CPU Time	Percent Correct
20	1000	2.93	44
20	100	3.06	16
20	500	2.98	45
20	Wa *	3.01	44
20	Wa/2	3.07	25
20	2Wa	3.05	45
50	2Wa	4.17	48
7 **	2Wa	1.97	29

* Wa = average value of first Walsh transform coefficient.
** using "standard warping functions"

TABLE XVI

PDF ESTIMATION: SINUSOIDAL FUNCTIONS
(40 TERMS)

Number of Training Vectors	Value of T	Average CPU Time	Percent Correct
50	2Wa *	4.48	47

* Wa = average value of first Walsh transform coefficient.

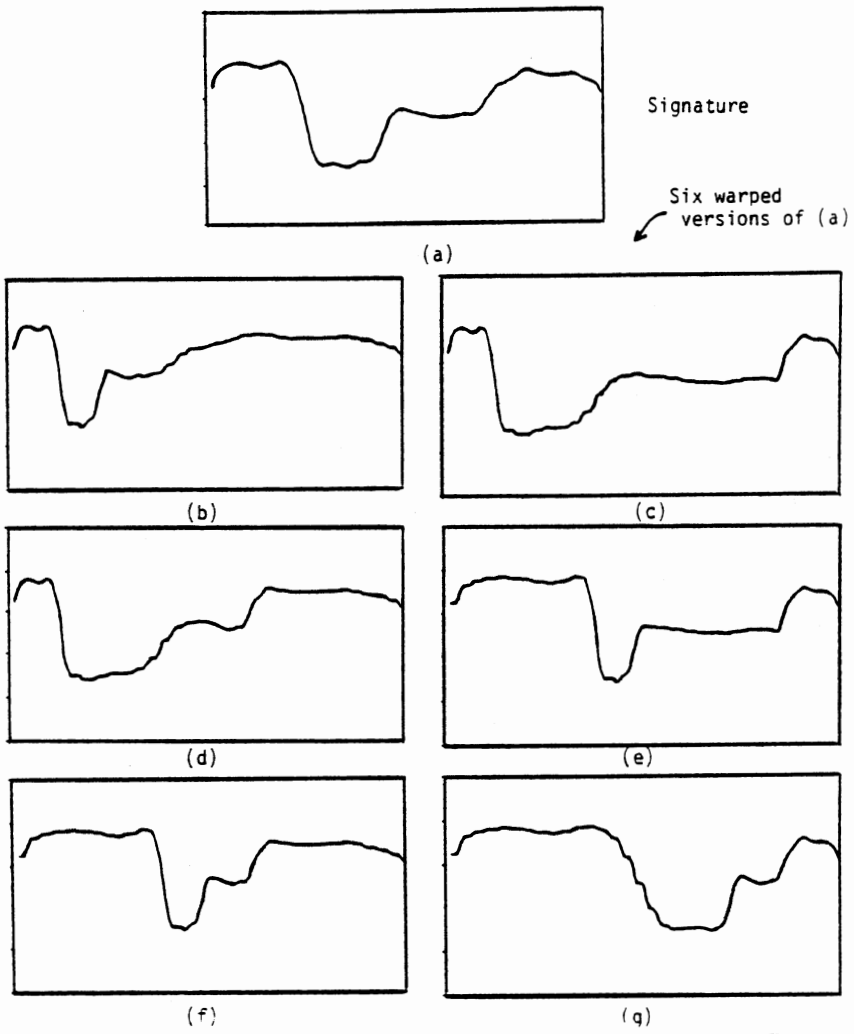


Figure 41. Signature and 6 Warped Versions

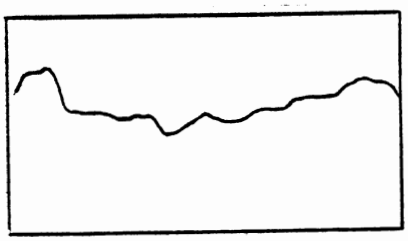


Figure 42. "Class Centroid "

was obtained without using prewarping. Prewarping before calculating a centroid may lead to a less dismal picture, and what happens in the Walsh transform domain is another question, but the purpose here is only to illustrate a possible problem with the above mentioned tacit assumption.

This concludes the discussion of the statistical pattern recognition application of on the job training. However, there are two more applications of OJT that need to be covered. The next two sections consider (1) an OJT based signature recognition scheme using singular value decomposition, and (2) a method of using OJT to automatically select the parameters for the prewarping filters.

OJT in Conjunction with Singular Value Decomposition (SVD)

Another signature recognition method considered in this work uses OJT in conjunction with singular value decomposition. An nxm real matrix S can be expressed as

$$S = A\ Q\ B^T \tag{100}$$

where A is an nxm matrix, Q is an mxm diagonal matrix, and B is an mxm orthonormal matrix. This is known as singular value decomposition (SVD). Here it is assumed that $n < m$; therefore, Equation (100) can be expressed in terms of partitioned matricies:

$$S = [U\ W]\begin{bmatrix} D & 0 \\ 0 & 0 \end{bmatrix}\begin{bmatrix} V^T \\ C^T \end{bmatrix} \tag{101}$$

U is an nxn matrix, D is an nxn diagonal matrix, and V^T is an nxm matrix composed of orthonormal row vectors. This simplifies at once to

$$S = U\ D\ V^T \tag{102}$$

Taking the transpose of both sides of Equation (102) leads to the following expression:

$$\underline{s}_i = V\underline{a}_i \tag{103}$$

where the elements of the vector $\underline{s}_i$ are from the ith row of S, and the elements of the vector $\underline{a}_i$ are from the ith row of the matrix resulting from the multiplication UD. Now suppose that the vectors $\underline{s}_i$, i = 1,2,...,n, are the OJT vectors for a signature class. (An interesting reference on this type of application of SVD is [31]). Let $\underline{x}$ be a candidate picked out by the search window, and consider the following overdetermined system of equations:

$$V\ \underline{a} = \underline{x} \tag{104}$$

V is the matrix containing the orthonormal vectors. The pseudoinverse solution $\hat{\underline{a}}$ (i.e., the least squares solution) is given by

$$\hat{\underline{a}} = (V^TV)^{-1}V^T\underline{x} \tag{105}$$

but since $V^TV = I$, the pseudoinverse solution reduces to

$$\hat{\underline{a}} = V^T\underline{x} \tag{106}$$

Consider the least squares error given by

$$E = ||V\hat{\underline{a}} - \underline{x}||^2 = ||V\ V^T\underline{x} - \underline{x}||^2 \tag{107}$$

If $\underline{x}$ is actually one of the training vectors used to find V by means of singular value decomposition, E = 0 since an exact solution exists (see Equation (103)). Otherwise, E > 0. What is hoped for is that if $\underline{x}$ is

in the same class as the training vectors, then E will be smaller than will be the case if $\underline{x}$ is not in this class.

In the experiment using the SVD approach the training set was formed using the standard warping functions in conjunction with SLPF and 64 point standard length stretching. The SLPF parameters used were length = 3, NSKIP = 8, and threshold = 1. Equation (107) was used as a distance measure. The results, based on 100 search problems, were as follows: average CPU time = 4.44 seconds, and percent correct = 51. The results for the SVD method are similar to the results described in the previous sections as far as the success rate is concerned.

Using OJT to Choose SLPF and SRA Parameters

In the initial work on prewarping and direct template matching it was observed that certain SLPF and SRA parameters (NSKIP = 8 and threshold = 1 for both cases) gave the best results for 100 random search problems. In this section a method of using on the job training to automatically select SLPF and SRA parameters for each individual search problem will be considered. The objective is to provide a means of choosing these parameters when confronted with a "real world" signature search problem.

Let S(n) be the given signature after applying either SLPF or SRA and stretching to the standard length N, and let V(k,n), k = 1,2,...,M be the kth warped version created by OJT (again, after prewarping and stretching to the standard length N). Recall that the basic idea of both SLPF and SRA is to make a signature and its warped versions "look like" each other. With this in mind, consider the following error measure:

$$E = \sum_{k=1}^{M} \sum_{n=1}^{N} [S(n) - V(k,n)]^2 \tag{108}$$

If the prewarping filter is doing what it is designed for, E should be small. Therefore, one way to choose "optimum" filter parameters is to try various combinations of NSKIP and threshold (and also filter length in the case of SLPF) and select the combination that gives the smallest value of E.

Error measures other than Equation (108) can be used. Robinson and Treitel [32] describe a normalized "coherency function" for a set of M sequences as

$$S = \frac{2}{M(M-1)} \sum_{i=1}^{M} \sum_{i>k} \frac{R_{ik}(0)}{[R_{ii}(0)R_{kk}(0)]^{1/2}} \tag{109}$$

where $R_{ii}(0)$ is the zero-lag autocorrelation of the ith sequence, and $R_{ik}(0)$ is the zero-lag crosscorrelation of the ith and kth sequences. S = 1 if the sequences are identical. Another possibility described by Robinson and Treitel is the "semblence" coefficient given by

$$S_c = \frac{\sum_{i=1}^{M} \sum_{j=1}^{M} R_{ij}(0)}{M \sum_{i=1}^{M} R_{ii}(0)} \tag{110}$$

S_c = 1 if the sequences are identical.

Figure 43 shows the algorithm used to select SLPF parameters in the experiments. The algorithm for selecting SRA parameters is similar (except that only two nested loops are needed instead of three). Note that some range of possible parameter values must be assumed. The range of values that should be assumed for the threshold depends on the amplitude of the waveforms being filtered, especially in the case of

```
Start
E(minimum) =  (initialize to a large number)
DO LENGTH = 2,4
  DO NSKIP = 6,10,2
    DO THRESH = .5,1.5,.5

        Apply SLPF to the signature S(n) and to each
        warped version V(k,n), k = 1, ..., M.  Stretch
        all sequences to standard length N.

        Calculate  the error (E)

        If ( E. LT. E(minimum) )  THEN

                E(minimum) = E
                Optimum length = LENGTH
                Optimum threshold = THRESH
                Optimum NSKIP = NSKIP

        END IF

    END DO
  END DO
END DO
End
```

Figure 43. Algorithm Flowchart for Automatic Selection of SLPF Parameters

SRA. When this method was tried on real data, the range for the threshold was selected as follows: let $R = S_{max} - S_{min}$, where S_{max} and S_{min} are the maximum and minimum values, respectively, of the signature. The three threshold values used in the loop are R/20, 2R/20, 3R/20. Experiments with automatic parameter selection using real data will be presented shortly.

Table XVII and XVIII show experimental results for automatic SLPF and SRA parameter selection using 100 random search problems. (The window search limits were [4N/5, 5N/4]). In regard to Table XVII, it is interesting to note that all three error measures (Equation (108), (109), and (110)) gave approximately the same results. Compared to the percent correct figures obtained using parameter "twiddling," the numbers in Tables XVII and XVIII are somewhat disappointing; it was hoped that since the parameters were selected for each individual search problem the percent correct figures would improve. Nevertheless, it is believed that the objective of providing a means of choosing the parameters automatically has been met. In regard to CPU times, note that the automatic selection of SLPF parameters is much slower than is the case for SRA parameters. This is because there are 3 SLPF parameters to select, but only 2 SRA parameters. Table XVII also includes the results obtained by fixing one of the 3 SLPF parameters. The CPU time requirement is reduced, and in one case (fixing NSKIP at 8) the success rate showed improvement (51%) over the result obtained by adjusting all 3 parameters. It should also be noted in passing that the SLPF parameter selection algorithm sequentially tests 27 combinations of parameters. Since no one test depends on any of the others, here is a theoretical opportunity for parallel processing. All 27 tests could be

TABLE XVII

RESULTS: AUTOMATIC SELECTION OF SLPF PARAMETERS WITH OJT

Number of Training Vectors	Error Measure	Average CPU Time	Percent Correct
7	Eqn.(108)	2.37	46
10	Eqn.(108)	3.38	41
7	Eqn.(109)	2.98	45
7	Eqn.(110)	2.99	47
Best results using parameter "twiddling"			
		.082	54
Results obtained by automatic selection of 2 of the 3 parameters			
7	eqn.(108)	.855	45 (Length fixed)
7	eqn.(108)	.917	51 (NSKIP=8 fixed)

7 training vectors: standard warping function OJT
10 training vectors: segment and random warp OJT

TABLE XVIII

RESULTS: AUTOMATIC SELECTION OF SRA PARAMETERS WITH OJT

Number of Training Vectors	Error Measure	Average CPU Time	Percent Correct
7	eqn.(108)	.436	40
Best results using parameter "twiddling"			
		.054	49

run simultaneously, i.e., in parallel. A similar argument can of course be made for the SRA parameter selection scheme.

Table XIX shows the results obtained when the SLPF parameters in the "hybrid" signature recognition scheme are selected using OJT (with NSKIP = 8 fixed). (This method uses SLPF based direct template matching to select M preliminary candidates and then uses dynamic programming warping to make the final selection. A more detailed explanation can be found in the section entitled "Experimental Results With Direct Template Matching"). The percent correct figures are significantly improved over those shown in Table XVII, but of course there is a price to be paid in terms of CPU time. Nevertheless, when 10 preliminary candidates are selected, the CPU time requirement is almost an order of magnitude less than that required using dynamic programming warping (with SRA based data reduction using Itakura's method with the L2 distance measure), and the percent correct figure increases from 66 percent to 72 percent. This "hybrid" method is clearly promising.

In the section describing the SRA and SLPF algorithms some differences between the two methods were noted (see Figure 37). It was decided to run additional experiments comparing the two prewarping methods in conjunction with the automatic parameter selection routines. The next section considers these experiments, and also includes results obtained using real well log data.

SLPF and SRA: More Experimental Results

In Chapter II the random search problem generator was described in detail. It was noted there that the "blocky" logs are lowpass filtered (see Figures 10 and 11). In all of the work described to this point the

TABLE XIX

EXPERIMENTAL RESULTS FOR THE "HYBRID" METHOD

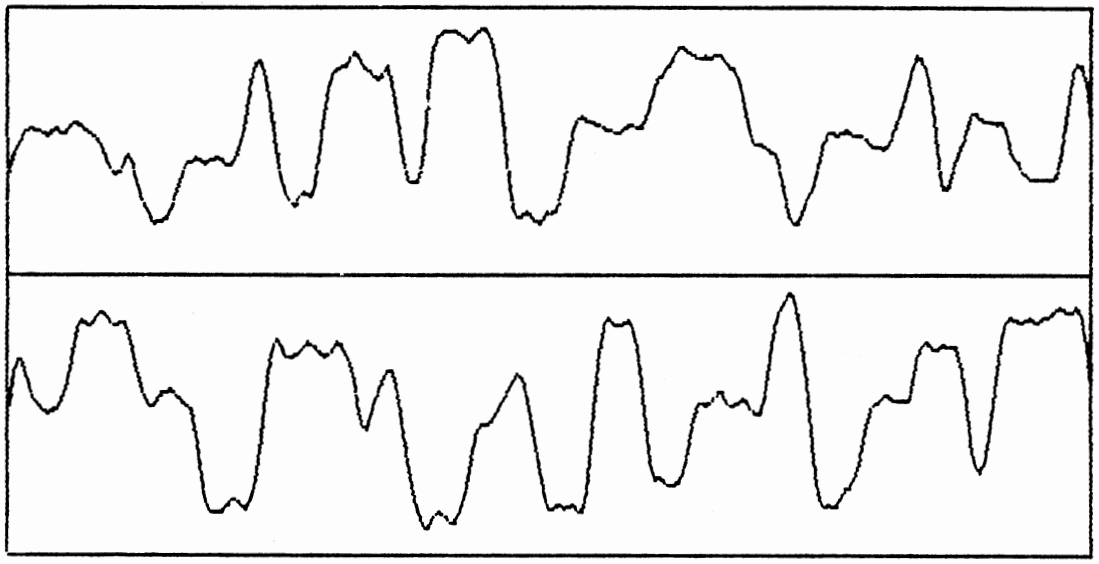
Number of Preliminary Candidates (M)	Average CPU Time	Percent Correct
5	1.54	68
10	2.18	72
15	2.75	72
25	3.99	78

(number of training vectors = 7; error measure: Eqn.(108)

same lowpass filter cutoff frequency (MCUT = 60) was used in this generator. (As noted in Chapter II, the cutoff frequency is the point on the 256 point DFT spectrum where the lowpass shape starts to roll off.) If the cutoff frequency is lowered, the slopes observed on these logs are changed, i.e., the logs become even less "blocky." Figure 44 shows how the "character" of a log is changed by decreasing the cutoff frequency (results are shown for MCUT values of 60, 50, 40, and 30). It was decided to see how lowering the cutoff frequency affects the results for direct template matching using SLPF or SRA. In particular it was desired to find out if changing the model in this manner would reveal any dramatic differences between SRA and SLPF. The results (shown in Table XX) are for the most part inconclusive. However, it is interesting to note that the results for MCUT = 60 and MCUT = 40 are practically reversed as far as the percent correct figures are concerned.

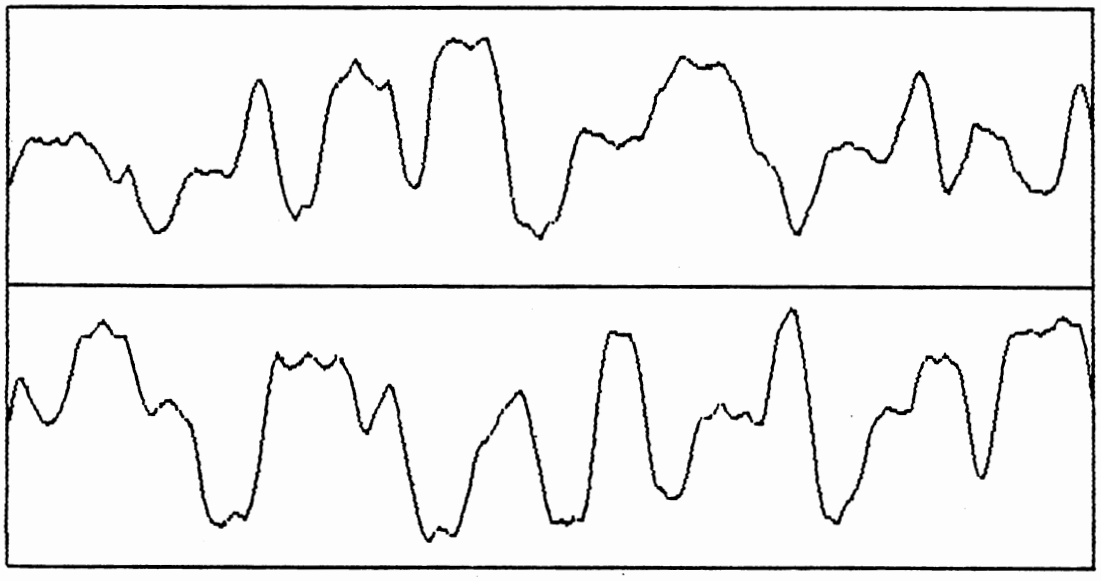
Results using real log data will now be presented. Once again, the real example is the gamma ray logs first shown in Chapter I (Figure 4). Figure 45 shows the results for direct template matching with SLPF with a automatic selection of SLPF parameters. Figure 46 shows the results using SRA. The OJT method was based on the standard warping function approach. Average value subtraction was used for level shifting. The results are good (fit = 0.7 for both), but as with other results shown for this data, the algorithms have difficulty with the ill-defined top bed boundary.

As noted in the section entitled "experimental results using direct template matching," the composite success rate for SLPF and SRA (using the best parameters for each) was 74%. It was therefore decided to carry the automatic selection of SRA and SLPF parameters one step



(a)

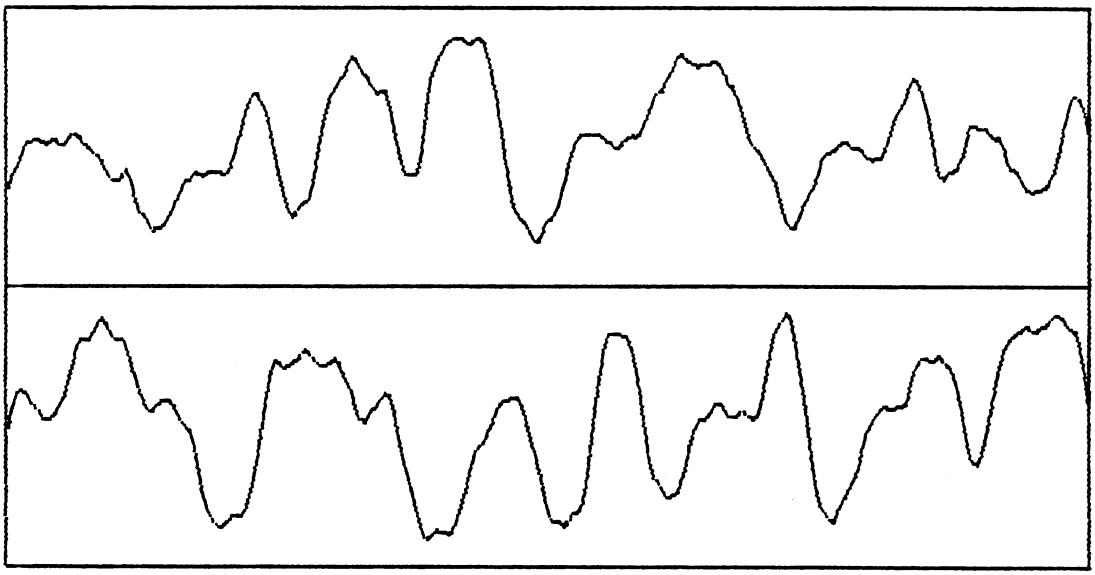
MCUT = 60



(b)

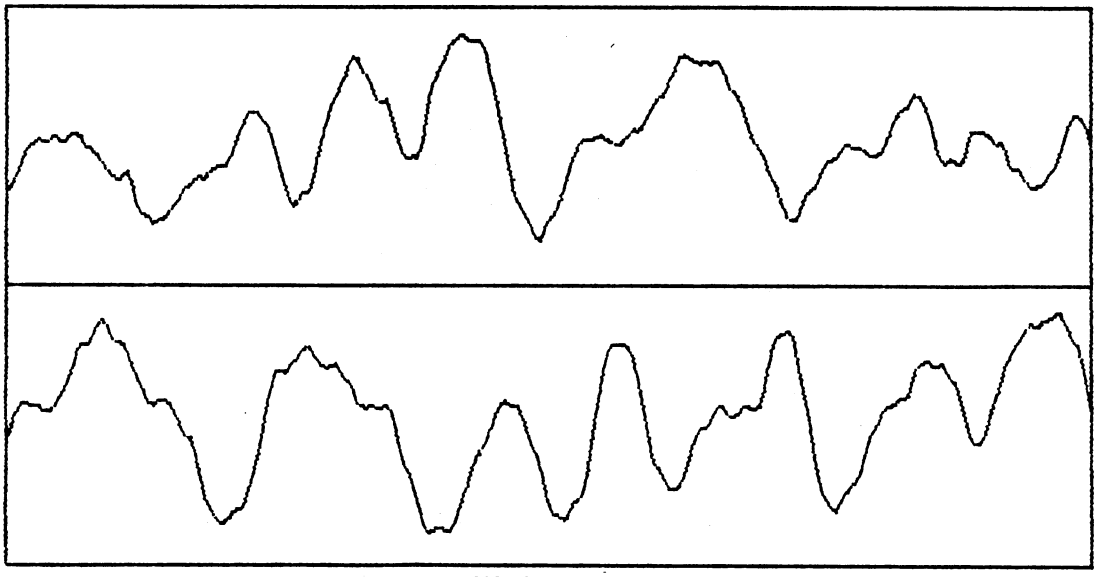
MCUT = 50

Figure 44. Effect of Changing the Lowpass Filter Parameter MCUT



(c)

MCUT = 40



(d)

MCUT = 30

Figure 44. Continued

TABLE XX

SRA AND SLPF COMPARED

Number of Training vect.	Lowpass filt. MCUT *	Prewarping Filter	Average CPU Time	Percent Correct
7	60	SLPF	2.34	46
7	60	SRA	.44	40
7	50	SLPF	2.40	52
7	50	SRA	.44	47
7	40	SLPF	2.27	39
7	40	SRA	.44	47
7	30	SLPF	2.17	28
7	30	SRA	.43	32

*MCUT is the lowpass filter parameter in the random log generator. See also Figure 44.

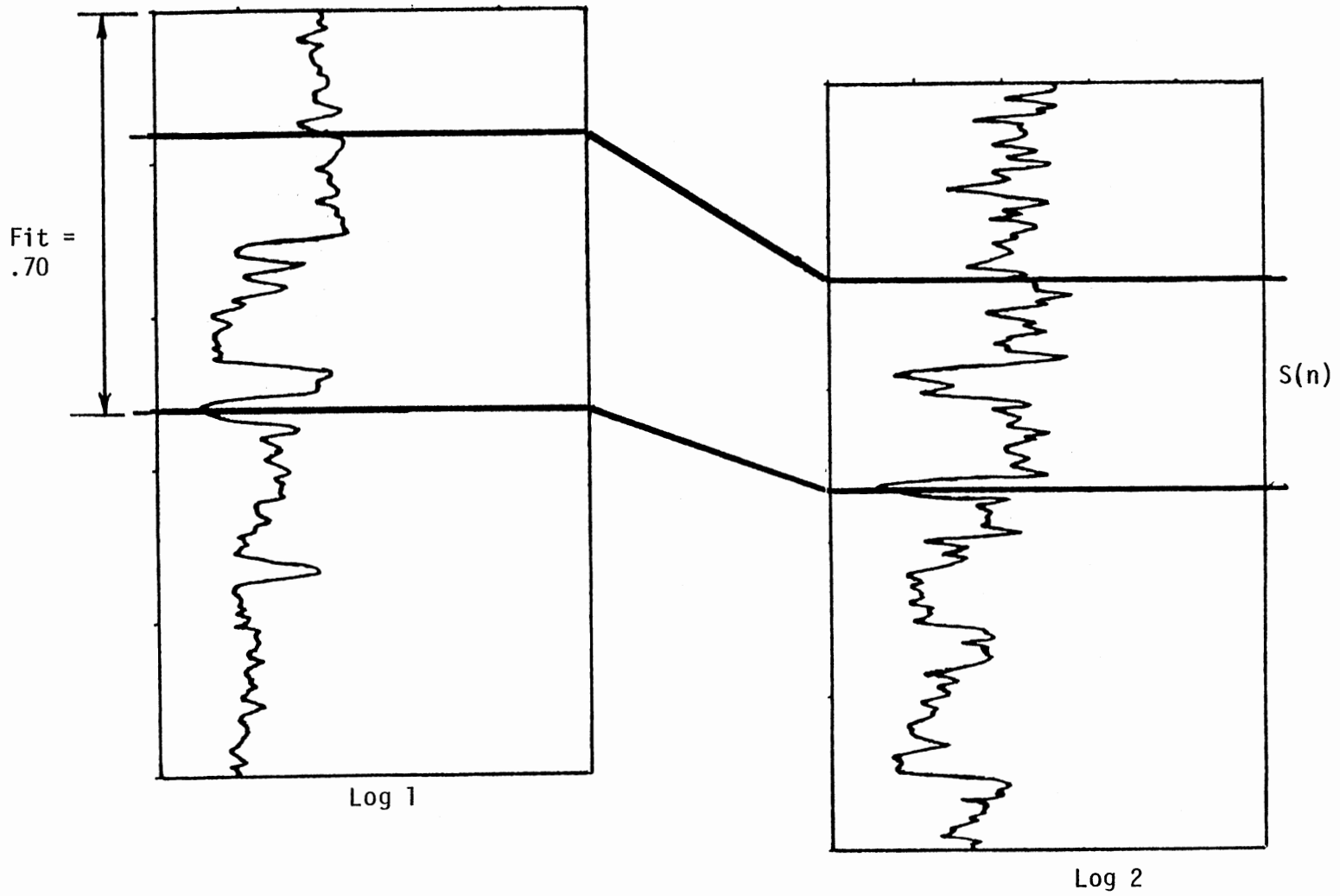


Figure 45. Signature Recognition Using Direct Template Matching with SLPF on Gamma Ray Logs

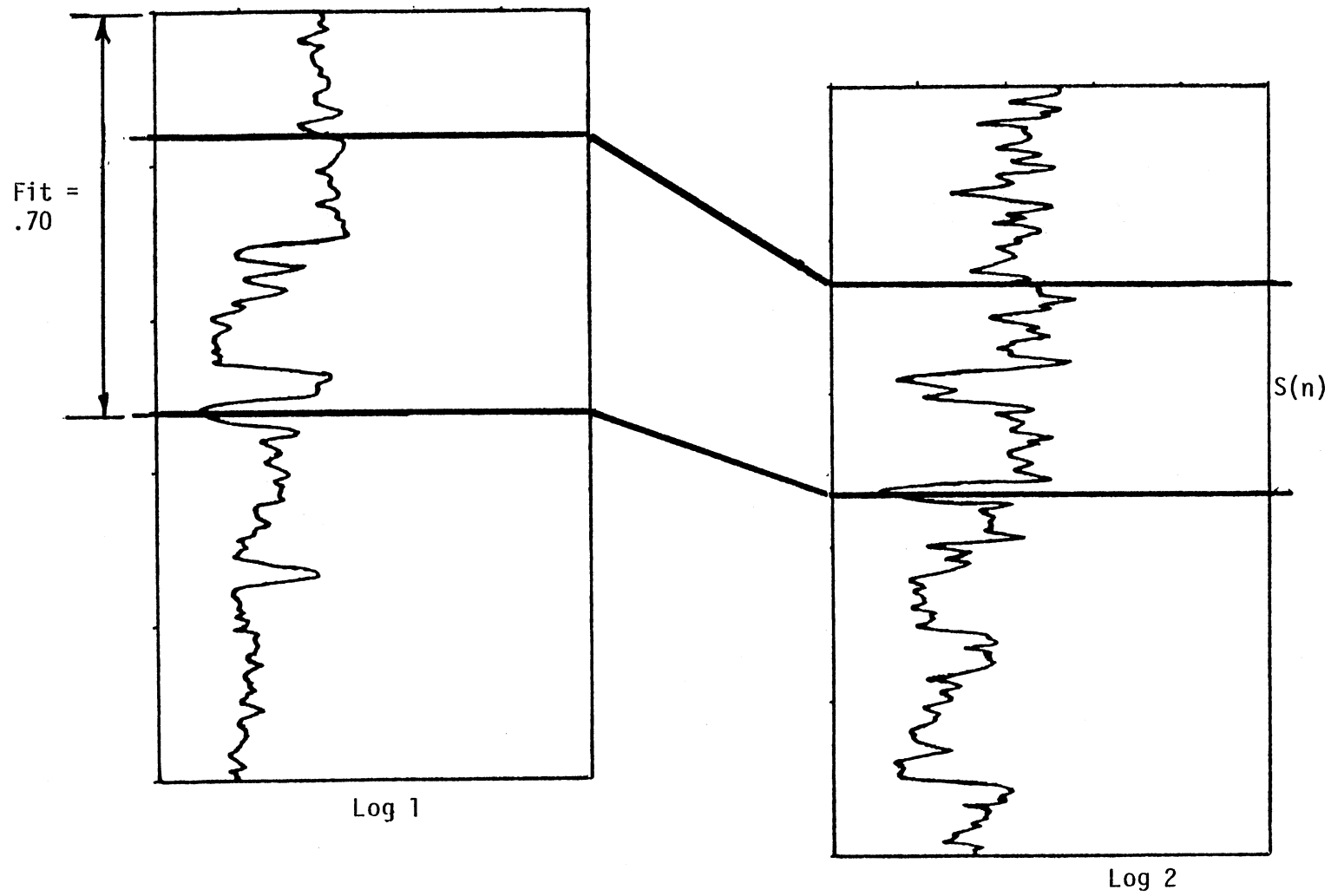


Figure 46. Signature Recognition Using Direct Template Matching with SRA on Gamma Ray Logs

further: for each random search problem, use OJT to evaluate the error (Equation (108)) for both SLPF and SRA and then use the prewarping method that yields the lowest error. In the process of deciding which to use, automatic parameters selection also occurs. Unfortunately, the experimental result for 100 search problems was only 40% correct -- no better than using SRA alone, and worse than using SLPF alone. Nevertheless, the fact that the composite success rate was 74% is a strong reason for believing that a good way of combining these methods ought to exist. More work is needed in this area.

This concludes the discussion of applications for on the job training. However, before moving on to the last chapter, there is one more base that needs to be touched: the use of DFT magnitude coefficients and linear prediction coefficients as pattern vector features. These coefficients have been found to be useful in a variety of signal processing applications, so the question of their usefulness in the well log signature recognition problem naturally arises.

Using DFT Magnitude Coefficients and Linear Prediction Coefficients as Features

Discrete Fourier transform magnitude coefficients and linear prediction coefficients (LPC) are sometimes mentioned in the literature as being useful features for pattern recognition. (Linear prediction coefficients have been extensively used for speaker recognition with excellent results.) This section describes two experiments with well log signature recognition using these features.

Both experiments used SLPF for prewarping, with parameters length = 3, NSKIP = 8, and threshold = 1. The search window limits were [4N/5,

5N/4]. In both cases, SLPF and standard length stretching (64 points) was applied to the signature, and the parameters (DFT magnitude or linear prediction coefficients) were extracted. SLPF was then applied to the entire log being searched, and candidates were selected by the search window. Each candidate was stretched to the standard length, and the appropriate parameters were extracted. Let $\underline{a}_S$ represent the parameters for the signature, and let $\underline{a}_C$ represent the parameters for the candidate. The distance measure used was the L2 distance:

$$D = (\underline{a}_S - \underline{a}_C)^T(\underline{a}_S - \underline{a}_C) \tag{111}$$

For the experiment with DFT magnitude coefficients, the feature vector consisted of the magnitude of the first 10 coefficients from a 64 point DFT. For the linear prediction experiment, a 10th order model was used. Experimental results for 100 random search problems were as follows: 39% correct for the DFT magnitude coefficients, and 14% correct for the linear prediction coefficients. The average CPU times were 2.05 and .742 seconds, respectively. The results do not compare favorably with those obtained using the first 10 (complex) DFT coefficients (see Table XII), which shows that the phase information is important for the pattern recognition problem being dealt with here. (Phase information is lost when using DFT magnitude coefficients or linear prediction coefficients.) Figure 47 illustrates why phase information is important. Both waveforms have the same autocorrelation (and hence the same DFT magnitude and the same linear prediction coefficients), but they are obviously not in the same signature class. Therefore, a signature search method based on either of these phase-destroying parameters may very well find an "answer" which is an excellent match in

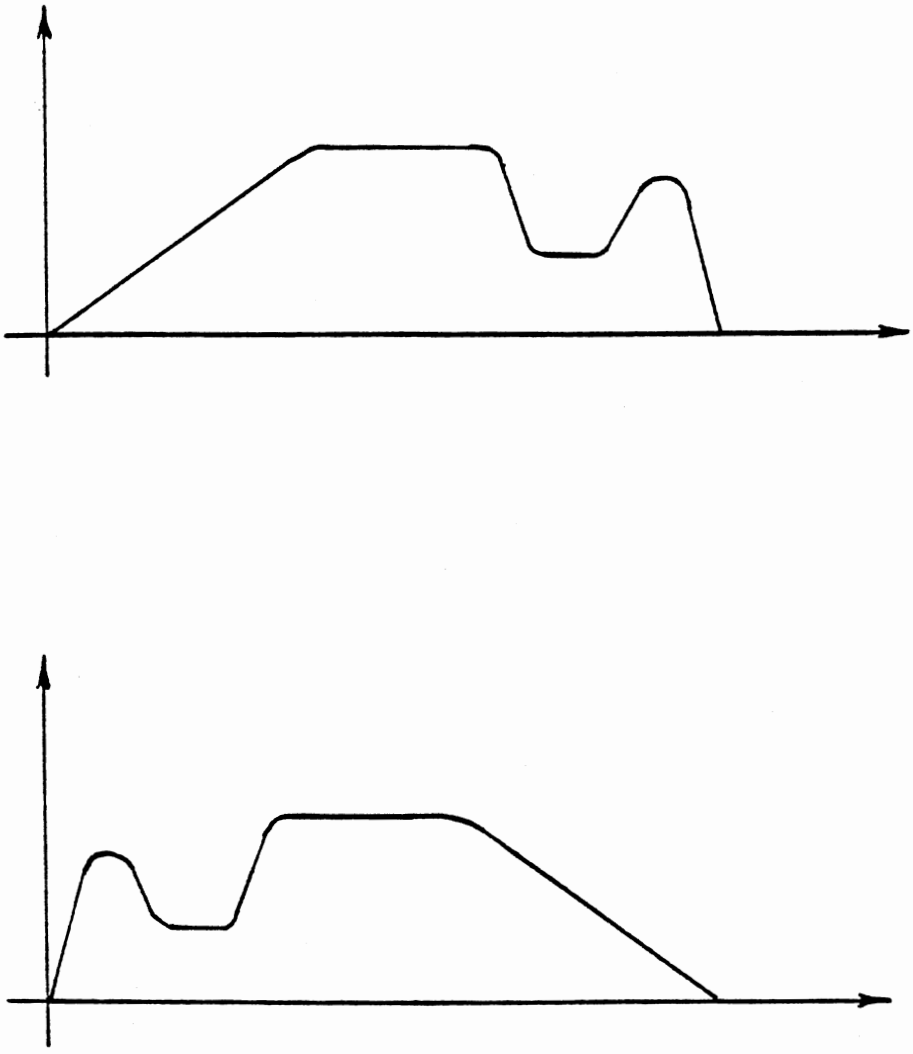


Figure 47. Different Classes, but same DFT Magnitude

terms of those parameters and yet is far off the mark in terms of what is really being searched for. However, it would be hasty to conclude that LP and/or DFT magnitude coefficients are useless for the well log signature recognition problem. They may be useful when used in conjunction with other methods. More work needs to be done in this area.

CHAPTER V

CONCLUSIONS

Several different proposed solutions to the well log signature recognition problem have been presented. The signature recognition problem considered herein is essentially a pattern recognition problem involving waveform shapes. Experimental results based on real well log data have been presented, but for the most part the signature recognition algorithms have been evaluated based on 100 problems created by a novel random synthetic well log signature recognition problem generator. It is believed that the random problem generator provides a means of objectively testing signature recognition techniques, although there is no doubt that there is room for improvement here. The major experimental results are summarized in Table XXI.

The reader should remember when looking at the percent correct figures in Table XXI that (1) the "simple search method" based on an assumption of uniform warping scores zero percent correct on these 100 problems, and (2) the random problem generator often creates problems which are extremely difficult. In fact, industry representatives at the annual meeting of the Oklahoma State University Research Consortium on Well Log Data Enhancement via Signal Processing have remarked that these simulated signature recognition problems are more difficult than those generally encountered in the "real world." If this is indeed the case,

TABLE XXI

SUMMARY OF RESULTS BASED ON 100 RANDOM PROBLEMS

Method	Av. CPU Time	Percent Correct
DYNAMIC PROG. WARPING (L2)		
Segmentation with Itakura's method, fixed parameters (twiddling)	8.0	89
Segmentation with Itakura's method, auto. parameter selection	8.7	72
Data reduction (SRA) with Itakura's method, fixed SRA parameters	16.0	66
(with L1 distance meas.	16.0	76)
DIRECT TEMPLATE MATCHING		
SRA, "twiddling" parameters	.054	49
SLPF, "twiddling" parameters	.082	54
SRA, OJT parameter selection	.436	40
SLPF, OJT parameter selection, (with NSKIP=8 fixed)	.917	51
HYBRID METHODS		
SLPF/DPW with 10 preliminary candidates (using "twiddling" SLPF parameters)	1.32	69
SLPF/DPW with 10 preliminary candidates (SLPF parameters selected with OJT)	2.18	72
STATISTICAL (operating directly on 64 point sequences) **		
Euclidean distance from class centroid	.27	53
Clustering transformation	.29	38
STATISTICAL (Walsh transform domain; 10 element vectors) **		
Euclidean dist. from class centroid	1.80	54
Clustering transformation	1.78	52
Mahalanobis distance	2.89	36
PDF estimation (20 sinusoids)	3.05	45
OTHER		
Singular value decomposition **	4.44	51
"Simple method" based on the assumption of uniform warping	-	0

** using SLPF with "twiddling" parameters

percent correct figures in the vicinity of 50 percent can plausibly be viewed as good results.

Direct template matching based on straight line prediction filtering (SLPF) or sample rate adjustment (SRA) shows great promise because of its speed. Of the two prewarping filter algorithms considered, SLPF seems to have the most merit. It should be pointed out that both SRA and SLPF are special cases of nonuniform decimation algorithms; there is plenty of room for creativity here in terms of designing different filters. However, it must be admitted that since these methods are heuristic in nature there seems to be no systematic way to go about discovering good ones.

Dynamic programming warping in conjunction with speedup techniques (segmentation or data reduction) has been demonstrated to be a viable approach to the signature recognition problem. This method outperforms direct template matching in terms of the percent correct figures summarized in Table XXI, but it is significantly slower. More work is needed to refine these speedup techniques.

The signature recognition methods based on statistical pattern recognition are disappointing since the percent correct figures obtained were in no case improved over the best case observed for SLPF based direct template matching. However, the possibility exists that if additional constraints were imposed on the "on the job training" (OJT) scheme presented in this work -- constraints on the type of warping allowed based on geological knowledge -- the statistical pattern recognition methods would fare better.

The "on the job training" scheme is clearly useful for automatic selection of prewarping filter parameters. Of course, there is plenty

of room for improvement of the parameter selection process since the percent correct figures obtained using this method (in conjunction with direct template matching) are lower than the best figures obtained by simply "twiddling" these parameters.

The hybrid technique combining dynamic programming warping with SLPF based direct template matching appears to have special promise since compared to dynamic programming warping operating alone (Itakura's method in conjunction with SRA based data reduction, using the L2 distance measure) the results were improved in terms of both success rate and CPU time requirements. These improvements were noted with both SLPF parameter twiddling and SLPF parameter selection by means of "on the job training."

In the next section, two possible extentions of this work are briefly explored: the application of prewarping filters to vector sequences and the use of prewarping filters in the point-to-point correlation of waveforms.

Suggestions for Future Research

The application of nonlinear prewarping filters to vector sequences is one possible extention of the work described in this dissertation. Consider a vector signature $\underline{s}(n)$ and a vector candidate sequence $\underline{x}(n)$:

$$\underline{s}(n) = \begin{bmatrix} S_1(n) \\ S_2(n) \\ \vdots \\ S_M(n) \end{bmatrix} \qquad \underline{x}(n) = \begin{bmatrix} X_1(n) \\ X_2(n) \\ \vdots \\ X_M(n) \end{bmatrix} \tag{112}$$

The distance between $\underline{s}(n)$ and $\underline{x}(n)$ could be defined as

$$D = \sum_{k=1}^{M} \sum_{j=1}^{N} ||S_k(j) - X_k(j)||^p \tag{113}$$

There is one note of caution here: one would want to normalize the component sequences in some sense so that the error measure is not dominated by a minority of them. (Of course, one could deliberately choose to weight some of the component sequences more heavily than others).

An example of such a vector sequence is one where each component sequence corresponds to a different type of log (gamma ray, resistivity, sonic, neutron, spontaneous potential, etc.) Another example is where the component sequences represent time varying speech waveform parameters, e.g., short time energy, pitch, formant frequencies, linear prediction coefficients, etc. [27]. The warping phenomenon arises in problems in speaker recognition because a speaker has difficulty speaking at the same rate each time he utters a reference phrase. Since dynamic programming warping is usually used to solve the time alignment problem, there is a potential application for nonlinear prewarping filters in speech analysis.

There are two possible ways to apply nonlinear prewarping filtering to a vector sequence. The first method is to apply filtering to each individual component sequence, perhaps allowing the filter parameters to be different for each case. Since the resulting component sequences would usually be of different lengths, they would all have to be stretched to some standard length to create a filtered version of the overall vector sequence. The second method is to apply filtering to the vector sequence as such. The extension of sample rate adjustment to the vector case is fairly simple: the distance from the reference point $\underline{s}(i)$ to the point under consideration $\underline{s}(i+k)$ could be defined as

$$D = \sum_{j=1}^{M} ||S_j(i+k) - S_j(i)||^p \tag{114}$$

As with the original SRA algorithm, the decision to discard $\underline{s}(i+k)$ is based on the parameter NSKIP and threshold. The extension of straight line prediction filtering to the vector case depends on fitting a straight line to points in (M+1) space. The decision whether or not to keep point $\underline{s}(k)$ is made by fitting a straight line to points $\underline{s}(k-N)$, $\underline{s}(k-N+1)$, ..., $\underline{s}(k-1)$, where N is the filter length. Let $\hat{\underline{s}}(k)$ denote the predicted value of $\underline{s}(k)$. The prediction error is given by

$$E = [\hat{\underline{s}}(k) - s(k)]^T[\hat{\underline{s}}(k) - \underline{s}(k)] \qquad (115)$$

As with the original SLPF algorithm, the decision to discard $\underline{s}(k)$ is based on whether or not E is less than some specified threshold.

Applying the filtering to each component sequence individually is a very flexible scheme since it allows for the fact that in some applications the component waveforms may not track each other very well, and may in fact have a distinctly different character. (See, for example, Figure 9.15 in [27], which shows how speech derived linear prediction coefficients vary with time). However, it has one serious drawback: the correspondence arrays for the individual sequences will be different. (A correspondence array is a record of the point mappings from the original sequence to the filtered sequence.) Consider the well log signature recognition problem discussed in this dissertation. SRA or SLPF is applied to the log being searched, and then candidates X(n) are extracted with a sliding window and compared to the SRA/SLPF reduced signature. When the best such candidate is found, the corresponding section on the original (i.e., unfiltered) log must be determined; the correspondence array is used for this purpose. But if the original log is a vector of logs with individual correspondence arrays for each

component sequence, each correspondence array will point to a different set of boundary points on the original log. A possible solution to this problem would be to select candidates from the original (unfiltered) log (thus making the question of correspondence trivial) and apply prewarping filtering to candidates as they are selected on an individual basis. However, there would be a high price to be paid for this solution in terms of increased CPU time requirements.

The correspondence problem does not arise in applications where the following problem is posed: given a test sequence $\underline{s}(n)$, find the best match from a set of previously stored reference sequences $\{\underline{x}_i(n)\}$, $i = 1,2,\ldots,K$. In this case there is no sliding window search to perform. An example of such an application is automatic speaker recognition.

Another possible approach to the problem of signature recognition on a vector of logs is to use the Karhunen-Loeve transform to derive "principle component" logs [33]. The first principle component log has a great deal of information common to all of the original logs; therefore, this transformation can perhaps be used to convert a vector signature recognition problem to one involving only one individual sequence.

Another possible extension of this research is the application of prewarping filters to point-to-point correlation of waveforms. An example of the kind of problem where this is done is the analysis of diplog data [12,34]. By "point-to-point correlation" it is meant that the "significant" points in the waveforms are selected, and then the matching is done in terms of these points only. Applying dynamic programming warping after using SRA or SLPF for data reduction is in reality an example of such an operation. However, in this research little

attention was paid to the actual warping function produced by the dynamic programming warping algorithm; the goal was to obtain the error measure for the matchup of the signature and a given candidate. But if the warping function is extracted from the dynamic programming procedure, it can be used in conjunction with the correspondence arrays for the filtered sequences and their original versions to plot a point-to-point correlation such as shown in Figure 48.

The two sequences shown in Figure 48 are 256 point "random logs" created by a modified version of the random log generator discussed in detail in Chapter II. The generator starts by creating a "blocky" log, as shown in Figure 9. However, instead of extracting a signature, warping it, and building another random log around it, the second "blocky" log is a warped version of the first one. Both "blocky" logs are then lowpass filtered and corrupted by noise as before.

For a typical point-to-point correlation application, researchers are interested in a small set of selected points -- that is, it is not desirable to have the picture cluttered up by minor details. This suggests that the way to choose the filter parameters is to select a range for the desired number of points in the filtered sequence and then iteratively adjust the appropriate parameters until this goal is met. The results shown in Figure 48 were obtained with straight line prediction filtering with parameters length = 3 and NSKIP = 10; the threshold parameter was adjusted until the number of points was between 60 and 70 (out of the original 256). The black dots on the waveforms of Figure 48 are the significant points selected in this manner. The range chosen dictated that roughly 25 percent of the points are "significant." The points selected in this manner are not necessarily those

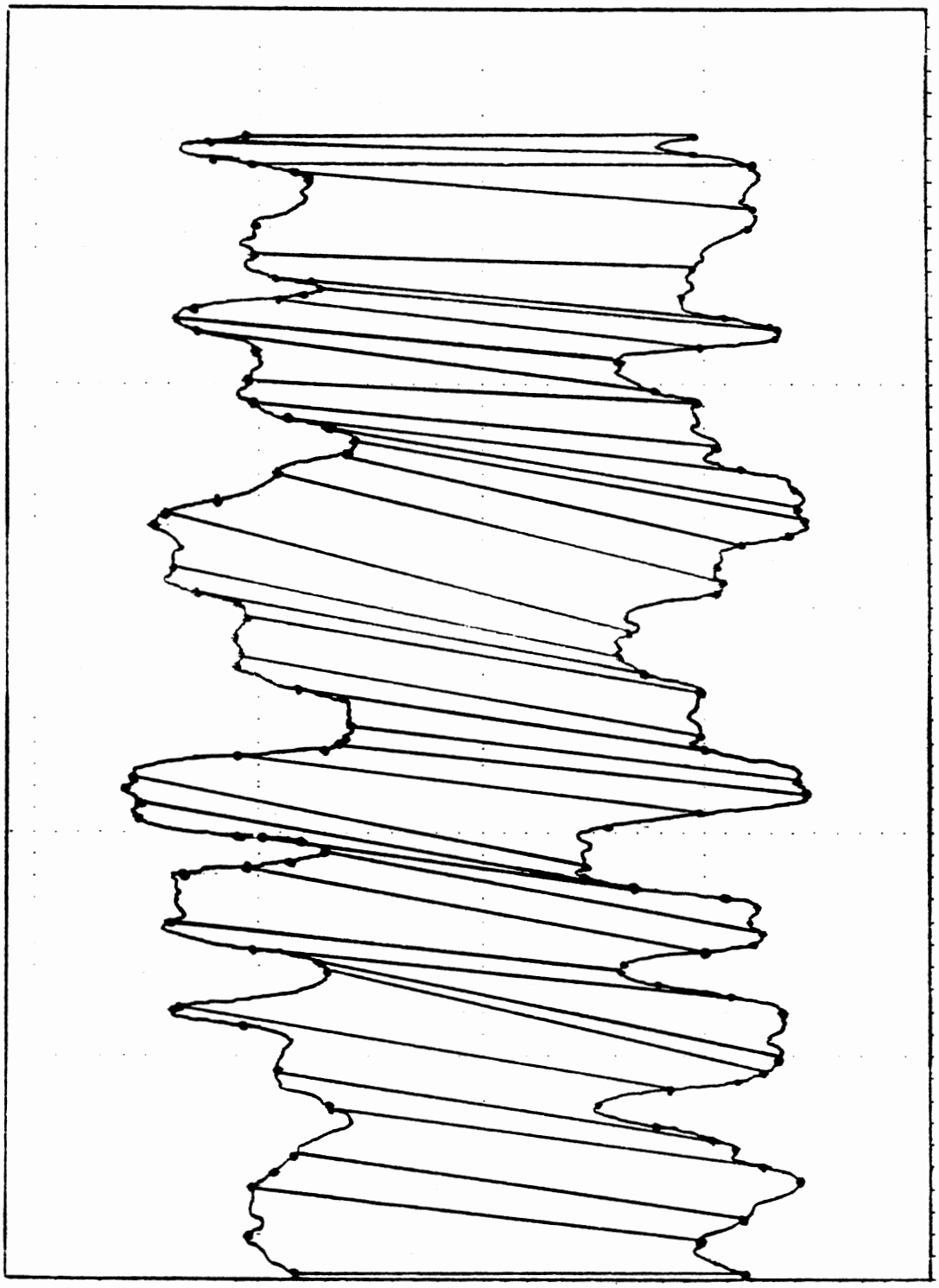


Figure 48. Point to Point Correlation

one would select by visual inspection. In this connection, it should again be remarked in passing that nonlinear prewarping filters other than SRA or SLPF could probably be designed; no claim is made here that the ad-hoc designs presented in this work are the final word in nonlinear prewarping filters. Nor is there anything magical about the figure 25 percent; it just so happens that this choice led to reasonable results for this example. Furthermore, the possibility exists that the usefulness of nonlinear prewarping filters for the selection of "significant points" could be enhanced by first performing some other type of preprocessing on the logs, such as lowpass filtering.

Having selected the significant points, the next step is to match the two sequences of significant points using dynamic programming warping. For this example, Itakura's method was used to warp the top sequence of Figure 48 to fit the bottom sequence. Reconstruction of the warping function revealed the point-to-point mappings shown in Figure 49. Observe that at this stage the picture is cluttered by multiple mappings, i.e., cases where one point on the top sequence is mapped to two different points on the bottom sequence. Such multiple mappings need to be resolved in some manner to "clean up" the results. It was decided to keep the mapping with the smallest mapping error. That is, if X(k) is mapped to both Y(i) and Y(j), the selection is made on the basis of the error terms given by

$$\begin{aligned} E_1 &= ||X(k) - Y(i)||^2 \\ E_2 &= ||X(k) - Y(j)||^2 \end{aligned} \tag{116}$$

The picture can be cleaned up even more by "thinning out" mappings from points on the top sequence which are within one point of each other.

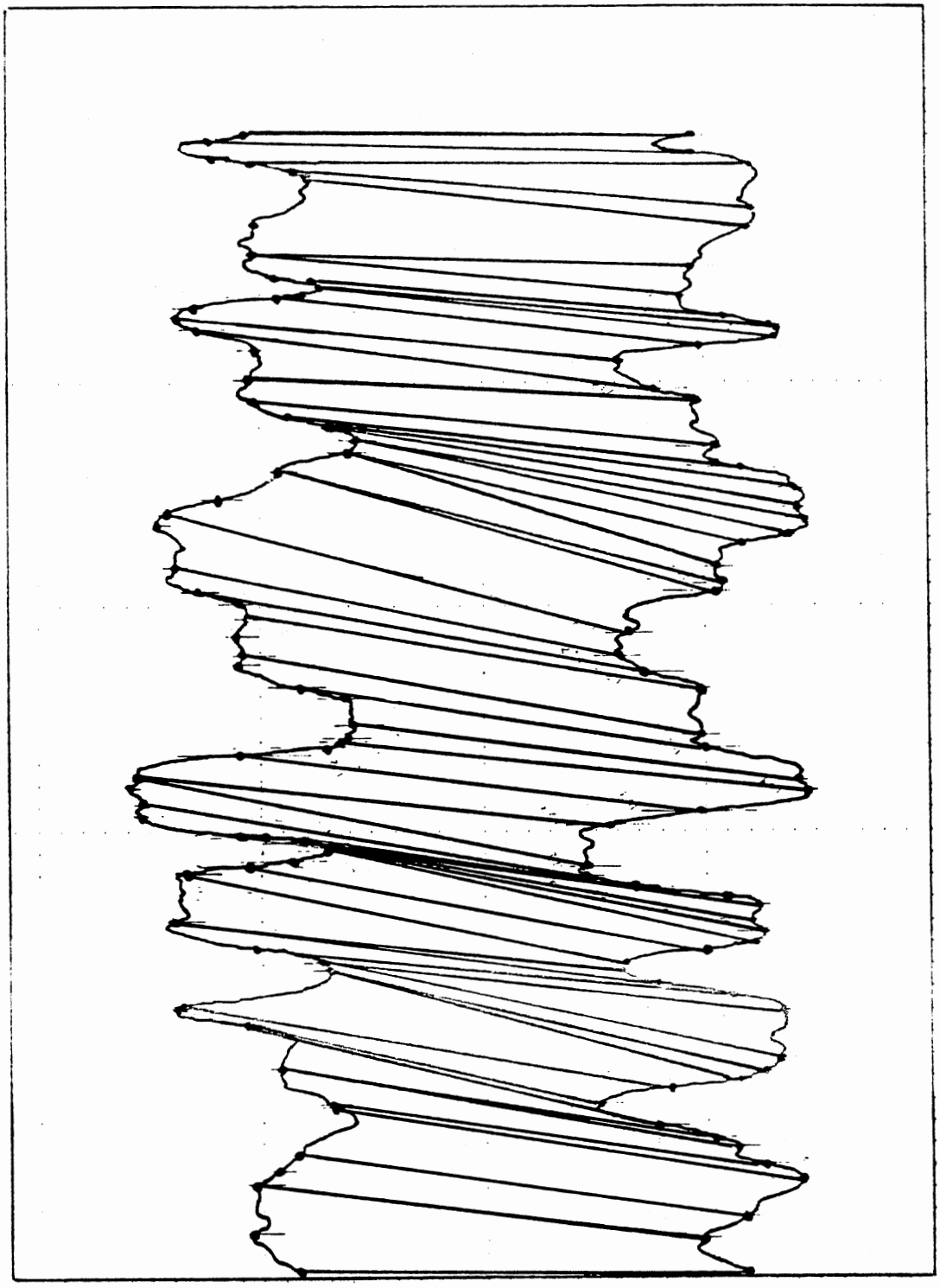


Figure 49. Point to Point Correlation Before "Cleanup"

For example, if both X(k) and X(k+1) are mapped, keep only the mapping with the smallest error. Figure 48 shows the final point-to-point correlation after employing the "cleanup" methods described above. For some individual points there are mappings which disagree with the result dictated by visual inspection, but overall it is thought that the results shown are good enough to recommend this as a topic for future research.

REFERENCES

[1] J.E. Robinson, "Transforms to Enhance Correlation of Mechanical Well Logs." Mathematical Geology, V.7, No. 4, 1975, pp. 323-335.

[2] G.H. Grandlund, "Fourier Preprocessing for Hand Print Character Recognition." IEEE Trans. on Computers, Feb. 1972, pp. 195-201.

[3] Charles T. Zahn and Ralph Z. Roskies, "Fourier Descriptors for Plane Closed Curves." IEEE Trans. on Computers, March 1972, pp. 269-181.

[4] Albert J. Rudman and Robert F. Blakely, "Fortran Program for Correlation of Stratigraphic Time Series." Indiana Geological Survey Occasional Paper 14, 1976.

[5] Albert J. Rudman and Robert W. Lankston, "Stratigraphic Correlation of Well Logs by Computer Techniques." American Association of Petroleum Geologists Bulletin, V. 57, 1973, pp. 577-588.

[6] Byung-Doo Kwon, Robert F. Blakely, and Albert J. Rudman, "Fortran Program for Correlation of Stratigraphic Time Series (Part 2. Power Spectral Analysis)." Indiana Geological Survey Occasional Paper 26, 1978.

[7] Byung-Doo Kwon and Albert J. Rudman, "Correlation of Geologic Logs with Spectral Methods." Mathematical Geology, V. 11, No. 4, 1979, pp. 373-390.

[8] Hwei P. Hsu, Applied Fourier Analysis. Hartcourt Brace Javonovich, 1984, pp. 76-77.

[9] Franklin Kemp, "An Algorithm for the Stratigraphic Correlation of Well Logs." Mathematical Geology, V. 14, No. 3, 1982, pp. 271-285.

[10] Joseph B. Kruskal, "An Overview of Sequence Comparison: Time Warps, String Edits, and Macromolecules." SIAM Review, Vol. 25, No. 2, April 1983, pp. 201-237.

[11] Kenneth R. Anderson and James E. Gaby, "Dynamic Waveform Matching." Information Sciences, V. 31, 1983, pp. 221-242.

[12] Mark G. Kerzner, "Formation Dip Determination -- An Artifical Intelligence Approach." The Log Analyst, V. 24, No. 5, pp. 10-22.

[13] Mark G. Kerzner, "A Solution to the Problem of Automatic Depth Matching." SPWLA 25th Annual Logging Symposium, 1984.

[14] F. Itakura, "Minimum Prediction Residual Principle Applied to Speech Recognition." IEEE Trans. Acoustics, Speech, and Signal Proc., Feb. 1975, pp. 67-72.

[15] H. Sakoe and S. Chiba, "Dynamic Programming Algorithm for Spoken Word Recognition." IEEE Trans. Acoustics, Speech, and Signal Proc., Feb. 1978, pp. 43-49.

[16] A.D. Gordon and R.A. Reyment, "Slotting of Borehole Sequences." Mathematical Geology, V. 11, No. 3, 1979, pp. 309-327.

[17] Yao-Chou Cheng and Shin-Yee Lu, "Waveform Correlation by Tree Matching." IEEE Trans. Pattern Anal. and Mach. Intel., May 1985, pp. 299-305.

[18] Kenneth R. Anderson, "Syntactic Analysis of Seismic Waveforms Using Augmented Transition Network Grammars." Geoexploration, V. 20, 1982, pp. 161-182.

[19] Donald E. Kirk, Optimal Control Theory. Prentice Hall, 1970, p. 54.

[20] David J. Burr, Bryan D. Ackland, and Neil Weste, "Array Configurations for Dynamic Time Warping." IEEE Trans. Acoustics, Speech, and Signal Proc., Feb. 1984, pp. 119-128.

[21] T. Pavlidis, "Waveform Segmentation through Functional Approximation." IEEE Trans. Computers, July 1973, pp. 689-697.

[22] D.M. Hawkins and D.F. Merriam, "Optimal Zonation of Digitized Sequential Data." Mathematical Geology, V. 5, No. 4, 1973, pp. 389-395.

[23] Andrew P. Witkin, "Scale-Space Filtering: A New Approach to Multi-Scale Description." Proceedings, IEEE International Conference on Acoustics, Speech, and Signal Processing, 1984, pp. 39A.1.1-39A.1.4.

[24] Andrew F. Blumenthal, Larry S. Davis, and Azriel Rosenfeld, "Detecting Natural 'Plateaus' in One-Dimensional Patterns." IEEE Trans. on Computers, Feb. 1977, pp. 178-179.

[25] R. Webster, "Automatic Soil-Boundary Locations from Transect Data." Mathematical Geology, V. 5, No. 1, 1973, pp. 27-37.

[26] Mark G. Kerzner, "An Analytical Approach to Detailed Dip Determination Using Frequency Analysis." SPWLA 23rd Annual Logging Symposium, 1982.

[27] Lawrence R. Rabiner and Ronald W. Schafer, Digital Processing of Speech Signals. Prentice Hall, 1978.

[28] Julius T. Tou and Rafael C. Gonzales, Pattern Recognition Principles. Addison-Wesley, 1974.

[29] Bricker, et. al., "Statistical Techniques for Talker Identification." Bell System Tech. Journal, V. 50, No. 4, April 1971, pp. 1427-1454.

[30] T.W. Anderson, Introduction to Multivariate Statistics. John Wiley and Sons, 1958.

[31] William Menke, Geophysical Data Analysis: Discrete Inverse Theory. Academic Press, 1984.

[32] Enders A. Robinson and Sven Treitel, Geophysical Signal Analysis. Prentice Hall, 1980.

[33] Robert G. Hayes, "The Use of Wireline Logs in Lithology Determination." Oklahoma State University Research Consortium on Well Log Data Enhancement via Signal Processing, Final Report, 1985-1986 Program, pp. 5.1-5.43.

[34] Diplog Analysis and Practical Geology. Dresser Atlas, Dresser Industries, Inc., 1983.

VITA

John W. Cartinhour, Jr.

Candidate for the Degree of

Doctor of Philosophy

Thesis: PATTERN RECOGNITION FOR A CLASS OF WARPED WAVEFORMS WITH APPLICATION TO WELL LOG SIGNATURE RECOGNITION

Major Field: Electrical Engineering

Biographical:

Personal Data: Born in Rockville Centre, New York, January 9, 1950, the son of Juliet S. and John W. Cartinhour. U.S. Army Veteran (1968-1971).

Education: Graduated from Livingston High School, Livingston, N.J., in June 1968; received the Bachelor of Science Degree, Magna Cum Laude, in Engineering Technology from the University of Arkansas at Little Rock in May, 1982; received the Master of Science Degree in Electrical Engineering from Oklahoma State University in May, 1984; Completed requirements for the Doctor of Philosophy Degree in Electrical Engineering at Oklahoma State University in May 1987.

Professional Experience: Research Assistant, School of Engineering Technology, University of Arkansas at Little Rock, 1981-1982; Graduate Research Associate, School of Electrical and Computer Engineering, Oklahoma State University, 1984-1987.