The University of Alberta

Signal Classification in Digital Telephone Networks

by ©

Jeremy Sewall

Submitted to the Faculty of Graduate Studies and Research in partial fulfillment of the requirements for the degree of Master of Science.

Department of Electrical Engineering

Edmonton, Alberta

Spring 1996

ISBN 0-612-10757-4

University of Alberta

Library Release Form

Name of Author: Jeremy Sam Sewall

Title of Thesis: Signal Classification in Digital Telephone Networks

Degree: Master of Science

Year this Degree Granted: 1996

Jeremy Sewall
12958-101 Street
Edmonton, Alberta
T5E 4E7

January 5, 1996

University of Alberta

Faculty of Graduate Studies and Research

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research for acceptance, a thesis entitled Signal Classification in Digital Telephone Networks submitted by Jeremy Sam Sewall in partial fulfilment of the requirements for the degree of Master of Science.

Dr. B.F. Cockburn

Dr. W.D. Grover

Dr. X. Sun

Dr. A. Basu

Date: Jan. 3, 1996 BC

Dedicated to the memory of my dear grandmother,

Lillian Bruun.

Abstract

Algorithms for the classification of voice-band signals in digital telephone networks are studied, developed, tested, and evaluated. Signal types that are resolvable include speech, white noise, voice-band data (VBD) modems, and facsimile (FAX). VBD and FAX signals can be further classified into subgroups based on the modulation method. The algorithms developed are designed to be used in the real-time monitoring of multiple voice-band channels. To achieve this objective, the algorithms must be of low computational complexity, have highly accurate classification rates, and be able to resolve useful signal classification groups.

The developed algorithms are based upon time series discriminant variables computed from normalized autocorrelation lags of the passband signal, and the normalized central second-order moment of the full-wave rectified passband signal. Classification performance is evaluated using provably optimal linear discriminant functions, quadratic discriminant functions, and heuristic methods. Quadratic discriminant functions are found to yield the greatest classification accuracy. Overall misclassification error rates are less than 0.2% percent for most signal types, and on the order of one percent for two particular groups. These classification performances are achievable after monitoring only 256 milliseconds of signal activity. The length of time required for classification depends upon the specific algorithm implementation, and ranges from 32 to 256 milliseconds.

Other classification methods are evaluated including statistical signal characterization, amplitude and phase histogramming, and monitoring for Frequency Shift Keying signaling between facsimile stations.

Acknowledgements

First and foremost, I would like to express a heart-felt thank-you to my supervisor Dr. Bruce Cockburn for his years of guidance, assistance and limitless patience. Dr. Cockburn is an Associate Professor at the University of Alberta, and an Affiliate Professor with TRLabs, Edmonton.

Without the support and help freely given by several other individuals and organizations, this thesis would never have been completed. I would like to thank:

- Mr. Anthony Soong, a Ph.D. Candidate from the U of A, for his hours of time invested in helping me to understand the theory of signal discrimination and communications.
- Dr. Wayne Grover, U of A Professor and TRLabs Networks and Systems Manager, for introducing me to the problem of signal discrimination as well as his help and guidance in this project.
- Dr. Anup Basu, U of A Associate Professor, who first suggested that we explore using autocorrelation sequences for classifying signals, and agreeing to examine this thesis.
- Dr. Xiaoling Sun, U of A Assistant Professor, for reading and examining this thesis.
- Mr. Jim Slevinsky, AGT research engineer seconded to TRLabs, for his helpful consultations and assistance in editing.
- Mr. Allan Pollard, engineer from MTS, for first proposing this research topic and for his helpful remarks and discussions.
- TRLabs and all of the staff and students there for their friendship, and assistance.
- My best friend Nicole, thanks for being there. All of my other friends and family for encouraging me to further my studies, and giving me the opportunity to do so.

Table of Contents

List of Figures

List of Tables

List of Acronyms

ADPCM	Adaptive Differential Pulse Coded Modulation
Bell 103	300 bps full-duplex FSK modem communication standard
Bell 212A	1200 bps full-duplex FSK modem communication standard
B-ISDN	Broadband Integrated Services Digital Network
BPF	Band-Pass Filter
bps	bits per second
CCITT	The International Telegraph and Telephone Consultative Committee (CCITT was a permanent organ of the ITU until February 28, 1993 when it was replaced by the ITU-T)
codec	coder/decoder
DFT	Discrete Fourier Transform
DPSK	Differential Phase Shift Keying
DS0	64 kbps serial digital signal standard for voice-band channels
DS1	1.544 Mbps serial digital signal standard (carries 24 DS0s)
DSI	Digital Signal Interpolation
DSP	Digital Signal Processor (Processing)
FAX	Facsimile transmission and reception device
FFT	Fast Fourier Transform
FIR	Finite Impulse Response
FSK	Frequency Shift Keying
GOS	Grade of Service
HPF	High-Pass Filter
IIR	Infinite Impulse Response
ISI	Inter-Symbol Interference
ITU	International Telecommunications Union

ITU-T	ITU Telecommunication Standardization Sector (formerly the CCITT)
kbps	kilobits per second (thousands of bits per second)
LPF	Low-Pass Filter
MATLAB	Matrix-oriented software package
Mbps	Megabits per second (Millions of bits per second)
ms	millisecond
N-ISDN	Narrowband Integrated Services Digital Network
NRZ	Non-Return to Zero
PC	Personal Computer (based on Intel x86 or compatible processor)
PCM	Pulse Coded Modulation
PLL	Phase Locked Loop
PSD	Power Spectral Density
PSTN	Public Switched Telephone Network
QAM	Quadrature Amplitude Modulation
SNR	Signal-to-Noise Ratio
SPSS	Statistical software Package for Social Scientists
T1	Electrical signal standard for transporting one DS1
TCM	Trellis-Coded Modulation
TC-QAM	Trellis-Coded Quadrature Amplitude Modulation
telco	abbreviation for "telephone company"
V.17	14,400 bps half-duplex QAM facsimile communication standard
V.21	300 bps full-duplex FSK modem communication standard
V.22	1200 bps full-duplex FSK modem communication standard
V.22bis	2400 bps full-duplex QAM modem communication standard
V.27ter	4800 bps half-duplex DPSK facsimile communication standard

V.29	9600 bps half-duplex QAM facsimile communication standard
V.32	9600 bps full-duplex TC-QAM modem communication standard
V.32bis	14.4 kbps full-duplex TC-QAM modem communication standard
V.34	28.8 kbps full-duplex TC-QAM modem communication standard
VBD	Voice Band Data
VCO	Voltage Controlled Oscillator
VLSI	Very Large Scale Integration

Chapter 1

1.0 Introduction

Within telephone networks it is often necessary to determine what type of voiceband (VB) traffic is being carried on network channels. There are several reasons why a solution to the classification problem is desirable: First is security. Often telephone companies (telcos) provide different billing rates for voice, data, and FAX users. Thus, some users attempt to save money by paying for one type of service while using it for another purpose. Second, telcos may want to use the traffic type information to encourage heavy data or FAX users to purchase special lines for their specific needs. A third reason is network planning. It would be useful to any network planner to have knowledge of the characteristics of the different types of traffic which are on the network, such as the specific holding times and the distribution of calls of each traffic type. For example, the peak times for FAX traffic may tend to be from 12:00 midnight to 1:00 am. This is known as the busy hour. Thus, planners may provide for network capacity for these special services and even create new services for non-voice traffic. A final reason that we give is signal compression. If telcos wish to perform compression on signal streams, then they should be aware of the type of traffic being compressed. For example, speech signals can be easily compressed with very little loss in signal quality or comprehensibility. However, if a 28.8 kbps data signal is compressed using ADPCM, the encoded digital data will be destroyed. Clearly only lossless compression methods should be used on data signals. Thus knowledge of the signal type is critical when selecting a compression method.

Several classification methods have been proposed in the literature: some simply address the problem of discriminating voice from non-voice traffic [79, 64, 47]; other methods distinguish between various voice-band data (VBD) signal types [8, 27, 53]. Since both facsimile machines and data modems utilize the same types of modulation standards, we will refer to both signal types as VBD. Fig. 1 shows the context of the problem we are considering in this thesis. At the customer's premises different types of equipment may be connected to the Public Switched Telephone Network (PSTN) analog loop. The figure shows three possible devices including an ordinary telephone, a facsimile machine, and a computer with a VB modem. Within the PSTN, the telephone companies may wish to know what types of traffic are being carried over their digital carrier systems.

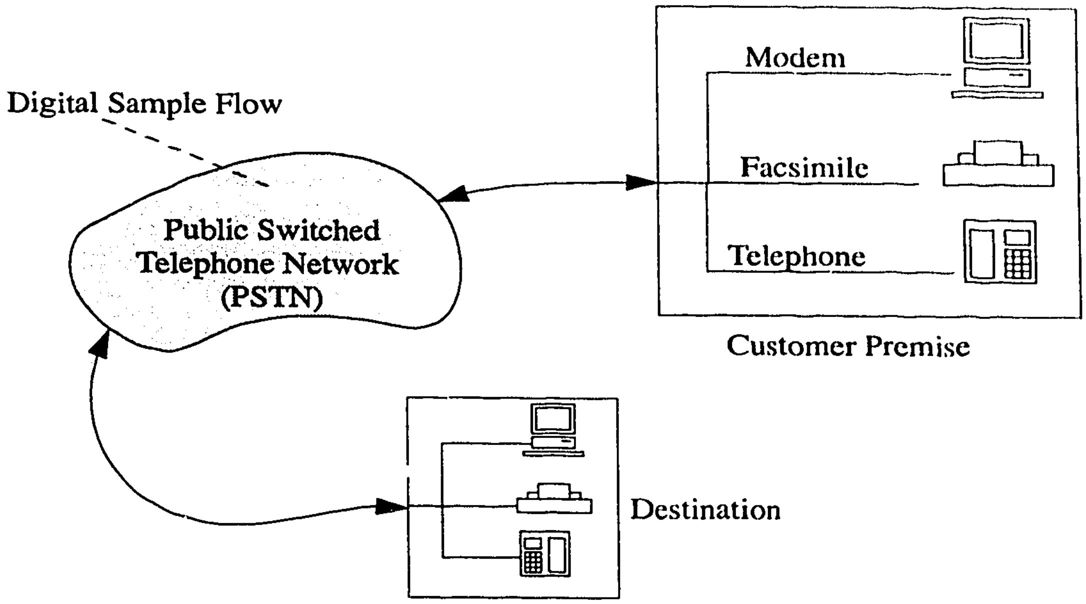


FIGURE 1. Generalized conceptual model of problem statement.

The time series of four different signal types are plotted in Fig. 2. Actual call recordings were used to generate the plots. The signals have been sampled at 8000 samples per second as they would be within an internal digital trunk in the network. The first plot in Fig. 2 is of a portion of a spoken word. Plots two, three, and four are from a recorded 28.8 kbps V.34 modem, a 14.4 kbps V32bis modem, and a 9600 bps V.29 facsimile machine, respectively. These signals represent the dominant types of signals that would be observed in the PSTN today. From the plots it is clear that speech signals are quite different from high-speed modem and facsimile signals. Some portions of unvoiced speech sounds (e.g. "shhhh") are more similar to the non-voice signals shown. In contrast to the obvious differences between speech and non-speech signals, the time series of the non-speech signals appear to be very similar. This observation forces us to search for other representations of the signals that will make their differences more apparent.

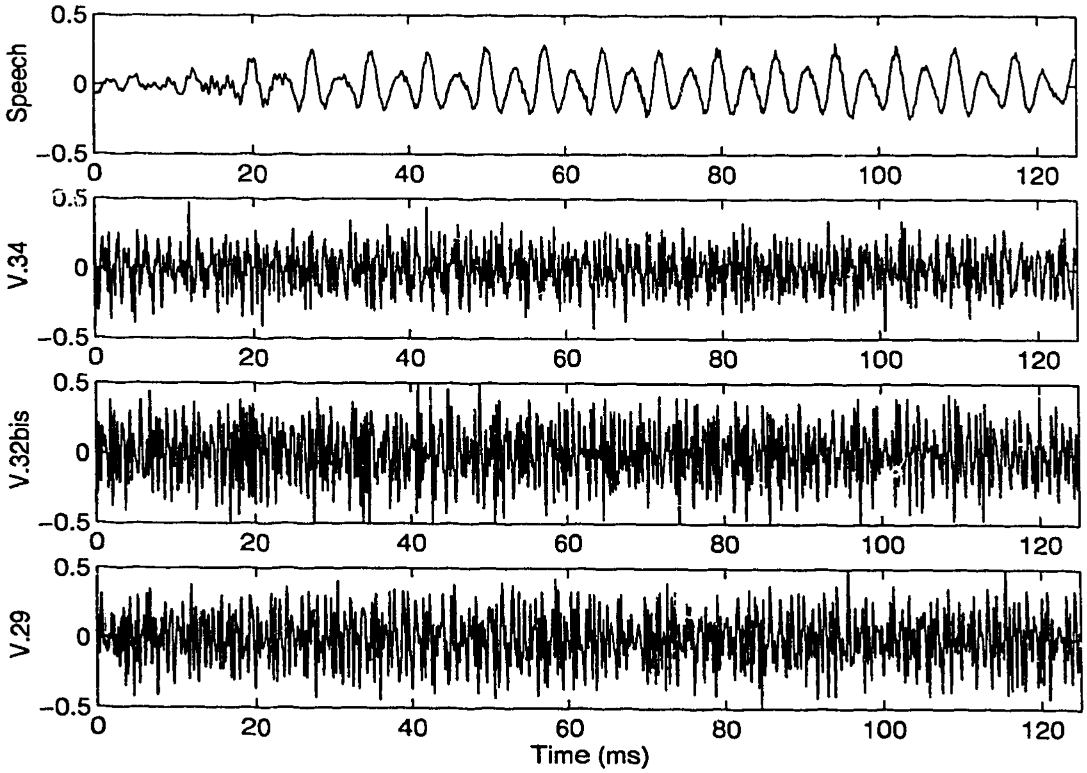


FIGURE 2. Time series of four signal types

The Power Spectral Density (PSD) for the same four signals is shown in Fig. 3. The signal length used to generate the PSD is the same as the signal length used in Fig. 2

(i.e. the signal lengths are 125 ms, or 1000 samples). The plots show a difference in the PSD of each signal type, the greatest difference being between the voice and non-voice PSDs. Even though the PSDs of each of the non-voice signals are quite similar, there are differences in their center frequencies and bandwidths. These differences could be used as the basis for a classification system, however there are drawbacks to this simplistic solution, the worst of which being high computational complexity. Computing the PSD of a signal, or even estimating it, requires a relatively large number of computations. For example, if the PSD was computed from the Discrete time Fourier Transform (DFT) of the signal, the complexity of the PSD algorithm would be of order N^2 where N is the sequence length. Another drawback is the length of the signal sequence required to form an accurate PSD. As the plots in Fig. 3 show, the differences between the spectra of some non-voice signals are very subtle, and thus require an accurate PSD to differentiate them. A final drawback is the difficult task of matching an observed PSD to the expected classes of PSD. This problem can be solved by pattern matching, however the complexity of matching arrays of PSD values is relatively high when compared to other classification methods that are based on single-valued variables.

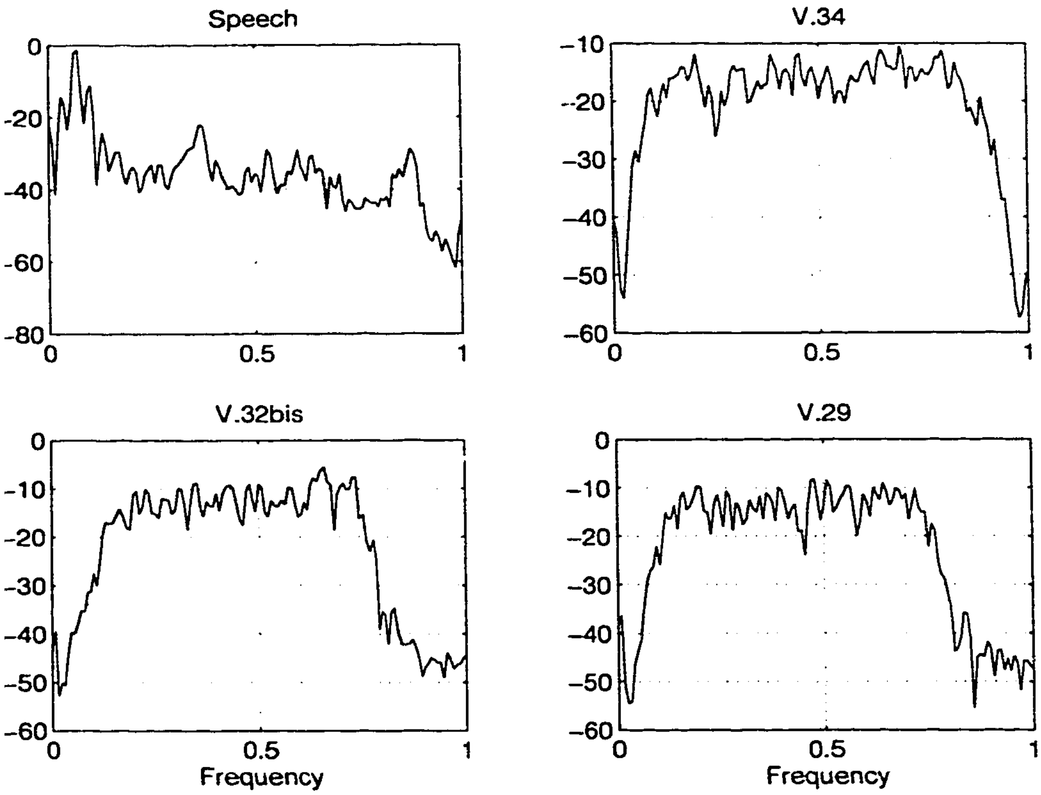


FIGURE 3. Power Spectral Densities of four signal types.

The intent of this thesis was to evaluate algorithms that have been developed for ining what type of traffic is being carried on the digital channels within the PSTN. ition, new, improved algorithms were to be developed and their performance stud- ie traffic was to be sorted into one of three major categories including voice, data X (facsimile). Data and FAX traffic were to be further divided into sub-categories ling to the transmission protocol and the data rate. The monitoring point will gener- a T1 digital telephone signal which carries 24 x 64 kbps time-division-multiplexed els, using 8-bit PCM (Pulse Coded Modulation) and µ-law companding. This is not a restriction to the validity of the classification methods, however, but simply an expected application environment.

In approaching the classification problem we made the following additional assumptions: We assume that knowledge of temporal call boundaries is not available to the classifier, and that only one direction of a bidirectional call is observable. We also assume that full-time monitoring of a channel is not possible or cannot be guaranteed. These assumptions immediately eliminate solutions that rely on observing modem training and handshaking sequences. Our assumptions are convenient and appropriate for several reasons: (1) a traffic monitor may only have access to a single direction of a call, (2) network signalling information may not be easily accessible, and (3) full-time monitoring of every channel may not be feasible.

A few products are available commercially for solving aspects of the problem under consideration, some that meet our assumptions and others that do not. These include the CTel NET-MONITOR System 2432 [16]; the AT&T Voice/Data Call Classifier [2]; the Tellabs Digital Channel Occupancy Analyser [72]; and the MPR Teltech Ltd. Service Discrimination Unit [50]. These systems have the disadvantages of being physically large, relatively expensive, and restricted in their classification resolution. As an example, one of the above machines was tested under working conditions by a TRLabs corporate sponsor, and found to perform with a 38% misclassification rate given typical network traffic. Our goal was to operate with a misclassification rate of less than 1% using compact, inexpensive hardware. The above mentioned products are discussed further in Section 2.2 on page 35.

The organization of this thesis is as follows: Chapter 1 gives an introduction to the classification problem. Chapter 2 provides useful background on non-voice signals and discriminant analysis. Chapter 3 reviews published voice-band signal classification algorithms and describes commercially available products. Chapter 4 discusses the research equipment used in this study. Chapter 5 presents the proposed classification algorithms and their resulting performance. Chapter 6 discusses other classification algorithms that were explored, but not included in the final proposed classifier design. Chapter 7 contains a discussion of classification system design considerations and recommendations. Finally, Chapter 8 concludes the thesis and gives directions for future follow-up work in the area of voice-band signal classification.

ɹiterature Review

In this chapter we present a review of the published literature and of commercial ts that are relevant to this thesis work. Several other papers have been written on ic of speech vs. non-speech discrimination as well as VBD signal discrimination. ıave also been other machines developed to solve problems similar to the one pro- here. We review the papers and devices, and justify the need for this further :h.

ıignal Discrimination Theory

A common thread between much of the work done on voice-band signal discrimi- is that there are very few papers that build upon previous research. Most of the ; reviewed here stand on their own, and present methods that are not proven optimal, n provably superior to any other. Many of the most popular discriminant variables ed by several researchers, such as zero-crossing detectors and second-order statisti- riables. The major differences between the papers are often only slight variations of he discriminant variables are applied and how the classification is performed. Little l is apparently given to optimal discrimination methods, even though discriminant sis has been a well-established research field in mathematics for many years.

Speech Versus Non-Speech Discrimination

First we address the problem of speech vs. non-speech discrimination. A major :ation of systems with this capability is the real-time statistical multiplexing of speech and data signals into a single channel. On average, speech activity occurs only about 40% of the time [79]. Thus it is desirable to be able to detect when speech activity occurs, and then multiplex a data signal into the frequent intervals with no speech activity. This requires only a speech detector. Speech vs. non-speech discrimination is required when receiving a signal that has been multiplexed with both speech and data signals. The discriminator must separate the interwoven streams of data and speech signals.

Since the average talk spurt activity lasts only about one second [79], the detector and discriminator must respond very quickly. The length of time required to detect speech is often called the "*hangover time*", while we will call the time required for discrimination between VBD and speech the "*response time*". For large scale applications it may also be necessary to monitor several channels simultaneously in real time. This requires low complexity algorithms. Finally, the accuracy of the discriminator must be very high for the multiplexor to function properly. Thus we have three criteria for evaluation of discrimination methods: response time, algorithm complexity, and classifier accuracy.

We note that many of the papers dealing with speech vs. non-speech discrimination also address the topic of VBD sub-class discrimination.

2.1.1.1 Y. Yatsuzuka

Y. Yatsuzuka [79] wrote a key paper on the problems of speech detection and VBD discrimination as they applied to DSI-ADPCM systems. DSI (Digital Speech Interpolation) channels can utilize pauses in speech to multiplex in other signals, which usually carry data. ADPCM refers to a method of performing adaptive differential pulse code modulation, which effectively compresses a 64 kbps digital voice-band signal into a lower bit-rate digital signal (32 kbps) by utilizing the fact that speech signals do not vary as rapidly from sample to sample as VBD.

Yatsuzuka based the speech detector discriminants on short-time energies, zero-crossing rates, and sign bit sequences (which effectively indicate zero crossings). The speech detector also has classification capabilities, and can group speech signals into narrow-band, wide-band, or low-power sound subclasses. Figures 4 and 5 give block diagrams for Yatsuzuka's speech detector and voice-band data discriminator, respectively.

Discrimination between VBD and speech is performed using discriminant variables based on short-time energy, zero-crossing rate, and the coefficients of an adaptive predictor. VBD subclasses include 4800 bps 8-phase PSK or 8-QAM, or 9600 bps 16-QAM. (Since the paper was published in 1982, the VBD signals that were considered in [79] no longer correspond to those in use in modem communications systems.) These subclasses are discriminated using a coefficient of variation of the short-term amplitude distribution of the input signal and a prediction gain. We are most interested in the discrimination abilities of Yatsuzuka's algorithm, not the speech detector.

To discriminate between VBD and speech, a complex set of heuristics is followed. Short-time energy, zero-crossing rate, and predictor coefficients are all considered. To compute the short-time energy, a signal segment of 32 ms (256 samples) is divided into r blocks. Each block is 2 ms long, therefore the number of blocks is 16 and the length of each block m is 16. The short time energy of the r^{th} block containing m samples is computed as shown in equation (1), where X is the received signal sample stream and $X[t]$ is a particular sample at time t.

$$Q_r = \frac{1}{m}\sum_{j=1}^{m} X[m(r-1)+j]^2 \tag{EQ 1}$$

The zero-crossing rate is also computed for every block, and then the average of the crossing rates for all blocks. The adaptive predictor coefficients are also used as shown in Fig. 5. The adaptive predictor has 4 taps, each of which is used as a discriminant variable.

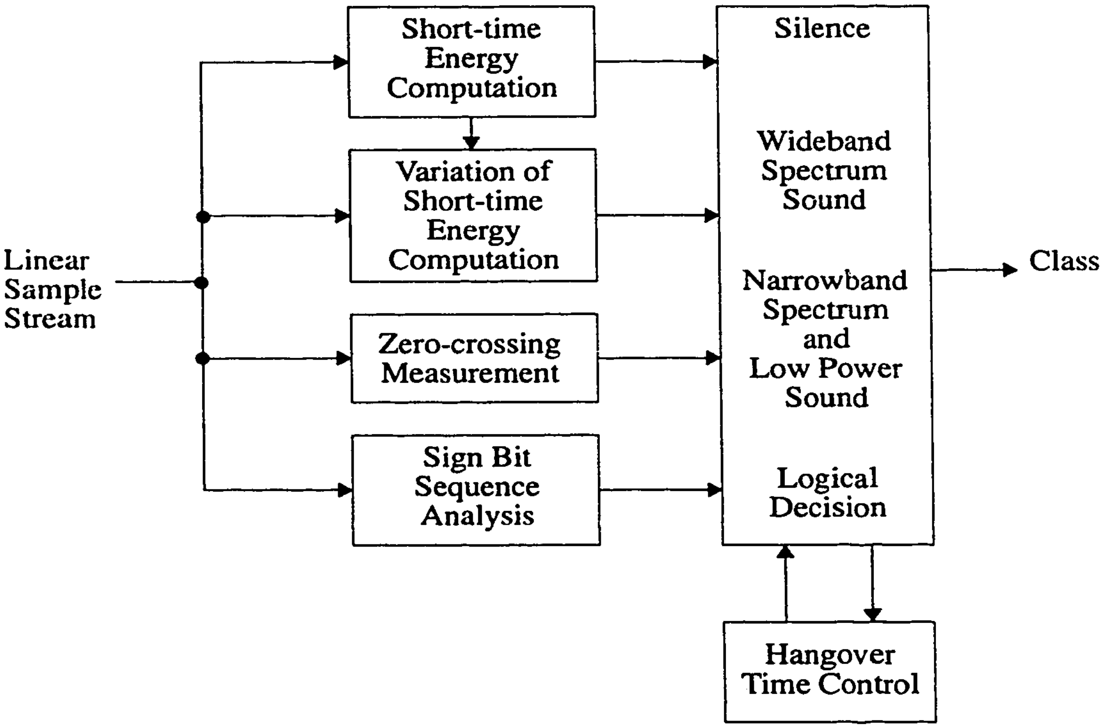

Linear Sample Stream
Short-time Energy Computation
Variation of Short-time Energy Computation
Zero-crossing Measurement
Sign Bit Sequence Analysis
Silence
Wideband Spectrum Sound
Narrowband Spectrum and Low Power Sound
Logical Decision
Class
Hangover Time Control

FIGURE 4. Configuration of Yatsuzuka's speech detector.

These variables, combined with a complex set of heuristics, form the speech vs. VBD discriminator. To subclassify VBD signals, the average of the adaptive predictor gains and the coefficient of variation of the signal are both used as discriminant variables. The two subclasses are separated by another set of heuristic rules. Yatsuzuka mentions that the performance of the VBD subclassifier was only tested on simulated signals with no simulated line impairments. Performance was reported to be good, but it could be improved further by allowing the classifier to have more time to analyze the signals (i.e. by using longer signal segments and a longer hangover time).

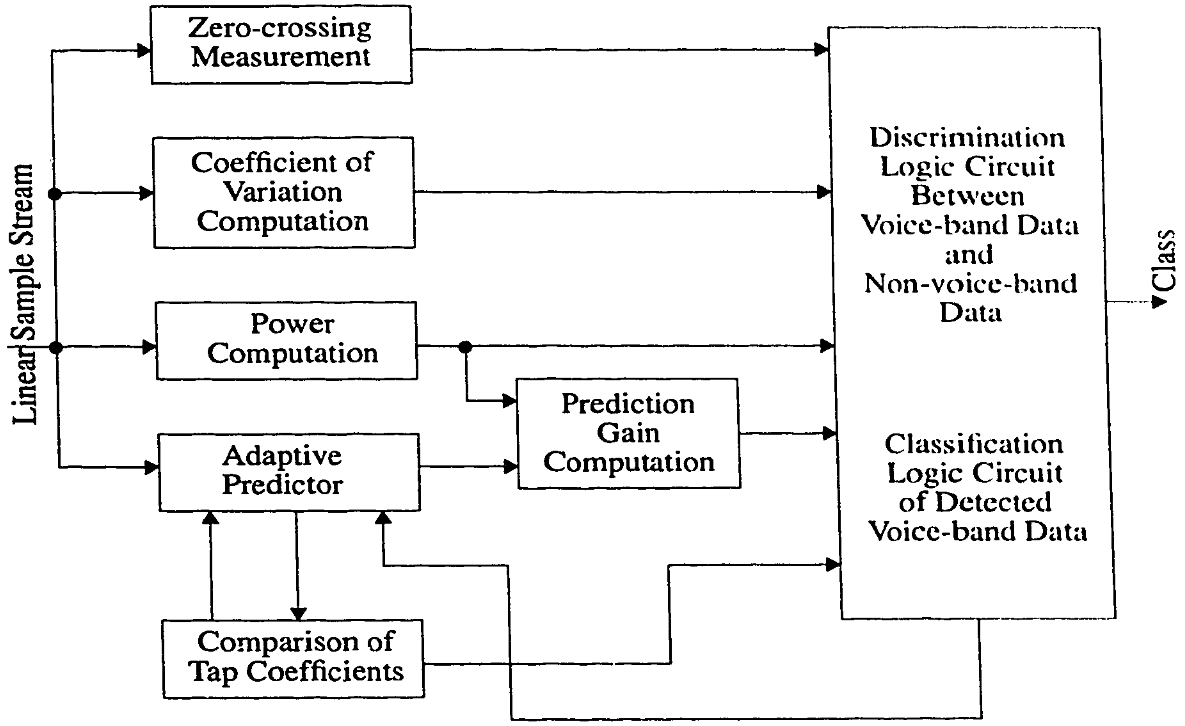

Linear Sample Stream
Zero-crossing Measurement
Coefficient of Variation Computation
Power Computation
Adaptive Predictor
Prediction Gain Computation
Comparison of Tap Coefficients
Discrimination Logic Circuit Between Voice-band Data and Non-voice-band Data
Classification Logic Circuit of Detected Voice-band Data
Class

FIGURE 5. Configuration of Yatsuzuka's voice-band data discriminator.

The discriminant variables presented in [79] are relatively complex. Several functions must be computed, and an adaptive predictor must be implemented. The discriminator utilizes 16 energy computations (one for each block), 4 adaptive predictor taps, the average of the adaptive predictor taps, 16 coefficient of variation computations (one for each block), and a zero-crossing average over all blocks. The classification strategy is based upon a large set of heuristic rules. For example, "For voiceband data, $Av(Z_r)$ appears always to have a value in a range between 3 and 11 because of the energy distribution around 2 kHz." Another drawback is that the system was only tested with ideal simulated VBD signals, although line impairments are known to cause difficulties for classifiers.

Finally, only 2 types of VBD signals were considered, which were considered fast at the time of Yatsuzuka's writing; however, they no longer represent mainstream VBD communications signals. On the plus side, Yatsuzuka's algorithms perform very well for detecting speech and classifying speech into wideband, narrowband and low power subclasses.

2.1.1.2 C. Roberge and J.P. Adoul

Roberge and Adoul [64] published a paper in 1986 on the topic of speech vs. VBD discrimination, again intending the algorithms to be used for the statistical multiplexing of speech with VBD. For this technique to work, the transmitter must be capable of detecting speech and silence intervals, when VBD signals can be inserted. At the receiver end, unless special markers are inserted in the signal, speech and VBD signals must be discriminated. Roberge and Adoul present a method for discriminating between these two types of signals. The only VBD signal considered was 9600 bps QAM. The authors claim that this type of signal is more difficult to identify since FSK and PSK signals have constant envelopes, while QAM signals do not. FSK and PSK signals can be easily identified by a constant envelope detector.

Roberge and Adoul define a method for computing discriminants that is based on a fixed window length N. The first discriminant, Z_0, is the zero-crossing count in the window. This discriminant is said to roughly reflect the dominant frequency of the signal. The second discriminant, Z_1, is the zero-crossing count of the difference signal. The difference signal is the received signal subtracted by a time delayed version of itself, where the time delay is one sample period. This is effectively the discrete first derivative of the received signal. This discriminant is said to represent the second resonant frequency of the vocal tract, or the second formant.

The authors note that the longer the observation windows are made, the better the classifier performs. However, the misclassification rate will not reach zero. (Misclassification rates do not necessarily reach zero since the probability density functions of the discriminant variable(s) for each group may overlap somewhat. In the case of quadratic discrimination, the variables may not be sufficiently correlated to perform accurate classification. Perfect separation is required for 0% misclassification of all cases, which is not generally achievable for many of the signal generation processes that exist in nature. Speech is one such signal type.) Also, in some cases the response time of the classifier is a factor, and thus observation window lengths must be kept to a minimum.

Three classification methods are presented, all based on the use of simple decision boundaries. Fig. 6 shows the concept of each classification method. The diagrams show shaded areas which represent the region in the two-dimensional discriminant space that belongs to VBD signals.

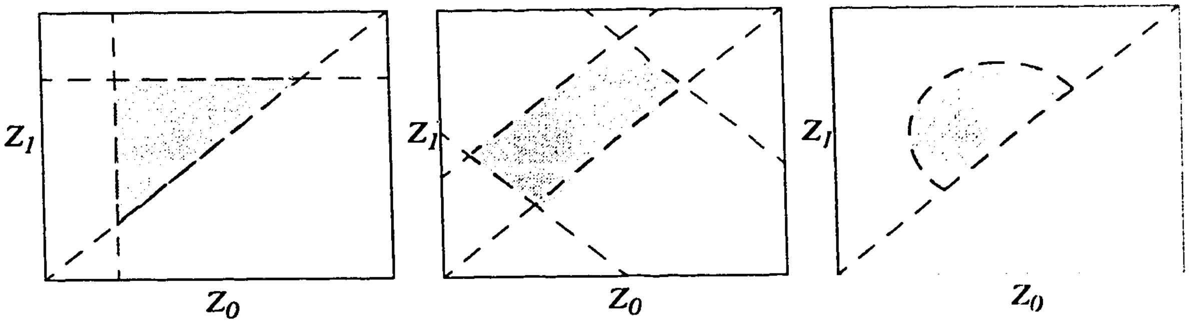


FIGURE 6. Three classification methods from Roberge and Adoul.

The authors claim that their algorithm performed with approximately 0.03% misclassification of speech signals as VBD in normal conditions, as compared to a reported misclassification rate of 0.09% reported by Yatsuzuka [79]. The hangover and response times are not given however. In addition, there was only one type of modem signal, 9600 bps QAM, was considered. Details of the modem structure were not given. Basing the tests on only one type of modem is a major weakness; however, it was a realistic assumption given the problem being considered. The discriminator was designed to operate on a channel in which it was known a priori that speech would be multiplexed with a specific implementation of 9600 bps VBD. This assumption is not valid for the more general problem that we are considering in this thesis.

The authors have shown that separation of speech from VBD should be an easy task, and can be performed well with two simple zero-crossing discriminant variables. Although the discriminant variables are simple, the classification strategy again is not. The classifier uses a set of heuristic rules which, while not as complex as Yatsuzuka's, still cannot be shown to be optimal. Decision boundaries are drawn from observations of actual tests. No optimal or automated method for choosing decision boundaries is given.

2.1.1.3 S. Casale, C. Giarrizzo, and A. La Corte

Casale et al. [13] published a paper in 1988 reporting on a speech vs. VBD discriminator implemented on a DSP. The discriminator is intended to be used as a filtering stage prior to the compression of voice-band signals. The job of the discriminator is to prevent VBD signals from being compressed since the intended compression scheme would destroy the VBD signal.

Discriminant variables presented in [13] include the short-time energy of an N sample window, and the zero-crossing rate of the received signal and its first derivative. The short-time energy is said to be lower in general for speech signals, and it varies from window to window considerably more for speech than for VBD. We note that the discriminants utilized here appear to be a combination of those presented by Yatsuzuka [79] and Roberge et al. [64].

The above discriminants present nothing new; however, they were implemented on a real-time system based on a Texas Instruments TMS320C20 DSP. An interesting feature of the system is that the received signal was separated into three streams. One stream is filtered by a high-pass filter, a second stream is filtered by a low-pass filter, and the third stream is not filtered at all. The discriminant variables are computed for all streams. A value of $N = 256$ samples was chosen as a compromise between misclassification rate and response time.

The types of VBD signals tested include V.21 channel 1, V.21 channel 2, V.23, and V.27. Misclassification rates are reported to be below 0.2% for all classes. An interesting table is included in the paper that reports the number of instructions as well as memory requirements for each stage of the classifier. We include that table here. Note that the instruction cycle time for the TMS32020 fixed-point processor is 200 ns.

TABLE 1. Processor cycles and memory requirements, per sample.

Routines	Instruction Cycles	ROM (bytes)	RAM (bytes)
Companded PCM to linear PCM conversion	9	532	2
Synchronization	5	8	2
Low-pass and high-pass filters	132	396	28
Revision buffer memory pointers	12	28	8
Differenced signals and zero-crossings	62	124	12

TABLE 1. Processor cycles and memory requirements, per sample.

Routines	Instruction Cycles	ROM (bytes)	RAM (bytes)
Short-time energy computation and test	26	64	14
Tests of zero-crossing results	54	144	30
Revising output condition	6	14	-
Start of the processor	83	196	32
TOTAL	389 (77.8 μs)	1506	128

The totals in the table indicate that 389 processing cycles are required per sample. Each 256 sample window sees 160,000 processing cycles, and the discriminator requires 99,584 of those cycles. Therefore we can conclude that the processor load due to the classifier is about 62%. The authors claim that two channels can be classified using a single processor of this type, however their own test data appears to refute this claim unless certain functions can overlap when monitoring two channels. If we consider instead a modern (but still inexpensive) fixed-point digital signal processor with a cycle time of 16.7 ns, then it would appear that this classifier could perform full-time monitoring of 19 channels.

These results indicate that roughly 34% of processing cycles are spent filtering the incoming data. Also, since the data is split into 3 streams, all other functions must be computed in triplicate. Obviously if the filtering stages could be removed, and only one data stream was required, the number of processing cycles could be substantially reduced. We will argue later on that in this way full-time monitoring of at least 24 channels is feasible using an inexpensive DSP.

2.1.1.4 K. Shimokoshi and Y. Hashitsume

In 1989 K. Shimokoshi and Y. Hashitsume [68] described a neural network implementation of a speech vs. non-speech discriminator. Their system is intended to be used in a packet-switched network for identifying the type of traffic being transmitted. Certain types of traffic, such as VBD signals, are susceptible to signal degradation caused by bandwidth compression, silence compression, and echo cancelling. For these reasons, the authors proposed a method for discriminating speech from non-speech signals. One type of packet network alluded to by the authors is ATM (Asynchronous Transfer Mode). The authors predict that speech signals will still make up the bulk of traffic carried via ATM, so various lossy compression methods such as DSI will continue to be used extensively. The authors also assume that VBD signals will continue to be used extensively. (This may be a false assumption if technologies such as N-ISDN achieve wide acceptance.)

The authors propose a method for discriminating speech vs. non-speech signals by combining a back propagation neural network with digital signal processing. Test results are presented for an implementation on the general purpose TMS320C25 DSP. A block diagram of the resulting classifier is given in Fig. 7.

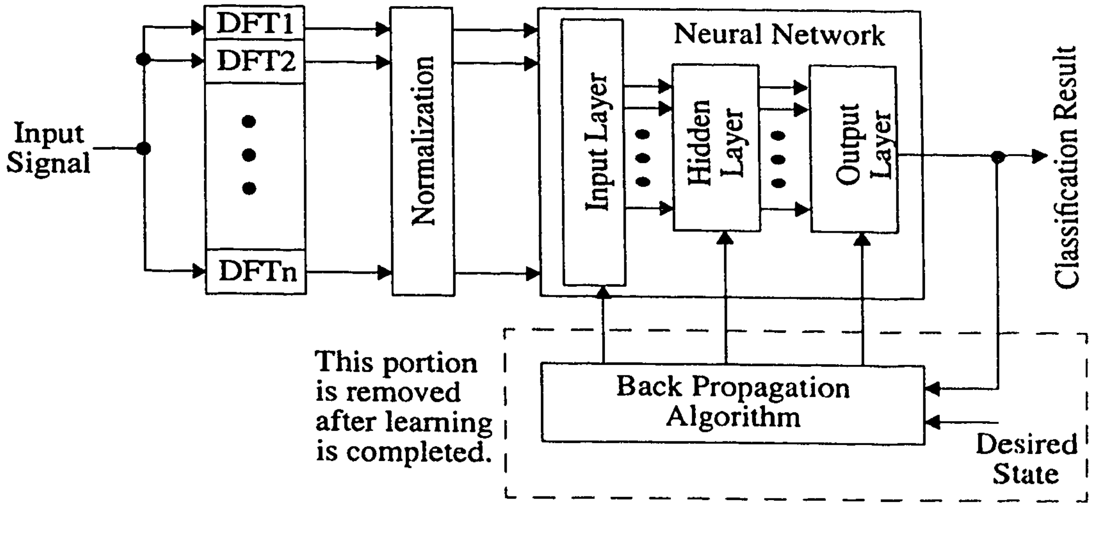


FIGURE 7. Shimokoshi and Hashitsume's voice/non-voice discrimination model.

To perform discrimination, an 8-channel octave filter bank converts the incoming signal to its respective spectra. Each node in the filter bank consists of a DFT implementing Goertzel's algorithm. The DFT frame length is equal to the packet length of four ms (32 samples) in the test system. The resulting discrete spectra are input into a neural network which performs learning via a back propagation algorithm. The neural net was trained using Japanese speech from an adult male, five different DTMF (Dual Tone Multi-Frequency) tones, and a V.23 modem. (Note that a V.23 modem is a 1200 bps FSK modem, similar to V.22.)

The system was also tested using signals from an adult male speaking English as well as Japanese, a 200 bps V.21 modem, 300 bps V.21, 1200 bps V.23, and DTMF signals. Even though these were signals that were tested but not learned explicitly, the classifier still handled them reasonably well. The misclassification rate for speech as non-speech was 13.8% for Japanese and 10.5% for English. Non-speech data signals were never misclassified; however, DTMF tones were misclassified as VBD 0.7% of the time. The authors conclude that the high misclassification rate for speech signals is not important since this would simply result in fewer signals being compressed in the network. That is, the cost of misclassifying VBD as speech is much higher than vice versa.

We note that the modems tested in [68] are not representative of the most common modems used today. Also, the amount of computation required for performing the 8-channel DFT bank is very great. (This is the portion of the system implemented by a DSP. The neural network was also implemented using the DSP chip. Future work was to include mapping the neural network to specialized hardware.) It is reported that the overall classification system requires 5 MIPS (Million Instructions Per Second). This represents a heavy load on the DSP that was used. (A standard 40 MHz TMS320C25 DSP can ideally perform 10 MIPS if every instruction can be performed in one cycle. This is rarely possible.) The authors do not report the response times of the system.

2.1.1.5 Summary

The papers summarized above generally yielded acceptable classifier accuracies, depending upon the intended application. Shimokoshi and Hashitsume [68] presented a neural network system that yielded a very high rate of misclassification of speech signals and was also relatively computationally intensive. The other methods proposed yielded acceptably low misclassification rates for our application. However, all of these methods utilize discriminant variables that are computationally intensive and classification strategies that are heuristically based. Thus it is still desirable to develop more efficient discriminant variables that still yield low misclassification rates. It is also desirable to study these discriminant variables in a systematic way to evaluate their effectiveness. The classification strategy should be based upon provably optimal methods where available. Heuristic rules are difficult to form, and they cannot be proven optimal.

2.1.2 VBD Signal Discrimination

Next we consider the separate problem of classifying VBD signals into subclasses. For example, we may wish to identify the modulation method in use, the carrier frequency, baud rate, or bit rate. We define *classifier resolution* to be the number of subclasses that a classification algorithm can resolve from the original group. We also retain the three criteria for algorithm evaluation given in the previous section. Thus we now have four criteria for evaluating discrimination methods: response time, algorithm complexity, classifier resolution, and classifier accuracy.

2.1.2.1 Radio Signal Monitoring

The problem of monitoring communications with the intent of identifying the type of signal being transmitted is not limited to voice-band applications. It is also a major concern of law enforcement agencies or military groups who are interested in monitoring radio signals. The purpose may be to be able to intercept communications, to ensure that operators are obeying the limits of operation parameters, or to identify unauthorized use of a particular channel. Four papers were selected for review on this topic in order to determine the amount of cross-over between classification of radio communications signals and voice-band signals. The four papers were as follows: F. Jondral [46], J. Aisbett [1], Y.T. Chan and L.G. Gadbois [14], L.V. Dominguez, J.M.P. Borrallo, J.P. Garcia, and B.R. Mezcua [18].

In general the goal of these papers is developing methods for identifying the modulation method being used for communications in the short wave band (3 MHz to 30 MHz). This problem is of interest because there is no fixed channel allocation pattern in this band. When a fixed channel pattern exists, the problem of signal identification is significantly reduced.

We do not consider the different methods of constructing receivers here, although this is a major concern in the four papers mentioned. What we are interested in are the discriminant variables used, and the classification methods for obtaining conclusions.

Jondral [46] specified that histograms of the amplitude, instantaneous frequency, and relative zero phase of the received signal at time instants t_m, are stored for signal segments $M = 4096$ samples long. The sampling rate in this case is one of {2.2, 4.4, 5.5, 11.1, 22.2} kHz, depending upon the bandwidth of the signal being tested. The amplitude, instantaneous frequency, and relative zero phase histograms have 32, 128, and 32 bins, respectively.

Classification is performed using the histograms of these three variables and what are called polynomial classifiers. Unfortunately, the reference for polynomial classification given by Jondral [46] is a paper written in German. We were unable to locate a comparable English language paper describing this classification method; however, it appears to simply be a variation of the optimal linear and quadratic discriminant functions described by Shumway [69]. The method is based on computing a discriminant function that minimizes the mean square error in classification. The resulting discriminant functions are a system of linear or quadratic equations. Thus the discriminant polynomial may take on the form of a linear or quadratic equation.

The classes that are discriminated by Jondral's method are two-level ASK, two-state FSK, four-state FSK, two-state PSK, amplitude modulation, single-sideband modulation with suppressed carrier, and noise. QAM signals were not included. Results presented by Jondral indicate that the total misclassification rate of the test set was 2.3% using a quadratic discriminant polynomial with 496 terms. The linear discriminant functions lead to poorer results, with a total misclassification rate of 7.8%. This indicates that a slight improvement is achieved by using a quadratic discriminant function since the covariance matrices of the discriminant variables are different. We can conclude that the method proposed yields reasonably good misclassification rates, has a reasonably small and simple discriminant variable set, and resolves the considered signal types into useful subclasses for the intended application. However, the response time of the system is quite long (from 0.18 to 1.86 seconds), and the subclasses identifiable by the given discriminant variables are not applicable to the VBD signals we are considering. Also, the discriminant variables used are not particularly sensitive to discriminate signals that use identical modulation methods, but differ in their power spectra in some small way.

Aisbett [1] extended Jondral's work to allow the classifier to identify further subclasses of signals based upon the estimated signal-to-noise ratio (SNR). The estimators are based on the way in which Gaussian noise affects the variance of the existing discriminant variables. In total, six variables were considered. The classification method used by Aisbett was strictly linear. Only mean differences in discriminant variables were considered. The signal types considered were slightly different from Jondral's, and include AM voice, AM binary, DSB voice, DSB binary, FM voice, FM binary, CW, and noise. No experimental results were reported.

The important contribution of Aisbett's [1] work is that it was shown theoretically that standard time-domain parameters of signal envelope and instantaneous frequency are strongly influenced by both the modulation type and the SNR. The author notes that further work needs to be done to understand the effects of non-Gaussian noise.

Cha[illegible] Gadbois [14] presented a novel technique for classifying signals based upon their modulation method. The discriminant variable R is the ratio of the variance of the analytic envelope to the square of the mean. Since FM has a constant envelope, and AM does not, this variable is effective for discriminating these two classes. However, the authors conclude that more work needs to be done to discriminate between signals such as

FM and PM since they will have the same R values. Due to the limited number of discriminable subclasses, we will not consider this algorithm further.

Finally, we consider the work of Dominguez, et al. [18]. The authors present a unified system for the general automatic classification of radio communication signals. Three parts of the system are identified including pre-analysis, discriminant extraction, and classification. Again, we are only concerned with discriminant extraction and classification since the pre-analysis section is dependant upon the transmission medium of the signals under test.

Discriminant variable vectors are formed from the complex envelope, phase, and instantaneous frequency histograms. The entire histogram contents were used as discriminant variables, thus the number of variables was quite high. Note that the computation of the complex envelope requires more calculations than all of the other discriminant variables combined.

The classifier took into account the correlation between discriminant variables. Thus we can say that the classification method is similar to a quadratic discriminant function, although it is not identified as such in [18]. The classes that the algorithm attempted to discriminate are based on AM, FM, and PSK modulation methods. Again, these classes are not directly related to the problem of VBD signal discrimination since the dominant modulation methods used are QAM and combined ASK-PSK. Later we will see that in fact the variance of the complex envelope of a VBD signal is a poor discriminant variable for most signals used in modern VBD modems. It is an excellent discriminant, however, for FM and AM signal discrimination.

Although the problem of classifying radio communication signals is similar to that of classifying VBD signals, there are fundamental differences that generally render the algorithms incompatible with one another. The main difference lies in the vastly different types of modulation methods used in each situation. For effective subclassification of VBD signals we need to discriminate between FSK, PSK, combined PSK-ASK, and QAM signals. Also, various types of QAM may be used in VBD connections and these sub-types must also be identified. However, it is instructive to examine the methods used for radio communication signal classification since valuable insights can be gained. First of all, it was shown that the amount of noise present in a signal can be an important factor to consider. Second, it is difficult to separate constant envelope signals using the given discriminant variables used so far. Finally, optimal methods for performing classification, such as linear and quadratic discriminant functions, can be used with good success even if the conditions for optimality are not strictly met.

2.1.2.2 J.E. Hipp

J.E. Hipp presented a paper in which signals using different modulation methods are classified based on statistical moments of the demodulated signal and the signal spectrum [27]. The method could be applied to any general system in which a received signal is band-limited and the center frequency is within 10% of the center of the band. Six discriminant variables were used including the: (1) standard deviation of the demodulated amplitude distribution, (2) asymmetry of amplitude distribution, (3) spread in signal phase relative to a fixed estimated carrier, (4) standard deviation with a threshold of 3 dB above the noise floor of the spectrum, (5) standard deviation with a threshold at 3 dB below the peak of the spectrum, and (6) standard deviation with a threshold at 3 dB below the peak of the squared signal spectrum. Obviously, the computational power required to obtain these discriminant variables is considerable. Indeed this is the major drawback ot the proposed system.

Hipp indicated that many more discriminant variables were considered; only the most useful discriminants were retained in the final system. SPSS was identified as one useful tool for ranking the discriminant variables.

Classification was also performed by SPSS. Consequently we can say that the quadratic method was not strictly used; however, the results can be considered to be very close to what would have been obtained an optimal quadratic discriminant function. The classification classes include noise, single sinusoidal tone, AM, DSB, ICW (Intermittent Continuous Wave), FM, BPSK, FSK, and SSB. Again we note that the most common VBD signals were not included in the study. However, the classification method presented should work well for any signals that vary in their power spectra.

Results were obtained by performing cross-validation in all cases. (*Cross-validation* is a method of predicting future classifier performance based on a limited set of training cases. The training cases are divided into a learning group and a test group. Discriminant functions are formed using the learning group and then tested on both classes independently.) Only simulated signals were considered, with some noise parameters added. Standard channel distortions, with the exception of additive Gaussian noise and frequency offsets, were not considered. The results will not be included here; however, it is of interest to note that the overall correct classification rate was 95.347%.

We can conclude that while Hipp's algorithm is very powerful and could likely be applied to the signals we are considering, the computational complexity of the discriminant variables is too high. In order to compute the variables the signal must first be demodulated with an estimated carrier and the power spectra must be either computed or estimated. Then the statistics for these two signal representations must be computed.

Hipp's paper does outline an excellent analysis structure. The system is separated into preprocessing, discriminant extraction, and classification stages. The discriminant variables that are considered are analyzed for their statistical significance, and a "best" subset of variables is extracted. The classification is performed using cross-validation to ensure accurate results and some channel distortions are included in the simulations. This type of systematic analysis of the problem was also adopted in this thesis.

2.1.2.3 R.J. Mammone et al.

Mammone et al. [53] wrote a paper on the estimation of carrier frequency, modulation type, and bit rate of an unknown modulated signal. The signal types of interest to the authors are CW, BPSK, and QPSK. The discriminants are extracted from the phase derivative of the analytic signal. The analytic signal is a complex valued representation of the real valued received signal, with a single sided Power Spectral Density (PSD). The equation for the analytic signal is given by equation (2) where *z(k)* is the analytic signal, $x(k)$ is the real valued band-pass received signal, and $\hat{x}(k)$ is the Hilbert transform of $x(k)$. This signal representation may be computed by first computing the FFT of the received signal, operating on the frequency domain signal, and then computing the inverse FFT. The resulting analytic signal can be used to indicate the envelope of the received signal without regard to the carrier frequency. Discriminant variables are chosen from the analytic signal to indicate the carrier frequency, modulation type, and bit rate of the signal. The envelope of the analytic signal is also computed to detect amplitude modulation and power fluctuations.

$$z(k) = x(k) + j\hat{x}(k) \qquad \text{(EQ 2)}$$

Two problems with applying this method to VBD signals appear immediately. First, the signal types do not represent typical VBD signals well. Thus the method used for estimating the bit rate will be of no use. Second, the computational complexity of the algorithm is very high. This problem is common to most published work on signal classification. Computing the power spectrum is often seen as the best way to classify signals. While this may be an effective approach for some classification problems, it is also requires a large amount of computing power. This limits the usefulness of these types of methods for real-time, multi-channel applications. Also, it has not been shown that the power spectrum must be directly computed from the time series signal. As we will see later, it is possible to determine much of the relevant power spectrum information by using simple measures of the time series.

2.1.2.4 S.S. Soliman and Shue-Zen Hsue

Soliman and Hsue [70] presented a paper in which they restrict their attention to *M*-ary PSK signals. The intent was to be able to use the *n*th moment of the phase of the signal, where *n* is even, to identify the number of points in the PSK constellation. In prior work, Soliman and Hsue presented methods based on zero-crossing observations for discriminating between PSK and FSK signals. Interestingly the authors note that very little work has been done on the problem of modulation type classification, especially work that is not ad hoc in nature.

The authors [70] show a figure comparing the mean values for the higher order moments of CW, BPSK, QPSK, 8PSK, 16PSK, and 32PSK signals. The values of n considered are {2, 4, 6, 8, 10}. The authors clearly show that by using these moments as discriminant variables, only CW, BPSK, and QPSK are effectively discriminated since the mean values of their moments are unique. All higher M-ary PSK signals fall into the same category as QPSK. That is, they have the same moment mean value for all classes. Thus we can quickly see that this method has limited usefulness, especially under realistic noise conditions which cause the variance of the moments to increase beyond their mean differences.

2.1.2.5 Summary

We conclude this section by observing that, prior to this thesis, there was a need for further work to be done on systematically determining a low-complexity, accurate, and high-resolution algorithm for classifying VBD signals. Surprisingly little work has been done (and published in the open literature) on the specific signals that are seen in the PSTN. In the next section we present the work of researchers who have considered this very problem and shown promising results.

2.1.3 Nevio Benvenuto's Work

Nevio Benvenuto et al. has published papers on the topics of speech vs. non-speech discrimination [6] as well as VBD signal discrimination [8, 6, 7]. We consider his work to be important enough to deserve its own section in this review. Benvenuto's work served as the basis for the most powerful and elegant discriminant variables presented in this thesis.

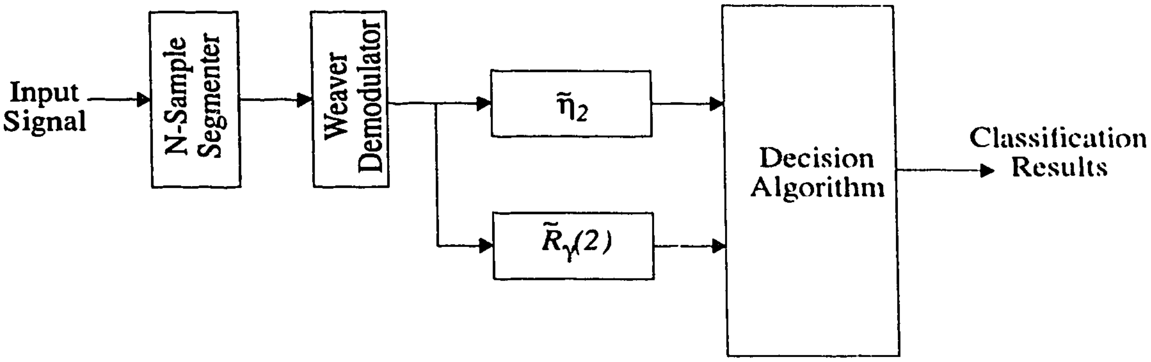


FIGURE 8. Generalized block diagram of a Benvenuto classifier.

Using two discriminants, Benvenuto reports that speech and VBD signals can be distinguished from one another in as little as 32 ms [6]. The normalized second lag of the autocorrelation sequence (ACS) $\tilde{R}_\gamma(2)$, and the normalized central second-order moment of the amplitude components $\tilde{\eta}_2$ of the complex baseband signal are computed as the discriminants. Benvenuto notes that the second lag of the ACS is positive for voice and negative for non-voice signals. Benvenuto goes on to explain that the central second-order moment is an approximate indicator of the non-voice signal complexity and is thus also useful in a voice vs. non-voice discrimination algorithm. Note that Benvenuto always assumes that 8000 sample per second discrete time signals are being classified.

Prior to the computation of the discriminant values, the incoming signal under test is sampled and divided into N sample segments. Each segment must be checked to ensure that there is sufficient signal activity throughout its duration. Benvenuto describes his two discriminants in [6] and [7]. The discrete time baseband signal is denoted by $\gamma(n)$, where n

is the discrete time index. It is obtained by complex demodulation of the passband signal segment followed by low-pass filtering. A carrier of 2000 Hz is chosen, which is approximately the channel centre. The ACS at lag k is given by equation equation (3):

$$R_{\gamma}(k) = \frac{1}{N}\sum_{i=1}^{N} \gamma(i+k)\gamma^{*}(i), \qquad \text{(EQ 3)}$$

where $\gamma^{*}(i)$ is the complex conjugate of $\gamma(i)$. The values $R_{\gamma}(k)$ are then usually normalized with respect to $R_{\gamma}(0)$ and denoted by $\tilde{R}_{\gamma}(k)$. Since for cyclostationary processes the average power is given by $R_{\gamma}(0)$, the baseband signal is thereby effectively normalized by its average power. The normalized central second-order moment of $|\gamma(n)|$ is given by equation equation (4):

$$\tilde{\eta}_2 = \frac{m_2}{m_1^2} - 1 \qquad \text{(EQ 4)}$$

where m_1 and m_2 are defined in equations equation (5) and equation (6) as:

$$m_1 = \frac{1}{N}\sum_{i=1}^{N} |\gamma(i)| \qquad \text{(EQ 5)}$$

$$m_2 = \frac{1}{N}\sum_{i=1}^{N} |\gamma(i)|^2 \qquad \text{(EQ 6)}$$

Benvenuto's FORTRAN pseudo-code for computing these discriminant variables is shown below as it appeared in [6]. Preceding the code is a description of the variables.

n	discrete time signal index
NSMPL	length of input signal
$\gamma(n)$	complex valued sequence following Weaver demodulation
$P_{Th} = \gamma_{Th}^{2}$	noise rejection threshold
RO	zero'th autocorrelation lag value

$R2$	second autocorrelation lag value
ET	intermediate value
A	intermediate value
B	intermediate value
P	estimated average power over interval L
N	segment length
L	subsegment length
i	DO loop index

```
START:       $n = 0$
BEGIN:       $n = n + 1$
             IF ($n >$ NSMPL) STOP
             IF ($|\gamma(n)| < \gamma_{Th}$) GO TO BEGIN
INITIALIZE:  $R0 = 0, R2 = 0, ET = 0, q = 0$
COMPUTE:     $P = 0, A = 0, B = 0$
             FOR ($i = 0$) TO ($L$ - 1) DO
                 $P = P + |\gamma(n)|^2 / L$
                 $A = A + u(n+2)u(n) + v(n+2)v(n)$
                 $B = B + |\gamma(n)|$
                 $n = n + 1$
             END DO
             $n = n - 1$
             IF ($P < P_{Th}$) GO TO BEGIN
UPDATE:      $R0 = R0 + (L)(P)$
             $R2 = R2 + A$
             $ET = ET + B$
             $q = q + 1$
             IF ($q <= N / L$) THEN
                 $n = n + 1$
                 GO TO COMPUTE
             ELSE
                 OUTPUT: $n$, $\tilde{R}_\gamma(2) = -R2 / R0$, $\tilde{\eta}_2 = R0 / ((1 / N)\, ET^2) - 1$
                 GO TO BEGIN
             END IF
END:
```

Note that the algorithm breaks a signal segment N samples long, into subsegments of length L. Each subsegment must have a minimum average power to be eligible for inclusion in the discriminant variable computation. If a subsegment does not contain enough power then the entire segment is rejected. An improvement to this algorithm is to throw out subsegments without enough average power, but to retain the previous subsegments for a limited time. This allows brief silent periods between speech activity to be tolerated by the algorithm, resulting in more effective use of speech signals.

Benvenuto found experimentally that the normalized second lag, $\tilde{R}_{\gamma}(2)$, is sufficient for separating voice from VBD when used as a discriminant along with the central second-moment. These functions yield two discriminants capable of classifying voice and non-voice signals using segments as brief as 32 ms. The presence of signal activity can be verified by estimating the average power of the segment. Benvenuto shows that using these two discriminants, with 32 ms signal segments, speech is misclassified as VBD about 1% of the time. Conversely, VBD is rarely misclassified as speech if the decision boundaries, i.e. the discriminant thresholds, are well chosen. The scatter plot in Fig. 9 shows how well Benvenuto's method can separate speech from one type of VBD signal. The VBD signal used to generate the plot was a V.32bis modem. The 'x' marks indicate VBD discriminant function computations, and the 'o' marks indicate speech discriminant function computations. A simple classification procedure would be to allocate any observations that fall in the lower left quadrant to the VBD class, and all other observations would be allocated to the speech class.

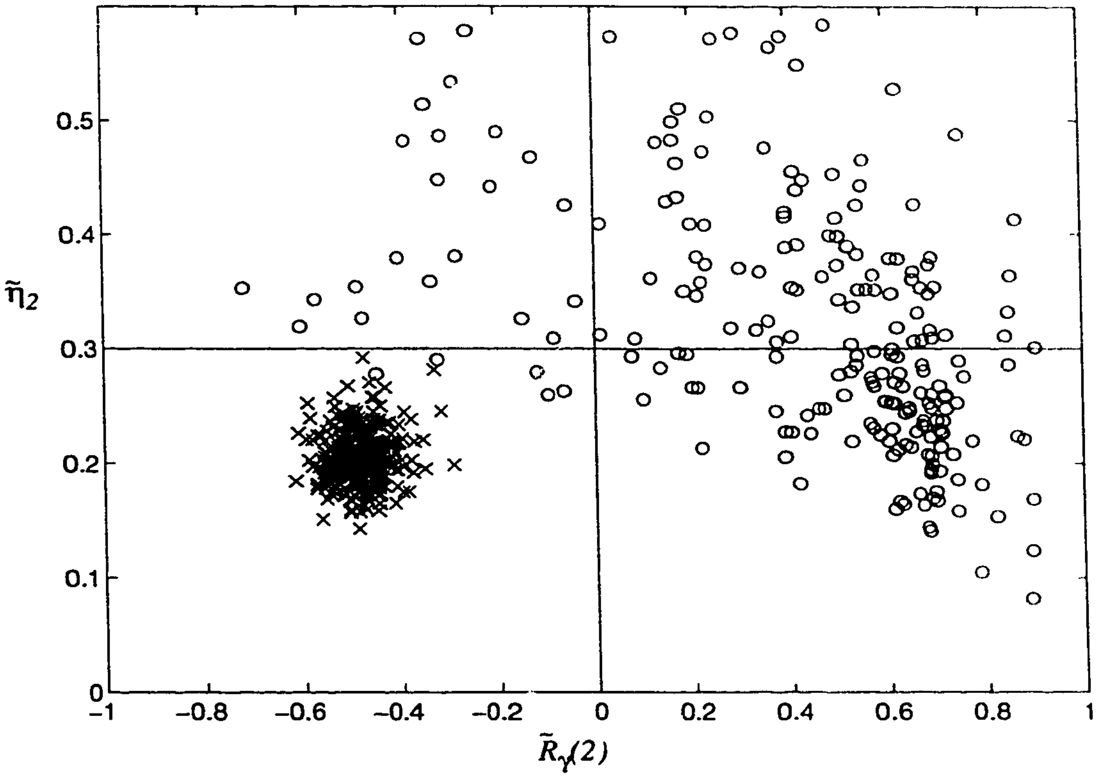


FIGURE 9. Scatter plot of Benvenuto's discriminant variables.

Benvenuto also considered several other VBD signal types including FSK, PSK, and QAM. (The specific standards were AT&T 103J, 202S, 212A, 201C, 208B, 2096A, IBM 3864-1, and CCITT V.29.) Benvenuto notes that the central second-order moment is an approximate indicator of the constellation complexity. We verified this claim for vastly different signals such as FSK and QAM. For QAM this discriminant varies greatly; however, this discriminant is not useful for subclassifying VBD signals with similar constellation types, such as 16-QAM and 8-PSK. This discriminant appears mainly useful for identifying FSK signals, which are rarely used by modern modems and facsimile machines (except for signalling).

The three papers written by Benvenuto that have been referenced here [8, 6, 7] provide a good starting point for a classification system. However, the discriminant variables are not carefully evaluated for their relative statistical significance, nor are various discriminant functions examined for their optimality properties. For these reasons, we decided to reconsider Benvenuto's original discriminant variables and possibly derive new ones. Applying the resulting variables to a set of modem signals, including recently developed 28.8 kbps modems, was also considered a worthwhile extension of Benvenuto's investigation.

Another question that remains is why Benvenuto used a complex demodulation stage in his experiments. Benvenuto argues that it is more convenient to operate on the approximately demodulated baseband signal [7]. If this stage could be avoided, however, the computational complexity of Benvenuto's algorithm could be dramatically reduced.

2.2 Available Products

2.2.1 CTel NETMONITOR System 2432

Currently there are at least three products on the market that address the problem of signal classification in telephone networks: CTel (Compression Telecommunications Corporation, Germantown, Md, USA) sells the NET-MONITOR System 2432 that has the ability to monitor the network, recording call type (voice, FAX, data), duration, disposition, and channel occupancy among other statistics [16]. Also, certain quality factors are monitored such as echo path loss, echo path delay, signal level, noise, and PCM coding errors. Several signaling environments are supported including E&M, R2D, CCITT #5, and CCITT #7 on T1/E1 trunks. The system is supported by a text or windowing-based software package that can report on the performance of several trunks over independent time intervals. Several trunks are monitored by multiplexing them into the monitoring system, hence simultaneous monitoring of more than one trunk is not possible. DSPs within the monitoring systems perform traffic classification, tone detection, demodulation, and spectral analysis.

2.2.2 Tellabs Digital Channel Occupancy Analyser

Tellabs Limited (Lisle, IL, USA) also sells a machine that performs similar functions to that of the CTel system. It is called the Tellabs Digital Channel Occupancy Analyser (DCOA). The system may monitor up to 120 digital channels (5 PCM trunks), distinguishing voice from data traffic [72]. The system runs on a personal computer (PC) and provides software for generating reports. The analysis of 10 signal channels requires 1.1 seconds. TS16 line signaling is supported, but Tellabs claims that with slight alterations other trunk types may also be monitored.

2.2.3 AT&T Voice/Data Call Classifier

The third product, called a Voice/Data Call Classifier, is made by AT&T. The system is designed for DS1 trunks, where it can monitor all 24 channels simultaneously [2]. Calls are classified as being either voice or voiceband data. Data calls are further classified by their speed as being high (>7199 bps), medium (4800 bps), low (2400 bps), very low (<1201 bps), or unknown. (The criteria for this determination is not known. Some modulation standards such as V.32bis allow for several different fall-back modes, each having a unique bit rate. This makes it difficult to determine the precise bit rate of a signal without sophisticated demodulation techniques. If such techniques are not used, then the classified bit-rate may only be the upper limit of a particular signal.) Two facsimile rates are also identified. The system requires continual access to the DS1 trunk in order to establish call boundaries. Included with the system are two 600 MByte hard disk drives, the Unix operating system, and two POTS (Plain Old Telephone System) lines for remote access via modems. The entire system requires 500 Watts of power.

2.2.4 MPR Teltech Service Discrimination Unit

MPR Teltech (Burnaby, BC) developed a device called the "Service Discrimination Unit" (SDU). In addition R.A. Law, et al., published a paper in 1991 on the capabilities of the SDU [50]. The SDU is designed to operate at the T1 level. Inside the SDU on each processing card are up to 4 Motorola DSP56001 digital signal processors, all controlled by two Motorola 68HC11 processors. Up to 10 processing cards can be included in a single SDU, with each card populated by a full complement of processors that are able to monitor a single T1 line.

The SDU is intended to be used for monitoring specialized FAX networks. These networks offer discounted FAX transmission rates and thus must be monitored to ensure that the channels are not used for speech calls. All types of FAX traffic are discriminated as well as certain VBD signal types including Bell 103, Bell 212, CCITT V.22, and V.22bis. (It is not clear which FAX modulation standards are identifiable.) The system is capable of not only monitoring the individual DS0's, but it can also interrupt service. Even though the system uses four DSPs running at 24 MHz and two controller processors for each T1 line, it cannot perform full-time monitoring of all DS0 channels if all 24 are in use. Under optimal conditions, 16 DS0s may be monitored full-time from a single DS1 stream.

The signal discriminator uses a conditional pre-filter to remove DC components and to compensate for noisy lines, a band-energy estimator, and a tone detector. Thus signals are generally classified according to their spectral properties by using a bank of infinite impulse response (IIR) band-pass filters. Signal segments containing 128 samples are used to compute the discriminant variables. Short-time energy variances are also used to identify speech segments. The authors report that typical response times are 2 to 3 seconds. We can conclude that this device uses a computationally intensive discrimination algorithm, yet has very poor response times and is not capable of discriminating modern VBD signals. We also suspect that FAX traffic is simply identified by observing the initial handshaking tones, probably accounting for the inclusion of a tone detector in the classifier.

2.3 Summary

Although judging from the volume of published work it would seem that the classification problem has already been solved, this is not true at all. The Tellabs DCOA system has been tested on-site by a telco sponsor of TRLabs and has been reported to be 72% accurate in its call classification in one evaluation study. As of yet, the accuracy of the other two systems has not been tested, nor is it published by the manufacturers. In fact, the manufactures have refused to release models of their products for evaluation, making one even more doubtful of performance claims. In addition, no theoretical basis or treatment of methods used by these systems has been disclosed or published in the open literature.

Another drawback that these systems share is their reliance on knowledge of call boundaries. A desirable feature would be to examine a call at any point in time, and successfully classify it. By knowing the call boundaries, the classification problem is greatly simplified since the setup signal protocol for FAX and data calls is well known, and easily identifiable. A classification system that is accurate without requiring knowledge of the call boundaries will be even more accurate if that knowledge is also provided.

An additional drawback is the sheer magnitude of the existing machines. They are physically large, consume a great deal of power, and cost a great deal of money. This is especially true of the AT&T system. Often multiple DSPs operate concurrently to perform classification. It would be very attractive to have a machine (ultimately a single chip, perhaps) that is far simpler in design. The goal of this thesis work is to develop algorithms for performing classification for a wider variety of signal types, that requires far less computing power, and achieves a higher degree of accuracy than the systems considered. The resulting algorithms should also not rely on call boundary information, and should be able to operate in an environment where full-time monitoring of all channels is not necessarily possible.

Chapter 3

3.0 Background Material

Chapters two, three, and four contain the background material that is required for complete understanding of the algorithms presented in Chapter 5. Combined, these chapters represent about half of the content of this thesis. This extensive treatment of the groundwork leading up to Chapter 5 is necessary for the reader to be able to refer back to vital information. For example, all of the modem standards classified need to be discussed in order to make the reader aware of their relative differences. If the reader is already familiar with the background material, they can begin reading Chapter 5 immediately and only refer to the background sections when necessary.

3.1 Characteristics of Analogue Telephone Channels

Essential to understanding the problem at hand is having a clear picture of the total telephone network, from the handset or data device at your site through the vast global digital telephone network, terminating at a destination device. The total network behaves as a virtual single machine, the single largest machine in the world. The network interconnects widely different communication mediums including wire underground, above ground, and in the ocean; fiber optics spanning between major cities; radio towers on hilltops; and satellite communications in the heavens. For the vast majority of the network infrastructure in North America, digital communication is used exclusively. Only the vast investment in the millions of copper subscriber loops has so far prevented the entire network from being digital right up to the customer's premises.

This thesis work is concerned with monitoring the network from within, at any point in the network where the signal traffic is digital. Even an analogue line could be monitored with the appropriate digitizing equipment. However, to limit the scope of this project, only a subset of the variety of communications mediums will be considered even though the theoretical aspects of the solution could be applied to any digital voice-band signal. The medium being considered are the 24 channel, 1.544 Mbps DS1/T1 signals that are sent and received at the COs (Central Offices). COs are the locations where analogue telephone loops that lead to and from our wall jacks are terminated and digitized into bidirectional Pulse Coded Modulation (PCM) streams of digital data. Exiting the COs may be DS1 (or larger) trunks which lead to the next level in the hierarchy of the network, where switching of calls can take place. Each DS1 signal contains 24 tributary signals called DSOs as well as a few synchronization bits known as *framing bits*. Each DSO contains one direction of a customer's call. Each DSO carries 8000 samples per second, where each sample is an 8-bit A-law or μ-law codeword. Companding using either the A-law or μ-law standard ensures a roughly equal signal-to-noise ratio at all signal amplitudes.

The following is a discussion of some of the technical limitations that the telephone network imposes upon the traffic it carries. These limits and impairments become important later on when we attempt to simulate VBD signals as they are transmitted through the PSTN.

3.1.1 Model of a Point-to-Point Connection

The telephone network as seen by subscribers can be modeled as shown in Fig. 10. The entire point-to-point connection (from one end to the other) consists of a handset at each end (FAX machine, modem, or telephone), a pair of two-wire subscriber loops, two hybrids, and fully digital network interconnection stages. (It is possible for the interconnection stages to also require hybrids in some cases.) The four-wire portion of the network is where the two directions of a call are transmitted using two separate channels. Each direction is said to utilize a pair of wires (implementations may not require a pair of wires, however).

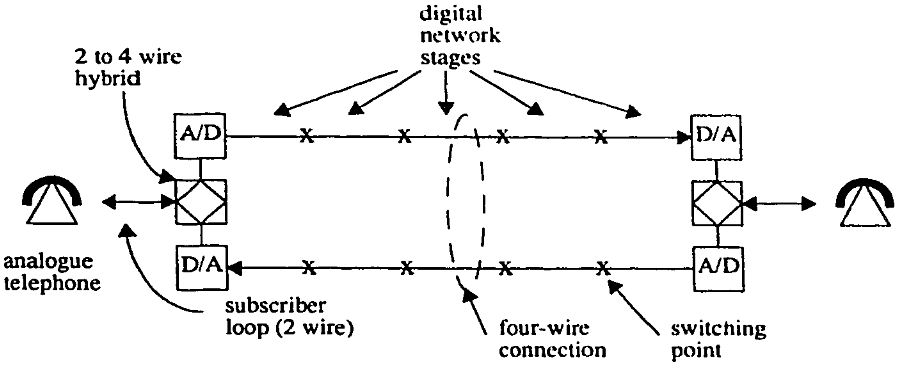


FIGURE 10. Model of point-to-point connection.

Currently, most homes and businesses only have analogue service to the wall outlet. Connected through the wall outlet to the two-wire subscriber loop may be a telephone, FAX machine, data modem, or some other device (such as a communication device for the hearing impaired). All three of these devices must have standard analogue telephone interfaces and thus must conform to pre-determined standards regarding their transmitted signal power and bandwidth. ITU Recommendation V.2 [30] specifies the power levels that should be used by VBD devices that are connected to the PSTN.

The subscriber loops are connected to the digital system via hybrids at the first CO stage. A hybrid is an interface between a two-wire signal and a four-wire signal. Either within the hybrid or close to it are CODECs (coder/decoder) for converting analogue signals to and from companded PCM streams. Wherever possible, analogue transmission lines are implemented as differential signals for noise rejection. The two-wire loop that reaches the subscribers is a current loop signal.

The center of Fig. 10 shows two lines with several "x" marks on them. Each "x" represents a switching point in the network, and the spans between "x" marks represent the digital trunk connections. (Only rarely in North America are these trunks not digital.) There is a limit to the number of links that a connection may pass through due to standards protecting the GOS (Grade of Service) or line quality. One advantage that digital signals have over analogue signals is that there is no power loss caused by switching or transmission. Losses may be intentionally introduced at digital switching stages to meet network design goals. Also, unless a digital bit error occurs, the received digital signal is identical to the transmitted signal. Timing delays are still present however. The losses and delays for each stage are selected when designing the network to conform with appropriate performance standards related to echo propagation and stability.

3.1.2 Voice Network Complications and Design Considerations

In the following sections we will discuss some of the complications, limits, and impairments affecting PSTN VB channels. An excellent reference describing the various conditions that can be expected on a North American call is the Bell Labs 1982/83 End-Office Connection Study (EOCS) [12]. This study will be referenced several times when discussing network impairments.

3.1.2.1 Echo Delay

Audible echoes resulting from electrical signal reflections are produced wherever there is an impedance mismatch in an electrical connection. Major sources of echo in the telephone network include the connection between the handset and the two-wire loop, bridged taps on the two-wire loop, and the hybrids. The hybrids also cause echoes because they do not perfectly separate the transmit and receive channels from the two-wire loops. In other words, there is signal leakage from the receive channel to the transmit channel.

Two types of echoes are generally considered: talker echo and listener echo. *Talker echo* occurs when the talker hears their own voice after some delay, similar to the effect of yelling across a valley. *Listener echo* occurs when the listener in a conversation hears the talker once, and then again after a delay. This is similar to being on the opposite side of a valley from a screaming fool. You hear the fool scream once, then again after a delay, and possibly even several more times.

These echoes are always present to some degree within the telephone network, however they are usually after such short delays that we do not notice them. Echoes imperceptible to the human ear, however, may still disturb the performance of high-speed electronic hardware. Modems must take the echo delay problem very seriously, and usually incorporate echo cancellation technology. This usually consists of a circuit that subtracts a delayed and attenuated version of the transmitter signal from the received signal. The exact delay value to be used can be determined by using cepstrum analysis, correlation, or some other method at call setup [11].

Echoes can cause significant effects with respect to the classification problem. Since some telephone networks already provide echo cancellation, the echo problem may be non-existent or minimized. However, significant echoes may be present in poorer connections, in which case the classification algorithm will need to consider their presence.

The first step in understanding the effects of echo delay as it relates to this project is to incorporate echoes in any simulations performed and to observe the changes that take place. Since the ITU (International Telecommunication Union) has specified maximum acceptable limits of echo delay and power, that information was used to generate simulated echo situations. (The ITU-T was formerly called the CCITT, which is an acronym for International Telegraph and Telephone Consultative Committee. The ITU encompasses the ITU-T, and in the past the CCITT.) Data from real telephone connections may not reflect the worst case echo delays since telephone systems in North America are relatively well designed for the most part, and the echo delay performance measures are a statistical process where the worst case echoes rarely occur.

Since the power of echoes is usually attenuated to a level much lower than that of the original signal, their effects on the classification performance were not found to be very significant in most cases. However, when classifying split-band modems it is important to consider the presence of echoes since the echo power may be in a separate frequency band from the main signal power. Thus any complete classification method based upon the spectral properties of a signal must consider echoes.

3.1.2.2 Stability (Ringing and Singing)

This phenomenon is an important design consideration for network designers; however, it doesn't apply to the proposed problem. The phenomenon is this: The signal gain at each stage of the network must be carefully planned to protect the end-to-end signal from oscillating (or *ringing*). If a line signal is about to start oscillating, it will exhibit what is called *singing*, where the talker's voice has an eerie ringing (or often described as a "rainbarrel" sound) effect added to it. By properly providing sufficient losses in the handsets and interconnection stages of the network, ringing can be avoided. A classifier might have difficulty identifying a call type in the presence of ringing, but this would be a fault of the network equipment and would be beyond the control of the proposed system.

3.1.2.3 Volume (Loss)

When designing handsets or any of the analogue equipment connected to the subscriber loop, the volume of the microphone and speaker combination must be considered. The ITU has set specifications for the acceptable volume level. In the case of modems, the standard response is to use an automatic gain control device to adjust the line level to conform to ITU standards. This will be a consideration for the proposed system because it determines the expected signal power levels for data and FAX signals. Since telephones don't continually set their transmitted power level, voice signal power fluctuates widely as a consequence of the variable volumes in normal human speech patterns. This is certainly one feature to consider when trying to differentiate between voice and voice-band data. Voice-band data calls will show constant average power levels since that is specified by the ITU in the newer V.34 standards. The power level is maintained in these systems even during constant zero or one data transmissions because of the use of a randomizing scrambler at the transmitter, and the transmitted signals are carrier based.

Related to the volume issue is the flat attenuation or loss for a connection. This is usually measured by transmitting a 1004 Hz test-tone over an end-to-end connection. The received power is compared to the transmitted power level to indicate the end-to-end loss. According to the 82/83 EOCS [12], the mean customer premises-to-customer premises 1004 Hz loss was measured to be about 16.5 dB.

3.1.2.4 PCM Coding and Compression

When an analogue telephone signal is digitized, normally μ-law or A-law PCM (Pulse Code Modulation) is used. μ-law and A-law refer to specific standard transfer functions that are used to perform non-linear amplitude compression, usually known as *companding*. With these methods, greater digital precision is given to small signal values than large signal values so as to obtain roughly the same signal-to-noise ratio in the presence of quantization noise (defined later) at all volumes.

PCM is a method of converting an analogue signal into a digital stream by periodically quantizing the incoming signal and then encoding the quantized values as digital words. Each digital word is then transmitted through the telephone network serially. A standard CO in North America uses μ-law PCM, with 8 bits per word. This leaves 255 quantization levels in total to be used for both the positive and negative signal values. (There are 256 possible word values, however only 255 of them encode distinct values because 10000000_2 and 00000000_2 both represent zero.) Table 2 gives the μ-law encoding/decoding table, while Table 3 is for A-law [3]. The most significant codeword bit is a sign bit (0 for positive, 1 for negative).

TABLE 2. Encoding/decoding table for mu-law PCM.

Input Amplitude Range	Step Size	Segment Code S	Quantization Code Q	Decimal Code Value	Decoder Amplitude
0-1	1	000	0000	0	0
1-3	2	000	0001	1	2
3-5	2	000	0010	2	4
...	...	...	...	...	...
29-31	2	000	1111	15	30
31-35	4	001	0000	16	33
...	...	...	....	...	...
91-95	4	001	1111	31	93

TABLE 2. Encoding/decoding table for mu-law PCM.

Input Amplitude Range	Step Size	Segment Code S	Quantization Code Q	Decimal Code Value	Decoder Amplitude
95-103	8	010	0000	32	99
...	...	...	...	...	...
215-223	8	010	1111	47	219
223-239	16	011	0000	48	231
...	...	...	...	...	...
463-479	16	011	1111	63	471
479-511	32	100	0000	64	495
...	...	...	...	...	...
959-991	32	100	1111	79	975
991-1055	64	101	0000	80	1023
...	...	...	...	...	...
1951-2015	64	101	1111	95	1983
2015-2143	128	110	0000	96	2079
...	...	...	...	...	...
3935-4063	128	110	1111	111	3999
4063-4319	256	111	0000	112	4191
...	...	...	...	...	...
7903-8159	256	111	1111	127	8031

TABLE 3. Segmented A-law encoding/decoding table.

Input Amplitude Range	Step Size	Segment Code S	Quantization Code Q	Decimal Code Value	Decoder Amplitude
0-2	2	000	0000	0	1
2-4	2	000	0001	1	3
...	...	...	...	...	...
30-32	2	000	1111	15	31
32-34	2	001	0000	16	33
...	...	...	....	...	...
62-64	2	001	1111	31	63
64-68	4	010	0000	32	66
...	...	...	...	...	...
124-128	4	010	1111	47	126
128-136	8	011	0000	48	132
...	...	...	...	...	...
248-256	8	011	1111	63	252
256-272	16	100	0000	64	264
...	...	...	...	...	...

TABLE 3. Segmented A-law encoding/decoding table.

Input Amplitude Range	Step Size	Segment Code S	Quantization Code Q	Decimal Code Value	Decoder Amplitude
496-512	16	100	1111	79	504
512-544	32	101	0000	80	528
...	...	...	...	...	...
992-1024	32	101	1111	95	1008
1024-1088	64	110	0000	96	1056
...	...	...	...	...	...
1984-2048	64	110	1111	111	2016
2048-2176	128	111	0000	112	2112
...	...	...	...	...	...
3968-4096	128	111	1111	127	4032

A fundamental problem in PCM is quantization noise. When a signal is sampled and the sample is converted into a digital value, the set of possible values is limited. In the case of T1 lines, there are only 255 possible quantization levels. The closest quantization level to the actual sampled signal value is chosen for transmission. The difference between the resulting digital value and the sampled signal value is called the *quantization error.* *Quantization noise* is the effect of all quantization errors in the signal. When the digital signal is converted back to an analogue signal, the resulting signal will contain small random inaccuracies due to this source of noise.

Companding is a means of reducing the effects of quantization noise. Companding allocates more quantization levels, with smaller intervals between them, at small signal levels. The trade-off is that high signal values are allocated fewer intervals, and are thus subject to larger absolute quantization errors.

Another limitation of PCM systems is the sampling frequency. Nyquist's theorem states that to capture all of the information in an input signal up to a certain frequency f_0, then we must bandlimit the signal to f_0 and sample it at a rate of at least $2*f_0$. Any frequency information above half the sampling rate will be *aliased*, that is, be mixed up irreversibly with lower frequency information. To prevent aliasing, filters are normally placed before quantizing. To get acceptable telephone signals we want a bandwidth of about 3500 Hz. Thus we sample at 8000 samples per second. Eight bits per sample yields an adequate SNR, and implies a bit rate of 64 kbps per channel. This bandwidth is reserved for each direction of a telephone connection through the rest of the telephone system.

With respect to the classification problem, the companding of the incoming data must either be removed or at least considered when the signal is being studied. Also, the bandwidth limitations imposed by the quantization frequency must be considered.

3.1.2.5 Attenuation Distortion

Attenuation Distortion (AD) is also known as the frequency response of a channel, and is measured in terms of frequency-dependent loss relative to the 1004 Hz power level. The EOCS [12] gives a plot of the measured AD for short, medium, and long connections. Note that these results are only for tests performed from end-office to end-office. The results of a previous loop only study are also shown in the EOCS, and combined with the end-office results to form a customer-premise-to-customer-premise result. The AD plots for all three connection types are nearly identical, suggesting that the largest factor affecting AD is the frequency response of the codec filters. This is an important fact to consider when designing a classifier that is based upon the spectral properties of a signal. We can expect nearly all connections through a modern digital network to have the same basic channel frequency response.

3.1.2.6 Envelope Delay Distortion

Envelope Delay Distortion (EDD) is frequency-dependent and defined as the difference between the envelope delay at the present frequency with respect to the envelope delay of a given reference frequency. The *envelope delay* is defined as the negative of the derivative of the phase of the received signal with respect to the reference frequency [12]. The EOCS also contains measurements of EDD over short, medium, and long distance connections. Again, the EDD is not strongly dependent upon the length of a connection. EDD is mainly affected by the length of the 2-wire loop that connects the customer premises to the CO. This is also important to consider when designing a classifier, since most channels will have roughly the same EDD characteristics within a typical range.

3.1.2.7 Frequency Offset

Frequency offset is a simple phenomenon that has been virtually eliminated from modern data modem and FAX connections because of the widespread use of highly accurate and stable crystal-stabilized clock generators in these devices. Another source of frequency offset is the networks itself, especially where frequency division multiplexed (FDM) trunks are used. Since FDM is rarely used for analog network signals, frequency offsets have been virtually eliminated from the PSTN.

Frequency offset is a linear shift in frequency of a transmitted signal. For example, if the frequency offset of a channel is +1 Hz, a transmitted 1000 Hz tone would appear to be a 1001 Hz tone at the receiver. Frequency offsets rarely exceed 1 Hz, even though most modem equipment can tolerate a plus or minus 7 Hz frequency offset. Frequency offsets of this magnitude are not audible to most people. (Some individuals have the ability to precisely determine the pitch of a sound. These people are said to have "perfect pitch".)

3.1.2.8 Additive Noise

The *additive noise* in PSTN channels is *colored* rather than *white*. That is, the noise is non-Gaussian, band limited, and the power spectrum of the noise is shaped. Several factors contribute to additive noise including crosstalk, power line coupling, companding, digitization, and so on. Additive noise is normally measured using a weighting filter to model the subjective effects of noise on human hearing. These measurements will be discussed further in a later section.

3.1.2.9 Other Analogue Impairments

A good reference on the subject of analogue impairments is W.D. Reeve's book [63] "Subscriber Loop Signaling and Transmission Handbook." Other impairments discussed in this book include crosstalk, second-order intermodulation distortion, third-order intermodulation distortion, phase jitter, amplitude jitter, phase hits, gain hits, and general line dropouts. Another good reference on the effects impairments have on data signals is Bingham's book [11] "The Theory and Practice of Modem Design." We do not emphasize these impairments here since they have a lesser effect upon the ability of our classification algorithms to perform their function, and these impairments are less common in digital transmission networks.

3.2 Common Data Modulation Methods for Telephone Channels

Of the multitude of possible data modulation methods to choose from, only a few are used to a significant extent on voice-band telephone channels. Older communication standards utilized simple modulation methods such as AM (Amplitude Modulation), FM (Frequency Modulation), and FSK (Frequency Shift Keying). However, mainstream data and FAX communications no longer use these standards for high-speed connections due to their limited abilities to utilize the full available channel bandwidth. Consequently, these obsolete methods will not be considered in this thesis. (In fact, FSK is still considered in this thesis as it pertains to the signaling method used by FAX machines.) Today the bulk of digital communications is performed at speeds of 2400 bps and greater using sophisticated phase and amplitude modulation methods. The current maximum data rate is 28.8 kbps, which is very near the theoretical limit of roughly 30 kbps for the capacity for a telephone channel [51]. This limit can be determined using Shannon's theorem for channel capacity, an estimate of the signal-to-noise ratio, and the channel bandwidth. Refer to equation (7) for a sample calculation where the bandwidth W is 3000 Hz, the signal-to-noise ratio P/N_0W is 30 dB, and the resulting units are bits-per-second. The main ITU-approved methods are variations of passband PAM (Pulse Amplitude Modulation). PAM is a modulation method that transforms an input bitstream into a sequence of amplitude- and phase-encoded complex symbols (according to Lee and Messerchmidtt [51]). The symbols are subsequently frequency shifted into the passband by multiplying by a sinusoidal carrier.

$$C = W\log\left(1 + \frac{P}{N_0W}\right) = (3000)\log\left(1 + 10^{30/10}\right) = 29,901 \qquad \text{(EQ 7)}$$

A general passband PAM transmitter is shown in Fig. 11. The input to the transmitter is a serial stream of bits which are converted to two-coordinate symbols in the coder. The set of possible symbols is referred to as an *alphabet*. Alphabet symbols may contain both real and imaginary components. A plot of the alphabet on a graph with real and imaginary axes is referred to as a *constellation map*. The next stage in the transmitter passes the symbol stream, which is still a discrete signal, through a transmit filter. The transmit filter is also commonly called a *pulse shaper.* Its pu·pose is to generate a continuous analogue (or high sample rate discrete) signal from the incident stream of discrete symbols.

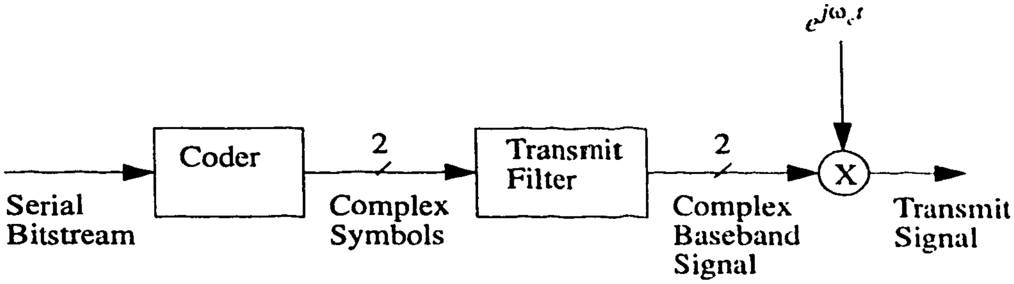
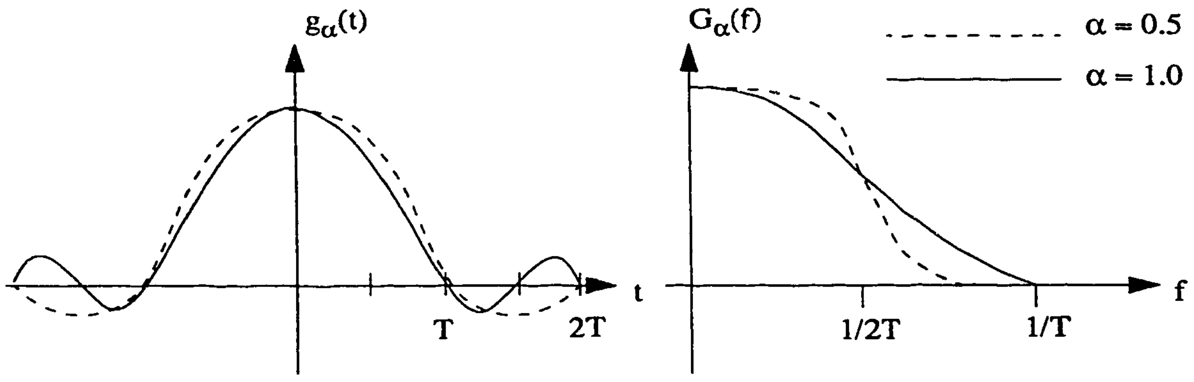


FIGURE 11. General passband PAM transmitter

The pulse shaping task is normally performed by a *raised-cosine filter.* The shape of the filter is defined by the ITU specifications so that the power spectrum of the transmitted signal is appropriate for voice-band transmission. Pulse shaping filters the time series of symbols to strive for minimum intersymbol interference within the available bandwidth of the transmission channel. Equation (8) defines the generalized time domain representation of a raised-cosine filter. The 'α' term varies inversely with the roll-off (steepness) of the filter. The raised-cosine essentially looks like a modified 'sinc' filter in the time domain, however it doesn't resemble a sinc function in its frequency response. The equation and diagrams of Fig. 12 illustrate the shape of an ideal raised-cosine pulse shaping filter in the time (a) and frequency (b) domains. Note that implementations of raised cosine filters cannot provide the noncausal impulse response of the ideal filter. There is in fact some variation in how real filters are implemented.

$$g_{\alpha}(t) = \left(\frac{\sin\left(\pi\frac{t}{T}\right)}{\pi\frac{t}{T}}\right)\frac{\cos\left(\alpha\pi\frac{t}{T}\right)}{1-\left(2\alpha\frac{t}{T}\right)^2} \quad \text{(EQ 8)}$$



(a) (b)

FIGURE 12. Raised-cosine filter (a) impulse and (b) frequency response.

The output from the transmit filter is a complex baseband signal that is ready to be frequency shifted and transmitted as shown in Fig. 11. This is performed using the multiplier and sinusoid (or a pair of orthogonal sinusoids) at the carrier frequency. Finally, the signal is transmitted through the subscriber loop.

3.2.1 FSK

Frequency Shift Keying (FSK) is an important modulation method that is used for signaling in FAX communications. FSK is a popular signaling method since the corresponding modulator and demodulator are easy to implement. Also, FSK signals may be either *coherently* or *incoherently* detected (explained later), allowing for even more flexibility in the design of receivers.

Normally binary frequency-shift keying is used. In this method there are two frequencies that can be transmitted, say f_1 and $f_2 = f_1 + \Delta f$. The modulator simply encodes a logic zero as a tone at frequency f_1, and a logic 1 as a tone at frequency f_2. Fig. 13 shows an example of an FSK modulator [65].

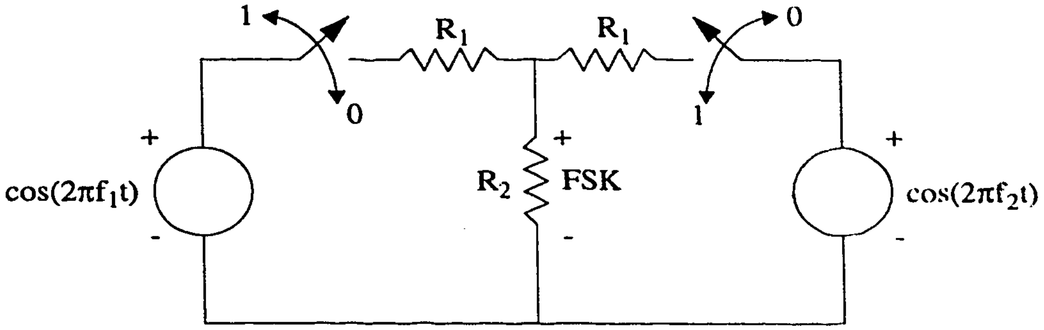


FIGURE 13. Example of an FSK modulation method using switches.

Another interesting, and popular, method for modulating an FSK signal is shown in Fig. 14. Here a voltage controlled oscillator (VCO) is used to alter the frequency of the output signal. The input to the VCO is a bipolar non-return-to-zero (NRZ) signal. In a bipolar signal logical zeros are encoded as a particular voltage level, while logical ones are encoded as the negative of the voltage level of logical zeros. NRZ signals remain at the respective voltage level for the duration of a symbol, rather than returning to zero prior to transmitting the next symbol. This method has gained its popularity since it is easy to implement in hardware or on a DSP platform.

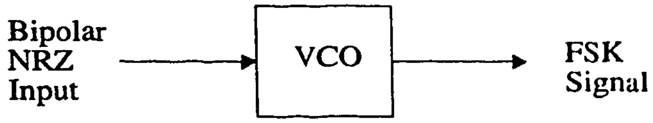


FIGURE 14. Example of an FSK modulation method using a VCO.

When demodulating an FSK signal, either coherent or incoherent methods may be used. A standard matched filter coherent detector (where the symbol timing is known and can be exploited) may be used; however, this is not the simplest method. Roden also points out other technical drawbacks to this method [65]. This leads us to the incoherent methods. A simple and practical *incoherent detector* is shown in Fig. 15. Here the output of two narrow bandpass filters tuned to f_1 and f_2 are sent to envelope detectors and compared. If there is power at frequency f_1 (f_2), then z_1 (z_2) will be the envelope of the f_1 (f_2) sinusoid. The output of the system is a bipolar NRZ signal, just like the signal that was input to the second modulator model.

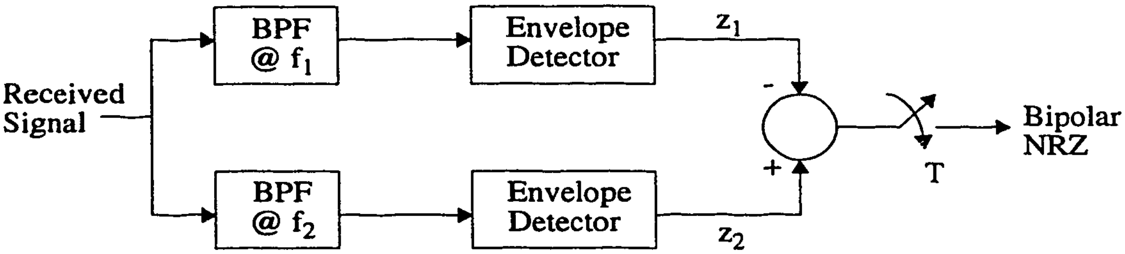


FIGURE 15. Example of an FSK incoherent detector using tuned BPFs.

3.2.2 DPSK

Differential Phase Shift Keying (DPSK) is primarily used for low to medium speed (4800 bps and less) data and FAX transmissions. DPSK is a specific variation of passband PAM that was popular prior to the adoption of QAM protocols such as V.22bis. A possible constellation for DPSK is shown in Fig. 16. The constellation plot of Fig. 16 shows four constellation points, equally distributed around a circle and separated by $\pi/2$ phase differences. The amplitudes of each constellation point vector are identical, hence there is no amplitude modulation being performed. The only difference between points is the phase angle of the signal relative to an implied carrier.

DPSK modulation is based on the constellation points being separated by a phase difference, where coding is performed on the premise that the phase *difference* from one symbol to the next indicates the information being carried. With a four-point constellation

2 bits of data are carried per symbol. The data transfer rate is thus twice the symbol rate. Data transfer is measured in units of bits per second (bps). The symbol rate is also known as the ***baud rate***. The differential characteristic of DPSK is an attractive feature since the demodulator does not need to know the absolute phase position of the transmitted signal, only the relative phase changes. Also, the magnitude of the signal is not relevant to the information being carried. Of course magnitude is an important design consideration for optimal performance without distortion since a signal that is transmitted with too much power will overload the telephone loop receiver.

The constellation of a DPSK signal may have more or fewer than the four points shown in Fig. 16. One can verify that the maximum bit rate of a DPSK signal is equal to $\lfloor \log N \rfloor F_s$, where F_s is the symbol rate of an N-point constellation and the logarithm has a base of two.

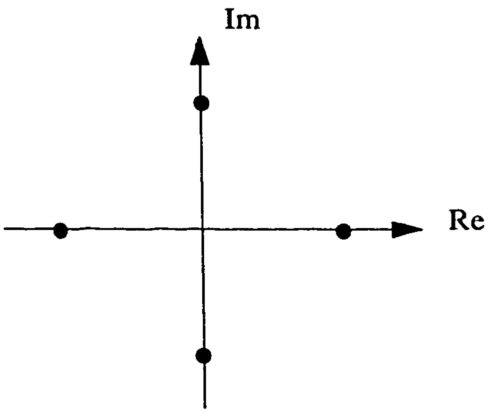


FIGURE 16. Possible four-point DPSK constellation.

3.2.3 QAM

An extension of DPSK is *Quadrature Amplitude Modulation* (QAM). QAM is the most popular modulation method in use for standard FAX and data communications. Rather than only modulating the phase angle of the signal, as in DPSK, this method also allows the signal amplitude to be modulated at the same time. The resulting constellations can be much more complex, containing up to thousands of points. A simple case of QAM is the four-point constellation, which is actually the same constellation as DPSK; however, the symbol mapping is arbitrary for QAM, whereas for DPSK the next state symbols are based upon the current symbol plus or minus some phase offset.

To perform QAM modulation, two data signals are modulated using orthogonal carriers (sine and cosine signals). Then, the two modulated signals are added together and shifted into the passband. Since the two component signals are 90 degrees out of phase, the resulting composite signal can be thought of as being two-dimensional. In a constellation plot illustrating this characteristic, the real axis corresponds to the modulated cosine component, and the imaginary axis corresponds to the modulated sine wave component. The imaginary signal component is often called the "*quadrature*" component. Fig. 17 illustrates a general QAM transmission system.

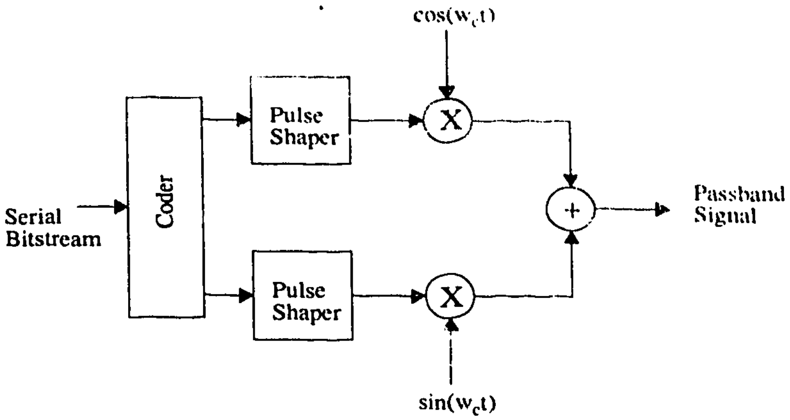


FIGURE 17. General QAM transmitter.

The main variations between the different QAM modulation methods lie in the structure of their signal constellations. To improve the speed of transmission in a QAM system, there are two options: The first is to increase the symbol rate. By increasing the symbol rate, the throughput (bit rate) of the modulator is increased in direct proportion. However, there are hard limits to the symbol rate that cannot be exceeded due to the band-limited characteristics of the telephone channel. So another method must be used for achieving greater speeds. By increasing the number of points in the constellation, the amount of information encoded by each symbol is increased. This also increases the bit rate without increasing the symbol rate. However, there are practical limits to the size of the constellation due to the noise inherent in a telephone line and the growing cost of the required transmitters and receivers. If too many symbols are used, the channel distortions will cause intersymbol interference that renders the signal undetectable. In fact, the constellations sizes that are used in many current modems have a probability of symbol error that is near one. This may seem hard to believe since it seems impossible to detect the transmitted symbols without error. It is only with the addition of coding techniques and sophisticated line equalization methods that the probability of symbol error can be reduced to a tolerable level. To illustrate this point we refer to a figure and equations in [62] on page 656. The equation for the tight upper bound of the probability of symbol error for an optimally detected QAM signal is given by equation (9), where M is the number of constellation points, E_{av}/N_o is the SNR/bit, and Q is known as the Q function. (The Q function is equal to one minus the cumulative distribution function of a Gaussian random variable. Since a primary noise source in communications is thermal noise, which has a Gaussian distribution, noise is often modelled as Gaussian.) Consider a case where M=128, and the SNR is 30 dB. This would correspond to a typical V.32bis modem signal. For this example, we compute P_M and find it to be 0.91. With this high of a probability of symbol error, the signal is not useful unless other methods are utilized to reduce this probability.

$$P_M \leq 1 - \left[1 - 2Q\left(\sqrt{\frac{3E_{av}}{(M-1)N_o}}\right)\right]^2 \qquad \text{(EQ 9)}$$

Constellation design is a major concern when dealing with QAM systems. An optimal design makes use of the full bandwidth available and is not overly prone to errors. In fact, there are ways of implementing QAM that have built-in error detection and possibly error correction.

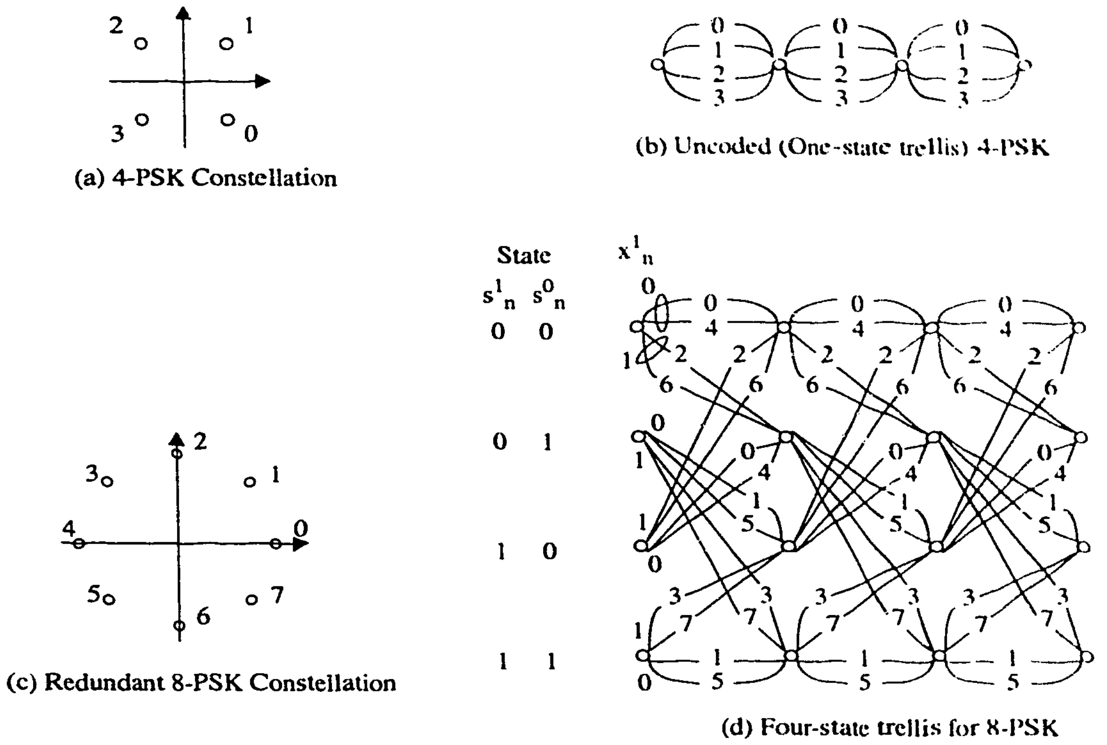
3.2.4 Trellis-Coded QAM

Trellis coding is a coding method that has been widely applied to QAM. Following standard terminology, a trellis-coded QAM system will be referred to as TC-QAM. In TC-QAM the coder is implemented as a finite state machine (FSM) [11, 51, 75, 74]. The modified coder implements *signal-space coding*. This means that symbol decoding does not just depend upon where the particular symbol lies in the constellation space, but it is now dependent upon the prior sequence of symbols. This dependence is what allows the coding gain in the system. *Coding gain* is an improvement in effective SNR (Signal to Nois Ratio) over the original signal; in this case the gain comes in the form of increased error immunity, or forward-acting error minimization [11]. The coding gain for a simple four-state TCM could be 3 dB. More complex TCM schemes can achieve 6 dB of coding gain, or more [74].

The advantage of trellis coding comes at a cost. To implement the coding, extra points must be added to the constellation that do not directly increase the bit-rate. This results in either a smaller minimum distance between constellation points, or increased transmitted power. These two disadvantages, however, are usually outweighed by the increased reliability of a multidimensional constellation space. Ungerboeck showed that implementing trellis coding can increase the SNR without compromising signal bandwidth at all [74]. The cost comes in the form of increased complexity of the transmitter encoder and the receiver detector.

Trellis coding utilizes the past and present values of M inputs to generate M_l present outputs, where $M_l > M$. The outputs are then mapped into a signal space containing 2^{Ml} distinct points. It has been shown [74] that with $M_l = M+1$, a 6 dB coding gain can be achieved. This is the type of trellis coding that is used in several modem standards such as V.32bis and V.32. The receiver of a trellis-coded signal must implement a soft-decision (decisions are based on past, present, and future information), maximum-likelihood sequence decoder for decoding the noisy received signal.

Consider the diagrams shown in Fig. 18. The codes shown here were the first trellis codes discovered by Ungerboeck in 1975 [74]. Figures 18 (a) and (b) are for a 4-PSK constellation with no coding. The trellis in this case has only one state, and is shown in Fig. 18 (b). Every connected path that can be taken through the diagram represents an allowed signal sequence. For the uncoded case, all sequences are allowed. Figures 18 (c) and (d) show an 8-PSK constellation and its corresponding four-state trellis. Note that in both systems, four transitions are permitted from every state, allowing for 2 bits to be encoded per symbol. The subsets (0,4), (1,5), (2,6), and (3,7) have maximum distances between them. Consider the top-left trellis state. From this state there are four possible paths to take, each corresponding to one of the possible two bit patterns that is to be sent. Transitions are performed for pairs of two parallel transitions. The first transition is performed for the first bit (x^1_n) of the two bit sequence to be encoded, and the second transition corresponds to the second bit of the sequence. If the trellis was formed such that all possible trellis states were valid from every current state, pairs of transitions would not be needed. The trellis shown in Fig. 18 (d) was found heuristically, and performs bettei than its "non-parallel" counterpart [74].



(a) 4-PSK Constellation

(b) Uncoded (One-state trellis) 4-PSK

(c) Redundant 8-PSK Constellation

(d) Four-state trellis for 8-PSK

FIGURE 18. Trellis codes for non-redundant 4-PSK and redundant 8-PSK.

Consider a sequence of bits to transmit as an example: x^1_n, x^2_n = (0,1), (1,0), (0,0). The original state is s^1_n, s^2_n = (0,0) which is in the top left corner of the trellis diagram. The order of the symbols would be (0, 4, 6, 1). The method for traversing the trellis is to use x^1_n to choose the pair of next-state transitions, and x^2_n to select the particular transition from the pair. The net effect of the trellis code is to limit the possible next-state transitions that are allowed to be transmitted. Since the receiver is aware of the code, it can use this information to improve its detection performance.

An important point regarding trellis coding is that the shape of the signal spectrum may be altered. This should be considered when studying trellis-coded signals, especially when simulating them. However in conventional modems, the constraints imposed on the standard trellis cause the spectrum of the signal that is input into the pulse shaping filters to be flat. In other words, the trellis coder is designed so that it does not alter the frequency spectrum of the signal it codes. Thus in standard modems the shape of the modulated signal is dictated by the shape of the filters, and is not affected by the trellis coding.

Trellis coding does not affect the modulator of a transmitter, only the encoder and decoder portions of a modem. Partly for this reason, the use of trellis coding does not affect the signal classification problem provided that the discriminant variables are based only on spectral characteristics of the signal under test. Trellis coding does imply that constellation sizes are large for high bit-rates, which is of some importance.

3.3 Data Communication Standards and Recommendations

The ITU has successfully standardized virtually all data communications via voice-band telephone channels. A few older de facto standards still are in use, such as the Hayes AT command set and the older Bell 103 and Bell 212A modems. Table 4 summarizes the voice-band data communications standards that we judged to be relevant to this thesis [31, 32, 33, 36, 37, 39]. Relevance is based on the extent of use on standard telephone lines as well as leased lines by data modems. (The standards used by modern modems were determined by looking at the manufacturers' specifications for several different brands of commercial modems, and Rockwell modem chip documentation.) Information for the ITU standards and recommendations is taken from the actual ITU specification documents [58, 31, 32, 33, 36, 37, 39].

TABLE 4. Data modem standards and recommendations.

Standard	Duplex (full, half)	Max Bit Rate (bps)	Max Baud Rate (Hz)	Carrier Frequency (Hz)	Modulation Method	Max Constellation Size
Bell 103	full	300	300	1270/1070 2225/2025	FSK	N/A
Bell 212A	full	1200	600	1200/2400	DPSK	4
V.21	full	300	300	1200/2400	FSK	N/A
V.22	full	1200/600	600	1200/2400	DPSK	4
V.22bis	full	2400	600	1200/2400	QAM	16
V.32	full	9600	2400	1800	TC-QAM	32
V.32bis	full	14,400	2400	1800	TC-QAM	128
V.34	full	28,800	3429	1600-1959	TC-QAM	960

3.3.1 Bell 103

Bell 103 is a 300 bps de facto standard that is used in Canada and the U.S. The modulation method is FSK (Frequency Shift Keying) with one bit being transmitted per baud. Two frequencies are used for each transmit direction, one frequency for logical high and the other for logical low. Thus four different frequencies are involved altogether. Bell 103 and Bell 202 were the earliest commercially important modems available, and were released by AT&T in the late 1950s [58]. The Bell 103 protocol is now obsolete, but is still supported by modern modems for compatibility with older equipment. This standard is not used outside North America.

3.3.2 Bell 212A

Bell 212A is a 1200 bps de facto standard that is used in Canada and the U.S. During the 1960s modems were developed that used four-phase modulation, including the Bell 212A that was part of a family of DPSK (Differential Phase Shift Keying) modems

[58]. Transmission is at a baud rate of 600 Hz, with split-band full-duplex transmission. A *split-band modem* is one that uses separate carrier frequencies (shown in table 4) for the caller and call answerer to achieve full-duplex operation. The bands are separated by a spectral null (a guard band with no signal energy). The DPSK constellation contains 4 points, allowing 2 bits to be transmitted per baud. This protocol is rarely used anymore since the lowest grade modems that can now be purchased are 2400 bps.

3.3.3 V.21

V.21 is an international standard that was adopted by the CCITT in 1964, with the last amendment being made in 1984 [31]. The CCITT developed several international modem standards before changing its name to ITU-T in 1993. The V.21 standard is similar to Bell 103 insofar as it uses a 300 bps data rate and FSK modulation. However, since the signaling frequencies used are slightly different, the standards are incompatible. This standard is not used in North America and is now essentially obsolete.

3.3.4 V.22

The ITU's response to Bell 212A was the V.22 standard, but again the two standards are incompatible. It was not ratified until 1980, and was last amended in 1984 [32]. The standard uses a 1200 bps data rate and 4 point DPSK modulation. V.22 is used outside of North America, but is now largely obsolete.

3.3.5 V.22bis

V.22bis is the revised version of the V.22 standard. (The "bis" suffix in V.22bis refers to the Latin word bis that means "twice" or "again". In French the term bis is commonly used to indicate a second appearance or revision.) V.22bis supports 2400 bps communications and was the first modem standard to use true QAM. The standard was ratified by the ITU in 1984 [33]. The baud rate is 600 Hz, with four bits being transmitted per baud. In order to reach this data rate, a 16-point QAM constellation is used. All of the standards mentioned so far are split-band full-duplex, implying that they have a separate channel for each direction of communication. The remaining standards discussed below use a single channel for both send and receive, with echo cancellation used to obtain full duplex operation. Fig. 19 shows the primary 16-point V.22bis constellation.

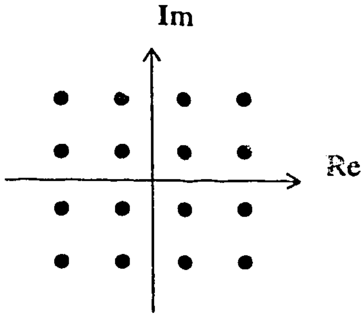
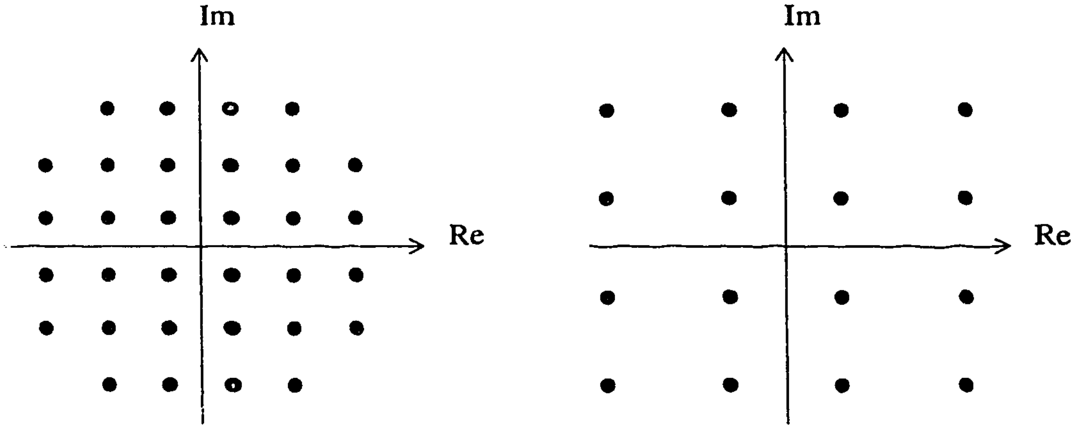


FIGURE 19. V.22bis constellation

3.3.6 V.32

V.32 is a widely used full-duplex, TC-QAM standard that supports 9600 bps communication. This recommendation was not officially approved until mid-1993 [36]; however, it was already in use before becoming fully ratified. The baud rate is 2400 symbols per second (i.e., greater than the carrier frequency), with 4 data bits are transmitted per baud. However, since trellis coding is used, an extra bit is transmitted for each baud. Hence, the constellation has 32 points. Fig. 20(a) shows the primary 32-point V.32 constellation. An additional 16-point constellation, shown in Fig. 20(b), is also available for a non-redundant coding mode. This constellation is provided for modems that do not support trellis coding.

Since V.32 requires full-duplex operation to use the same carrier frequency for both signal directions, echo cancellation must be performed at each end of the call to eliminate (as much as possible) the sender's own signal from the demodulation portion of the modem. This is costly, but the benefit is full-duplex operation, meaning that both directions of the connection can be active at the same time, without the bandwidth overhead of splitband operation. Also, unlike the predecessors to this standard, it more fully uses available channel bandwidth. Note that some references indicate that the constellation for V.32 is a diamond shape. This is a different representation of the same constellation, with a 45 degree phase rotation. This difference can also be noted for other constellations.



(a) 32-point constellation for use with Trellis coding

(b) 16-point constellation for non-redundant coding

FIGURE 20. V.32 signal constellations, 9600 bps.

The reason for the 45-degree phase rotation is explained by Bingham [11]. Since most modems are required to transmit several different constellations as a result of supporting multiple modulation standards, it is desirable that the *M* points of the constellations all lie on the same integer grid. Also, it is desirable that each constellation have approximately the same average signal energy. The solution to this requirement is to rotate constellations with M odd by 45 degrees. An *M odd* constellation is one where the total number of constellation points is an odd power of two. Depending upon your point of view, the 45 degree constellation rotation may or may not be important. Since both constellation representations are geometrically identical, with their inter-symbol distances being of identical proportions, the spectral content of signals using either constellation representation will be identical. However, if demodulation is to be performed, the rotation becomes an important consideration for correct signal detection.

Fall-back options within the V.32 standard define 4800 bps and 2400 bps communications standards. These require smaller constellations which are subsets of the original 32-point constellation. A 16-point QAM constellation identical to that of V.22bis with no trellis coding is used for 9600 bps communications. For 4800 bps a standard 4-point constellation is used. The fall-back options are selected by two communicating modems if a satisfactory 9600 bps connection cannot be maintained.

3.3.7 V.32bis

V.32bis is the mainstream modem protocol in use today, although it is being rapidly replaced by V.34. The bit rate in V.32bis is 14,400 bps, using trellis-coded QAM, 2400 baud symbol rate, and 6 data bits per baud [37]. Full-duplex operation is defined using the same 1800 Hz carrier in both directions. Since trellis coding is used, an extra bit is transmitted per baud, requiring the constellation to have 128 points. The primary V.32bis constellation is shown in Fig. 21. V.32bis uses either a reduced constellation or the V.32 standard as a fall-back mode if the line quality is poor. This standard is very popular due to availability of inexpensive VLSI implementations, relatively fast bit rate, error resistance, and reliability.

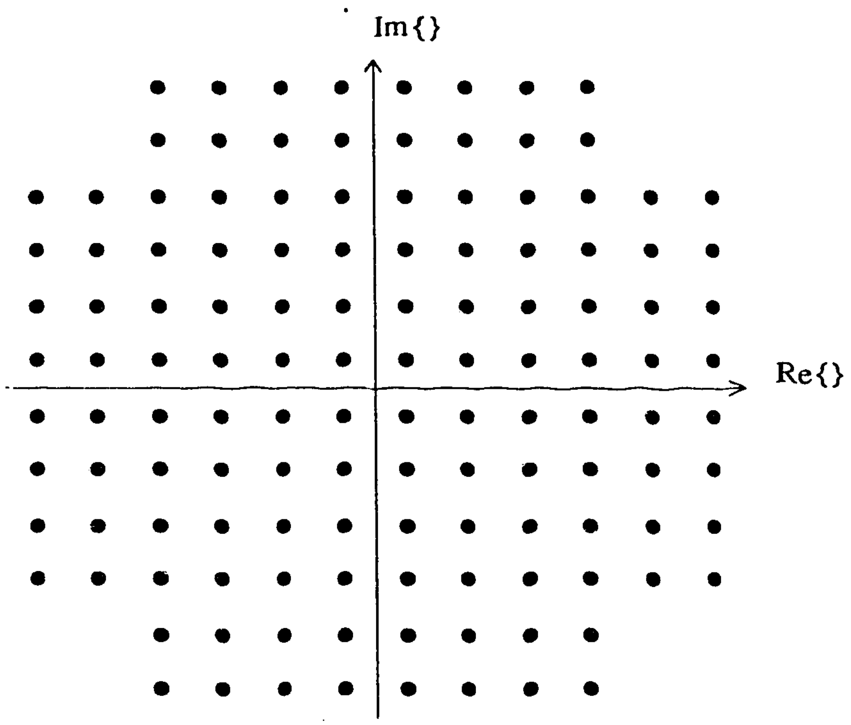


FIGURE 21. V.32bis constellation, 14,400 bps

V.32bis was approved as a recommendation by the ITU in 1991. Included as options and extensions are the following synchronous signaling rates: 12,000 bps trellis-coded QAM; 9600 bps trellis-coded QAM; 7200 bps trellis-coded QAM; 4800 bps unencoded QAM. The 12,000 bps constellation is shown in Fig. 22. The 9600 bps constellation is identical to that of the V.32 32-point constellation. The 7200 bps constellation is the same as that of V.22bis. Finally, the 4800 bps constellation is a simple 4-point QAM constellation that may be considered to be DPSK modulation since the symbol vector amplitudes are constant.

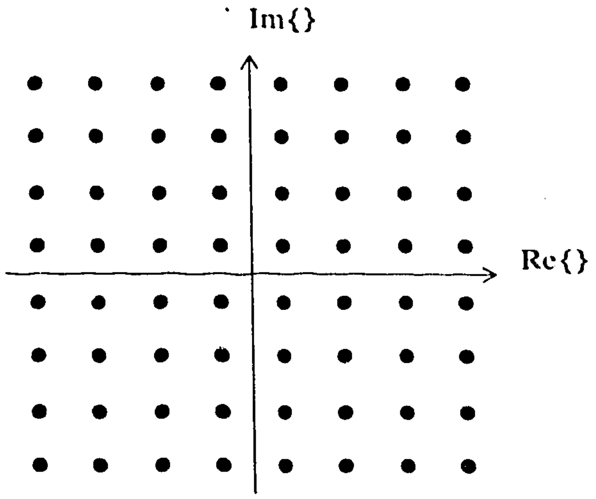


FIGURE 22. V.32bis constellation, 12,000 bps

3.3.8 V.34

The V.34 recommendation has recently been approved by the ITU [39] and is now gaining widespread popularity. This standard allows data communications at rates of up to 28.8 kbps, providing the line conditions are favorable. This standard will likely mark the end of analogue modem development for the telephone network since the theoretical information capacity limits have been reached. (An extension for V.34 is being developed that will allow es of up to 33.6 kbps which requires at least 3400 Hz of bandwidth and an SNR of at least 30 dB. However, it is extremely rare to ever find a public telephone line that will allow this data rate. Some North American users of 33.6 kbps modems have reported successful 33.6 kbps connections in two of every 100 connections. This data is taken from a USENET newsgroup called "comp.sys.modems".) The next step beyond V.34 is likely to be extending purely digital data communications all the way to the subscriber set. Technologies such as N-ISDN will dictate the future means of data communications over purely digital networks from handset (station) to handset (station).

A de facto 28.8 kbps protocol dubbed V.fast is also in use, however it is being phased out due to the ratification of V.34. V.fast is not compatible with V.34, however it is in widespread use due to its early arrival to market. Several features are shared by V.fast and V.34. V.fast provides for a maximum bit-rate of 28,800 bps, has a variable symbol rate up to a maximum of 3429 Hz, and uses a 1024-point QAM constellation with trellis coding.

V.34 is an extremely complex standard and we will only give a short description here. We will emphasize aspects of V.34 that will affect the way we classify these types of signals. The following lists the essential characteristics of V.34 [39]:

- full-duplex or half-duplex modes
- echo cancellation techniques required for channel separation
- synchronous QAM with mandatory selectable symbol rates of 2400, 3000, 3200 symbols/s and optional rates of 2743, 2800, and 3429 symbols/s
- selectable carrier frequency ranging from 1600 to 1959 Hz
- data rates ranging from 2400 bps to 28.8 kbps, in increments of 2400 bps
- trellis coding at all data rates
- optional auxiliary 200 bps signalling channel
- adaptive techniques to allow full use of available bandwidth including pre-emphasis, precoding, line probing, and constellation shaping *
- the V.32bis standard is a supported subset of V.34
- circular superconstellations with up to 960 points also supported

* Pre-emphasis a linear equalization method where the transmitted signal spectrum is shaped to compensate for channel amplitude distortion. Precoding is a non-linear equalization method performed at the transmitter by modifying the coding function in order to compensate amplitude distortion in the channel. Line probing is a method for transmitting periodic signals through a channel to determine the channel characteristics. Constellation shaping is a method for improving noise immunity in a QAM signal by introducing a non-

uniform two-dimensional probability distribution to the transmitter constellation points. Refer to [39] for more detail on these methods.

The features that are of the most importance to the classification problem are the ones that shape the frequency response of the signal being transmitted. In most modes of operation, the V.34 signal will occupy virtually all of the available channel bandwidth. In addition, the spectrum of the transmitted signal will be predistorted if the channel response is not flat at all frequencies in the normal channel bandwidth.

3.4 FAX Communication Standards and Recommendations

Table 5 shows the standards that are used by Group 3 facsimile systems. These standards are specified by the ITU-T T.4 recommendation [40] for Group 3 facsimile apparatus. Another important ITU-T recommendation is T.30 [41], which specifies the procedures for document transmission. The list of standards will likely include a 28,800 bps standard similar to V.34 in the near future. Note that all of the facsimile transmission protocols are half-duplex only. FAX machines by nature are one-way transmission devices and hence to not require full-duplex operation for the main portion of a connection.

TABLE 5. FAX transmission standards and recommendations

Standard	Duplex (full, half)	Bit Rate (bps)	Baud Rate (Hz)	Carrier Frequency (Hz)	Modulation Method	Constellation Size
V.17	half	14,400	2400	1800	TC-QAM	128
V.27ter	half	4800/2400	1600/1200	1800	DPSK	16/4
V.29	half	9600/7200	2400	1700	QAM	16/8
V.?	half	28,800	?	?	?	?

It is important to note that these are only the data modulation standards for facsimile transmission. Separate standards are used to define signalling. In standard T.30 [41] the signaling is specified to be V.21 channel 2. In other words, the signaling is to be binary-coded FSK at 300 bps. This signaling standard is used to transmit critical information regarding the type of document transfer that is to take place.

3.4.1 V.17

The V.17 recommendation was adopted by the ITU in 1991 [38] at the same time as V.32bis. This standard is effectively a half-duplex version of V.32bis. The same constellations, carrier frequency, and symbol rate as those of V.32bis are used. This makes it very difficult to distinguish between a V.17 signal and either a V.32 or V.32bis signal. The only way these two types of signals can be separated is by observing the handshaking sequences that precede and follow the data transfer portion of a call, or by identifying whether a call is full or half-duplex. At present it does not appear that V.17 is in wide-spread use although this situation should change in the coming years.

3.4.2 V.27ter

In 1976 the CCITT approved recommendation V.27ter [34]; the most recent amendment was made in 1984. The modulation method used for V.27ter is DPSK with 16- or 4-point constellations. The constellations are simply 4 or 16 points distributed on a circle, where the smaller 4-point constellation is used for a fall-back mode on lossy or noisy lines. This transmission protocol was the first widely used Group 3 facsimile transmission protocol and, consequently, there is a large number of FAX machines that use it. The 16-point constellation is needed for transmitting 4 bits per baud to produce in the 4800 bps data rate.

3.4.3 V.29

In 1976 recommendation V.29 [35] was also approved; the most recent amendment to V.29 was made in 1984. The V.29 protocol calls for 9600 bps communication using the 16-point QAM constellation is shown in Fig. 23. However, as can be seen in Fig. 23, the constellation is not conventional orthogonal-coordinate QAM. It may be more accurately called PSK with AM. A fall-back mode of 7200 bps is provided, using the 8-point constellation shown in Fig. 24. A final fall-back mode of 4800 bps uses a 4-point constellation (not shown).

Another interesting point with this standard is the use of a 1700 Hz carrier frequency. This carrier is unusual, and can be exploited to distinguish V.29 transmissions from transmission using the usual 1800 Hz carrier found in most other high-speed modulation protocols.

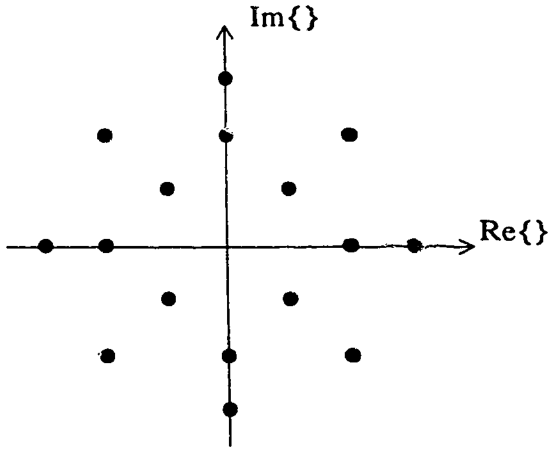


FIGURE 23. V.29 constellation, 16 points, 9600 bps.

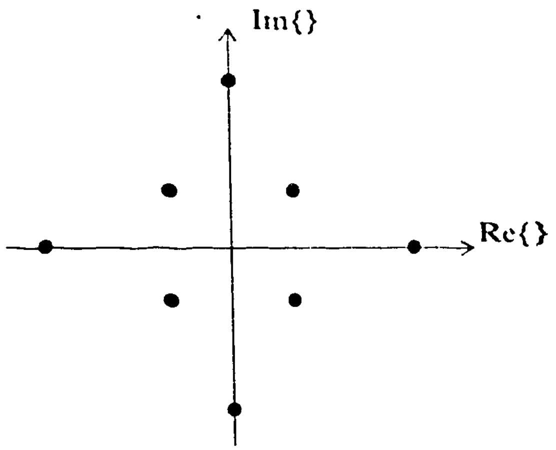


FIGURE 24. V.29 constellation, 8 points, 7200 bps

3.5 Discriminant Analysis

3.5.1 What is Discriminant Analysis?

Lachenbruch writes that discriminant analysis is "the basic problem... [of assigning] an observation *x* of unknown origin to one of two (or more) distinct groups on the basis of the value of the observation" [48]. The observation *x* can also be called a *feature* or *discriminant variable*. For example, if we are observing a time-varying voltage signal *v(t)*, then the observation of *v(t')* at a fixed time *t'* is a feature. If the signal was binary and *v(t)* could assume one of two possible levels {+5 V, -5 V} corresponding to logical one or zero, then we would have one feature variable *v* and two groups {logical one, logical zero}. These groups may also be called *classes*. In this example, simple voltage measurement and comparison of the feature variable with a mid-point threshold could determine the class membership of the observation.

In the example above it is easy to see how observations of a signal could be easily classified on the basis of a single discriminant variable in the best case. Very often, however, we cannot utilize such a simple rule for classification or allocation of an observation to a class. For example, class membership may not be reliably given by the value of any single available variable. Thus a set of feature variables could form distributions that can either be expressible analytically or only available as empirical data. An additional complication arises when the available feature variables are not sufficient for correct classification. Consider the scatter plots in Fig. 25. Figure 25 (a) represents a case where two feature variables are required for accurate classification of two classes (denoted by '+' and 'o'), whereas in Fig. 25 (b) only one feature variable is needed for classification.

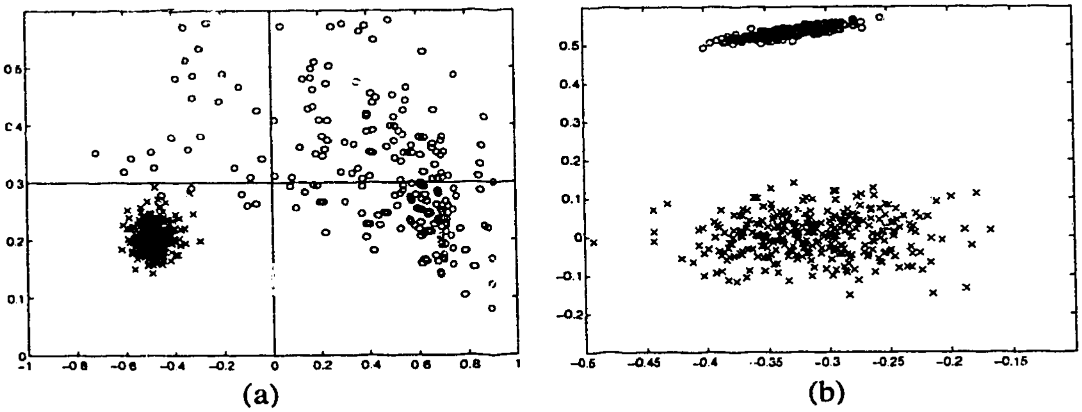


FIGURE 25. Scatter plots of a two discriminant variables.

For a brief overview of discriminant analysis, refer to P.A. Lachenbruch's book [48] by the same title. For an in-depth and modern treatment of the subject, refer to G.J. McLachlan's book [54] "Discriminant Analysis and Statistical Pattern Recognition."

3.5.2 Discriminant Variables and Selection

When performing discriminant analysis, there are often many features that could be used as discriminant variables; however, including all possible features is usually impractical or infeasible. The task of selecting a best subset of feature variables and estimating their "potency" as discriminants is the subject of M. Ben-Bassat's paper [4]. Other discussions of the topic can be found in many discriminant analysis reference books including [48] and [54]. The topic of feature selection and analysis is relatively mature, hence there are many software tools for rapidly implementing the most popular algorithms. For this reason we only give a brief discussion of the field here.

The most basic method of feature selection is to grade the effectiveness of each feature according to its expected probability of misclassification or probability of error. If the cost of misclassification is equal and nonzero for all classes, while the cost of correct classification is zero, then the risk for each feature is equivalent to its probability of error. So, if we only want to minimize the overall error rate of a classifier, then it makes sense to use the probability of error as our criteria for feature selection. Why then has so much research been done on finding alternative methods for feature selection? Well, Ben-Bassat [4] gives four reasons for not using the probability of error (P_e) rule:

1. Features that are very close in performance may not be differentiated effectively by the P_e rule.
2. The P_e rule allows selection of individual features based in their individual expected performances, however this does not guarantee good subset selection.
3. When performing sequential classification, where each feature is consulted on its own, the P_e rule is not sufficient. This is because the criterion for feature selection is often to achieve a certain probability of error with a minimal number of features. What Ben-Bassat calls "myopic" procedures are usually used which do not lend themselves well to the P_e rule.
4. The computational complexity of computing P_e is too large in many cases where there is a large number of classes, variables, and cases.

Unfortunately, there is no optimal or ideal alternative to the P_e rule [4]. The rule that is chosen is based typically on criteria that the classification system designer deems to be important.

We now mention a few methods for evaluating features. There are three general types of measures: information measures, distance measures, and dependence measures. Interestingly, most of the measures studied by [4] were shown to yield very similar feature variable rankings other than in exceptional circumstances. Thus one can reasonably conclude that the computational complexity of the evaluation rule should be a major factor in selecting feature evaluation methods (as well as the availability of software to perform the analysis).

3.5.2.1 Information Measures for Feature Evaluation

By measuring the amount of information gained by using a given feature, we have an indicator of how valuable that feature is to a classifier. For example, Shannon's entropy could be used to determine how much information is contained in a feature variable. Note that this is not the same as the entropy of the signal being monitored. Uncertainty is another name for information. In other words, the more information a feature yields, the higher its uncertainty. This type of measure leads to a rule that states that feature X is to be preferred over feature Y if the uncertainty function concerning the true class of X is less than that of Y. Information measures include Shannon, Quadratic, Daroczy, f-entropy, f strictly concave, and Renyi [4]. Renyi's measure is actually a generalization of all entropy measures, and can be tuned to the particular application.

3.5.2.2 Distance Measures for Feature Evaluation

Distance measures are used to determine how well the class distributions are separated by a given feature. These measures lead to rules that state that feature X is preferred over feature Y if the class probability distributions are "farther apart" for feature X than Y. A simple example is to use the differences in mean values of the distributions as a measure of group separation. This is an oversimplification, but it illustrates the idea behind distance measures. Distance measures include Bhattacharyya, Matusita, Kullback-Liebler, Kolmogorov, and Lissack-Fu [4].

Five distance measures are used by SPSS for feature evaluation [73]. These include Wilk's lambda (Λ), unexplained variance (R), Mahalanobis distance (${D^2}_{ab}$), smallest F ratio (F_{ab}), and Rao's V (V). The equations forming the basis of each of these methods are given by (EQ 10, 11, 12, 13, 14). Note that the following notations are used:

- g number of groups
- q number of feature variables used for analysis
- X_{ijk} the value of feature variable i, case k, in group j
- $\bar{X}_{ij}$ the mean value of feature variable i in group j
- $\bar{X}_j$ the mean value vector for all variables in group j
- n_j the sum of case weights in group j (case weights are all equal)
- n the total sum of weights
- w_{il} within-groups sums of squares (an element of the inverse within-groups covariance matrix
- W crossproduct matrix of the within-groups sums of squares
- t_{il} total sums of squares
- T crossproduct matrix of the total sums of squares
- C pooled within-groups covariance matrix

$$\Lambda = \frac{|W_{11}|}{|T_{11}|} \qquad \text{(EQ 10)}$$

$$D_{ab}^{2} = -(n-g) \sum_{(i=1)}^{q} \sum_{(l=1)}^{q} w_{il} (\bar{X}_{ia} - \bar{X}_{ib}) (\bar{X}_{la} - \bar{X}_{lb}) \qquad \text{(EQ 11)}$$

$$= (\bar{X}_a - \bar{X}_b)' C^{-1} (\bar{X}_a - \bar{X}_b)$$

$$F_{ab} = \frac{(n-q-g+1) n_a n_b}{q (n-q) (n_a - n_b)} D_{ab}^{2} \qquad \text{(EQ 12)}$$

$$R = \sum_{(a=1)}^{s-1} \sum_{(b=a+1)}^{s} \frac{4}{\left(4 + D_{ab}^{2}\right)} \qquad \text{(EQ 13)}$$

$$V = -(n-g) \sum_{(i=1)}^{q} \sum_{(l=1)}^{q} w_{il} (t_{il} - t_{il}) \qquad \text{(EQ 14)}$$

3.5.2.3 Dependence Measures for Feature Evaluation

Dependence measures yield feature evaluation rules that are based on the dependence between the random variable representing the true class, and the feature variable. Thus, if the dependence is high, the feature is deemed to be more valuable. An example of a dependence measure is the correlation coefficient of two random variables. The more highly correlated are the variables, the higher is their dependence [4].

3.5.3 Discriminant Functions

R.H. Shumway's paper [69] "Discriminant Analysis for Time Series" is considered by many to be the benchmark study of linear and quadratic discriminant functions. It was used as the primary reference to this section.

Suppose we have a discrete time observation vector $\{x(t), t = 0, 1, ..., T\text{-}1\}$, that consists of T observations and represents either a time series discriminant variable, or an array of discriminant variables. Now suppose that there are q possible classes that an observation may belong to, denoted by $E_1, E_2, \ldots, E_q$. In other words, a T-dimensional space is partitioned into q disjoint regions $E_1, E_2, \ldots, E_q$. The mathematical rule used to assign or allocate an observation $\boldsymbol{x}$ to population i is called the *discriminant function*, and it is formed by one or more discriminant variables.

The true *prior probability* that $\boldsymbol{x}$ belongs to class i will be denoted by π_i. If $\boldsymbol{x}$ has a probability density function $p_i(\boldsymbol{x})$ when $\boldsymbol{x}$ belongs to class i, then the probability $P(j \mid i)$ of misclassifying an observation from class i into class j is given by equation (15), where i does not equal j. Consider an example where we wish to determine the probability of misclassifying an observation from class 1 into class 2 as shown in Fig. 26, where x is one-dimensional. This misclassification probability will be the area occupied by the tail of $p_1(x)$ in region E_2, as indicated by equation (16).

$$P(j|i) = \int_{E_j} p_i(\boldsymbol{x})\, dx \qquad \text{(EQ 15)}$$

$$P(2|1) = \int_{E_2} p_1(x)\, dx = \int_{x_0}^{\infty} p_1(x)\, dx \qquad \text{(EQ 16)}$$

The overall *probability of error*, P_e, considering the prior probabilities for membership in each class i, is given by equation (17).

$$P_e = \sum_{i=1}^{q} \pi_i \sum_{j \neq i} P(j|i) \qquad \text{(EQ 17)}$$

If our cost function for misclassification is equal to the probability of error, then the optimal method for minimizing the cost is *Bayes' Rule*. In this rule, x is allocated to class l if equation (18) holds for all j not equal to l. If this relation is not true, the resulting classification is not optimal.

$$\frac{p_l(x)}{p_j(x)} > \frac{\pi_j}{\pi_l} \qquad \text{(EQ 18)}$$

Bayes' Rule follows from *Bayes' Theorem*, which gives the equation for the a *posterior probability* of an observation x being from class i given the *a priori* probabilities of group membership, and the probability distribution of x for the various classes. If the a priori probabilities are not known, they can be assumed to be equal for each group at the expense of losing the optimality properties of the rule. The a priori probabilities can also be used to weight a cost function for misclassification of each class. Bayes' Rule is an implementation of a linear discriminant function, which will be discussed in the next section.

The sketch in Fig. 26 represents the probability density functions of one discriminant variable that is used to separate two groups. Since the groups have overlapping probability densities, no classifier based on mean differences will be able to perfectly classify all observations into their correct class. By using Bayes' Theorem we can determine the probability that an observation belongs to either of the two classes. The highest group membership probability is thus used to allocate the observation into a class. In Fig. 26 two sample observation values for x are shown, and indicated by x_1 and x_2. If the correct group membership of these two observations are group one and group two respectively, by using Bayes' Rule we will correctly classify x_1, but not x_2.

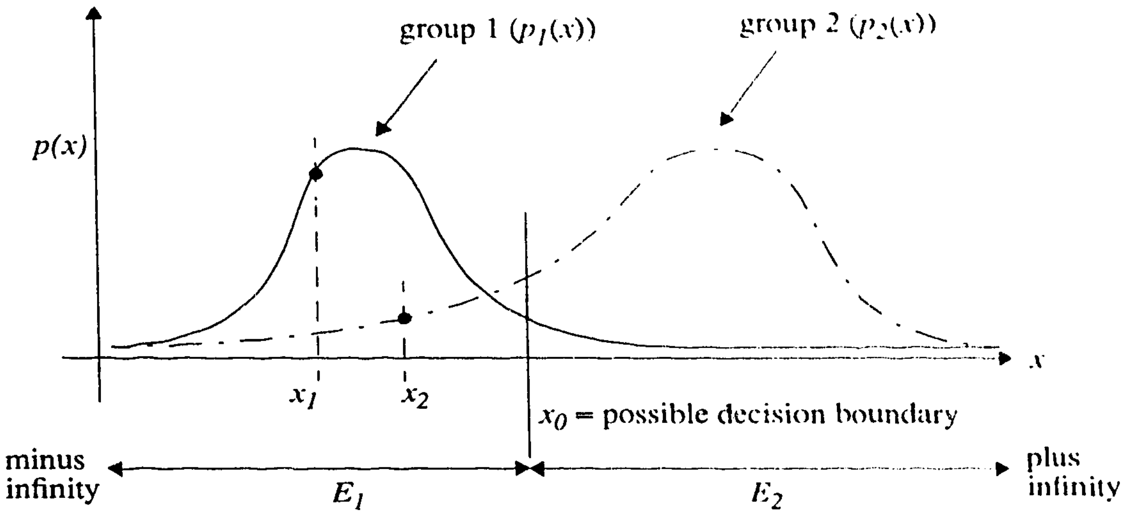


FIGURE 26. Probability densities for two groups and one discriminant variable.

3.5.3.1 Linear Discriminant Functions

To use a linear discriminant function optimally, we must assume (1) that x has unequal mean vectors for each class, (2) that the covariance matrices are equal, and (3) that x is normally distributed. A *linear discriminant* function is simply a linear combination of the independent discriminant variables. Critical to finding a good linear discriminant function is choosing the best possible coefficients of the linear equation. The mean vector of x is defined by equation (20), and the T x T covariance matrix R_j is given by equation (21), where x is the multivariate normal vector in equation (19). The covariance matrix R_j contains elements $r_j(t\text{-}u)$ which each correspond to the covariance between $x(t)$ and $x(u)$ for class j. In this thesis all discriminant variables are single valued, so $x(t)$ is one variable, and $x(u)$ is another. In equation (22) we show the covariance equation for $r_j(t\text{-}u)$, where E is the expectation operator. Note that the prime mark in equation (20) indicates the transpose operation. If T is equal to one, then x is a single value, μ_j is the mean of x for class j, and R_j is the variance of x for class j.

$$x = (x(0), x(1), \ldots, x(T-1)) \qquad \text{(EQ 19)}$$

$$\mu_j = (\mu_j(0), \mu_j(1), \ldots, \mu_j(T-1))' \qquad \text{(EQ 20)}$$

$$R_j = \{r_j(t-u), t, u = 0, 1, \ldots, T-1\} \qquad \text{(EQ 21)}$$

$$r_j(t-u) = Cov(x(t), x(u)) = E[(x(t) - \mu_j(t))(x(u) - \mu_j(u))] \qquad \text{(EQ 22)}$$

Note that in this discussion we allow x to be a vector. This convention permits x to represent a single time series variable, or a series of separate discriminant variables. To consider a set of time-series discriminant variables we require additional mathematics.

From these equations, the linear discriminant function for the two-class case (q = 2) is defined in equation (23). Note that the covariance matrix R is not subscripted since we assume that R_1=R_2. Here the rule is to accept the hypothesis 1 (i.e. the observation belongs to class 1) if equation (23) exceeds a threshold value K. The threshold value K is selected to yield the desired type I and type II errors, which are $P(1|2)$ and $P(2|1)$ respectively.

$$d_L(x) = (\mu_1 - \mu_2)'R^{-1}x - \frac{1}{2}\mu_1'R^{-1}\mu_1 + \frac{1}{2}\mu_2'R^{-1}\mu_2 \qquad \text{(EQ 23)}$$

The linear discriminant function shown in equation (23) appears to be very complex to compute. However, the function can simply be interpreted as a sum of discriminant variables multiplied by coefficients, added to a constant value. Since x is a vector, it may be used to represent a set of discriminant variables. The only complex part of the function is the initial computation of the coefficients and constant value. Once these values are determined, computing $d_L(x)$ for a particular observation vector is straightforward. In order to calculate the coefficients the discriminant variable mean values and covariance matrices must be known for each class.

To evaluate the value of a linear discriminant function given an observation vector of discriminant variables, a multiplication is performed for each variable, and an addition is performed for all variables plus a constant. Therefore, the number of operations (where an addition or a multiplication is considered one operation) is equal to $(T+(T-1)+1) = 2T$. To calculate a linear discriminant function, twice as many operations are required as there are discriminant variables.

When we have more than 2 classes, Shumway gives an intermediate form equation (24) and combines it with Bayes' Rule equation (18) to obtain an overall classification rule equation (25). Note that we will have one intermediate form equation for every class.

$$g_i(x) = \mu_j' R^{-1} x - \frac{1}{2}\mu_j' R^{-1} \mu_j \qquad \text{(EQ 24)}$$

$$u_{lj}(x) = (g_l(x) - g_j(x)) > (\ln(\pi_j) - \ln(\pi_l)) \qquad \text{(EQ 25)}$$

This rule states that if the linear discriminant function equation (25) holds true for $j = 1, 2, \ldots, q$ and $j \neq l$, then we should allocate observation x into class l. For the assumptions made at the start of this section, this rule can be proven optimal with respect to misclassification error rates. Shumway continues to give an analysis of the multivariate time series discriminant variable with multiple classes linear discriminant functions. We will not repeat that discussion here since those types of discriminant variables were not used in this thesis.

Another method for handling the multiple-class problem is based on Fisher's original method of constructing linear discriminant functions [48]. This is also the method of choice for SPSS, and will be discussed following the definition of Fisher's linear discriminant functions.

The formulae given above are somewhat complicated. We can give an equation for a linear discriminant function that has the simple form of equation (26). Here the complexity of the equation is hidden by the coefficients ($B_1, B_2, \ldots, B_q$) and the constant B_0. The discriminant variables are represented by ($X_1, X_2, \ldots, X_q$). The formulae for computing the coefficients (shown in equation equation (23)) are designed to maximize the difference in the value of the discriminant function for different groups of observations.

$$D_L = B_0 + B_1X_1 + B_2X_2 + \ldots + B_qX_q \qquad \text{(EQ 26)}$$

An alternative form of the optimal linear discriminant functions are Fisher's linear discriminant functions [48]. These functions are designed to maximize the ratio of the difference of the means of the linear combination in the groups to the variance. One function is defined as a linear combination of all discriminant variables for each class. Thus there are as many functions as there are classes. An observation is simply allocated into the class which has the largest Fisher function value. This method results in the same classification accuracy as the optimal linear discriminant functions combined with Bayes' theorem, however the posterior probability of group membership is assumed to be unknown. Equations (27), (28), and (29) give the equations for Fisher's functions, the coefficients, and the constant terms, respectively. Note that the *W* and *X* terms were defined earlier in this chapter as the inverse of the covariance matrix and the mean values, respectively. In this case the number of variables i ranges from 1 to q, and the number of classes j ranges from 1 to g. As the equations indicate, these functions contain coefficients that are easy to evaluate and straightforward to implement even where multiple groups are considered.

$$D_j = b_{1j}X_1 + b_{2j}X_2 + \dots + b_{qj}X_q + a_j \qquad \text{(EQ 27)}$$

$$b_{ij} = (n-g)\sum_{l=1}^{q} w_{il}\bar{x}_{li} \qquad \text{(EQ 28)}$$

$$a_j = \log p_j - \frac{1}{2}\sum_{i=1}^{q} b_{ij}\bar{x}_{ij} \qquad \text{(EQ 29)}$$

We now turn our attention to the canonical vector approach to dealing the with the multiple-class problem based on Fisher's original method. This discussion follows that given in [71] and [48]. The method is based on canonical variates developed from the between-group and within-group covariance matrices which are the combined covariance matrix for all groups, $\boldsymbol{B}$, and the covariance matrix within each group, $\boldsymbol{W}$, respectively. The formulae used for estimating these matrices from discriminant variable data are given in equations (30) and (31). In these equations $\boldsymbol{x}$ is the vector of discriminant variables, g is the number of classes or groups, and n_i is the number of discriminant variable cases for each class.

$$\boldsymbol{B} = \frac{1}{g}\sum_{i=1}^{g} (\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})' \qquad \text{(EQ 30)}$$

$$\boldsymbol{W} = \frac{1}{\sum n_i - g}\sum_{(i=1)}^{g}\sum_{(j=1)}^{n_i} (\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})(\bar{\boldsymbol{x}}_i - \bar{\boldsymbol{x}})' \qquad \text{(EQ 31)}$$

The average value vector for the discriminant variables over all classes is given by equation (32).

$$\bar{x} = \frac{1}{\sum n_i} \cdot \sum_{(i=1)}^{g} \sum_{(j=1)}^{n_i} x_{ij} \qquad \text{(EQ 32)}$$

Fisher suggested that to find a set of discriminant functions, we determine the compound λ that maximizes γ as shown in equation (33). Note that all of the variables in this equation denote matrices.

$$\gamma = \frac{\lambda' \boldsymbol{B} \lambda}{\lambda' \boldsymbol{W} \lambda} \qquad \text{(EQ 33)}$$

By differentiating equation equation (33) with respect to λ and solving for the roots we can determine the properties required by λ to satisfy Fisher's suggestion. As it turns out, the solutions are the eigenvalues of $\boldsymbol{W}^{1}\boldsymbol{B}$. Also, there can be no more than min(g-1, q) nonzero solutions. This is an important point to consider when there are fewer discriminant variables than classes. The resulting eigenvectors are used as the coefficients for the linear discriminant functions. The classification rule is now defined as: Allocate an observation to class Π_i if equation (34) is true, where y and v are defined by equations (35) and (36). This rule is only optimal if all eigenvectors are used for forming the discriminant functions, and there are more variables than classes.

$$\left(y - \frac{1}{2}v_i\right)' v_i = max\left[j, \left(y - \frac{1}{2}v_j\right)' v_j\right] \qquad \text{(EQ 34)}$$

$$y = \lambda' x \qquad \text{(EQ 35)}$$

$$v = \lambda' \bar{x}_i \qquad \text{(EQ 36)}$$

The coefficients of the linear discriminant functions for the multiple-class problem can be calculated using the equations listed above. Once the coefficients are determined, the functions can be applied to the discriminant variable data to form the probability density functions of the discriminant functions so that Bayes' theorem can be used for classification. This canonical vector ı..ethod is used by SPSS for classification.

3.5.3.2 Bayes' Theorem

We have mentioned Bayes' Theorem a few times already, and so it deserves a brief description. S. Geisser [21] considered Bayes' theorem and its applications to Discrimination Theory. We will use this paper as a main reference on Bayes' theorem, however most statistics reference books will include a discussion of the theorem. Another useful discussion of Bayes' theorem is given by M. Ben-B^ssat [4].

The central idea of Bayes' theorem is that if we know the prior probabilities of each class and the distributions of the discriminant variables for each class, then we can determine the a posteriori probability that an observation x belongs to class i.

Let the number of classes be given by q. The prior probability vector for all classes can be represented by $\Pi = (\pi_1, \pi_2, \ldots, \pi_q)$. If we consider the multivariate case, then the discriminant variable (feature) is a multidimensional vector $\boldsymbol{x}$. A particular case of a feature is denoted by x_j. Let X_j denote the feature vector j. Also, let $P_i(x_j)$ denote the conditional probability density function for feature j being from class i at the value x_j. From these definitions, we give Bayes' theorem in equation (37), which is the equation for the a posteriori probability of case x_j being from class i.

$$\hat{\pi}_i(x_j) = \frac{\pi_i P_i(x_j)}{\sum_{k=1}^{q} \pi_k P_k(x_j)} \qquad \text{(EQ 37)}$$

From this equation, we can compute the a posteriori probabilities that a particular observation belongs to each of the classes. Then we allocate the observation to the class with the highest a posteriori probability of membership. This yields a simple and effective method for performing classification. Optionally, the prior probabilities can be weighted to place more emphasis on correctly classifying certain groups of observations. For a complete discussion on how SPSS utilizes Bayes' theorem, please refer to [71].

3.5.3.3 Quadratic Discriminant Functions

If we assume that the mean values of the PDFs of the discriminant variables are equal but the within-groups covariance matrices are not, then we cannot use the linear discriminant function as given before (or we must realize that it will not be optimal since the assumptions are violated). If the covariance matrices are different from class to class, the discriminant variables have different covariances for each class. In this case Shumway [69] suggests the use of an approximated *quadratic discriminant function*. In the course of this thesis work, however, true quadratic discriminants were not implemented. Instead a variation of the quadratic discriminant function was used in this thesis. We name this variation to be "pseudo-quadratic". This function is the result of a simplification of the optimal quadratic discriminant function that is defined by Shumway [69], and is used extensively by the SPSS software [55]. We will give the basic formulation of the optimal quadratic discriminant function, and explain in detail the pseudo-quadratic form used in SPSS.

Consider the two-group case where both the mean values and the covariance matrices of the discriminant variables differ. Shumway defines the resulting discriminant function as the sum of a linear and a quadratic function as shown in equation (38), and whose elements are the same as for the linear discriminant function discussed earlier. Evaluation of this discriminant function would require $[(2T^2-T)+(T+(T-1))+(T+(T-1))]=(2T^2+3T-2)$ operations. Compare this complexity with that of a linear discriminant function requiring only $2T$ operations. The quadratic discriminant function is said to be order T^2 and the linear discriminant function is said to be order T, where T is the number of discriminant variables in the system.

$$d'_Q(x) = \frac{1}{2}x'\left(R_2^{-1} - R_1^{-1}\right)x + \left(\mu'_1 R_1^{-1} - \mu'_2 R_2^{-1}\right)x \qquad \text{(EQ 38)}$$

A special case occurs when the mean values are all equal to zero, or in other words, the signal being monitored is *zero-mean stationary stochastic*. The discriminant function may now be expressed in a purely quadratic form equation (39). The multi-group and multivariate forms will not be discussed here.

$$d_Q(x) = x'\left(R_2^{-1} - R_1^{-1}\right)x \qquad \text{(EQ 39)}$$

Now we consider the approximation of the optimal quadratic discriminant function that is used by SPSS [71], called a *pseudo-quadratic* approximation. The difference between the pseudo-quadratic discriminant function and the optimal quadratic discriminant function is that classification is based on the discriminant functions, not on the original variables. The covariance matrices used in the pseudo-quadratic function are formed from the canonical discriminant functions for each group, not the original variables. Consider equation (39): Here the quadratic discriminant function is formed from the feature variable $\boldsymbol{x}$, and the covariance matrices $\boldsymbol{R}$ of the two groups. In the pseudo-quadratic form, the $\boldsymbol{R}$ matrices are replaced by the covariance matrices of the canonical linear discriminant function(s). The standard canonical discriminant function coefficient matrix is formed by solving a general eigenvalue problem from the unscaled discriminant function coefficient matrix [71].

All of this results in a classifier that has quadratic properties, but is not purely quadratic. What is effectively achieved is a warping of the feature space to account for the correlations between the linear discriminant functions. A purely quadratic form would utilize the correlations between feature variables to achieve an optimal discriminator. Refer to the *SPSS Statistical Algorithms* manual for a detailed mathematical explanation of the process [71].

When results are obtained using the pseudo-quadratic discriminant function, we can safely conclude that the results would either be identical or improved by using the optimal quadratic discriminant function. Therefore, the results that we will present later can be considered to represent a lower bound on the classification performance achievable if quadratic discriminant functions are used.

3.5.4 Discrimination Example

We will now describe an example to illustrate how the concepts discussed earlier are applied to a real discrimination problem. The data we will use for this example is from 16,3688 cases involving both actual recorded signals and some simulated signals. Two discriminant variables are included, denoted by *V1* and *V2*, that are to be used to separate the signals into two classes. Actual class membership is known a priori for all cases. The functions used to compute the discriminant variables is not important here, nor do we need to know what the classes represent. In this example we will show how to evaluate the discriminant variables using the Mahalanobis distance measure, generate a linear discriminant function, and perform classification.

Fig. 27 shows the contour map of the three-dimensional histogram representing the probability densities of the two discriminant variables. One axis on the horizontal plane is for *V1* and the other axis is for *V2*, however the axes show a modified scale. The height of the histogram represents the probability density of the two variables. The lower left cluster corresponds to one class, and the upper right cluster corresponds to the other class. From the contour map we can see that there is clearly a mean difference between the distributions of the two classes for each discriminant variable. The map even reveals that there are several other classes of signals that could be discriminated.

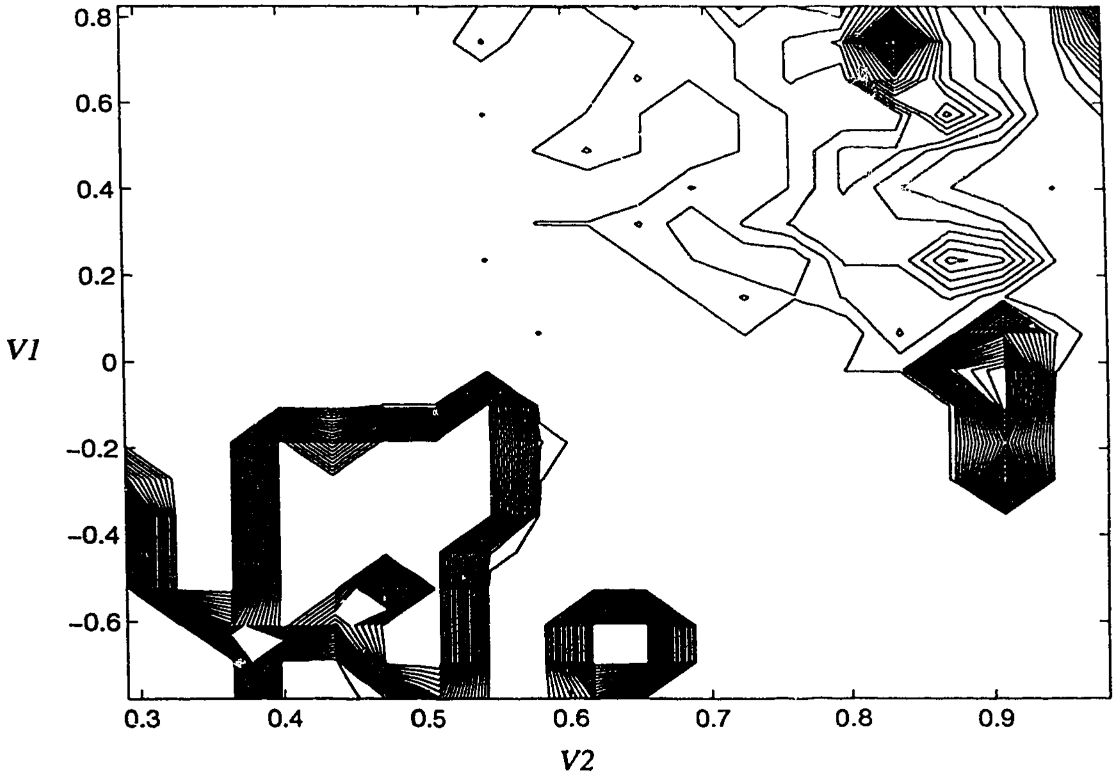


FIGURE 27. Contour map of three dimensional probability density of *V1* and *V2*.

3.5.4.1 Discriminant Variable Evaluation Using Mahalanobis Distance

To determine the relative worth of each discriminant variable we will compute the Mahalanobis distance for both variables, between the only two groups. Recall that the equation for the Mahalanobis distance is as given by equation (40) for groups *a* and *b*.

$$D^2_{ab} = -(n-g) \sum_{(i=1)}^{q} \sum_{(l=1)}^{q} w_{il} (\bar{X}_{ia} - \bar{X}_{ib}) (\bar{X}_{la} - \bar{X}_{lb}) \qquad \text{(EQ 40)}$$

$$= (\bar{X}_a - \bar{X}_b)' C^{-1} (\bar{X}_a - \bar{X}_b)$$

In this case, the number of groups *g* is two, the number of variables *q* is two, and the total sum of weights *n* is one. Four mean values must be computed as well as the within-groups sums of squares (or covariance matrix). Since there are over 16,000 cases, we will not show the computations for the mean values nor the within groups sums of squares (or covariance matrix). The mean values is given in Table 6 and the pooled within-groups covariance matrix is given in Table 7. Since the within-groups sums of squares are elements of the inverse of the pooled within-groups covariance matrix, the covariance matrix is included here. Note that we must assume that the covariance matrices for each variable are the same, or use the pooled within-groups covariance matrix as we have done in this example. The Mahalanobis distance for each variable is now computed in equations (41) and (42), and shown in Table 8.

$$D^2{}_{V1} = (0.4281 - 0.8849)\left(\frac{1}{0.006389}\right)(0.4281 - 0.8849) = 32.66 \qquad \text{(EQ 41)}$$

$$D^2{}_{V2} = (-0.4511 - 0.4599)\left(\frac{1}{0.0434}\right)(-0.4511 - 0.4599) = 19.12 \qquad \text{(EQ 42)}$$

TABLE 6. Mean values for each variable and group.

Mean Value	*V1*	*V2*
Group 1	0.4281	-0.4511
Group 2	0.8849	0.4599

TABLE 7. Within-groups covariance matrix.

Covariance	*V1 (l=1)*	*V2 (l=2)*
V1 (i=1)	6.389E-03	-
V2 (i=2)	3.426E-03	0.0434

TABLE 8. Mahalanobis distances.

D Squared	
V1	32.66
V2	19.12

From the resulting Mahalanobis distance computations we can conclude that *V1* separates the two group's mean values better than *V2*. If we could only select one discriminant variable to be used by the classifier, we should choose *V1* if the Mahalanobis distance is used as the evaluation measure.

3.5.4.2 Generation of a Linear Discriminant Function

Now we can compute the coefficients of a linear discriminant function. Since there are only two groups and two variables we only require one discriminant function. Recall the equation of a linear discriminant function as given by equation (23). This representation is difficult to manage, so we will revert to the simplified version shown in equation (26). The function coefficients for the simple form are given in Table 9. The computations are not shown for brevity. Finally, the linear discriminant function is shown in equation (43).

TABLE 9. Unstandardized canonical linear discriminant function coefficients.

Coefficients	
V1	9.56
V2	2.43
(Constant)	-3.32

$$D_L = -3.32 + 9.56\,(V1) + 2.43\,(V2) \qquad \text{(EQ 43)}$$

3.5.4.3 Classification

Once the discriminant function has been determined, we can proceed to test it on our data by computing the function value for each observation case and performing a classification. To perform the classification, Bayes' Rule will be used. Bayes' Rule requires the a priori probabilities of each group to be known. We will assume that each group has an equal a priori probability. The probability density function of the variable D_L needs to be known for each group. This is easily computed but we don't show the calculations here since a computer is required to perform the computations. Instead, we will show the PDFs for each variable and class graphically in figures 28, 29, 30, and 31 as histograms which represent the probability mass functions. The result of Bayes' Rule is the a posteriori probability that an observation belongs to a particular group. In Table 10 we show the results for a few test observations. By substituting the values for *V1* and *V2* into equation (43) one can verify the discriminant function values shown. Note that the last line of the table lists a case where the class 2 observation was misclassified as class 1. This was the only observation out of 16,368 observations that was erroneously classified. The resulting classification system accuracy is over 99.99%. Equation (44) gives a sample calculation for the a posteriori probability that an observation belongs to class one.

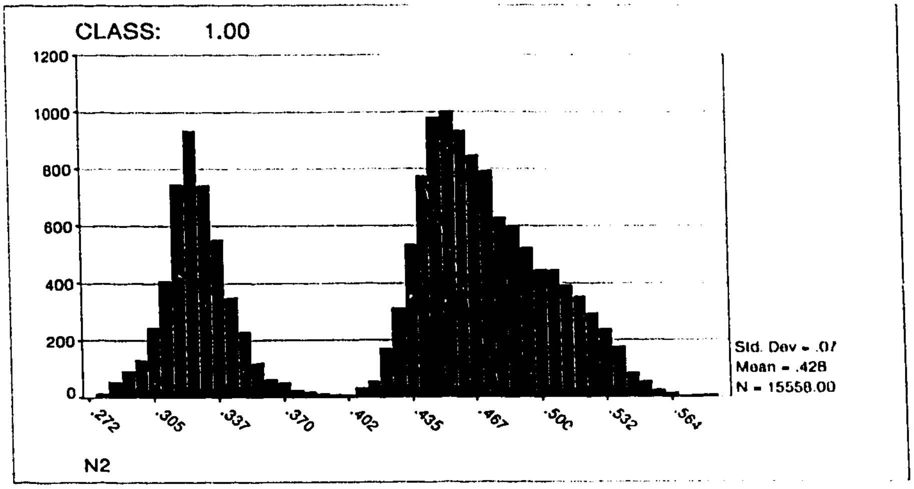


FIGURE 28. Histogram of variable 1, class 1.

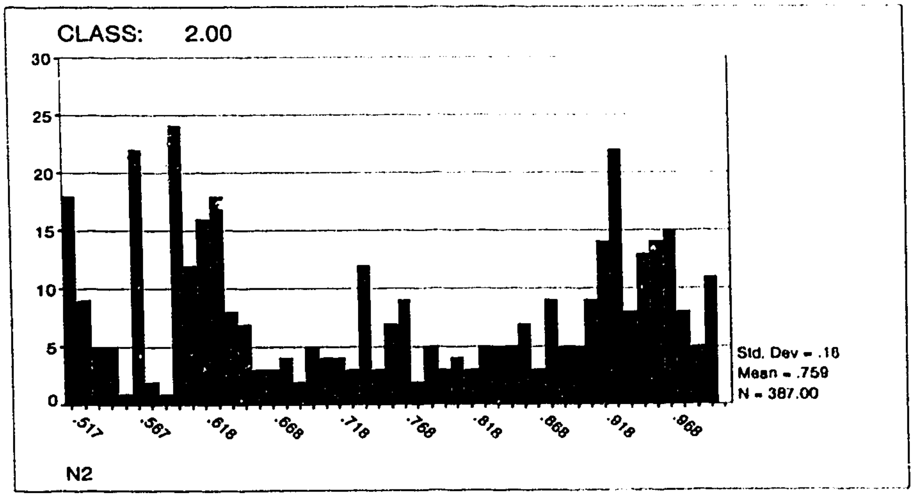


FIGURE 29. Histogram of variable 1, class 2.

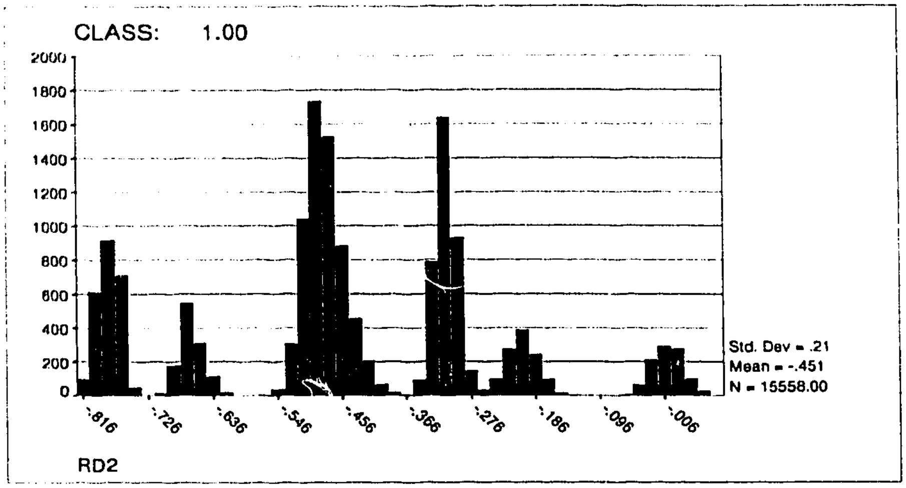


FIGURE 30. Histogram of variable 2, class 1.

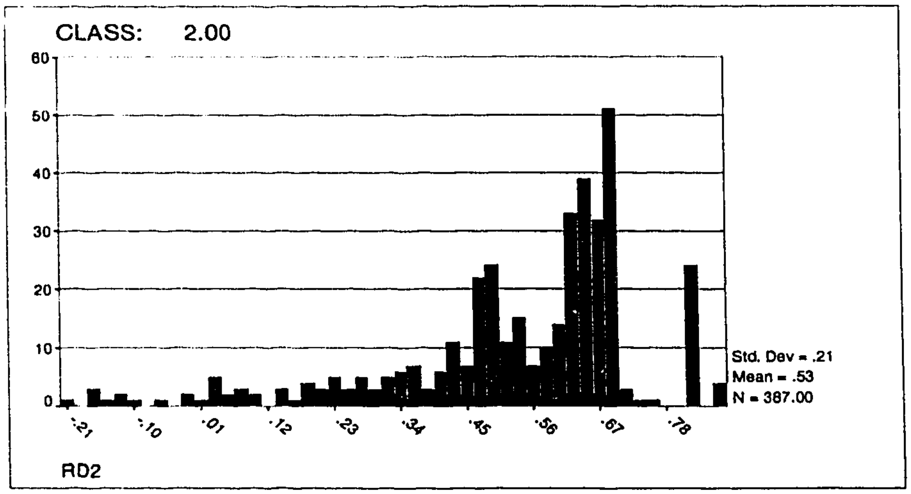


FIGURE 31. Histogram of variable 2, class 2.

$$\hat{\pi}_i(D_L) = \frac{\pi_i P_i(D_L)}{\sum_{k=1}^{p} \pi_k P_k(D_L)} = \frac{\pi_1 P_1(-0.03821)}{\sum_{k=1}^{2} \pi_k P_k(-0.03821)} = \frac{(0.5)(1.0)}{(0.5)(1.0) + (0.5)(1.0)} = 1.0 \qquad \text{(EQ 44)}$$

TABLE 10. Sample discriminant function computations.

True Class	V1	V2	Discriminant Function Value	A Posteriori Probability for Class 1	A Posteriori Probability for Class 2	Classified Class
1	.3423	.0041	-.03821	1.00000	.00000	1
1	.3473	-.0066	-.01649	1.00000	.00000	1
1	.3257	-.0192	-.25431	1.00000	.00000	1
1	.3339	-.0114	-.15639	1.00000	.00000	1
1	.3124	-.0154	-.37227	1.00000	.00000	1
2	1.0000	.4668	7.37287	.00000	1.00000	2
2	.9286	.4800	6.72202	.00000	1.00000	2
2	.6247	.6137	4.14284	.00043	.99957	2
2	.6015	.6543	4.02017	.00096	.99904	2
2	.5950	.6301	3.89902	.00214	.99786	2
2	.6645	.1047	2.77373	.77922	.22078	1

3.5.5 Statistical Software (SPSS)

In the course of this research, several methods were implemented as MATLAB code for performing group allocation and discriminant analysis. However, it was found that many of the algorithms for performing discriminant analysis were readily available in existing software packages. In addition, the canned algorithms had been optimized for computational efficiency on certain computer platforms. For this reason, it was decided that we should divert our efforts to using an existing professional statistical analysis software package. It was determined that SPSS (Statistical Package for Social Scientists) was a suitable tool. In addition to the SPSS Base package, the SPSS Professional package [55] was also needed in order to perform cluster and discriminant analysis. In this section we discuss some of the features of the SPSS Graduate Student Pack for Windows, Version 6.1.

The documentation for the SPSS Professional package [55] [71] gives explanations for all of the discriminant analysis tools that are available to the software user. Important to this research was the section regarding discriminant analysis.

SPSS allows the user to compute various statistics on the data that have been entered, including means, univariate ANOVAs, box-M's, Fisher's Coefficients, unstandardized coefficients, within-groups correlation, within-groups covariance, separate-groups covariance, and total covariance. These statistics are useful for understanding the nature of the data, but do not aid the classification or feature evaluation strategies.

Feature evaluation and selection may be performed in a stepwise fashion, using a number of different evaluation criteria. The available feature evaluation methods are Wilk's lambda, unexplained variance, Mahalanobis distance, Smallest F ratio, and Rao's V. These methods were briefly mentioned earlier.

The SPSS classification procedure is based upon Bayes' theorem, where the prior probabilities may be adjusted to suit the situation. In general, only linear discriminant functions may be used for classification. However, it is stated in [55] that for the two-class case, logistic regression may be used in place of quadratic discrimination. SPSS also has the ability to implement pseudo-quadratic discriminant functions. Optimal quadratic discriminant functions are not necessarily implementable since the classification stage of SPSS is based on using the linear discriminant functions even for the case of quadratic discrimination. Specifically, the between groups covariance matrices of the linear discriminant functions are used for pseudo-quadratic discrimination rather than the between groups covariance matrices of the individual discriminant variables.

Another useful feature of SPSS is its ability to give the coefficients of Fisher's linear discriminant functions [55]. Fisher's functions are a combination of optimal linear discriminant functions with Bayes' classification rule incorporated within them. The resulting functions yield classification accuracies identical to optimal linear discriminant functions, however the posterior probability of class membership is not available since Bayes' theorem is not used separately for classification. Fisher's discriminant functions can be directly applied to the discriminant variables and the function results are directly used for performing classification. A function is calculated for each class, and the class with the highest function value is then the class to which an observation is allocated. In contrast, optimal linear discriminant function results must be input into Bayes' theorem to calculate the posterior probabilities of class membership. Due to this fundamental difference, Fisher's functions are simpler to implement in a classifier that is external to SPSS. For a complete account of Fisher's discriminant functions consult [48] and [54].

The SPSS version used for this thesis was designed for the DOS/Windows 3.1 platform. Most of the commands available in the software were executable from pull-down menus and forms in a graphical format. However, the user may also write script files so that the particular steps of a statistical analysis may be easily reproduced. We will now discuss the general sequence of events required to perform discriminant analysis and classification using SPSS. The <FILE> notation is used to represent the execution of a command in the "File" pull-down menu.

1. Discriminant variable data must be loaded before any analysis can be performed. The discriminant variables can be calculated using any method the user wishes, as long as the data is formatted in a way that SPSS can read. For example, the discriminant variables for this thesis were calculated using MATLAB, and then stored in a tab-delimited ASCII format which is readable by SPSS using the <File><Read ASCII Data> command. The known class membership of each discriminant variable case must also be entered into SPSS.

2. The next step is select the <Statistics><Classify><Discriminant> command. When that command is executed a form is displayed which must be filled out. First the variable that defines the known class membership must be identified. Next, the discriminant variables intended for inclusion in the analysis must also be identified. The following five steps outline the sub-forms that must also be completed before analysis can proceed.

3. The <Select> form: This form allows the user to select cases from the discriminant variable data that will be used for construction of the discriminant functions, or training. The remaining data cases will not be used for training, but the classifier will attempt to perform classification using all available data. This allows the user to perform cross-validation of the classification accuracy results.

4. The <Statistics> form: This form has three sub-regions allowing the user to instruct SPSS to output various statistical information. Various descriptive statistics of the discriminant variables can be output including means, univariate ANOVAs, Box's M, within-groups correlation, within-groups covariance, separate-groups covariance, and the total covariance. By default SPSS will output the standardized canonical discriminant functions, however these functions can only be applied to normalized discriminant variables. Therefore, in this form the user can instruct SPSS to output the unstandardized canonical discriminant functions which can be directly applied to the discriminant variables even if every variable is measured in different units and has different ranges. Finally, the Fisher's linear discriminant function coefficients can also be output.

5. The <Method> form: This form is only valid if a <Use stepwise method> is selected from the main discriminant analysis form. If a stepwise method for selection of discriminant variables is used, then the method(s) for evaluating the relative worth of each variable must be specified. The methods available are Wilks' lambda, unexplained variance, Mahalanobis distance, smallest F ratio, and Rao's V.

6. The <Classify> form: This form has four regions. Region one allows the user to specify the a priori probabilities for each class. Region two is used to specify if linear or pseudo-quadratic discrimination is to be used. If the within-groups covariance matrix is specified, linear classification is performed using the pooled within-groups covariance matrix. If the separate-groups covariance matrix is specified, pseudo-quadratic classification is performed using linear discriminant functions, however the classification is implemented using the sep-

arate within-groups covariance matrices for each class. The rest of the form allows the user to select the type of plots and results data that they wish to have SPSS output.

7. The <Save>· form: This final form is used to specify which classification information is to be saved with the discriminant variable data. For each case SPSS can save the predicted group membership, the discriminant function scores, and the posterior probabilities of group membership.
8. Now that the options in all forms have been selected, the class variable is defined, and the discriminant variables have been specified, the discriminant analysis run can be executed. SPSS will display results in a log file, graphical charts, and in the form of added variables to the data.

There are many other features that are useful in SPSS for studying the discriminant variables. Histograms can be plotted, statistics computed, scatter diagrams can be constructed, and so on. For a complete discussion of the capabilities of SPSS consult the user manuals. To find a discussion on the SPSS discriminant analysis capabilities consult [55].

Chapter 4

4.0 Research Infrastructure

In the following sections we describe the equipment and software that was used to gather and generate signal data. A comprehensive table of all recorded and simulated signal samples used in this research is included as Appendix . The recorded and simulated signals used in this study total approximately 65 million 8-bit µ-law PCM samples, or about 2.25 real-time hours.

4.1 NeXT Workstation

The NeXT workstation was a natural choice for use as a data gathering tool due to its inclusion of a Motorola M56001 DSP on the motherboard, and serial channels for the control of devices such as FAX/modems. Although NeXT no longer manufactures the "black box" NeXT workstations, the operating system bundle is still supported, and various distributors still have stocks of NeXT peripherals. As mentioned above, an excellent feature of the NeXT machine is the built-in DSP. The DSP is a Motorola 56000 series fixed-point processor that has the necessary power to perform signal processing tasks on voice-band signals.

In our configuration, a NeXT workstation is connected to an analogue telephone line via the built-in DSP and a Hayes ISDN Extender, as shown in Fig. 32. This channel facilitates call recording with either 8-bit µ-law or 16-bit linear accuracy. It is convenient that the 8-bit µ-law coded recording of the telephone line is the same as the coding done in most COs. In this way some of the signal distortion introduced by quantizing at the COs is modeled by the Extender. A high-speed FAX/modem is also connected to one of the

NeXT workstation serial ports allowing simultaneous operation of the Hayes Extender and FAX/modem. With the aid of a software package (written as part of the thesis project) the FAX/modem can be put into operation and the phone line monitored via the Extender. Also, an analogue telephone is connected to the same telephone line allowing normal voice calls to be recorded and/or controlled. The Extender is capable of supporting up to three analogue telephone devices such as a normal phone, FAX machine, and a modem.

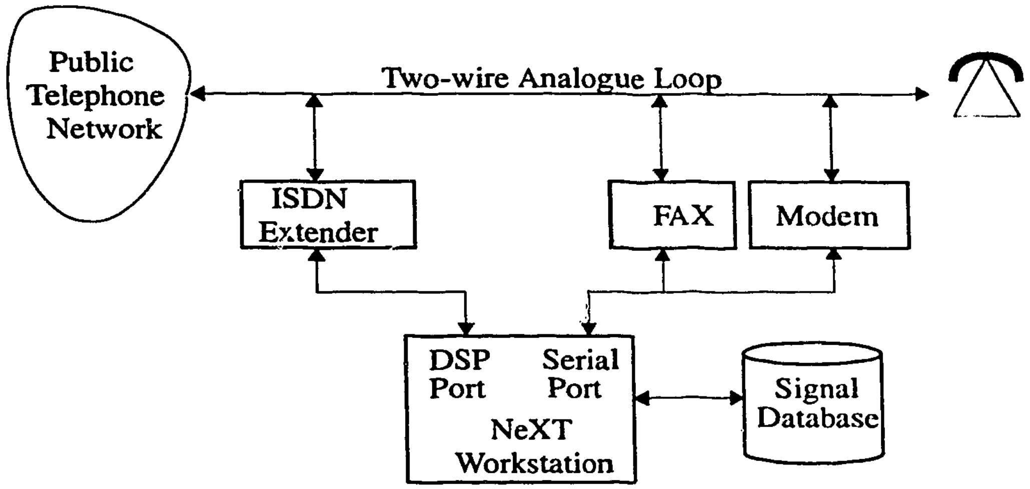


FIGURE 32. Sketch of research system.

An inherent problem with this data gathering system is that the monitoring point is a 2-wire connection rather than the 4-wire connection that would otherwise be monitored within the PSTN. The effects of this problem were studied and compensated for. This problem affects different signal types in different ways. First consider speech signals. Normal conversations tend to be half-duplex, when only one party speaks at a time. Thus, monitoring at the 2-wire point is very similar to monitoring at the 4-wire point, except the flat attenuation for each signal direction and the echo strengths will be somewhat different. The same argument can be made for FAX calls since they are strictly half-duplex in nature. The same cannot be said for VBD communications, which are usually full-duplex.

In this case, monitoring at the 2-wire point is similar to monitoring the nearest end-user modem. The far-end modem signal will be attenuated by about 16.5 dB at the near-end 2-wire loop, hence the near-end signal is far stronger and will dominate any recorded signal.

Due to the on-board DSP in the NeXT workstation, it is possible to perform real-time signal processing on incoming voice-band signals. However, this capability was not used to perform signal classification since the target system is to function at the T1 level. For this reason, another test-bed system in under development and is described later in this chapter.

Many of the recorded signals that appear in Appendix are speech signals. All speech signals were recorded in a standard way. Both male and female talkers would phone the analog telephone connected to the NeXT computer. The age of the talkers ranged from 18 years to approximately 60 years. The computer was programmed to answer incoming calls and transmit a prerecorded message. The prerecorded message was "Hello, please leave your message now." The callers were instructed to read out-loud two sentences. The sentences were "Nine rows of soldiers stood in a line." and "The beach is dry and shallow at low tide.". The callers were instructed to speak a normal conversational volume and pace. These sentences were selected by O'Neal and Stroh for their 1972 study on the affects of PCM on speech [56].

Four other voice recordings were made. Two recordings were made of a person attempting to mimic the sound of a modem. One recording was made of a person whistling, and another of simple silence. The signals were used to explore the robustness of the developed classification methods.

To offset the complications introduced by the 2-wire monitoring point problem we also performed extensive signal simulations, which are discussed next in this chapter.

4.2 MATLAB Simulations

MATLAB is a software system that is very useful as an off-line signal processing tool because it is optimized to process vectors and matrices efficiently. Also included in MATLAB are extensive graphical user interface programming capabilities. These and other features made MATLAB a natural choice for the coding and testing of off-line simulation and classification algorithms.

MATLAB is a matrix-oriented mathematical software package available in TRLabs on the workstation computers. Included in the software is what is called the Signal Processing Toolbox. This toolbox is a collection of signal processing functions that are very useful for examining discrete time signals such as voice-band communications. Included in the toolbox are modulation/demodulation functions, Fourier transforms, spectral analysis, filtering, and a host of other helpful tools. Also, the programming language of MATLAB allows C-like code to be written to perform other necessary tasks, and build efficient user interfaces. User interfaces for MATLAB may be textually based or graphical. Programming interfaces of either type is quick and efficient.

Since MATLAB is matrix-oriented, it is ideally suited for processing arrays such as discrete time signals. Thus, MATLAB is an excellent tool for simulating the various types of voice-band communication that exist. Work was done on simulating QAM and DPSK signals corresponding to the various ITU standards and recommendations mentioned earlier. The simulation models include several realistic features of telephone communication channels such as Gaussian noise, 8-bit μ-law PCM, talker and listener echoes, flat attenuation, attenuation distortion, envelope delay distortion, and frequency offsets. MATLAB provides for the incorporation of external data, which is a useful feature for processing the recorded data from actual recorded signals.

Extensive simulations were performed using MATLAB on DEC 5000 Alpha workstations. Simulations focussed on creating accurate models for the different transmission methods of the FAX and data modulation standards. All simulations involved the study of QAM signals that correspond to each of the relevant ITU standards and recommendations. Through these simulations, a greater understanding of QAM was obtained and insights to possible classification tools were made. The block diagram in Fig. 33 shows a system model that was developed for simulations. The program that corresponds to this model is named GENERATOR. Various parameters were changed from one simulation run to the next, such as the modulation method network impairment parameters. The graphical user interface developed for this simulation system is shown in Fig. 34.

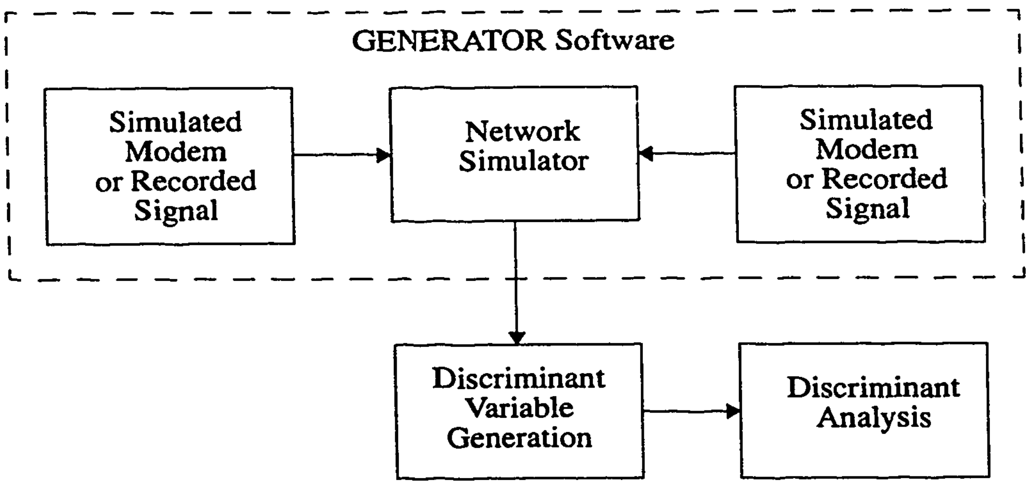
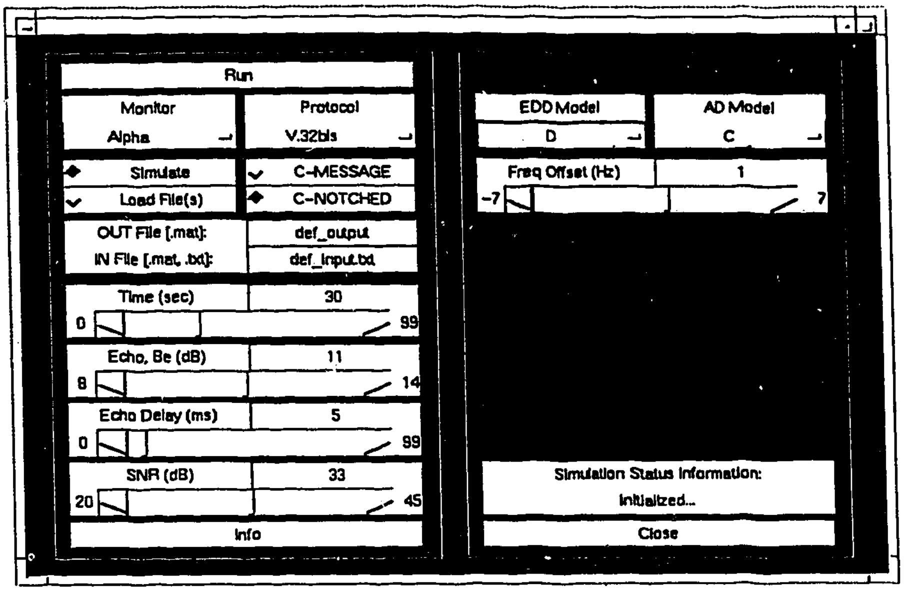

GENERATOR Software
Simulated Modem or Recorded Signal
Network Simulator
Simulated Modem or Recorded Signal
Discriminant Variable Generation
Discriminant Analysis

FIGURE 33. General simulation model.



Monitor: Alpha

Protocol: V.32bis

◆ Simulate
✓ Load File(s)

✓ C-MESSAGE
◆ C-NOTCHED

OUT File [.mat]: def_output
IN File [.mat, .txt]: def_input.txt

Time (sec): 30

Echo, Be (dB): 11

Echo Delay (ms): 5

SNR (dB): 33

EDD Model: D

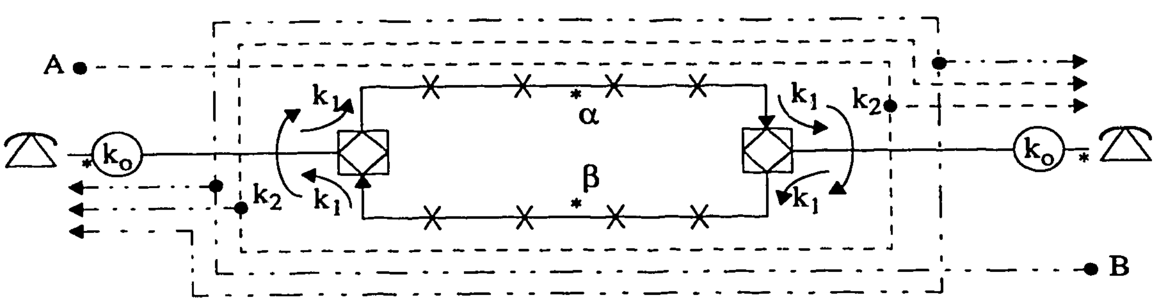
AD Model: C

Freq Offset (Hz): 1

Simulation Status Information:
Initialized...

FIGURE 34. Graphical interface of GENERATOR program.

Within GENERATOR, the user can accurately simulate an end-to-end modem or FAX call, not including the setup or training sequences. Only the high-speed data portions of the calls are simulated since these are what takes up a bulk of the time during a VBD call or FAX call. Fig. 35 shows the conceptual overview of the simulation structure overlaying a simple network model.



* = monitoring point
x = network link point
k_o, k_1, k_2 = losses

FIGURE 35. Network simulation block diagram.

Fig. 35 shows the general structure of the network transmission model, while Fig. 36 shows the impairment block that the simulated or imported signals are first sent through. The source signals, A and B, can either be simulated or imported from a previous recording. The library of possible simulated signal types includes all of the major ITU type modem and FAX signals with the exception of V.34. The V.34 standard was found to be prohibitively complex to simulate effectively.

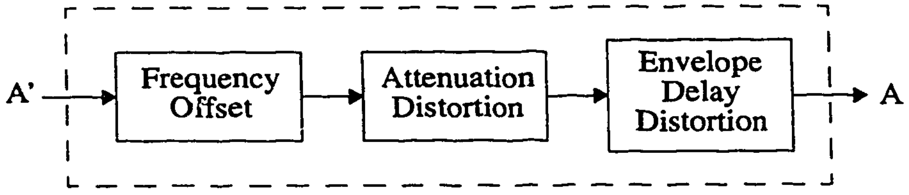


FIGURE 36. Impairment block.

As Fig. 35 illustrates, there are 4 possible monitoring points in the simulated network including the caller and answer 2-wire loops, as well as the caller transmit and receive directions of the 4-wire connection. Signal sources *A* and *B* can be predistorted within the impairment block shown in Fig. 36. The dashed lines of Fig. 35 represent the paths of the A and B signals as they are sent and echoed in the simulated network. The output arrows from the dashed lines are summed to obtain the received signals at the customer premises. A general loop loss is defined by k_0 for each subscriber loop. The hybrids have two losses associated with them, denoted by k_1 and k_2. The interconnection stages are assumed to be lossless digital links. Before the monitored signal is saved from the monitoring points, white noise can be added and the signals can be impaired by several methods. The code used to simulate the signal impairments is given in Appendix B.

4.2.1 Simulated Signals

The simulated signal types include all of the VBD and FAX signal types with the exception of V.34. Nine signal types can be simulated, including V.22, V.22bis, V.27ter, V.27ter (fall-back mode), V.29, V.29 (fall-back mode), V.32, V.32bis, V.17. In the case of V.27ter and V.29, the first fallback mode of each standard was also simulated. Note that V.27 fall-back mode has a different PSD from the normal mode. The V.29 fall-back mode was implemented since it is frequently used, and this signal was also used for exploring the performance of some proposed constellation discrimination algorithms. Refer to Appendices C, D, E, and F for sample code that was used in the generation of simulated QAM signals.

The signals can be monitored from one of four points within the simulated network. Both of the four-wire monitoring points, α and β, were used as actual recording points for the simulated signals. These two points are known as "alpha" and "beta" in the GENERATOR software. Most signal simulations were approximately 25 seconds in duration.

4.2.2 Impairment Models

Complex impairment models were required to simulate the various possible analogue impairments that can be expected in the PSTN. Not all impairments are possible or practical to simulate. Thus only a reasonable subset of the possible impairments was selected, the choice of impairments being based upon the affect that impairments have on VBD communications since these are the only types of signals simulated. The impairments modeled include additive noise, frequency offsets, attenuation distortion (AD), and envelope delay distortion (EDD). The impairments are in addition to the flat attenuation and echoes found in the network transmission model discussed earlier. This subset of impairments selected can be justified by results given in [12], [11], and [63]. Another major problem that VBD modems face is impulse noise, which often results in a line drop; however, this type of impairment is not modeled since we are concerned with monitoring successful communications and not predicting line drops.

In total, five different impairment models were selected for use within the simulator. Although the design of the simulator permits great flexibility in the selection of impairments, only five combinations of impairments levels were used. These five models were chosen to correspond very closely to those used by Jablon [42] in his work on blind equalization of voice-band modem signals. The five models represent a range of impairment levels from light, to moderate, to severe. Table 11 shows the impairment levels; individual impairments are discussed in the following sections.

TABLE 11. Five impairment models implemented by the GENERATOR.

Model No.	C-Notched SNR (dB)	Envelope Delay Distortion (@600 Hz, @3000 Hz) in (μs) rel. to 1700 Hz	Attenuation Distortion Slope (@604 Hz, @3004 Hz) in (dB) rel. to 1004 Hz	ΔF (Hz)	Echo Delay (ms)	B_e (dB)
1	35	A (1000, 950)	A (0.0, 0.0)	0	5	11
2	33	B (1170, 1350)	B (0.6, 1.2)	0	5	11
3	32	C (1850, 1350)	B (0.6, 1.2)	0	5	11
4	30	D (2300, 1850)	C (1.1, 8.0)	1	5	11
5	30	D (2300, 1850)	C (1.1, 8.0)	0	5	11

In Table 11 the B_e term refers to the transhybrid loss factor. This factor is used for determining the amount of signal that is passed from the receive side of the hybrid back to the transmit side. This is a major source of echoes.

4.2.2.1 Additive Noise

Conventionally, additive noise is measured over the PSTN with a weighting filter. This filter is often a C-Message filter or a C-Message filter plus a C-Notch filter. These filters are weighted according to the subjective effects of noise at different frequencies. In order to add the correct amount of noise power to the simulated signals, it is necessary to measure the noise in the same way as the EOCS [12] study that performed the original measurements. Thus we designed the C-Message and C-Notched filters from their specifications in IEEE Standard 743-1984 [29]. This standard gives the poles and zeros for an analog C-Message and C-Notch filter as well as the ideal filter responses. However, the simulation system requires a discrete time implementation of the filters.

Two main methods for transforming analogue filters into digital filters are discussed in [57]. These methods are the bilinear transform and the impulse invariant transform. The bilinear transform has the advantage of having equi-ripple pass and stop-bands, and can be used for any type of filter (high-pass, low-pass, etc.) transformation. In contrast, the impulse invariant method can only be used on low-pass and band-pass filters, and the stop-band ripple is not controllable. The serious problem of aliasing high-frequency portions of the signal is a limitation of the impulse invariant method.

Both methods were used to try to convert the pole/zero filter descriptions from [29] into the digital domain. It was found that while the bilinear transform method performed worse that the impulse invariant method for the C-Message filter, the opposite is true for the C-Notch filter. This result can be explained by the characteristics of each type of transform, and their intended application.

The C-Notch design used a bilinear transform, and prewarping around the 1000 Hz frequency point. The analog filter pole locations in rad/sec are {-197 +/- j5640, -1310 +/- j6209, -249 +/- j7132} and the zero locations are {+/- j6202, +/- j6346, +/- j6494}. The subsequent frequency response of the discrete filter is given in Fig. 37. This design fits within the specifications set out in [29].

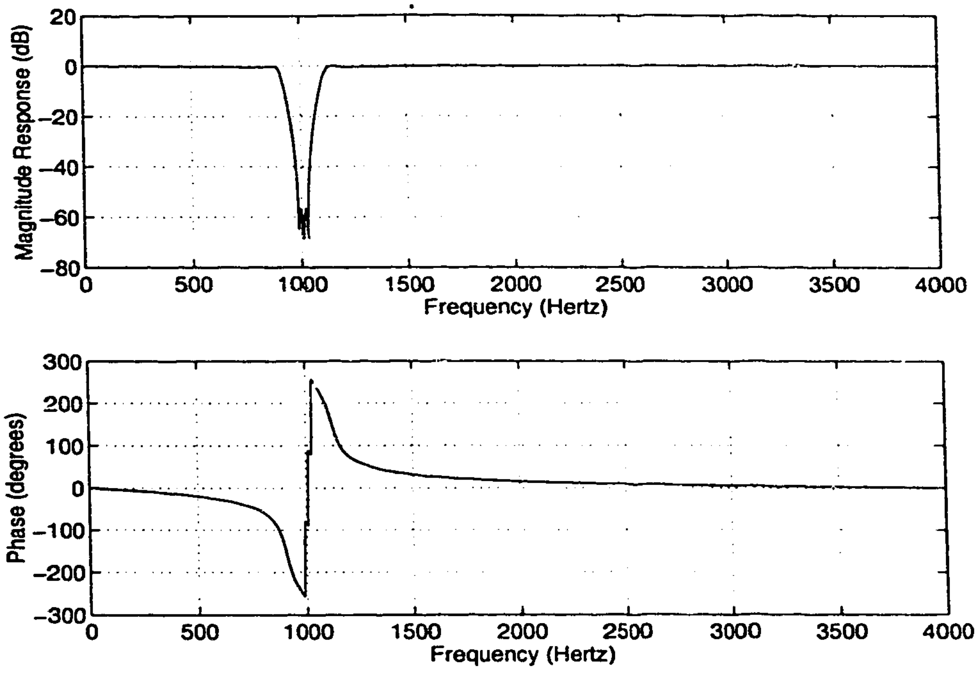


FIGURE 37. Frequency response of digital C-Notch filter.

The C-Message filter designed by the impulse invariant transform method is shown in Fig. 38 along with the ideal response (in x marks). As the plot shows, the designed filter matches the ideal filter shape well within the IEEE allowable tolerances, except at high frequencies. However, when the digital filter design deviates from the ideal response, the loss is already quite high, and these frequencies will be filtered away by any band-pass filter in the transmission network. The analog pole locations for the C-Message filter are {-1502 +/- j1267, -2439 +/- j5336, -4690 +/- j15267, -4017 +/- j21575} and the zeros are at {0, 0, 0, 0, infinity, infinity, infinity, infinity} (measured in rad/sec). Code for generating the C-Notched and C-Message filters is given in Appendices G and H, respectively.

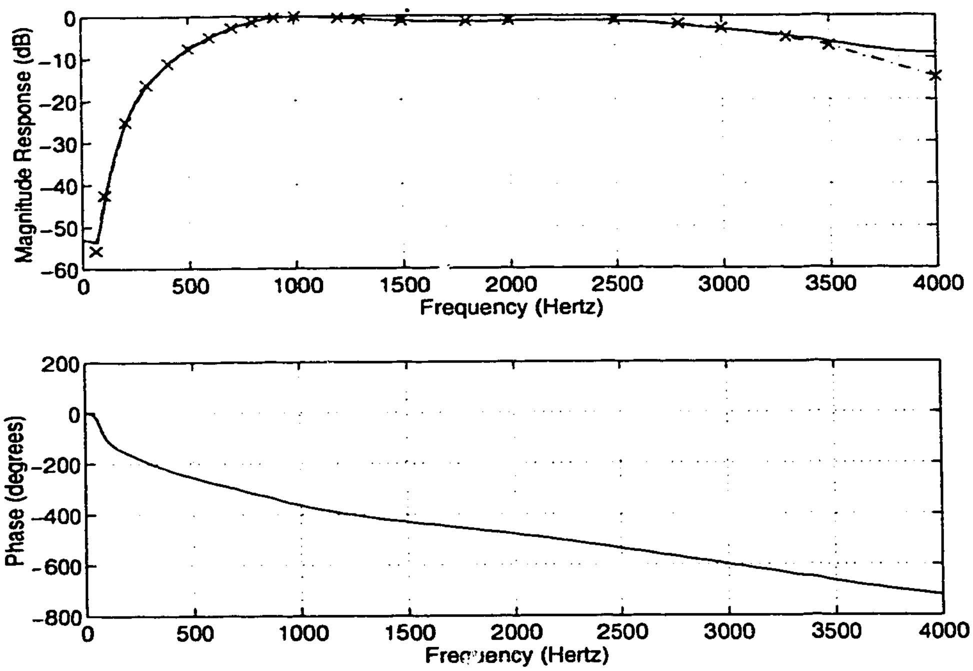


FIGURE 38. Frequency response of C-Message digital filter design.

4.2.2.2 Frequency Offsets

Frequency offsets were set to a specified constant level, usually about 1 Hz, and were implemented by simply modifying the transmitter carrier frequency by the appropriate amount.

4.2.2.3 Attenuation Distortion

Attenuation distortion was difficult to model since it required a linear phase filter with a generalized magnitude response specification. The Parks-McClellan algorithm [57] was used to transfer the general magnitude response specification to an FIR (Finite Impulse Response) filter design. MATLAB was used as the tool to perform the transformation. Three different AD models were selected. The first has no distortion. The second model is equivalent to the mean values of a medium length connection in the ECOS [12].

The third model is equivalent to the 90% values of a medium length connection in the ECOS. These three models represent the ideal case, average case, and a worst case, respectively. Code for generating the AD filters is given in Appendix I.

Forty tap designs were used for the AD filters. Note that there are only two actual filters since the first model has no distortion. The frequency response of the second design is given in Fig. 39, and the frequency response of the third design is given in Fig. 40. The x marks indicate the ideal frequency response for the model, which was obtained from the ECOS. Note that there are x marks above the normalized frequency point of 0.9, and below the normalized frequency point 0.05. These marks were set in the design, and are not specified in the ECOS document.

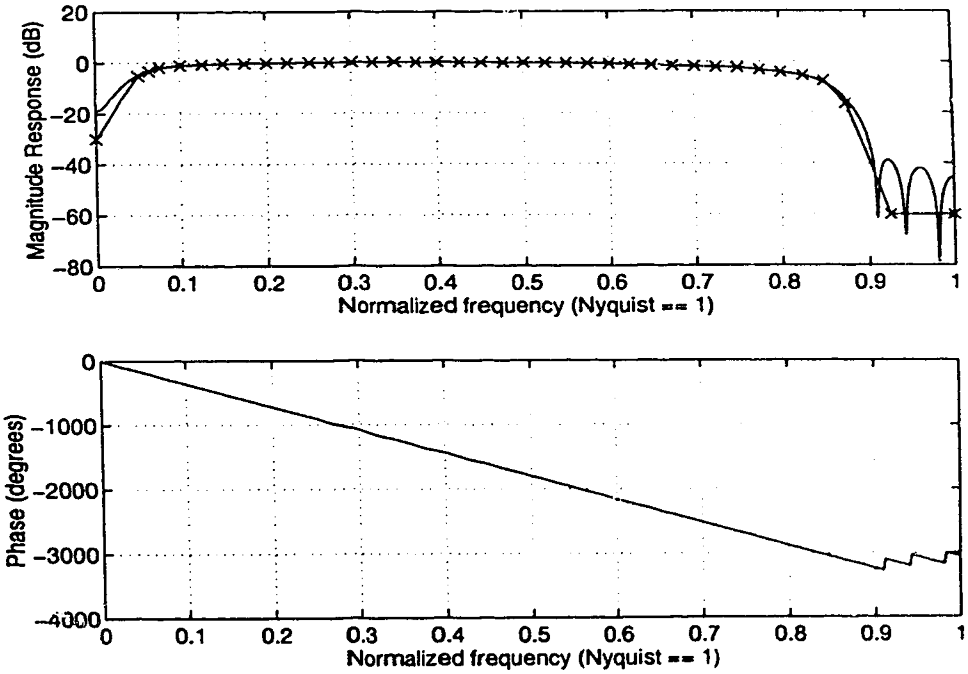


FIGURE 39. Frequency response of AD model 2 FIR filter.

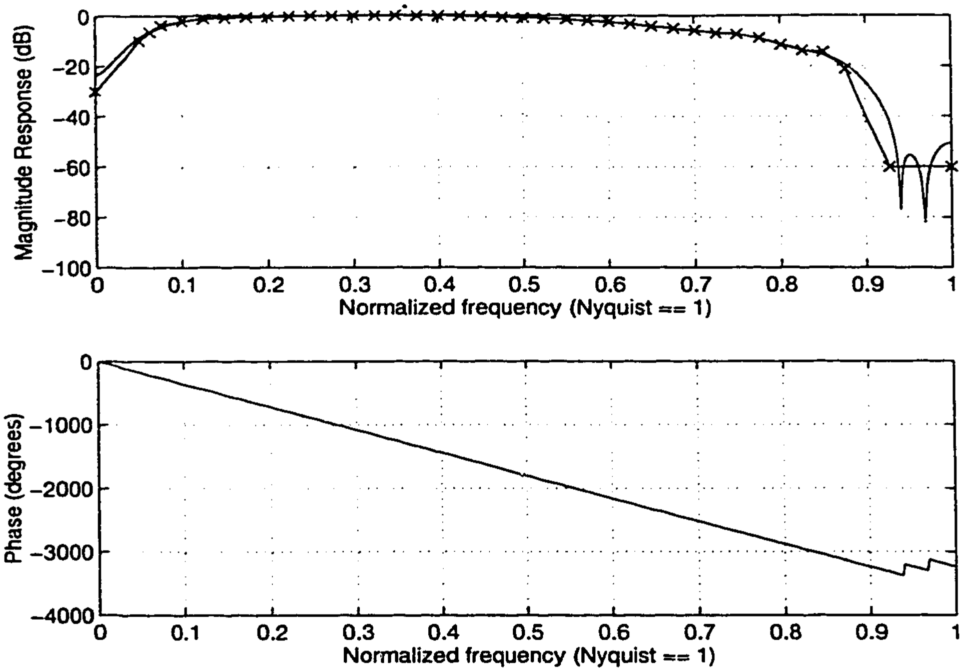


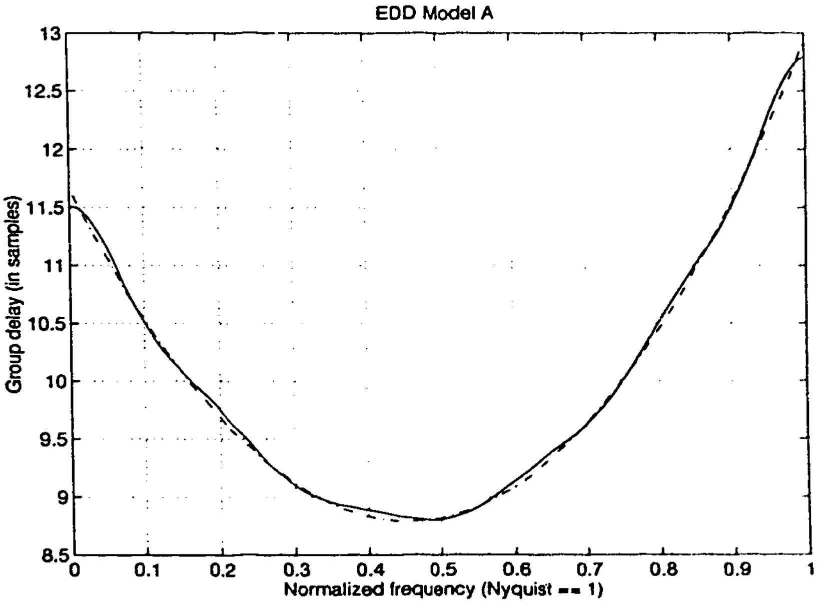
FIGURE 40. Frequency response of AD model 3 FIR filter.

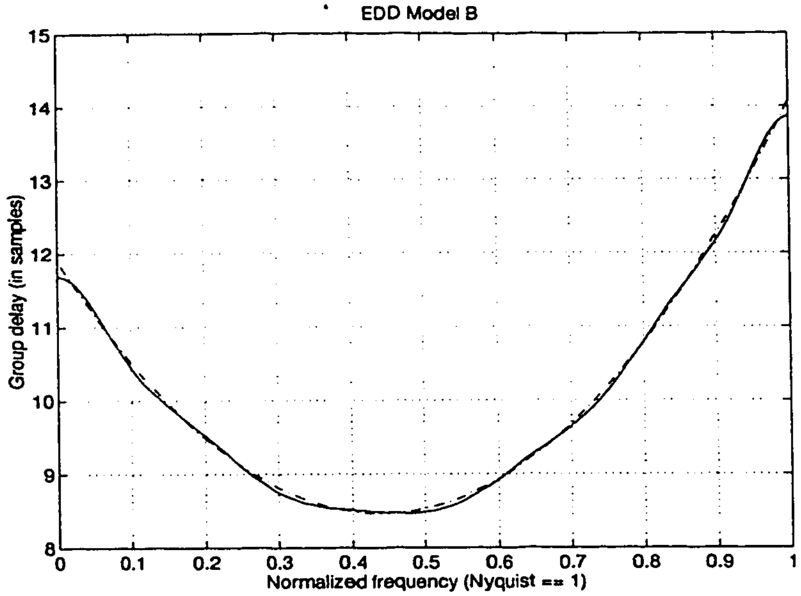
4.2.2.4 Envelope Delay Distortion

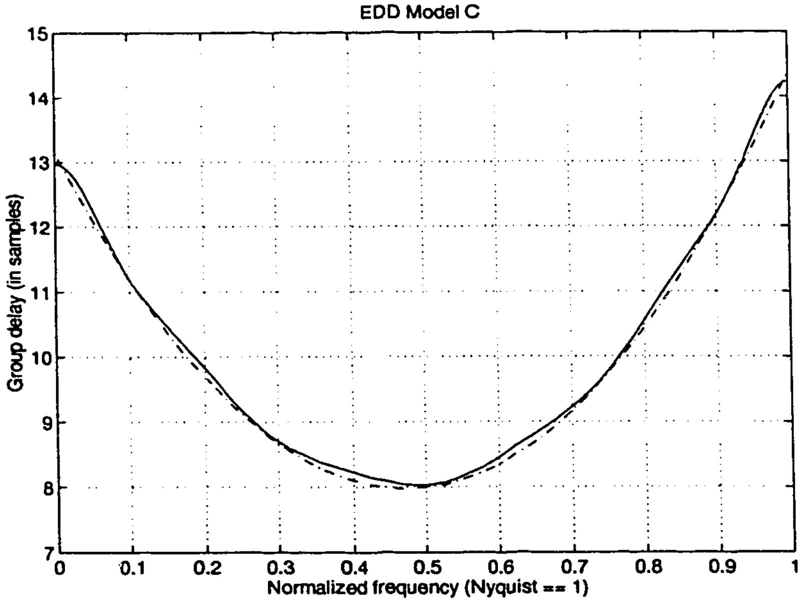
The EDD filters were the most difficult to design since the filter specifications are given as a specified group delay. The ECOS gives plots for the group delays of various channels, however the group delays can be approximated by a parabolic function that matches 3 different points. The points are specified in the ECOS. Selection of the EDD models to use was based upon a study of blind equalization of modems done by Jablon [42]. In the study Jablon used 4 different EDD models. Those same EDD models were generated for this work.

The four ideal EDD models used are shown in Fig. 42. From these descriptions, a system of equations can be solved to obtain the parabolic equation for each model. For each filter this equation is then integrated to determine the desired phase response of the filter. The final filter design must have an all-pass magnitude response, while maintaining the desired phase response.

In order to perform the filter design, software written by Markus Lang (Rice University) was employed. The software algorithms correspond to an all-pass filter design methodology set out in [49], which is based upon a generalized Remez algorithm [57]. A unique optimum convergence is guaranteed, and the algorithm reaches convergence quickly. The final designs yield IIR filters of order 10 for this application. The magnitude responses for the filters are so flat that they only have at most 2e-13 dB of ripple. In Fig. 41 the group delays of the designed filters are laid over the ideal group delays, plus a constant offset. (It is not possible to implement a real filter with a negative group delay.) As the figure shows, the designed filters match the ideal models very well. A summary of the ideal models is shown in Fig. 42. The code used to generate the EDD models is given in Appendix J.




EDD Model B
Group delay (in samples)
15
14
13
12
11
10
9
8
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Normalized frequency (Nyquist == 1)


EDD Model C
Group delay (in samples)
15
14
13
12
11
10
9
8
7
0
0.1
0.2
0.3
0.4
0.5
0.6
0.7
0.8
0.9
1
Normalized frequency (Nyquist == 1)

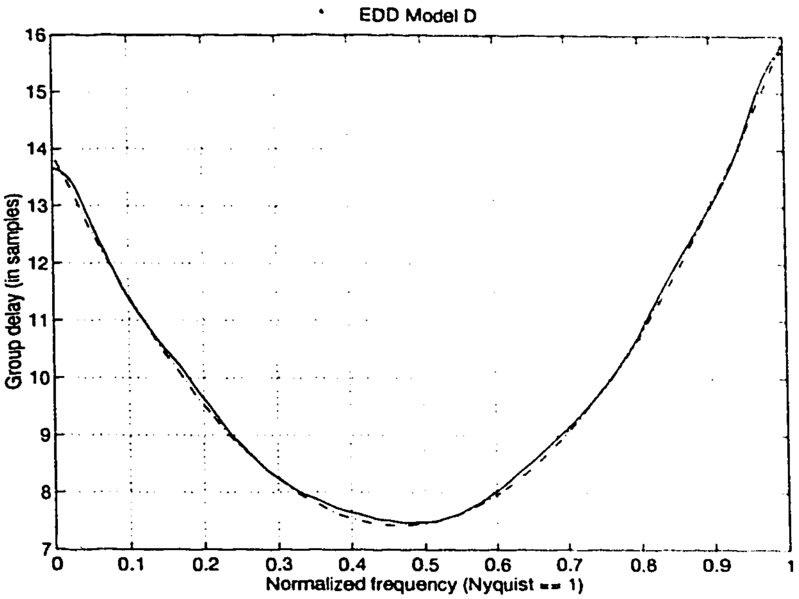


FIGURE 41. Group delay plots for EDD ideal and designed models.

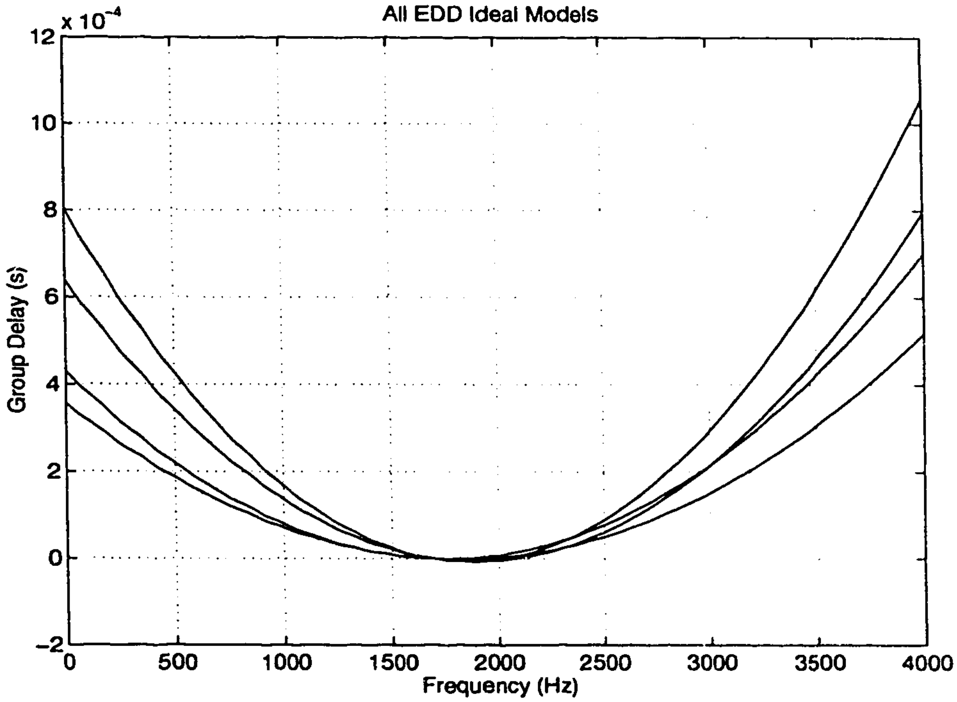


FIGURE 42. Summary of ideal EDD group delay models.

4.3 T1 and DSP Testbed

In order to test the algorithms in real time at the T1 rate, a Personal Computer (PC) based test-bed is being developed by another graduate student, Baolian Xu. The system incorporates a T1 bidirectional interface, a floating point DSP, and an ISA bus based Intel 80486 type host computer system with its own memory, hard disk storage, video, and keyboard interface. This system is under development at the time of this writing. In the future, real-time studies of the algorithms developed in this work will be tested on this test-bed. Fig. 43 shows the general architecture of the test-bed. The dotted line at the top of the figure represents a hardware loop-back mode. In this configuration, the host computer can be used to load sample signal data from hard disk to the T1 card, then the T1 card can transmit a T1 signal, receive the T1 signal, and forward the received signal samples to the DSP board. This allows for completely self-contained testing.

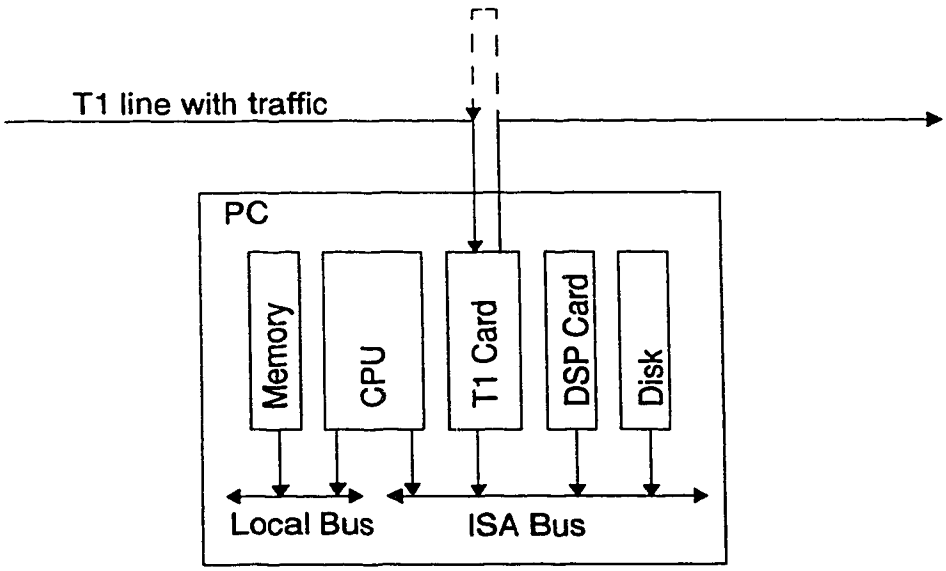


FIGURE 43. Baolian Xu's PC testbed.

Chapter 5

5.0 Primary Methods and Results

5.1 Statistical Signal Characterization

5.1.1 Statistical Pattern Recognition

Conventional signal characterization methods such as FFT (Fast Fourier Transforms), DFT (Discrete FT), cross-correlation and autocorrelation have proven to be effective in earlier studies, yet they often require a great deal of computational power to perform. For this reason, it would be desirable to utilize an equally effective yet more computationally efficient classification method. Such alternatives do exist, and have been applied to other problems with success, providing certain assumptions can be met.

Statistical pattern recognition is a well-established field of engineering [20]. It is a classification technique that has been applied to a broad range of problems including optical character recognition, machine vision, meteorology, psychology, waveform analysis, and many others. The general principle is that a single or multivariate process may be described by discriminant variables (also known as *features*) that are representative of the different classes in the process. The discriminant variables taken together form a *discriminant variable vector* that can be used to classify the current sample under test using some decision algorithm. *Pattern recognition* is then the problem of estimating probability density functions for different classes in a specific application. Mathematical statistics is the basis for this subject.

5.1.2 Discriminant Functions Utilized

Hirsch describes a specific method for applying statistical pattern recognition [28]. The method presented, called *statistical signal characterization* (SSC), is a way of applying statistical pattern recognition to a time-varying received signal. SSC statistically characterizes a signal by exploiting discriminants that are measured in samples of the signal. Hirsch proposes that four basic signal properties be calculated and used to decide upon the class of the signal under test. Before describing the discriminant variables, or *parameters* as Hirsch calls them, some terminology should be discussed.

Consider a general time varying signal as shown in Fig. 44. Each of the numbered boxes is called a segment. *A segment* is defined as the portion of the signal that lies between adjacent local maxima and local minima. The entire signal under test is divided up into N segments. The absolute signal amplitude difference (measured in units of volts, amperes, et cetera) in a segment is termed the *segment amplitude*, A_i. The *segment period*, T_i, is the time elapsed during a segment. These two discriminant variables of a signal are used to perform classification. For brevity we will refer to the segment amplitude and segment period as just the amplitude and period, respectively.

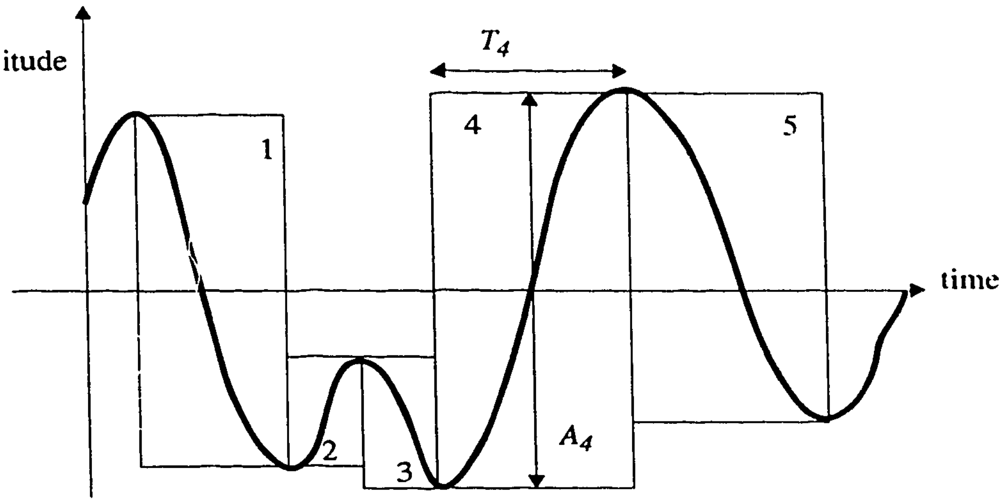


FIGURE 44. SSC discriminant variable description.

[irsch's four discriminants are the amplitude mean, period mean, amplitude mean ι, and the period mean deviation. According to Hirsch, these second order statistiıminants are all that is required to successfully distinguish many apparently simils from one another. The discriminant calculation equations are given below.

$$M_a = \frac{1}{N}\sum_{i=1}^{N} A_i \text{, amplitude mean} \qquad \text{(EQ 45)}$$

$$M_t = \frac{1}{N}\sum_{i=1}^{N} T_i \text{, period mean} \qquad \text{(EQ 46)}$$

$$D_a = \frac{1}{N}\sum_{i=1}^{N} |A_i - M_a| \text{, amplitude mean deviation} \qquad \text{(EQ 47)}$$

$$D_t = \frac{1}{N}\sum_{i=1}^{N} |T_i - M_t| \text{, period mean deviation} \qquad \text{(EQ 48)}$$

The remainder of the SSC method involves determining the window size N and : decision rules for classifying signals with the calculated discriminants. Generally, if the application is not accurately modelled mathematically, extensive empirical data must be gathered to infer decision tables for accurately classifying the signals. The fundamental contribution of Hirsch's work appears to lie in discriminant variable selection, not in classification methodology. Using these discriminant variables, any appropriate classification method could be used.

5.1.3 MATLAB Implementation of Algorithm

To evaluate the SSC method, we integrated it into the existing MATLAB simulation structure. The method has been taken to the point of calculating the required discriminants, tabulating the results, and displaying the results on plots. Code was not written to attempt to perform classification automatically, although the method has proven itself to be reasonably effective for the simulated modem signals. To see the effectiveness of this method refer to the following results section. The reason the SSC discriminant variables were not studied further is that they were found to be incapable of differentiating many of the signal subclasses we wish to classify. For example, V.22*bis* channel 2, V.29, V.32, and V.32*bis* signals are all poorly distinguished from one another using this method. However, the SSC discriminant variables do have some merit in that they are extremely simple to implement and understand.

5.1.4 Results

Through the course of performing many simulation studies and reading other references on statistical pattern recognition, it became evident that the four discriminants described by Hirsch would not be sufficient for this research problem. First of all, since the telephone channel contains several sources of attenuation, comparison of segment amplitude means between two signals would not likely be successful. Different talkers or modems typically transmit different average signal power levels, and network losses will not be the same for all monitored signals, thus the segment amplitude means cannot effectively be compared. A potential solution to this problem could be to use a relative discriminant such as relative deviation or relative variance or to normalize power levels. In this way the amplitude deviation or variance could be divided by the amplitude mean and the resulting normalized discriminant would not depend upon the absolute signal amplitude.

The plots shown in Fig. 45 illustrate the classification capabilities of the SSC discriminants. Each plot refers to a different discriminant. The discriminants are C_a, M_t, D_t, V_t, S_t, and C_t, referring to the relative amplitude deviation ($C_a=M_a/D_a$), period mean, period mean deviation, period variance, period standard deviation, and relative period deviation, respectively. Each column of each plot represents the discriminant variable "margins" a different class of signal. The term "margins" is used here to represent the mean value, standard deviation, and extremum of a random variable. There are eight classes, with one column for each class of signal monitored at the α point in the network. The second group of eight columns represent the same classes, but monitored at the β point in the network. This is useful for recognizing the two directions in a splitband modem call. The 'o' marks represent the mean value for the discriminant as applied to that particular class of signals. The '+' marks represent the standard deviation, and the 'x' marks represent extremum. Ten simulated 0.5 second long signals from each class were used to generate the statistics. Each signal was generated using a C-Notched SNR of 33 dB, 5 ms end-to-end echo delay, and a trans-hybrid loss of 11 dB. Other impairments were not included in the simulation. Finally, the horizontal lines represent the resulting discriminant values obtained from a test signal. In this case, the test signal was a simulated V.32bis modem. Table 12 shows the class definitions used for this experiment.

TABLE 12. SSC simulation classes.

Class Number	Signals Included
1	V.22
2	V.22bis
3	V.27ter
4	V.27ter Fall Back Mode (2400 bps)
5	V.29
6	V.29 Fall Back Mode (7200 bps)
7	V.32
8	V.32bis, V.17

The simulation results indicate that classes 5, 6, 7, and 8 are indistinguishable using these discriminant variables. All of these classes contain QAM signals that have similar operating parameters. The SSC parameters were also applied to the baseband representations of the same signals. Since the baseband QAM signal is complex-valued, the SSC parameters were applied to both the magnitude and phase components independently. This modification was not found to improve the discrimination power of the SSC method.

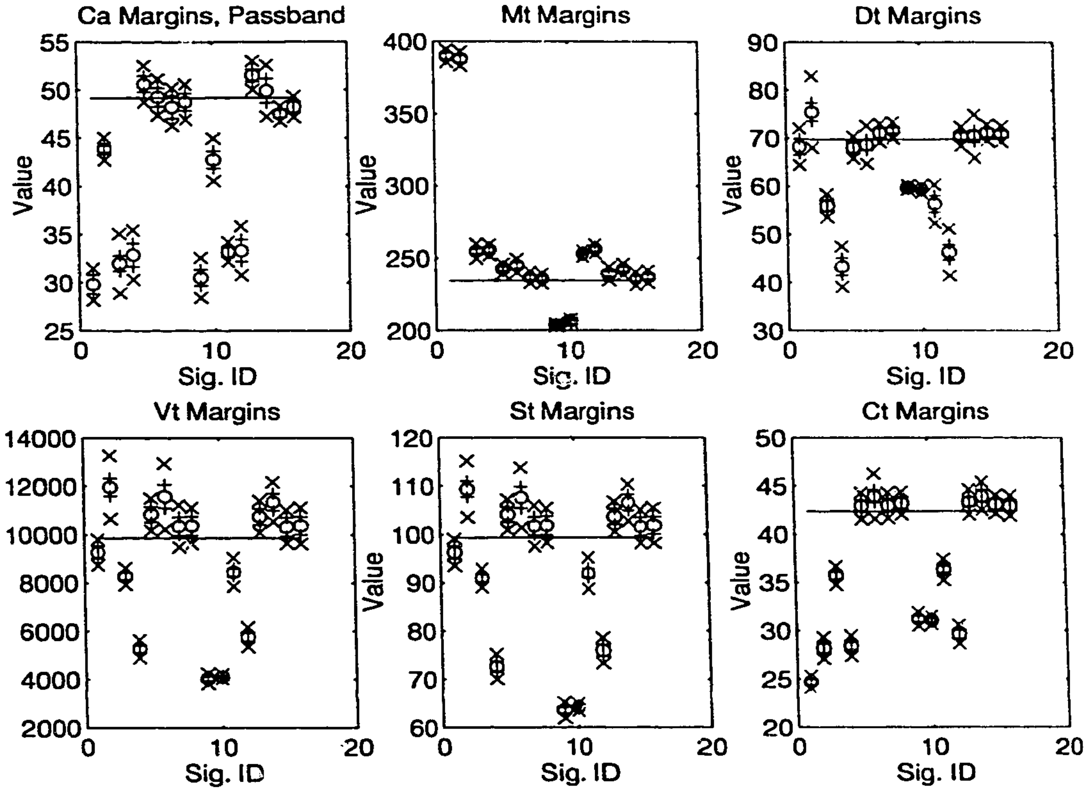


FIGURE 45. Results from SSC analysis of a V.32bis signal.

A weakness of the SSC method is that it does not exploit all of the characteristics of QAM signals if strictly applied to the passband signal representation. Transferring a passband signal to the baseband is undesirable since a carrier frequency must be assumed, and the algorithm complexity is significantly higher. Since QAM is by far the dominant modulation method for data and facsimile transmissions, it would be desirable to base a statistical characterization method upon as many features of QAM as possible. Such features would include the carrier frequency, symbol rate, constellation size, and bandwidth. Another source of features that should be included is the phase information of the baseband QAM signal. Since QAM is both amplitude and phase-modulated, half of the signal content is included in the phase changes of the signal, and this information should be exploited to a greater extent than is possible by the SSC method.

5.2 Sniffer for FAX Detection

"Sniffer" is the name we gave to a small program that can be used to detect the presence of FSK signaling in FAX communications, and recover important handshaking information. This DSP program has the ability to incoherently detect FSK signals, decode their content, and search for control message bit sequences that indicate the type of call in progress. This method is very powerful and simple enough to be run on a single DSP for monitoring several channels because of the low complexity of the algorithm. A complete V.21 modem can be implemented by an 18 MHz fixed-point DSP using public domain programs available from the manufacturer, Texas Instruments. The drawback to the method proposed here is that the signaling portion of a facsimile call must be observed, implying that full-time monitoring of a channel must be performed.

5.2.1 FSK Demodulation

Several methods of demodulating FSK signals are possible. The code for the FSK demodulation stage used in this thesis was developed from sample code distributed free-of-charge by Texas Instruments (TI). The TI code was intended to be used as a V.21 modem implemented on the TMS320C17 TI fixed-point DSP. We only implemented the portion used for demodulating channel two (F_c = 1750 Hz) since that is all that is used by Group III FAX machines. Fig. 46 shows the block diagram of the FSK demodulator.

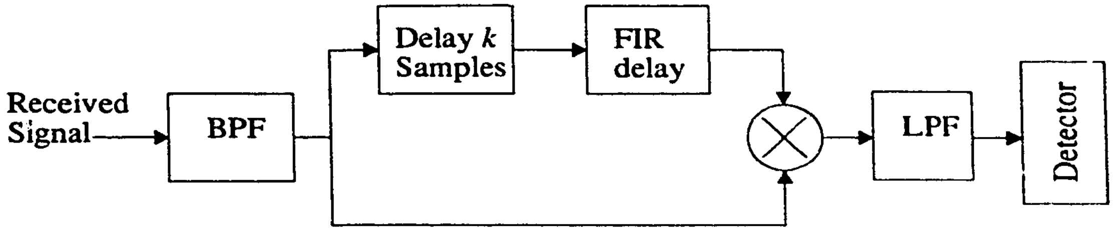


FIGURE 46. FSK demodulator (based on TI sample files).

The initial BPF stage removes the portion of the received signal outside the individual FSK communication channels. In the case of FAX FSK signals, only one channel of the V.21 standard is in use, therefore the BPF stage can be neglected. Then, the filtered signal is multiplied by a time-delayed version of itself. The time delay is implemented in two stages. First the signal is delayed by an integer number k of sample periods. This can be simply implemented by a k stage clocked buffer. Next, a partial sample period delay is implemented by a single pole FIR filter given by the TI software. Finally, the multiplied signal is low-pass filtered to remove unused high-frequency product components. The resulting low-frequency signal is then incoherently demodulated, and is ready for detection. Note that the sample rate of the LPF output is identical to the sample rate of the input.

Now we explain the mathematical theory behind this FSK demodulation method. Let $R(t)$, as shown in equation (49), denote the low-pass filtered signal. Let ω_c be the carrier frequency in radians and $\Delta\omega$ be the relative frequency offset of the modulator. The frequency of the received signal is either ω_c-$\Delta\omega$, indicating a logical zero, or, ω_c+$\Delta\omega$ indicating a logical one. Angle σ is simply an unknown phase offset.

$$R(t) = \cos((\omega_c \pm \Delta\omega)t + \sigma) \tag{EQ 49}$$

Now multiply a delayed version of the received signal, by the original signal to obtain $\hat{R}(t)$, as shown in equation (53).

$$R(t-n) = \cos((\omega_c \pm \Delta\omega)(t-n) + \sigma) \tag{EQ 50}$$

$$R(t)R(t-n) = \hat{R}(t) \tag{EQ 51}$$

$$\hat{R}(t) = \cos((\omega_c \pm \Delta\omega)t + \sigma) \times \cos((\omega_c \pm \Delta\omega)(t-n) + \sigma) \tag{EQ 52}$$

$$\hat{R}(t) = \frac{1}{2}\cos(2(\omega_c \pm \Delta\omega)t - (\omega_c \pm \Delta\omega)n + 2\sigma) + \frac{1}{2}\cos((\omega_c \pm \Delta\omega)n) \tag{EQ 53}$$

If we set $\omega_c = \frac{\pi}{2n}$, low-pass filter $\hat{R}(t)$ to remove the double frequency component, and ignore the 1/2 amplitude component, then we obtain the result:

$$\cos\left(\frac{\pi}{2} \pm \Delta\omega n\right) = -\sin(\pm\Delta\omega n) = \mp\sin(\Delta\omega n) \tag{EQ 54}$$

The final equation equation (54) indicates that the sign of the output of the demodulator after low-pass filtering recovers the originally transmitted symbols. This signal can be detected if the demodulated signal is sampled (sliced) within symbol periods, and not at intersymbol boundaries.

5.2.2 Detection

The output from the demodulator is a 8000 sample per second signal that is positive or negative depending upon the symbol being sent. The next stage is to perform detection where the 8000 sample/second signal is converted into a 300 symbol/second stream. This can be easily done in an incoherent way with a simple sign detector and an incoherent slicer. (Slicing is similar to decimation. The 8000 sample/second signal is effectively sampled at 300 samples/second.) The output of the sign detector is sliced at a constant rate of 300 Hz, which is the nominal symbol rate. In order to ensure that slicing is not inadvertently performed at symbol boundaries (where the symbol is changing) we can count the number of consecutive positive and negative demodulator outputs that are received. If the original input stream is 8000 samples/second, then 26.667 samples correspond to one baud. If the input stream was interpolated to form a 9600 sample/second signal, then 32 samples would form a single baud interval. The middle of a baud should be used as the slicing point. For short duration signals, it is not necessary to reconstruct the precise timing clock.

5.2.3 HDLC Decoder

ITU Recommendation T.30 [41] specifies the signaling format used in all standard Group 3 FAX transmissions. The 300 bps FSK signals contain HDLC coded bitstreams. An eight-bit HDLC [10] flag sequence, {0111 1110}, is used to denote both the beginning and end of a frame. This flag provides a method for gaining bit and frame synchronization at the receiving end.

The flag sequence is transmitted for at least one second before address and data fields are sent. In the general switched telephone network, the address field is always {1111 1111}. The control field format is {1100 X000}, where X is a control bit that is set to 0 for all non-final frames and 1 for a final frame. In all cases, the useful information indicating the type of facsimile transmission is in the last frame of a series, where $X = 1$.

Digital Identification Signal (DIS) or Digital Confirmation Signal (DCS) identifiers appear in the information portion of a frame. The DIS identifier sequence is {0000

0001 } and the DCS identifier is {1100 0001 }. Following the DIS or DCS code is a 72-bit sequence containing all of the information about the type of FAX transmission being established. The DIS information indicates the caller's capabilities, and the DCS information indicates the called station's confirmation of what capabilities it is compatible with. Thus, by decoding this stream of bits, we obtain all of the information necessary for identifying a FAX transmission. The most important bits within the 72-bit sequence, are numbers 11, 12, 13, and 14, which indicate the modulation standard and signalling rate. Table 13 lists the possible signalling rates defined in ITU Recommendation T.30 [41].

TABLE 13. DCS and DIS signalling rate identifiers.

Bit 11	Bit 12	Bit 13	Bit 14	DIS signalling rate	DCS signalling rate
0	0	0	0	V.27ter fall back mode	2400 bps, V.27ter
0	1	0	0	V.27ter	4800 bps, V.27ter
1	0	0	0	V.29	9600 bps, V.29
1	1	0	0	V.27ter and V.29	7200 bps, V.29
0	0	1	0	Not used	14400 bps, V.33
0	1	1	0	Reserved	12000 bps, V.33
1	0	1	0	Not used	Reserved
1	1	1	0	V.27ter, V.29, and V.33	Reserved
0	0	0	1	Not used	14400 bps, V.17
0	1	0	1	Reserved	12000 bps, V.17
1	0	0	1	Not used	9600 bps, V.17
1	1	0	1	V.27ter, V.29, V.33 and V.17	7200 bps, V.17
0	0	1	1	Not used	Reserved
0	1	1	1	Reserved	Reserved
1	0	1	1	Not used	Reserved
1	1	1	1	Reserved	Reserved

It is reasonable to expect an update to the T.30 recommendation which will contain DIS and DCS codes to allow signalling rates of up to 28.8 kbps using a derivative of the V.34 standard. Also note that V.33 communications are allowed by the T.30 standard. V.33

is a 4-wire leased line modem standard that was replaced by V.17 for FAX use. V.17 is dedicated to FAX use.

5.2.4 MATLAB Implementation of Algorithm

Two MATLAB functions were created to implement the "sniffer" facsimile detector. One function named "fskdemodTI", implements the FSK demodulation. The other function, named "sniffer", implements the detection, HDLC decoding, and keyword searching. The MATLAB code for both of these functions is given in Appendix K.

5.2.5 Results

To test the facsimile detector, all of the recorded FAX calls were used as input to the functions. As expected, the sniffer worked well and completely recovered the DIS and DCS information for every call. The FSK demodulator used very few computational cycles. However, the HDLC decoding algorithm seems to be somewhat inefficient since it took many more cycles to execute than demodulation and detection. The HDLC algorithm should be streamlined in future work.

Finally we note that the DCS information is typically transmitted more than once per call because it is sent between each page. The reason for this is that the receiving facsimile station may indicate that a received page was in error, and the sending facsimile station can thereby use a lower speed modulation protocol for the next transmission. This will mean that the sniffer will usually have more than one chance at detecting the DCS sequence per call.

5.3 Autocorrelation Lag Analysis

We now describe modifications to Benvenuto's algorithm and propose a larger, more effective set of discriminant variables. Generally the modifications are to: (a) perform full-wave rectification rather than complex demodulation; (b) use an unbiased estimate of the ACS; (c) determine an optimal subset of the ACS lags to compute; and (d) estimate the information capacity of the signal under test using the ACS. In addition, we fully analyze the discriminant variables and then develop near-optimal methods for performing classification using these variables.

5.3.1 Discriminant Variables

5.3.1.1 Full-wave rectification and the normalized central second-order moment

As mentioned earlier, complex demodulation requires a complex multiplication stage followed by a low-pass filter. This can be configured as a Weaver demodulator [19]. (A Weaver demodulator has a complex multiplication stage followed by two low-pass filters as shown in Fig. 47.) The main benefits of performing this operation are that the resulting signal is conveniently in the baseband, and the amplitude of the complex envelope of an FSK or DPSK signal is constant. However, since these signals are rarely used by modern modems, and when using a modern DSP it is just as easy to operate on the passband signal as the baseband signal, we chose to find an alternative to complex demodulation. If the normalized central second-order moment discriminant is applied to the full-wave rectified signal, the mean value is non-zero and the useful information of the discriminant is retained. Also note that since the numerical mean value of the signal under test is zero (unless a DC bias is present), this cannot be used directly for normalizing the central second-order moment. The normalized central second-order moment $\tilde{\eta}_2$ for FSK signals becomes a nearly constant non-zero value from one segment to the next. Also, the modified discriminant has the same structure for QAM and PSK signals as before.

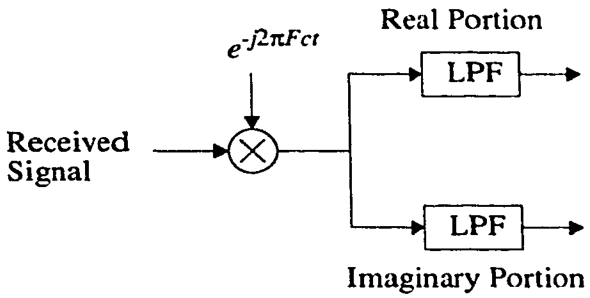


FIGURE 47. Weaver demodulator.

Full-wave rectification is performed on a sampled signal using a DSP by either computing the absolute value of the signal or by stripping off the sign bit of the PCM samples. The equation for $\tilde{\eta}_2$ remains the same, but m_1 and m_2 are now redefined as given in equation (55) and equation (56).

$$m_1 = \frac{1}{N}\sum_{i=1}^{N}\hat{a}(i) \qquad \text{(EQ 55)}$$

$$m_2 = \frac{1}{N}\sum_{i=1}^{N}\hat{a}(i)^2 \qquad \text{(EQ 56)}$$

Here $\hat{a}(i)$ is the real valued, full wave rectified, passband signal segment under test, and N is the segment length.

5.3.1.2 Autocorrelation sequence

The underlying justification for this set of discriminant variables is the differing power spectral densities (PSDs) of the respective signal types. This is illustrated by the plot in Fig. 48. The plot shows the PSDs of speech, V.34, and V.22bis signals versus frequency, normalized to 4 kHz. Clearly these PSDs are quite different. Speech is the signal that can vary the most in its PSD properties. The PSD of speech changes from word to word, speaker to speaker, and mood to mood. However, it is normally centered around 900 Hz and tapers off in the upper frequencies. In fact, the variability of speech power spectra can be used as an aid to classification.

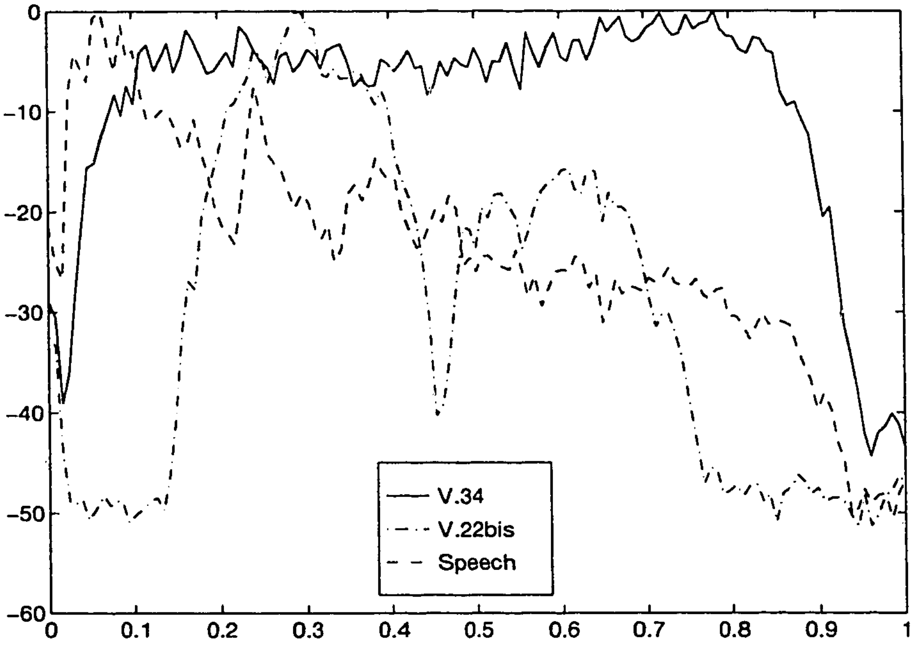


FIGURE 48. Power spectral densities of 3 signals vs. normalized frequency.

The autocorrelation sequence of a signal can be obtained as a linear transformation of the PSD. The opposite is also true. It is even possible to use a subset of lags in an autocorrelation sequence for estimating the PSD. The first 20 lags of the autocorrelation sequences corresponding to the PSDs of Fig. 48 are displayed in Fig. 49. (Recall that the sampling rate is always assumed to by 8000 samples per second, so the lags are 125 μsec apart.) The plot shows that the first 20 lags, out of a possible 2000 in this case, contain much of the information that is present in the PSD. This statement is supported by the fact that the autocorrelation sequences die out very quickly compared to the total length of the sequence. Thus, we can conclude that by using a subset of the initial autocorrelation sequence lags as discriminant variables, we should be able to effectively classify signals that have different PSDs.

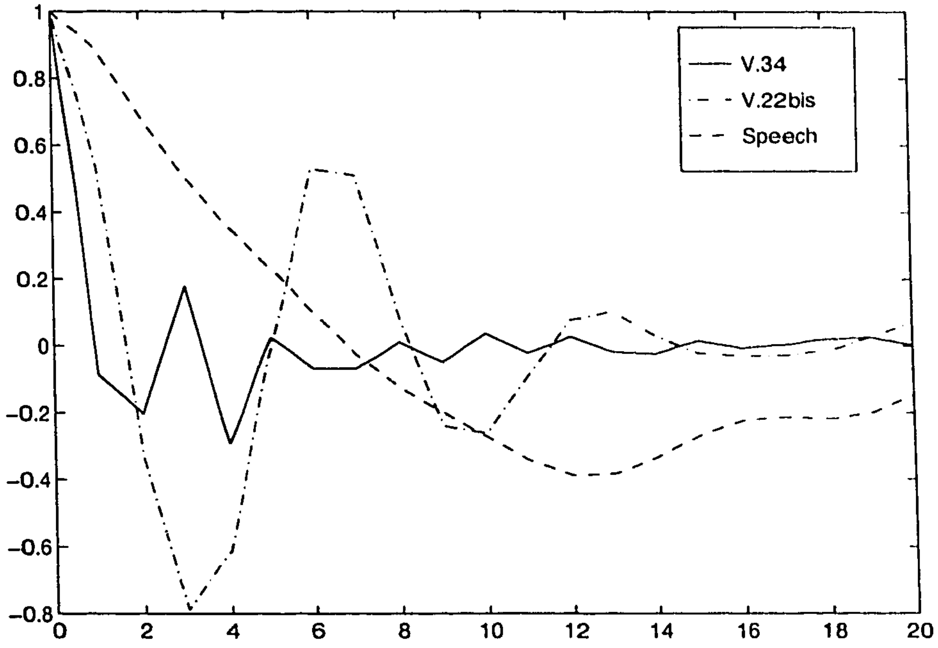


FIGURE 49. Autocorrelations of three signals; (a) Speech; (b) V.22bis; (c) V.34.

Benvenuto [6] proposed that the autocorrelation sequence (ACS) be computed following complex demodulation of the source signal. Since we do not perform complex demodulation, the equation for the ACS estimate must be rewritten as shown in equation equation (57). We also now choose to use an unbiased estimator for the ACS [57]. The difference between this unbiased representation and that given by Benvenuto is that the summation only includes N-$|k|$ points, and is averaged over N-$|k|$. This change can be significant as k approaches N.

$$R_d(k) = \frac{1}{N-|k|}\sum_{i=1}^{N-|k|} d(i+k)\,d(i) \qquad \text{(EQ 57)}$$

Here $d(i)$ is the real valued passband signal segment under test, N is the segment length, and k is the lag number. By computing a real ACS estimate rather than a complex-valued one, we reduce the number of required multiplications by a factor of 2, as well as one addition per sample.

We now discuss the properties of the ACS, its limitations in VB discrimination, and appropriate methods for selecting the most helpful lags to compute. A simple method for selecting the lags is to accumulate statistics from estimates of the ACS of actual test signals. Lags that yield maximal group separation should be chosen. While this method of lag selection is easy to automate, it is not useful if test signals are not readily available. Most VBD signals use some form of QAM, so we will concentrate on that case first.

The general equation for a QAM signal is given by equation (58), where T is the baud period, F_c is the carrier frequency, A_m is the symbol amplitude, and θ_n is the symbol phase. The contribution of the pulse shape filter is represented by $g_T(t)$.

$$u_{mn}(t) = A_m g_T(t)\cos(2\pi F_c t + \theta_n) \qquad \text{(EQ 58)}$$

The baseband signal can be represented by an infinite sum of symbols multiplied by the pulse shape, as in equation (59).

$$v(t) = \sum_{n=-\infty}^{\infty} A_n e^{j\theta_n} g_T(t-nT) \qquad \text{(EQ 59)}$$

Both the ACS and the power spectral density of $v(t)$ can be derived from equation (59). Some lags of interest are given in equation (60) and equation (61). The σ_a^2 term is the variance of the symbol set.

$$R_v(\tau = 0) = \sigma_a^2 \qquad \text{(EQ 60)}$$

$$R_v(\tau = nT) = 0, n = 1, 2, 3, \ldots \qquad \text{(EQ 61)}$$

Here we assume that the bauds are uncorrelated. These equations show that the ACS has periodic zeros with period T for the baseband signal. At the other lags, the ACS is based upon the pulse shaping filter, and any other spectral discriminant variables of the channel.

The passband signal is frequency shifted by F_c from the baseband. Since the ACS can be obtained from the inverse Fourier transform of the power spectral density [62], a frequency shift in the power spectrum translates into a multiplication by a cosine in the ACS. Now, the ACS will have zero crossings with period T as well as $1/F_c$. The full equation for the time average ACS (over one baud interval) of passband QAM is:

$$R_u(\tau) = \frac{1}{2T}\sigma_a^2 R_g(\tau)\cos(2\pi F_c \tau) \qquad \text{(EQ 62)}$$

where $R_g(\tau)$ is the ACS of the pulse $g_T(t)$. The corresponding PSD equation (63) can be obtained by determining the Fourier transform of the ACS, averaged over a baud interval.

$$S_u(f) = \frac{1}{4}\frac{\sigma_a^2}{T}\left(\left|G_T(f - F_c)\right|^2 + \left|G_T(-f - F_c)\right|^2\right) \qquad \text{(EQ 63)}$$

Consider two passband QAM signals that are identical in constellation, baud rate, and spectral shaping, but differ in carrier frequencies. The optimal ACS lags to use as discriminants will be where there is maximal difference between the lags of the two signals, and where the variance of those lags is minimized. (The variance of the estimated lags increases with the lag value, so it is desirable to use low-order lags for discriminant functions.) Precise determination of the optimal lags for discriminating between signals depends upon the baud rate and carrier frequencies of those signals.

Limitations for using ACS lags as discriminants can be seen from the above equations. All of the information present in the PSD also exists in the ACS, just in a different form. Thus, any signals that have different spectra could conceivably be discriminated by the ACSs. However, all information about the QAM constellation is lost when the ACS is computed and normalized. (Note that the time series signal cannot be recovered from the ACS.) Thus we will never be able to use this method to classify signals that have identical pulse shaping filters, carriers, and baud rates, but only differ in their constellation types.

5.3.1.3 Estimated Information Capacity

Shannon showed that the more information carried by a signal, the more statistically random it appears to an observer. Thus, the more information a signal contains, the closer its ACS gets to being simply a spike at the zero-th lag, with all other lags being small and uncorrelated. If the magnitudes of the estimated lags are summed, the result is a discriminant that is an approximate indicator of the reciprocal of the information capacity of the signal under test. This discriminant equation is given by:

$$I = \sum_{k=1}^{K} R_d(k), \qquad \text{(EQ 64)}$$

where K is the number of computed lags, and $R_d(k)$ is the ACS of the passband segment. The value of K gives the number of lags to compute. If only 3 lags are computed, then it only makes sense to use those lags for this estimation. Experimentally, $K=3$ was found to perform well. Signals with less information content generally yield higher I values.

5.3.2 Identifiable Signal Classes

It is critical to identify a representative set of signal classes that we can discriminate using the available discriminant variables. For our application, a critical requirement is to separate speech from non-speech signals. Thus initially we have two main classes, speech and non-speech. Further classification is necessary to separate FAX from VBD signals. Finally, it would be desirable to resolve the different modulation methods used by non-speech signals.

For further subclassification of VBD and FAX signals, seven subclasses were identified after studying the PSDs of different signals. The classes are shown in Table 14. Classes were determined by first plotting representative PSDs of every non-speech signal type that was available, the PSDs of several different speech signals, and the PSD of a random PCM sample stream. By visually comparing each PSD plot we were able to determine which signals had distinctive PSDs. The PSDs of non-speech signals can also be predicted by the ITU recommendations and standards. The resulting classes all have unique power spectra.

TABLE 14. VBD, FAX, and speech subclassification classes.

Group No.	Signals included
1	V.22 and V.22*bis* forward channels
2	V.22 and V.22*bis* reverse channels
3	V.34 at speeds greater than 14.4 kbps
4	V.29 all speeds
5	V.32, V.32*bis*, and V.17 at speeds greater than 2,400 bps
6	V.27*ter* at 4,800 bps
7	V.27*ter* at 2,400 bps

TABLE 14. VBD, FAX, and speech subclassification classes.

Group No.	Signals included
8	speech
9	random PCM samples

These classes combined with a speech class and a random PCM class form a total of nine identifiable classes using only PSD differences as the criteria. Random PCM samples are included as a class to simulate channels that are not being used for voice-band communications, but are carrying direct binary data.

From Table 14 it is clear that a problem exists in class five, since several signal types are lumped together. We cannot discriminate V.17 signals from V.32 and V.32*bis* solely on the basis of their power spectra. V.17 is used for 14.4 kbps FAX communications, while V.32 and V.32*bis* are both data standards. Thus, using this class structure, in some cases we cannot distinguish between FAX and VBD signals. This is an expected limitation however, since V.17 and V.32*bis* are virtually identical standards, except one is half-duplex and the other is full-duplex, respectively. By monitoring both directions of a 4-wire connection it is possible to determine if a call is half or full-duplex by comparing the relative power levels of the two channel directions. This is a method that could be used to discriminate between V.17 and V.32/V.32*bis* signals.

Now that the recorded and simulated signals have been discussed, the discriminant variables have been defined, and the class structure has been given, we can calculate the probability mass functions for each variable and class. This information was compiled and is shown in Appendix L. Each plot in the appendix shows the probability mass function (or histogram in this case) of a single discriminant variable calculated from signals belonging to a particular class. The histograms indicate the shape of the probability distributions as well as the mean values and standard deviations. By examining this graphical representation we can get an idea of which variables have larger mean differences between classes, and will therefore be useful in a linear discriminant function.

5.3.3 Analytic Method for Selecting the Best Lags

As alluded to in the previous section, a method has been developed for determining the "best" lags to use as discriminant variables for separating two specific QAM signals. The method is from the form of the equations for the ACS of QAM signals. There is no reason that the method could not be extended to any signal type, provided it is possible to represent the signal by a compact analytical equation. We will first develop a completely analytical method.

The equation for a transmitted baseband QAM signal is given by equation (65), where the signal *v(t)* is represented as an infinite sum of symbols multiplied by shaped pulses. The symbols have both amplitude and phase components, A_n and $e^{j\theta_n}$, respectively. Since the information sequence $\{A_n e^{j\theta_n}\}$ is random, *v(t)* can be interpreted as a sample function of a random process *V(t)*.

$$v(t) = \sum_{n=-\infty}^{\infty} A_n e^{j\theta_n} g_T(t-nT) \qquad \text{(EQ 65)}$$

The passband representation of QAM signal waveforms has the general form shown in equation (66). The subscript *m* may take on the values $\{1, 2, ..., M_l\}$, and the subscript *n* may take on the values $\{1, 2, ..., M_2\}$. M_l and M_2 are the number of levels in the real and quadrature components, respectively, of the signal.

$$u_{mn}(t) = A_m g_T(t) \cos(2\pi F_c t + \theta_n) \tag{EQ 66}$$

The pulse shaping filter usually has a square-root raised cosine type of impulse response [11]. The equation for the raised cosine is given in equation (67), where α is the roll-off factor that ranges from zero to one. Pulse shaping filters have a frequency response that is the square root of the frequency response of equation (67). When an identical filter is used at the receiver end, inter-symbol interference (ISI) is minimized.

$$g_T(t) = \left(\frac{\sin\left(\pi\frac{t}{T}\right)}{\pi\frac{t}{T}}\right)\left(\frac{\cos\left(\alpha\pi\frac{t}{T}\right)}{1-\left(2\alpha\frac{t}{T}\right)^2}\right) \tag{EQ 67}$$

Next we can develop an analytical representation for the autocorrelation function of a QAM signal. We will operate on the baseband representation of the signal, $v_{mn}(t)$, and then shift the resulting function to the passband. First we find the mean value function $E[V(t)]$ of the random process $V(t)$ for which $v(t)$ is a sample function. That is, $v(t)$ is a single possible outcome from the random process $V(t)$. Note that the particular mean value function that we have in mind is a function of time, and is the average of all symbols combined with the pulse shaping filter. (i.e. The "mean pulse" is intended, not the simple mean of the signal.) The first line of equation (68) states that the expected value function of $V(t)$ is the weighted ensemble average of all possible outcome functions $v(t)$. The second line of equation (68) uses the probability mass function $P[]$ to represent the probability that a given symbol will equal symbol i.

$$E[V(t)] = \sum_{v} v(t) P[v(t)] \qquad \text{(EQ 68)}$$

$$= \sum_{(i=1)}^{M} \sum_{(n=-\infty)}^{\infty} P\left[A_n e^{j\theta_n} = i\right] g_T(t-nT)$$

$$= \sum_{n=-\infty}^{\infty} E\left[A_n e^{j\theta_n}\right] g_T(t-nT)$$

Next we use the equation for the expected value to form the general equation of the autocorrelation $R_V(\tau)$ of the random process $V(t)$. The autocorrelation of continuous time domain signal at a given point in time t is a function of τ, where τ is the distance from the middle of the autocorrelation.

$$R_V(t+\tau, t) = E[V(t) V(t+\tau)] \qquad \text{(EQ 69)}$$

$$= \sum_{(n=-\infty)}^{\infty} \sum_{(l=-\infty)}^{\infty} E\left(A_n e^{j\theta_n} A_l e^{j\theta_l}\right) g_T(t-nT) g_T(t+\tau-lT)$$

We note here that a simplification can be performed by recognizing that the autocorrelation of the information sequence is given by equation (70) and can be substituted into equation (69) to obtain equation (71).

$$R_a(n) = E\left[A_m e^{j\theta_m} A_{n+m} e^{j\theta_{n+m}}\right] \qquad \text{(EQ 70)}$$

$$R_V(t+\tau, t) = \sum_{(n=-\infty)}^{\infty} \sum_{(l=-\infty)}^{\infty} R_a(l-n) g_T(t-nT) g_T(t+\tau-lT) \qquad \text{(EQ 71)}$$

By rearranging the summation indices, we can further simplify as shown in equation (72).

$$R_V(t+\tau, t) = \sum_{(m=-\infty)}^{\infty} R_a(m) \sum_{(n=-\infty)}^{\infty} g_T(t-nT) g_T(t+\tau-nT-mT) \qquad \text{(EQ 72)}$$

At this point we have determined the autocorrelation sequence of the random process *V(t)*. Note that the second summation in equation (72) is periodic with period *T*. This means that the overall equation for $R_V(t+\tau, t)$ is also periodic. Recall the equation for the mean value of *V(t)* from equation (68). The mean value is also periodic with period *T*. A random process that has a periodic mean value and periodic autocorrelation is said to be a *cyclostationary process* [62]. To determine the power spectral density (PSD) of a cyclostationary process, we compute the Fourier transform of the autocorrelation function over the period *T*. The time average autocorrelation function of *V(t)* is shown in equation (73) and equation (74).

$$\bar{R}_V(\tau) = \frac{1}{T}\int_{-\frac{T}{2}}^{\frac{T}{2}} R_V(t+\tau, t)\,dt \tag{EQ 73}$$

$$\bar{R}_V(\tau) = \frac{1}{T}\sum_{m=-\infty}^{\infty} R_a(m)R_g(\tau - mT) \tag{EQ 74}$$

It can be shown that by taking the Fourier transform of equation (74) we obtain the PSD given by equation (75) and equation (76).

$$S_V(f) = \int_{-\infty}^{\infty} \bar{R}_V(\tau)e^{-j2\pi f\tau}d\tau = \frac{1}{T}S_a(f)|G_T(f)|^2 \tag{EQ 75}$$

$$S_a(f) = \sum_{m=-\infty}^{\infty} R_a(m)e^{-j2\pi fmT} \tag{EQ 76}$$

Everything derived so far has been for a baseband QAM signal. We can easily derive the results for a passband signal by simply multiplying in the time domain by a sinusoid, or shifting by the carrier frequency in the frequency domain. Equation (77) gives the time average autocorrelation function for the passband QAM signal.

$$R_U(\tau) = \frac{1}{T}\sum_{m=-\infty}^{\infty} R_a(m) R_g(\tau - mT)\cos(2\pi f_c t) \tag{EQ 77}$$

An important special case exists for QAM. If the information sequence contains symbols that are uncorrelated and have zero mean, the autocorrelation in equation (77) simplifies to the form shown in equation (79).

$$R_a(n = 0) = \sigma_A^2, R_a(n \neq 0) = 0 \tag{EQ 78}$$

$$R_U(\tau) = \frac{1}{T}\sigma_a^2 R_g(\tau)\cos(2\pi F_c\tau) \tag{EQ 79}$$

These results indicate some very important discriminant variables of the autocorrelation sequence. First, the autocorrelation is only a linear transform of the PSD. Therefore we can conclude that only signals that differ in their PSD can be classified using their autocorrelations, and visa versa. Second, for the special case of zero mean uncorrelated information sequences, the autocorrelation reduces to a scaled and modulated version of the pulse shape autocorrelation. Therefore we can also conclude that only signals that differ in their carrier frequencies and/or their pulse shapers can be discriminated using only their autocorrelations. Finally, we note that since the received signals must be normalized to unity power, no information from the transmitted symbol sequence is present in the autocorrelation sequence.

Now we move on to determining which lags of the discrete autocorrelation sequence are the "best" to use in a discriminator. Recall that discrimination using a linear discriminant function requires that the mean values of the discriminant variables to be different. If the discriminant variables are based on the autocorrelation sequences, we must therefore maximize the absolute difference between the respective variables of the form shown in equation (80). Two different QAM signals are represented by *U1* and *U2*.

$$D_R(\tau) = |R_{U1}(\tau) - R_{U2}(\tau)| \qquad \text{(EQ 80)}$$

To locate the local maxima in the equation given by equation (80) we can compute the first derivative and solve for the roots. This was attempted, however the actual equations that result from the derivative operation were too large to manage manually, and the roots could not be solved for using the available mathematical software. Thus we concluded that the most efficient method of finding a solution is to perform the computations numerically.

Consider the following example. Two types of signals that are particularly difficult to discriminate are V.29 and V.32bis. These are both QAM signals with the same baud rate F_s=2400 Hz. The only spectral difference between the signals is that their carriers differ by 100 Hz. The carriers are 1700 Hz and 1800 Hz for V.29 and V.32bis, respectively. This difference is difficult to resolve effectively with a linear discriminator since the mean values of the autocorrelation functions will be nearly identical for both signal types. Therefore, we would like to identify the autocorrelation lags that are most valuable for discriminating these two signal types. We can use the equations given above for the autocorrelations, and numerically solve for the lags that lead to the largest mean value difference.

Before proceeding we note that noise is not considered in these calculations. If we consider the noise to be additive Gaussian then the only effect this will have on the results is to increase the variance of the lag values. The mean values of the lags will remain unaffected. However, if the noise is not Gaussian, it will affect the mean values of the lags, and hence the best lags to use as discriminant variables may change. The autocorrelation at lower lag values tends to have lower variances, hence we should try to choose the lowest lag values possible. We cannot directly show the variance in the autocorrelation function since noise was not included in this analysis.

First we plot the autocorrelation sequence for a V.29 signal in Fig. 50. The autocorrelation is represented by $R_{ul}(\tau)$, with F_c=1700 Hz, F_s=2400 Hz, and a typical roll-off value of α=0.15.

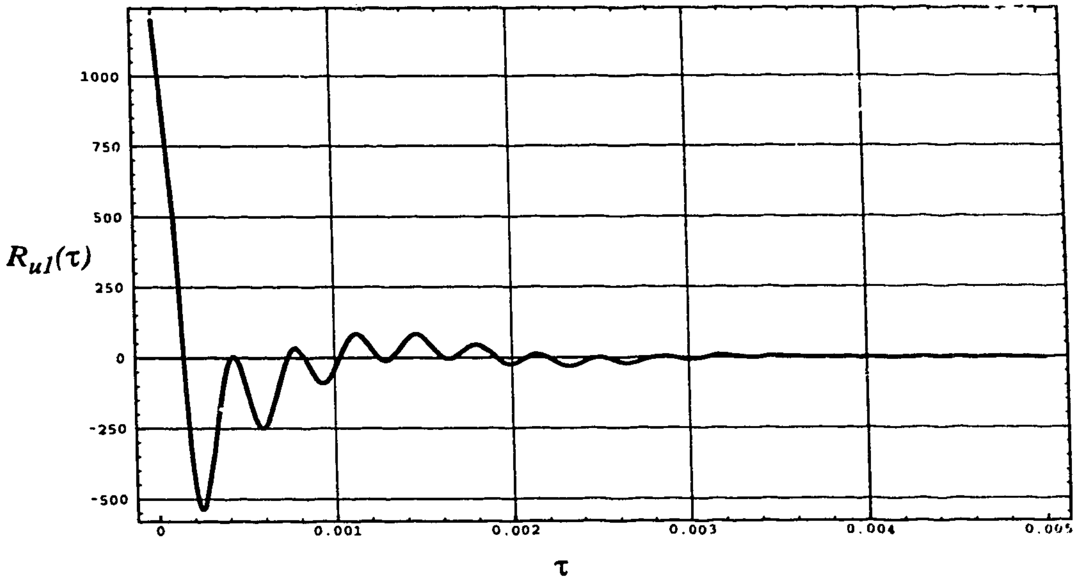


FIGURE 50. Plot of R_{ul} versus τ.

Next we plot the autocorrelation of a V.32bis signal in Fig. 51. All parameters are identical here with the exception of F_c, which is now 1800 Hz. Note how similar the shape of this plot is compared to that of Fig. 50.

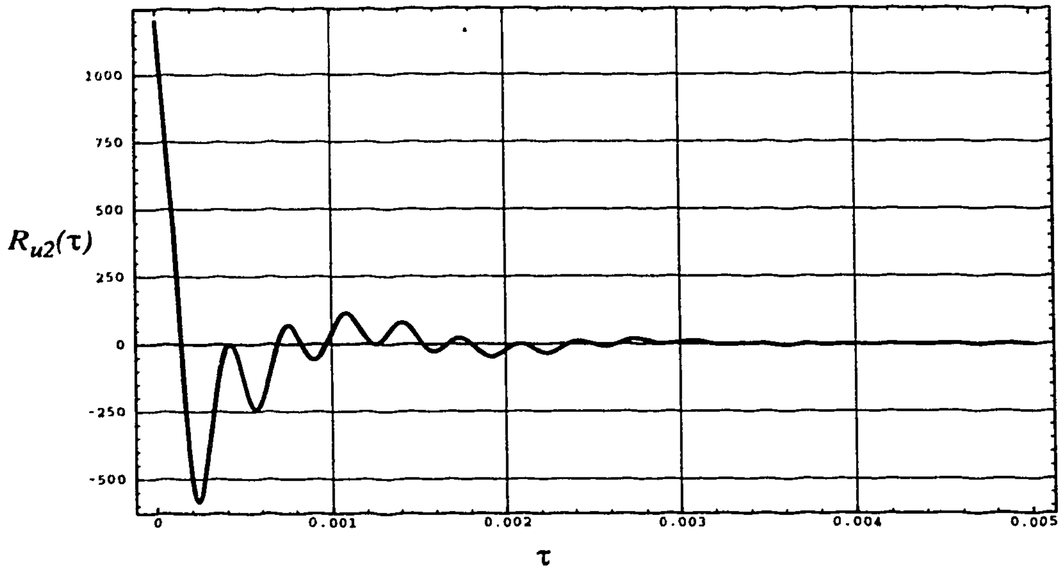


FIGURE 51. Plot of R_{u2} versus τ.

In Fig. 52 we plot the difference between the two autocorrelations. This plot represents the mean distance between each autocorrelation value for the two signals. Note that up to this point we are still dealing with continuous time representations of the signals. The local maxima and minima identify the largest mean differences between the autocorrelations.

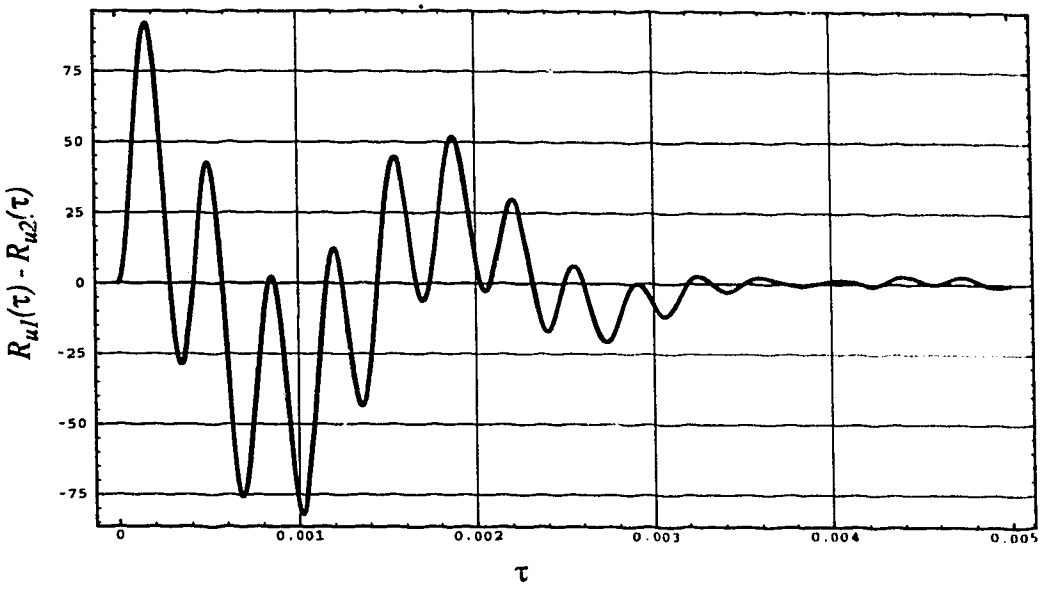


FIGURE 52. Plot of the difference between R_{ul} and R_{u2} vs. τ.

To locate all of the local maxima and minima we compute the first derivative of the autocorrelation difference and solve for the roots. Figure 53 shows a plot of the derivative.

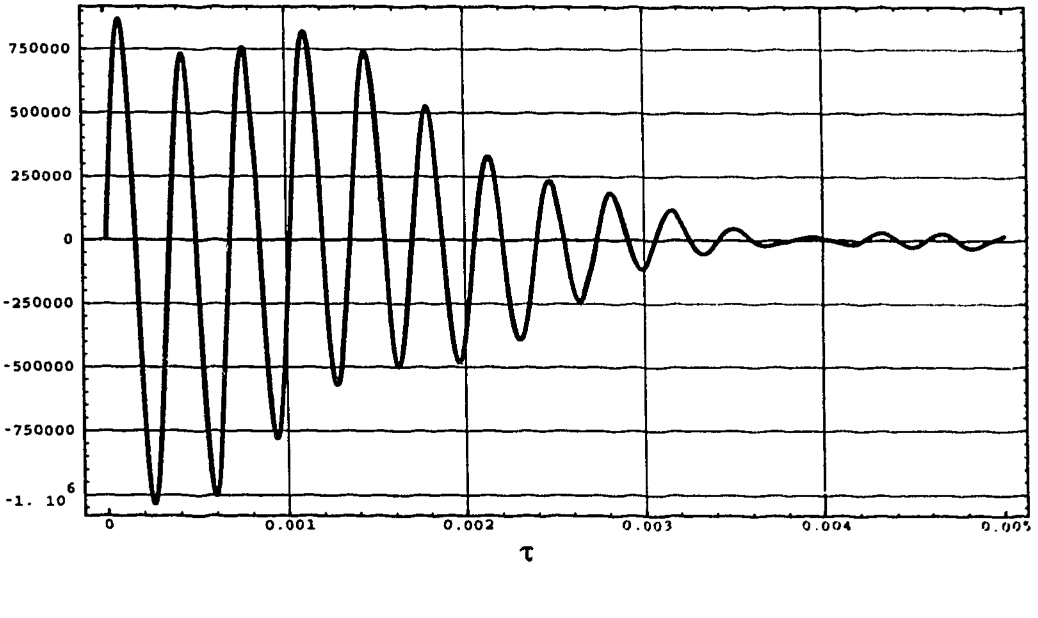


FIGURE 53. Plot of the derivative of the difference between R_{ul} and R_{u2} versus τ.

Finally we can convert the results to the discrete time domain and determine which !ags are the most useful for discriminating V.29 from V.32*bis*. The sampling rate is taken to be 8000 samples per second. This also sets the period between autocorrelation lags. Fig. 55 shows the absolute value of the differences between the discrete time autocorrelation sequences of the two signals. Where the difference value is high, we can conclude that the autocorrelation lag has a relatively large mean value difference for the two signal types. From the plot we can conclude that lags k=1 and k=8 are the most valuable for discriminating V.29 signals from V.32bis signals given that the channels have only additive Gaussian noise, the roll-offs of the pulse-shapers are equal with α=0.15, and a linear discriminator is used.

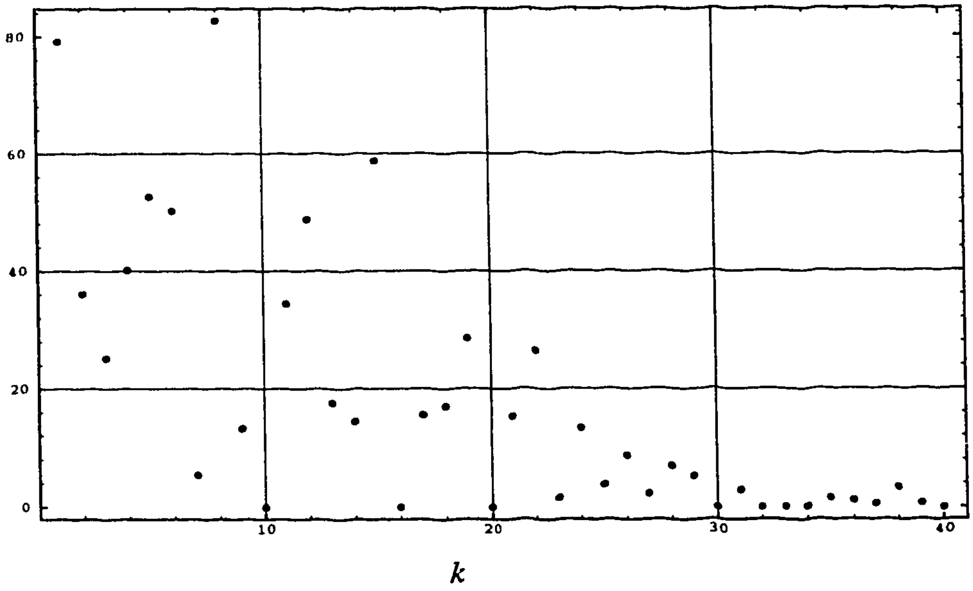


FIGURE 54. Plot of the autocorrelation differences versus lag k for $\alpha_1=\alpha_2=0.15$.

As an experiment, we repeated the same analysis using a slightly different roll-off factor. For this experiment α_1=0.2. Fig. 55 is a plot showing the results. From the figure we can state that the best two lags are still k=1 and k=8. It is also evident that the roll-off factors do play a significant role in the choice of best lag. Lags four, five, and six may be very valuable since they do have a substantial mean value difference, and the variance for smaller lags is also larger.

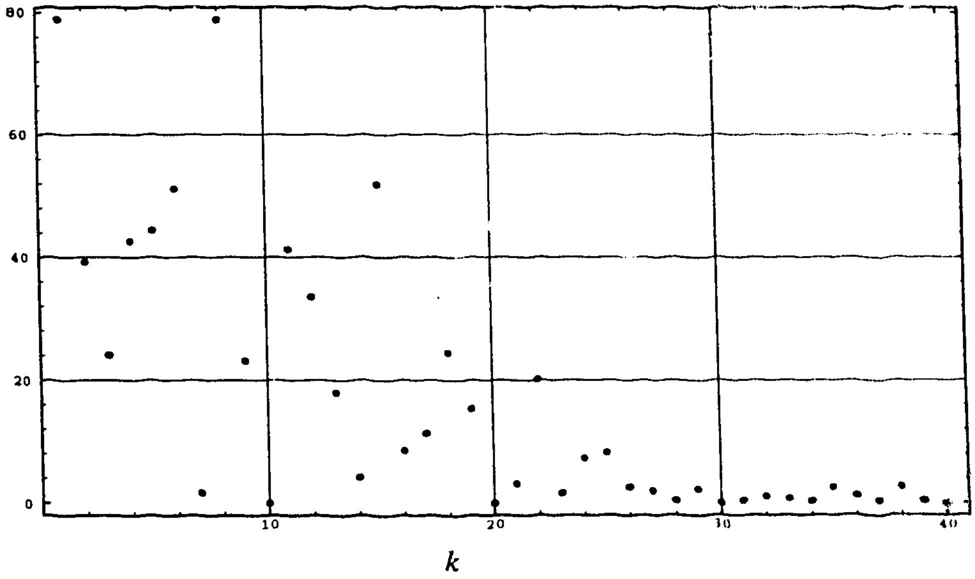


FIGURE 55. Autocorrelation differences vs. lag k (α_1=0.2 and α_2=0.15).

From the complicated analysis presented in this section we concluded that completely analytical techniques for lag selection are not practical. At some point numerical estimates must be used since the required equations are too complex for manual or automated solution. The method for selecting the best lags that was presented here is highly dependant upon the signal parameters input into the equations. Actual channels do not have simple additive Gaussian noise, therefore we have not included a noise model in this analysis. It should be noted that any impairment which distorts the channel frequency response also affects best lag selection. However, we have shown a systematic method for determining the relative importance of autocorrelation lags as discriminant variables that can be applied to arbitrary QAM signals. In the following sections we will empirically determine the best lags to use as linear discriminant functions and then compare the results of the two methods.

5.3.4 Discriminant Variable Selection using Empirical Methods

Another method for selecting discriminant variables is to use empirical observations of the values of each of the variables for a representative population. This method is only useful if it is possible to obtain a sufficiently large population for classification, and if the sample population is representative of the actual population that is to be classified. This method of variable selection will be used later in this chapter and compared to the results obtained from the analytic method.

5.3.5 Discriminant Functions and Classification Methods

Two forms of discriminant functions, linear and quadratic, are discussed next. Some other heuristic classification methods are also examined.

5.3.5.1 Simple Decision Boundaries

The simplest method of performing classification is to study a scatter plot and then draw simple decision boundaries as appropriate. If an observed instance of a discriminant variable falls in a particular zone, classify it accordingly. This is the method used by Benvenuto for the case of two variables and two classes. The method works well when the discriminant variables have substantially different mean values for each group, and the standard deviations are small compared to the distance between means. This method is illustrated well in Fig. 56. Here it is easy to see where to draw decision boundaries.

This method is only appropriate for simple cases however. Consider the problem of simply observing a multivariate system with 10 or more discriminant variables and 9 or so classes. Most people are not capable of conceptualizing a 10-dimensional space. Th. fore, it is desirable to develop automated methods for developing decision boundaries. Such methods have been well researched, and are readily available in off-the-shelf software such as SPSS.

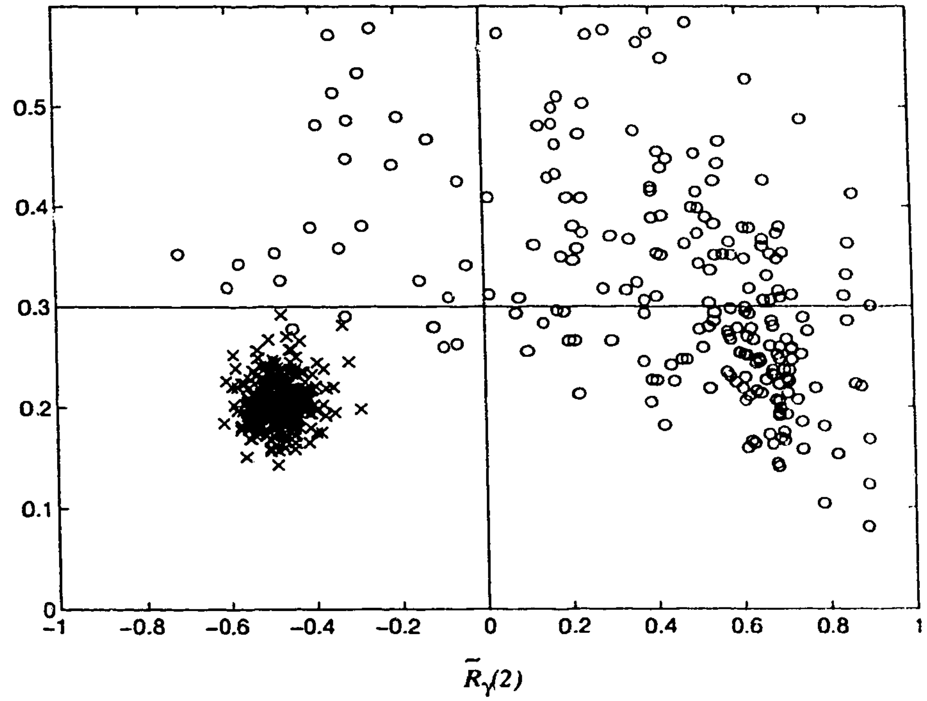


FIGURE 56. Scatter plot for one VBD signal and one speech signal.

5.3.5.2 Linear Discriminant Function

Linear discriminant functions were discussed earlier. The method is based upon forming a linear combination of the discriminant variables and using the resultant values for classificatior. The classif.:atiuñ could be performed using a technique such as Bayes' Theorem.

Linear discriminant functions can be proven optimal if the assumptions of unequal mean values and equal covariance matrices can be met [69]. To test these assumptions, actual discriminant variable measurements were input into SPSS for analysis. The results are discussed later.

5.3.5.3 Quadratic Discriminant Function

Quadratic discriminant functions are optimal if the mean values of the discriminant variables are equal and the covariance matrices are different [69]. If the mean values are also different, a combined linear and quadratic form should be used. To test these assumptions, actual discriminant variable measurements were again input into SPSS for analysis.

5.3.6 MATLAB Algorithm Implementation

MATLAB code was developed to extract discriminant variable measurements. The code computes the first ten lags of the ACS, as well as the central second-order moment. Adjustable parameters include the signal segment length (*N*), sub-segment length (*L*), and silence threshold (P_{th}). The results from the computations were stored in a format that is readable by both SPSS and MATLAB. Refer to Appendix C for the code that was used to generate the discriminant variables.

In addition to basic discriminant variable computation, MATLAB code was developed for performing classification using Bayes' Theorem, automatically learning the discriminant variable distributions, and performing cross-validation. Results from these tools indicated that the variables being considered were very powerful, and correct classification rates were going to be near 100% for most classes. However, due to the fact that the code may contain inaccuracies, we choose to present results obtained from SPSS in performing discriminant analysis. Results from the SPSS analysis and the developed classification code were similar, but more statistical information is available output from the SPSS implementation.

5.3.6.1 Results from Benvenuto's Algorithm

Benvenuto's algorithm in [6] was implemented with slight corrections to the loop counters and logical relations due to errors discovered in the pseudo-code. This allowed for confirmation of Benvenuto's findings, as well as providing a comparison point for other algorithms. The parameters used were identical to those in [6], namely N=256, L=16, g_{Th}=33, P_{Th}=1089, with thresholds for $\tilde{\eta}_2$ and $\tilde{R}_{\gamma}(2)$ at 0.3 and 0, respectively. Fig. 56 shows typical scatter plots for a V.32*bis* signal and a speech signal, indicated by 'x' and 'o' marks, respectively.

The plot shows the classification boundaries as solid lines. All points that exist in the lower left quadrant are classified as VBD, and the points in the other three quadrants are said to be speech. The misclassified points for the speech signal can be easily seen in the lower left quadrant. There were also some circles that lay outside the range of the plot.

Benvenuto's algorithm was executed on all of the available signal recordings. It was found that the algorithm yielded a 0.12% misclassification rate for VBD signals and a 2.32% misclassification rate for speech signals. These results were obtained using the same parameters mentioned above. Misclassification performance can be improved by using a more sophisticated classification technique, such as Bayesian allocation [21], as opposed to the simple decision boundaries in Fig. 56.

We found that using the thresholds given, on average 70% of the speech signal samples and 0% of the VBD signal samples were rejected as silence. Adjusting the thresholds can reduce the silence rate for speech signals at the expense of increasing the misclassification rate since the classifier may attempt to operate on noisy segments.

While this algorithm performs well, it is more computationally intensive than desired and it also is unable to effectively resolve many data signal types. We proceeded from this starting point hoping that algorithm enhancements would yield lower misclassification rates, simpler computational complexity, and further subclassification of VBD signals.

5.3.6.2 Results from modified Benvenuto Algorithm

The modified algorithm, with no demodulation stage, was implemented and tested on all of the available signal recordings [67]. In the implementation it is possible to control the number of ACS lags to compute and use for classification. As expected, with more lags included, greater classifier accuracy could be obtained. Eventually, however, adding more lags to the classifier did not improve performance since there is too much variance introduced to provide for improved group separation.

For comparison, scatter plots are shown in Fig. 57 that correspond to those of Fig. 56, except now the modified algorithm results are shown. All parameters and signals are identical.

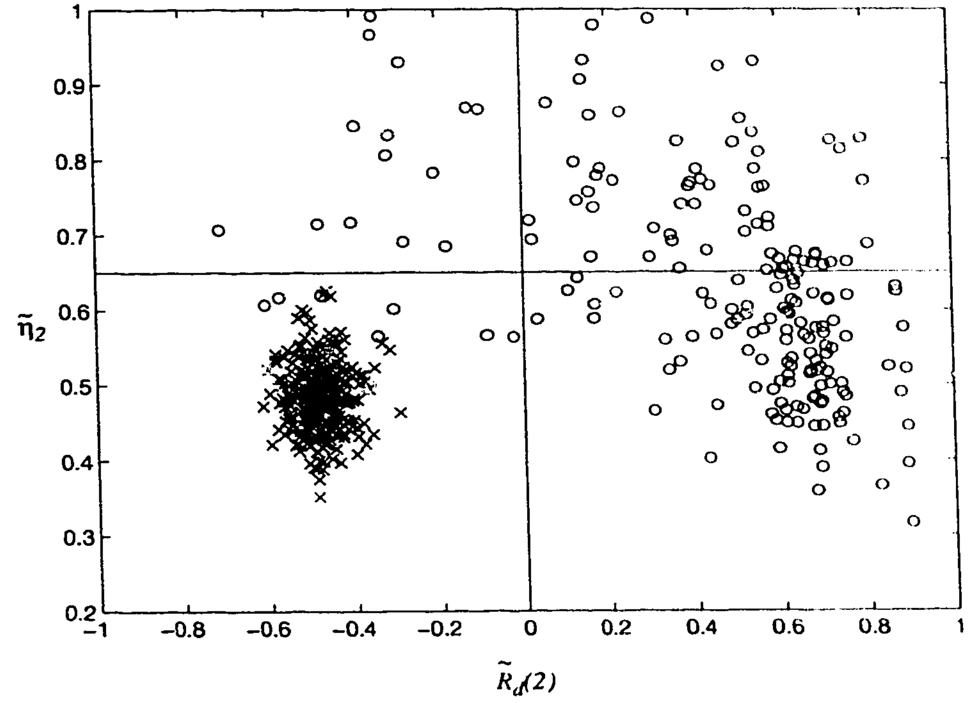


FIGURE 57. Scatter plot for one VBD signal and one speech signal.

Using only the two discriminant functions shown, VBD versus speech classification can still be performed with misclassification rates of 0.04% and 2.61%, respectively, with the decision boundaries were set at $\tilde{\eta}_2$ =0.65 and $\tilde{R}_d(2)$ =0. The advantage of the modified algorithm is that fewer computations need be performed since a Weaver demodulator is replaced by full-wave rectification. The degree of this advantage depends upon the type of filtering stage that is used. However, it is clear that complex demodulation requires at least (a) one complex multiply per sample; (b) two multiplies and an addition per sample to compute each ACS lag; and (c) the vector magnitude at each sample is needed to compute the central second-order moment. The modified algorithm reduces this to one multiply per sample to compute each ACS lag; eliminates the need to find the vector magnitudes at each sample to compute the central second-order moment; and eliminates complex demodulation and filtering. In our implementation this translated into a 10-fold reduction in floating point operations (using a 9th order low-pass Butterworth filter in Benvenuto's original algorithm).

Bayesian allocation was also performed on the discriminant results. Using this technique, rather than simple hard decision boundaries, reduced the misclassification rates for VBD and speech to 0.10% and 2.13% respectively. Again, the same parameters were used. To further reduce misclassification, however, more discriminants and/or larger values of *N* must be used.

The main advantage of the modified algorithm lies in its improved ability to subclassify VBD signals. The number and selection of lags is variable depending upon the classification accuracy and class resolution desired. By studying the probability mass functions of each lag as it is applied to all of the signals, we were able to determine which lags contribute the most to the classification accuracy. This can be done using standard discriminant analysis techniques [4, 54]. As each lag is added to the classifier, its effect can be evaluated, or the joint contributions of combinations of discriminants can be also evaluated.

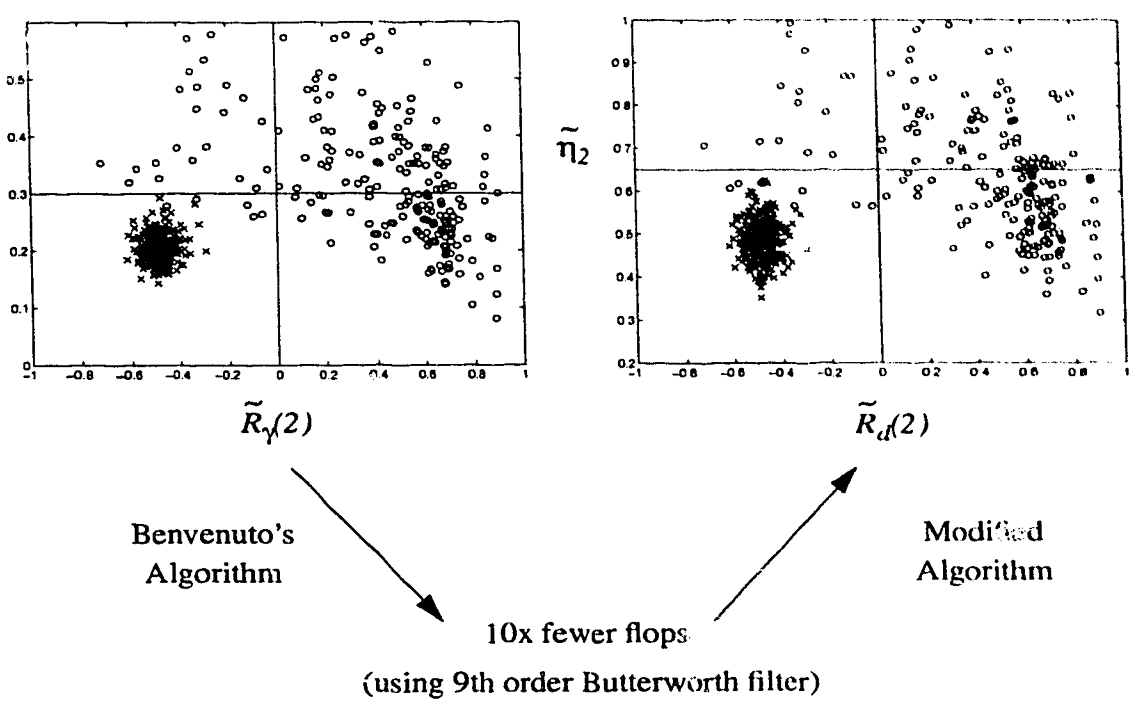


FIGURE 58. Scatter plots for Benvenuto's algorithm and the modified algorithm.

Fig. 59(a) and Fig. 59(b) illustrate the advantage of adding just one lag to the classifier. Fig. 59(a) shows the scatter plot of two VBD signals which are V.34 and V.22*bis*, which are denoted by 'x' and 'o' marks, respectively. From the plot it is obvious that there is no way to reliably separate the two signal types using only the discriminants shown, even though the signals are vastly different. Fig. 59(b) shows the scatter plot of the same two VBD signals, with $\tilde{R}_d(1)$ used instead of $\tilde{\eta}_2$. This plot clearly shows that the two classes can now be easily resolved.

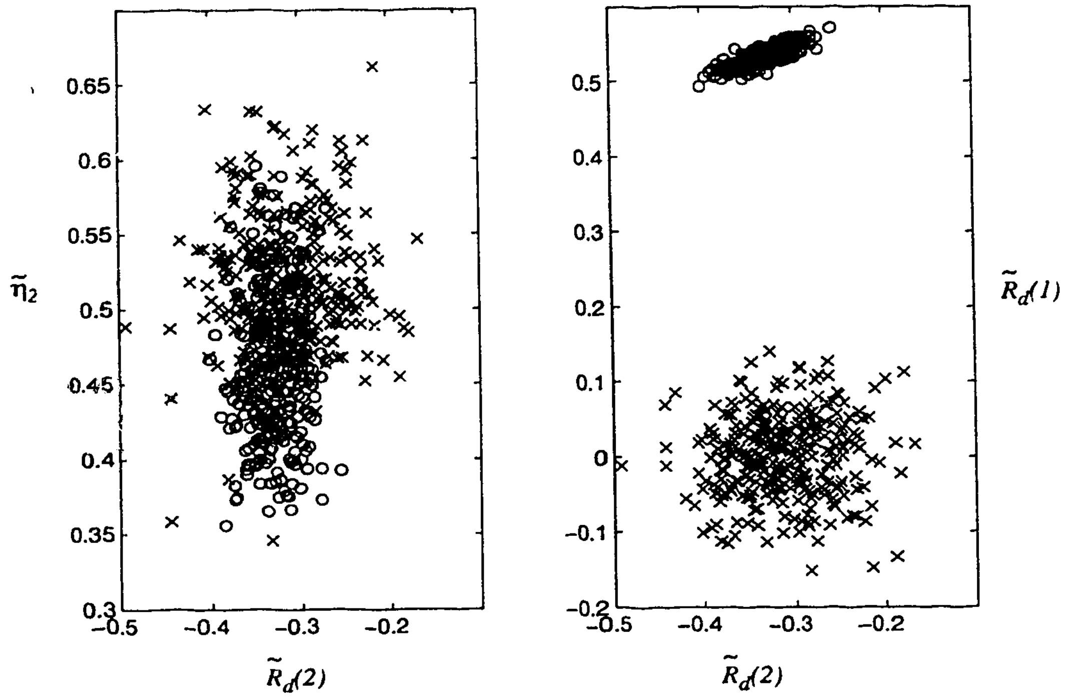


FIGURE 59. Scatter plots for two VBD signals with different variables.

A new algorithm was developed using $\tilde{\eta}_2$, $\tilde{R}_d(1)$, $\tilde{R}_d(2)$, and $\tilde{R}_d(3)$ as discriminants and using the same algorithm parameters as above. Bayesian allocation was used to assign signal segments to particular classes. The misclassification rates were 0%, 0.16%, 2.89%, 19.33%, 45.21%, 1.29%, and 1.07% for classes one through seven, respectively, 0.07% for VBD segments, and 0.44% for speech segments. Incidentally, speech was misclassified as class three (V.34) in all cases.

In these experiments all classes were accurately classified except for classes 4 (V.29) and 5 (V.32, V.32*bis*, V.17). These classes are very similar and vary mainly in their carrier frequencies, so the classifier could not separate the two signal types in a short span of time with the given discriminants. To resolve these two classes, it was necessary to study the equations for the ACS of each signal and determine the optimal lags to monitor. These were found to be lags k=1 and k=8. Lag eight was therefore added for further experiments. Using this lag in conjunction with lags at k=*1*, *2*, and *3*, we found that all seven VBD classes could be effectively classified. The misclassification rates (for 32 ms. segments) for this experiment were 0%, 0.16%, 3.12%, 17.10%, 31.75%, 0.96%, and 0.75% for classes one through seven, respectively, 0.03% for VBD segments, and 0.25% for speech segments. Thus, lag eight helped resolve the classes, but not adequately.

To further reduce the misclassification rates, longer signal segments and more lags must be used. For example, using N=1024, the misclassification rates are 0%, 0%, 0.19%, 2.97%, 13.9%, 0%, and 0% for classes 1 through 7, respectively, 0% for VBD, and 0% for speech signals. Using N=2048 yields near 0% misclassification for all classes except four and five, which were 1.2% and 2.49% respectively. Increasing the segment length beyond 2048 samples does not improve classifier performance. A good trade-off between segment length and classification performance is at N=1024. Using the first ten lags and N=1024, all classes have 0% misclassification rates except classes four and five which are 2.65% and 7.79%, respectively. In the following sections we present a more systematic approach to determining the design parameters of the classification system.

5.3.7 SPSS Discriminant Analysis

In the following discussion, the discriminant variables used for study are based on computations using N=2048, and L=16. All of the recorded and simulated signals were considered in the analysis. Other values for N are considered later in order to study the trade-off between rapid response times (N is small), and high classification accuracy (N is large). The discriminant variables that were calculated for the experiments include the first

10 normalized ACS lags and the normalized central second-order moment. We attempted to distinguish all nine classes of signals discussed earlier. The prior group allocations of each case are known and the prior probabilities are all assumed to be equal.

TABLE 15. Number of cases considered (*N*=2048).

Class	No. Cases
1	2393
2	1424
3	2348
4	1879
5	4387
6	1183
7	970
8	810
9	974
Total	16368

5.3.7.1 Assumption Validation

First we examine the mean values and covariance matrices of the test data, in a group-wise fashion. These results will indicate whether the assumptions necessary for optimal performance of linear and quadratic discriminant functions can be met. Tab!e 16 shows the group means while Table 17 shows the group standard deviations. Note that the standard deviations progressively increase as the lag value increases. It is easy to see that the most important lags for use as discriminant variables will have low values of *k*. (Note that for convenience we will refer to the normalized autocorrelation sequence values as {Rd1, Rd2,..., Rd10} and the normalized central second-order moment as N2.)

TABLE 16. Group means (*N*=2048).

Class	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
1	.54933	-.31064	-.79678	-.63442	-.02861	.50224	.50546	.11604	-.18926	-.21037	.41251
2	-.27444	-.77753	.67221	.28368	-.72214	.14483	.47225	-.30202	-.16131	.22613	.38800
3	-.03209	-.25964	.10475	-.20775	.02000	-.08150	-.03982	-.00094	-.05461	.02490	.51096
4	.18536	-.46831	-.07317	-.06384	-.16020	.02203	-.02310	-.02245	.05586	-.00182	.46856
5	.13524	-.48510	-.05795	-.10568	-.13253	.05328	-.02881	.03377	.07176	-.00141	.46361

TABLE 16. Group means (*N*=2048).

Class	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
6	.17936	-.66603	-.26222	.13334	.03441	.04994	.08362	-.01260	-.03590	-.00249	.31470
7	.17153	-.78400	-.34622	.34166	.22199	-.02875	.00406	-.03714	-.10464	-.00463	.31955
8	.71393	.45993	.31726	.08430	-.02763	-.12392	-.22230	-.27917	-.27384	-.25810	.88494
9	-.00026	-.00174	.00095	-.00037	-.00021	.00026	-.00078	.00025	.00083	.00102	.33342
Total	.16780	-.40602	-.09063	-.09949	-.10379	.08693	.09414	.01988	-.04619	.02126	.45072
|4-5|	.05012	.01679	.01522	.04184	.02767	.03125	.00571	.01132	.01590	.00041	.00495

Consider classes four and five from Table 16. These classes correspond to the V.29 and V.32bis signals that were examined in the section on analytic lag selection. From the table we can rank the lags according to their mean value differences with respect to these two signal types. The ranking would be {Rd1, Rd4, Rd6, Rd5, Rd2, Rd9, Rd3, Rd8, Rd7, Rd10}. Now compare these rankings with those obtained in the analytical analysis section. The first four empirically ranked discriminant variables agree well with those of the analytical section, with the exception of Rd8. We can conclude that an accurate analysis must be sensitive to the channel distortions actually seen in the PSTN. We cannot simply assume that only additive Gaussian noise will be encountered.

TABLE 17. Group standard deviations (*N*=2048).

Class	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
1	.01703	.01315	.03493	.02424	.03622	.02997	.03175	.04451	.04561	.05027	.05768
2	.020[illegible]	[illegible]	.03558	.05456	.01834	.07366	.05730	.05936	.08333	.04944	.06160
3	.0423[illegible]	[illegible]	.04327	.04897	.03402	.03586	.02835	.02854	.02864	.02642	.02531
4	.08717	.03281	.12391	.02851	.04106	.05068	.03735	.03219	.03741	.03354	.01977
5	.08121	.02206	.11628	.03523	.03138	.04452	.04424	.02936	.03062	.03749	.03037
6	.02514	.01613	.04708	.02936	.04023	.03743	.04245	.03155	.03679	.03245	.01703
7	.01421	.01059	.03369	.02837	.03993	.03717	.03917	.03793	.03801	.03638	.00787
8	.22492	.24366	.26376	.28685	.27369	.23232	.19859	.19229	.19340	.21261	.16281
9	.02116	.02111	.02127	.02221	.02210	.02217	.02278	.02323	.02119	.02262	.01100
Total	.25761	.28720	.39205	.28192	.22419	.19596	.23408	.13001	.12443	.13215	.12730

The group means are unequal for all classes in at least one discriminant variable column. When the mean values are compared with the standard deviations, we can see that the classes can be separated quite well purely on the basis of their mean differences. However, consider classes 4 and 5. These have very similar mean values for most variables, and the mean differences are not significant compared to the standard deviations. (Later, when we allow the segment size N to be smaller, the standard deviations rise substantially, making the problem of group overlap even worse.) Thus we can predict that a linear discriminant function will not perform well for separating these two classes. A linear discriminator would in fact perform better if these problem classes were lumped together since their mean values are so similar. This is true for all classes in the system.

We can compare the results of Table 16 to the analytical estimates obtained for "best" lag selection. If our criteria is mean distance between classes, then from Table 16 we can see that classes 4 and 5 are separated well by Rd8, as compared to the standard deviations for that variable. Rd1 has a comparable mean difference to Rd8, however the standard deviation in Rd1 is larger than the mean difference making the variable useless in a linear discriminant function for these two classes. The analytical method also predicted that Rd8 would be a good choice for a discriminant variable. This theoretical result is verified by experimental observation. It remains to be seen which variable is optimal, however.

Incidentally, the only major differences between these two types of signals is that class 4 is half-duplex QAM with a 1700 Hz carrier, and class 5 is full-duplex QAM with an 1800 Hz carrier. The only significant difference is the 100 Hz carrier frequency separation. This is the reason the signals are so difficult to separate with discriminants based upon second order statistics and the PSDs. After we examine the covariance matrices we will be able to determine if these classes are appropriate for classification by a quadratic discriminant function. Table 18 contains the values of the pooled within classes covariance matrix which is generated by averaging all of the within classes covariance matrices together.

TABLE 18. Pooled within classes covariance matrix (*N*=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	5.562402 4E-03										
Rd2	2.297169 8E-03	3.759036 0E-03									
Rd3	- 3.680887 1E-03	1.607008 8E-03	9.639778 3E-03								
Rd4	1.449200 8E-03	1.666215 1E-03	2.877454 8E-03	5.323239 9E-03							
Rd5	1.065434 8E-03	2.022976 1E-03	1.562611 5E-03	3.423924 7E-03	4.789462 4E-03						
Rd6	- 1.053030 0E-03	7.004148 4E-04	3.555209 5E-03	1.781098 2E-03	2.627608 7E-03	[illegible]					
Rd7	9.482655 5E-04	4.296675 5E-04	- 6.546217 0E-04	1.508077 2E-03	1.451[illegible] 7E-[illegible]	[illegible]420 1E-03	3.435606 9E-03				
Rd8	2.801073 0E-04	- 8.413651 6E-05	- 5.467311 5E-04	3.090163 0E-04	5.956287 9E-04	2.958449 0E-04	1.846[illegible] 4E-03	3.080986 3E-03			
Rd9	- 8.700386 2E-04	- 6.967631 2E-04	- 1.380963 8E-04	- 1.240603 0E-03	- 5.315709 5E-04	8.243287 9E-04	5.749211 0E-05	1.416205 0E-03	3.497700 6E-03		
Rd10	- 2.077343 7E-04	- 1.391356 9E-03	- 2.344806 6E-03	- 1.525087 6E-03	- 1.172683 3E-03	- 8.414741 5E-04	4.620770 0E-04	5.169516 6E-04	1.756262 5E-03	3.608513 7E-03	
N2	- 7.340424 6E-04	- 6.367013 7E-04	- 2.829045 3E-04	- 8.135866 9E-04	- 7.571278 2E-04	- 5.363743 8E-04	- 4.961742 2E-04	- 3.300947 9E-04	- 1.508200 8E-05	2.055184 0E-04	2.543125 4E-03

TABLE 19. Within classes correlation matrix (*N*=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	1.00000										
Rd2	.50237	1.00000									
Rd3	-.50268	.26696	1.00000								
Rd4	.26632	.37248	.40169	1.00000							
Rd5	.20642	.47677	.22997	.67810	1.00000						
Rd6	-.21058	.17038	.54006	.36409	.56628	1.00000					
Rd7	.21692	.11956	-.11375	.35264	.35787	.31792	1.00000				
Rd8	.06766	-.02472	-.10032	.07630	.15506	.07949	.56765	1.00000			
Rd9	-.19725	-.19216	-.02378	-.28751	-.12988	.20788	.01658	.43141	1.00000		
Rd10	-.04637	-.37778	-.39757	-.34797	-.28208	-.20892	.13123	.15504	.49435	1.00000	
N2	-.19517	-.20593	-.05714	-.22112	-.21694	-.15863	-.16786	-.11793	-.00506	.06784	1.00000

TABLE 20. Between classes covariance matrix for class 1 (*N*=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	.0003										
Rd2	.0002	.0002									
Rd3	-.0005	-.0002	.0012								
Rd4	-.0003	-.0003	.0005	.0006							
Rd5	.0002	-.0002	-.0009	.0001	.0013						
Rd6	.0001	-.0001	-.0007	-.0001	.0008	.0009					
Rd7	.0002	.0002	-.0003	-.0006	-.0003	.0003	.0010				
Rd8	.0003	.0005	-.0001	-.0008	-.0011	-.0005	.0009	.0020			
Rd9	.0000	.0004	.0007	-.0002	-.0013	-.0013	-.0002	.0013	.0021		
Rd10	-.0004	-.0001	.0010	.0008	-.0004	-.0011	-.0014	-.0007	.0012	.0025	
N2	-.0003	-.0002	.0007	.0004	-.0003	-.0004	-.0003	-.0003	.0002	.0007	.0033

TABLE 21. Between classes covariance matrix for class 2 (*N*=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	.0004										
Rd2	-.0003	.0003									
Rd3	-.0007	.0006	.0013								
Rd4	.0011	-.0010	-.0019	.0030							
Rd5	.0000	.0000	-.0002	.0000	.0003						
Rd6	-.0015	.0012	.0026	-.0040	-.0003	0054					
Rd7	.0010	-.0009	-.0016	.0029	-.0004	-.0035	.0033				
Rd8	.0010	-.0008	-.0019	.0026	.0006	-.0040	.0018	.0035			
Rd9	-.0016	.0014	.0027	-.0044	.0001	.0058	-.0045	-.0037	.0069		
Rd10	.0000	-.0001	.0003	.0001	-.0009	.0005	.0012	-.0016	-.0006	.0024	
N2	.0000	.0000	.0000	-.0001	.0001	.0000	-.0002	.0002	.0002	-.0003	.0038

TABLE 22. Between classes covariance matrix for class 3 (*N*=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	.0018										
Rd2	-.0016	.0032									
Rd3	-.0014	.0006	.0019								
Rd4	.0016	-.0025	-.0009	.0024							
Rd5	-.0010	.0012	.0004	-.0010	.0012						
Rd6	.0006	-.0015	.0000	.0006	-.0005	.0013					
Rd7	.0000	.0004	-.0004	-.0001	-.0001	-.0003	.0008				
Rd8	-.0002	-.0003	.0004	.0000	.0002	.0001	-.0003	.0008			

TABLE 22. Between classes covariance matrix for class 3 (*N*=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd9	.0004	-.0001	-.0005	.0002	-.0003	.0001	.0000	-.0002	.0008		
Rd10	-.0003	.0004	.0002	-.0003	.0002	-.0003	.0001	-.0001	-.0002	.0007	
N2	-.0001	.0005	-.0002	-.0003	.0001	-.0003	.0002	-.0002	.0001	.0000	.0006

TABLE 23. Between classes covariance matrix for class 4 (*N*=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	.0076										
Rd2	.0023	.0011									
Rd3	-.0107	-.0029	.0154								
Rd4	-.0006	-.0006	.0006	.0008							
Rd5	.0026	.0005	-.0040	.0001	.0017						
Rd6	-.0037	-.0011	.0050	-.0002	-.0011	.0026					
Rd7	.0020	.0005	-.0029	-.0001	.0003	-.0008	.0014				
Rd8	.0016	.0005	-.0022	-.0001	.0004	-.0012	.0006	.0010			
Rd9	-.0020	-.0006	.0029	.0000	-.0007	.0009	-.0009	-.0003	.0014		
Rd10	.0015	.0004	-.0021	.0000	.0004	-.0008	.0004	-.0001	-.0003	.0011	
N2	.0002	.0000	-.0004	.0000	.0001	-.0001	.0001	.0000	.0000	.0001	.0004

TABLE 24. Between classes covariance matrix for class 5 (*N*=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	.0066										
Rd2	.0008	.0005									
Rd3	-.0093	-.0008	.0135								
Rd4	.0006	-.0005	-.0011	.0012							
Rd5	.0010	-.0002	-.0018	.0004	.0010						
Rd6	-.0028	-.0002	.0038	-.0008	-.0003	.0020					
Rd7	.0027	.0001	-.0037	.0004	.0000	-.0010	.0020				
Rd8	.0003	.0001	-.0004	-.0001	-.0001	-.0005	.0001	.0009			
Rd9	-.0005	.0001	.0007	-.0003	-.0002	.0002	-.0007	.0002	.0009		
Rd10	.0018	.0002	-.0025	.0003	.0001	-.0009	.0007	-.0004	.0000	.0014	
N2	-.0009	-.0001	.0012	-.0001	-.0001	.0004	-.0004	.0000	.0001	-.0003	.0009

TABLE 25. Between classes covariance matrix for class 6 (*N*=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	.0006										
Rd2	.0002	.0003									

TABLE 25. Between classes covariance matrix for class 6 (N=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd3	-.0011	-.0002	.0022								
Rd4	-.0002	-.0004	.0003	.0009							
Rd5	.0007	.0001	-.0016	.0001	.0016						
Rd6	-.0001	.0001	.0000	-.0008	-.0002	.0014					
Rd7	-.0004	-.0002	.0007	-.0002	-.0012	.0008	.0018				
Rd8	.0001	.0000	-.0001	.0003	.0000	-.0009	-.0002	.0010			
Rd9	.0001	.0001	-.0001	.0001	.0004	-.0006	-.0012	.0004	.0014		
Rd10	.0000	.0000	.0000	.0000	.0001	.0002	-.0003	-.0007	.0002	.0011	
N2	.0001	.0000	-.0002	-.0001	.0001	.0000	.0001	.0000	.0000	.0000	.0003

TABLE 26. Between classes covariance matrix for class 7 (N=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	.0002										
Rd2	.0001	.0001									
Rd3	-.0005	-.0002	.0011								
Rd4	-.0003	-.0003	.0005	.0008							
Rd5	.0004	.0000	-.0012	-.0002	.0016						
Rd6	.0003	.0003	-.0007	-.0010	.0004	.0014					
Rd7	-.0003	.0001	.0008	-.0003	-.0014	.0001	.0015				
Rd8	-.0002	-.0002	.0006	.0007	-.0006	-.0012	.0002	.0014			
Rd9	.0001	-.0001	-.0003	.0004	.0007	-.0006	-.0012	.0004	.0014		
Rd10	.0001	.0000	-.0003	-.0002	.0005	.0006	-.0005	-.0011	.0001	.0013	
N2	.0000	.0000	.0001	-.0001	-.0001	.0001	.0002	-.0001	-.0002	.0000	.0001

TABLE 27. Between classes covariance matrix for class 8 (N=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	.0506										
Rd2	.0415	.0594									
Rd3	.0102	.0420	.0696								
Rd4	.0226	.0487	.0658	.0823							
Rd5	.0109	.0376	.0561	.0694	.0749						
Rd6	.0024	.0198	.0381	.0481	.0564	.0540					
Rd7	-.0014	.0069	.0167	.0263	.0340	.0377	.0394				
Rd8	-.0019	-.0027	-.0018	.0037	.0142	.0225	.0302	.0370			
Rd9	-.0090	-.0166	-.0180	-.0164	-.0050	.0086	.0187	.0309	.0374		
Rd10	-.0158	-.0308	-.0328	-.0336	-.0237	-.0084	.0075	.0206	.0337	.0452	
N2	-.0095	-.0134	-.0128	-.0161	-.0143	-.0110	-.0077	-.0058	-.0018	.0039	.0265

TABLE 28. Between classes covariance matrix for class 9 (*N*=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	.0004										
Rd2	.0000	.0004									
Rd3	.0000	.0000	.0005								
Rd4	.0000	.0000	.0000	.0005							
Rd5	.0000	.0000	.0000	.0000	.0005						
Rd6	.0000	.0000	.0000	.0000	.0000	.0005					
Rd7	.0000	.0000	.0000	.0000	.0001	.0000	.0005				
Rd8	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0005			
Rd9	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0004		
Rd10	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0005	
N2	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0000	.0001

TABLE 29. Covariance matrix with 16367 degrees of freedom (N=2048).

Var.	Rd1	Rd2	Rd3	Rd4	Rd5	Rd6	Rd7	Rd8	Rd9	Rd10	N2
Rd1	.0664										
Rd2	.0373	.0825									
Rd3	-.0684	-.0010	.1537								
Rd4	-.0356	-.0246	.0752	.0795							
Rd5	.0268	.0225	-.0487	-.0130	.0503						
Rd6	.0200	-.0056	-.0431	-.0324	-.0022	.0384					
Rd7	.0064	-.0191	-.0285	-.0222	-.0187	.0378	.0548				
Rd8	.0105	-.0002	-.0373	-.0227	.0161	.0089	.0027	.0169			
Rd9	-.0116	-.0099	.0044	.0033	.0008	-.0076	-.0138	.0061	.0155		
Rd10	-.0283	-.0220	.0300	.0196	-.0163	-.0103	-.0011	-.0059	.0067	.0175	
N2	.0105	.0230	.0131	-.0033	.0001	-.0075	-.0120	-.0049	-.0029	-.0051	.0162

The separate group covariance matrices show that the covariances of each group are different. For example, the covariance between Rd1 and Rd2 is different for most classes. (The covariance values are 0.0002, -0.0003, -0.0016, 0.0023, 0.0008, 0.0002, 0.0001, 0.0415, and 0.0000 for classes one through nine, respectively.) This leads us to conclude that linear discriminant functions will not lead to optimal classification. An optimal result is obtained by using a combined linear and quadratic discriminant function

[69]. However, since the mean values for most classes differ substantially in many of the variables, a linear discriminant function will likely work very well for classification. The problem classes (4 and 5) will likely need to be classified using either a combined linear/ quadratic function, or a purely quadratic discriminant function. The optimal solution would be to use all of the available discriminant information, which would require the use of Shumway's [69] combined linear and quadratic discriminant function.

5.3.7.2 Discriminant Variable Selection

Next, we perform discriminant analysis on the 11 discriminant variables. This is done to evaluate the relative merits of each variable to an overall classification scheme. The subject of discriminant analysis was discussed earlier, thus only results are presented here.

The primary measure for discriminant variable selection in SPSS is Wilks' Lambda [55]. Other measures include unexplained variance, Mahalanobis distance, smallest F ratio, and Rao's V. All measures can be used in stepwise, forward, or reverse selection algorithms.

TABLE 30. Discriminant variable rankings (all classes, *N*=2048).

Rank	Wilks' Lambda	Mahalanobis Distance	F-ratio	Rao's V	Unexplained Variance
1	Rd2	Rd4	Rd4	Rd2	Rd2
2	Rd3	Rd8	Rd1	Rd4	Rd1
3	Rd7	Rd5	Rd5	Rd5	Rd4
4	Rd1	Rd7	Rd8	Rd7	Rd5
5	Rd4	Rd9	Rd7	Rd1	Rd3
6	Rd5	Rd6	Rd9	Rd6	Rd6
7	Rd6	Rd10	Rd6	Rd3	Rd8
8	Rd8	Rd1	Rd10	Rd9	Rd7
9	N2	N2	N2	Rd8	Rd9

TABLE 30. Discriminant variable rankings (all classes, N=2048).

Rank	Wilks' Lambda	Mahalanobis Distance	F-ratio	Rao's V	Unexplained Variance
10	Rd9	Rd3	Rd3	N2	N2
11	Rd10	Rd2	Rd2	Rd10	Rd10

Another way of ranking the discriminant variables is to first define the set of classes on which the analysis is performed, and then use the five available methods for discriminant variable selection. For example, the primary task of the classifier is to differentiate speech from non-speech signals. Therefore we have a two-class system. discriminant variable evaluation was performed on all of the signals, with all non-speech signals lumped together. Table 31 clearly shows that the discriminant variables are ranked similarly by all five methods. It is interesting to note that N2 is the most valuable variable for discriminating speech from non-speech, and the first lag of the ACS is the next best feature. Benvenuto also used the N2 variable, however he intended it to be used for separating various VBD modulation methods. Also, Benvenuto relied upon the second lag of the autocorrelation sequence as his primary discriminant for speech vs. non-speech discrimination when it may have been better to select the first lag. In fact, lags one, four, and nine all are ranked higher than lag two, according to Table 31. This is true regardless of the discriminant variable evaluation method used. (If Wilks' lambda is smaller for one variable than another, that does not necessarily imply that the misclassification rate will also be lower for that variable used alone.)

TABLE 31. Discriminant variable rankings (speech vs. non-speech, N=2048).

Rank	Wilks' Lambda	Mahalanobis Distance	Smallest F-ratio	Rao's V	Unexplained Variance
1	N2	N2	N2	N2	N2
2	Rd9	Rd9	Rd9	Rd9	Rd9
3	Rd4	Rd4	Rd4	Rd4	Rd4
4	Rd1	Rd1	Rd1	Rd1	Rd1

TABLE 31. Discriminant variable rankings (speech vs. non-speech, *N*=2048).

Rank	Wilks' Lambda	Mahalanobis Distance	Smallest F-ratio	Rao's V	Unexplained Variance
5	Rd2	Rd2	Rd2	Rd2	Rd2
6	Rd8	Rd8	Rd8	Rd8	Rd8
7	Rd3	Rd3	Rd3	Rd3	Rd3
8	Rd10	Rd10	Rd10	Rd10	Rd10
9	Rd7	Rd7	Rd7	Rd7	Rd7
10	Rd5	Rd5	Rd5	Rd5	Rd5
11	Rd6	Rd6	Rd6	Rd6	Rd6

Another interesting case is to study the discriminant variable rankings if only non-speech signals are considered. Table 32 shows the results where the speech class (eight) is removed from the analysis.

TABLE 32. Discriminant variable rankings (all non-speech classes, N=2048).

Rank	Wilks' Lambda	Mahalanobis Distance	F-ratio	Rao's V	Unexplained Variance
1	Rd4	Rd4	Rd4	Rd4	Rd2
2	Rd2	Rd2	Rd5	Rd2	Rd4
3	Rd5	Rd6	Rd2	Rd6	Rd5
4	Rd6	Rd5	Rd6	Rd8	Rd6
5	Rd7	Rd1	Rd1	Rd3	Rd1
6	Rd3	Rd3	Rd3	Rd7	Rd3
7	Rd8	Rd10	Rd10	Rd10	Rd7
8	Rd1	Rd8	Rd8	Rd5	Rd10
9	Rd10	Rd7	Rd7	Rd1	Rd8
10	N2	Rd9	Rd9	Rd9	Rd9
11	Rd9	N2	N2	N2	N2

Note that the N2 variable is almost the least effective variable for classification of VBD signals, yet it is the most effective variable for separating speech from non-speech signals. Also note that Rd4 is considered the most effective variable for non-speech signal discrimination. A closer examination of the results from SPSS indicates that Rd4 has the largest Mahalanobis distance between classes 4 and 5, which are the problem classes. This explains the importance of Rd4.

Now say that we are limited in the number of discriminant variables that we can select. A possible ad hoc strategy for variable selection could be choose the best variable from each of Table 31 and Table 32. This would result in using N2 and Rd4. Now allow the selection of one more variable, and say it has to be one of the top five for both tables. This leaves us with Rd2. So, our total discriminant set could be {Rd2, Rd4, N2}. If we compare this discriminant variable set with the variable rankings for all classes in Table 30, we can see that Rd2 is a good choice with a rank of #1, and Rd4 is also good since it is ranked #4. However, N2 is ranked very low in the table. This can be explained by the method of discriminant variable selection used. All classes have equal prior probabilities, and equal costs for misclassification. In fact, the separation of speech from non-speech is very important and this should be reflected in the value of the various discriminant variables. Thus we should still include N2 in the computation of the discriminant functions since it is critical for the separation of speech from non-speech as indicated in Table 31. We will test this set later.

At this point, we have determined what type of discriminant function is necessary to obtain both good and optimal classifier performance (linear and quadratic). We have also determined the relative merits of each discriminant variable, and we have an ad hoc method for selecting a subset of variables for consideration. Next we study the classifier performances using different discriminant functions, different discriminant variable choices, different values of *N*, and finally we will give an estimated classification rate study for an actual classifier.

5.3.7.3 Discriminant Function Evaluation

We have a choice of discriminant functions to use. We can use linear, quadratic, or combined discriminant functions. SPSS will always use linear discriminant functions, however it can be forced to utilize the separate classes covariance matrices when performing classification using the linear discriminant functions. SPSS then "warps" the discriminant functions (to obtain their "canonical" form) to reflect the correlation between variables [71]. This is not necessarily equivalent to using quadratic discriminant functions; however, it can be equivalent if certain criteria are met. It is a close approximation to quadratic discrimination and is called "pseudo-quadratic" in this thesis. We will use results from this method in place of the optimal quadratic discriminant functions. For comparison, we will use variables obtained from *N=256*, 124712 cases are in the data set, and all of the discriminant variables will be included in the analysis. We expect to see a marked improvement in the classification of classes four and five when the pseudo-quadratic method is used. Table 33 shows the results for the linear classifier, and Table 34 shows the results for the pseudo-quadratic classifier. The linear case has an overall correct classification rate (P_c) of 91.14% if we assume that equal numbers of data samples are included in each group. The pseudo-quadratic case has $P_c = 98.2\%$.

TABLE 33. Classification performance (linear, N=1024, all variables).

Actual Group	Predicted Group								
	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	94.5%	5.5%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	18.4%	81.6%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	98.8%	1.2%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	1.1%	98.9%	.0%	.0%

TABLE 33. Classification performance (linear, N=1024, all variables).

8	.2%	.3%	4.3%	.7%	.2%	.0%	.1%	93.3%	.9%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 34. Classification performance (pseudo-quadratic, N=1024, all variables).

Actual Group	Predicted Group								
	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	99.7%	.3%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	1.3%	98.7%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%
8	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

As predicted, classes 4 and 5 are the most difficult to classify using the linear method. When the pseudo-quadratic method is used, classes four and five are resolved more reliably. The P_c values for groups four and five improve from 94.5% and 81.6% in the linear case, to 99.7% and 98.7% in the pseudo-quadratic case. This result agrees well with the theoretical predictions.

It is also interesting to note that speech signals are better classified when the pseudo-quadratic method is used. The marked difference between speech classification performances for linear and quadratic methods is not seen when $N = 2048$. This can be explained if we note where the misclassified speech segments (class eight) are allocated. The misclassified segments are usually allocated to the 3rd group, which is V.34 VBD. The PSD of group 3 is very similar to that of white noise being transmitted through the network, and being filtered at the various codecs, etc. When N is reduced to 256 samples, the total segment length is only 32 milliseconds long. Thus, the unvoiced "noise" parts of speech will easily be caught in this short time. For example, consider the sound a person makes when saying the word "sheesh". When a person forms the "sh" sound, the signal characteristics are similar to those of a noisy VBD signal. So, it is easy to see why speech can be misclassified as group 3 VBD. This effect is worse when N is short, since it is more likely that the entire duration of a signal segment may be a noise-like sound. As N increases, it is more likely that a mix of vowels and consonants will be sampled and thus classification is easier.

There is another result that the above two tables lead us to. Most of the classes are discriminated very well using the linear method. For example, using the pseudo-quadratic method on classes 1, 2, and 3 gains practically no additional classification performance, since the performance is already very high. Classification rates for classes 6, 7, and 8 are improved when using the quadratic method, but the performance gain is significantly reduced when slightly larger values of N are used. The critical classes that require quadratic discrimination then are classes 4 and 5. This leads us to the conclusion that a stepwise classification method may be a good alternative. This method would retain the simplicity of implementation of linear discriminant functions, while gaining the accuracy of the quadratic discriminant functions. Recall that computing quadratic discriminant functions for several variables and several classes can require considerable computation.

5.3.7.4 Classification Performances for Varying Combinations of Variables

In section 5.3.7.2 we determined a discriminant variable ranking structure. Now we test these rankings by using only highly ranked variables for inclusion in the discriminant functions, and measure the resulting probabilities of correct classification. In each case we use long signal segments (N = 2048), all classes of signals, linear discriminant functions, and the top three variables ranked by the Wilks' lambda method to give a fair comparison.

TABLE 35. Best non-speech variable set {Rd2, Rd4, Rd5} (all signals).

Actual Class	Predicted Class								
	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	99.4%	.0%	.6%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	80.7%	19.3%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	14.3%	85.7%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%
8	.2%	.1%	1.0%	.6%	.0%	.0%	.0%	87.2%	10.9%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 36. Best non-speech variable set {Rd2, Rd4, Rd5} (non-speech signals).

Actual Class	Predicted Class							
	1	2	3	4	5	6	7	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	100.0%	.0%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	93.7%	6.3%	.0%	.0%	.0%
5	.0%	.0%	.0%	6.7%	93.3%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 37. Best speech vs. non-speech variable set {Rd4, Rd9, N2} (all signals).

Actual Class	Predicted Class								
	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	93.7%	.0%	.0%	.0%	4.7%	1.6%	.0%	.0%

TABLE 37. Best speech vs. non-speech variable set {Rd4, Rd9, N2} (all signals).

Actual Class	Predicted Class								
	1	2	3	4	5	6	7	8	9
3	.0%	.0%	99.8%	.2%	.0%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	80.6%	19.4%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	25.5%	74.5%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%	.0%
7	.0%	14.2%	.0%	.0%	.0%	.0%	85.8%	.0%	.0%
8	1.7%	.1%	2.1%	.4%	.1%	.1%	.0%	86.2%	9.3%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 38. Best speech vs. non-speech variable set {Rd4, Rd9, N2} (all signals).

	Predicted Class	
Actual Class	1 (non-speech)	2 (speech)
1 (non-speech)	100.0%	.0%
2 (speech)	8.4%	91.6%

TABLE 39. Best variable set for all signals {Rd2, Rd3, Rd7} (all signals).

Actual Class	Predicted Class								
	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	50.3%	49.7%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	39.5%	60.5%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	99.2%	.8%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	.6%	99.4%	.0%	.0%
8	.0%	.0%	2.2%	.1%	.2%	.0%	.0%	86.9%	10.5%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 40. Best heuristically selected variable set {Rd2, Rd4, N2}.

Actual Class	Predicted Class								
	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	70.2%	.0%	.0%	.0%	.4%	29.4%	.0%	.0%
3	.0%	.0%	99.6%	.3%	.1%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	80.8%	19.2%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	25.3%	74.7%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%	.0%
7	.0%	.6%	.0%	.0%	.0%	.0%	99.4%	.0%	.0%
8	.0%	.0%	2.5%	.4%	.0%	.0%	.0%	97.2%	.0%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

5.3.7.5 Classification Performances for Varying *N*

The segment size parameter *N* (samples) is critical to the performance of the overall classification system. Thus it is straightforward to see that by increasing *N*, we reduce the amount of variance in the discriminant variables. In this section we present results demonstrating the effect of varying the value of *N*. It was found that the usable range of *N* is $\{32 \le N \le 2048\}$, so the values of *N* tested are {32, 64, 128, 256, 512, 1024, 2048}. Note that the values of *N* used for testing were all powers of two. This is partly since *N* must be evenly divisible by the subsegment length $L=16$. It was decided that if *N* is a power of two, then FFT computations would perform optimally on the signal segments should the FFT be required.

Four cases were studied, with *N* varied for each case. Case 1 uses linear discriminant functions and all of the available discriminant variables (N2, Rd1, Rd2,..., Rd10). Case 2 uses pseudo-quadratic discrimination and all of the available discriminant variables. Case 3 uses linear discriminant functions and a small subset of the discriminant variables {N2, Rd2, Rd4}. Case 4 uses pseudo-quadratic discrimination and a small subset of the discriminant variables {N2, Rd2, Rd4}. The results from each case are displayed in Fig. 60 and Fig. 61 using bar charts. The bars in each chart represent the mean classification probability over all classes.

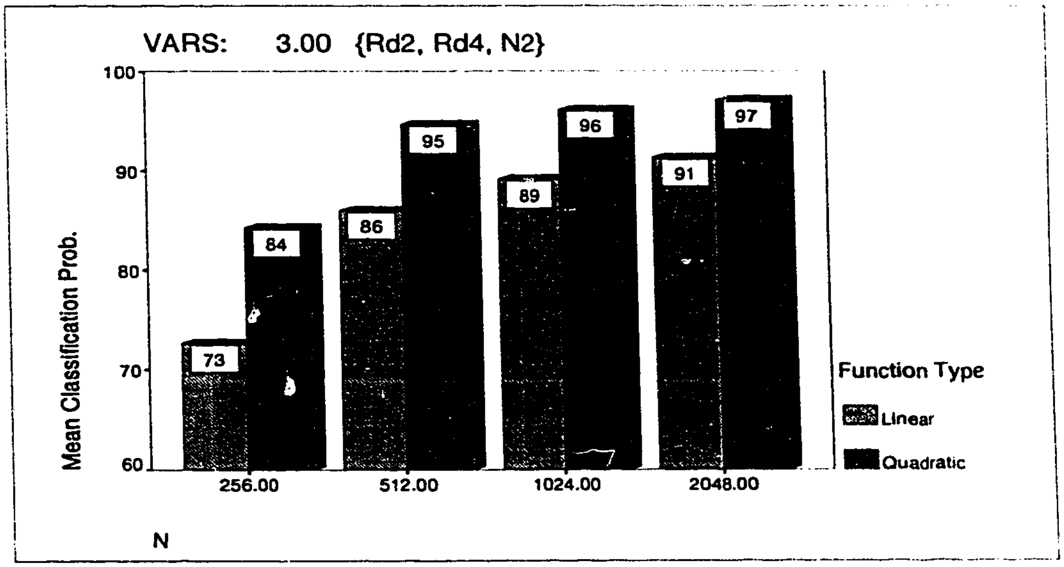


FIGURE 60. P_c vs. N using three variables, linear and pseudo-quadratic functions.

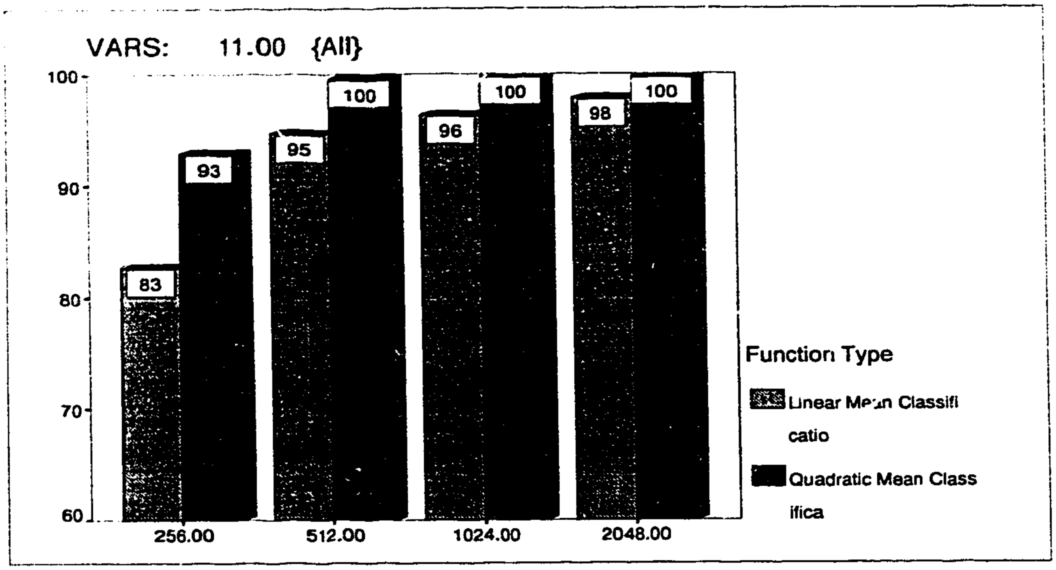


FIGURE 61. P_c vs. N using all variables, for linear and pseudo-quadratic functions.

The bar charts clearly show the advantage of using pseudo-quadratic discriminant functions over linear functions. The relationship between the size of N and the classification rate is also illustrated. It is important to note that the mean classification rate is not necessarily a good indicator of the overall classification performance. Often only a small number of classes in the overall classification group set are poorly classified; this results in a low mean classification rate even though most of the classes are correctly classified nearly 100% of the time. Refer to Appendix N for the standardized canonical discriminant function weights and individual class performances corresponding to the results presented in this section.

5.3.7.6 Estimated Actual Classification Performance

Here we try to estimate the performance of an actual implementation of our classifier. One potential problem with projecting classification performance from test data is that the classifier should not be develcped from the same data set as it is tested on. If this were to be done then the model developed for classification fits the sample data better than it would fit another set of sample data from the same population. This leads to an inflated prediction of classification performance over that achievable using a true classification system [54].

A popular method for estimating actual classification rates is called *cross-validation*. The method simply requires independent data sets for training and evaluating the classification algorithm. If only one data set is available, it can be divided into two equally sizes subsets, or the subsets could be of different sizes. The best approach is based on the number of data samples available. We chose to separate a single large data set into two equal subsets, using random sample file selection to determine which samples were placed in each subset. This separation method was used since adjacent samples from a single signal file may be highly correlated and thus they should belong to the same subset. The training and evaluation subsets should be as independent as possible.

The discriminant functions are formed while using one subset as the training set. Using the other subset, the discriminant functions already generated are used to evaluate classification. *Shrinkage* is the drop in apparent accuracy when comparing the classification performances obtained by cross-validation to those obtained in the usual way. A good classifier should have a small amount of shrinkage [54]. Some cross-validation runs are summarized in Table 41. The second row of the table shows a negative shrinkage value. This value indicates that the test set signals were better classified than the training set signals. From the table we can conclude that there is no substantial shrinkage for the signals considered. This conclusion indicates that the discriminant variables and functions that we selected are strong indicators of the signal characteristics that we wish to classify.

TABLE 41. Cross-validation shrinkages.

N (samples)	Discriminant Variables	Discriminant Function	P_c Training Set (%)	P_c Test Set (%)	Shrinkage (%)
256	all	linear	78.33	78.04	0.29
256	all	pseudo-quadratic	87.27	88.85	(-) 0.58
2048	all	linear	95.76	95.58	0.18
2048	all	pseudo-quadratic	99.95	99.89	0.06

5.3.8 Summary of Results

In this section we present several plots which illustrate the discrimination performance for each of the nine classes. The controlled factors are N, the discriminant variables (either all eleven or a subset of three {N2, Rd2, Rd4}), and the type of discriminant function (either linear or pseudo-quadratic). Each plot shows the probability of correct classification P_c on the y-axis. The x-axis represents the nine different signal classes. The probability of correct classification given a particular N value is plotted for each class, and connected by a line. By observing the plots we can determine which of the controlled factors affect each class the most. Note that the maximum value of P_c is 100%.

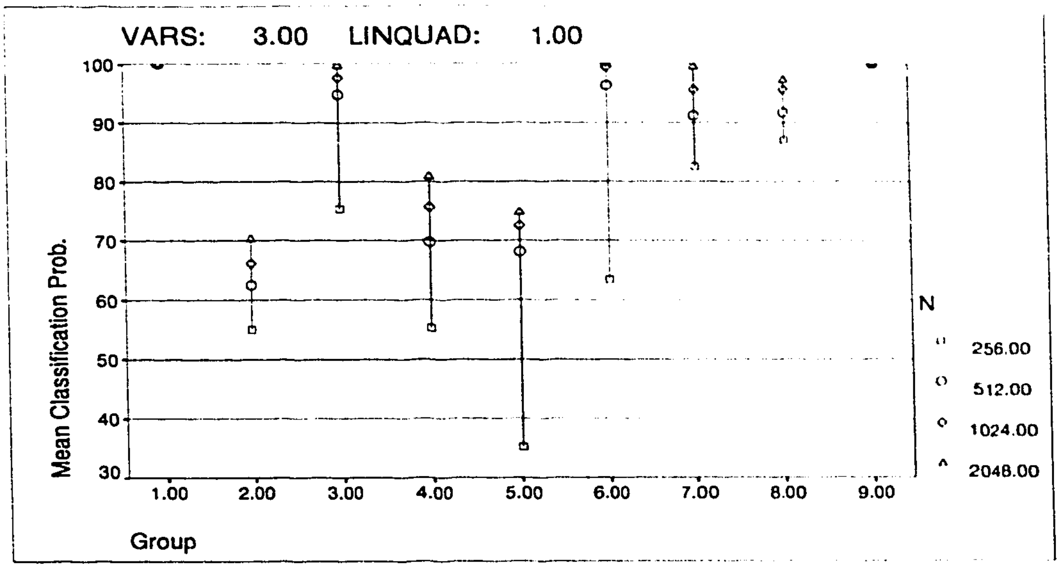


FIGURE 62. *Pc* vs. Group, varying *N*, using three variables and a linear function.

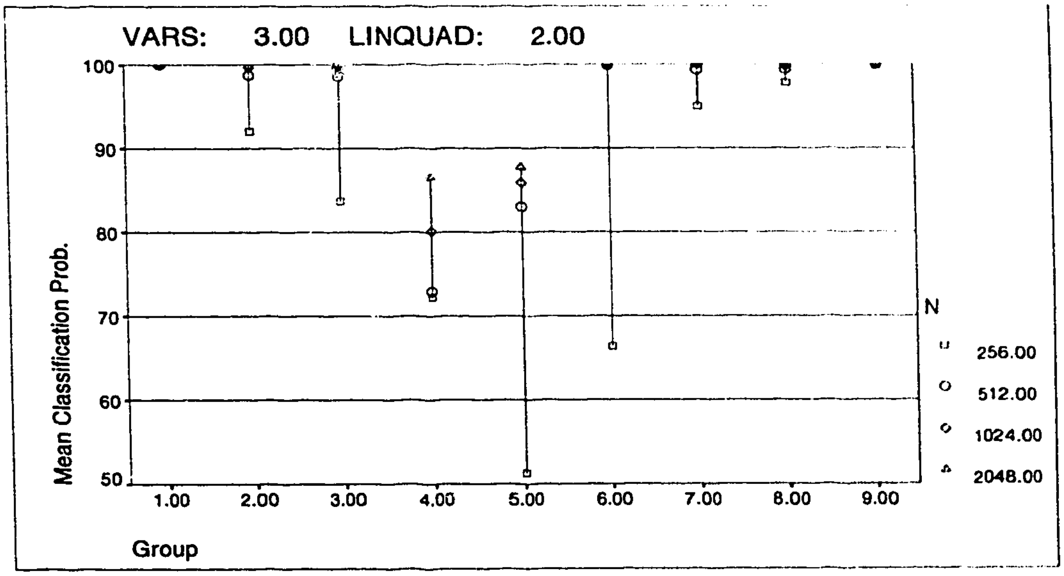


FIGURE 63. *Pc* vs. Group, varying *N*, three variables, pseudo-quadratic function.

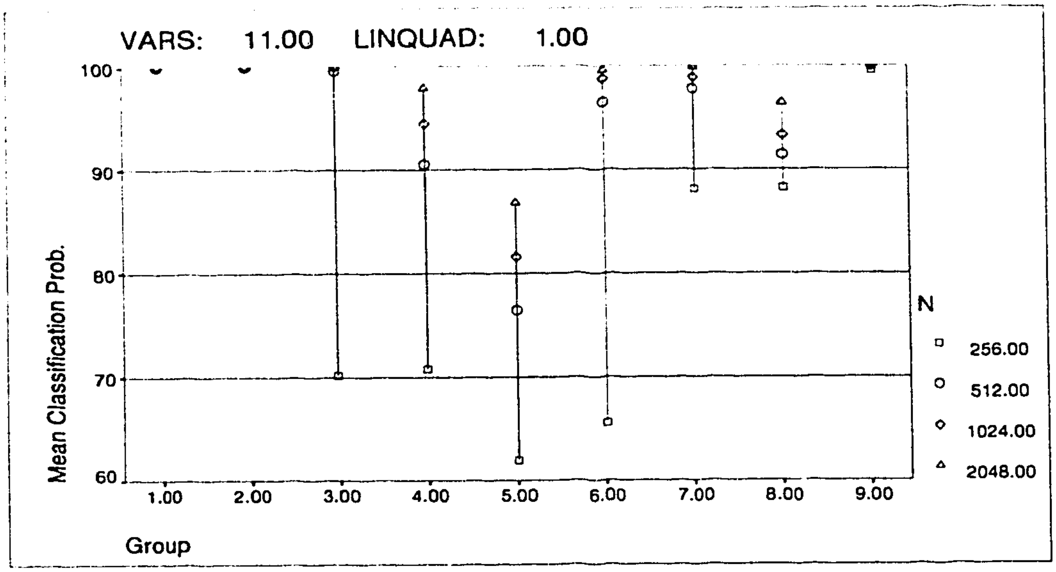


FIGURE 64. *Pc* vs. Group, varying *N*, all variables, and a linear function.

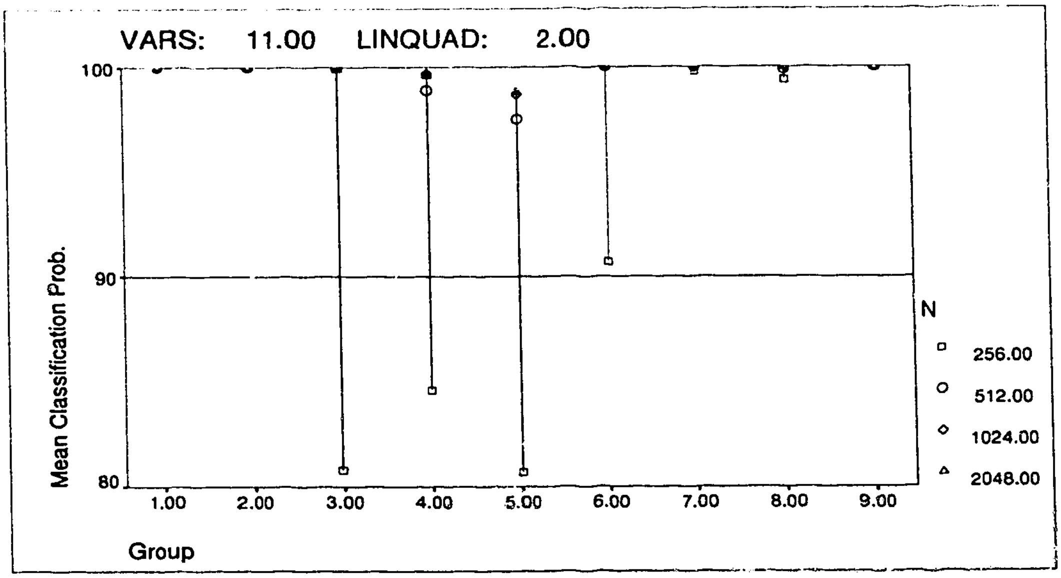


FIGURE 65. *Pc* vs. Group, varying *N*, all variables, pseudo-quadratic function.

The plots show that the most problematic classes are numbers three, four, five, and six. When all of the available discriminant variables are used and a pseudo-quadratic discriminant function is used for classification, N must be at least 512 samples to guarantee 98% classification accuracy for all groups. See Fig. 65 for an illustration of this point. If we force the classifier to use only linear discriminant functions, N must be as large as possible for accurate classification over all classes.

Next we present plots that illustrate the characteristics of each discriminant variable. The box charts below show the median, interquartile range, extreme values, and outliers (shown as open circles) for each discriminant variable versus each classification group. (Each quartile contains 25% of the sample cases. The boxes in the charts contain two quartiles and thus 50% of the total sample cases.) Box charts are useful for determining which variables separate the different classes best. Since the correlations between variables are not shown, these plots do not indicate the relative effectiveness of the variables for a quadratic discriminant function system. A purely quadratic discriminant function does not utilize the mean values of the discriminant variables. In all cases N=2048.

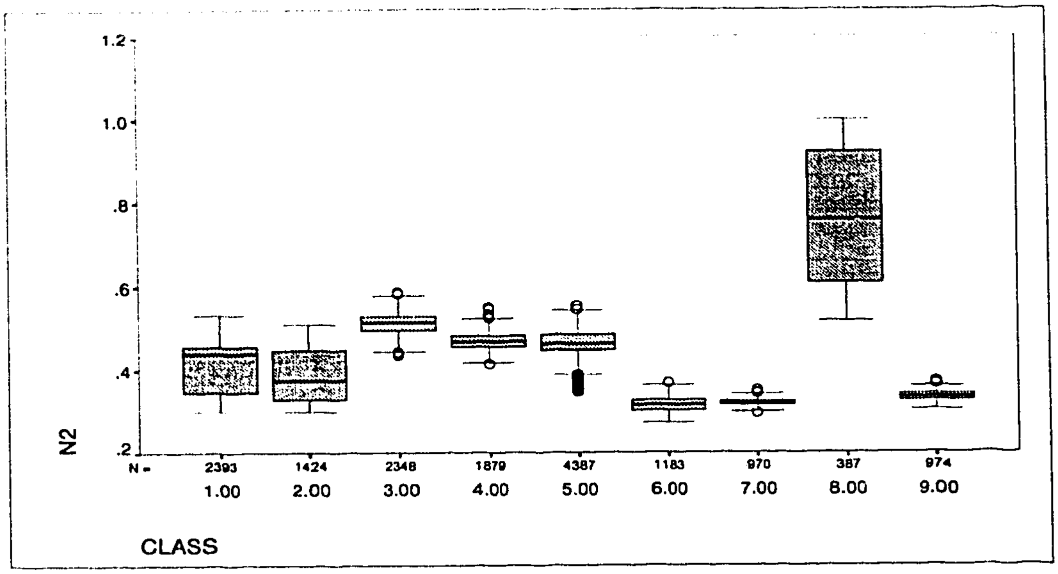


FIGURE 66. Boxplots of N2 vs. class.

The box plots in Fig. 66 demonstrate N2's ability to discriminate all nine classes of signals. We already know from earlier analysis that N2 is not useful for discriminating between various VBD signals. This is verified by the plots. For example, the boxes plotted for classes one and two are nearly identical. The remaining VBD classes (3, 4, 5, 6, 7, 9) have boxes that overlap with those of classes one and two. These overlapping boxes indicate that a linear discriminant function based on N2 will not separate VBD signals from one another very well. On the other hand, the box plot for speech signals (class eight) is clearly separated from the VBD signals, indicating that a linear discriminant function using N2 as a discriminant variable will easily separate speech from VBD signals. The remaining box plots are contained in Appendix O.

5.3.8.1 Robustness Test

In a previous chapter it was mentioned that four voice signal recordings were made of people trying to "fool" the classifier. In this section we present the results of testing those signals. The signal segment length used for these tests was $N = 1024$. Two different classifier configurations were tested. The first configuration used three discriminant variables {Rd2, Rd4, N2} with linear discriminant functions. The second configuration used all available discriminant variables {Rd1, Rd2,...,Rd10, N2} and quadratic discriminant functions. These two configurations are intended to represent the worst can best types of classifiers.

In the tests we want modem "mimicry" to be caught. First of all, we know that a person cannot make a particular sound for an indefinite amount of time. People must breathe. Machines like modems and facsimile machines have very constant power outputs during the bulk of their communications. People have a very difficult time holding sounds for an extended period of time. However, it is interesting to see what the short-time classification performance of the two configurations is for these "bad" signals. Note that the classifier was trained using "nice" speech signals.

The first classifier configuration was tested on the mimicked signals. The results indicated that the signals were frequently misclassified. When a person imitated a modem, they were often misclassified as class 1. This class corresponds to a low-speed modem signal. When the whistle recording was tested, it was frequently misclassified as class 3. We give an explanation for this later. In all cases the results were not consistent. The signals were classified as one particular class for a short number of intervals, and then the classification result would change. It is reasonable to assume that different people will sound like different types of modems when they try to mimic the noise-like sound that modems make.

In the second configuration, far better results were obtained. The signals in which a person tried to mimic a modem were never misclassified. 100% of these signal segments were correctly classified as speech. The signal in which the talker whistled was misclassified 5% of the time as class three. This can be explained as follows: When a person whistles into a microphone in their telephone handset, they must blow quite hard. The rushing of air over the microphone opening causes broad spectrum noise to be combined with the talker's whistle. Also, when a person begins to whistle, the beginnings of the whistle usually just come out as a rush of air with no tone. The whistler then corrects his/her mouth shape to force a whistle tone. This noise is then filtered by the telephone network codecs, and the result looks very much like the spectrum of a V.34 modem (class 3). A close examination of the whistle recording revealed that there were several instances where the talker had to stop for breath, or momentarily lost tone. These are the points in time that correspond to misclassifications as class 3.

We conclude that while people may be able to mimic modem sounds for a short duration, they cannot continue successful mimicking for very long periods of time. When they stop for breath, or change the shape of their mouth in a small way, they will be caught. The constant nature of modems and FAX signals is not duplicable by humans without the aids of recording devices. We have also seen that if the classifier is given enough discriminant variables and allowed to use quadratic discriminant functions, people will have a very difficult time fooling the classifier into misclassifying human generated signals as VBD signals.

5.3.8.2 Real-time Computation

The discriminants described above are appropriate for real-time computation due to their simple summation structures. Other elements of a complete classifier include a power threshold monitor and a voltage amplitude monitor. These are used to ensure that the signal segment being tested is not silence, such as the periods of time between voiced words in speech. See Benvenuto's paper [6], or section 2.1.3, for pseudo-FORTRAN code for computing the discriminants. The front end of our algorithm contains a full-wave rectifier stage rather than a Weaver demodulator as in Benvenuto's algorithm. In order for the discriminants to be useful, a classification method must be selected. A block diagram of the system is shown in Fig. 67. Note that not all of the discriminant variables shown in Fig. 67 must be included in a particular system design.

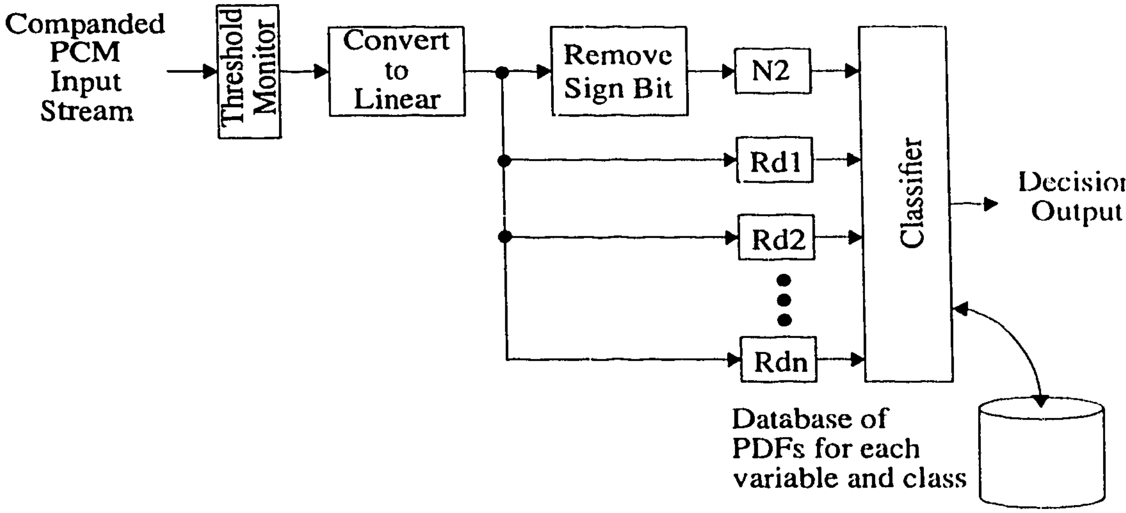


FIGURE 67. Block diagram of classification system.

5.4 Classification of Companded Signals

Up to now we have assumed that all monitored companded signals will be converted to linear before the discriminant variables are evaluated. The process of converting from μ-law to linear may be implemented using a table look-up. Since there are 256 possible codewords that are in the code, we require 256 table entries for the conversion. Also, 8-bit μ-law PCM codewords expand to 12-bit linear words. Therefore, we will need approximately 512 bytes of table storage. Each codeword conversion requires a single reference to the conversion table. While this method is very fast, it requires a significant amount of DSP on-chip memory.

We can use other methods to reduce the amount of memory required for the conversion. These methods are based on the formation of linear segments from the companding table. For example, the μ-law table has 16 segments, each of which contains 16 levels. The quickest algorithm we were able to find required 9 TMS320 series DSP operations to perform the conversion of one codeword to its linear form. This method was discovered in a public software archive at Texas Instruments. We can conclude that without using a look-up table, approximately 10 times more operations are required to convert from a companded to a linear signal. Since the discriminant variables themselves may each only require about three or four operations per sample to calculate, we obviously need to perform the linearization task in as few cycles as possible.

We tested a novel approach to classifying companded signals without performing linearization. In order to use the companded version of a signal for classification, the bit ordering must be altered somewhat. Consider the μ-law code table that is abbreviated in Fig. 68. The binary numbers represent the codewords as they would be received within a network. If we simply plot the values of the companded PCM sample stream, the result will not look like the original signal since the positive and negative portions have been separated. If we could invert the bits marked in Fig. 68 then the resulting sample stream would have large values for maximum positive words, and small values for maximum negative words. The most significant bit could then be used as the sign bit. If this signal representation was plotted in time it would look very much like a "squashed" version of the original time-series signal. The peaks of the signal would be smaller due to the companding operation.

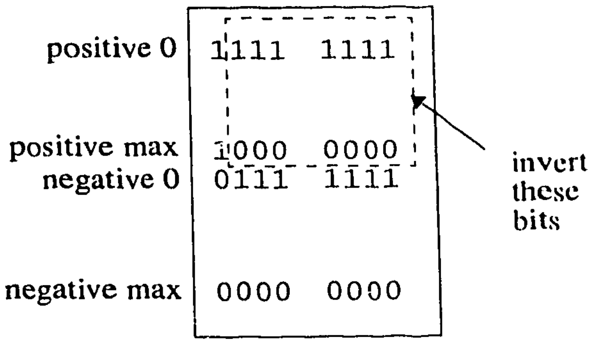


FIGURE 68. Abbreviated µ-law code table.

Flipping the bits in the top half of the codeword matrix can be performed using a bit-wise inversion operation. One possible method is to XOR the observed codeword with $(0111\ 1111)_2$. Note that this operation must only be performed on the codewords with a most significant bit equalling one. An additional bit test operation may be required to ensure this; however, some processors provide bit test and set operations in a single instruction.

We implemented an algorithm for modifying companded samples as shown using MATLAB to test the effects of classifying companded signals. The discriminant variables {Rd1, Rd2,..., Rd10, N2} were calculated using N=2048, L=16, and P_{Th}=1089. Four classifier configurations were used: (1) using linear discriminant functions and three variables

{Rd2, Rd4, N2}, (2) using pseudo-quadratic discriminant functions and three variables {Rd2, Rd4, N2}, (3) using linear discriminant functions and all available variables, and (4) using pseudo-quadratic discriminant functions and all available variables. The same nine level class structure was used, as described earlier. Results are shown in Table 42 for both linearized and companded signal classification performances. The results for the linearized case are taken from section 5.3.7.5.

TABLE 42. Classification results for companded signals.

Configuration	Overall Probability of Correct Classification (Linearized)	Overall Probability of Correct Classification (Companded)
1	91%	87.27%
2	97%	93.70%
3	98%	95.88%
4	100%	97.92%

We can see from the table that the results have degraded. A closer inspection of the results reveals a general reduction in classification performance across all classes of signals. Speech signals appear to be affected slightly worse than other classes. We conclude that if conserving memory space is critical in a particular classifier design, it is possible to perform a single operation on each companded PCM to allow the companded signal to be classified directly. Classification accuracy is reduced by approximately 4% for linear discriminant function classifiers and about 2% for pseudo-quadratic classifiers. Otherwise, at the expense of dedicating 512 bytes of memory to a look-up table, we can classify the linearized signals using a conversion method that only requires one operation per PCM sample.

Chapter 6

6.0 Carrier Detection, Timing Recovery, and Other Methods

The classification methods discussed so far can discriminate speech from VBD, and subclassify VBD signals according to their PSD characteristics. We also studied the problem of classifying VBD signals according to their QAM constellation maps. The results from this study, and other less important methods, are presented in this chapter.

6.1 Carrier Detection and Timing Recovery

Discriminant variables based on the autocorrelation sequence of a signal can only resolve signal types that have different power spectra. Therefore, we require other methods to discriminate QAM signals that differ only in their constellation patterns. The advantage of knowing a signal's constellation pattern is that then the precise bit rate of the signal can be determined. If necessary, the encoded binary bit-stream signal could even be recovered.

One possible approach to constellation discrimination is to demodulate the received signal and recover the timing. The resulting signal would have a two-dimensional scatter diagram. This diagram could be studied for patterns indicating the number of constellation points in use, and their orientation. The carrier frequency used can be easily estimated since there is a limited number of carriers present in VBD communications.

In the context of data transmissions, *carrier detection* is the process of generating a clock at the receiver that corresponds to the transmitter carrier in both frequency and phase.. *Timing recovery* refers to generating a clock at the receiver that corresponds to the transmitter baud clock, i.e. the symbol rate clock. Both of these clocks must be generated at precisely the correct frequency and phase with their transmitter counterparts to successfully receive the transmitted binary bitstream. In order to conserve power, no pilot tone is provided for this task. Since no spectral impulse is present at the carrier frequency, this method is known as suppressed carrier as opposed to large carrier. Large carrier signals incorporate a strong spectral component at the carrier frequency to permit easy carrier recovery using a narrow band-pass filter.

The issues of recovering the carrier and baud clocks is the subject of entire reference books, and several chapters in most digital communications books. A modem system designer need only consult a reference book and select an established method for carrier detection rather than develop a new one (if that is possible). For this reason, we will give only a short overview of the standard methods.

There are several different ways of performing this task. Normally, each clock is generated using information recovered from the other. For example, information from the symbol timing recovery circuitry is used to help lock the phase of the carrier. The ITU specifies that for most protocols detection must still be possible even if the carrier is off-frequency plus or minus 7 hertz at the receiver. The modulation rate (baud rate) of the transmitter must be within plus or minus 0.01% of the nominal value. By using this information, most established detection methods are based on the fact that the carrier and baud rate do not fluctuate very much. This way, Phase Locked Loops (PLLs) or Band Pass Filters (BPFs) can be used effectively.

6.1.1 Forward-Acting Methods

This method of carrier detection is based on a simple strategy [11]. If a QPSK or 4-QAM signal is to be considered, there are only 4 possible constellation points on a circle separated by multiples of π/2 radians. Thus, if the incoming signal is "frequency quadrupled", the phase changes in that signal would only be in multiples of 2π, and the signal would contain a discrete component at four times the carrier frequency. A suitable frequency quadrupling operation is performed by a fourth power circuit. Alternatively, two squaring circuits in series could be used if a fourth power circuit is too expensive to implement. A BPF centered at $4*F_c$ could be used to filter out the component corresponding to the quadrupled carrier. Then the carrier could be recovered by frequency division, say using a phase-locked loop circuit. This simplistic method actually works well for small QAM constellations, and is not overly difficult to implement in hardware. However, certain signal patterns may not yield the $4*F_c$ component.

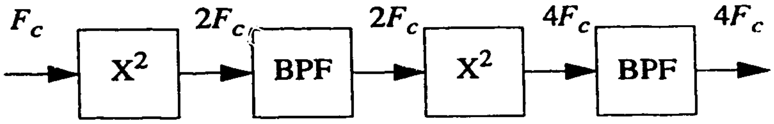


FIGURE 69. Forward-acting carrier recovery circuit.

This method can be expanded to larger constellations; however, the resulting component at $4*F_c$ will not be the only strong component of the signal. Consequently phase jitter would likely be a serious problem. For this method to work with 16-QAM, a precise PLL must be used to recover the $4*F_c$ component. The trade-off of this method is theoretical simplicity versus the complexity of an extremely narrowband PLL.

6.1.2 Feedback Methods

Feedback methods are based on the generation (or synthesis) of a low-frequency control signal input into a VCO (Voltage Controlled Oscillator) that produces an estimate of the carrier. This low-frequency signal is generated from the phase error of the generated estimated carrier signal with respect to the received modulated carrier. There are several reasonable ways of performing this type of detection, so only one will be briefly discussed here.

One method, called demodulation/remodulation, is performed by using the locally generated carrier to first demodulate and then remodulate the limited baseband signal and then compare the remodulated passband signal to the original passband signal. A phase error is generated from this comparison and subsequently used to control a VCO to correct the frequency of the generated carrier. This method is illustrated in Fig. 70.

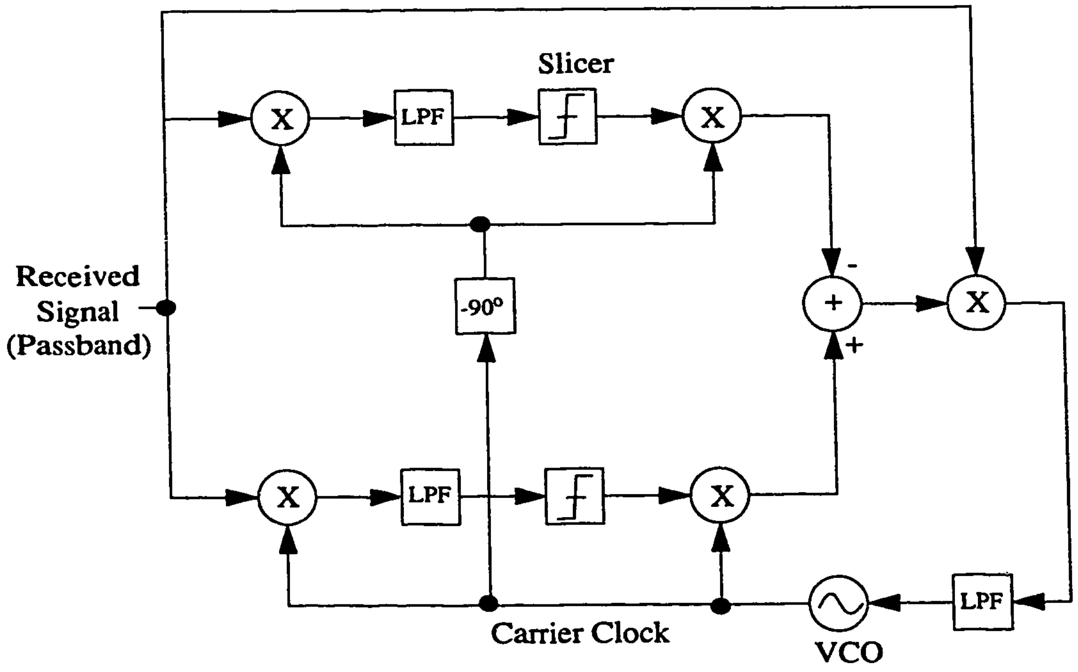


FIGURE 70. Feedback carrier recovery circuit using demodulation/remodulation.

6.1.3 Decision-Directed Methods

Decision-directed methods are an improvement over both forward-acting and feedback methods. The disadvantage of feedback methods is that they are subject to intersymbol interference as well as signal distortion noise, hence the received signal may not be able to provide enough information to successfully generate a carrier frequency clock. This is a result of the use of non-rectangular pulse shaping filters at the transmitter and quantization noise in the telephone network. This problem can be avoided by decision-directed methods.

The main idea of decision-directed methods is that the symbol timing is extracted before the carrier is recovered, as shown in Fig. 71. Here, the phase error is only estimated at the symbol times, or where there is a maximal eye opening (referring to the eye diagram of a signal). This way, since there should be minimal intersymbol interference, the phase error estimate is only dependant upon the current symbol. Recall that in feedback methods, the phase error calculation is based on an unpredictable combination of past, present, and future symbols. Thus the problem of carrier detection is now dependant upon first solving the timing recovery problem.

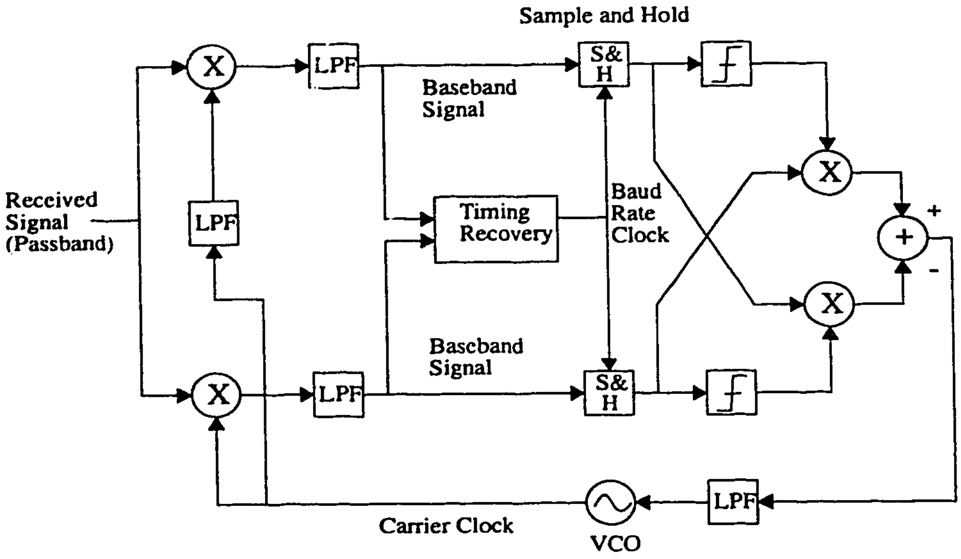


FIGURE 71. Basic decision-directed carrier recovery loop circuit.

6.1.4 Timing Recovery

The problem of recovering symbol timing can be solved in two general ways, with either a BPF or a PLL [11]. Timing recovery is normally only performed on a demodulated signal at the baseband. However, for conventional QAM, it may be performed on the pass-band signal. This is an important feature of QAM, allowing the carrier detection to be performed after timing recovery. Again, as with carrier detection, timing recovery can be performed by forward-acting systems or feedback systems.

6.1.5 MATLAB Algorithm Implementation

Consider a sample application: V.29 signals use a form of modulation that can be considered QAM. (The constellations of V.29 signals are not rectangular.) Either 16, 8 or 4 points are used, depending upon which mode the V.29 transmitter is using. In all three modes the carrier frequency is 1700 Hz and the symbol rate is 2400 symbols/second. The transmitter carrier is allowed to deviate from the nominal value by plus or minus one Hz, and the network frequency offset is rarely over one Hz. (A deviation greater than plus two Hz or minus 1 Hz occurred in 0.43% of all network connections according to [12]. All stable frequency offsets observed were the result of a single frequency division multiplexing CO). Therefore, we can say that the carrier frequency used at the receiver should not deviate from the nominal value by more than 1 Hz.

If the carrier signal used at the receiver was slightly out of phase with the transmitter carrier, the constellation pattern would have a fixed phase offset. If the receiver carrier was not identical in frequency to the transmitter carrier, then the constellation at the receiver would be rotating at a constant rate. The scatter plot in Fig. 72 illustrates what a 16-point V.29 constellation looks like if the receiver carrier frequency is 1 Hz higher than it should be. Note that it is still possible to see rings in the constellation. If we now plot the histogram of the symbol vector magnitudes, as shown in Fig. 73, a pattern is evident. The peaks in the histogram correspond to the constellation symbol vector magnitudes. For the V.29 16-point constellation there are three unique symbol magnitudes.

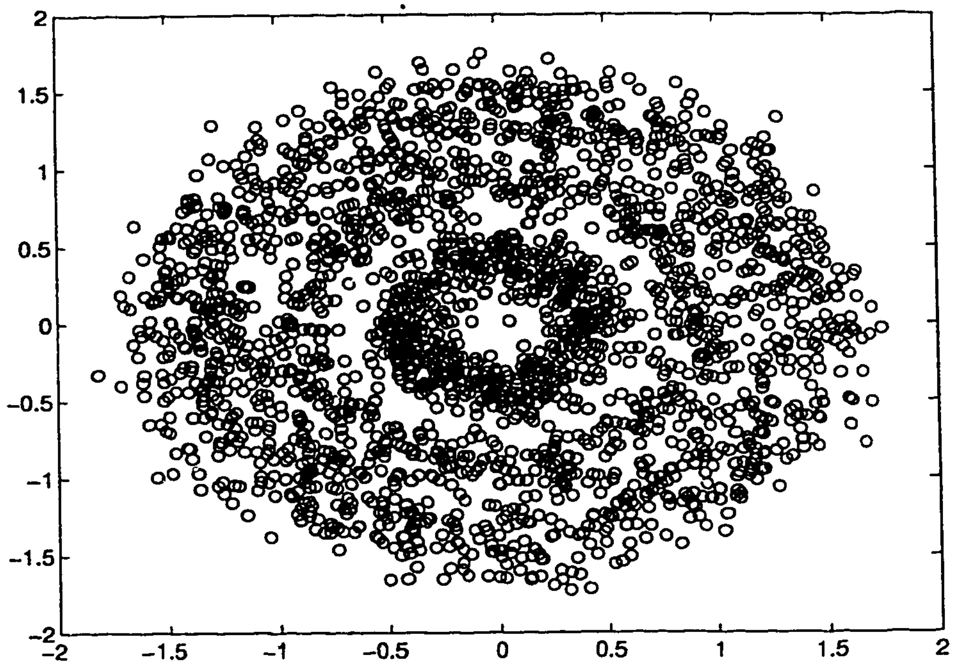


FIGURE 72. Scatter plot of rotating constellation.

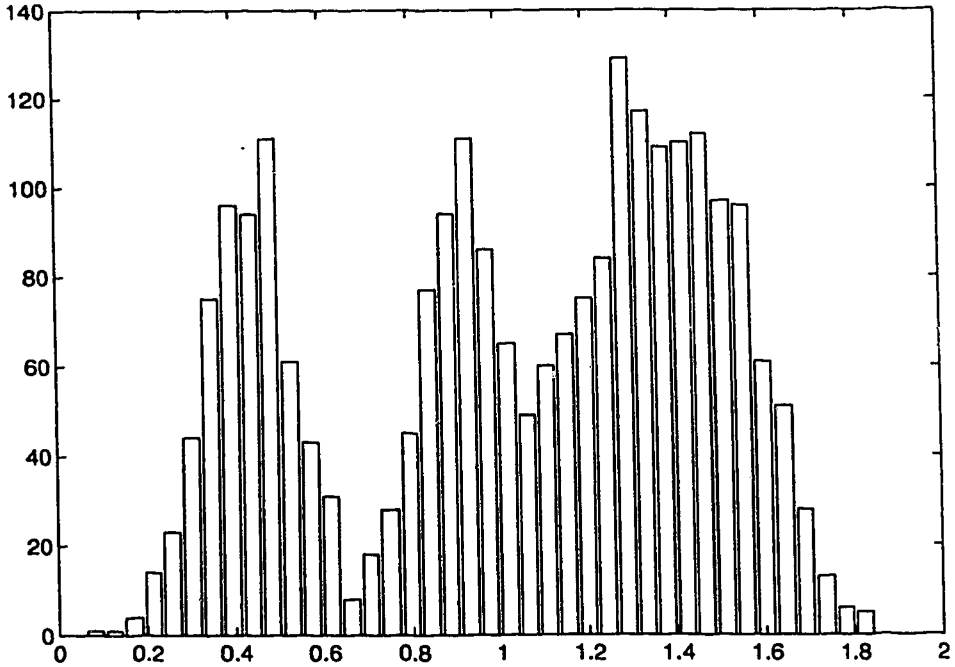


FIGURE 73. Histogram of symbol vector magnitudes.

The receiver baud clock must be precisely locked in frequency and phase with the received signal timing, otherwise the signal symbols will not be recovered. If there is a constant difference between the frequency of the transmitter and receiver timing clocks, the constellation will appear to move in and out of "focus" at the receiver end. Any deviation from the optimal slicing phase will result in a dispersion of the constellation points.

The proposed technique is as follows: (1) Demodulate the received signal using the nominal 1700 Hz carrier frequency. (2) Use a forward-acting timing recovery method to generate a symbol rate clock. (3) Slice the demodulated signal using the generated symbol rate clock. (4) Generate vector amplitude histograms (where the amplitude is the vector length of the complex valued demodulated signal symbol) to be used for classification. Next we discuss the MATLAB tests that were developed for classifying V.29 16- and 8-point constellations.

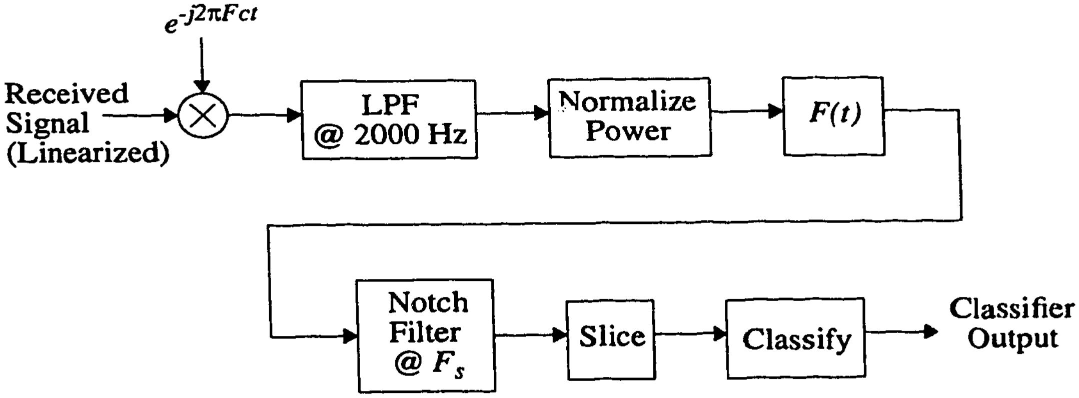


FIGURE 74. Generalized V.29 constellation classifier.

6.1.5.1 Demodulation

As shown in Fig. 74, the demodulation stage involves shifting the passband signal down to the baseband by multiplying by a nominal carrier. Then the shifted signal is filtered using a low-pass filter (LPF) with a cutoff at 2000 Hz. The resulting signal is the complex-valued baseband QAM signal.

6.1.5.2 Timing Recovery

The baseband QAM signal is normalized to unity power in order to for the classifier to be immune to the relative losses of each channel monitored. Next a nonlinear operation *F(t)* is performed on the signal in order to generate a frequency component at the symbol rate. The nonlinear operation can be either an absolute value, square, or fourth power function. In the case of QAM signals, the absolute value operation yields the strongest spectral line at the baud rate [11]. The spectral component is then filtered using a narrow band-pass filter. The resulting signal is a sinusoid locked in phase and frequency with the QAM symbol rate clock. This is known as forward-acting timing recovery, as mentioned earlier. An implementation of forward-acting timing recovery for QAM signals is given as MATLAB code in Appendix P.

6.1.5.3 Slicing

After compensating for delays within the timing recovery subsystem, the generated symbol rate clock is used to provide the timing for a slicer. The slicer operates by sampling the baseband signal at each symbol interval. In order for sampling to occur between regular sample intervals (at the 8000 Hz rate) the signal must be interpolated. The output of the slicer can be plotted as a two-dimensional scatter diagram. If the system is performing properly, and the channel SNR is sufficiently high, the QAM constellation in use is visible in the scatter diagram.

6.1.5.4 Classification

Finally, the output from the slicer can be classified. There are many possible methods of performing this stage. In the case being considered, simply monitoring the amplitude histogram of the sliced baseband QAM signal is sufficient. Signal segments of N samples are used as the basis for forming decisions.

The most accurate and general method that was tested was to compare the received signal scatter diagram to that of the ideal 8- and 16-point constellations. The algorithm is outlined in the following pseudo-code:

```
set the number of incremental phase rotations = I;
for all ideal templates do this
    for phase_rotation = 1 to I
        rotate the ideal template;
        for all points on the ideal template do this
            for all points in the observed scatter diagram do this
                compute the distance between current ideal point and observed point;
            end for
        end for;
        select only the minimum distance for each scatter diagram point;
    end for;
end for;
compare the mean minimum distances for each ideal template considered;
the template with the smallest mean minimum distance is output;
```

The pseudo-code rotates the ideal constellation a certain number of times. This must be done since the carrier frequency used by the receiver is not locked to that of the transmitter, hence there will be some arbitrary phase offset between the observed scatter diagram and the ideal constellation map. Since the absolute difference between the optimal receiver carrier frequency and the nominal frequency used is likely to be small, there will not be a significant amount of rotation in the observed scatter diagram providing the value of N is kept small. An improvement would be to lock the carrier frequency as well as the timing.

Other methods of performing classification were also studied. One such method was to compute the same mean minimum distance as before, but to "fold-over" the ideal and observed constellations. The constellations are symmetrical, so they can be folded over into a single quadrant. This will reduce the number of points in the ideal constellation map, and hence reduce to computational complexity of the classifier.

A third method is to ignore the constellation point phase values. Only the vector magnitudes of each symbol point are monitored. Again the mean minimum distances between the ideal constellation map and the observed scatter diagram is used as the criteria for classification.

A fourth method is to not only monitor the vector magnitudes, but also their frequency of occurrence. Each observed symbol point is allocated into one possible ideal constellation point. By monitoring the statistics of how often each ideal point is allocated, we can determine whether the observed signal is a good match to that constellation. Every point in a standard QAM constellation is equiprobable. We can compute the frequency of occurrence of observed symbols being allocated into each ideal constellation point bin. The closer the observed frequencies of occurrence correspond to the ideal case, the more confident the classifier can be in its decision. One particular implementation of this algorithm is expressed as MATLAB code in Appendix Q.

Several methods of performing classification are possible, and some have been briefly discussed. As the results shown in the next section illustrate, it is not easy to observe the constellation of a received signal as the SNR drops and the constellation complexity increases. Thus we conclude that more emphasis should be placed on developing methods for recovering observed signal constellations than on the method used for identifying which constellation has been recovered.

6.1.5.5 Results

The algorithms discussed in the previous sections were tested on simulated V.29 signals, recorded V.29 signals, simulated V.32 signals, and recorded V.32 signals. Due to the high probability of symbol error in the unequalized received V.32 signals, the constellation pattern could not be extracted with any of the attempted methods. It was possible to observe the constellation pattern of the V.29 signals, and classify them with 100% accuracy. The main factor limiting the accuracy of the V.29 constellation classifier was the actual method used to form classification results. The forward-acting timing recovery method was adequate for recovering the signal constellations, and can be verified visually from figures 75 and 76. By using the computationally complex method of classification that involved computing the mean minimum distance from all observed symbols to all possible ideal constellations, 100% correct classification was achieved for all V.29 signals.

The ideal constellations for V.29 signals are given in the ITU standard document. Refer to Appendix R for MATLAB code that was used to supervise V.29 signal classification.

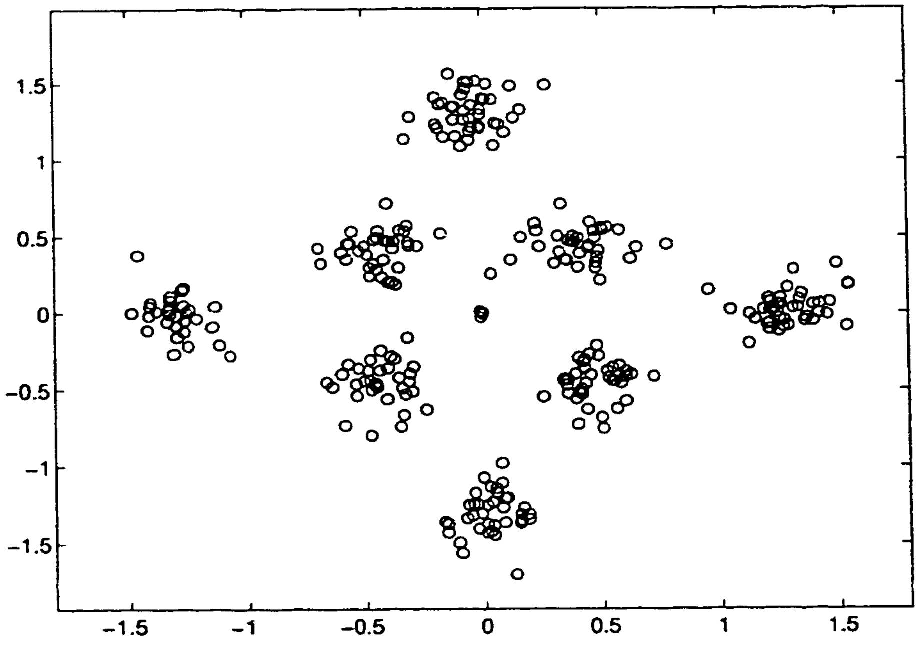


FIGURE 75. Recovered scatter diagram of an 8 point V.29 constellation.

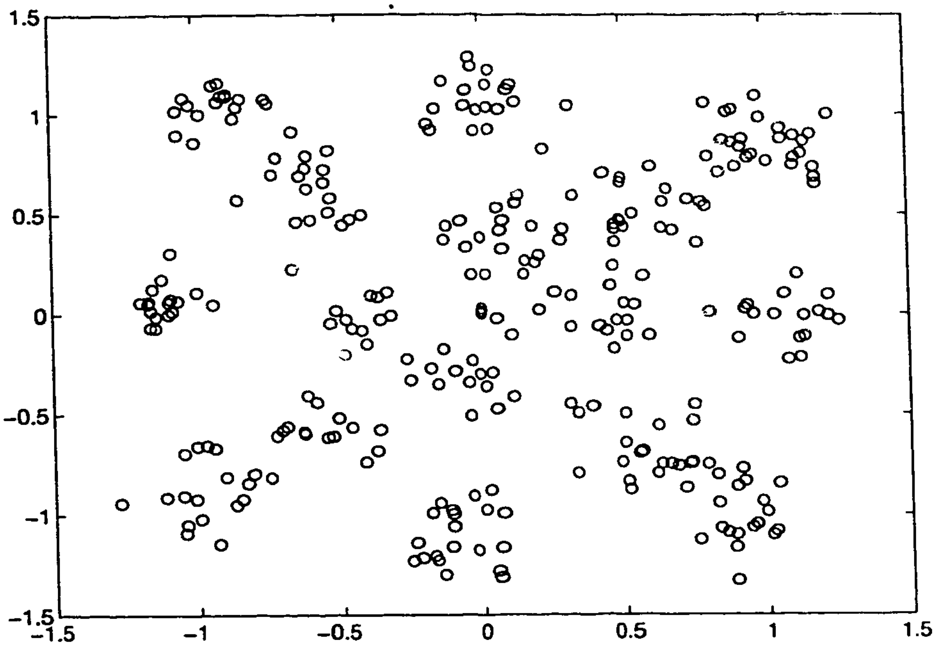


FIGURE 76. Recovered scatter diagram of a 16 point V.29 constellation.

The scatter plots from Fig. 75 and Fig. 76 illustrate the results that can be achieved by using this method to recover a signal's constellation pattern. The signals used for these tests were moderately distorted. The plots show the results after monitoring 1024 samples (1/8 second) from each signal being monitored. Each 'o' mark represents a symbol from the received signal. From the diagrams it is clear that the signals have been sufficiently demodulated and sliced to allow for identification of the constellation type. The equivalent plot for V.32 signals has no visible pattern due to the high probability of symbol error in an unequalized, noisy channel.

In conclusion we can state that by using a nominal carrier frequency for demodulation, forward-acting spectral line timing recovery, and some type of classification method, we can correctly identify the QAM constellation in use by V.29 signals in typical and worst case channel conditions. For signals with constellations that contain greater than 16 symbol points other methods of extracting the constellation must be explored. Perhaps the trellis-coding present in the transmitted signal could be exploited by the classifier detector to overcome the low SNR. These methods must reduce the probability of symbol error in the received signal.

6.2 Other Methods

We conclude this chapter by mentioning a few other methods that were evaluated but omitted in our final classifier design. However, the methods may still be useful, depending upon the needs of the classifier. These methods have been verified with the MATLAB simulation model shown earlier.

6.2.1 Zero-Crossing Monitoring for Detecting Carriers

Zero-crossing detection is a very simple process to implement. Simply, every time the signal crosses zero, it is recorded in a counter. Then, after a predetermined sample period, the counter value is divided by the time elapsed giving a number in Hz. This number can be used as a crude estimate of the carrier frequency of a signal, providing one is present. If this number is constant (within some degree of accuracy), or averages to a constant over many blocks, from one time frame to another, then it can be safely assumed that there is either a carrier or constant tone present in the signal. Simulations have shown that for standard FAX and data modulation rates, the zero crossing method will indicate the carrier frequency within 2% accuracy. This method may be useful for a quick evaluation of whether or not a carrier is present, and at what frequency. Since standard modulation methods use only a small variety of carrier frequencies, it is not difficu t to reliably guess which is in use.

6.2.2 Delta Phase Histogramming

"Delta phase histogramming" is a name we have given to a method that involves recording the phase changes in a signal with a histogram. Since QAM signals vary in both phase and amplitude, monitoring the phase changes exploits one source of information regarding the signal. In the simulation, the estimated carrier frequency obtained via zero crossing detection is used to frequency shift the incoming signal to the baseband. Then the signal is filtered and further studied. Since the baseband signal is complex-valued, it is split into its phase and magnitude components. Then, the phase change distances are measured (in radians) and recorded in a histogram. The expected result is to see that there are higher frequencies of phase changes corresponding to the signal constellation points. Consider a 4-point QAM signal. The minimum phase change from one constellation point to another is $\pi/2$. Thus, it is expected that there will be a peak in the delta phase histogram at $\pi/2$ radians. Also, there should be peaks at 0, π, and $3*\pi/2$. In fact, this has been verified by simulations. Even though the estimated carrier frequency used to generate the baseband signal was usually slightly inaccurate, for low-speed signals the delta phase histograms do yield some information which can be viewed as a signature of the signal under study.

An important feature of the delta-phase histogram method is that it is not dependant upon the magnitude of the signal being studied. Because of this, the histogram will never need to be scaled to correspond to an expected structure. Also, it should be noted that this method's attractiveness breaks down at high data rates (32 constellation points or greater) and with the introduction of significant noise. Noise within the system introduces small phase changes which confuse the issue. However, the method still somewhat useful for identifying low speed signal constellations such as V.22*bis*.

The next sequence of figures shows the resulting histograms from performing delta-phase histogram analysis on four different signals. The four signals include 2400 bps V.27*ter*, 4800 bps V.27*ter*, 7200 bps V.29, and 9600 bps V.29 which utilize 4-PSK, 8-PSK, 8-QAM, and 16-QAM modulation methods, respectively. Each signal contains one second of real-time signal recording. The top histogram is generated by the angle of the approximately demodulated signal recording. The bottom histogram is generated by the delta-phase changes observed in the approximately demodulated signal.

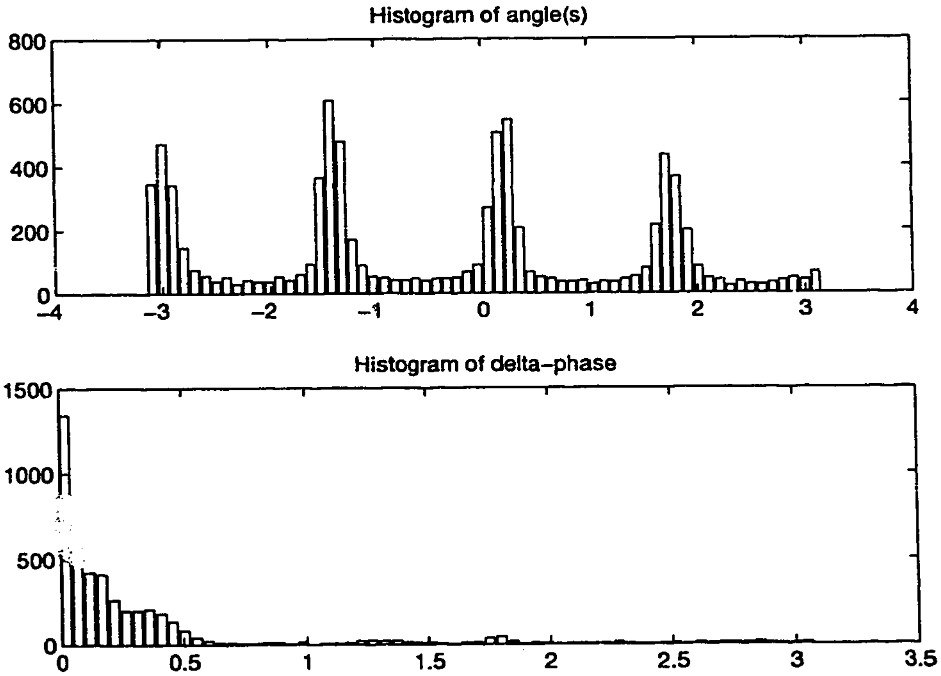


FIGURE 77. Histograms of a 2400 bps V.27*ter* signal.

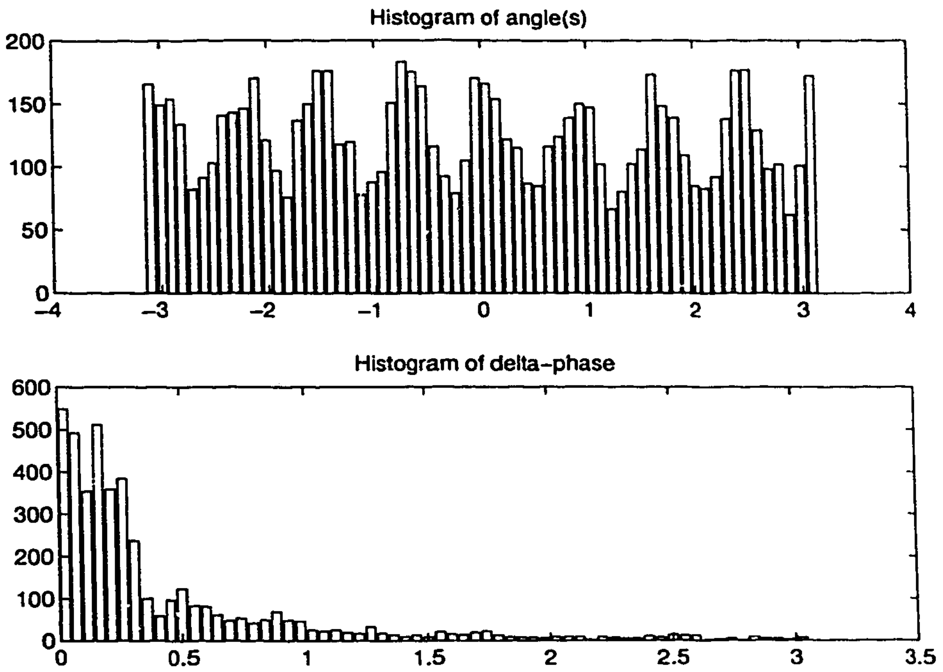


FIGURE 78. Histograms of a 4800 bps V.27*ter* signal.

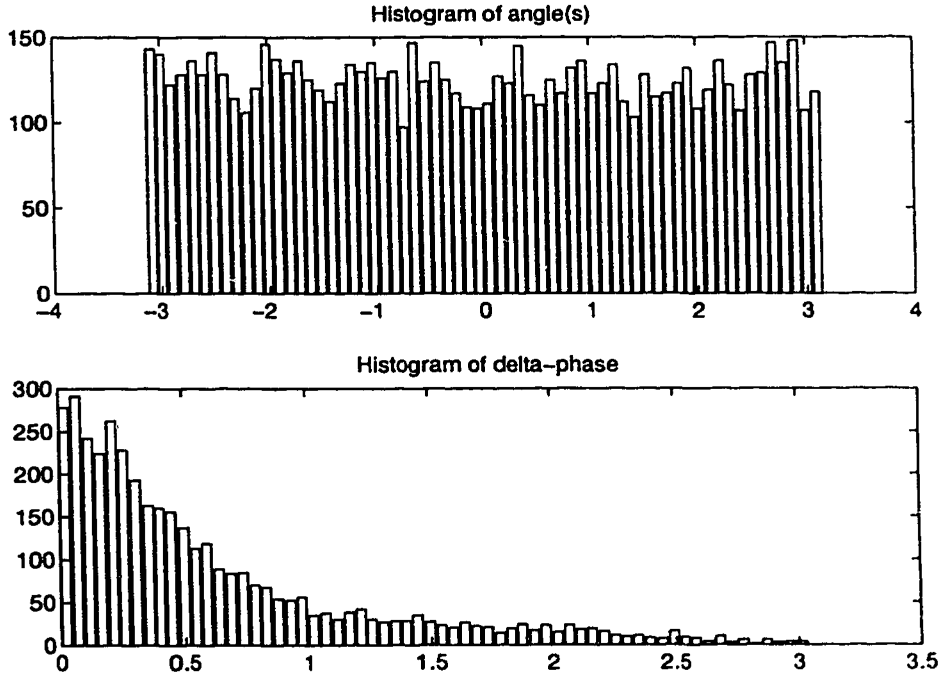


FIGURE 79. Histograms of a 7200 bps V.29 signal.

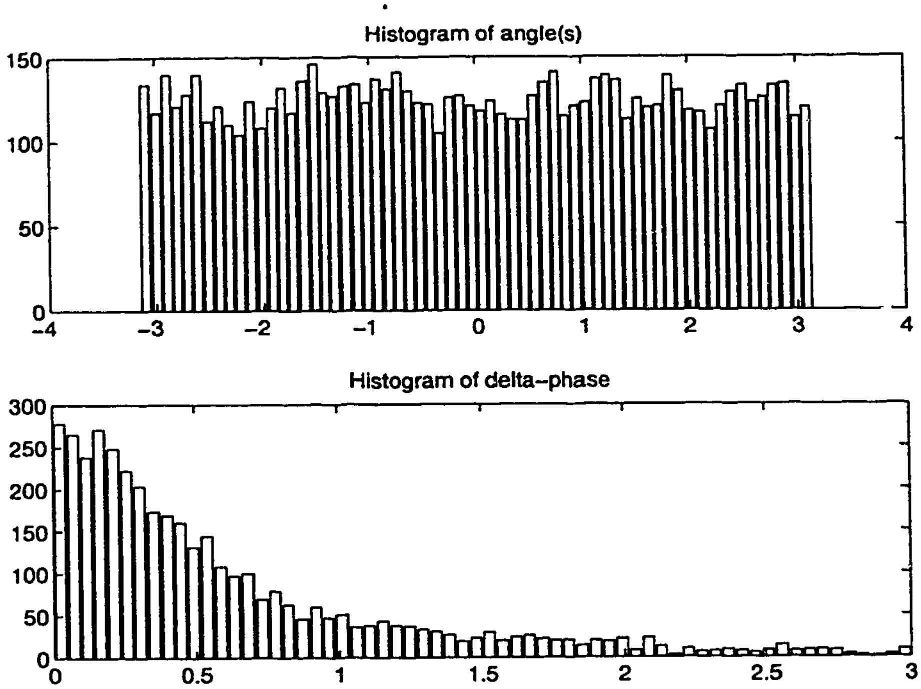


FIGURE 80. Histograms of a 9600bps V.29 signal.

The figures show that the delta-phase method reveals some information about low speed FSK, and even QAM signals. However, as the complexity of the signal constellation is increased, the effectiveness of the delta-phase method diminishes. This method is not useful for classifying the constellation patterns of practical signals including V.32*bis*.

Chapter 7

7.0 Design Recommendations and Summary

In this chapter we discuss classification system design considerations, and recommend an approach for constructing a complete system. In particular, we recommend a signal class structure, a set of discriminant variables, a set of discriminant functions, a classification method, and finally we propose a complete classification system architecture.

7.1 Class Structure

Determining the class structure to be used is a critical first step. The class structure will be based on the needs of a particular application as well as the capabilities of the best discriminant variables available. We assumed earlier that the classification system was to discriminate as many different classes as possible. Given that the best discriminant variables available were based on the normalized central second-order moment and the autocorrelation sequence, we formed a class structure where every signal class has a unique power spectral density pattern. The class structure that we recommend is shown in Table 43.

TABLE 43. VBD, FAX, and speech subclassification classes.

Group No.	Signals included
1	V.22 and V.22*bis* forward channels
2	V.22 and V.22*bis* reverse channels
3	V.34 at speeds greater than 14.4 kbps
4	V.29 all speeds
5	V.32, V.32*bis*, and V.17 at speeds greater than 2,400 bps
6	V.27*ter* at 4,800 bps
7	V.27*ter* at 2,400 bps

TABLE 43. VBD, FAX, and speech subclassification classes.

Group No.	Signals included
8	speech
9	random PCM samples

The classes shown in Table 43 may be easily extended or abbreviated to suit a particular application. For example, FSK signals such as V.21 may be added to the table. Any signal type that has a unique power spectral density may be included as a separate class. However, the nine classes shown are representative of the bulk of VBD communications that one would expect to observe in the PSTN today.

By reducing the number of classes considered, the total complexity of the classification system can be reduced. For example, fewer discriminant variables may be required, simpler discriminant functions may be used, and less memory will be needed to store the required classification statistical information (such as probability density functions for each variable and class).

7.2 Discriminant Variables

Once the class structure is determined, we can select a good set of discriminant variables. The results in Chapter 5 indicated that the choice of discriminant variables to utilize in the classifier is highly dependant upon the classes to be discriminated. We assumed in Chapter 5 that the classes shown in Table 43 were to be resolved. (Simpler class structures require significantly fewer discriminant variables, can operate using smaller values for N, and will have better accuracy.) To achieve a classification rate of at least 97% for example, we have several options available. We can select a small subset of discriminant variables {Rd2, Rd4, N2} and use long signal segments ($N >= 2048$ for pseudo-quadratic discriminant functions or $N > 2048$ for linear discriminant functions), or we can use a larger set of discriminant variables {Rd1, Rd2,..., Rd10, N2} with a small segment length (N >= 512 for pseudo-quadratic discriminant functions or N >= 2048 for linear discriminant functions). Of course there are other combinations that can be used as well. We determined that minimizing the computational requirements of the classification system was a more important goal than the shortest possible response time. Therefore, we recommend a small set of discriminant variables {Rd2, Rd4, N2}, and a comparatively long segment length. Recall from Chapter 5 that this set of discriminant variables was found heuristically to perform well at both classifying VBD subclasses, and to distinguish between speech and VBD signals. Specific determination of the segment length required can only be done once the discriminant functions have been chosen.

Other parameters pertaining to the calculation of the discriminant variables must also be chosen, including the subsegment length L and the average power threshold P_{Th}. The best values to use were experimentally determined to be $L = 16$ and $P_{Th} = 1089$ (units are the minimum Q size from the μ-law table in order to correspond to the μ-law decoder output values). These values achieve a satisfactory trade-off between noise/silence rejection and good utilization of speech segments. The power threshold may be increased if only VBD signals are to be classified.

7.3 Discriminant Functions

Either linear or quadratic discriminant functions may be used by the classifier. If linear functions are used, the classifier complexity will be low, but long segment lengths (e.g. 1024 samples) will be required to achieve a satisfactory classification accuracy (such as 97%). Quadratic discriminant functions are more costly to compute since they are second order discriminant variables, but classification accuracy is improved in some classes and shorter segment lengths (e.g. 256 samples) may give satisfactory accuracy.

We note from the results presented in Chapter 5 that classification performance is substantially improved by using quadratic discriminant functions for only two classes out of nine. Taking this into consideration, it does not seem worthwhile to increase the classifier complexity dramatically by using quadratic discrimination functions. Therefore, we propose a stepwise classification method using linear discriminant functions at a first stage and quadratic discriminant functions at a second stage. The class structure presented in section 7.1 must be modified as shown in Table 43 to merge classes four and five.

TABLE 44. VBD, FAX, and speech subclassification classes, stage one.

Group No.	Signals included
1	V.22 and V.22*bis* forward channels
2	V.22 and V.22*bis* reverse channels
3	V.34 at speeds greater than 14.4 kbps
4, 5	V.29 all speeds, V.32, V.32*bis*, and V.17 at speeds greater than 2,400 bps
6	V.27*ter* at 4,800 bps
7	V.27*ter* at 2,400 bps
8	speech
9	random PCM samples

Stage one of the classification system will use three discriminant variables {Rd2, Rd4, N2} and linear discriminant functions. In Table 45 we present the classification performances achievable using the new class structure and the three discriminant variables. Referring to Table 45 we see that the segment length required to achieve a classification performance of 96.13% is N >= 1024. If the classification result from stage one states that the observed signal belongs to class four, then stage two is invoked. Stage two of the classifier uses the same discriminant variable set, but a quadratic discriminant function is now used to resolve classes 4 and 5. A quadratic system based on these variables and $N = 1024$ can achieve a classification performance of 84.70% for the two remaining classes. With this two-stage method we retain the low complexity of linear discriminant functions, and the benefits in classification accuracy of quadratic discriminant functions.

TABLE 45. Classification performances of a two-stage classifier {Rd2, Rd4, N2}.

Classification Accuracy P_c	$N = 256$	$N = 512$	$N = 1024$	$N = 2048$
Stage 1	82.00%	94.23%	96.13%	97.18%
Stage 2	68.52%	81.14%	84.70%	87.74%

From Table 45 we can see that even with large values of N, the classification performance of stage two is still poor (less than 90%) if only three discriminant variables are used. We consider this to be an unsatisfactory performance level. In order to improve the accuracy of stage two, we searched for another discriminant variable to add to the system. Table 46 presents the discriminant variable ranking when only considering classes 4 and 5 of the original class structure. The criteria for discriminant evaluation was to maximize the minimum Mahalanobis distance between the groups. From the table we see that Rd6 is the top ranked variable. Now we can test stages one and two of the system with the new discriminant variable added.

TABLE 46. Discriminant variable ranking for classes 4 and 5.

Ranked Discriminant Variables (using Mahalanobis distance for variable evaluation)
Rd6
Rd10
Rd5
Rd2
Rd4
Rd9
Rd1
Rd7
Rd8

TABLE 46. Discriminant variable ranking for classes 4 and 5.

Ranked Discriminant Variables (using Mahalanobis distance for variable evaluation)
N2
Rd3 was not included

The results from testing the two-stage classifier using the enlarged discriminant variable set {Rd2, Rd4, Rd6, N2} are shown in Table 47. From the table we can see that stage one of the classifier is not substantially improved with the addition of the Rd6 variable. Stage two, however, sees a dramatic improvement in classification accuracy at all segment lengths. Considering these results, we propose that a two-stage classifier use segment lengths $N = 2048$, subsegment lengths $L = 16$, power threshold $P_{Th} = 1089$, the class structure shown in Table 43, and the discriminant variable set {Rd2, Rd4, Rd6, N2}. Tables containing all of the standardized canonical discriminant function weights, Fisher's linear discriminant function coefficients, as well as a detailed listing of class-wise classification performances are included in Appendix S. Note that the Fisher's linear functions are only applicable for the linear classification stage.

TABLE 47. Performance of two stage classifier {Rd2, Rd4, Rd6, N2}.

Classification Accuracy P_c	$N = 256$	$N = 512$	$N = 1024$	$N = 2048$
Stage 1	84.92%	96.19%	97.74%	98.27%
Stage 2	75.05%	94.89%	98.16%	99.54%

We also propose an alternative to the two stage classification scheme which is simpler to implement. In this scheme only linear discriminant functions are used, along with all of the available discriminant variables. The argument here is that the discriminant variables can be efficiently implemented in a DSP that has multiply-and-accumulate (MAC) instruction that executes in a single processor cycle. These MAC operations form the core of the autocorrelation lag calculations. By calculating a few more autocorrelation lags for discriminant variables we can achieve a high classification accuracy using only linear discriminant functions. Thus an easily implemented classifier can be constructed using several autocorrelations lags and the normalized central second-order moment as discriminant variables, and simple linear discriminant functions. A further simplification in this scheme is to use Fisher's linear discriminant functions since they are easy to directly apply to the available discriminant variables.

We presented the performances achievable using all eleven discriminant variables {N2, Rd1, Rd2,..., Rd10} and linear discriminant functions in Chapter 5. As a starting point for an easily implemented classifier, we also include the Fisher function coefficients for the cases where N = {256, 512, 1024, 2048} and all nine classes are considered. The function coefficients are included in Appendix T.

7.4 Classification Method

The classification method is the technique used to allocate an observation into one of a set of classes. Bayes' Rule is optimal for performing this function. The operations required to implement Bayes' Rule are:

1. Calculate the discriminant variables.
2. Calculate the linear or quadratic discriminant functions using the variables.
3. For each function calculate the posterior probability of class membership for each class using Bayes' theorem. Extra information required to use Bayes' theorem includes the a prior probabilities of class membership (which may be assumed to be equal for all classes) and the probability density functions for each function in each class.
4. The observation is then allocated to the class with the highest a posteriori probability of membership.

The a posteriori probabilities form an indication of confidence in the classifier output. This information may be unnecessary in a particular application. If this information is not required, simple decision boundaries may be calculated from the discriminant function statistics or Fisher's linear functions can be used. We consider the a posteriori probabilities to be valuable in the classification system, and so they are retained in the recommended classifier architecture.

A requirement for using Bayes' Rule is that the probability density functions (PDF) for every variable be known for each class. This is why a database is included in the system block diagram in Fig. 81. The database holds all of the PDFs for each variable and class. Alternatively, the PDFs for each discriminant function may be stored. (In the case of linear discriminant functions, it is equivalent to use the function PDFs or the variable PDFs.) In the proposed system where there are four variables and nine classes, as many as 36 PDFs must be stored for stage one alone in order to use Bayes' rule for classification. The number of bins in the probability mass function (discrete version of the PDF) is variable. If 256 bins were used, we would require a 9216 element matrix of values to store all of the PDFs. (In the MATLAB implementation of Bayes' Rule classification, there were 256 bins used to store each PDF. SPSS internally uses a chi-square distribution fit to the actual PDFs.) In this thesis we have not included the specific PDFs that were obtained from analyzing the training data. A prohibitive amount of paper space would be required to record each PDF (or probability mass function since we are dealing with discrete systems) in a table. A critical phase of the development of a classification system is training. The data used to train a classifier must be carefully selected to represent the test population well. The process of training involves calculating the discriminant variables for a large set of training signals, recording the PDFs and variable covariances. By referring to the suggested system design proposed here, one can develop a classifier system and accumulate the necessary statistical information to utilize Bayes' Rule for classification. If one was to use the specific discriminant function coefficients described here, and their corresponding PDFs, in a specific classifier implementation they should not expect to discover the same operating classification accuracy without performing training. Training must be performed using a set of signals representative of the population to be classified. Even though every attempt has been made in this thesis study to use accurate signal models for classifier training, it is possible that there will be differences in the actual classifier environment.

One alternative to using Bayes' rule is to simply use decision boundaries. These boundaries may be formed by observing the PDFs of the discriminant functions. Decision boundaries require less storage space than complete PDFs; however, the precision and optimality of Bayes' Rule is lost in the classification. This is a trade-off that must be made by the designer.

A final alternative to using Bayes' theorem for performing class allocation is to use Fisher's linear discriminant functions. Fisher's functions are a combination of optimal linear discriminant functions and Bayes' rule for classification. The resulting functions can be applied directly to the discriminant variables, where a function is defined as a linear combination of discriminant variables for every class. The class with the highest function value for a particular observation is the class that the observation is allocated into. This method is simple to implement, however the posterior probability of group membership is not available from Fisher's functions. The classification accuracy is identical for Fisher's functions and optimal linear functions classified using Bayes' theorem.

7.5 Complete S·:stem

A complete classification system block diagram is shown in Fig. 81. The system parameters we propose include a two stage classifier using segment lengths $N = 2048$, sub-segment lengths $L = 16$, power threshold $P_{Th} = 1089$, the class structure shown in Table 43, Bayes' Rule for class allocation, and the discriminant variable set {Rd2, Rd4, Rd6, N2}. The second stage of ·he classifier discriminates between V.29 signals and V.32/V.32*bis* signals in class four of Table 43. The resulting average (over all classes) classification performance is expected to be 98.27% and 99.54% for stages one and two, respectively. Note that the stage two classification performance only refers to the separation of classes 4 and 5.

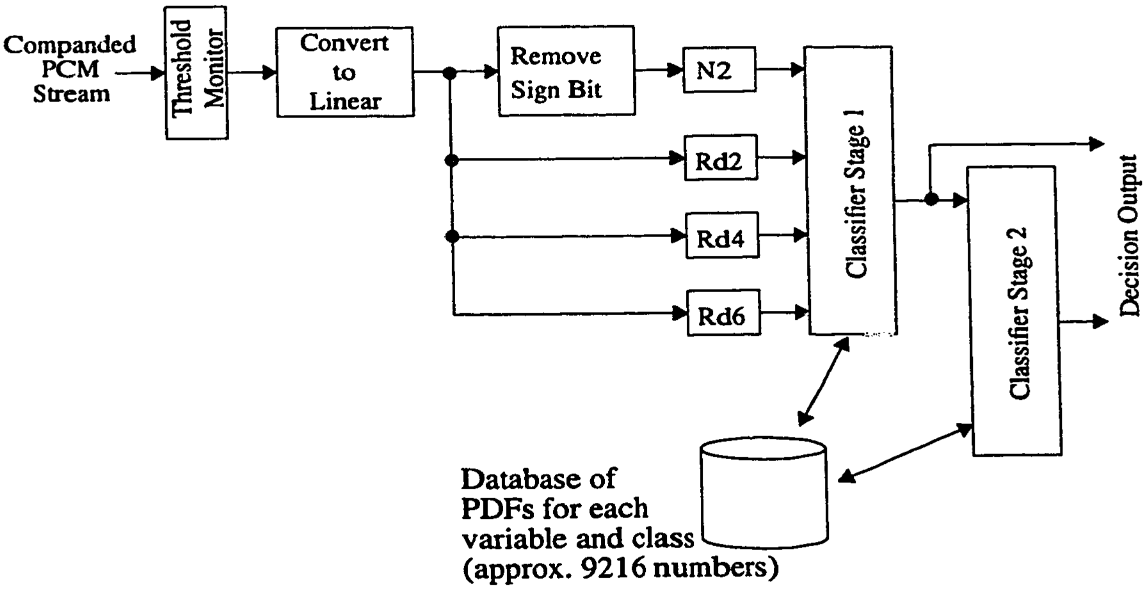


FIGURE 81. Recommended classification system.

In section 7.3 we also referred to an alternative classification scheme that could be easily implemented using Fisher's linear discriminant functions. Figure 82 illustrates the architecture of that scheme, where the parameters are the same as for the two-stage scheme previously mentioned. The advantage of this scheme is that the classification stage is very simple to implement since Fisher's linear discriminant functions are used. The variable or function PDFs are no longer needed to perform classification. The disadvantages are that more discriminant variables must be calculated to achieve a comparable classification accuracy, and the classifier cannot provide the posterior probability of group membership for each classification decision. The disadvantage of having to calculate a few more autocorrelation lags for discriminant variables may turn out to be insignificant in an actual implementation of the classifier. The reason for this is that each additional autocorrelation lag only requires a single MAC instruction per sample to compute. This extra computation may prove to be insignificant compared to the overhead processing required to run the real-time classifier.

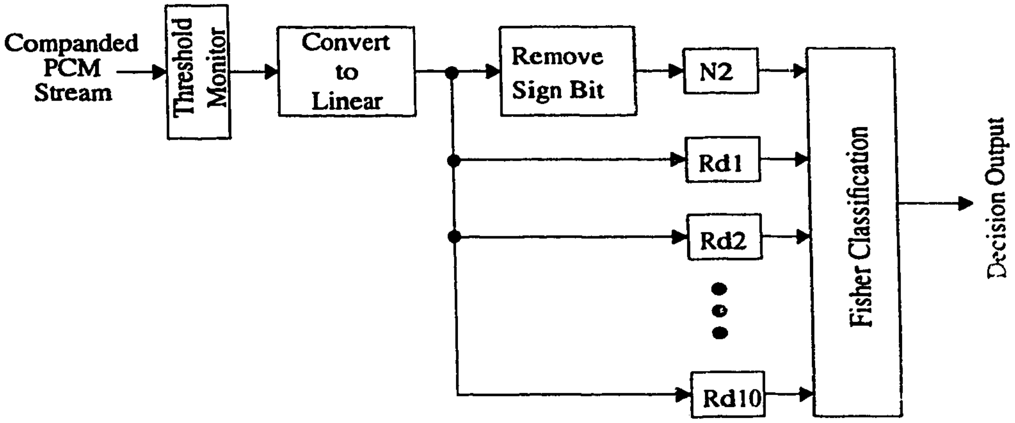


FIGURE 82. Alternative recommended classification system.

7.6 DTMF Signals

An important type of non-speech signal that is used in the network are Dual-Tone Multi-Frequency (DTMF) signals. These signals are transmitted through the network when a touch-tone telephone button is depressed during a call. Each tone is actually a combination of two different sinusoidal tones. The PSD of each of the 12 DTMF signals is unique, and hence they could be discriminated using the currently recommend classification system. However, DTMF signals can be as short as 50 ms, corresponding to 400 PCM samples. If we are to be able to detect the minimum allowable DTMF tones, the classifier segment length N can be no greater than 400 samples. If we compare this segment length to the results in Table 47 we can interpolate and estimate the classification performance to be about 90% for each stage with the given discriminant variables. In order to improve this performance we need to increase the number of discriminant variables. Referring to Fig. 61 in section Fig. 5.3.7.5 we can see that to achieve a high classification accuracy for $N = 400$, we will need to use at least 11 discriminant variables as well as quadratic discriminant functions.

Recall that the complexity of calculating a quadratic discriminant function is order T^2, where T is the number of discriminant variables. Linear discriminant functions are only order T. Therefore, we would like to avoid the use of quadratic discriminant functions where several variables are involved, especially when the segment length is short. (The shorter the segment length, the more often decisions must be made.) We conclude that the benefits of using the existing classifier to guarantee classification of minimum length DTMF signals are outweighed by the increased computational cost. We suggest that a secondary method may be the best choice for detecting DTMF tones.

A second alternative for DTMF tone detection exists. The proposed system uses a fixed-length signal segment for the basis of discriminant variable calculation. If we allow the length of a signal segment to vary up to a maximum length, N, then we can construct an algorithm that would be able to "catch" minimal length DTMF tones. This algorithm would be more complex, but the final classification system would be able to retain its low complexity discriminant variables and functions while also being able to classify all DTMF signals. Design of this algorithm is left to future work. It is also possible that the simplest solution is to use off-the-shelf DTMF detection circuits.

7.7 Real-Time Classification

We claimed earlier that the designed system would be suitable for real-time classification. Referring to the suggested classifier architecture shown in Fig. 81, we can estimate the number of calculations that a required for full-time monitoring of a single channel. The first block contains a threshold monitor. In the worst case, the threshold monitor will have to compare every new sample to a given threshold. This would require one operation per sample. If traffic is present, an absolute level threshold does not need to be compared to for every sample. When signal segments are being accumulated, the zeroth lag of the autocorrelation sequence is used as an estimate of the average power level of a sub-segment. The computations needed for this estimator will be included in a later part of the classifier. We need to now account for the comparison that is performed between a power threshold and the estimated average power, once every sub-segment interval (which is 16 samples in this design). Next we consider the μ-law to linear conversion stage. In the best case, a simple table look-up will be performed, requiring one operation per sample. If this stage is poorly designed it may require several more operations per sample. Sign bit removal requires one operation per sample.

Computation of the discriminant variables is next. We can refer to the original equations for these variables. Note that the zeroth lag must also be computed in order to normalize the autocorrelation lags. Referring to [67] we see that the normalized central second-order variable requires two additions and one multiplication operation per sample. After each sub-segment is accumulated, a division and a subtraction are required. To calculate the autocorrelation lags we require one multiplication and one addition per sample. After a sub-segment has been accumulated another division is required. Since we require three lags to be computed as well as the zeroth lag, we can multiply these operations by four. Finally, one operation is required to normalize each autocorrelation lag at the end of a sub-segment. We can now add up the required operations prior to the classification stages. The total calculations required per sample up to this point is:

$$(1+1/16+1+1)+(3+1/16+1/16)+(1+1+1/16)4+3(1/16) = 14.625 \text{ operations/sample}$$

Now we can consider the cost of performing classification. Stage one of the classifier requires four variables, and can classify eight classes using linear discriminant functions. Since there are four variables, there will be at most four functions. Each function requires $2T$ operations, where $T = 4$ (the number of variables). Stage two of the classifier requires only one quadratic function since there are only two classes. This function requires $2T^2+3T-2$ operations, where T is four again. In total, to compute the discriminant functions we require:

$$2(4)+2(4)^2+3(4)-2 = 50 \text{ operations/segment}$$

Note that these 50 operations are only required after a complete segment has been accumulated. In the suggested design the segment length is 2048 samples, and therefore the number of operations per sample required to compute the discriminant functions is 0.024.

Next, classification must be performed. This can either be done using Bayes' Rule, or some other simpler method. Since these operations are again only required once per segment, they will not heavily affect the overall system load. The most significant contribution to the system load is the computation of the discriminant variables.

We now estimate that approximately 15 operations per sample are required to perform classification of a single voice-band signal. Since the sample rate is 8000 Hz, we conclude that 120,000 operations per second must be performed if the suggested system is used. Let us assume that this is an extreme lower bound. (In fact, if we use a modern DSP such as the TMS320C30 from TI, additions and multiplications can often be performed in a single processor cycle. This could actually reduce the estimated number of processor cycles to less than 15.) Let us say that an actual implementation requires ten times more processor cycles than the estimate, due to unforeseen processing overhead. We then increase our estimate to 1.2 million operations per second. A TMS320C30 processor can perform 40 million operations per second, and therefore classify 33 voice-band channels full-time, in real-time, simultaneously. A standard E1 signal (European equivalent to T1) carries 32 voice-band channels. We conclude that a single TMS320C30 processor implementing the suggested classifier design could classify every channel carried by an E1 or T1 trunk, in real-time. (The upper-bound on the number of channels that could be monitored is 330 according to our estimates.)

Note that 208 processing cycles are available per received PCM sample from a T1 signal when a 40 MHz TMS32C30 DSP is used. We have estimated that 15 operations are required per PCM sample using the recommended two-stage classification architecture, which would leave 193 cycles per PCM sample for processing overhead (assuming each required operation executes in a single cycle). If we increased the number of discriminant variables to eleven, the required classifier operations would increase to about 40 per sample, leaving about 168 operations per sample unused. This is a 13% reduction in the cycles available for overhead processing. The point here is that it is quite probable that there will be enough excess processing cycles in a T1-based classifier to compute several autocorrelation lags and use simple linear discriminant functions.

7.8 Field Testing

The next phase of the signal classification project is to implement a prototype system that can be used for field testing. The prototype should be flexible enough to allow the designer to easily alter the system parameters including N, P_{Th}, discriminant variable set, and discriminant function selection. The prototype system must provide a method for training the classifier. Training is performed by learning the statistics of known signal classes. Therefore, a mechanism must be in place to allow the designer to specify the class of observed traffic, and force the classifier to learn the statistical patterns of that traffic. As a first step in designing a system, the discriminant function weights that have been described in this thesis can be used. Since the signals used to generate these functions were not obtained from actual network traffic, there may be some differences. Using representative signals for training a classifier is critical to achieving good classification performance.

The prototype system must also allow for data logging. Not only should the classification results be logged, but the system load must also be monitored. A precise determination of the amount of processor cycles required to implement the classification algorithms must be known. With this information we could then extrapolate to determine the maximum number of channels that can be monitored in real-time on a full-time basis, given a particular classifier design.

A complete classification system may also have the luxury of access to call boundary information. This information could be extracted from an out-of-band signalling system such as SS7. By overlaying classifier decisions with call boundary information, a complete system could form conclusions about the type of calls that are being monitored. For example, the system could report that an entire call was VBD, or that a call began as speech and ended as VBD. This is a higher level of classification that would be useful for determining traffic usage information such as call holding times for various types of traffic.

Chapter 8

8.0 Conclusions and Future Work

In this thesis we have presented methods for classifying voice-band signals. We found that the most effective discriminant variables were those based on the autocorrelation sequence of the signal being monitored. Through analytical means we showed, however, that discriminant variables of this type are fundamentally limited in their ability to distinguish certain signals. In particular, only signals with different power spectra can be separated.

In this thesis we have presented modifications to Benvenuto's original work [6] that reduce computational complexity, improve classification accuracy, and broaden the range of VBD subclassification categories. The algorithm discussed has been shown to perform with misclassification rates of 0% for many signal groups, and on the order of 1% for two particular similar groups (V.29 and V.32/V32*bis*). Identifiable signal groups include speech and VBD that vary in their power spectral densities. Depending upon the application requirements, classification times range from 32 to 256 milliseconds for 93% worst-case accuracy (using 11 discriminant variables, 32 ms signal segments, and pseudo-quadratic discriminant functions).

In most cases linear discriminant functions were found to perform adequately. However, to distinguish some types of signals from one another, it was necessary to use quadratic discriminant functions. These functions can discriminate signals whose discriminant variables have identical mean values but differ in their covariance matrices. By using quadratic discriminant functions, signal segment lengths can be reduced while retaining the same classification performance. For example, using linear discriminant functions on eleven variables generated from 2048 sample long segments, we can achieve a classification performance of 98% over nine classes. Alternatively, if a pseudo-quadratic discriminant function is used, the signal segment length can be reduced to 512 samples for a classification performance of 100%. If the signal segment length remains constant, we can reduce the number of discriminant variables from eleven to three by switching from linear discriminant functions to pseudo-quadratic functions, and still attain the same classification performance.

A complete classification system was proposed. The proposed system requires four discriminant variables to be computed including {Rd2, Rd4, Rd6, and N2}. Classification is performed in two stages, the first of which uses linear discriminant functions, and the second stage uses quadratic discriminant functions. Stage one classifies the observed signal into one of eight different classes. If stage one allocates an observation into the fourth class, stage two is invoked to refine the classification into one of two subclasses. This two-stage configuration retains the high classification accuracy of quadratic discriminant functions while still keeping the number of discriminant variables required and algorithm complexity at a minimum. The resulting system has an overall classification performance of 98.27% for stage one and 99.54% for stage two, using segments 256 ms long.

Other classification techniques were developed. Notably, a "sniffer" program was described that detects and decodes the FSK signaling used by facsimile machines. Other methods that we developed included timing and carrier recovery, amplitude and phase histogramming, and statistical signal characterization.

The problem of classifying DTMF signals was briefly mentioned. Since DTMF signals have clearly different PSDs we can classify them using autocorrelation sequence based discriminant variables. However, the minimum length of a DTMF signal is only 50 ms. This length may be shorter than the fixed classifier segment length *N*. To allow the proposed classification system to monitor minimum length DTMF signals, the algorithm for collecting signal segments must be altered. The signal segment length must be made adaptable. The effects that an adaptable segment length algorithm has on classification have yet to be studied. The complexity of this new algorithm also needs to be examined.

The next phase of the classification project should include field testing. All of the results and analysis presented here were obtained from simulated and recorded telephone calls. In the future, real-time classification of actual PSTN traffic should be performed to verify the algorithms. Also, a prototype system would allow a precise determination of the processing requirements for various classifier designs. Once these requirements are known, we can determine the number of voice-band channels that may be monitored simultaneously by a single processor.

To subclassify VBD signals that differ only in their constellation patterns, alternative methods must be explored. By identifying the constellation pattern that is in use by a QAM signal we could precisely determine the bit rate and operational parameters of the transmitter. Due to the highly complex constellation patterns that are in use (up to 960 points for V.34), line equalization must be performed to reduce the probability of symbol error in order to make the monitored signal constellation pattern resolvable. Line equalization involves filtering a received signal to compensate for the transmission channel distortions. In the open literature there are methods of equalization that both equalize the channel and lock the phase of the receiver without requiring a training sequence. These methods, known as *blind equalizers*, are based on higher-order cumulants. N.K. Jablon has shown that by using Godard-based blind equalizers it is possible to identify the constellation in use by V.32bis signals [42]. References [9, 15, 17, 22, 23, 24, 25, 26, 43, 44, 45, 52, 59, 60, 66, 73, 76, 77, 78] all pertain to the many methods of blind equalization that have been developed and analyzed. Benvenuto has also published a paper regarding the use of Godard based blind equalizers in VBD signal classification [5]. Future work on the classification problem should investigate applying modern blind equalization algorithms to the problem of QAM constellation identification in voice-band channels.

The following list outlines several possible future projects:

1. Develop software for a prototype system. The hardware should be "off-the-shelf" to allow for a short design cycle and the use of well developed design tools. Software development can be partitioned into two sub-projects. These are listed next.
2. Develop host processor supervisory and control software. This software will be responsible for managing the flow of data through the prototype hardware. User commands must be interpreted and useful classification results displayed as well as logged to files.
3. Develop DSP software for calculation of the discriminant variables and performing classification. The classification results are periodically transferred to the host processor for logging and interpretation
4. Field trials must be performed with the prototype unit. Once all of the software has been developed, the test system must be trained and tested using signals obtained in real-time from actual networks.
5. Develop algorithms for performing the classification of VBD signal constellations. This research will likely involve implementing a blind equalization algorithm and determining a method for matching observed constellation scatter diagrams to a set of expected maps.

The original research proposal stated that even in the worst case, where no improved solution to the problem were to be found, at least a better understanding of the problem would be gained. The best case research result was to develop a solution better than those previously published and then to implement it in a combination of hardware and software. Another possible project conclusion was to enhance existing classification hardware by improving its performance and accuracy. The actual outcome of the project is a combination of the later two predictions. Published algorithms were modified and improved, implemented in an off-line capacity, and evaluated through detailed simulations. Finally, recommendations were developed for a prototype classifier to be developed as a subsequent project.

Chapter 9

9.0 References

[1] J. Aisbett, "Automatic Modulation Recognition Using Time Domain Parameters," Signal Processing, Vol. 13, October 1987, pp. 323-328.

[2] AT&T Network Systems, AT&T Voice/Data Call Classifier information pamphlet.

[3] Bellamy, John, "Digital Telephony", 2nd Edition, John Wiley and Sons, 1991.

[4] M. Ben-Bassat, "Use of Distance Measures, Information Measures and Error Bounds in Feature Evaluation," Handbook of Statistics, Vol. 2, North-Holland Publishing Co., 1982, pp. 773-791.

[5] N. Benvenuto, "Classification of Voiceband Data Signals Using the Constellation Magnitude," IEEE Transactions on Communications, Vol. 43, No. 11, November 1995, pp. 2759-2770.

[6] N. Benvenuto, "A Speech/Voiceband Data Discriminator," IEEE Trans. on Commun., Vol. 41, No. 4, April 1993, pp. 539-543.

[7] N. Benvenuto, W.R. Daumer, "Classification of Voiceband Data Signals," Proc. ICC., Atlanta, GA, April 1990, pp. 1010-1013.

[8] N. Benvenuto, "Detection of Modem Type and Bit Rate of FSK Voiceband Data Signals," IEEE ICC., Vol. 2, 1989, pp. 966-970.

[9] A. Benveniste, M. Goursat, "Blind Equalizers," IEEE Transactions on Communications, Vol. COM-32, No. 8, August 1984, pp. 871-883.

[10] D. Bertsekas, R. Gallager, "Data Networks", second edition, Prentice Hall, New Jersey, 1992.

[11] Bingham, John A.C., "The Theory and Practice of Modem Design", John Wiley and Sons, 1988.

[12] M. B. Carey, H. T. Chen, A. Descloux, J. F. Ingle, and K. I. Park, "1982/83 End Office Connections Study: Analog Voice and Voiceband Data Transmission Performance Characterization of the Public Switched Network," AT&T Bell Laboratories Technical Journal, Vol. 63, No. 9, November 1984, pp. 2059-2119.

[13] S. Casale, C. Giarrizzo, A. La Corte, "A DSP Implemented Speech/Voiceband Data Discriminator," IEEE Global Telecommunications Conference and Exhibi-

tion, Communications for the Information Age, IEEE, New York, 1988, pp. 1419-1427.

[14] Y.T. Chan, L.G. Gadbois, "Identification of the Modulation Type of a Signal," Signal Processing, Vol. 16, February 1989, pp. 149-154.

[15] S. Chen, S. McLaughlin, P.M. Grant, B. Mulgrew, "Multi-Stage Blind Clustering Equaliser," IEEE Transactions on Communications, Vol. 43, No. 2/3/4, February/March/April 1995, pp. 70-705.

[16] CTel, NET-MONITOR System information pamphlet, Germantown, MD, USA.

[17] Z. Ding, R.A. Kennedy, B.D.O. Anderson, C.R. Johnson, Jr. "Ill-Convergence of Godard Blind Equalizers in Data Communication Systems," IEEE Transactions on Communications, Vol. 39, No. 9, September 1991, pp. 1313-1327.

[18] L.V. Dominguez, J.M.P. Borrallo, J.P. Garcia, B.R. Mezcua, "A General Approach to the Automatic Classificaiton of Radiocommunication Signals," Signal Processing, Vol. 22, March 1991, pp. 239-250.

[19] L.E. Franks, "Signal Theory," Rev. Ed., Dowden and Culver, 1981.

[20] Fukunaga, Keinosuke, "Introduction to Statistical Pattern Recognition", 2nd Edition, Academic Press, INC., San Diego, California, 1990.

[21] S. Geisser, "Bayesian Discrimination," Handbook of Statistics, Vol. 2, North-Holland Pub., 1982, pp. 101-120.

[22] D.N. Godard, "Self-Recovering Equalization and Carrier Tracking in Two-Dimensional Data Communications Systems," IEEE Transactions on Communications, Vol. COM-28, No. 11, November 1980, pp. 1867-1875.

[23] D.N. Godard, "Passband Timing Recovery in an All-Digital Modem Receiver," IEEE Transactions on Communications, Vol. COM-26, May 1978, pp. 517-523.

[24] D. Hatzinakos, "Blind Equalization Based on Prediction and Polycepstra Principles," IEEE Transactions on Communications, Vol. 43, No. 2/3/4, February/March/April 1995, pp. 178-181.

[25] D. Hatzinakos, C.L. Nikias, "Blind Equalization Using a Tricepstrum-Based Algorithm," IEEE Transactions on Communications, Vol. 39, No. 5, May 1991, pp. 669-681.

[26] D. Hatzinakos, "Structures for Polyspectra-Based Blind Equalizers," 1991 Military Communications Conference - MILCOM '91, September 1991, Vol. 3, pp. 1272-1276.

[27] J.E. Hipp, "Modulation Classification Based on Statistical Moments," IEEE MILCOM '86 Conf., October 1986, pp. 20.2.1-20.2.6.

[28] Hirsch, Herbert L., "Statistical Signal Characterization", Artech House, INC., Norwood, Massachusetts, 1992.

[29] IEEE Standard 743-1984, "IEEE Standard Methods for Measuring the Transmission Characteristics of Analog Voice Frequency Circuits", 1984.

[30] ITU-T Recommendation V.2, Fascicle VIII.1, Geneva, 1984.

[31] ITU-T Recommendation V.21, "300 Bits per Second Duplex Modem Standardized for use in the General Switched Telephone Network," Fascicle VIII.1, Geneva, 1984.

[32] ITU-T Recommendation V.22, "1200 Bits per Second Duplex Modem Standardized for use in the General Switch Telephone Network and on Point-to-Point 2-wire Leased Telephone-Type Circuits," Fascicle VIII.1, Geneva, 1984.

[33] ITU-T Recommendation V.22bis, "2400 Bits per Second Duplex Modem Using the Frequency Division Technique Standardized for use in the General Switch Telephone Network and on Point-to-Point 2-wire Leased Telephone-Type Circuits," Fascicle VIII.1, Geneva, 1984.

[34] ITU-T Recommendation V.27ter, "4800/2400 Bits per Second Modem Standardized for use in the General Switched Telephone Network," Fascicle VIII.1, Geneva, 1984.

[35] ITU-T Recommendation V.29, "9600 Bits per Second Modem Standardized for use on Point-to-Point 4-wire Leased Telephone-Type Circuits," Fascicle VIII.1, Geneva, 1984.

[36] ITU-T Recommendation V.32, "A Family of 2-wire, Duplex Modems Operating at Data Signalling Rates of up to 9600 bit/s for use on the General Switched Telephone Network and on Leased Telephone-Type Circuits," Electronic Document, Geneva, 1993.

[37] ITU-T Recommendation V.32bis, "A Duplex Modem Operating at Data Signalling Rates of up to 14400 bit/s for use on the General Switched Telephone Network and on Leased Point-to-Point 2-wire Telephone-Type Circuits," Electronic Document, Geneva, 1991.

[38] ITU-T Recommendation V.17, "A 2-wire Modem for Facsimile Applications with Rates up to 14400 bit/s," Electronic Document, Geneva, 1991.

[39] ITU-T Recommendation V.34, "A Modem Operating at Data Signalling Rates of up to 28800 bit/s for use on the General Switched Telephone Network and on

Leased Point-to-Point 2-wire Telephone-Type Circuits," Electronic Document, Geneva, 1994.

[40] ITU-T Recommendation T.4, "Standardization of Group 3 Facsimile Apparatus for Document Transmission," Electronic Document, Geneva, 1993.

[41] ITU-T Recommendation T.30, "Procedures for Document Facsimile Transmission in the General Switched Telephone Network," Electronic Document, Geneva, 1993.

[42] N.K. Jablon, "Joint Blind Equalization, Carrier Recovery, and Timing Recovery for High-Order QAM Signal Constellations," IEEE Transactions on Signal Processig, Vol. 40, No. 6, June 1992, pp. 1383-1398.

[43] N.K. Jablon, "Joint Blind Equalization, Carrier Recovery, and Timing Recovery for 64-QAM and 128-QAM Signal Constellations," IEEE International Conference on Communications, Boston MA, June 11-14, 1989, pp. 1043-1049.

[44] N.K. Jablon, "Carrier Recovery for Blind Equalization," Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing, May 23-26, 1989, pp. 1211-1214.

[45] N.K. Jablon, C.W. Farrow, Shao-Ning Chou, "Timing Recovery for Blind Equalization," 22nd Asilomar Conference on Signals, Systems, and Computers, Oct 3-Nov 2, 1988, pp. 112-118.

[46] F. Jondral, "Automatic Classification of High Frequency Signals," Signal Processing, Vol. 9, October 1985, pp. 177-190.

[47] H. Kobatake, K. Tawa, A. Ishida, "Speech/Nonspeech Discrimination for Speech Recognition System Under Real Life Noise Environments," Proc. ICASSP, IEEE Int. Conf. on Acoustics, Speech and Signal Processing, VI, 1989, pp. 365-368.

[48] P.A. Lachenbruch, "Discriminant Analysis," MacMillan Publishing Co., New York, 1975.

[49] M. Lang, "Allpass Filter Design and Applications," electronic document from Dept. of ECE, Rice University, Houston, TX, lang@ece.rice.edu, 1995.

[50] R.A. Law, T.W. Holm, N.B. Cox, "Real-Time Multi-Channel Monitoring of Communications on a T1 Span," IEEE Pacific Rim Conference on Communications, Computers and Signal Processing, May 9-10, 1991, pp. 306-309.

[51] Lee, Edward A., and Messerschmitt, David G., "Digital Communication", 2nd ed., Kluwer Academic Publishers, Boston, 1994.

[52] Y. Lou, "Channel Estimation Standard and Adaptive Blind Equalization," IEEE Transactions on Communications, Vol. 43, No. 2/3/4, February/March/April 1995, pp. 182-186.

[53] R.J. Mammone, R.J. Rothaker, C.I. Podilchuk, S. Davidovici, D.L. Schilling, "Estimation of Carrier Frequency, Modulation Type and Bit Rate of an Unknown Modulated Signal," Int. Conf. on Commun. '87, Seattle, WA, June 1987, pp. 28.4.1-28.4.7.

[54] G.J. McLachlan, "Discriminant Analysis and Statistical Pattern Recognition," John Wiley and Sons, Inc., New York, 1992.

[55] M.J. Norusis / SPSS Inc., "SPSS Professional Statistics 6.1," published by SPSS Inc., 1994.

[56] J.B. O'Neal, JR., R.W. Stroh, "Differential PCM for Speech and Data Signals," IEEE Trans. on Commun., Vol. COM-20, No. 5, October 1972, pp. 900-912.

[57] A.V. Oppenheim, R.W. Schafer, "Discrete-Time Signal Processing," Prentice-Hall, Inc., 1989.

[58] Pahlavan, Kaveh, and Holsinger, Jerry L., "Voice-Band Data Communication Modems-A Historical Review: 1919-1988", IEEE Communications Magazine, Vol. 26, No. 1, January 1988.

[59] C.B. Papadias, D.T.M. Slock, "New Adaptive Blind Equalization Algorithms for Constant Modulus Constellations," Proceedings of the 1994 IEEE International Conference on Acoustics, Speech and Signal Processing, Vol. 3, 1994, Piscataway, NJ., pp. III-321-324.

[60] G. Picchi, G. Prati, "Blind Equalization and Carrier Recovery Using a "Stop-and-Go" Decision-Directed Algorithm," IEEE Transactions on Communications, Vol. COM-35, No. 9, September 1987, pp. 877-887.

[61] Pollard, Al, Manitoba Telephone Systems, personal communication, May, 1994.

[62] J.G. Proakis, M. Salehi, "Communication Systems Engineering," Prentice Hall, NJ, 1994.

[63] W.D. Reeve, "Subscriber Loop Signaling and Transmission Handbook: Analog," The Institute of Electrical and Electronics Engineers, Inc., New York, IEEE Press, 1992.

[64] C. Roberge, J-P. Adoul, "Fast On-Line Speech/Voiceband-Data Discrimination for Statistical Multiplexing of Data with Telephone Conversations," IEEE Trans. on Commun., Vol. COM-34, No. 8, August 1986, pp. 744-751.

[65] M.S. Roden, "Analog and Digital Communications Systems," 3rd Edition, Prentice Hall, New Jersey, 1991.

[66] F.J. Ross, D.P. Taylor, "An Enhancement to Blind Equalization Algorithms," IEEE Transactions on Communications, Vol. 39, No. 5, May 1991, pp. 636-639.

[67] J.S. Sewall, B.F. Cockburn, "Signal Classification in Digital Telephone Networks," Proceedings of the IEEE Canadian Conference on Electrical and Computer Engineering, September 5-8, 1995, Vol. 2, pp. 957-961.

[68] K. Shimokoshi, Y. Hashitsume, "A Study of Voice/Non-Voice Discrimination Method Using Neural Networks for Integrated Packet Switching System," Proceedings IEEE International Symposium on Circuits and Systems, V. 3, 1989, pp. 2096-2099.

[69] R.H. Shumway, "Discriminant Analysis for Time Series," Handbook of Statistics, Vol. 2, North-Holland Publishing Co., 1982, pp. 1-46.

[70] S.S. Soliman, Shue-Zen Hsue, "Signal Classification Using Statistical Moments," IEEE Transactions on Communications, Vo. 40, No. 5, May 1992, pp. 908-916.

[71] SPSS Inc., "SPSS Statistical Algorithms," 2nd Edition, Chicago, Illinois.

[72] Tellabs International Inc., Digital Channel Occupancy Analyser information pamphlet, Lisle, IL, USA.

[73] J.K. Tugnait, "Blind Equalization and Estimation of Digital Communication FIR Channels Using Cumulant Matching," IEEE Transactions on Communications, Vol. 43, No. 2/3/4, February/March/April 1995, pp. 1240-1245.

[74] Ungerboeck, Gottfried, "Trellis-Coded Modulation with Redundant Signal Sets Part I: Introduction", IEEE Communications Magazine, Vol. 25, No. 2, February 1987.

[75] Wei, Lee-Fang, "Trellis-Coded Modulation with Multidimensional Constellations", IEEE Transactions on Information Theory, Vol. IT-33, No. 4, July 1987.

[76] K. Wesolowski, "Adaptive Blind Equalizers with Automatically Controlled Parameters," IEEE Transactions on Communications, Vol. 43, No. 2/3/4, February/March/April 1995, pp. 170-172.

[77] S.L. Wood, J.R. Treichler, "Performance of the Radon Transform Method for Constellation Identification," 22nd Asilomar Conference on Signals, Systems, and Computers, Oct 3-Nov 2, 1988, pp. 119-123.

[78] K. Yamazadi, R.A. Kennedy, "On Globally Convergent Blind Equalization for QAM Systems," Proceedings of the 1994 IEEE International Conference on

Acoustics, Speech and Signal Processing, Vol. 3, 1994, Piscataway, NJ., pp. III-325-328.

[79] Y. Yatsuzuka, "Highly Sensitive Speech Detector and High-Speed Voiceband Data Discriminator in DSI-ADPCM," IEEE Trans. on Commun., Vol. COM-30, No. 4, April 1982, pp. 739-750.

Index

Appendix A

Note that the “?” marks in the “Time” column indicate that the length of the call has not been recorded in the table.

TABLE 48. Description of recorded data files

File Name	FAX/ Data/ Voice	Standard	bps	Org.	Dest.	Time (s)	Description
v32bis	d	V.32bis	14,400	local	492-3214	20	SupraFAXmodem; called UofA modem pool; negotiation
v32bis.2	d	V.32bis	14,400	“	“	20	SupraFAXmodem; called UofA modem pool; no neg.
v32	d	V.32	9,600	“	“	20	including negotiation
v32.2	d	V.32	9,600	“	“	20	no negotiation
data1	d	V.22bis	2,400	“	“	25	including negotiation
data2	d	V.22bis	2,400	“	“	25	no negotiation
data3	d	V.32bis	12,000	“	“	25	including negotiation
data4	d	“	9,600	“	“	25	including negotiation
data5	d	“	12,000	“	“	25	no negotiation
data6	d	“	9,600	“	“	25	including negotiation
data7	d	“	9,600	“	“	25	no negotiation
data10	d	“	14,400	“	492-3214	45	Called UofA; incl. negotiation; modem option N8 forces bps
data11	d	“	12,000	“	“	60	“; “; N7
data12	d	“	9,600	“	“	60	“; “; N6
data13	d	“	9,600	“	492-0096	60	“; “; N6
data14	d	“	4,800	“	“	60	“; “; N4
data15	d	V.32bis	2,400	local	492-0096	60	“; “; N3
data16	d	“	2,400	“	492-0024	60	“; “; N3; retrain?
data17	d	“	1,200	“	492-0096	60	“; “; N2
data18	d	“	1,200	“	492-0024	60	“; “; N2
data19	d	“	300	“	492-0096	60	“; “; N1
data20	d	V.34	24,000/ 26,400	“	444-7685	90	called WorldGate; incl. negotiation; speed not forced (N0)
data21	d	“	28,800/ 28,800	“	“	“	“; “; “
data22	d	“	26,400/ 26,400	“	“	“	“; “; N13
data23	d	“	24,000/ 24,000	“	“	“	“; “; N12
data24	d	“	21,600/ 21,600	“	“	“	“; “; N11
data25	d	“	19,200/ 19,200	“	“	“	“; “; N10
data26	d	“	16,800/ 16,800	“	“	“	“; “; N9
drun1	d	V.34	28,800/ 28,800	TRLabs	U of A 492-3214	30	no setup; &N0
drun2	d	“	“	“	“	“	“
drun3	d	“	“	“	“	“	“

TABLE 48. Description of recorded data files

File Name	FAX/ Data/ Voice	Standard	bps	Org.	Dest.	Time (s)	Description
drun4	d	"	"	"	"	"	"
drun5	d	"	"	"	"	"	"
drun6	d	V.34	14,400/ 14,400	"	"	"	"; &N8
drun7	d	V.22bis	2,400	"	492-0024	"	"
drun8	d	"	"	"	"	"	"
drun9	d	"	"	"	"	"	"
drun10	d	"	"	"	"	"	"
drun11	d	"	"	"	"	"	"
drun12	d	V.32bis	14,400	"	TRLabs	"	"
drun13	d	"	"	"	"	"	"
drun14	d	"	"	"	"	"	"
drun15	d	"	"	"	"	"	"
drun16	d	"	"	"	"	"	"
drun17	d	"	12,000	"	"	"	"; &N7
drun18	d	"	"	"	"	"	"
drun19	d	"	9,600	"	"	"	"; &N6
drun20	d	"	"	"	"	"	"
drun21	d	"	"	"	"	"	"
drun22	d	"	"	"	"	"	"
drun23	d	"	"	"	"	"	"
drun24	d	"	7,200	"	"	"	"; &N5
drun25	d	"	"	"	"	"	"
drun26	d	"	4,800	"	"	"	"; &N4
drun27	d	"	"	"	"	"	"
drun28	d	"	"	"	"	"	"
-	-	-	-	-	-	-	-
fax1	f	V.17	14,400	local	441-3600	?	fax to main office; setup included; two pages plus cover
fax2	f	V.17	12,000	"	"	?	"
fax3	f	V.17	9,600	"	"	?	"
fax4	f	V.17	7,200	"	"	?	"
fax5	f	V.27ter	4,800	"	"	?	"
fax6	f	V.27ter	2,400	"	"	?	"; "; no cover page; error after p. 1; part of p. 2
fax9	f	V.29	7,200	local	492-1811	?	fax out; setup included; two pages plus cover
fax10	f	V.27ter	4,800	local	"	?	"
fax11	f	V.27ter	2,400	local	"	?	" ; errored
fax12	f	V.29	9,600	local	"	?	"
fax13	f	V.29	9,600	local	"	?	" ; errored
fax14	f	V.29	9,600	local	"	?	"
fax15	f	V.29	9,600	local	"	?	"
fax16	f	V.29	9,600	local	"	?	"
fax17	f	V.29	7,200	local	"	?	"

TABLE 48. Description of recorded data files

File Name	FAX/ Data/ Voice	Standard	bps	Org.	Dest.	Time (s)	Description
fax21	f	V.29	7,200	local	604-721-0852	?	"
fax22	f	V.27ter	4,800	local	"	?	"
fax23	f	?	2,400	local	"	?	errored fax
fax24	f	?	?	441-3600	local	?	fax received; errored
fax26	f	?	?	"	local	?	fax received; errored
-	-	-	-	-	-	-	-
voice1	v	-	-	498-8397	local	70	Male/Female conversation
voice_1_6_14_17_8	v	-	-	remote	local	?	Pre-recorded message (my voice); two sentences read by others
voice_1_6_14_40_23	v	-	-	"	"	?	"
voice_1_6_14_44_36	v	-	-	"	"	?	Pre-recorded message; person mimicing modem
voice_1_6_14_45_13	v	-	-	"	"	?	"
voice_1_6_14_46_2	v	-	-	"	"	?	Pre-recorded message; person whistling
voice_1_6_14_47_48	v	-	-	"	"	?	Pre-recorded message; person saying nothing
voice_1_6_14_49_27	v	-	-	"	"	?	"
voice_1_6_15_21_15	v	-	-	"	"	?	"
voice_1_6_15_2_18	v	-	-	"	"	?	"
voice_1_6_15_38_52	v	-	-	"	"	?	"
voice_i_6_15_44_14	v	-	-	"	"	?	"
voice_1_6_15_4_27	v	-	-	"	"	?	"
voice_1_6_15_54_39	v	-	-	"	"	?	"
voice_1_6_15_5_35	v	-	-	"	"	?	"
voice_1_6_15_5_4	v	-	-	"	"	?	"
voice_1_6_17_11_41	v	-	-	"	"	?	"
voice_1_6_17_54_22	v	-	-	"	"	?	"
voice_1_6_18_55_44	v	-	-	"	"	?	"
voice_1_7_10_33_11	v	-	-	"	"	?	"
voice_1_7_13_17_[illegible]	v	-	-	"	"	?	"

TABLE 48. Description of recorded data files

File Name	FAX/ Data/ Voice	Standard	bps	Org.	Dest.	Time (s)	Description
voice_1_7_ 14_49_38	v	-	-	"	"	?	"
voice_1_7_ 14_57_46	v	-	-	"	"	?	"
voice_1_7_ 18_6_59	v	-	-	"	"	?	"
voice_1_7_ 20_32_3	v	-	-	"	"	?	"
voice_1_7_ 7_21_34	v	-	-	"	"	?	"
voice_1_7_ 7_57_37	v	-	-	"	"	?	"
voice_1_8_ 14_51_6	v	-	-	"	"	?	"
voice_1_8_ 16_58_8	v	-	-	"	"	?	"
voice_1_8_ 18_25_20	v	-	-	"	"	?	"
voice_1_8_ 8_17_1	v	-	-	"	"	?	"
voice_1_9_ 14_26_10	v	-	-	"	"	?	"
-	-	-	-	-	-	-	-
sim1	d	V.22	1200	-	-	10	simulated call; recorded at alpha point in 4-wire connection
sim2	d	V.22	1200	-	-	30	" ; beta
sim3	d	V.22bis	2400	-	-	10	" ; alpha
sim4	d	V.22bis	2400	-	-	30	" ; beta
sim5	f	V.27ter	4800	-	-	10	" ; alpha
sim6	f	V.27ter	2400	-	-	10	" ; alpha ; fallback mode
sim7	f	V.29	9600	-	-	10	" ; alpha
sim8	f	V.29	7200	-	-	10	" ; alpha ; fallback mode
sim9	d	V.32	9600	-	-	10	" ; alpha
sim10	d	V.32bis	14,400	-	-	10	" ; alpha
sim11	f	V.17	14,400	-	-	10	" ; alpha ; identical simulation to V.32bis ; correct?
sim12	d	V.22	1200	-	-	30	" ; beta
sim13	d	V.22bis	2400	-	-	30	" ; beta
sim_1_a_1	d	V.22	1200	-	-	25	impairment model 1, alpha monitoring point
sim_1_a_2	d	V.22bis	2400	-	-	25	impairment model 1, alpha monitoring point
sim_1_a_3	f	V.27ter	4800	-	-	25	impairment model 1, alpha monitoring point
sim_1_a_4	f	V.27ter	2400	-	-	25	impairment model 1, alpha monitoring point
sim_1_a_5	f	V.29	9,600	-	-	25	impairment model 1, alpha monitoring point
sim_1_a_6	f	V.29	7,200	-	-	25	impairment model 1, alpha monitoring point
sim_1_a_7	d	V.32	9,600	-	-	25	impairment model 1, alpha monitoring point
sim_1_a_8	d	V.32bis	14,400	-	-	25	impairment model 1, alpha monitoring point
sim_1_a_9	f	V.17	14,400	-	-	25	impairment model 1, alpha monitoring point

TABLE 48. Description of recorded data files

File Name	FAX/ Data/ Voice	Standard	bps	Org.	Dest.	Time (s)	Description
sim_1_b_10	d	V.22	1200	-	-	25	impairment model 1, beta monitoring point
sim_1_b_11	d	V.22bis	2400	-	-	25	impairment model 1, beta monitoring point
sim_2_a_1	d	V.22	1200	-	-	25	impairment model 2, alpha monitoring point
sim_2_a_2	d	V.22bis	2400	-	-	25	impairment model 2, alpha monitoring point
sim_2_a_3	f	V.27ter	4800	-	-	25	impairment model 2, alpha monitoring point
sim_2_a_4	f	V.27ter	2400	-	-	25	impairment model 2, alpha monitoring point
sim_2_a_5	f	V.29	9,600	-	-	25	impairment model 2, alpha monitoring point
sim_2_a_6	f	V.29	7,200	-	-	25	impairment model 2, alpha monitoring point
sim_2_a_7	d	V.32	9,600	-	-	25	impairment model 2, alpha monitoring point
sim_2_a_8	d	V.32bis	14,400	-	-	25	impairment model 2, alpha monitoring point
sim_2_a_9	f	V.17	14,400	-	-	25	impairment model 2, alpha monitoring point
sim_2_b_10	d	V.22	1200	-	-	25	impairment model 2, beta monitoring point
sim_2_b_11	d	V.22bis	2400	-	-	25	impairment model 2, beta monitoring point
sim_3_a_1	d	V.22	1200	-	-	25	impairment model 3, alpha monitoring point
sim_3_a_2	d	V.22bis	2400	-	-	25	impairment model 3, alpha monitoring point
sim_3_a_3	f	V.27ter	4800	-	-	25	impairment model 3, alpha monitoring point
sim_3_a_4	f	V.27ter	2400	-	-	25	impairment model 3, alpha monitoring point
sim_3_a_5	f	V.29	9,600	-	-	25	impairment model 3, alpha monitoring point
sim_3_a_6	f	V.29	7,200	-	-	25	impairment model 3, alpha monitoring point
sim_3_a_7	d	V.32	9,600	-	-	25	impairment model 3, alpha monitoring point
sim_3_a_8	d	V.32bis	14,400	-	-	25	impairment model 3, alpha monitoring point
sim_3_a_9	f	V.17	14,400	-	-	25	impairment model 3, alpha monitoring point
sim_3_b_10	d	V.22	1200	-	-	25	impairment model 3, beta monitoring point
sim_3_b_11	d	V.22bis	2400	-	-	25	impairment model 3, beta monitoring point
sim_4_a_1	d	V.22	1200	-	-	25	impairment model 4, alpha monitoring point
sim_4_a_2	d	V.22bis	2400	-	-	25	impairment model 4, alpha monitoring point
sim_4_a_3	f	V.27ter	4800	-	-	25	impairment model 4, alpha monitoring point
sim_4_a_4	f	V.27ter	2400	-	-	25	impairment model 4, alpha monitoring point
sim_4_a_5	f	V.29	9,600	-	-	25	impairment model 4, alpha monitoring point
sim_4_a_6	f	V.29	7,200	-	-	25	impairment model 4, alpha monitoring point
sim_4_a_7	d	V.32	9,600	-	-	25	impairment model 4, alpha monitoring point
sim_4_a_8	d	V.32bis	14,400	-	-	25	impairment model 4, alpha monitoring point
sim_4_a_9	f	V.17	14,400	-	-	25	impairment model 4, alpha monitoring point
sim_4_b_10	d	V.22	1200	-	-	25	impairment model 4, beta monitoring point
sim_4_b_11	d	V.22bis	2400	-	-	25	impairment model 4, beta monitoring point
sim_5_a_1	d	V.22	1200	-	-	25	impairment model 5, alpha monitoring point
sim_5_a_2	d	V.22bis	2400	-	-	25	impairment model 5, alpha monitoring point
sim_5_a_3	f	V.27ter	4800	-	-	25	impairment model 5, alpha monitoring point
sim_5_a_4	f	V.27ter	2400	-	-	25	impairment model 5, alpha monitoring point
sim_5_a_5	f	V.29	9,600	-	-	25	impairment model 5, alpha monitoring point
sim_5_a_6	f	V.29	7,200	-	-	25	impairment model 5, alpha monitoring point
sim_5_a_7	d	V.32	9,600	-	-	25	impairment model 5, alpha monitoring point
sim_5_a_8	d	V.32bis	14,400	-	-	25	impairment model 5, alpha monitoring point

TABLE 48. Description of recorded data files

File Name	FAX/ Data/ Voice	Standard	bps	Org.	Dest.	Time (s)	Description
sim_5_a_9	f	V.17	14,400	-	-	25	impairment model 5, alpha monitoring point
sim_5_b_10	d	V.22	1200	-	-	25	impairment model 5, beta monitoring point
sim_5_b_11	d	V.22bis	2400	-	-	25	impairment model 5, beta monitoring point
-	-	-	-	-	-	-	-
dlib1	-	-	-	-	-	6	information signal from file v32bis; ie. no neg, no retrain
dlib2	-	-	-	-	-	20	information signal from file v32bis.2
dlib3	-	-	-	-	-	8	information signal from file v32
dlib4	-	-	-	-	-	20	information signal from file v32.2
dlib5	-	-	-	-	-	13	information signal from file data1
dlib6	-	-	-	-	-	25	information signal from file data2
dlib7	-	-	-	-	-	9	information signal from file data3
dlib8	-	-	-	-	-	9	information signal from file data4
dlib9	-	-	-	-	-	25	information signal from file data5
dlib10	-	-	-	-	-	15	information signal from file data6
dlib11	-	-	-	-	-	25	information signal from file data7
dlib14	-	-	-	-	-	24	information signal from file data10
dlib15	-	-	-	-	-	41	information signal from file data11
dlib16	-	-	-	-	-	44	information signal from file data12
dlib17	-	-	-	-	-	42	information signal from file data13
dlib18	-	-	-	-	-	38	information signal from file data14
dlib19	-	-	-	-	-	46	information signal from file data15
dlib20	-	-	-	-	-	51	information signal from file data16
dlib21	-	-	-	-	-	41	information signal from file data17
dlib22	-	-	-	-	-	46	information signal from file data18
dlib23	-	-	-	-	-	45	information signal from file data19
dlib24	-	-	-	-	-	62	information signal from file data20
dlib25	-	-	-	-	-	50	information signal from file data21
dlib26	-	-	-	-	-	62	information signal from file data22
dlib27	-	-	-	-	-	62	information signal from file data23
dlib28	-	-	-	-	-	62	information signal from file data24
dlib29	-	-	-	-	-	62	information signal from file data25
dlib30	-	-	-	-	-	62	information signal from file data26
dlib31	-	-	-	-	-	25	information signal from file fax1
dlib32	-	-	-	-	-	25	information signal from file fax2
dlib33	-	-	-	-	-	38	information signal from file fax3
dlib34	-	-	-	-	-	50	information signal from file fax4
dlib35	-	-	-	-	-	75	information signal from file fax5
dlib36	-	-	-	-	-	75	information signal from file fax6
dlib37	-	-	-	-	-	38	information signal from file fax9
dlib38	-	-	-	-	-	50	information signal from file fax10
dlib39	-	-	-	-	-	25	information signal from file fax11
dlib40	-	-	-	-	-	30	information signal from file fax12

TABLE 48. Description of recorded data files

File Name	FAX/ Data/ Voice	Standard	bps	Org.	Dest.	Time (s)	Description
dlib41	-	-	-	-	-	7	information signal from file fax13
dlib42	-	-	-	-	-	30	information signal from file fax14
dlib43	-	-	-	-	-	30	information signal from file fax15
dlib44	-	-	-	-	-	30	information signal from file fax16
dlib45	-	-	-	-	-	38	information signal from file fax17
dlib46	-	-	-	-	-	36	information signal from file fax21
dlib47	-	-	-	-	-	56	information signal from file fax22
dlib48	-	-	-	-	-	27	information signal from file fax23
dlib49	-	-	-	-	-	14	information signal from file fax24
dlib50	-	-	-	-	-	15	information signal from file fax26
rand1	rand.	-	-	-	-	125	random PCM sample stream
rand2	rand.	-	-	-	-	125	random PCM sample stream

Appendix B

Function code for distorting simulated network signals.

```
function ylin =
warp(a,b,t,samp,dup,be,snr,delay,fd,ad,edd,sim,mon,cmess)

% ylin = WARP(a,b,t,samp,dup,be,snr,delay,fd,ad,edd,sim,mon,cmess)
%
% Function WARP requires a pair of modulated signals to transmit
% over the phone line (a,b), the time vector (t) corresponding to (a,b),
% and the sampling rate (samp) of the system.  This function will
% implement mu-law encoding, additive noise, transhybrid losses, echos,
% frequency offsets, attenuation distortion, and envelope delay
% distortion.

% The companded signal 'y' is returned, and corresponds to the warped
% (distorted) signal from the specified monitoring point.
%
% The monitoring point could either be at the caller 2-W, answer 2-W,
% or the alpha/beta monitoring points.


 % Distort the signals before sending around the echos

 % frequency offset is implemented right at the modulation

 % then add attenuation distortion
 [bb,aa] = adgen(ad,samp);
 a = filter(bb,aa,a);
 if (dup ==2)
    b = filter(bb,aa,b);
 end


 % finally, incorporate the envelope delay distortion
 [bb,aa] = eddgen(edd,samp);
 a = filter(bb,aa,a);
 if (dup ==2)
    b = filter(bb,aa,b);
 end


  % Normalize the input waveforms to the average power
  % level that would normally be seen on the output of
  % a modem.

  dBmmodem = -9;
  Psignal = sum(a.^2)/length(a);
  Pmodem = 1e-3 * 10^(dBmmodem/10);
  k = sqrt(Pmodem/Psignal);
  a = a .* k;
  if (dup == 2)
  b = b .* k;
end;


% Now we can send the signals around the hybrids and
% incorporate the echos.  Recorded signals don't go
% through this.  They are simply attenuated.

if (sim==1)
% Generate all echoed signals for the a path
t1 = floor(samp * delay/(2*1000));
l = length(a);
k0 = 5;
k1 = 3.5;
loss1 = 10^(-(k0+k1)/20);
loss2 = 10^(-(k0+3*k1+be)/20);
loss3 = 10^(-(k0+5*k1+2*be)/20);
a1 = loss1 .* a';
a2 = loss2 .* a';
a3 = loss3 .* a';
a1 = [a1(l-t1+1:l) a1(1:l-t1)];
a2 = [a2(l-3*t1+1:l) a2(1:l-3*t1)];
a3 = [a3(l-5*t1+1:l) a3(1:l-5*t1)];
a4 = loss1 .* a1;
a5 = loss1 .* a3
a6 = loss1 .* a2;
a4 = [a4(l-t1+1:l) a4(1:l-t1)];
a5 = [a5(l-t1+1:l) a5(1:l-t1)];
a6 = [a6(l-t1+1:l) a6(1:l-t1)];


 % Generate all echoed signals for the b path only if full duplex
 if (dup == 1)
    b1 = zeros(size(b'));
    b2 = b1;
    b3 = b1;
    b4 = b1;
    b5 = b1;
    b6 = b1;
 else
    b1 = loss1 .* b';
    b2 = loss2 .* b';
    b3 = loss3 .* b';
    b1 = [b1(l-t1+1:l) b1(1:l-t1)];
    b2 = [b2(l-3*t1+1:l) b2(1:l-3*t1)]
    b3 = [b3(l-5*t1+1:l) b3(1:l-5*t1)];
    b4 = loss1 .* b1;
    b5 = loss1 .* b3;
    b6 = loss1 .* b2;
    b4 = [b4(l-t1+1:l) b4(1:l-t1)];
    b5 = [b5(l-t1+1:l) b5(1:l-t1)];
    b6 = [b6(l-t1+1:l) b6(1:l-t1)];
 end
```

```
% analysis signal generation (depending upon spec'd point)
if (mon == 1)            % alpha
   ylin = a1 + a3 + b2;
elseif (mon == 2)        % beta
   ylin = b1 + b3 + a2;
elseif (mon == 3)        % caller
   ylin = a + a6 + b4 + b5;
elseif (mon == 4)        % answer
   ylin = b + a4 + a5 + b6;
else
   % error
   return
end

else    % (sim==0) and we have a recorded signal
   k = 10^(-17/20);          % 17 dB end-to-end loss
   ylin = a .* k;
end     % if (sim==1)

% add in white noise, according to C-Notch or C-Message weighting
% parameters.  Note that this additive noise is at a specified
% power level.  The signal to noise ratio then is dependant upon
% the power level of the signal.  We will assume that the signal
% power level follows the average which is about -27 dBm0, at the
% receiving end of the call.

n = randn(size(ylin));
% measure the noise level
if (cmess==1)
   % cmessage noise measurement
   nlevel = cmess_measure(n,samp);
else
   % cnotched noise measurement
   nlevel = cnotch_measure(n,samp);
end

% adjust the noise vector power level to the specs
% If we are at alpha or beta, we are making things worse
% by using the same noise power, but this is a
% pessimistic approx.
PtonedBm = -12;
end2endloss = 6.8;                  % end office to end office loss
Ptone = 1e-3 * 10^((PtonedBm-end2endloss)/10);
Pn = sum(n.^2)./length(n);          % measured noise power
Po = Ptone * 10^(-snr/10);          % desired noise power after weights
SNRo = 10 * log10(1e12*Po);         % desired noise power (dBrnC)
Pno = Pn * 10^((SNRo-nlevel)/10); % desired noise power  weighted (W)
a = sqrt(Pno/Pn);
n = n.*a;

% add them up, and ylin had better be at a suitable power level
ylin = ylin + n;
```

Appendix C

Function code for generating pulse shaping filters.

```
function [t,h] = shape(fs,samp,a,n)

% FUNCTION [t,h] = shape(fs,samp,a,n)
%
% This function creates the filter coefficients required for pulse
% shaping with an alpha valued roll-off square root raised cosine
% filter.  The filter coefficients are returned in the h vector,
% along with the corresponding time vector, t.  The baud (symbol) rate
% is given by fs, the sampling rate is from samp, alpha is from a,
% and the number of filter points is specified by n.

T = 1/fs;
t = [-(n/2)/samp:(1/samp)+(1e-10):(n/2)/samp];

h = 4 * a / (pi*sqrt(T));
h = h .* (cos((1+a)*pi*t/T) + T*sin((1-a)*pi*t/T) ./ (4*a*t));
h = h ./ (1 - (4*a*t/T).^2);
```

Appendix D

Function code for supervising the generation of a simulated QAM signal.

```
function [y,t] = qammod(symb,fs,fc,time,samp,a)

% Function qammod(symb,fsmod,fcmod,time,samp,a) is used to perform
% qam modulation given the symbol set, symbol rate, carrier freq,
% time to simulate, and sampling rate.  The modulated signal is
% returned with its time vector.  The a value is for the raised
% cosine alpha roll-off factor.


if symb == 'a'
   [y,t] = qam4(samp,time,fc,fs,a);
elseif symb == 'b'
   [y,t] = qam8(samp,time,fc,fs,a);
elseif symb == 'c'
   [y,t] = qam8b(samp,time,fc,fs,a);
elseif symb == 'd'
   [y,t] = qam16(samp,time,fc,fs,a);
elseif symb == 'e'
   [y,t] = qam16b(samp,time,fc,fs,a);
elseif symb == 'f'
   [y,t] = qam32(samp,time,fc,fs,a);
else
   [y,t] = qam128(samp,time,fc,fs,a);
end
```

Appendix E

Function code for demodulating a QAM signal.

```
function [y] = qamdemod(x,fc,t,pass,samp)

% [y] = QAMDEMOD(x,fc,t,pass,samp)
%
% Function QAMDEMOD is used to demodulate QAM signals.  The input signal
% (x) is demodulated from the passband to the baseband, and a complex
% valued baseband signal is returned.  The carrier freq. to demodulate
% at is specified by (fc) and the max baseband signal allowable is
% specified by (pass).  (pass) is used for a lowpass baseband filter.
% The time vector of (x) is given by (t).  The sampling rate is given by
% the (samp) variable.
%
% Modifications:  Use only FIR filters since they have linear
% phase, and the group delay is easily computed.  IIR's were
% used before, but the group-delays are difficult to determine,
% and the phase is non-linear.

demod = exp(-j*2*pi*(fc)*t)';

if ((fc == 1200) | (fc == 2400))
   b = fir1(30,[2*(fc-600)/samp 2*(fc+600)/samp]);
   x = filter(b,1,x);
   x = x .* demod;
   b = fir1(30,(2*600/samp));
   y=filter(b,1,x);
else
   x = x .* demod;
   b = fir1(30,(2*pass/samp));
   y=filter(b,1,x);
end
```

Appendix F

Function code for generating a 16-QAM simulated signal.

```
function [y,tt] = qam16(samp,time,fc,fs,a)

% Function qam16(samp,time,fc,fs) requires the sampling rate (samp),
% the total signal time (time), the carrier freq (fc), and the
% symbol rate (fs) to perform modulation.  The modulated signal is
% returned.
%
% The modulation method is 16-QAM, with sinc symbol filtering.
% The input symbols are randomly generated.  Also, the time vector is
% returned.  The (a) value specifies the square root raised cosine
% pulse shaping roll-off factor.  (a)=0 invokes ideal nyquist rate
% pulse shaping.

wc = 2*pi*fc;
tempsamp = 9600;
n = 48;
t = (0:(1/fs):time);

% generate random input symbols and do pulse shaping
r = ceil(16*rand(size(t)));
len = length(r);

% define the 16-QAM symbol set;
symb = [1+1i,1+3i,3+1i,3+3i ...
       ,1-1i,1-3i,3-1i,3-3i ...
       ,-1+1i,-1+3i,-3+1i,-3+3i ...
       ,-1-1i,-1-3i,-3-1i,-3-3i];
symb = symb/max(abs(symb));
for i = 1:length(r)
   r1(i)=real(symb(r(i)));
   r2(i)=ima[illegible](i)));
end

% do pulse shaping.
if (a ~= 0)
   [ttemp,h] = shape(fs,tempsamp,a,n);
   r1p = resample(r1,tempsamp,fs,h);
   r2p = resample(r2,tempsamp,fs,h);
   tt = (0:(length(r2p)-1))/tempsamp;
else
   r1p = resample(r1,tempsamp,fs);
   r2p = resample(r2,tempsamp,fs);
   tt = (0:(length(r2p)-1))/tempsamp;
end

% modulate input symbols using QAM
y1 = r1p' .* sin(wc*tt)';
y2 = r2p' .* cos(wc*tt)';
y3 = y1 + y2;
```

```
y = resample(y3,samp,tempsamp);
tt = (0:(length(y)-1))/samp;
```

Appendix G

Function code for measuring C-Notch weighted noise.

```
function dBrn = cnotch_measure(n,samp)

% This code computes the C-notch filter coefficients,
% and then calculates the C-Notched noise power level
% of a signal.
%
% For the C-Notched measurement, a sinusoidal tone
% must be added to the signal, and then filtered
% out by a notch filter.


% first develop the cnotch filter
z = [+i*6202 +i*6346 +i*6494];
z = [z conj(z)];
p = [-197+i*5640 -1310+i*6209 -249+i*[illegible]];
p = [p conj(p)];

% use a bilinear transform to change to digital filter,
% using prewarping to match 1000 Hz point
[zd,pd,kd] = bilinear(z',p',1,samp,1010);

% convert from zero-pole representation to transfer function
[bn,an] = zp2tf(zd,pd,kd);

% then develop the cmessage filter
z=[0 0 0 inf inf inf inf];
p=[-1502+i*1267 -2439+i*5336 -4690+i*15267 -4017+i*21575];
p = [p conj(p)];

% use an impulse invariant transform to change to digital filter
[bs,as] = zp2tf(z',p',6.25e16);
[bm,am] = impinvar(bs,as,samp);

% now add a sinusoid to the signal
A = 1;
t = [0:1/samp:(length(n)-1)/samp];
x = A .* sin(2.*pi.*1010.*t);
n = n + x;

% apply the cmessage filter
n = real(filter(bm,am,n));

% apply the cnotched filter
n = real(filter(bn,an,n));

% measure the resulting power level
dBrn = 10 * log10(1e12*sum(n.^2)/length(n));
```

Appendix H

Function code for measuring C-Message weighted noise.

```
function dbrn = cmess_measure(n,samp)

% This routine computes the C-message digital filter
% coefficients, and then measures the noise power of
% an input noise vector.


z=[0 0 0 inf inf inf inf];
p=[-1502+i*1267 -2439+i*5336 -4690+i*15267 -4017+i*21575];
p = [p conj(p)];

% use an impulse invariant transform to change to digital filter
[bs,as] = zp2tf(z',p',6.25e16);
[b,a] = impinvar(bs,as,samp);

y = real(filter(b,a,n));
dbrn = 10 * log10(1e12*sum(y.^2)/length(y));
```

Appendix I

Function code for generating attenuation distortion filters.

```
function [b,a] = adgen(model,samp);

% This function is used to generate an FIR filter
% corresponding to a specific model of Attenuation
% Distortion.  There are 3 models (1,2,3).

F = [0 204 254 304 404 504 604 704 804 904 1004 1104 ...
      1204 1304 1404 1504 1604 1704 1804 1904 2004 ...
      2104 2204 2304 2404 2504 2604 2704 2804 2904 3004 3104 3204 ...
      3304 3404 3504 3704 4000];
M2 = -1 .* [30 5.1 3.3 1.8 1.1 .7 .4 .3 .2 .1 0 -.1 -.1 ...
      -.2 -.2 -.2 -.1 0 0 .1 .1 .2 .3 .4 .6 .875 1.15 1.425 1.7 ...
      2.0 2.3 3.1 4.1 5.3 7.4 16.5 60 60];
M3 = -1 .* [30 10 6.4 3.7 2.0 1.3 .9 .6 .4 .2 0 .1 0 .1 ...
      .1 .2 .2 .3 .4 .6 .9 1.0 1.4 1.9 2.3 3.175 4.05 4.925 5.8 ...
      6.8 7.3 8.8 11.5 13.8 14.4 21.2 60 60];

if (model == 1)
   b = 1;
   a = 1;
   return;
elseif (model == 2)
   M = M2;
elseif (model == 3)
   M = M3;
else
   b = 1;
   a = 1;
   return;
end


m = 10.^(-M/20);
m = 1./m;

f = F./(samp/2);

N = 40;

b = firls(N,f,m);
a = 1;
```

Appendix J

Function code for generating envelope delay distortion filters.

```
function [b,a] = eddgen(model,samp)

% FUNCTION [b,a] = eddgen(model,samp)
%
% This function is used to generate an EDD (Envelope
% Delay Distortion) allpass filter.  There are 4
% possible models to choose from.


% first see if we already have an EDD file for this
% filter.  If so, just load it and we are done.

if (exist([‘edd_filter_’ int2str(model) ‘.mat’]) == 2)
   eval([‘load edd_filter_’ int2str(model)]);
   return;
else
   % do the work of building a filter

   f0 = 16;
   j = sqrt(-1);
   f = [0:f0:4000];
   m = model;

   F = [600 1700 3000];
   EDD = [ 1000 0 950;
         1170 0 1350;
         1850 0 1350;
         2300 0 1850;] ./ 1e6 ./ (2.*pi);

   for i=1:size(EDD,1)
      lhs = EDD(i,:);
      rhs = [ F(1)^2 F(1) 1;
             F(2)^2 F(2) 1;
             F(3)^2 F(3) 1;];
      abc(i,:) = lhs / rhs’;
   end

   % New method using Markus Lang’s software

   N = 10;                  % allpass degree
   w = f.*2.*pi;
   om = w./samp;            % frequency points in rads
   W = 1;                   % no weighting function tau0 = 0;
   tau0 = 0;

   M = ones(1,length(f));
   H = M .* exp(-j .* (abc(m,1).*(w.^3)./(12*pi^2) + ...
      abc(m,2).*(w.^2)./(4*pi) + abc(m,3).*w));
   bw = -1 .* unwrap(angle(H));
   [p] = apdesz(bw, W, om, N, tau0, 0, `t');

   b = fliplr(p);
   a = p;

   eval([`save edd_filter_' int2str(model) ` b a']);

end     % if saved filter exists
```

Appendix K

FSK Demodulator Code

```
function [y,tt,d] = fskdemodTI(x,samp,fc,fs,df)

% FUNCTION [y,tt,d] = fskdemodTI(x,samp,fc,fs,df)
%
% This function is used to demodulate FSK signals,
% and try to determine the signal data.  The method
% used is very simple and elegant.  It was discovered
% in the TI dsp public archives.
%
% Inputs are x, samp, fc, fs, and df which represent
% the signal, sample rate, carrier frequency, symbol rate,
% and frequency offset, respectively.
%
% Outputs are y, tt, and d which represent the demodulated
% signal, the time vector, and the detected signal respectively.

wc = 2*pi*fc; N = length(x);

% determine time delay for demodulator
n = pi/(2*wc);

% i is the fractional delay part
i = n*samp - round(n*samp);
w = wc / samp;

% convert n (s) to n (samples)
n = round(n*samp);

% now solve for r=-b1
a = cos(w);
r1 = (-1*(a-2*i*a) + sqrt((a-2*i*a)^2 - 4*(i-1)*i)) / (2*(i-1));
r2 = (-1*(a-2*i*a) - sqrt((a-2*i*a)^2 - 4*(i-1)*i)) / (2*(i-1));
b1 = -1*min([r1 r2]);

% multiply received signal by time delayed version of itself
y = x(2:N-n) + b1*x(1:N-n-1);
y = x(2+n:N) .* y;

% setup a time vector
tt = (0:length(y)-1)/samp;

% lowpass filter the signal to remove the high freq stuff
N = 34;
Wn = (2*fs-df) / (samp/2);
b = fir1(N,Wn);
a = 1;
y = filter(b,a,y);
d = sign(y);
```

Sniffer Program Code

```
function [DISrate,DCSrate] =  sniffer(d,samp)

% FUNCTION [DISrate,DCSrate] =  sniffer(d,samp)
%
% This function is called the sniffer.  It will scan a
% signal, searching for special FSK modulated sequences that
% are used in FAX transmissions.
%
% Sniffer requires the signal stream, 'd', and the
% sampling rate of the stream, 'samp'.
%
% Sniffer outputs all of the DIS and DCS codes it detects as
% the matrices 'DISrate' and 'DCSrate'.
%
% Program written by Jeremy Sewall.


% For a V.21 sniffer, set the following:

fc = 1750;                               % carrier frequency
df = 100;                                % +/- change in frequency
fs = 300;                                % symbol rate
L = -1;                                  % logical low
H = 1;                                   % logical high
Hflag =     [L H H H H H H L];           % header flag
Haddress = [H H H H H L H H H];          % header address
Hcontrol = [H H L L L H L L L];          % header control
Header = [Hflag Haddress Hcontrol];      % complete header scanned for
Headerlength = length(Header);
DIS =       [L L L L L L L H];           % code for DIS identifier
DCS =       [H H L L L L L H];           % code for DCS identifier
type1 = [L L L L];                       % codes for different types of
type2 = [L H L L];                       % data signals
type3 = [H L L L];
type4 = [H H L L];
type5 = [L L H L];
type6 = [L H H L];
type7 = [H H H L];
type8 = [L L L H];
type9 = [L H L H];
type10 = [H L L H];
type11 = [H H L H];

[y,t,d] = fskdemodTI(d,samp,fc,fs,df);  % perform FSK demodulation

% detect signal; create 300 bps digital representation
count = 0;
z = 1;
prev = d(1);
for i=1:length(d)
   if (prev == d(i))
      count = count + 1;
   else
      numbauds = round(count*fs/samp);
      if (numbauds>0) data(z:z+numbauds-1) = prev*ones([1 numbauds]);
         end;
      count = 1;
      z = z + numbauds;
   end
prev = d(i);
end

% now search for DIS and DCS
z = 1;
Nheaders = 0;
NDIS = 0;
NDCS = 0;
zmax = 1;
i = 0;
while (i < length(data))
   i = i + 1;

   % First search for the header in the data
   if data(i) == Header(z)
      if z == Headerlength
         Nheaders = Nheaders + 1;

         % Found a header, so look for DIS or DCS
         if (data(i+1:i+length(DIS)) == DIS)
            NDIS = NDIS + 1;
            DISrate(NDIS,:) = data(i+19:i+22);
         elseif (data(i+1:i+length(DCS)) == DCS)
            NDCS = NDCS + 1;
            DCSrate(NDCS,:) = data(i+19:i+22);
         end
         z = 1;
      else
         z = z + 1;
      end
   else
      if (z~=1)
         z = 1;
         i = i-1;
      end
   end
 end
```

Appendix L

This appendix contains the probability mass functions (or histograms in this case) of eleven discriminant variables for each of nine classes. (Note that the area occupied by the histogram bars could be normalized to one to obtain the true probability mass functions.) Each plot is accompanied by four values in the plot titles, C, V, M, and S which correspond to the class number, variable number, discriminant variable mean value, and discriminant variable standard deviation, respectively. The discriminant variable numbers {1, 2,..., 11} correspond to variables {N2, Rd1, Rd2,..., Rd10}. All of the discriminant variables were applied to all available signal recordings and simulations. The parameters used for calculating the variables are L=16 samples, P_{th}=1029 μ-law decoder units, and N=1024 samples. If the N parameter (segment length) is increased, the standard deviation of each variable is decreased. The opposite is also true.

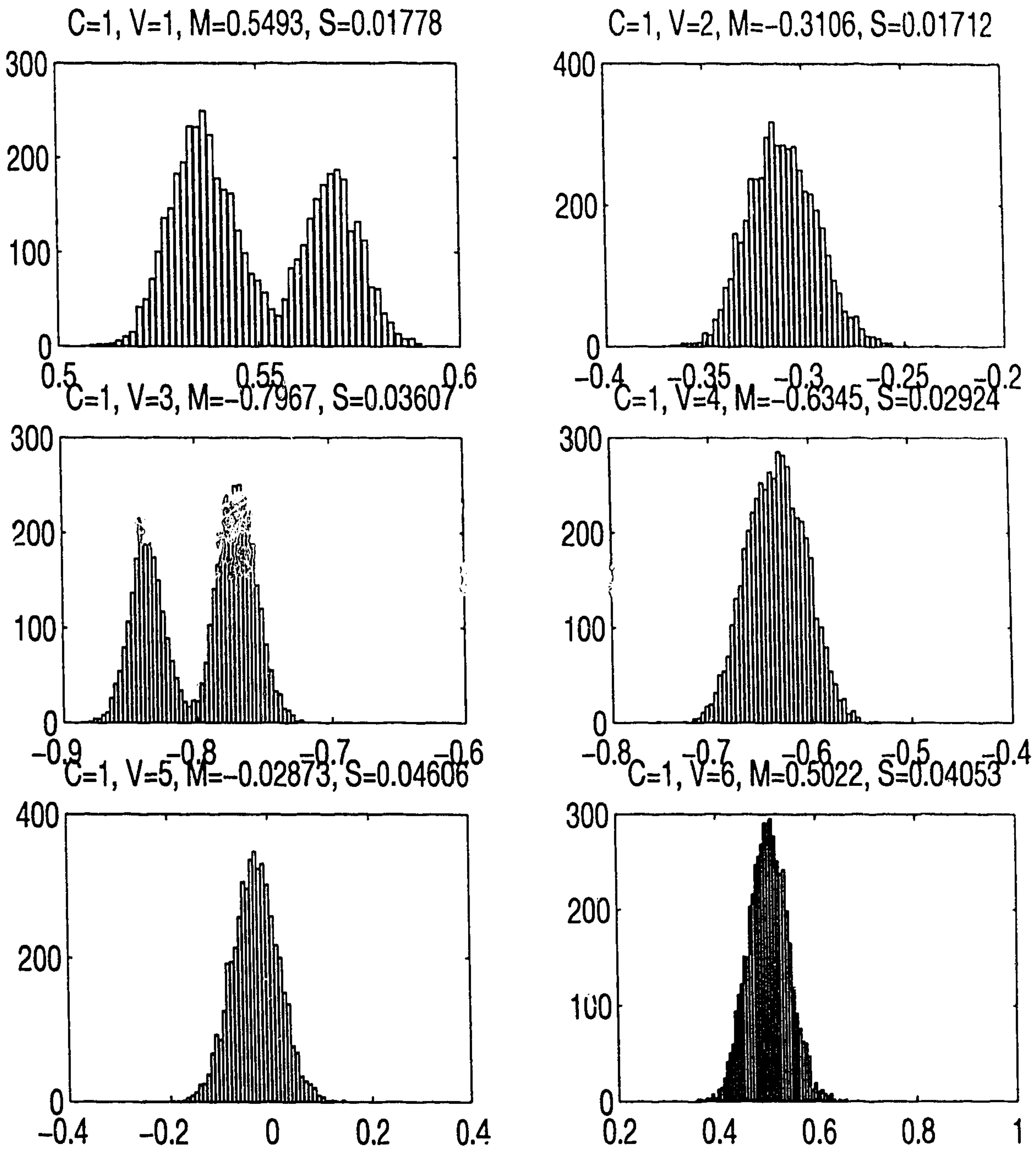


FIGURE 83. Histograms for Class 1.

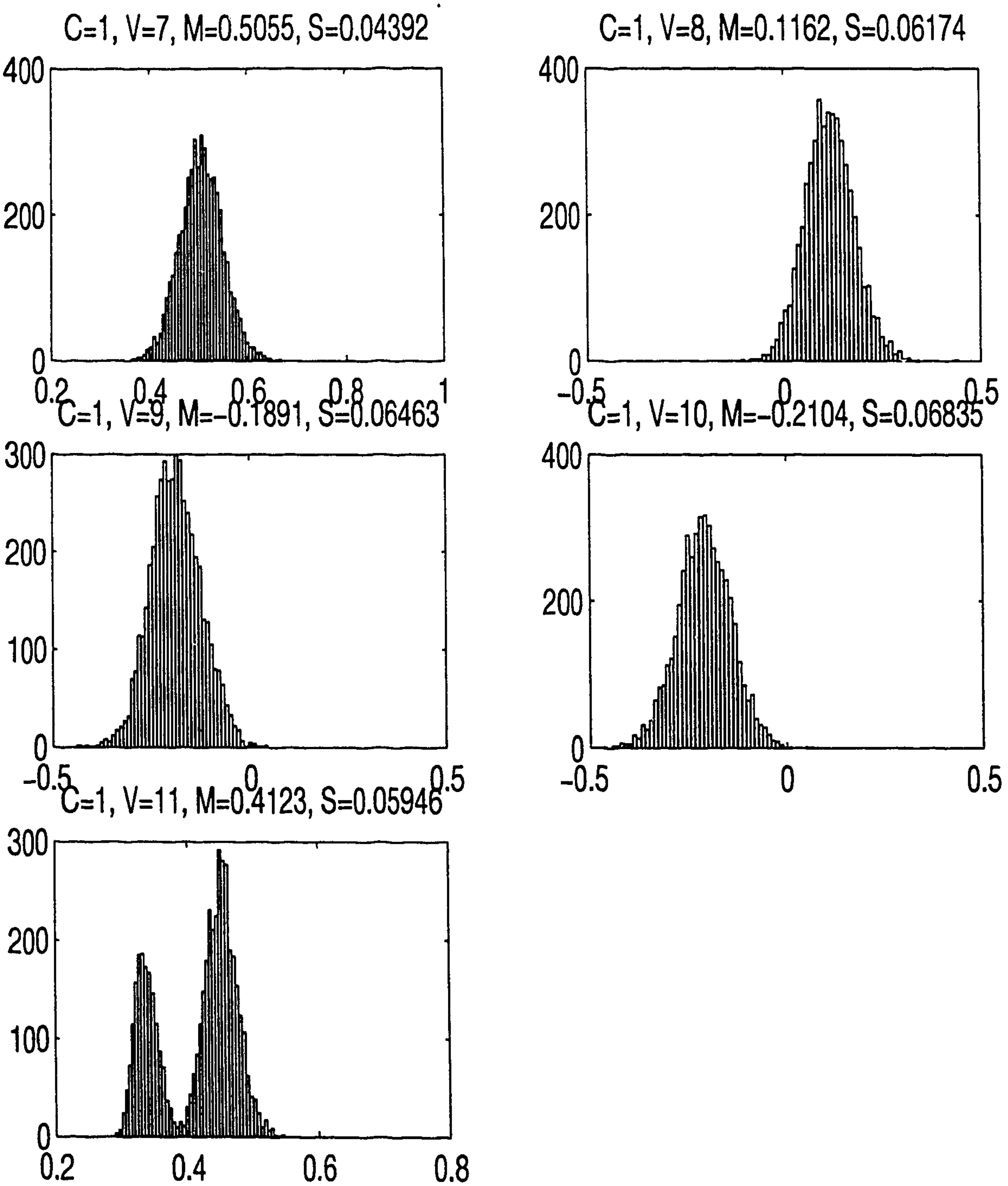


FIGURE 84. Histograms for Class 1.

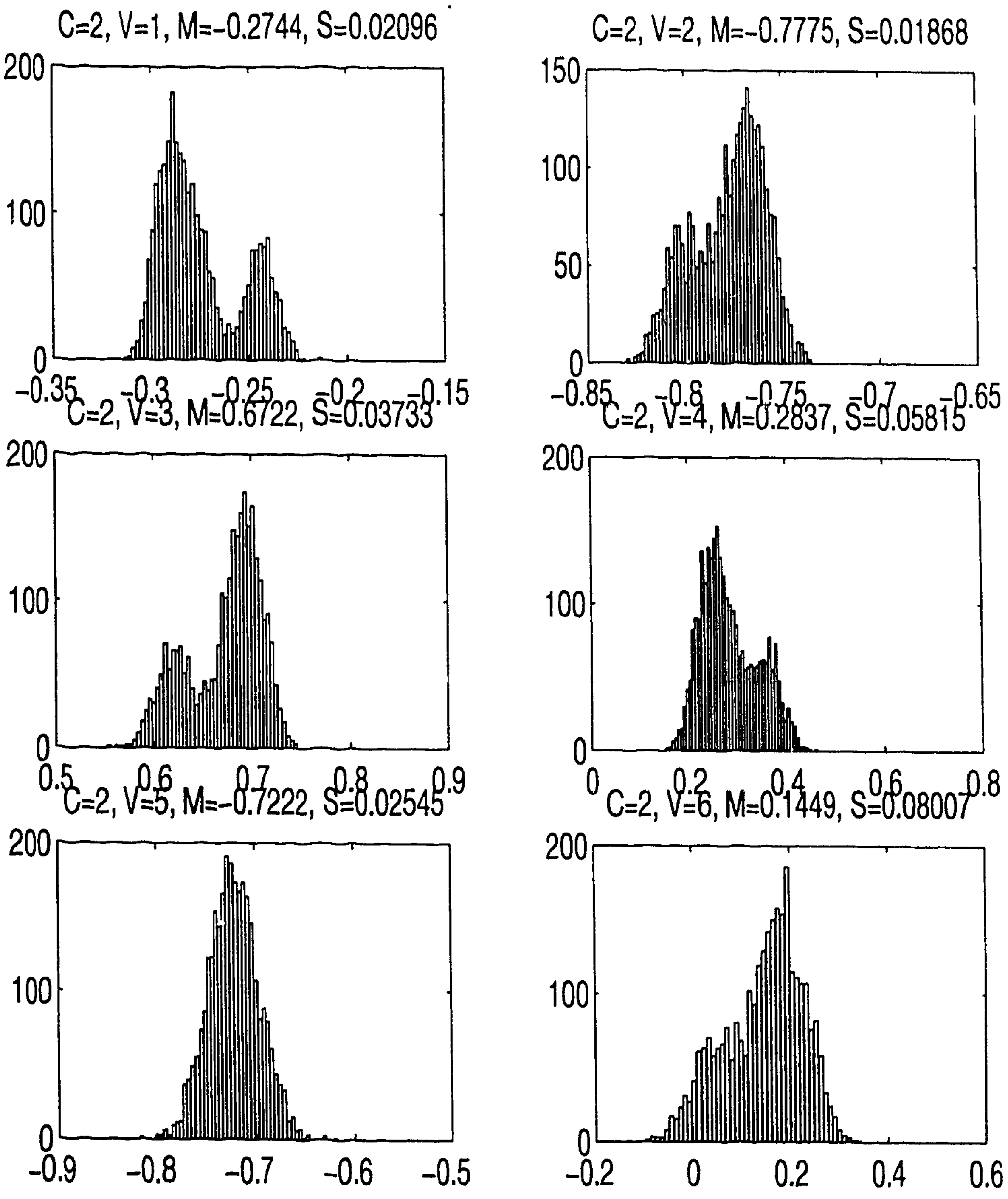


FIGURE 85. Histograms for Class 2.

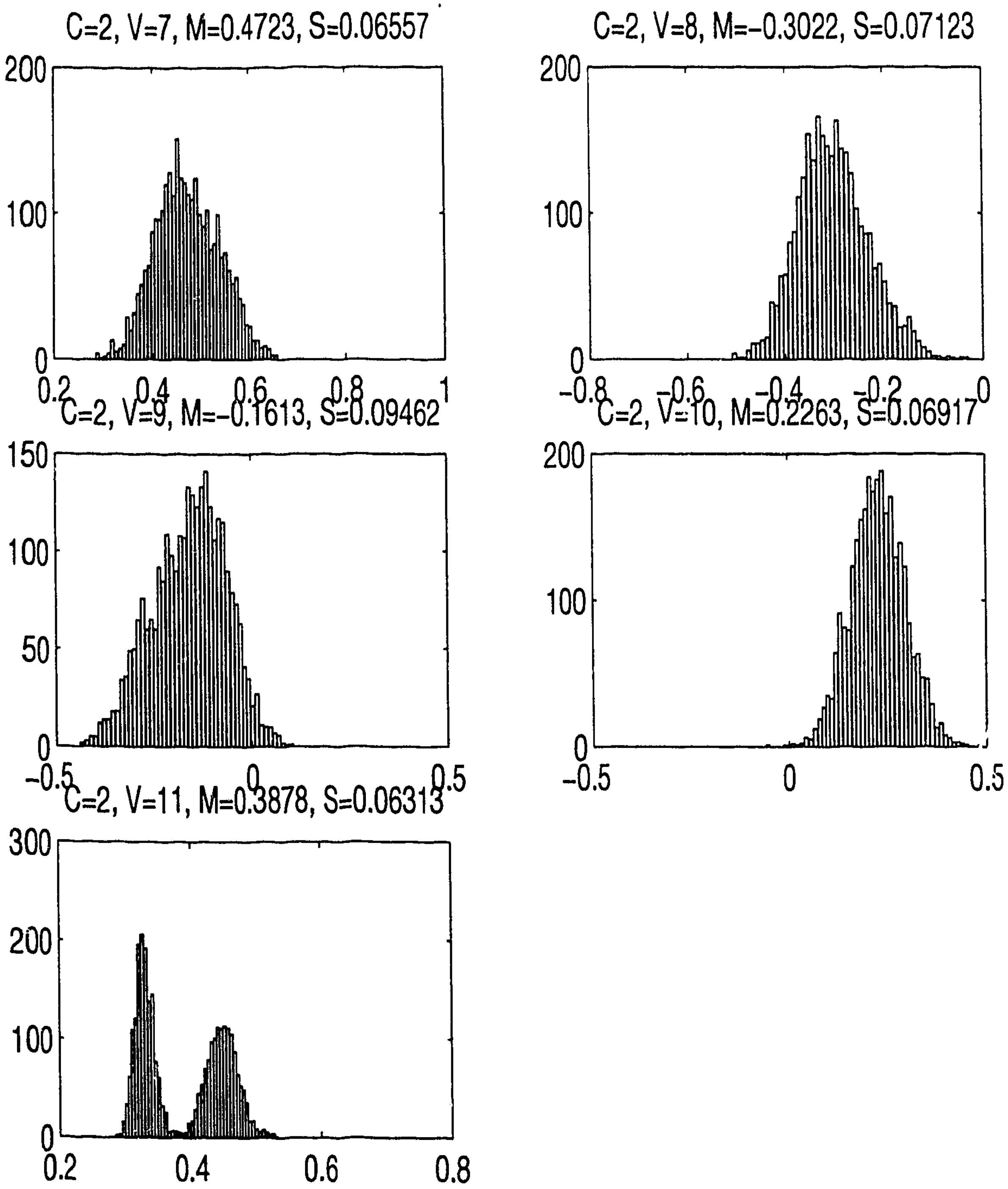


FIGURE 86. Histograms for Class 2.

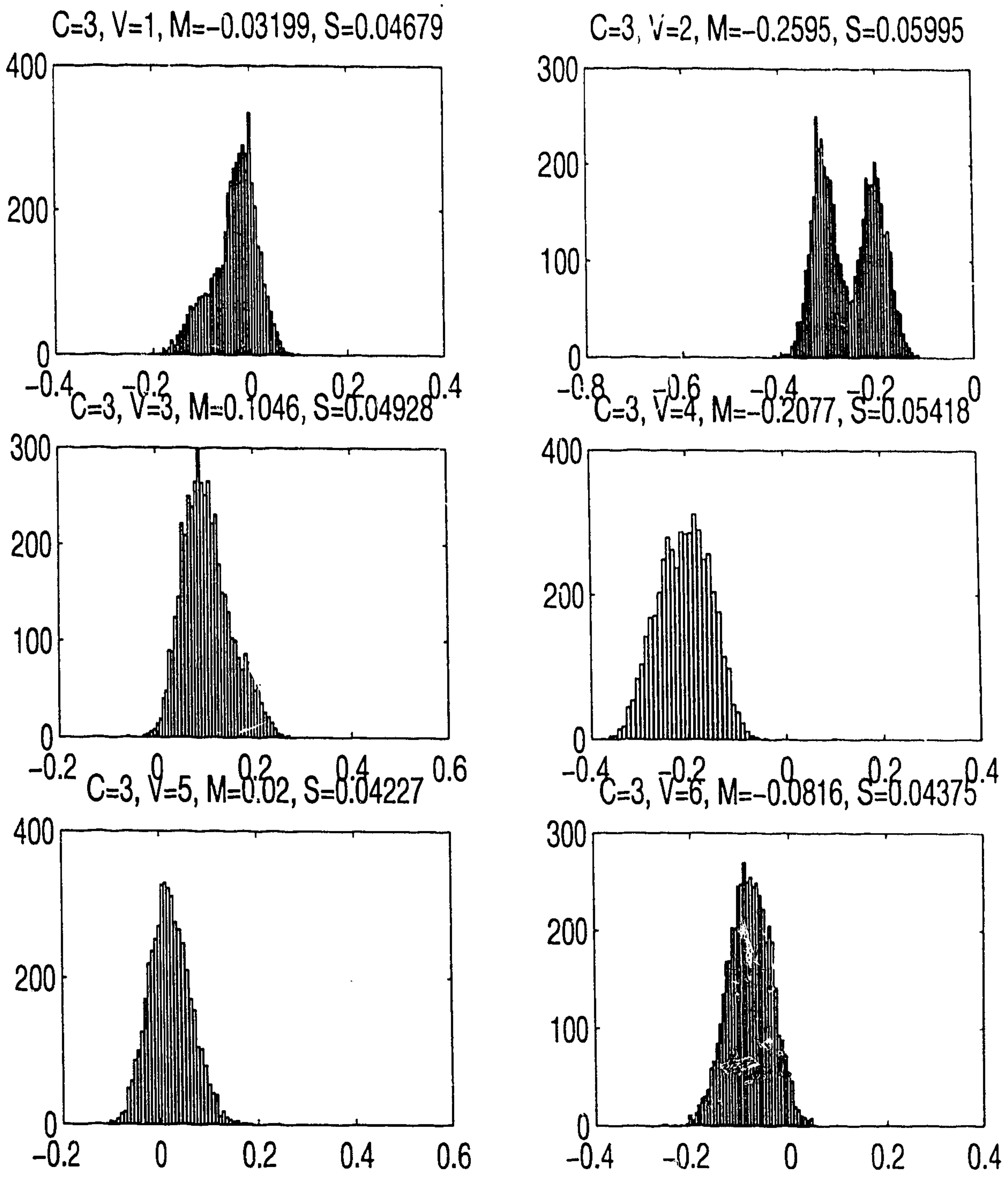


FIGURE 87. Histograms for Class 3.

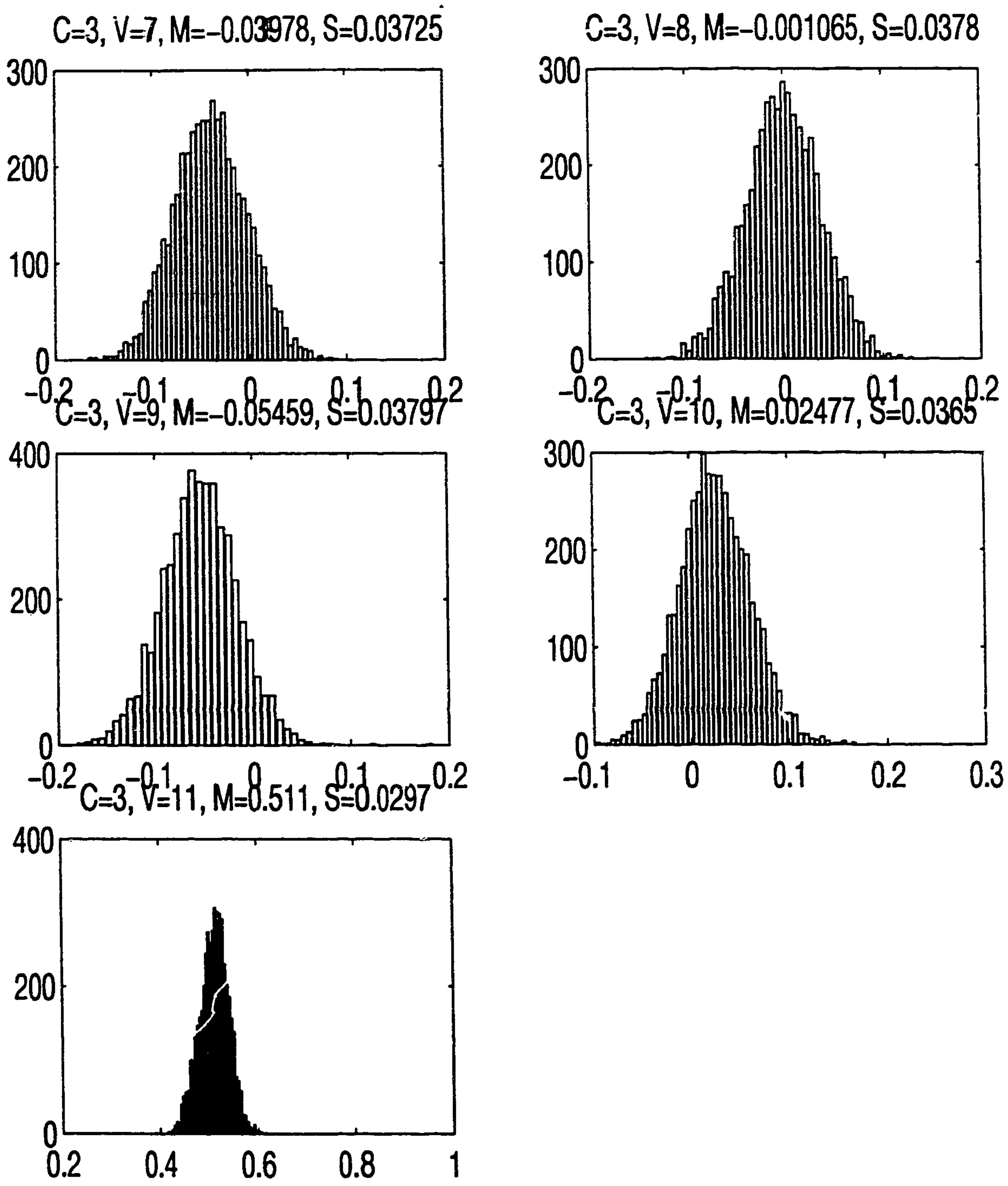


FIGURE 88. Histograms for Class 3.

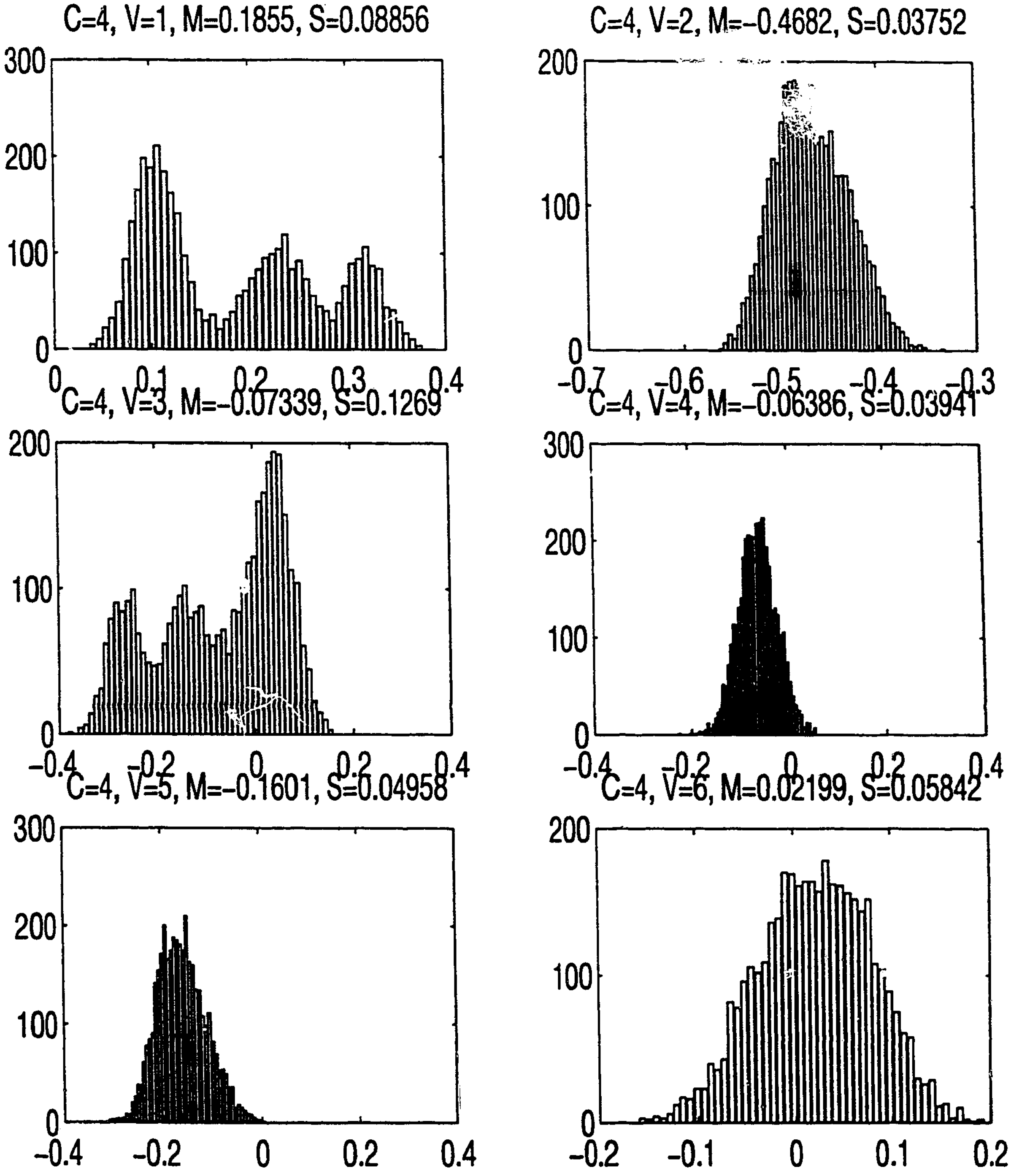


FIGURE 89. Histograms for Class 4.

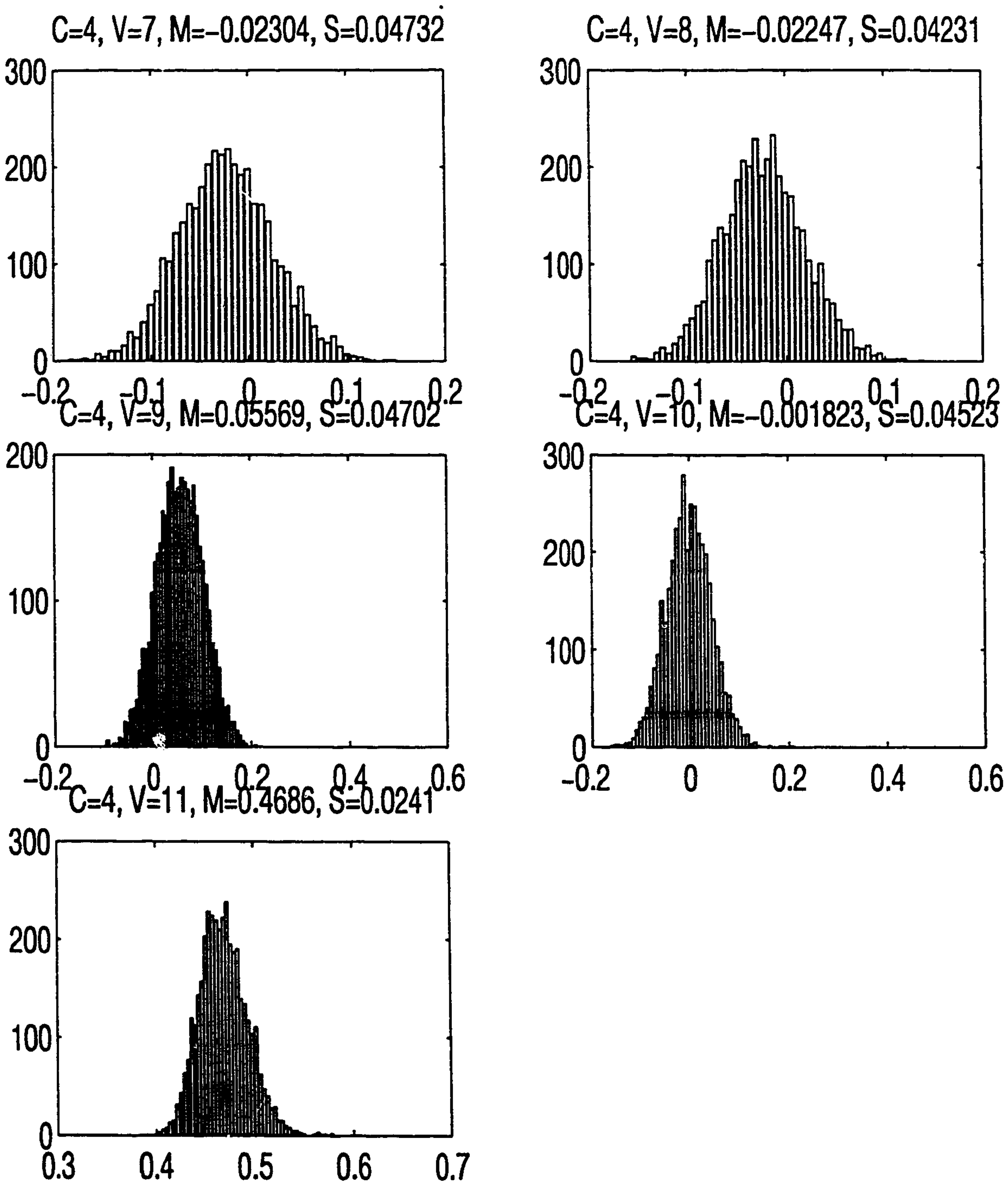


FIGURE 90. Histograms for Class 4.

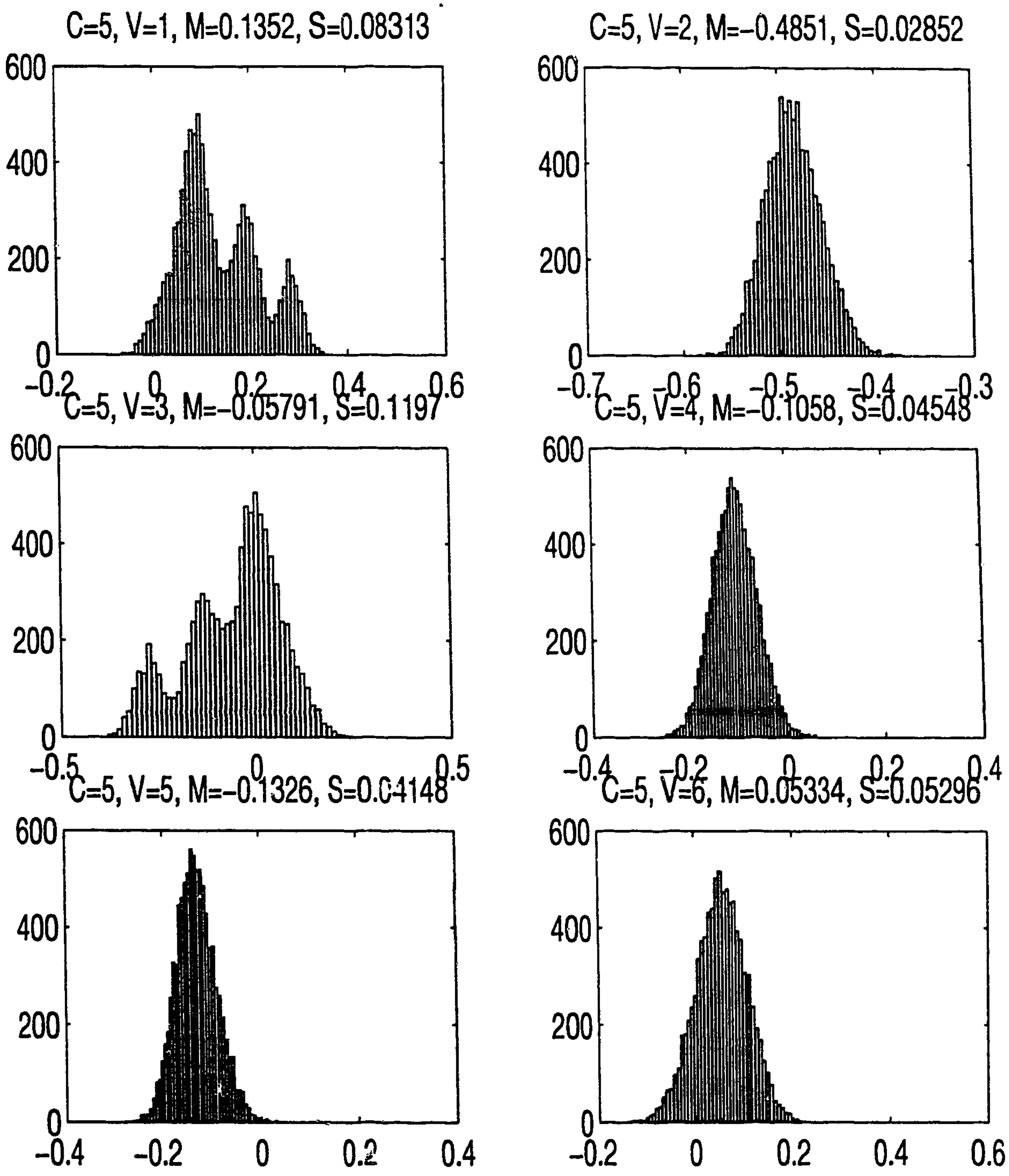


FIGURE 91. Histograms for Class 5.

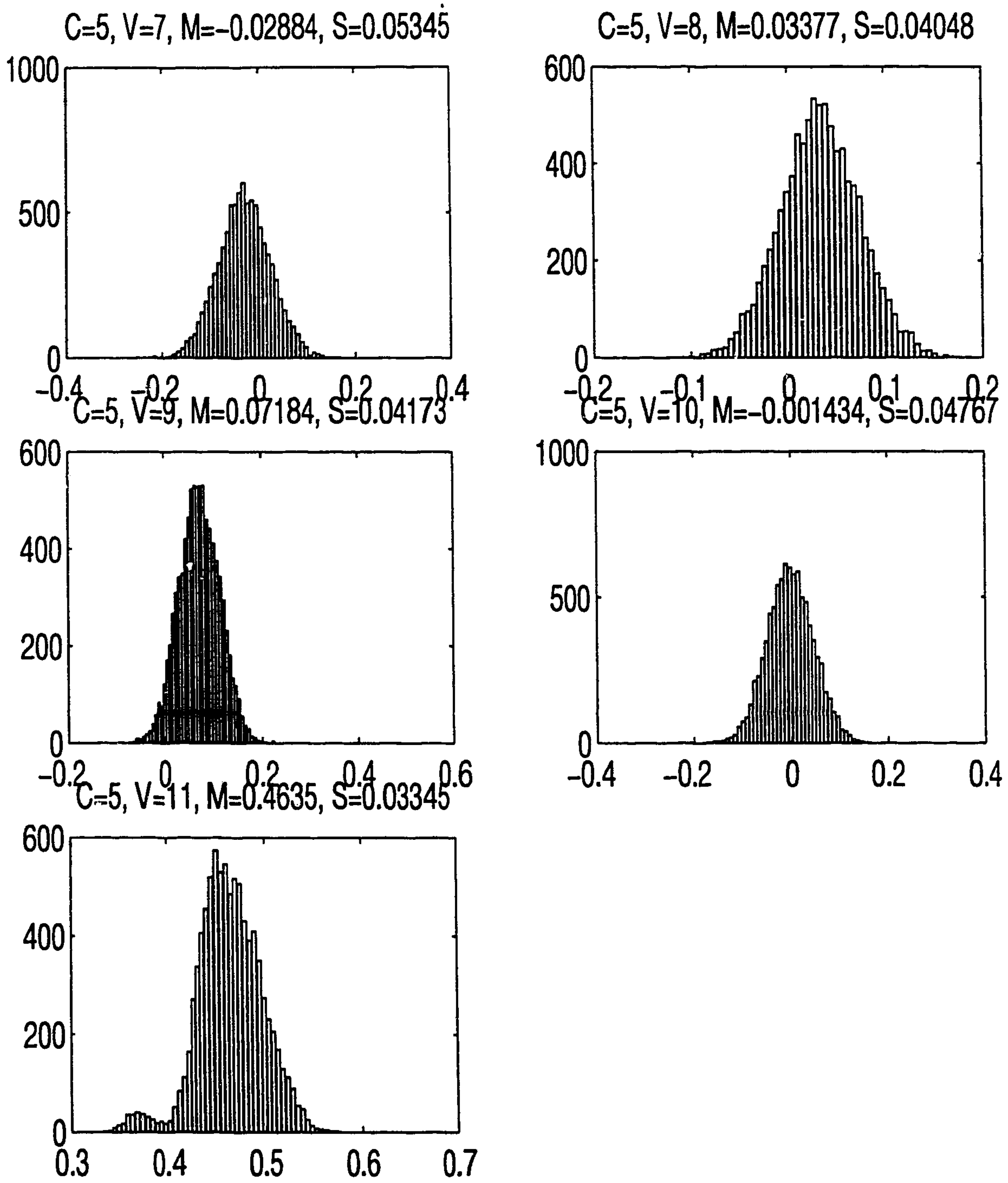


FIGURE 92. Histograms for Class 5.

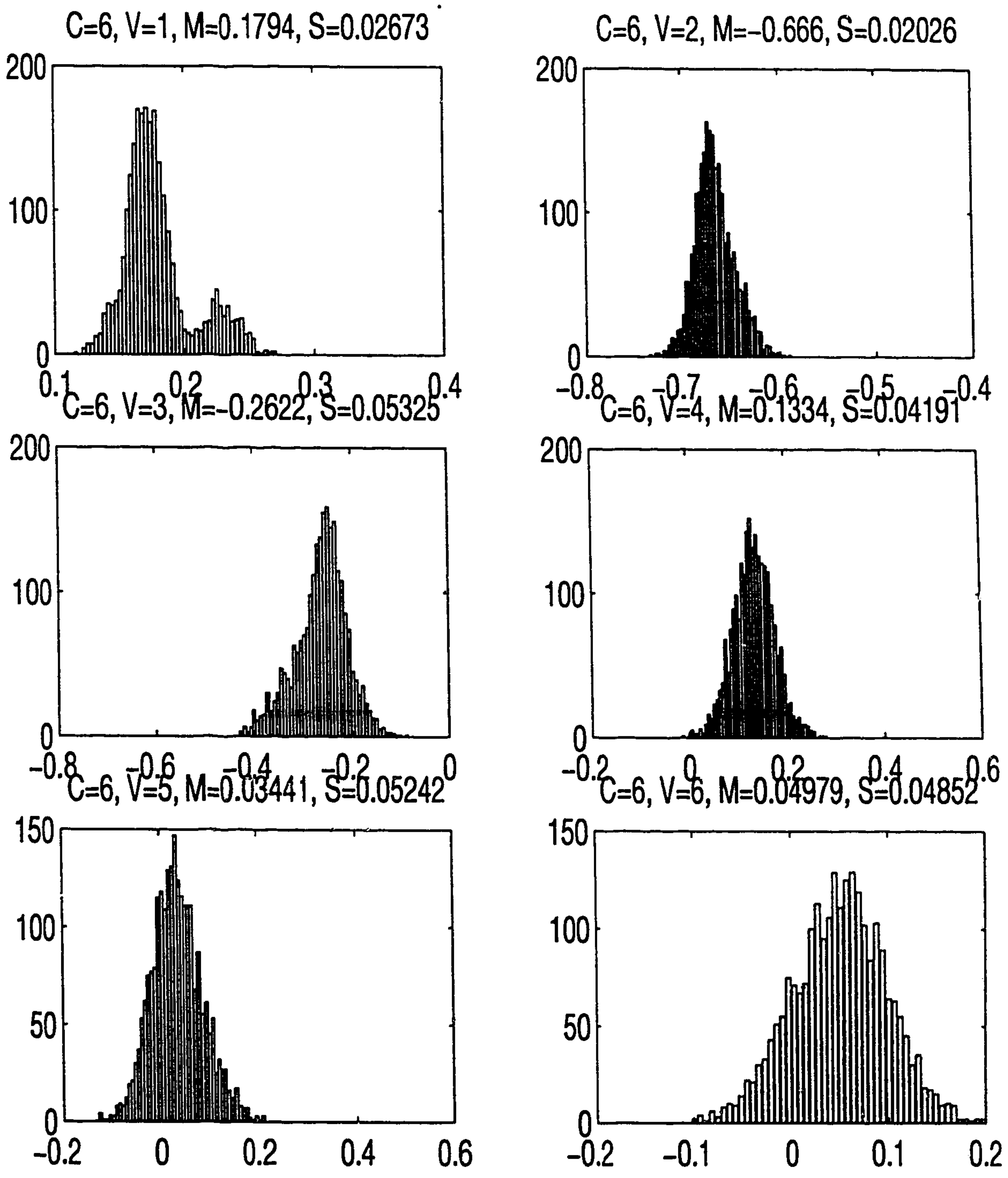


FIGURE 93. Histograms for Class 6.

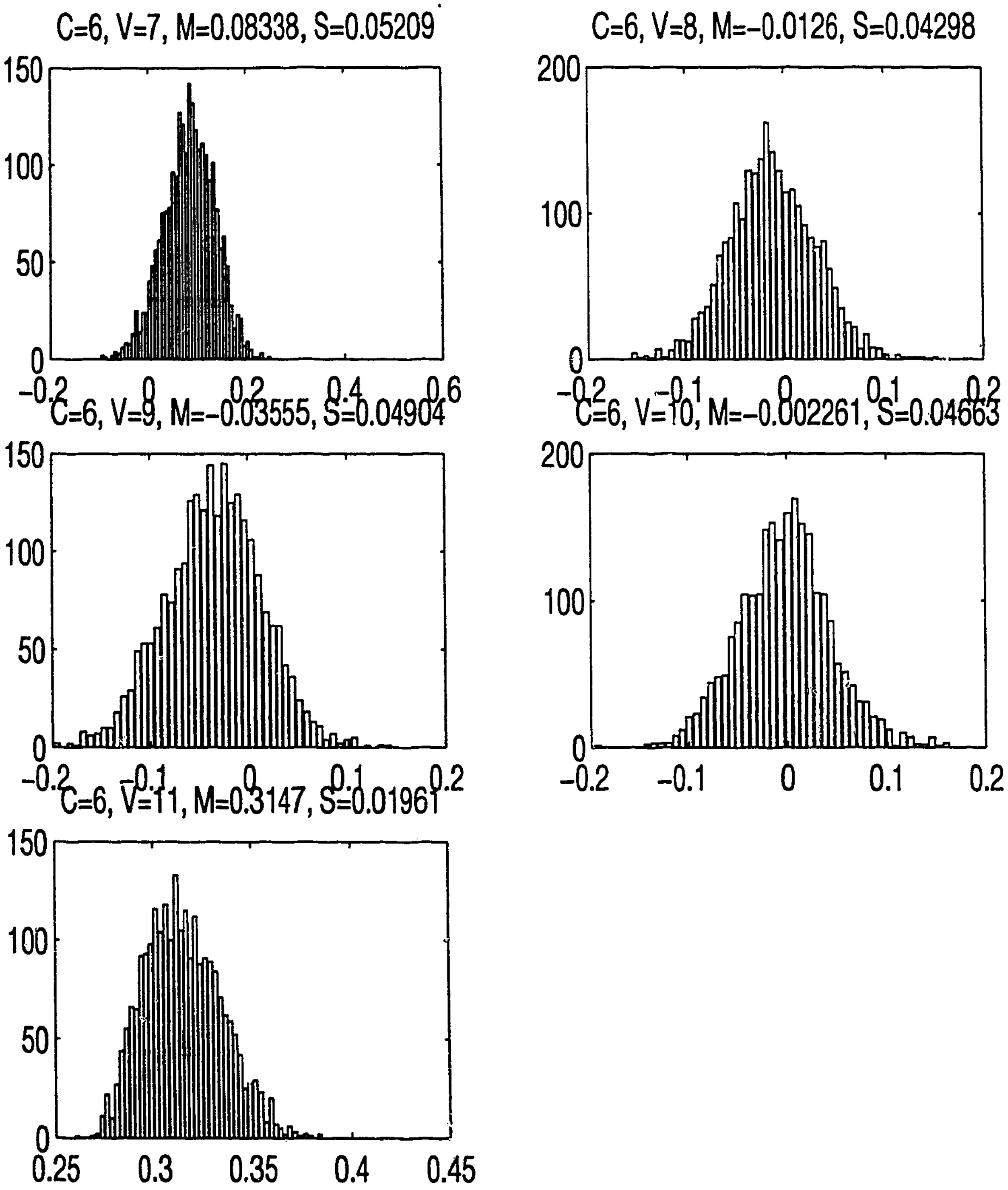


FIGURE 94. Histograms for Class 6.

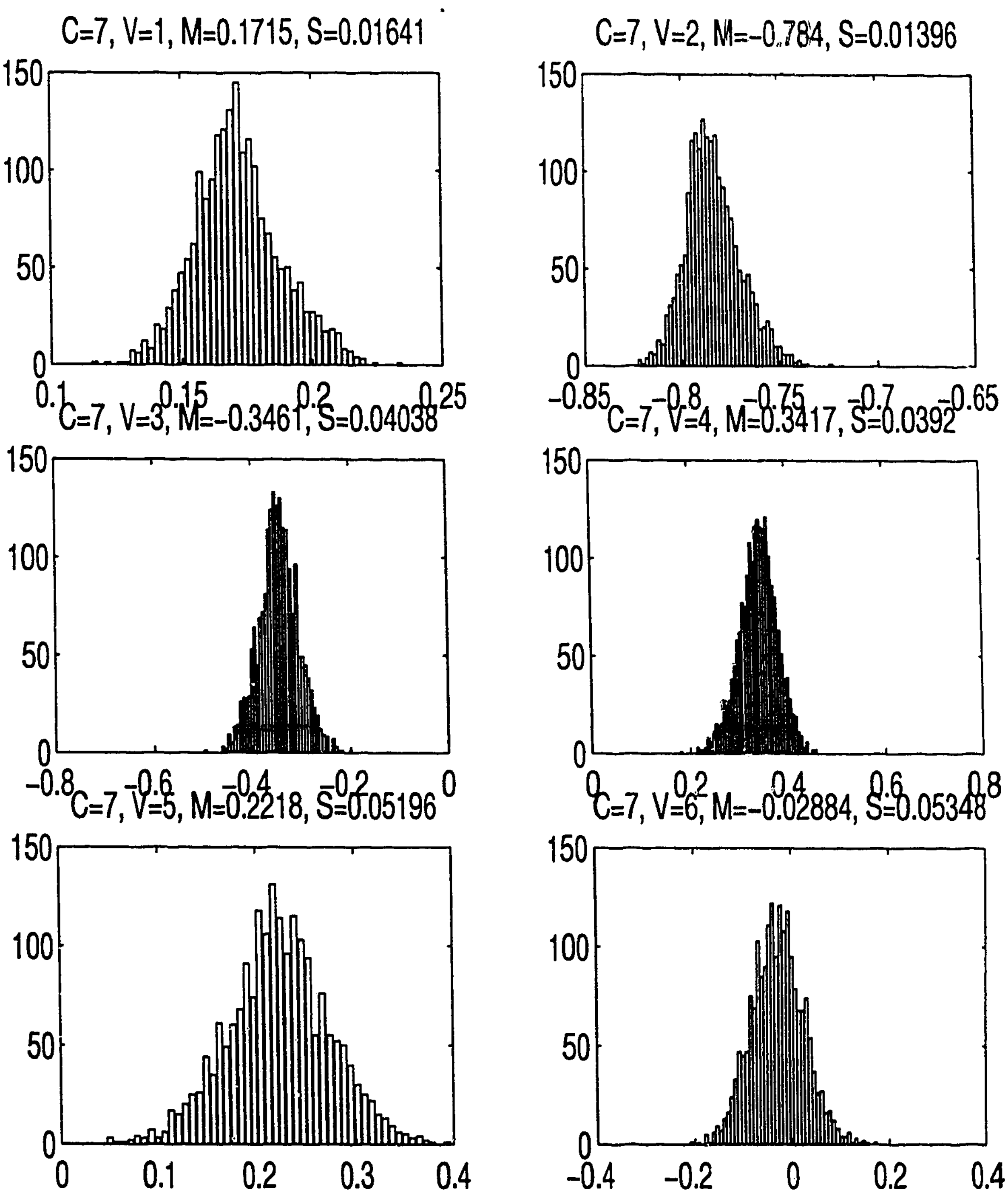


FIGURE 95. Histograms for Class 7.

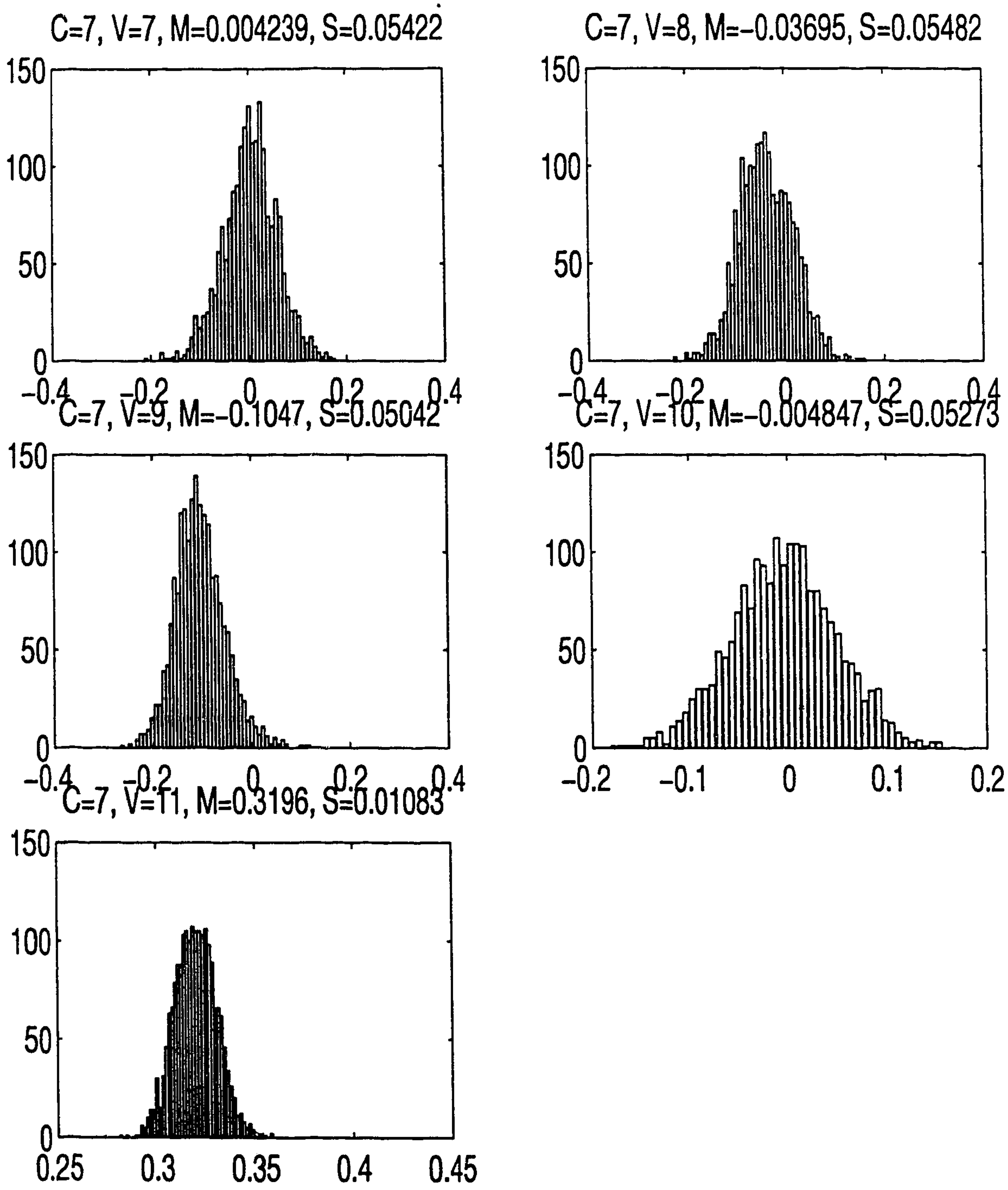


FIGURE 96. Histograms for Class 7.

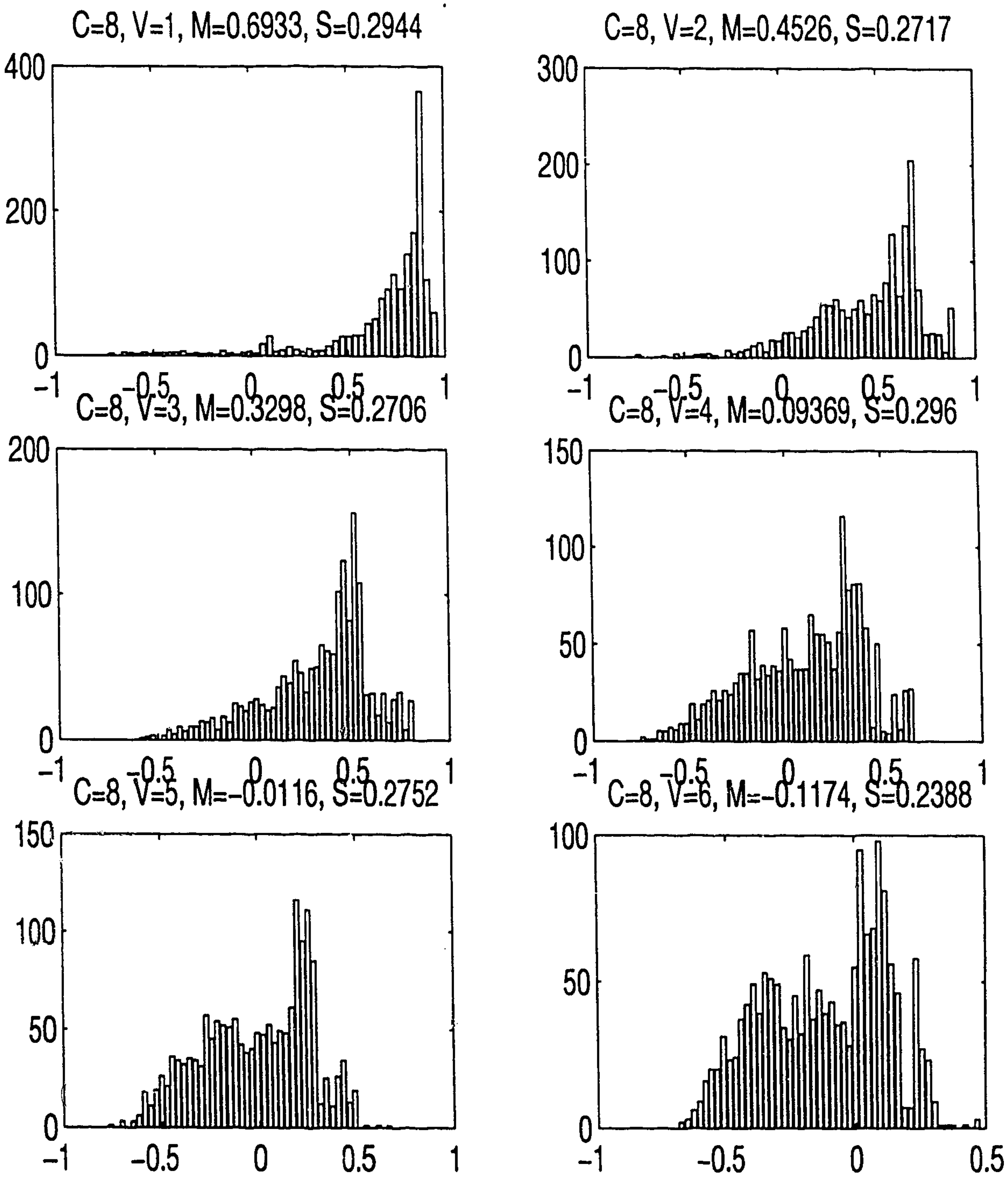


FIGURE 97. Histograms for Class 8.

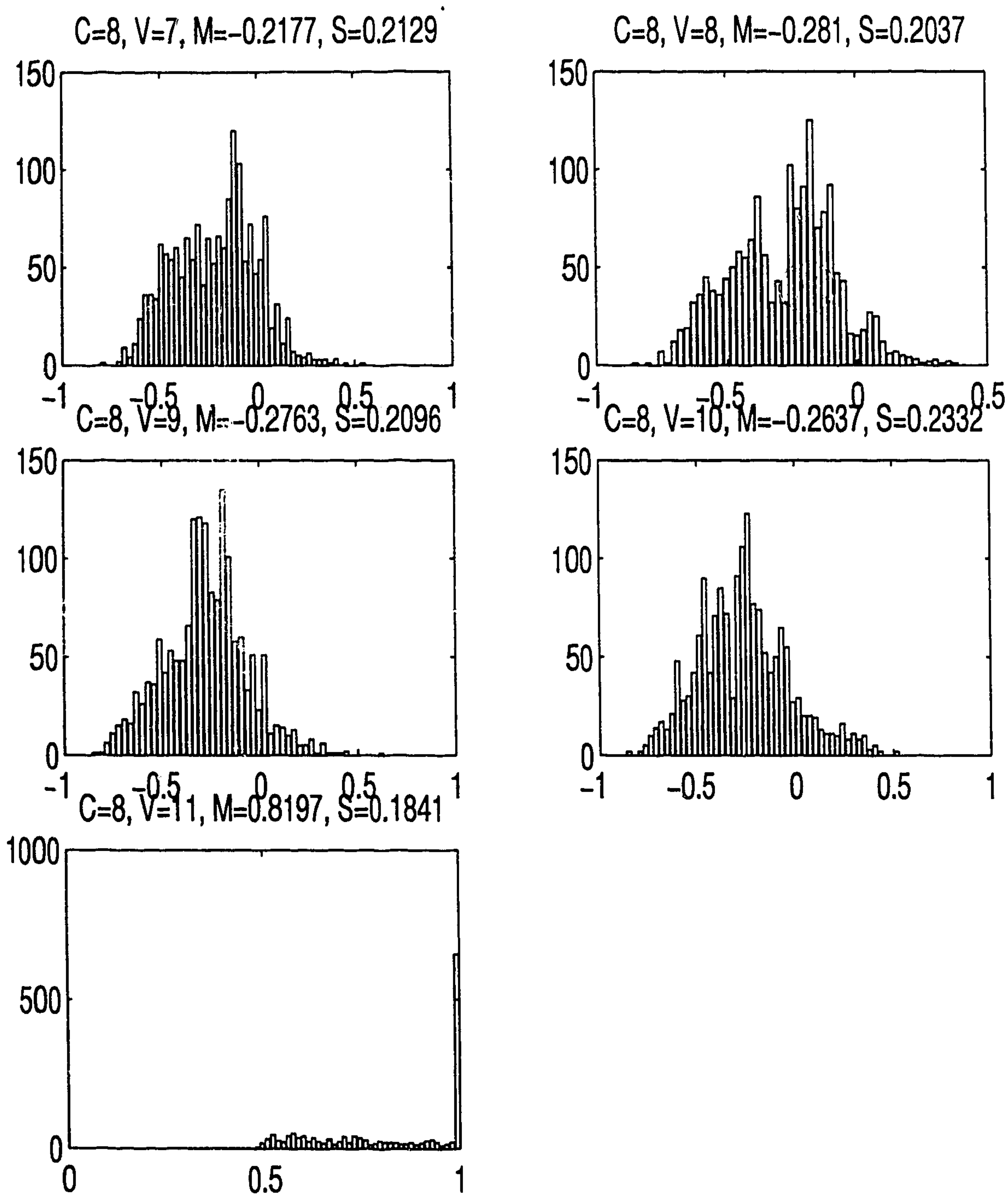
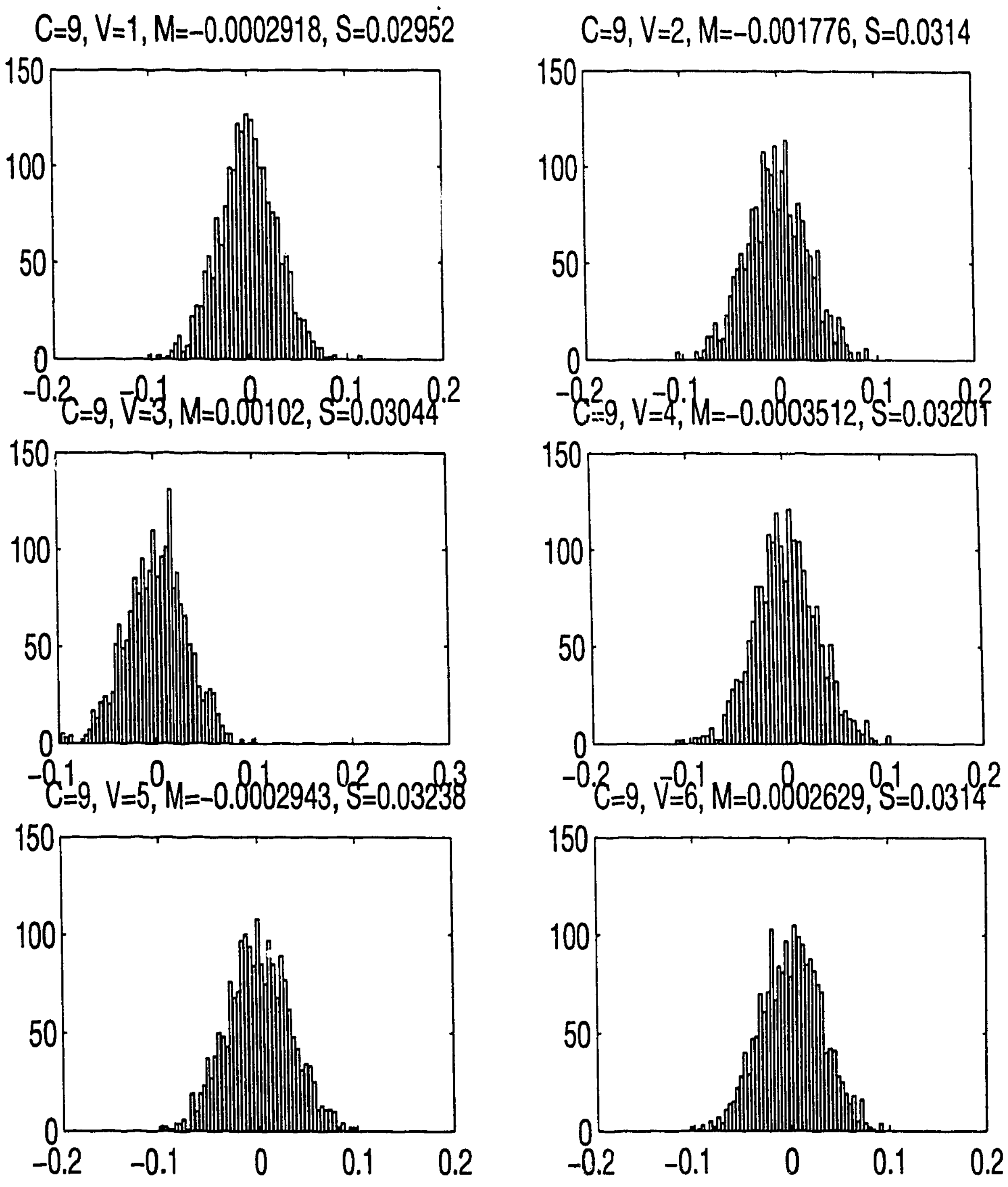


FIGURE 98. Histograms for Class 8.


C=9, V=1, M=-0.0002918, S=0.02952

C=9, V=2, M=-0.001776, S=0.0314

C=9, V=3, M=0.00102, S=0.03044

C=9, V=4, M=-0.0003512, S=0.03201

C=9, V=5, M=-0.0002943, S=0.03238

C=9, V=6, M=0.0002629, S=0.0314

FIGURE 99. Histograms for Class 9.

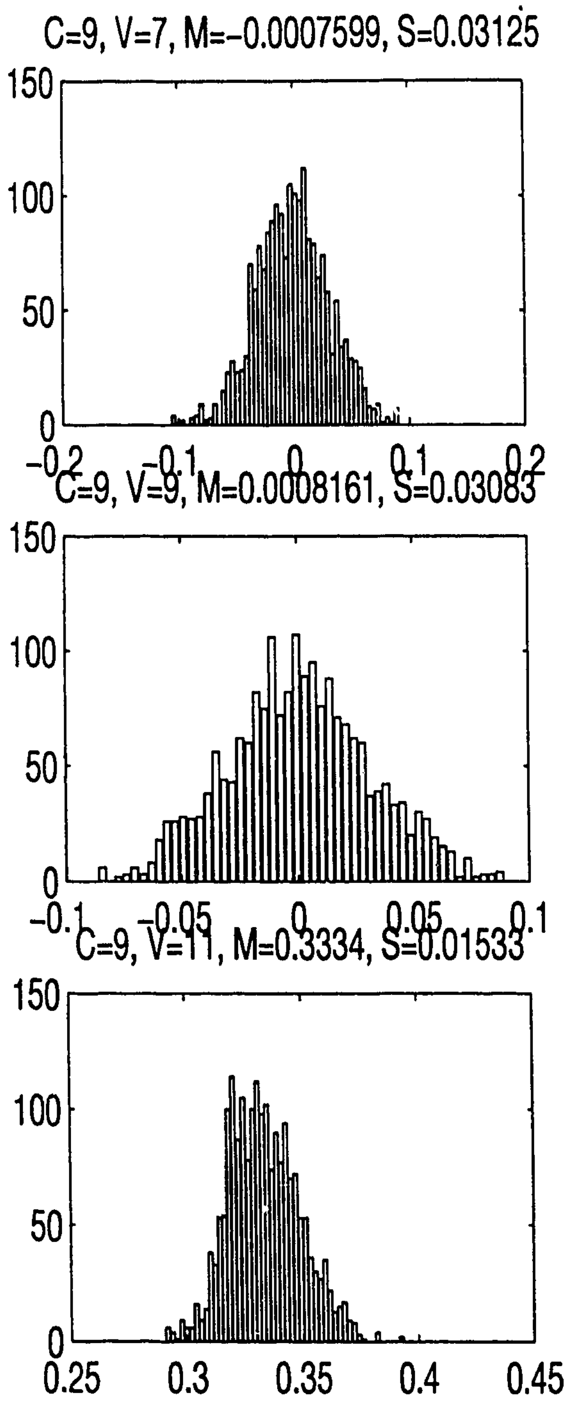


FIGURE 100. Histograms for Class 9.

Appendix M

Function code for computing discriminant variables.

```
function [ind,Rd,n2,ac] = ben6c(d,samp,N,L,threshold)

%  [ind,Rd,n2,ac] = ben6c(d,samp,N,L,threshold)
%
%  This file is a modified implementation of the discriminants that
%  Benvenuto speaks of in a couple of his papers.  The intent
%  is to apply them to my own signals to confirm his results.
%  It will also be interesting to see how well they perform
%  on the modern popular data and fax signals.
%
%  The received signal is passed in through the 'd' vector.  The
%  sampling rate 'samp', segment length 'N', subsegment length 'L',
%  and the threshold level 'threshold' are also passed in as variables.
%
%  The results returned are the indices 'ind', autocorrelation lag
%  values 'Rd', central second-order moment 'n2', and the estimated
%  autocorrelation periodicity 'ac'.
%
%  This file is only a function.  It is assumed that some
%  other program is loading the data files and compiling the
%  results.
%
%  Modifications:  Discriminant functions are applied to
%  the hard rectified passband signal.  The N-point sliding
%  window can now be formed by non-contiguous small L-point
%  segments.  Also, the first 10 lags of the autocorrelation
%  are computed, and the normalized central second-order moment.
%  We attempt to estimate the carrier frequency of the signal
%  by examining the periodicity of the autocorrelation sequence.
%
%  The Benvenuto bug (biased estimator) is fixed (Rd(k) equation)


% Setup the constants.

n = 0;
z = 0;
nsmpl = length(d);
Pth = L*threshold^2;
true = 1;
false = 0;
compute = true;
init = true;
contiguous = false;


while (n < (nsmpl-N-3))
   n = n + 1;
   if (init == true)
   R0 = 0;
   M1 = 0;
   FC = 0;
   clear y;
   q = 0;
   init = false;
end
if (abs(d(n)) > threshold) | (contiguous == true)
   R0temp = sum(d(n:n+L-1).^2);
   M1temp = sum(abs(d(n:n+L-1)));
   ytemp = d(n:n+L-1);
   n = n+L-1;
   if (R0temp >= Pth)
      R0 = R0 + R0temp;
      M1 = M1 + M1temp;
      if exist('y')
         y = [y ytemp];
      else
         y = ytemp;
      end
      q = q + 1;
      if (q < N/L)
         init = false;
      else
         z = z + 1;
         ind(z) = n;
         R0 = R0./N;
         Rd0(z) = R0;
         n2(z) = N*N*R0/(M1^2) - 1;
         Rd(1,z) = (1/(N-1))*sum(y(2:N).*y(1:N-1)) / R0;
         Rd(2,z) = (1/(N-2))*sum(y(3:N).*y(1:N-2)) / R0;
         Rd(3,z) = (1/(N-3))*sum(y(4:N).*y(1:N-3)) / R0;
         Rd(4,z) = (1/(N-4))*sum(y(5:N).*y(1:N-4)) / R0;
         Rd(5,z) = (1/(N-5))*sum(y(6:N).*y(1:N-5)) / R0;
         Rd(6,z) = (1/(N-6))*sum(y(7:N).*y(1:N-6)) / R0;
         Rd(7,z) = (1/(N-7))*sum(y(8:N).*y(1:N-7)) / R0;
         Rd(8,z) = (1/(N-8))*sum(y(9:N).*y(1:N-8)) / R0;
         Rd(9,z) = (1/(N-9))*sum(y(10:N).*y(1:N-9)) / R0;
         Rd(10,z) = (1/(N-10))*sum(y(11:N).*y(1:N-10)) / R0;

         % autocorrelation periodicity estimate
         Rxings = 0;
         Rx0 = 0;
         Rgap = 0;
         yr = [1 Rd(:,z)'];
         prevsign = sign(yr(i));
         for i=2:11
            if sign(yr(i)) ~= prevsign
               Rxings = Rxings + 1;
               Rx1 = i-0.5;
               Rx1 = (i*yr(i-1) - (i-1)*yr(i)) / (yr(i-1) - yr(i));
               Rgap = Rgap + Rx1-Rx0;
               Rx0 = Rx1;
            end
```

```
               prevsign = sign(yr(i));
            end
            if (Rxings ~= 0)
               ac(z) = Rxings./Rgap;
            else
               ac(z) = 1E-10;
            end
            init = true;
         end
         contiguous = true;
      else
         contiguous = false;
      end
   end
end
```

Appendix N

These tables contain detailed results of several different classification tests. The tests are refered to in section 5.3.7.5. Each table of coefficients represents the standardized canonical discriminant function coefficients. Standardization involves normalizing the discriminant variable mean to zero and standard deviation to one. Each table of classification performance results has rows that correspond to actual classes, and columns for predicted class memberships. Each row should total to 100%. Columns do not need to total to 100%. When the table titles refer to {all variables}, the variables used were {Rd1, Rd2,...,Rd10, N2}. Note that we will abbreviate "discriminant function" to DF, "linear function" to LF, and "pseudo-quadratic function" to QF here.

TABLE 49. Linear DF coefficients, *N*=2048, {all variables}

Variable	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
RD1	-.24995	.83985	-1.07196	2.47345	.05965	.05587	-.32484	.66124
RD2	-.19178	-.91710	1.74627	-1.59374	.41581	.01479	.10151	-.12063
RD3	.50344	.87415	-.89361	2.95928	-1.22105	.17307	-.27221	.85300
RD4	.91712	-.46895	.68057	-1.85185	1.39601	-.14051	.49651	-.22860
RD5	-.16190	-.36963	-1.33913	1.42835	-1.24593	-.63160	-.49341	.14238
RD6	-.85085	.10603	.68882	-1.26715	1.11361	.40562	1.15551	-.04408
RD7	.16505	1.11857	-.07845	.70584	-.85855	-.48633	-.99640	-.06162
RD8	-.43157	-.63310	.14679	-.49391	.27957	.60238	1.20296	.41831
RD9	.48812	.059[illegible]9	-.40410	.36958	.05202	.29045	-1.37891	-.23963
RD10	.21088	-.20322	.39309	-.13914	-.14359	-.01352	.77213	.08665
N2	-.07371	-.12736	.24839	.49480	.25273	-.11207	.35508	-.61538

TABLE 50. Classification results, *N*=2048, LFs, {all variables}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	97.9%	2.1%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	13.2%	86.8%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	99.6%	.4%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	.1%	99.9%	.0%	.0%
8	.0%	.0%	2.2%	.5%	.1%	.0%	.0%	96.4%	.7%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 51. Pseudo-quadratic DF coefficients, *N*=2048, {all variables}

Variable	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
RD1	-.24995	.83985	-1.07196	2.47345	.05965	.05587	-.32484	.66124
RD2	-.19178	-.91710	1.74627	-1.59374	.41581	.01479	.10151	-.12063
RD3	.50344	.87415	-.89361	2.95928	-1.22105	.17307	-.27221	.85300
RD4	.91712	-.46895	.68057	-1.85185	1.39601	-.14051	.49651	-.22860
RD5	-.16190	-.36963	-1.33913	1.42835	-1.24593	-.63160	-.49341	.14238
RD6	-.85085	.10603	.68882	-1.26715	1.11361	.40562	1.15551	-.04408
RD7	.16505	1.11857	-.07845	.70584	-.85855	-.48633	-.99640	-.06162
RD8	-.43157	-.63310	.14679	-.49391	.27957	.60238	1.20296	.41831
RD9	.48812	.05939	-.40410	.36958	.05202	.29045	-1.37891	-.23963
RD10	.21088	-.20322	.39309	-.13914	-.14359	-.01352	.77213	.08665
N2	-.07371	-.12736	.24839	.49480	.25273	-.11207	.35508	-.61538

TABLE 52. Classification results, *N*=2048, QFs, {all variables}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	99.9%	.0%	.0%	.0%	.0%	.1%	.0%
4	.0%	.0%	.0%	99.6%	.4%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	1.2%	98.8%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%
8	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 53. Linear DF coefficients, *N*=2048, {Rd2, Rd4, N2}

Variable	Func 1	Func 2	Func 3
RD2	.98768	.25664	-.37653
RD4	-.69322	.82435	-.17639
N2	.28781	.62535	.77297

TABLE 54. Classification results, *N*=2048, LFs, {Rd2, Rd4, N2}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	70.2%	.0%	.0%	.0%	.4%	29.4%	.0%	.0%
3	.0%	.0%	99.6%	.3%	.1%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	80.8%	19.2%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	25.3%	74.7%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%	.0%
7	.0%	.6%	.0%	.0%	.0%	.0%	99.4%	.0%	.0%
8	.0%	.0%	2.5%	.4%	.0%	.0%	.0%	97.2%	.0%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 55. Pseudo-quadratic DF coefficients, *N*=2048, {Rd2, Rd4, N2}

Variable	Func 1	Func 2	Func 3
RD2	.98768	.25664	-.37653
RD4	-.69322	.82435	-.17639
N2	.28781	.62535	.77297

TABLE 56. Classification results, *N*=2048, QFs, {Rd2, Rd2, N2}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	99.9%	.0%	.0%	.0%	.0%	.1%	.0%	.0%
3	.0%	.0%	99.9%	.0%	.0%	.0%	.0%	.1%	.0%
4	.0%	.0%	.4%	86.5%	13.0%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	12.3%	87.7%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%
8	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 57. Linear DF coefficients, *N*=512, {all variables}

Variable	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
RD1	-.36425	1.34053	1.03093	2.47237	.18353	.03495	-.36154	.67388
RD2	-.03837	-1.51311	-1.41704	-1.52113	.30044	.04890	.09669	-.13417
RD3	.36162	1.12645	.73837	2.83531	-.80386	.04034	-.22640	.82268
RD4	.86699	-.70537	-.52240	-1.92915	1.05177	-.05183	.47675	-.25104
RD5	-.32874	.14538	1.22238	1.48278	-.82359	-.67469	-.36665	.21934
RD6	-.53013	-.14269	-.69246	-1.36328	.75249	.51443	1.01927	.01101
RD7	.01994	.93809	-.20948	.86356	-.54130	-.60667	-.82443	-.03249
RD8	-.28553	-.54750	.00757	-.62003	.07417	.61445	1.08208	.36419
RD9	.25904	.17378	.32969	.38129	.14078	.25709	-1.17469	-.14479
RD10	.25863	-.26465	-.23884	-.15826	-.18058	.06064	.60799	.11818
N2	-.06053	-.20435	-.16878	.46653	.28597	-.05138	.41314	-.61736

TABLE 58. Classification results, *N*=512, LFs, {all variables}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	94.5%	5.5%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	18.4%	81.6%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	98.8%	1.2%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	1.1%	98.9%	.0%	.0%
8	.2%	.3%	4.3%	.7%	.2%	.0%	.1%	93.3%	.9%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 59. Pseudo-quadratic DF coefficients, *N*=512, {all variables}

Variable	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
RD1	-.36425	1.34053	1.03093	2.47237	.18353	.03495	-.36154	.67388
RD2	-.03837	-1.51311	-1.41704	-1.52113	.30044	.04890	.09669	-.13417
RD3	.36162	1.12645	.73837	2.83531	-.80386	.04034	-.22640	.82268
RD4	.86699	-.70537	-.52240	-1.92915	1.05177	-.05183	.47675	-.25104
RD5	-.32874	.14538	1.22238	1.48278	-.82359	-.67469	-.36665	.21934
RD6	-.53013	-.14269	-.69246	-1.36328	.75249	.51443	1.01927	.01101
RD7	.01994	.93809	-.20948	.86356	-.54130	-.60667	-.82443	-.03249
RD8	-.28553	-.54750	.00757	-.62003	.07417	.61445	1.08208	.36419
RD9	.25904	.17378	.32969	.38129	.14078	.25709	-1.17469	-.14479
RD10	.25863	-.26465	-.23884	-.15826	-.18058	.06064	.60799	.11818
N2	-.06053	-.20435	-.16878	.46653	.28597	-.05138	.41314	-.61736

TABLE 60. Classification results, *N*=512, QFs, {all variables}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	99.7%	.3%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	1.3%	98.7%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%
8	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 61. Linear DF coefficients, *N*=512, {Rd2, Rd4, N2}

Variable	Func 1	Func 2	Func 3
RD2	.94581	.31939	-.34876
RD4	-.62518	.84004	-.13620
N2	.30445	.56866	.81364

TABLE 62. Classification results, *N*=512, LFs, {Rd2, Rd4, N2}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	66.1%	.0%	.0%	.0%	2.1%	31.8%	.0%	.0%
3	.0%	.0%	97.6%	1.8%	.6%	.0%	.0%	.0%	.0%
4	.0%	.0%	.3%	75.6%	24.0%	.0%	.0%	.0%	.0%
5	.0%	.0%	.1%	27.3%	72.5%	.0%	.0%	.0%	.0%
6	.0%	.3%	.0%	.0%	.0%	99.4%	.3%	.0%	.0%
7	.0%	4.3%	.0%	.0%	.0%	.1%	95.6%	.0%	.0%
8	.2%	.1%	3.0%	.7%	.0%	.0%	.0%	95.6%	.4%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 63. Pseudo-quadratic DF coefficients, *N*=512, {Rd2, Rd4, N2}

Variable	Func 1	Func 2	Func 3
RD2	.94581	.31939	-.34876
RD4	-.62518	.84004	-.13620
N2	.30445	.56866	.81364

TABLE 64. Classification results, *N*=512, QFs, {Rd2, Rd4, N2}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	99.7%	.0%	.0%	.0%	.0%	.3%	.0%	.0%
3	.0%	.0%	99.7%	.3%	.0%	.0%	.0%	.0%	.0%
4	.0%	.0%	.9%	80.1%	19.0%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	14.1%	85.9%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%
8	.0%	.0%	.1%	.0%	.1%	.0%	.0%	99.8%	.0%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 65. Linear DF coefficients, *N*=1024, {all variables}

Variable	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
RD1	-.35435	-1.10414	-.37773	2.32395	.22843	.01469	-.33560	.65070
RD2	-.01436	1.46422	.79196	-1.54786	.26519	.10931	.09098	-.15263
RD3	.38727	-.71864	-.12245	2.45770	-.65652	-.13206	-.15528	.79573
RD4	.73497	.47378	.08954	-1.75528	.96644	.11315	.46332	-.29310
RD5	-.34637	-.11988	-.80730	1.44319	-.58799	-.79466	-.26094	.27205
RD6	-.41328	.04943	.42774	-1.26659	.53319	.66863	.97925	.03791
RD7	.00706	-.64834	.45702	.79074	-.32387	-.78105	-.71818	-.02058
RD8	-.29409	.35624	-.19717	-.61194	-.09992	.65768	1.02963	.35090
RD9	.18302	-.18199	-.17148	.35253	.14358	.18812	-1.04662	-.08217
RD10	.19693	.20751	.07401	-.20268	-.23259	.11684	.50732	.15162
N2	-.03282	.28135	.17979	.43115	.25659	.01754	.43456	-.64732

1	2	3	4	5	6	7	8	9
00.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
.0%	.0%	99.6%	.1%	.3%	.0%	.0%	.0%	.0%
.0%	.0%	.0%	90.5%	9.4%	.1%	.0%	.0%	.0%
.0%	.0%	.0%	23.4%	76.4%	.3%	.0%	.0%	.0%
.0%	.0%	.0%	.0%	.0%	96.5%	3.5%	.0%	.0%
.0%	.0%	.0%	.0%	.0%	2.2%	97.8%	.0%	.0%
.2%	.4%	5.5%	1.1%	.4%	.0%	.1%	91.4%	.9%
.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

Pseudo-quadratic DF coefficients, *N*=1024, {all variables}

Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
-.35435	-1.10414	-.37773	2.32395	.22843	.01469	-.33560	.65070
-.01436	1.46422	.79196	-1.54786	.26519	.10931	.09098	-.15263
.38727	-.71864	-.12245	2.45770	-.65652	-.13206	-.15528	.79573
.73497	.47378	.08954	-1.75528	.96644	.11315	.46332	-.29310
-.34637	-.11988	-.80730	1.44319	-.58799	-.79466	-.26094	.27205
-.41328	.04943	.42774	-1.26659	.53319	.66863	.97925	.03791
.00706	-.64834	.45702	.79074	-.32387	-.78105	-.71818	-.02058
-.29409	.35624	-.19717	-.61194	-.09992	.65768	1.02963	.35090
.18302	-.18199	-.17148	.35253	.14358	.18812	-1.04662	-.08217
.19693	.20751	.07401	-.20268	-.23259	.11684	.50732	.15162
-.03282	.28135	.17979	.43115	.25659	.01754	.43456	-.64732

TABLE 68. Classification results, N=1024, QFs, {all variables}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	98.9%	1.1%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	2.5%	97.5%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	.0%	100.0%	.0%	.0%
8	.0%	.0%	.0%	.1%	.0%	.0%	.0%	99.9%	.0%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 69. Linear DF coefficients, N=1024, {Rd2, Rd4, N2}

Variable	Func 1	Func 2	Func 3
RD2	.92283	.35041	-.31009
RD4	-.50952	.89489	-.09891
N2	.32375	.52005	.85098

TABLE 70. Classification results, N=1024, LFs, {Rd2, Rd4, N2}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	62.5%	.0%	.0%	.0%	4.7%	32.9%	.0%	.0%
3	.0%	.0%	94.7%	4.5%	.8%	.0%	.0%	.0%	.0%
4	.0%	.0%	1.6%	69.7%	28.5%	.1%	.0%	.0%	.0%
5	.0%	.0%	1.0%	30.6%	68.0%	.3%	.0%	.0%	.0%
6	.0%	1.3%	.0%	.0%	.3%	96.3%	2.1%	.0%	.0%
7	.0%	7.6%	.0%	.0%	.0%	1.2%	91.2%	.0%	.0%
8	.4%	.2%	5.6%	1.0%	.0%	.0%	.0%	91.7%	1.0%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 71. Pseudo-quadratic DF coefficients, *N*=1024, {Rd2, Rd4, N2}

Variable	Func 1	Func 2	Func 3
RD2	.92283	.35041	-.31009
RD4	-.50952	.89489	-.09891
N2	.32375	.52005	.85098

TABLE 72. Classification results, *N*=1024, QFs, {Rd2, Rd4, N2}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	98.7%	.0%	.0%	.0%	.0%	1.3%	.0%	.0%
3	.0%	.0%	98.6%	1.4%	.0%	.0%	.0%	.1%	.0%
4	.0%	.0%	2.3%	72.9%	24.8%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	17.0%	83.0%	.0%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.0%	99.9%	.0%	.0%	.0%
7	.0%	.7%	.0%	.0%	.0%	.1%	99.3%	.0%	.0%
8	.0%	.0%	.5%	.1%	.1%	.0%	.0%	99.4%	.0%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 73. Linear DF coefficients, *N*=256, {all variables}

Variable	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
RD1	.30499	1.45488	2.09478	.63224	.11874	.08988	-.14655	.60308
RD2	.18099	-1.86126	-1.47462	.13386	.02911	.16260	-.05174	-.23428
RD3	-.52422	1.50313	2.00174	.61923	-.36055	-.29958	.31201	.84105
RD4	-.66398	-1.38505	-1.41490	-.29464	.66887	.58682	.10324	-.58106
RD5	.42407	.79580	1.52209	-.06952	.14862	-.86501	.09025	.54295
RD6	.21748	-.43935	-1.17588	-.15763	.00029	.95445	.78726	-.43358
RD7	.05135	.78573	.35215	.73896	.23057	-.84673	-.56364	.43452
RD8	.22641	-.31711	-.45882	-.57484	-.35988	.58719	.75103	-.31798
RD9	-.09611	.12239	.35803	.03961	-.03775	-.02170	-.88392	.61614
RD10	-.14411	-.10655	-.21645	-.13823	-.21230	-.00158	.19433	-.34404
N2	.05902	-.14981	.36933	.33446	.03003	.05724	.04131	-.77163

TABLE 74. Classification results, *N*=256, LFs, {all variables}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	70.2%	28.9%	.8%	.1%	.0%	.0%	.0%
4	.0%	.0%	.0%	70.8%	28.8%	.3%	.0%	.0%	.0%
5	.0%	.0%	.0%	24.7%	62.0%	12.8%	.5%	.0%	.0%
6	.0%	.0%	.0%	1.2%	.0%	65.6%	33.1%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	11.9%	88.1%	.0%	.0%
8	.6%	.8%	7.5%	.4%	.8%	.0%	.3%	88.2%	1.4%
9	.0%	.0%	.4%	.0%	.0%	.0%	.0%	.0%	99.6%

TABLE 75. Pseudo-quadratic DF coefficients, *N*=256, {all variables}

Variable	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
RD1	.30499	1.45488	2.09478	.63224	.11874	.08988	-.14655	.60308
RD2	.18099	-1.86126	-1.47462	.13386	.02911	.16260	-.05174	-.23428
RD3	-.52422	1.50313	2.00174	.61923	-.36055	-.29958	.31201	.84105
RD4	-.66398	-1.38505	-1.41490	-.29464	.66887	.58682	.10324	-.58106
RD5	.42407	.79580	1.52209	-.06952	.14862	-.86501	.09025	.54295
RD6	.21748	-.43935	-1.17588	-.15763	.00029	.95445	.78726	-.43358
RD7	.05135	.78573	.35215	.73896	.23057	-.84673	-.56364	.43452
RD8	.22641	-.31711	-.45882	-.57484	-.35988	.58719	.75103	-.31798
RD9	-.09611	.12239	.35803	.03961	-.03775	-.02170	-.88392	.61614
RD10	-.14411	-.10655	-.21645	-.13823	-.21230	-.00158	.19433	-.34404
N2	.05902	-.14981	.36933	.33446	.03003	.05724	.04131	-.77163

TABLE 76. Classification results, *N*=256, QFs, {all variables}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
3	.0%	.0%	80.8%	19.0%	.0%	.0%	.0%	.2%	.0%
4	.0%	.0%	1.1%	84.6%	14.3%	.0%	.0%	.0%	.0%
5	.0%	.0%	.4%	17.3%	80.7%	1.6%	.0%	.0%	.0%
6	.0%	.0%	.0%	.0%	.3%	90.7%	9.1%	.0%	.0%
7	.0%	.0%	.0%	.0%	.0%	.2%	99.8%	.0%	.0%
8	.0%	.0%	.3%	.1%	.1%	.0%	.0%	99.4%	.2%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 77. Pseudo-quadratic DF coefficients, *N*=256, {Rd2, Rd4, N2}

Variable	Func 1	Func 2	Func 3
RD2	.98479	.04836	-.28539
RD4	.17543	1.06067	-.08609
N2	.35179	.35952	.92781

TABLE 78. Classification results, *N*=256, QFs, {Rd2, Rd4, N2}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	92.0%	.0%	.0%	.0%	.6%	7.4%	.0%	.0%
3	.0%	.0%	83.7%	15.3%	.9%	.0%	.0%	.1%	.0%
4	.0%	.0%	4.3%	72.3%	23.3%	.0%	.0%	.0%	.0%
5	.0%	.0%	1.1%	37.8%	51.3%	9.8%	.0%	.0%	.0%
6	.0%	.4%	.0%	.0%	1.4%	66.4%	31.9%	.0%	.0%
7	.0%	3.5%	.0%	.0%	.0%	1.5%	95.0%	.0%	.0%
8	.0%	.0%	1.3%	.0%	.8%	.0%	.0%	97.8%	.1%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.1%	99.9%

TABLE 79. Linear DF coefficients, *N*=256, {Rd2, Rd4, N2}

Variable	Func 1	Func 2	Func 3
RD2	.98479	.04836	-.28539
RD4	.17543	1.06067	-.08609
N2	.35179	.35952	.92781

TABLE 80. Classification results, *N*=256, LFs, {Rd2, Rd4, N2}

Class	1	2	3	4	5	6	7	8	9
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	55.0%	.0%	.0%	.0%	20.8%	24.2%	.0%	.0%
3	.0%	.1%	75.3%	22.3%	2.2%	.1%	.0%	.0%	.0%
4	.0%	.4%	6.7%	55.3%	37.3%	.3%	.0%	.0%	.0%
5	.1%	.3%	6.6%	45.1%	35.2%	12.4%	.3%	.0%	.0%
6	.0%	2.6%	.0%	.0%	2.8%	63.4%	31.1%	.0%	.0%
7	.0%	5.5%	.0%	.0%	.0%	11.9%	82.6%	.0%	.0%
8	.5%	.6%	8.2%	.7%	.1%	.0%	.0%	87.1%	2.8%
9	.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

Appendix O

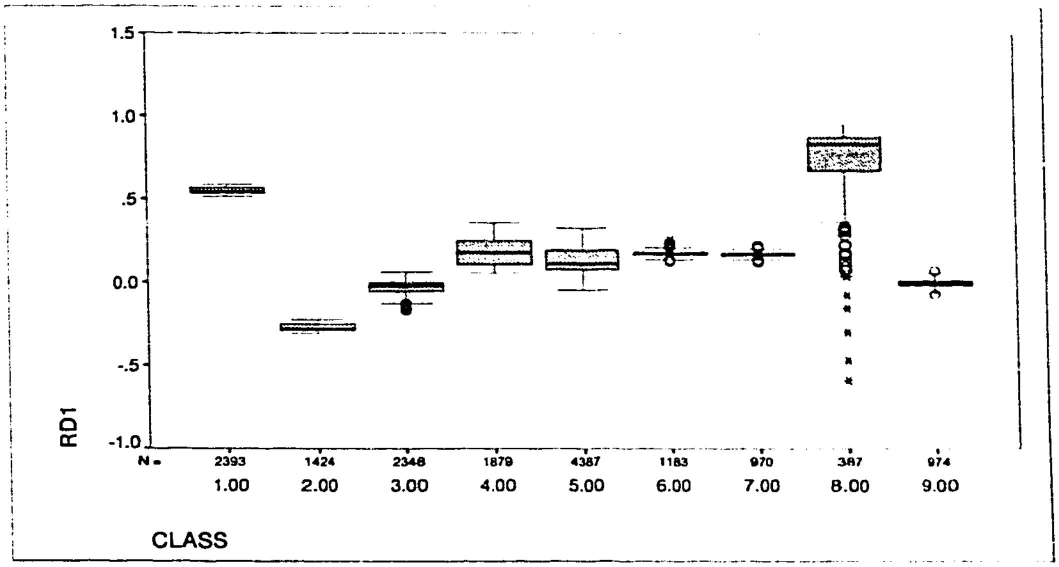


FIGURE 101. Boxplots of Rd1 vs. class.

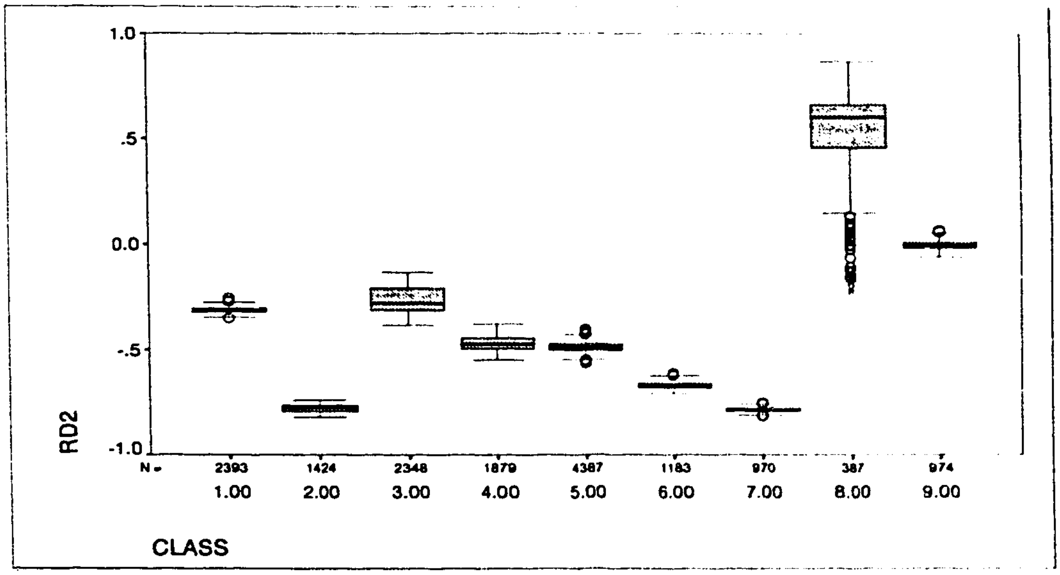


FIGURE 102. Boxplots of Rd2 vs. class.

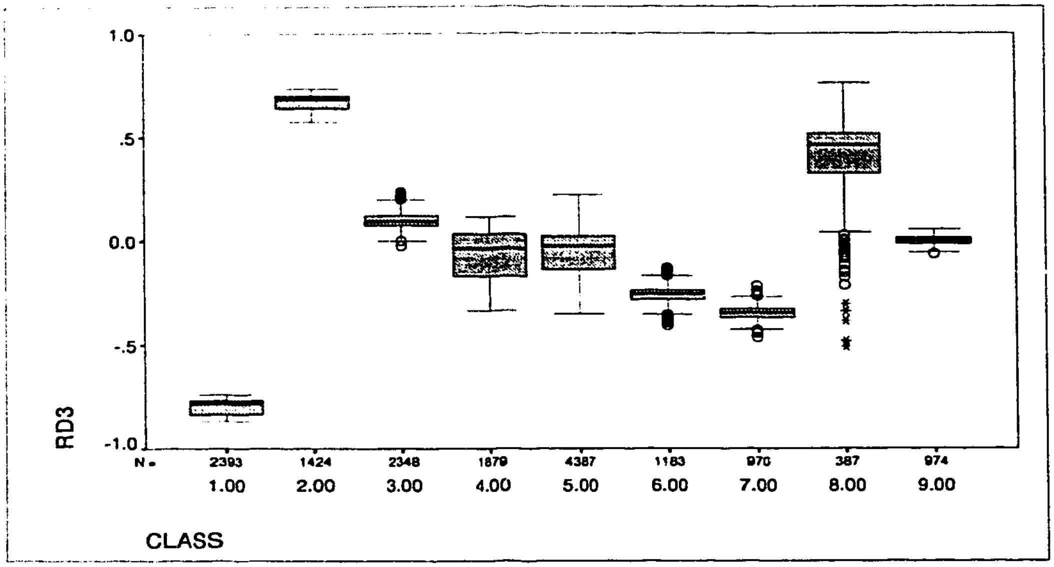


FIGURE 103. Boxplots of Rd3 vs. class.

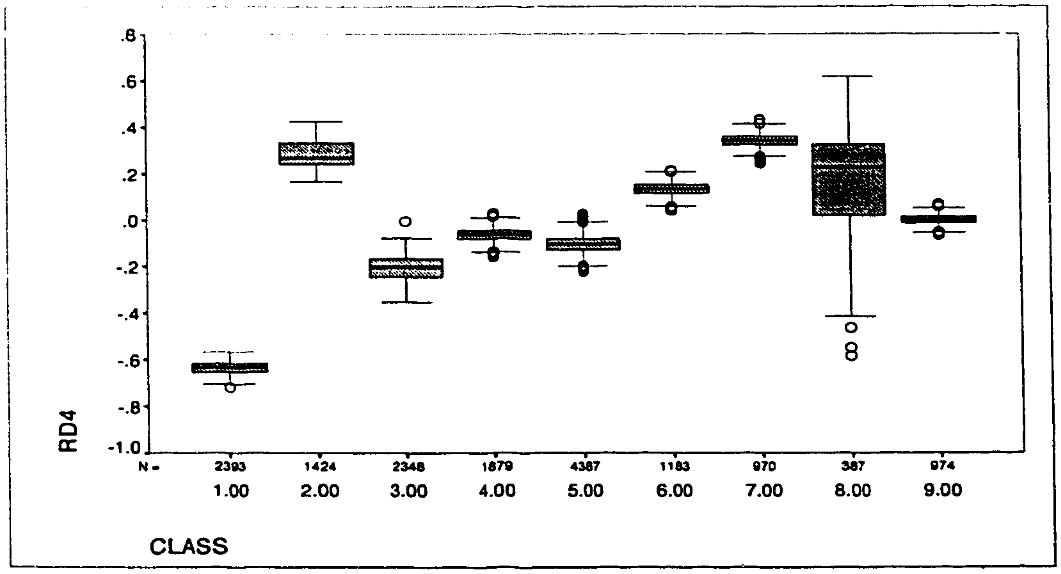


FIGURE 104. Boxplots of Rd4 vs. class.

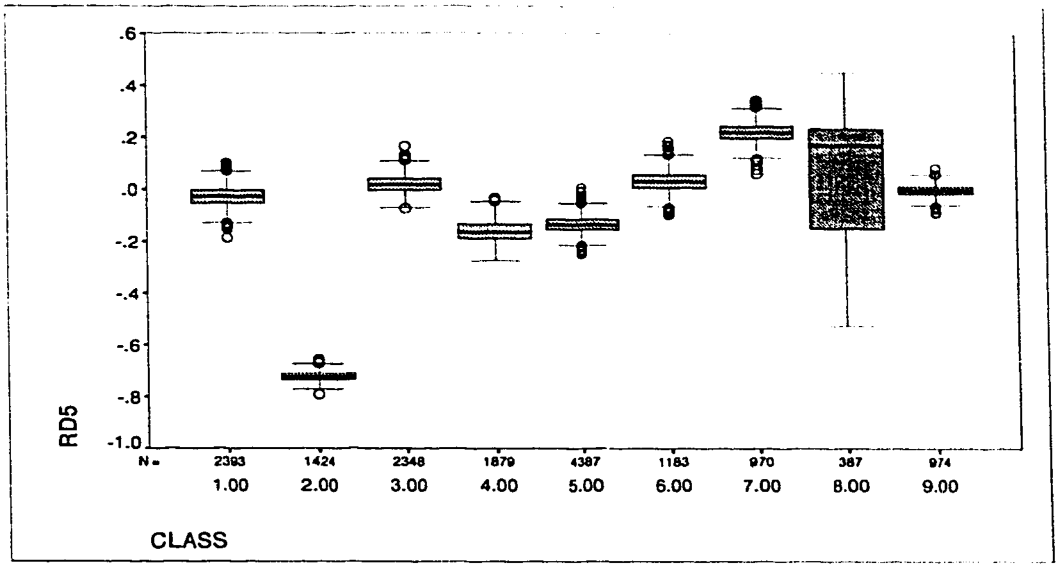


FIGURE 105. Boxplots of Rd5 vs. class.

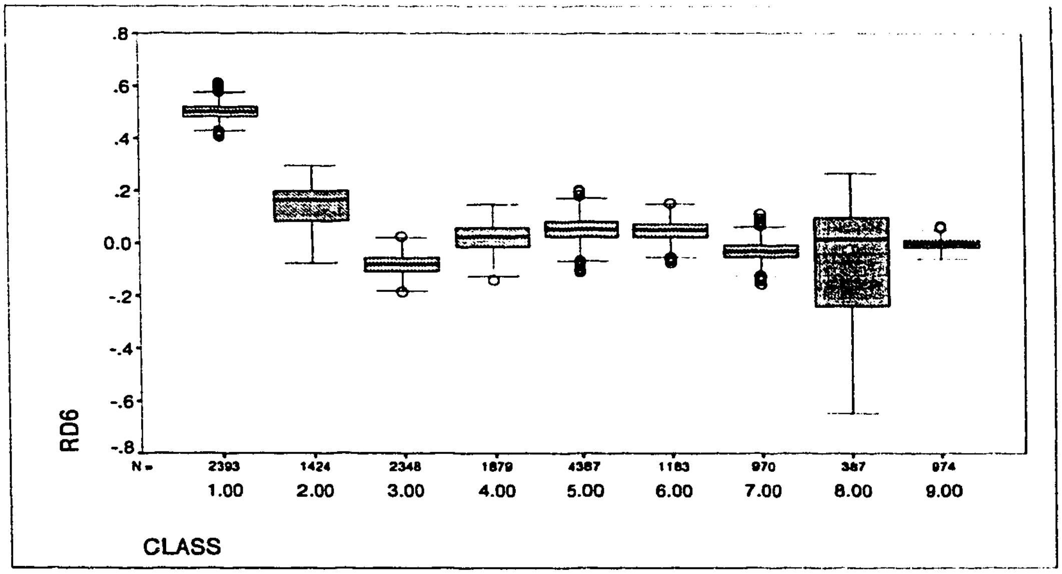


FIGURE 106. Boxplots of Rd6 vs. class.

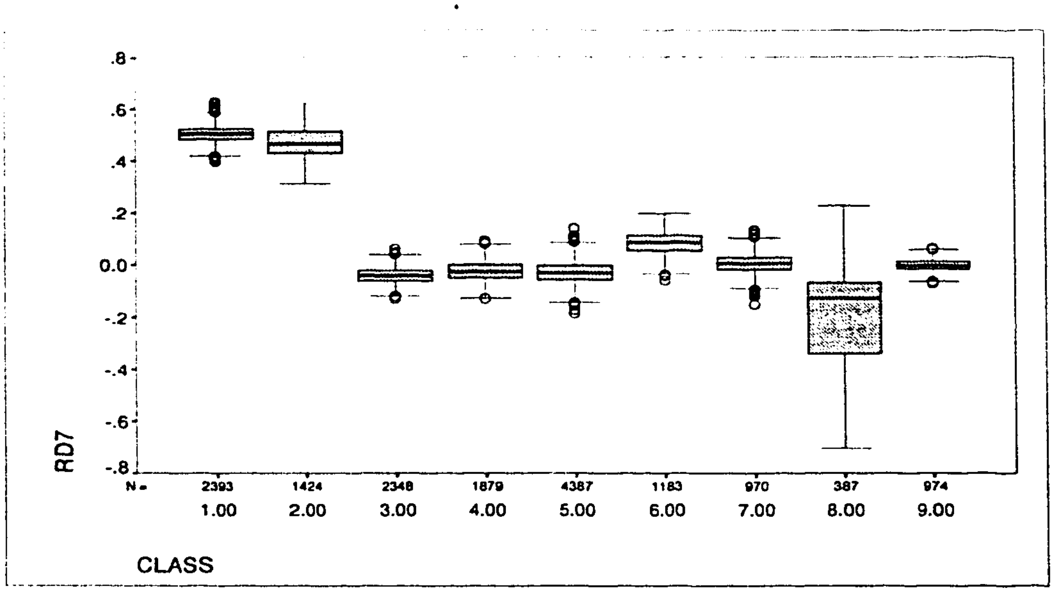


FIGURE 107. Boxplots of Rd7 vs. class.

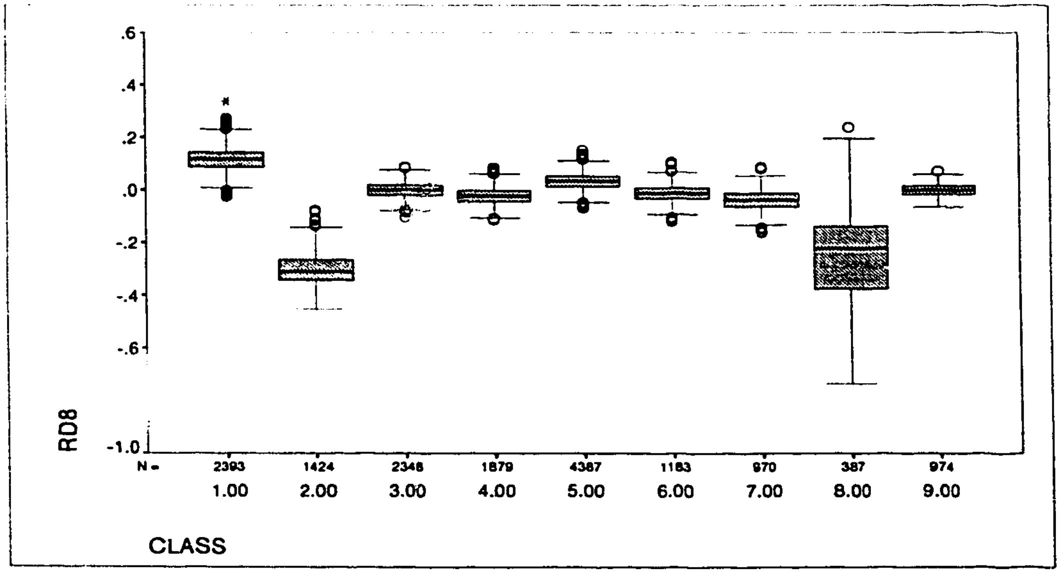


FIGURE 108. Boxplots of Rd8 vs. class.

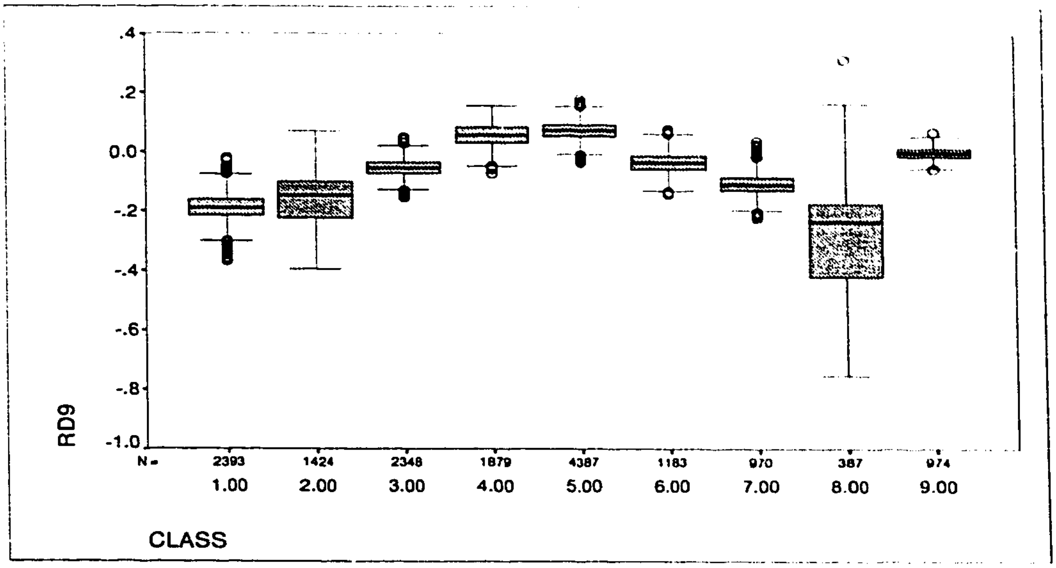


FIGURE 109. Boxplots of Rd9 vs. class.

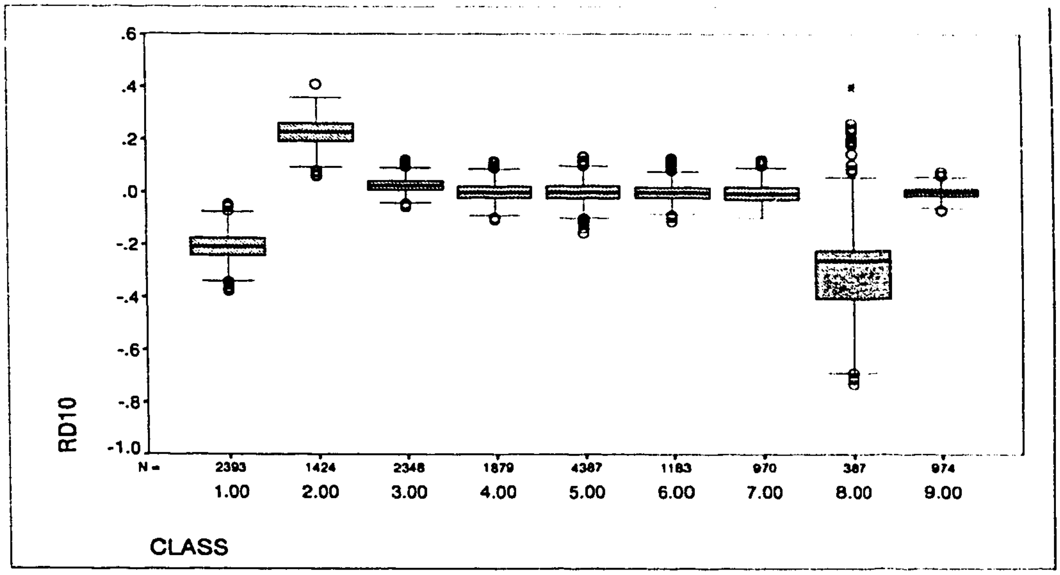


FIGURE 110. Boxplots of Rd10 vs. class.

Appendix P

```
function [y,fclk] = qamtiming4(d,samp,fsnom,offset)

% function [y,fclk] = qamtiming4(d,samp,fsnom,offset)
%
% This function extracts the timing signal from
% a baseband QAM signal, and performs slicing.
% It is critical that a QAM signal be operated
% on.
%
% d is the signal.  samp is the sampling rate of d.
% fsnom is the nominal baud rate.
%
% y is the resulting sliced signal (still passband).
% fclk is the baud rate
% clock that was generated.
%
% This version allows for a selectable delay (offset).
%


% first, normalize power
d = d ./ mean(abs(d).^2).^.5;

% next, perform a non-linear operation.
dd = abs(d);

% next, perform notch filtering at fsnom.
N = 4;           % Nth order filter
fd = 6;          % filter width = 2*fd Hz.
W1 = (fsnom - fd) ./ (samp./2);
W2 = (fsnom + fd) ./ (samp./2);
[b,a] = butter(N,[W1 W2]);
fclk = filter(b,a,dd);

% now, develop the baud clock and slice
fclker = interp(fclk,9);        % no delay caused by this element
fclker = sign(fclker);
flen = length(fclker);
fclkslice = fclker(2:flen) - fclker(1:flen-1);
fclkslice(fclkslice > 0) = zeros(size(fclkslice(fclkslice > 0)));
fclkslice = [abs(fclkslice) zeros(1,35)];
dd = interp(d,9);
fclkslice = fclkslice(offset+1:length(dd)+offset);
y = dd(fclkslice == 2);
```

Appendix Q

Function code for computing minimum distances between constellations.

```
function [mindis,ABavg] = mindisqam6(y,sym,symfreq)

% FUNCTION [mindis,ABavg] = mindisqam6(y,sym,symfreq)
%
% This function is used to determine the minimum
% distance between a list of constellation points
% from a test signal (y), and a given constellation
% pattern (sym).
%
% ABavg is returned and contains the average of the
% freq of occurance of each symbol point minus the
% ideal freq of occurance.  This is a measure of how
% close the measured points are to the expected distribution.
%
% The minimum distances for each point are returned.
%
% Also, the frequency of occurance of each constellation
% point is utilized.
%
% There is lots of room for optimization here.


% Only the constellation vector mags
% are considered.

ii = sqrt(-1);
yabs = abs(y);

% find all the distances
for sympoint = 1:length(sym)
   dis(sympoint,:) = abs(yabs(:)-sym(sympoint))';
end

% select the minimum distances
[mindis,minverts] = min(dis);

for i = 1:length(sym)
   AB(i) = sum(minverts(minverts==i)./i)./length(minverts) - symfreq(i);
end

ABavg = sum(abs(AB));
```

Appendix R

Code for supervising the classification of V.29 signals into their fall-back modes.

```
% This program is intended to perform subclassification
% of V.29 fax signals.  There are two speeds for the
% V.29 standard, one with a 16 point constellation, and
% the other with 8 points.  This program will distinguish
% the two (with luck).  There is also a 4 point version,
% but it is so rarely used that we ignore it.
%
% Seventh version.
%
% Only FIR filters are used throughout the program, so that
% the phases remain linear, and group-delays are easily computed.
%
% Constellation magnitudes are used for discrimination, since
% the phases are not locked, and may rotate and/or jitter.
%


clear;

samp = 8000;
fc = 1700;
fs = 2400;
fsnom = 2400;
pass = 2000;
M = 1000;
offset = 12;

% form the constellation templates (mags only)
% and their relative freqs of occurance.
sym8 = [2.^0.5, 3];
sym8freq = [1, 1]./2;
sym16 = [2.^0.5, 18.^0.5, 3, 5];
sym16freq = [1, 1, 1, 1]./4;
sym8 = sym8 ./ (mean(abs(sym8).^2)).^.5;
sym16 = sym16 ./ (mean(abs(sym16).^2)).^.5;


% load in the files
fid = fopen('v29files.txt', 'r');
s = fscanf(fid,'%s');
z = 0;
start = 1;
for i = 1:length(s)-3
   if s(i:i+3) == '.mat'
      z = z+1;
      fname(z,1:i+3-start+1) = s(start:i+3);
      start = i+4;
   end
end
```

```
fclose(fid);

[nfiles,junk] = size(fname);
for file = 1:nfiles
   file
   fname(file,:)
   eval(['load ',fname(file,:)]);
   if exist('recording')
      r = mu2lin(recording);
   else
      return;
   end

   % loop through the signal from start to finish
   nsegs = floor(length(r)./M);
   for j = 1:nsegs
      d = r(j*M-M+1:j*M);

      % first demodulate the signal.  Total delay at this
      % point should be the number of FIR taps divided by 2.
      % In this case, the delay is 15 sample intervals.
      t = [0:1/samp:(length(d)-1)/samp];
      d = qamdemod2(d,fc,t',pass,samp);

      % next, recover the signal constellation
      [y,fclk] = qamtiming4(d,samp,fsnom,offset);

      % finally, identify which of the two possible
      % constellations the signal under test is.
      [d8,ab8(j)] = mindisqam6(y,sym8,sym8freq);
      [d16,ab16(j)] = mindisqam6(y,sym16,sym16freq);

      j
      d8sum = sum(d8);
      d16sum = sum(d16);

      d8avg(j) = d8sum./length(d8);
      d16avg(j) = d16sum./length(d16);

      d8tot(j) = d8avg(j) .* ab8(j);
      d16tot(j) = d16avg(j) .* ab16(j);
      d8tot(j)
      d16tot(j)
   end     % loop through signal segments

   N8 = length(d8tot(d8tot<d16tot));
   N16 = length(d16tot(d16tot<d8tot));
   Ntot = length(d8tot);
   p8 = 100 .* N8 ./ Ntot;
   p16 = 100 .* N16 ./ Ntot;

   clf;
   subplot(2,2,1)
   plot(y,'o')
   axis([-2 2 -2 2]);
   title(['Constellation plot for ' num2str(fname(file,:))]);

   subplot(2,2,2)
   axis([0 1 0 1]);
   text(0.1,0.9,['N = ' num2str(M)]);
   text(0.1,0.7,['Nsegs = ' num2str(nsegs)]);
   text(0.1,0.5,['8-QAM = ' num2str(p8) ' %']);
   text(0.1,0.3,['16-QAM = ' num2str(p16) ' %']);

   subplot(2,1,2)
   plot(d8tot,'o')
   hold on
   plot(d16tot,'x')
   title('Mean min distance * ABavg vs. time');
   xlabel('time');
   ylabel('distance');
   legend('8-QAM','16-QAM',-1);
   plot(d8tot)
   plot(d16tot)
   hold off

   eval(['print plot' num2str(file) '.ps']);

   clear ab8 ab16 d8tot d16tot d8avg d16avg d8sum d16sum

end     % loop through signals
```

Appendix S

These tables contain detailed results of several different classification tests. The tests are referred to in section 7.3. Each table of coefficients represents either the standardized canonical discriminant function coefficients, or the Fisher's linear discriminant function coefficients. Standardization involves normalizing the discriminant variable mean to zero and standard deviation to one. Fisher's linear discriminant functions also require a constant term which is indicated by "*K*". Each Fisher function corresponds to a particular class, where the class number matches the function number (Func). Each table of classification performance results has rows that correspond to actual classes, and columns for predicted class memberships. Each row should total to 100%. Columns do not need to total to 100%. Four discriminant variables are used in all cases {Rd2, Rd4, Rd6, N2}. Note that we abbreviate standardized canonical "linear discriminant function" to LDF and "pseudo-quadratic discriminant function" to QDF here.

TABLE 81. LDF coefficients, *N* = 256.

Variable	Func 1	Func 2	Func 3	Func 4
N2	.37589	.15454	-.09385	.98024
RD2	.94070	-.25530	.25065	-.19930
RD4	.36074	.81108	.59863	.17954
RD6	-.24348	-.41104	.80483	.38827

TABLE 82. Fisher's linear coefficients, *N* = 256.

Var	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
N2	68.9408893	92.5273968	98.8631297	92.1039105	71.9853824	77.0362075	162.4976303	74.9532904
RD2	-41.5041534	-72.2110898	-30.4878477	-49.3729779	-68.1814534	-72.8896109	59.0690823	4.3349719
RD4	-38.1766011	27.7235116	.9706675	3.7444776	18.6267597	28.3422519	49.5752588	15.4056933
RD6	48.6869443	24.5381971	2.3901780	13.1703926	11.2718850	7.5155273	5.3802994	6.7737870
K	-47.0296219	-53.7488137	-31.3387053	-35.4116082	-40.0959504	-48.0008993	-73.0128182	-14.5715506

TABLE 83. Classification performance using LDFs, $N = 256$.

Class	1	2	3	4	5	6	7	8
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	80.6%	.0%	.0%	.9%	18.5%	.0%	.0%
3	.0%	.0%	78.7%	21.2%	.1%	.0%	.0%	.0%
4	.0%	.0%	6.2%	85.8%	7.5%	.5%	.0%	.0%
5	.0%	3.1%	.0%	3.0%	62.1%	31.8%	.0%	.0%
6	.0%	10.0%	.0%	.0%	12.6%	77.4%	.0%	.0%
7	.6%	.4%	7.7%	1.1%	.0%	.2%	87.0%	2.9%
8	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 84. LDF coefficients, $N = 512$.

Variable	Func 1	Func 2	Func 3	Func 4
N2	.31206	.38807	.27768	.88798
RD2	.89667	.39950	.14320	-.29686
RD4	-.55630	.69118	.53048	-.04099
RD6	.12553	-.56108	.82598	.09628

TABLE 85. Fisher's linear coefficients, $N = 512$.

Var	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
N2	98.8684419	96.3009518	132.1051800	115.9214429	71.2215833	73.2573147	246.4018611	100.3102699
RD2	-26.6660383	-128.989680	-19.4524852	-64.0273201	-108.846837	-133.804106	112.9150626	14.7063583
RD4	-66.6393461	68.6131589	-3.5571063	14.3498112	43.3349205	74.6856453	34.2905873	12.7411665
RD6	83.2226033	31.8952040	3.5267741	20.0756413	16.8995700	5.8356922	2.6437965	8.7499022
K	-68.6162354	-82.9198122	-38.5813891	-44.1584080	-52.8452080	-78.9177852	-121.130481	-18.7892564

TABLE 86. Classification performance using LDFs, N = 512.

Class	1	2	3	4	5	6	7	8
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	83.7%	.0%	.0%	.0%	16.3%	.0%	.0%
3	.0%	.0%	96.5%	3.5%	.0%	.0%	.0%	.0%
4	.0%	.0%	1.1%	98.6%	.3%	.0%	.0%	.0%
5	.0%	.0%	.0%	.3%	94.8%	4.8%	.0%	.0%
6	.0%	9.3%	.0%	.0%	.8%	89.9%	.0%	.0%
7	.3%	.2%	5.6%	1.1%	.0%	.1%	91.9%	.9%
8	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 87. LDF coefficients, N = 1024.

Variable	Func 1	Func 2	Func 3	Func 4
N2	.27784	.41643	.73927	-.53910
RD2	.88887	.45409	-.09413	.34201
RD4	-.71858	.60322	.38270	.34385
RD6	.24283	-.63133	.68385	.35529

TABLE 88. Fisher's linear coefficients, N = 1024.

Var	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
N2	123.2569353	113.1319312	158.0008225	138.8236206	84.9611686	85.8179829	309.4182095	117.1221580
RD2	-22.2941525	-179.810882	-25.9900121	-86.3626576	-149.000230	-187.091105	141.0266184	15.4155629
RD4	-99.0214389	100.0245438	-4.5363261	22.2868936	67.4829930	113.5418477	24.5405249	11.2339518
RD6	126.4382237	32.5223787	5.8365520	25.6547899	16.4018352	-4.1136448	-2.7307604	9.6204267
K	-94.1096225	-110.464366	-46.0553549	-54.6101832	-69.9756077	-108.588365	-162.127542	-21.5913603

TABLE 89. Classification performance using LDFs, $N = 1024$.

Class	1	2	3	4	5	6	7	8
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	84.1%	.0%	.0%	.0%	15.9%	.0%	.0%
3	.0%	.0%	98.6%	1.4%	.0%	.0%	.0%	.0%
4	.0%	.0%	.2%	99.8%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	.0%	98.6%	1.4%	.0%	.0%
6	.0%	4.6%	.0%	.0%	.2%	95.3%	.0%	.0%
7	.3%	.1%	2.9%	.5%	.0%	.1%	95.7%	.4%
8	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 90. LDF coefficients, $N = 2048$.

Variable	Func 1	Func 2	Func 3	Func 4
N2	.24412	.43734	.85548	-.30648
RD2	.90334	.48511	-.22422	.29733
RD4	-.84565	.58913	.25275	.41962
RD6	.32791	-.70151	.55662	.41796

TABLE 91. Fisher's linear coefficients, $N = 2048$.

Var	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8
N2	151.0523890	141.5430710	193.4337839	171.4426100	107.4566020	109.1980553	395.9726214	140.9630241
RD2	-23.8556946	-252.484026	-37.7819814	-120.509451	-206.758830	-261.935671	180.6489616	15.5765093
RD4	-142.921084	142.0703258	-1.3749801	35.3535529	100.0839426	166.0545712	20.2096102	12.4637840
RD6	183.7362644	32.9252593	11.3313737	34.5581576	16.8939263	-16.3335006	-12.9843622	10.1619505
K	-128.416437	-150.230948	-56.0835416	-69.9861778	-94.9355214	-150.807812	-220.485510	-25.5648705

TABLE 92. Classification performance using LDFs, $N = 2048$.

Class	1	2	3	4	5	6	7	8
1	100.0%	.0%	.0%	.0%	.0%	.0%	.0%	.0%
2	.0%	82.9%	.0%	.0%	.0%	17.1%	.0%	.0%
3	.0%	.0%	99.8%	.2%	.0%	.0%	.0%	.0%
4	.0%	.0%	.0%	100.0%	.0%	.0%	.0%	.0%
5	.0%	.0%	.0%	.0%	100.0%	.0%	.0%	.0%
6	.0%	1.2%	.0%	.0%	.0%	98.8%	.0%	.0%
7	.0%	.1%	2.3%	.2%	.0%	.0%	97.3%	.0%
8	.0%	.0%	.0%	.0%	.0%	.0%	.0%	100.0%

TABLE 93. QDF coefficients, $N = 2048$.

Variable	Func 1
N2	.16861
RD2	3.39794
RD4	3.47858
RD6	2.51624

TABLE 94. Classification performance using QDFs, $N = 2048$.

Class	1	2
1	100.0%	.0%
2	.7%	99.3%

TABLE 95. QDF coefficients, $N = 1024$.

Variable	Func 1
N2	.14705
RD2	3.43859
RD4	3.70292
RD6	2.29396

TABLE 96. Classification performance using QDFs, $N = 1024$.

Class	1	2
1	99.2%	.8%
2	2.3%	97.7%

TABLE 97. QDF coefficients, N = 512.

Variable	Func 1
N2	.11498
RD2	3.51222
RD4	3.90268
RD6	2.10615

TABLE 98. Classification performance using QDFs, N = 512.

Class	1	2
1	95.2%	4.8%
2	5.3%	94.7%

TABLE 99. QDF coefficients, N = 256.

Variable	Func 1
N2	-.04147
RD2	4.69437
RD4	4.99780
RD6	1.54664

TABLE 100. Classification performance using QDFs, N = 256.

Class	1	2
1	65.9%	34.1%
2	18.7%	81.3%

Appendix T

These tables contain detailed results of several different classification tests. The tests are referred to in section 7.3. Each table of coefficients represents the Fisher's linear discriminant function coefficients. Fisher's linear discriminant functions also require a constant term which is indicated by "*K*". Each Fisher function corresponds to a particular class, where the class number matches the function number (Func). Eleven discriminant variables are used in all cases {N2, Rd1, Rd2, ..., Rd10}, and all nine classes are considered.

TABLE 101. Fisher's linear coefficients, *N* = 2048.

Var	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8	Func 9
N2	212.11694	160.18760	218.92847	212.83918	209.95063	145.95359	145.53652	432.85801	145.79080
RD1	875.94746	825.55012	768.84103	827.41129	819.14977	817.08100	802.18383	844.92785	27.573014
RD10	-222.3448	-103.9965	-99.5957	-158.0692	-145.1447	-183.9603	-178.7361	-60.9235	2.1908949
RD2	-828.6326	-973.1521	-823.7389	-891.1987	-888.1300	-998.7772	-1064.560	-565.2314	-2.6903125
RD3	562.8487	813.52456	731.87374	667.91932	663.75840	647.98885	632.07713	680.49783	12.13639
RD4	-826.5912	-442.9293	-695.7858	-557.26847	-572.9140	-532.1079	-477.8168	-545.1935	1.1046181
RD5	452.55856	240.24371	658.41512	491.96595	502.02530	570.41042	654.89255	412.3547	7.0461214
RD6	-69.8583	-458.0063	-517.7260	-388.3004	-366.1489	-427.2913	-480.8689	-348.7634	8.4251207
RD7	471.17682	781.65222	411.7376	372.94330	339.55805	401.46195	345.29842	279.69814	4.1105047
RD8	-180.8892	-563.61748	-251.62575	-266.14260	-218.21501	-305.68455	-303.03306	-216.78176	[illegible]
RD9	-8.9547528	253.11320	161.79371	221.16412	189.50310	228.43158	233.28887	[illegible]	2.2890360
K	-562.01532	-626.51807	-277.28090	-292.09331	-288.01705	-324.59446	-394.65172	-457.75446	-26.508363

TABLE 102. Fisher's linear coefficients, N = 1024.

Var	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8	Func 9
N2	157.52026	121.05004	177.51978	159.79890	157.68626	106.16280	106.80934	322.97393	119.59128
RD1	720.98922	666.68143	640.39881	675.40431	669.88571	674.51558	665.15746	667.19401	14.666877
RD10	-162.64909	-70.175597	-77.771786	-110.97361	-102.81106	-124.95340	-122.64276	-72.432890	.0596696
RD2	-701.24595	-780.52110	-661.66197	-720.53820	-718.69495	-793.96808	-834.03406	-489.75283	3.1112337
RD3	486.51297	647.90606	600.16278	553.85896	551.70750	533.52400	518.97571	582.22605	10.079708
RD4	-652.50157	-393.16139	-566.03012	-478.98140	-487.49110	-453.77149	-413.60330	-487.55772	.1420894
RD5	423.05409	252.74795	518.08628	416.29049	420.80670	456.91135	502.69134	418.85744	11.710841
RD6	-161.99172	-349.48051	-408.49255	-323.52775	-308.90354	-336.28166	-362.35038	-355.07446	1.6220197
RD7	382.43629	509.19283	325.21736	300.43351	280.84326	312.70000	279.45625	285.12270	9.9491346
RD8	-176.55454	-361.46145	-206.1787	-211.37616	-181.96601	-228.30148	-225.37170	-222.14961	3.5371786
RD9	59.814639	159.99056	121.70271	159.64511	142.19340	155.38438	149.51837	120.08091	3.0247345
K	-405.89428	-437.06818	-224.55890	-232.52371	-230.07654	-254.92774	-300.97085	-339.74418	-22.132205

TABLE 103. Fisher's linear coefficients, N = 512.

Var	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8	Func 9
N2	122.54061	100.08542	146.32854	128.92659	126.38220	85.183005	88.024106	255.06397	101.97651
RD1	450.63477	403.34246	388.69582	417.27319	414.25188	418.34587	413.71283	407.03205	8.1426426
RD10	-86.762209	-41.569175	-41.213856	-60.027695	-53.637975	-67.635284	-68.031190	-50.925592	-2.2901033
RD2	-440.98493	-495.11678	-401.42314	-449.50696	-449.97425	-499.78251	-525.29207	-275.16950	6.5846289
RD3	280.39483	395.09019	366.90390	332.93375	333.41530	315.96817	306.30287	359.56739	6.5688956
RD4	-388.87957	-220.35472	-342.33286	-283.77083	-290.84507	-265.29004	-238.92789	-281.98272	4.4150793
RD5	259.95747	142.56766	313.94046	244.08433	247.70008	267.32930	294.15046	262.37755	10.327426
RD6	-92.849861	-194.89714	-240.29185	-181.99083	-172.36980	-186.79400	-201.39829	-219.86978	1.4983175
RD7	223.26012	283.87743	188.98424	171.01832	159.19450	181.55797	166.02805	172.19002	9.1427099
RD8	-92.018048	-205.56323	-117.51311	-118.32918	-98.902389	-126.70666	-126.90154	-144.42511	-.4044231
RD9	39.508923	90.307140	62.242925	86.492515	75.811429	84.138386	78.968557	68.257182	2.2783011
K	-260.77242	-280.77494	-147.22940	-153.44373	-152.12388	-167.33047	-198.08030	-232.19155	-19.188139

TABLE 104. Fisher's linear coefficients, $N_{\cdot} = 256$.

Var	Func 1	Func 2	Func 3	Func 4	Func 5	Func 6	Func 7	Func 8	Func 9
N2	104.60118	88.060527	111.461706	104.56803	101.05457	86.625760	91.876740	177.18566	78.410786
RD1	300.54292	260.44255	275.91353	280.35038	277.77726	276.34326	274.51542	294.29989	7.8429886
RD10	-42.884286	-20.195304	-29.015259	-29.936673	-27.611802	-32.386575	-31.083152	-42.788981	-4.3384005
RD2	-291.19417	-336.90810	-291.48994	-305.29581	-309.51153	-325.98768	-331.04532	-203.11726	-.4073058
RD3	200.73788	286.11467	241.78955	238.38596	240.84467	234.01465	231.91237	235.52741	1.3063380
RD4	-268.74683	-170.84959	-219.32192	-209.39714	-209.45708	-201.01082	-193.39020	-173.19145	8.1212519
RD5	204.23512	128.96740	198.25914	181.62025	181.53638	197.96925	204.99183	183.56553	9.2370252
RD6	-99.532179	-137.10018	-150.37602	-131.69555	-125.44946	-139.17863	-142.52341	-152.48215	-.1162775
RD7	153.06955	166.48529	122.17120	112.48479	105.16578	120.79578	114.63930	120.06448	10.728666
RD8	-64.172989	-115.99222	-78.593489	-71.031853	-62.677691	-77.109970	-75.654119	-107.34144	-4.6792982
RD9	27.286895	42.347318	42.247304	44.292668	39.027337	41.643159	37.035810	46.257314	3.6606442
K	-165.71855	-180.98446	-103.65630	-106.68331	-106.95710	-112.61481	-121.90664	-157.78074	-15.260450