UNIVERSITY OF CALIFORNIA

Los Angeles

Target Identification Processor For Wireless Sensor Network

A dissertation submitted in partial satisfaction of the

requirements for the degree

Doctor of Philosophy in Electrical Engineering

by

Tommy Yu

1999

The dissertation of Tommy Yu is approved.

Rajive Bagrodia

William Kaiser

Kung Yao

Gregory J. Pottie, Committee Chair

University of California, Los Angeles

1999

DEDICATION

This dissertation is dedicated to my parents.

Contents

List of Figures

ACKNOWLEDGEMENTS

First of all, I would like to express my deepest gratitude to my advisor, Professor Pottie, for his full support, encouragement and guidance. I would also like to give thanks to Professor Kung Yao, Professor William Kaiser, and Professor Rajive Bagrodia for investing their time to be on my dissertation committee and for their valuable input.

Next, I extend my thanks to my colleagues in UCLA for many helpful technical discussions. Among them are Daching Chen, Tailai Tung, Chichong Chen, John Siwko, Kathy Sorabi, and George Kondlyis.

My appreciation also goes to Karen Tokashiki, Mike Mendez, and Jeff Warner for providing a flexible working hour so my Ph.D. research is possible. I am grateful to TRW for providing financial support for my studies.

Most importantly, I would like to thank my parents from the bottom of my heart. Without their immeasurable love and support, I would not be able to have what I have today.

VITA

July 18, 1971	Born, ShaoGuan, GuangDong Province, China
1992	B.S. Electrical Engineering summa cum laude University of California, Los Angeles Los Angeles, CA
1992-1993	Member of Technical Staff Vitesse Semiconductor Corporation Camarillo, CA
1994	M.S. Electrical Engineering University of California, Los Angeles Los Angeles, CA
1994-1997	Member of Technical Staff TRW, Inc. Redondo Beach, CA
1997-	Senior Member of Technical Staff TRW, Inc. Redondo Beach, CA

PUBLICATIONS

T. Yu, D. Chen, G.J. Pottie, K. Yao, "Blind Decorrelation and Deconvolution Algorithm for Multiple-Input Multiple-Output System I: theorem derivation" Proceedings of SPIE, vol. 3807

D. Chen, T. Yu, G.J. Pottie, K. Yao, "Blind Decorrelation and Deconvolution Algorithm for Multiple-Input Multiple-Output System II: simulation results and performance analysis" Proceedings of SPIE, vol. 3807

ABSTRACT OF THE DISSERTATION

Target Identification Processor For Wireless Sensor Network

by

Tommy Yu

Doctor of Philosophy in Electrical Engineering

University of California, Los Angeles, 1999

Professor Gregory J. Pottie, Chair

Wireless sensor networks have many applications in the commercial world as well as in the military. In the area of defense applications, distributed wireless sensor networks will provide new information systems for both military battlefield situational awareness and national security. In particular, the wireless sensor network can be used for target and threat identification in the battlefield environment. The objective of this thesis is to develop an architecture for a target recognition processor that can be implemented readily using on-node microprocessors of the sensor network. The architecture of the target recognition processor consists of

three sub-processors: 1) source separation and deconvolution processor, 2) feature extraction processor, and 3) pattern recognition processor.

For the source separation and deconvolution processor, we consider a subspace based algorithm for blind source separation and deconvolution. A frequency domain parametric representation algorithm is chosen for the feature extraction processor. Finally, we analyze the performance of a Hidden Markov Model (HMM) based pattern recognition processor. Each subprocessor is considered and optimized in this research. In this thesis, we present a modular architecture for the target recognition processor to be implemented on a wireless sensor network. The modular structure simplifies the design optimization process at each stage versus the all-in-one approach. The proposed architecture further permits a set of choices of algorithms at deeper levels. With signal processing capabilities added to sensor networks, we are able to turn traditional information gathering sensor networks into smart networks that provide knowledge and intelligence for the environment. These smart sensor networks will greatly simplify the knowledge and intelligence gathering process for the decision making. As a result, a new generation of efficient systems is made possible with the deployment of these sensor systems.

Chapter 1

Introduction

Sensors have been widely used in the commercial and military worlds. There are many types of sensor for different applications. Seismic sensors are employed to sense vibrations propagated through the ground. Acoustic sensors are used to measure the pressure difference in the air. Infrared sensors are deployed to sense the thermal energy difference between an object and the background environment. There are also magnetic sensors for sensing the magnetic field difference induced by the object of interest. Sensors have been traditionally used for trigger applications. That is, these sensors are used to sense any unusual event and send out a warning if these events happen. Seismic intrusion detectors have been used successfully in the Vietnam war [55]. These seismic sensors were deployed around the perimeter of the army camp as part of the intrusion warning system. The deployment of these seismic sensors gave soldiers more response time in the event

of an intrusion. Unfortunately, these sensors have some dramatic shortcomings. The most serious limitations involve their inability to discriminate among target classes, *i.e.*, wheeled vehicles, tracked vehicles, personnel, fixed-wing aircraft, *etc.* These sensors indicate only activation of the sensor; therefore, an operator must interpret the class of target from the number and frequency of the activation. Furthermore, these sensors are bulky in general and they require wireline connection. The deployment of these sensors can be time-consuming which limits the size of the sensor network.

1.1 Overview of Sensors and Sensor Networks

The introduction of microelectromechanical systems (MEMS) makes low power microsensors possible. Using the existing silicon fabrication technology, we can build vibration, acoustic, and infrared sensors on a single CMOS circuit die. In the UCLA MEMS research center, new microstructures are developed for these low power sensors. Sensor element sensitivity for vibration is 1 x $10^{-6} g/(Hz)^{1/2}$ or less in the 1 - 1000 Hz bandwidth–a sensitivity equal to the local background vibration noise in the most quiet structures or field environments. Acoustic sensor

sensitivity achieved is below 35 dB(A) sound pressure level (SPL) with a bandwidth of 100 - 10,000 Hz, giving sensitivities at the most quiet environmental-background noise levels. Sensitivities of infrared sensor is only limited by fundamental sensor thermal noise to a value of 5 x $10^{-10}W/(Hz)^{1/2}$ with responsivity through the thermal infrared wavelength range. On the other hand, the rapid developments in integrated-circuit (IC) technology, starting with medium-scale integration (MSI) and progressing to large-scale integration (LSI), and now, very-large-scale integration (VLSI) of electronic circuits has spurred the development of powerful, smaller, faster, and cheaper digital computers and special-purpose digital hardware. These inexpensive and relatively fast digital circuits have made it possible to construct highly sophisticated digital systems capable of performing complex digital signal processing functions and tasks. Furthermore, the advances in wireless transceiver technology bring us the single chip CMOS transceiver optimized for efficient, low duty cycle, micropower operation. Since all these building blocks can be fabricated readily on a single silicon die, we can potentially construct a new generation of low power micro-sensor nodes.

The modern micro-sensor node shown in Figure 1.1 consists of sensors (acoustic, seismic, and magnetic) for measurement, low power microprocessor for on-node processing, and a RF module for wireless communication. These micro-sensors enable fundamental changes in applications for the home, office, clinic,

factory, vehicle, metropolitan area, and the global environment. Low-cost micro-sensors will monitor health and safety, assist in control of transportation systems, and introduce a new manufacturing information technology. In the area of defense applications, distributed micro-sensor systems will provide a tremendous amount of new information that can be used for both military battlefield situational awareness and national security. In the application of battlefield awareness, wide area distribution of micro-sensors provides seismic, acoustic, magnetic, and imaging tactical information. The on-node microprocessor detects and identifies threats and intrusions based on these sensor measurements. Information on threats and intrusions is then reported back to the command station through radio links.

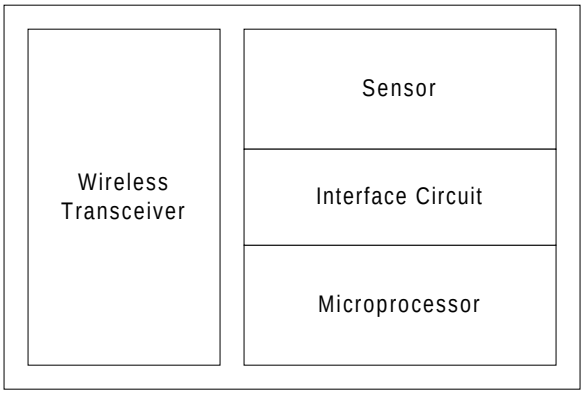

Figure 1.1: Microsensor Node

1.2 Overview of Dissertation Topics

The research objective of this thesis is to develop a target (threat and intrusion) identification processor that can be implemented readily using a low power on-node microprocessor. Our key contributions to this research include developing a modular architecture for target recognition as shown in Figure 1.2. There are several advantages of this new architecture. First of all, this architecture can be applied to the most general sensor network configuration since there is no specified requirement for sensor arrangement. Secondly, the modular architecture facilitates the design process for each sub-block. It also permits the implementation of a set of different algorithms using the same architecture. Furthermore, this architecture is made flexible for future technology upgrade insertion. The target identification processor proposed consists of three sub-processors: 1) source separation and deconvolution processor, 2) feature extraction processor, and 3) pattern recognition processor. Each subprocessor can be considered and optimized separately. Our key contributions in subprocessor design include presenting a new algorithm for source separation and deconvolution for the multiple-input multiple-output system, validating linear predictive coding (LPC) spectral analysis for the feature extraction processor, and applying hidden Markov Model (HMM) for the pattern recognition processor.

Chapter 2 presents background material on detection and system modeling. The source separation and deconvolution processor is presented in Chapter 3. We

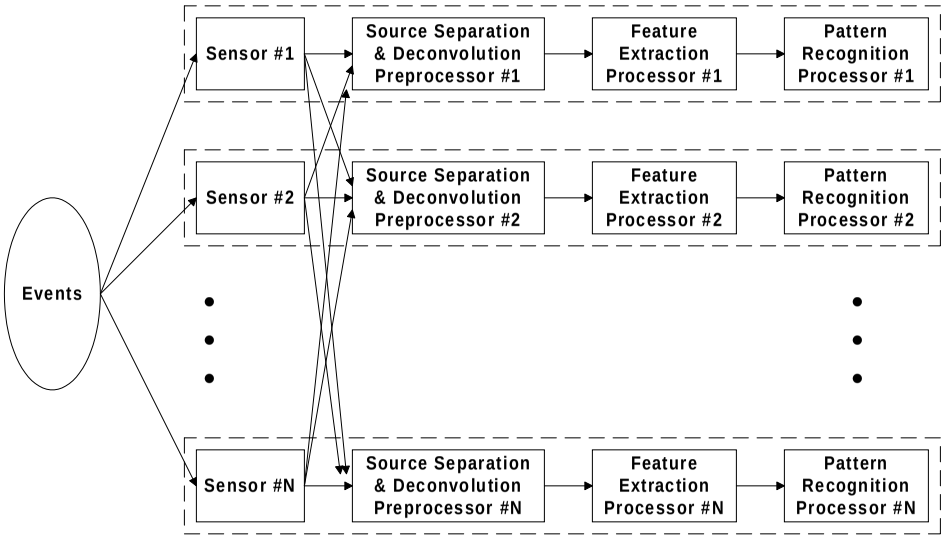

Figure 1.2: Target Recognition Processor Block Diagram

first address the modeling of our sensor measurement data. In our system, the sensor measurement data is modeled as the output of a multiple input multiple output (MIMO) system. The targets of interest and their associated activities are modeled as inputs to this MIMO system. In the real life deployment of the wireless sensor network, the environment being monitored can consist of many targets or sources. An example for battlefield reconnaissance is a vehicle convoy consisting of tanks, armored vehicles, and trucks. In order to facilitate the subsequent target identification processing and improve identification accuracy, we need to recover the input sources to the MIMO system in our model. In order to do that, we need to perform two operations. First, we need to perform source separation to separate signals from different sources. Second, the sensor measurement of micro-sensors can be distorted by the changing environment as well

as the propagation media of the measured signals. As a result, deconvolution is necessary to improve the signal to noise ratio (SNR) of our sensor measurement. The function of the source separation and deconvolution processor is to perform source separation and deconvolution. In this research, a new algorithm based on subspace decomposition and source decorrelation is presented for the source separation and deconvolution processor. The theoretical derivation and simulations of the algorithm are presented in Chapter 3. An adaptive version of this algorithm is also presented for practical implementation.

After signals from different sources are separated, identification processing can be performed on these signals. However, it is too computationally intense to perform identification on the raw measurements directly. The concept of the feature extraction processor is introduced in Chapter 4. The key idea of the feature extraction processor is to reduce the computational complexity. The function of the feature extraction processor is to convert raw measurement data into signature vectors. In this process, it is capable of compressing raw measurement data by several orders of magnitude with minimal or no distortion to the intrinsic character of the measurement data. The linear predictive coding (LPC) algorithm is chosen for the feature extraction processor in this research. A detailed outline of the LPC algorithm is presented in Chapter 4.

Signature vectors generated by the feature extraction processor are fed into the pattern recognition processor for the final identification. The architecture of our

pattern recognition processor is addressed in Chapter 5. The pattern recognition processor is a Bayesian classifier based on the concept of a signal model. Given a set of signal models $\{\lambda_l\}$ and a sequence of signature vectors $\{O\}$, we can evaluate the so-called a posteriori probabilities $P(O|\lambda_i)$ for each signal model $\{\lambda_i\}$. Under Bayes' criterion, the sequence $\{O\}$ should be associated with the signal model $\{\lambda_i\}$ with the highest probability measurement $P(O|\lambda_i)$. The signal model chosen in this research is the hidden Markov model (HMM). The concept of HMM will be addressed in Chapter 5. Finally, simulation results using real life measurement data are presented to demonstrate the performance of the target recognition processor.

Lastly, Chapter 6 presents concluding remarks on our research and addresses the direction for future research.

Chapter 2

Background

This chapter introduces background material to lay a technical foundation for the later chapters. It begin with an overview of detection and classification. Next is a review of system modeling and deconvolution. In particular, we consider a seismic system as an example. At the end, we summarize the list of problems to be considered in our research.

2.1 Detection and Classification

In the sensor network, there are many possible signal waveforms generated by the target of interest. In the transmission medium between the target and the sensor, *i.e.* ground for seismic waves, the signal waveforms may undergo some distortion. In many cases this distortion is caused by process which, because of

its complexity or randomness, may not be known precisely. The random nature of the signal and the noise accompanying such a signal do not allow us to determine with arbitrary precision the values of these parameters. As a result, statistical methods are used to guide the decision and estimation processes.

2.1.1 Hypothesis Testing

One of the most important statistical tools for making decisions is hypothesis testing. The hypotheses are statements of the possible decisions that are being considered. For example, in the sensor network problem we might select two hypotheses, a target is present or no target is present. Corresponding to each hypothesis there is a probabilistic description of the possible outcomes. Coupling this probabilistic description which a criterion or measure of goodness that the decision will satisfy on the average dictates a dichotomy (for two hypotheses) of the sample space over which the outcome of the experiment is defined. This dichotomy represents the best (optimum) decision rule subject to the criterion of goodness.

Denote the null hypothesis (H_0) the event that no target is present, and the alternative hypothesis (H_1) the event that a target is present. A criterion for making the decision must be selected. One reasonable criterion is to choose that hypothesis which is most likely to have occurred based on our observation. That is, given an observation y, which hypothesis is most probably true? The two

conditional probabilities are denoted

$$P(H_0|y) \quad \text{and} \quad P(H_1|y) \tag{2.1}$$

These are the probability that H_0 is true given observation y; and the probability that H_1 is true given observation y. These probabilities are called *a posteriori probabilities*, so this is called the *maximum a posteriori probability criterion.* The decision rule is to choose H_0 if

$$P(H_0|y) > P(H_1|y) \quad \text{or} \quad \frac{P(H_0|y)}{P(H_1|y)} > 1 \tag{2.2}$$

and choose H_1 otherwise.

In the hypothesis testing, there are two types of errors which can be made. If we say a signal is present when in fact it is not, an *error of the first kind* is made. That is, we choose H_1 given that H_0 is true. Denote the probability of this as $P(D_1|H_0)$. On the other hand, if H_0 is chosen when in fact H_1 is true, an *error of the second kind* is made. Denote the probability of this as $P(D_0|H_1)$. Finally, the average probability of error, denoted P_e, is

$$P_e = P(D_1|H_0)P(H_0) + P(D_0|H_1)[1 - P(H_0)] \tag{2.3}$$

2.1.2 Bayes Criterion

In the *maximum a posteriori probability criterion* stated in Section 2.1, there is no particular weighting given to the two types of errors. However, in many cases the consequences of each type of error are not equally important. To reflect these differences, costs may be assigned to each type of error.

Define C_{ij} as the cost associated with choosing hypothesis H_i when actually hypothesis H_j is true. The average cost for the two hypothesis example stated in Section 2.1 is given by

$$\begin{aligned} \bar{C} &= P(H_0)[P(D_0|H_0)C_{00} + P(D_1|H_0)C_{10}] \\ &\quad +[1 - P(H_0)][P(D_0|H_1)C_{01} + P(D_1|H_1)C_{11}] \end{aligned} \tag{2.4}$$

The criterion that minimizes the average cost is called *Bayes criterion*. Using this criterion, a decision rule is derived. In term of the hypothesis H_1, choose H_1 if

$$\frac{P(y|H_1)}{P(y|H_0} \leq \frac{P(H_0)(C_{10} - C_{00})}{[1 - P(H_0)](C_{01} - C_{11})} \tag{2.5}$$

By comparing Equations 2.5 and 2.2 for the case when $C_{10} - C_{00} = C_{01} - C_{11}$, it follows that the maximum a posteriori probability criterion is a special case of the Bayes criterion.

In all the previous cases only binary type decisions were considered. In our

problems, there are more than two possible decisions. We shall proceed as for the Bayes criterion and assume that cost functions and a priori probabilities are known. Assume that one of n hypotheses is to be chosen and assign a cost, C_{ij}, to each decision. This is the cost of choosing hypothesis H_i when hypothesis H_j is true. The average cost is then

$$\bar{C} = \sum_{i=1}^{n}\sum_{j=1}^{n} C_{ij} P(D_i|H_j) P(H_j) \tag{2.6}$$

where $P(D_i|H_j)$ denotes the probability of choosing hypothesis H_i given that H_j is true. The Bayes criterion is to minimize the average cost. This implies that for any received set of samples, the hypothesis H_j which yields the lowest average cost is chosen. Given the set of samples $\mathbf{y}$, the cost associated with hypothesis H_j is

$$C_j = \sum_{i=1}^{n} C_{ji} P(H_i|\mathbf{y}) \tag{2.7}$$

where $P(H_i|\mathbf{y})$ is the probability that hypothesis H_i is true given $\mathbf{y}$. This probability can be expressed

$$P(H_i|\mathbf{y}) = \frac{p_i(\mathbf{y}) P(H_i)}{p(\mathbf{y})} \tag{2.8}$$

where $p_i(\mathbf{y})$ is the probability density function for hypothesis H_i, and $p(\mathbf{y})$ is the marginal density function for $\mathbf{y}$. The cost associated with choosing H_j is then

$$C_j = \frac{\sum_{i=1}^{n} C_{ji} p_i(\mathbf{y}) P(H_i)}{p(\mathbf{y})} \tag{2.9}$$

Since $p(\mathbf{y})$ does not depend on the hypothesis, the decision rule is to choose that hypothesis H_j for which

$$\lambda_j = \sum_{i=1}^{n} C_{ji} p_i(\mathbf{y}) P(H_i) \tag{2.10}$$

is a minimum. This is the Bayes test for multiple hypotheses.

Of particular interest is the case where $C_{ij} = 1$ for $i \neq j$, and $C_{ii} = 0$. Thus, there is no penalty for correct decisions, and all errors have equal weighting.

To use the Bayes rule compute

$$\lambda_j = \sum_{i=1, i\neq j}^{n} P(H_i) p_i(\mathbf{y}) \quad \text{for all} \quad j \tag{2.11}$$

and choose that hypothesis corresponding to the index j for which this is a minimum. Consequently

$$\lambda_k - \lambda_j = P(H_j) p_j(\mathbf{y}) - P(H_k) p_k(\mathbf{y}) \tag{2.12}$$

Without loss of generality, assume that λ_j is the minimum term. Then $\lambda_k - \lambda_j > 0$, or

$$P(H_j) p_j(\mathbf{y}) > P(H_k) p_k(\mathbf{y}) \tag{2.13}$$

Therefore choosing the minimum λ_j corresponds to choosing the largest value of $P(H_j)p_j(\mathbf{y})$. The decision rule becomes: choose the hypothesis H_j for which $p_j(\mathbf{y})P(H_j)$ is a maximum. There is one more practical difficulty here. In general, the distribution $p_j(\mathbf{y})$ is unknown. As a result, the training is required.

2.2 System Modeling and Deconvolution

In Section 2.1, we outline the criterion for signal detection in the presence of distortion and noise. In this section, we address the issue of the system model for the propagation channel of our signals. In particular, we consider seismic waves.

The vertical component of a Rayleigh wave is commonly used as the energy vehicle to activate modern, seismic, unattended ground sensors (UGS). The amount of energy generated by a target of interest is site dependent, and the effect of terrain on Rayleigh wave generation and propagation is known to be substantial. The seismic wave is generated by the stress signal applied to the ground. This stress signal is intrinsic to the target of interest. However, the seismic wave measured at the sensor can be quite different from the original stress signal. The distortion is a joint effect of the geometric attenuation and reflection loss. In seismology, the usual procedure is to assume a layered earth model, and the requirement is to use the received signal to estimate the sequence of reflection coefficients corresponding to the various layers of the model. The received signal

is itself made up of echoes produced at the different layers of the model in response to the excitation which is ordinarily in the form of a short-duration pulse. The equally spaced time sequence of reflection coefficients may be viewed as the impulse response of the layered earth model. Using this layer model, the sensor measurement can be modeled as the output of a finite impulse response (FIR) system driven by the stress signal. In system modeling, the sensor measurement is defined as the *convolution* of the stress signal and the impulse response of the layer system.

Deconvolution refers to the problem of determining the impulse response of the system or the input signal. The output of the system is typically accessible, which means that the output signal of the system is available for processing. However, in our application, the system is unknown and its input is inaccessible. In other word, a precise knowledge of the input signal is not available. In this case, deconvolution is referred to as *blind deconvolution*. The deconvolution problem is further complicated by the fact there can be more than one input source. In this case, the problem of determining the input signals is called *blind source separation and deconvolution*. As we shall see in Chapter 3, a new algorithm is developed to solve this type of problem. While we only discuss the system model for seismic wave generation and propagation here, similar system models can be developed for other types of sensor measurements, such as acoustic waves.

2.3 Summary

The key objective of our research is to perform detection and classification using an unattended sensor network. In this chapter, we reviewed hypothesis testing for detection and the Bayes criterion. Prior to the hypothesis testing, blind deconvolution is required to recover the source signal. In the multiple sources case, blind source separation and deconvolution is required. This problem is solved in Chapter 3. As we have shown, the hypothesis testing is based on a conditional probability measure. In order to generate this probability measure, we need to have a signal model for our source signal. Since our sources are not deterministic signals, a stochastic signal model (HMM) is proposed in Chapter 5. The calculation of the probability measure and the training of the signal model are also solved.

Chapter 3

Source Separation and Deconvolution Preprocessor

In the remote sensing application, the unknown or changing environment should be considered when examining the sensor measurement. For example, in the case of seismic sensor measurements, the measurement is a function of ground conditions as well as the distance between sensor and source. In the model shown in Figure 3.1 [7] [11], the seismic measurement is modeled as the output of an unknown system driven by some unknown sources. The unknown system characterizes the propagation media of the seismic wave launched by the target, in this case, the ground conditions. The interaction between the target and the ground is characterized by the unknown input source. With this model, we have a blind identification and deconvolution problem. We want to identify the

channel (which characterizes both the ground and the distance between sensor and source) and recover the input source (which characterizes target motion). The channel response and input source will be used by subsequent processors as signatures for target identification.

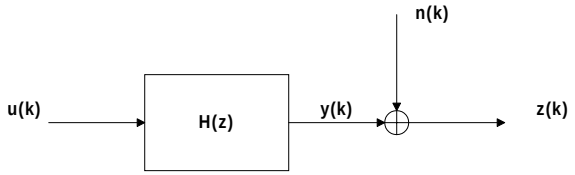

Figure 3.1: System Model for Seismic Wave Generation

Mendel investigated the blind deconvolution problem in [11] [7]. In this model, the input is assumed to be a Bernoulli-Gaussian process and the unknown channel response can be any auto-regressive moving-average (ARMA) model. With this model, Mendel developed an iterative algorithm for estimating the channel response and input source. While this algorithm works quite well for applications in oil exploration, there are two drawbacks for this algorithm in our application. First, the computational intensity of this algorithm is very large. It takes many iterations for the algorithm to converge. Furthermore, this algorithm is an expectation-maximization (EM) algorithm. The convergence is critically dependent on the initial start point. Second, the Bernoulli-Gaussian assumption for this input is too limited for our application.

Shalvi and Weinstein proposed a blind deconvolution algorithm based on higher order statistics in [31]. In this model, the channel response can be any

linear system. However, the system input has to be white and non-Gaussian. This algorithm estimates the deconvolution filter and recovers the input with an unknown delay. However, it does not estimate the channel response. While this new algorithm has looser constraints than the previous one proposed by Mendel, it has some problems as well: 1) it does not estimate the unknown channel response even though this channel response is very useful for tracking applications, 2) the unknown delay between the recovered input and the actual input is a function of the channel response, and it is very difficult to synchronize data between different sensors due to this unknown delay, and 3) the assumption that the input is white can be unrealistic. Furthermore, since this algorithm is based on higher order statistics, it requires longer data intervals for accurate estimation.

Moulines proposes another blind deconvolution algorithm in [23]. This algorithm is based on second order statistics. In this model, the system is a single input multiple output finite impulse response (FIR) linear system. Multiple outputs can be obtained through either using multiple sensors for measurement or sampling the output at rate higher than the input symbol rate. Since we have a multiple sensor network and the observation bandwidth of the source is less than the sampling rate, this assumption is valid here. The unknown input need not be white. The input is assumed to be persistently excited [12] but otherwise unknown. The persistently excited requirement is a very loose one and most broadband sources have this property. Moulines' algorithm estimates the system

transfer function by exploring the subspace structure of the received data. The deconvolution filter can be calculated once the channel response is determined. There are many advantages to Moulines' algorithm: 1) the input constraint is quite loose, which can be satisfied by most broadband sources, 2) it estimates the channel response and the delay between recovered input and actual input is well defined, and 3) the algorithm is based on second order statistics only.

In cases containing a single source, Moulines' algorithm is sufficient for doing data preprocessing. However, we rarely have a single source in a real life application. Multiple sources scenarios are generally considered in decorrelation and beamforming problems. In the beamforming problem, the channel is assumed to be non-dispersive. Since in a remote sensing application the channel will be dispersive, the traditional beamforming algorithm can not be applied to our problem directly.

A variety of algorithms have been proposed for solving the multiple sources separation problem. Weinstein *et al.* [27] investigated a 2 x 2 linear time invariant system. Two unknown inputs are assumed to be statistically uncorrelated. The algorithm further assumes the diagonal element of the transition matrix to be 1. That is, one signal can be viewed as interference for the other. Weinstein developed an iterative algorithm for estimating the unknown channel coefficients based on cross-correlation. However, the convergence of the algorithm is not guaranteed if both channel coefficients are unknown, and it depends on the initial

starting point of the algorithm. Furthermore, the algorithm only works for 2 x 2 systems and the generalization of this algorithm to more sources can be very difficult.

Yellin [28] [29] developed a decorrelation algorithm for multiple sources separation based on higher order statistics. However, this algorithm suffers the same drawbacks as Weinstein's method. Furthermore, since the algorithm is based on higher order statistics, the input has to be non-Gaussian.

Moulines' algorithm [23] for single input multiple outputs (SIMO) system offered simple and elegant solutions for system identification problem under quite general assumptions. Several papers [50] [51] [52] have tried to extend Moulines' algorithm from SIMO system to multiple inputs multiple outputs (MIMO). Gorokhov [50] stated that certain multivariate convolutive mixtures may be converted to instantaneous (static) mixtures of input signals. This static mixture can be further separated either using the structure of digital communications signals or various higher order statistics (HOS) identification techniques.

We propose a new blind separation and deconvolution algorithm for a MIMO system. This new algorithm is also a generalization of Moulines' algorithm from a SIMO system to a MIMO system similar to algorithms stated in [50] [51] [52]. Using the subspace decomposition method, the multivariate convolutive mixtures problem is converted to an instantaneous mixtures problem. These instantaneous mixtures can be separated using the decorrelation algorithm. Our algorithm is

different from previous algorithms [50] [51] [52] in two areas. First, the algorithm uses second order statistics only for both deconvolution and source separation. Some previous algorithms use higher order statistics for source separation. Second, the algorithm makes no assumption on the structure of the input signal. In digital communications, the data structure of the input signal is well defined. These data structures can be used for source separation. However, input signals in our application are not digital communications signals and their data structures are unknown. The assumptions we make for these input signals are that they are uncorrelated and colored. These assumptions are quite general for sources of interest in our application such as seismic signatures of tanks.

3.1 Source Separation and Deconvolution Algorithm

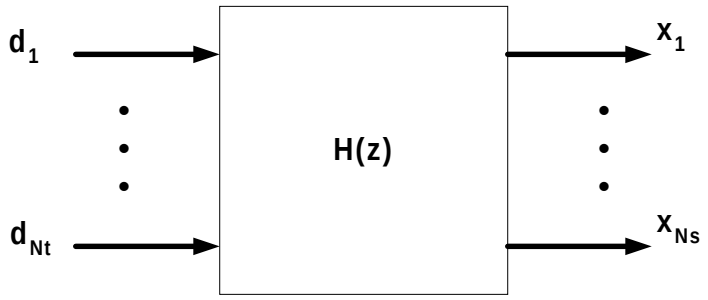

Figure 3.2: Multiple Input Multiple Output System

Consider a sensor network of N_s sensors and N_t sources as shown in Figure

3.2. Let $d_j(n)$ denote the signal from source j, $j = 1, \ldots, N_t$. The received signal at sensor i is given by:

$$x_i(n) = \sum_{m=-\infty}^{\infty} \sum_{j=1}^{N_t} d_j(m) h_{ij}(n-m) + b_i(n) \tag{3.1}$$

where $b_i(n)$ is the measurement noise at sensor i and it is assumed to be independent of the source signal. $h_{ij}(i = 1, \ldots, N_s; j = 1, \ldots, N_t)$ is the transfer function from source j to sensor i. In the multi-sensor network, h_{ij} characterizes the propagation medium from target j to sensor i.

Assume each channel can be modeled by a FIR filter of order M. Stacking N successive samples of the received signal sequence, *i.e.*, $X_i = [x_i(n), \ldots, x_i(n - N + 1)]^T$ (dim. N x 1), we obtain the following:

$$X_i(n) = H_i D + B_i \tag{3.2}$$

where $B_i = [b_i(n), \ldots, b_i(n - N + 1)]^T$ (dim. N x 1), and $D = [D_1^T, \ldots, D_{N_t}^T]^T$ (dim. $N_t(M + N)$ x 1). D_i is a vector of $(M + N)$ successive samples of the ith source signal sequence, *i.e.*, $D_i = [d_i(n), \ldots, d_i(n - N - M + 1)]^T$. H_i is the N x $N_t(M + N)$ filtering matrix from multiple sources to sensor i, *i.e.*, $H_i = [H_{i1}, \ldots, H_{iN_t}]$. H_{ij} is a N x $(M + N)$ matrix defined as:

$$H_{ij} \stackrel{\text{def}}{=} \begin{pmatrix} h_{ij}^{(0)} & \cdots & h_{ij}^{(M)} & 0 & \cdots & \cdots & 0 \\ 0 & h_{ij}^{(0)} & \cdots & h_{ij}^{(M)} & 0 & \cdots & 0 \\ \vdots & & & & & & \vdots \\ 0 & \cdots & \cdots & 0 & h_{ij}^{(0)} & \cdots & h_{ij}^{(M)} \end{pmatrix} \tag{3.3}$$

where $h_{ij}^{(k)}$ denotes the kth tap of h_{ij}.

Combine all sensor measurements into one vector $X = [X_1^T, \ldots, X_{N_s}^T]^T$ (dim. N_sN x 1) to obtain the following:

$$X = HD + B \tag{3.4}$$

where $B = [B_1^T, \ldots, B_{N_s}^T]^T$ (dim. N_sN x 1) and $H = [H_1^T, \ldots, H_{N_s}^T]^T$ (dim. N_sN x $N_t(M+N)$).

As in the method of Moulines [23] [26], the identification is based on the N_sN x N_sN auto-correlation matrix R_x for the measurement vector X:

$$R_x = E\{XX^T\} \tag{3.5}$$

where $E\{\cdot\}$ denotes mathematical expectation. Since the additive measurement noise is assumed to be independent of the transmitted sequence, R_x is expressed as:

$$R_x = HR_dH^T + R_b \tag{3.6}$$

where $R_d = E\{DD^T\}$ and $R_b = E\{BB^T\}$ respectively denote the auto-correlation matrices of the transmitted symbol vector D and of the measurement noise vector B. Assume all input sources are statistically uncorrelated, that is:

$$E\{d_i(t)d_j^*(t-\tau)\} = 0 \qquad \forall\tau, i \neq j \tag{3.7}$$

Then the source correlation matrix R_d (dim. $N_t(M+N)$ x $N_t(M+N)$) has the following structure:

$$R_d = \begin{pmatrix} R_d^1 & 0 & \cdots & \cdots & 0 \\ 0 & R_d^2 & 0 & \cdots & 0 \\ \vdots & & & & \vdots \\ \vdots & & & & 0 \\ 0 & \cdots & \cdots & 0 & R_d^{N_t} \end{pmatrix} \tag{3.8}$$

where R_d^i denotes the correlation matrix of source i and is assumed to be full-rank but otherwise unknown. The rank of R_d is $N_t(M+N)$.

Note that matrix H is full column rank, *i.e.*, $\text{rank}(H) = N_t(M+N)$. The requirement for this condition is:

1) the polynomials $H^{(ij)}(z)$ have no common zero, where $H^{(ij)}(z)$ is defined as:

$$H^{(ij)}(z) \stackrel{\text{def}}{=} \sum_{k=0}^{M} h_{ij}^{(k)} z^k \tag{3.9}$$

2) N is greater than or equal to the maximum degree M of the polynomials

$H^{(ij)}(z)$, *i.e.*, $N \geq M$

3) $N_s N \geq N_t(M+N)$

4) at least one polynomial $H^{(ij)}(z)$ has degree M for each j

In order to preserve the subspace structure of R_d, the dimension of R_x should be chosen to be greater than R_d, *i.e.*, $N_s N \geq N_t(M+N)$. Let $\lambda_0 \geq \lambda_1 \geq \cdots \lambda_{N_t(M+N)}$ denote eigenvalues of R_x. Without loss of generality, assume the noise distributions at all sensors are the same with power σ^2. Since R_d is full rank, the signal part of the correlation matrix R_x, *i.e.*, HR_dH^T has rank $N_t(M+N)$, hence:

$$\begin{aligned} \lambda_i > \sigma^2 \qquad & i = 0, \ldots, N_t(M+N)-1 \\ \lambda_i = \sigma^2 \qquad & i = N_t(M+N), \ldots, N_s N - 1 \end{aligned} \tag{3.10}$$

Denote the unit-norm eigenvectors associated with the eigenvalues $\lambda_0, \ldots, \lambda_{N_t(M+N)-1}$ by $S_0, \ldots, S_{N_t(M+N)-1}$; denote those corresponding to $\lambda_{N_t(M+N)}, \ldots, \lambda_{N_s N-1}$ by $G_0, \ldots, G_{N_s N - N_t(M+N)-1}$. Also define S (dim. $N_s N$ x $N_t(M+N)$) and G (dim. $N_s N$ x $(N_s N - N_t(M+N))$) as follows:

$$\begin{aligned} S &= \left[S_0, \ldots, S_{N_t(M+N)-1}\right] \\ G &= \left[G_0, \ldots, G_{N_s N - N_t(M+N)-1}\right] \end{aligned} \tag{3.11}$$

The correlation matrix R_x is thus also expressed as:

$$R_x = S\,diag(\lambda_0, \ldots, \lambda_{N_t(M+N)-1})S^T + \sigma^2 GG^T \tag{3.12}$$

The columns of matrix S span the so-called *signal subspace* (dim. $N_t(M+N)$), while the columns of G span its orthogonal complement, the *noise subspace*. Since H is a full column rank matrix, the *signal subspace* is also the linear space spanned by the columns of H. By orthogonality between the noise and the signal subspace, the columns of H are orthogonal to any vector in the noise subspace. That is,

$$G_i^T H = 0 \qquad\qquad 0 \leq i < N_s N - N_t(M+N) - 1 \tag{3.13}$$

Furthermore, let $H = [K_1, \ldots, K_{N_t}]$ where $K_i = [H_{1i}^T, \ldots, H_{N_s i}^T]^T$, the following is true as well:

$$\begin{aligned} G_i^T K_j = 0 \qquad\qquad & 0 \leq i < N_s N - N_t(M+N) - 1 \\ & 1 \leq j \leq N_t \end{aligned} \tag{3.14}$$

This relation is the basis for the new source separation and deconvolution algorithm. Note that the relation is linear in the unknown coefficients. This leads to the possibility of simple identification procedures, provided this equation actually characterizes the channel coefficients.

Let $G_i = [G_{i1}^T, \ldots, G_{iN_s}^T]^T$ where G_{ik} is N x 1 vector, and define $\mathcal{G}_i = [\mathcal{G}_{i1}^T, \ldots, \mathcal{G}_{iN_s}^T]^T$ (dim. $N_s(M+1)$ x $(M+N)$) where $\mathcal{G}_{ik}$ (dim. $(M+1)$ x $(M+N)$) has the

following structure:

$$\mathcal{G}_{ik} = \begin{pmatrix} G_{ik}^{(1)} & G_{ik}^{(2)} & \cdots & G_{ik}^{(N)} & 0 & \cdots & 0 \\ 0 & G_{ik}^{(1)} & G_{ik}^{(2)} & \cdots & G_{ik}^{(N)} & 0 & \cdots \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & G_{ik}^{(1)} & G_{ik}^{(2)} & \cdots & G_{ik}^{(N)} \end{pmatrix} \tag{3.15}$$

where G_{ik}^{l} denotes the lth coefficients of vector G_{ik}.

From Lemma 1 in [23], Equation 3.14 can be rewritten in the following format:

$$h_j^T \mathcal{G}_i = 0 \qquad \begin{aligned} 0 &\leq i < N_s N - N_t(M+N) - 1 \\ 1 &\leq j \leq N_t \end{aligned} \tag{3.16}$$

This relation can be used to estimate the channel coefficients even in the cases where R_d is unknown. Similarly to Moulines' algorithm, this method relies on the specific structure of H, the filtering matrix.

3.1.1 Algorithm Formulation

A blind identification procedure consists of estimating the $N_s N_t(M+1)$ x 1 vector $\mathbf{h}$ of channel coefficients, *i.e.*, $\mathbf{h} = [h_1^T, \ldots, h_{N_t}^T]^T$ and $h_j = [h_{1j}^{(0)}, \ldots, h_{1j}^{(M)}, \ldots, h_{N_s j}^{(0)}, \ldots, h_{N_s j}^{(M)}]^T$ (dim. $N_s(M+1)$ x 1), solely from the observations of X.

The orthogonality condition is linear in the channel coefficients. In practice, only sample estimates G_i of the noise eigenvectors are available and h_j is solved in the least square sense. Furthermore, it can be shown that $h_j (j = 1, \ldots, N_t)$ are linearly independent vectors (See Appendix A). The system identification problem can be restated as solving N_t linearly independent vectors, *i.e.*, $h_j (j = 1, \ldots, N_t)$ that minimize the following quadratic form

$$q(h_j) = h_j^T Q h_j \qquad \text{where} \qquad Q = \sum_{i=1}^{N_s N - N_t(M+N)} \mathcal{G}_i \mathcal{G}_i^T \tag{3.17}$$

Estimates of h_j can be obtained by minimizing $q(h_j)$ subject to a properly chosen constraint avoiding the trivial solution $h_j = 0$. In this new algorithm, the quadratic constraint is chosen. That is, minimize $q(h_j)$ subject to $\|h_j\| = 1$. If there is only one source (N_t=1), the solution is the unit-norm eigenvector associated with the smallest eigenvalue of matrix Q. This is the result Moulines presented in his paper [23]. However, in the case of multiple sources ($N_t > 1$), we claim the following:

Lemma 1:

The N_t unit-norm eigenvectors associated with the smallest N_t eigenvalues of matrix Q form the basis for $h_j, j = 1, \ldots, N_t$.

A proof is given in Appendix A. Denote N_t unit-norm eigenvectors associated with the smallest N_t eigenvalues of matrix Q by $f_i (i = 1, \ldots, N_t)$. By Lemma 1, solving for eigenvector f_i yields the following relation:

$$[h_1, \ldots, h_{N_t}] = [f_1, \ldots, f_{N_t}] * C \tag{3.18}$$

where C is a N_t x N_t matrix.

Let $f_i = [f_{i1}^T, \ldots, f_{iN_s}^T]^T$ (dim. $N_s(M+1)$ x 1) where f_{ij} is a $(M+1)$ x 1 vector, define $F = [F_1, \ldots, F_{N_t}]$ (dim. N_sN x $N_t(M+N)$) and $F_i = [F_{i1}^T, \ldots, F_{iN_s}^T]^T$ (dim. N_sN x $(M+N)$) where F_{ij} (dim. N x $(M+N)$) has the following structure:

$$F_{ij} = \begin{pmatrix} f_{ij}^{(0)} & \cdots & f_{ij}^{(M)} & 0 & \cdots & \cdots & 0 \\ 0 & f_{ij}^{(0)} & \cdots & f_{ij}^{(M)} & 0 & \cdots & 0 \\ \vdots & & & & & & \vdots \\ 0 & \vdots & \vdots & 0 & f_{ij}^{(0)} & \cdots & f_{ij}^{(M)} \end{pmatrix} \tag{3.19}$$

Let $y_j(n)(j = 1, \ldots, N_t)$ be the signal that satisfies the following:

$$y_j(n) = \sum_{k=1}^{N_t} c_{jk} d_k(n) \tag{3.20}$$

Let Y_i be $M+N$ successive samples from $y_i(n)$ and $Y = [Y_1^T, \ldots, Y_{N_t}^T]^T$. It can be shown that:

$$X = FY + B \tag{3.21}$$

After F is determined, a linear deconvolution filter E (dim. $N_t(M+N)$ x N_sN) can be found by:

$$E = (F^T F)^{-1} F^T \tag{3.22}$$

We can recover $y_j(n)(j = 1, \ldots, N_t)$ using this deconvolution filter E. However, the objective of the source separation and deconvolution algorithm is to recover $d_k(n)(k = 1, \ldots, N_t)$. From Equation 3.20, we observe $y_j(n)$ is a linear combination of $d_k(n)$. Using the property stated in Equation 3.7, recovering $d_k(n)$ from $y_j(n)$ becomes a problem of separating a mixture of independent signals. This problem can be solved readily using an algorithm proposed by Molgedey in [30]. The algorithm can be summarized as follows:

1. Once $y_j(n)(j = 1, \ldots, N_t)$ have been calculated, generate matrices M_0 (symmetric correlation matrix) and M_1 (time delayed correlation matrix).

$$\begin{aligned} M_0^{ij} &= E\{y_i(n) y_j(n)\} \\ M_1^{ij} &= E\{y_i(n) y_j(n+l)\} \end{aligned} \tag{3.23}$$

2. C can be solved as an eigenvalue problem. That is:

$$(M_0 M_1^{-1}) C = C(\Lambda_0 \Lambda_1^{-1}) \tag{3.24}$$

where Λ_0, Λ_1 are diagonal matrices.

Once C is determined, system transfer function coefficients $\mathbf{h} = [h_1^T, \ldots, h_{N_t}^T]^T$ can be solved using Equation 3.18. Let C^{-1} be the inverse of C. Input source

$D = [D_1^T, \ldots, D_{N_t}^T]^T$ can be solved as:

$$D_i = \sum_{j=1}^{N_t} C_{ij}^{-1} Y_j \tag{3.25}$$

In summary, this new blind source separation and deconvolution algorithm can be separated into two parts as shown in Figure 3.3:

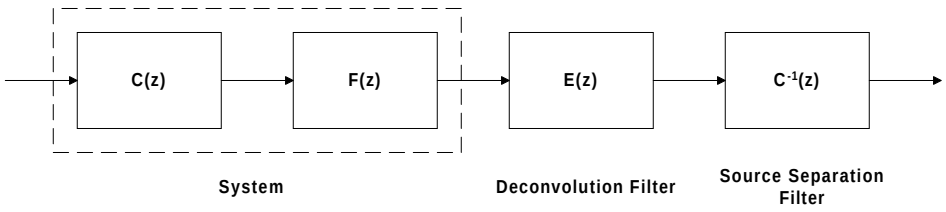

Figure 3.3: Blind Source Separation and Deconvolution Block Diagram

We now address the computational complexity of this new algorithm.

1. *Deconvolution*: In this operation, we estimate the channel dispersion of the system and equalize channel dispersion using deconvolution filter E. As a result, we reduce this MIMO system with FIR channels into a mixture system C. In this process, we perform one eigenvalue decomposition on R_x (dim. NN_s x NN_s), one eigenvalue decomposition on Q (dim. $N_s(M+1)$ x $N_s(M+1)$), and one matrix inversion on F (dim. N_sN x $N_t(M+N)$). From the conditions we state for N_s, N_t, M, and N, the computation is dominated by the eigenvalue decomposition on R_x. That is an $O((NN_s)^3)$ operation.

2. *Source Separation*: In this operation, we separate the mixture of independent sources using Molgedey's algorithm. In this process, we perform one matrix

inversion on M_1 (dim. N_t x N_t), one matrix inversion on C (dim. N_t x N_t), and one eigenvalue decomposition on $M_0 M_1^{-1}$ (dim. N_t x N_t). In this case, the computation is dominated by the eigenvalue decomposition which is an $O((N_t)^3)$ operation. In comparison to the deconvolution operation, the computation of the decorrelation operation is negligible.

We have shown the subspace-based parameter estimation scheme using the *noise subspace* of the sensor data. Similarly to Moulines's algorithm, our algorithm can be modified to use the signal subspace of the sensor data for parameter estimation.

The system identification based on the signal subspace can be stated as solving for N_t linearly independent vectors h_j that maximize the following quadratic form

$$\tilde{q}(h_j) = h_j^T \tilde{Q} h_j \qquad \text{where} \qquad \tilde{Q} = \sum_{i=1}^{N_t(M+N)} \mathcal{S}_i \mathcal{S}_i^T \tag{3.26}$$

where $\mathcal{S}_i$ is the matrix associated with eigenvector S_i and it is defined similarly to $\mathcal{G}_i$.

Under the unit-norm constraints, both the noise- and signal-related quadratic forms give identical solutions. The computational complexity comparison between the two depends on values of N_s, N_t, M, and N.

3.1.2 Simulation Results

We demonstrate the performance of the new algorithm through simulations. In practical situations, the ensemble average of the signal correlation matrix is unknown. It is replaced by its sample estimate. Two sets of simulations are performed to evaluate the performance of the algorithm. In the first set of simulations, we demonstrate the multiple sources separation and deconvolution capability of the algorithm. Two simulation cases are run and the following table summarizes the simulation setup:

Simulation Case	N_t	N_s	M	N	SNR(dB)	Data Size
1	2	5	4	6	40	3000
2	3	5	4	18	40	10000

N_t, number of inputs sources

N_s, number of sensors

M, maximum order of each channel

N, number of taps for the equalizer

The input sources are generated by passing white Gaussian sources through filter G_i defined as follows:

$$G_0 = [0.7620, 0.1270, 0.6350] \tag{3.27}$$

$$G_1 = [0.8018, -0.2673, 0.5345]$$

$$G_2 = [0.8165, 0.4082, 0.4082]$$

The unknown system is characterized by $H_{ij}(i = 1, \ldots, N_s; j = 1, \ldots, N_t)$ where each H_{ij} is a $(M+1)$ taps FIR filter. The coefficients of H_{ij} are produced by a Gaussian number generator. Each H_{ij} vector is normalized to have unit norm. White Gaussian noise is added to the output. The signal to noise ratio is defined as the power ratio between the received signal and the measurement noise at the output. 100 Monte-Carlo runs are conducted. We evaluate the performance of the algorithm by computing the average similarity between source inputs and recovered inputs and the average similarity between the unknown system transfer function and the estimated system transfer function. Similarity is defined as:

$$sim(a, b) = \frac{E\{ab\}}{\sqrt{E\{a^2\}E\{b^2\}}} \tag{3.28}$$

Plots of average similarity are shown in Figure 3.4, 3.5, 3.6, 3.7. From these plots, we can see the algorithm is capable of recovering multiple sources using multiple sensor measurements. The performance of the algorithm is very good since the average similarity is very close to 1.

We note that two conditions must be satisfied in choosing N. They are $N \geq M$ and $NN_s \geq N_t(M+N)$. In the event where N_s is much greater than N_t, N can be chosen to be close to M. That translates to a small size for R_x. The small dimension of R_x requires a small data block for the correlation matrix

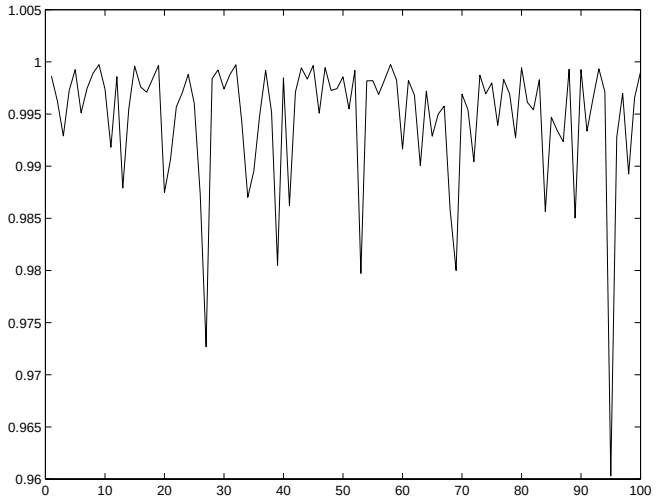

Figure 3.4: Similarity between transfer function vs. run (2 sources and 5 sensors)

mean similarity = 0.9947 and variance = 3.8154x10^{-5}

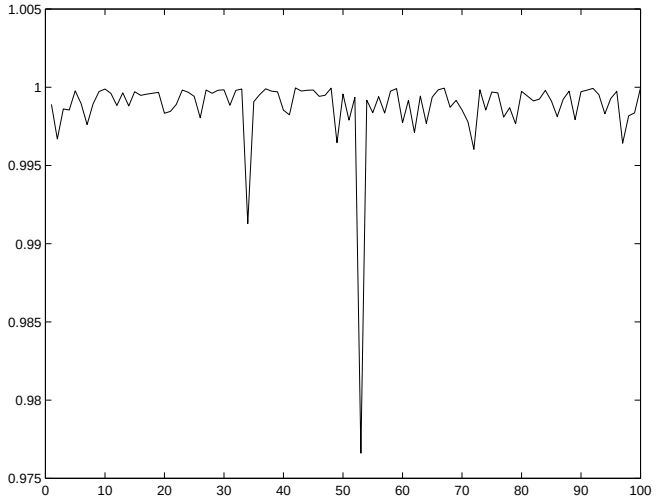

Figure 3.5: Similarity between data vs. run (2 sources and 5 sensors)

mean similarity = 0.9988 and variance = 6.3782x10^{-6}

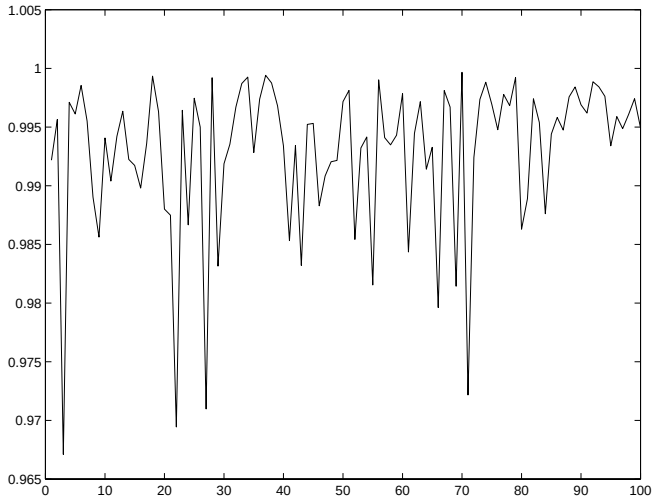

Figure 3.6: Similarity between transfer function vs. run (3 sources and 5 sensors)

mean similarity = 0.9929 and variance = 4.3428x10^{-5}

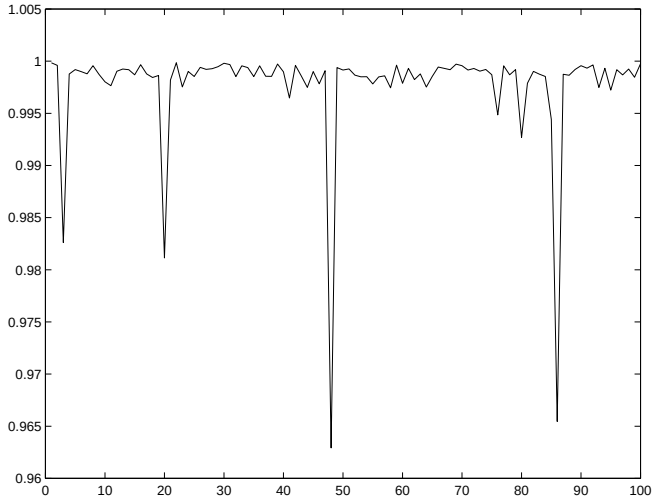

Figure 3.7: Similarity between data vs. run (3 sources and 5 sensors)

mean similarity = 0.9977 and variance = 2.9954x10^{-5}

estimation. On the other hand, if N_s is close to N_t, N can be quite large compared to M. That means a larger size of R_x which translates to a large data block for correlation matrix estimation, and thus much higher computational complexity.

In the second set of simulations, we investigate the performance of the algorithm versus the size of the measurement data. Two simulation cases are run and the following table summarizes the simulation setup:

Simulation Case	N_t	N_s	M	N	SNR(dB)
1	2	5	4	6	40
2	3	5	4	18	40

The input sources are generated by passing white Gaussian sources through filter G_i defined in Equation 3.27. The number of symbols used varies from 1000 to 10000 in increments of 1000. 100 Monte-Carlo runs are conducted for each symbol size and the average similarity is calculated. The coefficients of H_{ij} are produced by a Gaussian number generator and the coefficients are fixed throughout all simulation runs. The coefficients of the first simulation case are:

H11 = [-0.2068 -0.7964 0.0599 0.1376 -0.5482]

H21 = [0.6909 0.6899 -0.0218 0.1899 0.1013]

H31 = [-0.0783 0.3042 -0.2466 0.9150 -0.0572]

H41 = [0.0836 0.7830 0.0435 -0.0702 -0.6109]

H51 = [0.1256 -0.5699 0.3047 0.6925 -0.2951]

H12 = [0.3186 0.4657 -0.5918 -0.5351 0.2121]

H22 = [-0.2146 0.3703 0.4377 0.3820 0.6924]

H32 = [0.3661 0.6520 -0.6584 -0.0108 -0.0858]

H42 = [-0.6338 0.1017 -0.4174 0.5591 -0.3181]

H52 = [0.2178 0.0903 -0.3798 -0.8942 -0.0244]

The coefficients of the second simulation case are:

H11 = [-0.4579 0.2784 0.2300 0.7669 0.2679]

H21 = [-0.5120 0.3026 -0.8028 -0.0155 -0.0383]

H31 = [0.6136 0.5008 0.3959 0.0276 0.4639]

H41 = [0.3403 -0.1529 -0.2258 -0.1770 -0.8823]

H51 = [-0.1518 0.0768 0.2043 0.9367 -0.2278]

H12 = [0.3634 0.4659 0.5486 -0.5785 0.1236]

H22 = [0.1419 -0.6010 -0.4425 0.6455 -0.0784]

H32 = [0.3763 0.0849 -0.6133 -0.5401 0.4282]

H42 = [-0.5911 0.4862 0.3541 -0.5114 -0.1653]

H52 = [-0.4306 -0.7983 0.3575 -0.1880 0.1187]

H13 = [0.1615 0.0148 -0.6927 -0.6535 -0.2583]

H23 = [-0.4773 -0.4249 0.5926 0.0224 -0.4899]

H33 = [-0.0204 -0.5591 -0.6686 -0.1294 0.4725]

H43 = [0.0889 0.4539 -0.8073 -0.3184 -0.1814]

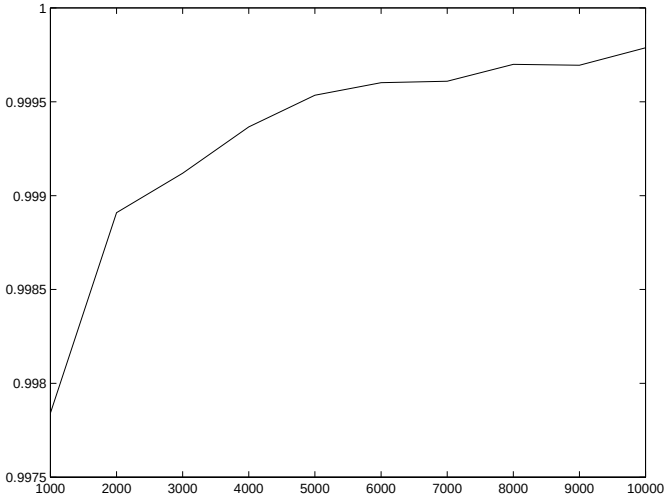

Figure 3.8: Similarity between transfer function vs. data size (2 sources and 5 sensors)

H53 = [-0.5722 -0.6222 0.4392 0.0053 -0.3042]

Plots of average similarity versus data size are shown in Figure 3.8, 3.9, 3.10, 3.11. From the plots, we can see the performance of the algorithm improves as the data size increases, as expected. Furthermore, we notice that we need longer data blocks to achieve the same performance as the number of sources increases. For the case of 2 sources and 5 sensors, it takes about 1000 data samples to achieve 0.9978 in similarity between transfer functions. For the case of 3 sources and 5 sensors, it takes about 10000 data samples to achieve 0.998 in similarity between transfer functions. That is a factor of 10 increase in computational latency. This increase in computational latency is caused by the increase of N, the number

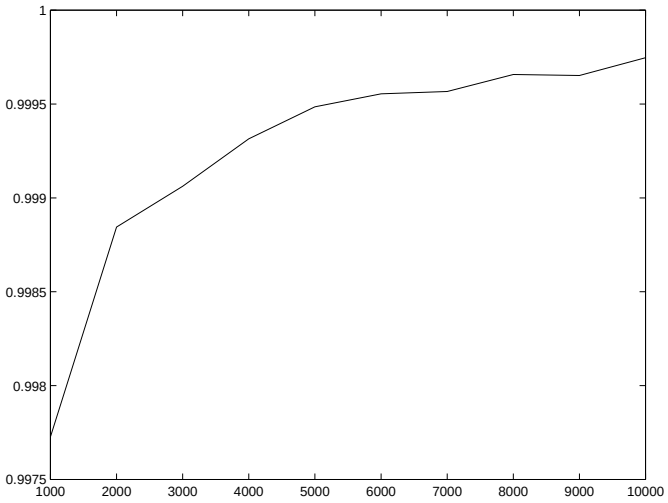

Figure 3.9: Similarity between data vs. data size (2 sources and 5 sensors)

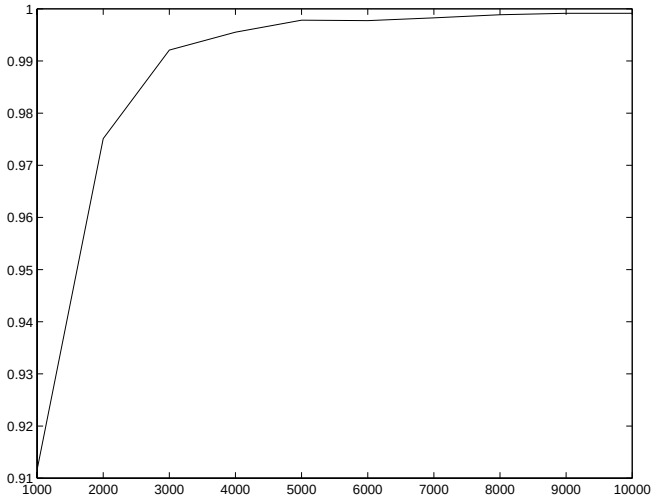

Figure 3.10: Similarity between transfer function vs. data size (3 sources and 5 sensors)

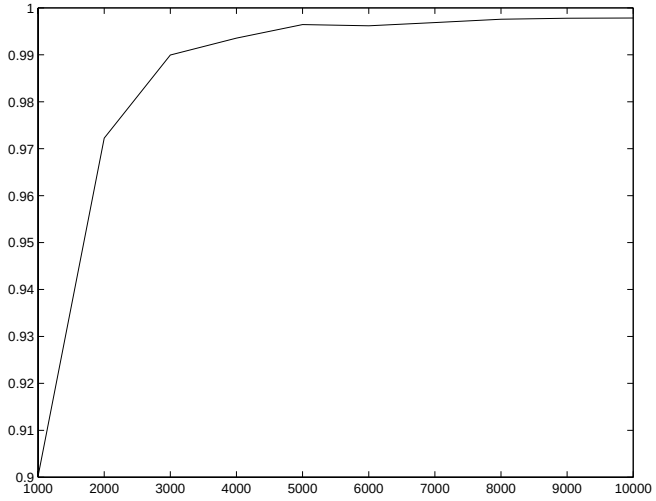

Figure 3.11: Similarity between data vs. data size (3 sources and 5 sensors)

of taps for the equalizer. As the number of sources increases, the number of taps for the equalizer needs to be increased to compensate for the channel. As a result, the number of time delay correlation coefficients we need to be estimated increases and we need larger data sets to achieve a comparable estimation. This property should be considered in the application of this algorithm. For sources that generate a short data set, we need to use a large number of sensors to recover a good data set. On the other hand, we can use fewer sensors for sources which generate very long data measurements.

In this section, we have presented a new algorithm for source separation and deconvolution for the multiple-input multiple-output system. The new algorithm improves on the previous algorithms [23] [27] [28] [30]. We consider blind source

separation and blind deconvolution jointly and solve these two problems together. This greatly expands the range of problems to which blind source separation and blind deconvolution can be applied. It is possible to perform a blind deconvolution operation even if there is more than one input source. On the other hand, blind source separation is possible for problems in which all the channels are dispersive. This algorithm is operating with block data only. In a real life application, it is preferable to have the algorithm operating in an adaptive mode. That is, the algorithms need to estimate the channel response and the deconvolution filter continuously with new incoming data. In section 3.2, we discuss the adaptive algorithm for source separation and deconvolution.

3.2 Adaptive Source Separation and Deconvolution Algorithm

Most of the adaptive algorithms for blind signal processing are implemented in the form of neural networks. Douglas [46] analyzed a class of adaptive networks for blind decorrelation of instantaneous signal mixtures using second order statistics. This class of algorithms can only decorrelate the signal mixtures, *i.e.*, it generates a set of uncorrelated signals out of a set of correlated signals. However, these recovered signals may not be the same as the original input sources. The

recovered signal is an orthogonal matrix transformation of the input source signal. Diamantaras [49] proposed the Asymmetric Principal Component Analysis (APCA) for adaptive blind source separation. In this neural network, the cross-correlation is used for source separation. All these neural networks are dealing with source separation for instantaneous signal mixtures. In practical systems, channels can be dispersive. The received signal is a combination of instantaneous and convolutive signal mixtures. In this paper, we propose an adaptive version of the blind deconvolution and source separation algorithm outlined in Section 3.1.

Consider a sensor network of N_s sensors and N_t sources as shown in Figure 3.2. The blind deconvolution and source separation algorithm proposed can be separated into two parts as shown in Figure 3.3. The blind deconvolution algorithm estimates $y_j(n)$ from $x_i(n)$. The relationship between $y_j(n)(j = 1, \ldots, N_t)$ and source signals $d_k(n)$ is shown in Equation 3.20. The blind source separation algorithm recovers $d_k(n)$ from $y_j(n)$.

The adaptive algorithm we propose consists of an adaptive deconvolution algorithm and an adaptive source separation algorithm. The adaptive deconvolution algorithm reduces the convolutive signal mixture problem into an instantaneous signal mixture problem. The adaptive source separation algorithm then separates the signal mixture and recovers the original signals. There is one subtle difference between decorrelation and source separation. The decorrelation algorithm turns

a set of correlated signals into a set of uncorrelated signals. However, this set of uncorrelated signals may not be the set of signals in which we are interested. On the other hand, the source separation algorithm goes one step further. It actually recovers the original source signal. In order to do that, the source separation algorithm needs more information, *i.e.*, time delayed cross-correlation.

We first present the adaptive deconvolution algorithm based on adaptive subspace tracking. The adaptive subspace tracking algorithm used was first proposed by Yang [33]. We then discuss the adaptive source separation algorithm. The source separation algorithm first uses Douglas' algorithm to decorrelate the signal mixture. The original input sources are recovered with one more matrix transformation. This matrix transformation is estimated using the time delayed cross-correlation.

3.2.1 Adaptive Deconvolution Algorithm

The operation of deconvolution recovers the source signals within an uncertainty of a matrix transformation. The deconvolution operation [1] can be summarized in the following procedures:

1. Estimating noise subspace G of R_x, where R_x is defined in Equation 3.5.

2. Generate matrix Q from subspace G, where Q is defined in Equation 3.17. Estimate noise eigenvectors $f_i (i = 1, \ldots, N_t)$ of matrix Q.

3. Generate matrix F from eigenvector f_i. The deconvolution filter E is

the inverse of matrix F. $F = [F_1, \ldots, F_{N_t}]$ (dim. N_sN x $N_t(M+N)$) and $F_i = [F_{i1}^T, \ldots, F_{iN_s}^T]^T$ (dim. N_sN x $(M+N)$) where F_{ij} (dim. N x $(M+N)$) is defined in Equation 3.19.

The noise subspace G is estimated using the method of adaptive subspace tracking. The adaptive noise subspace tracking is based on the algorithm proposed by Yang [33]. Given a matrix R_x (dim. N_sN x N_sN), the noise subspace G (dim. $N_sN - N_t(M+N)$ x $N_sN - N_t(M+N)$) can be obtained through minimizing cost function J_G given by

$$J_G = tr(G^T R_x G) \tag{3.29}$$

subject to the orthonormality constraint

$$G^T G = I \tag{3.30}$$

where $tr(\cdot)$ denotes the trace of a matrix.

The constrained minimization of the cost function J_G can be accomplished via a constrained gradient-search procedure. If the convergence rate is fixed, the noise subspace is updated as

$$G'(k) = G(k-1) - \mu \nabla J_G(k) \tag{3.31}$$

with $G(k)$ = GS-orthogonalization of the columns of $G'(k)$, where μ is the step size. $\nabla J_G(k)$ is an estimator of the gradient of J_G with respect to G. The gradient

of J_G is given by

$$\nabla J_G(k) = \frac{\partial}{\partial G}[tr(G^T R_x(k) G)] = 2R_x(k)G \tag{3.32}$$

In a nonstationary situation, $R_x(k)$ may be updated as

$$R_x(k) = (1-\alpha)R_x(k-1) + \alpha X(k)X^T(k) \tag{3.33}$$

The forgetting factor is controlled by the parameter α, $0 < \alpha < 1$, which is chosen for appropriate tracking of the parameters to be estimated. The step size μ should be chosen such that the estimate asymptotically converges to the desired noise subspace. A sufficient condition for this requirement is given by:

$$0 \leq \mu \leq \frac{1}{2\lambda_1} \tag{3.34}$$

where λ_1 is the maximum eigenvalue for matrix R_x. Since the λ_1 is not readily available, we can choose μ to be $\frac{1}{2tr(R_x)}$ which satisfies the above constraint.

The noise subspace tracking for matrix Q is similar to the noise subspace tracking for R_x. The update rule for noise subspace $\mathcal{F} = [f_1, \ldots, f_{N_t}]$ is given by

$$\mathcal{F}'(k) = \mathcal{F}(k-1) - \mu \nabla J_{\mathcal{F}}(k) \tag{3.35}$$

where $\mathcal{F}(k)$ = GS-orth. of the columns of $\mathcal{F}'(\|)$. $\nabla J_{\mathcal{F}}(k) = 2Q(k)\mathcal{F}$ is an estimator of the gradient of Q with respect to $\mathcal{F}$. Similarly to the previous section, $Q(k)$ can be estimated in an adaptive fashion.

$$Q(k) = (1-\alpha)Q(k-1) + \alpha \sum_{i=1}^{N_s N - N_t(M+N)} \mathcal{G}_i \mathcal{G}_i^T \tag{3.36}$$

Once the noise subspace $\mathcal{F}$ is known, matrix F can be determined through the transformation. The relationship between deconvolution filter E (dim. $N_t(M+N)$ x N_sN) and F is given by

$$E = (F^T F)^{-1} F^T \tag{3.37}$$

However, it is not necessary to calculate all $N_t(M+N)N_sN$ coefficients of E to construct the deconvolution filter. By exploring the structure of E and F, we can construct the deconvolution filter using N_tN_sN coefficients of E. Let $E = [E_{11}^T, \ldots, E_{1(M+N)}^T, \ldots, E_{N_t1}^T, \ldots, E_{N_t(M+N)}^T]^T$ where E_{ij} is the $\{(i-1)(M+N)+j\}$ row vector of E. Furthermore, E_{ij} can be expressed as $E_{ij} = [E_{ij1}^T, \ldots, E_{ijN_s}^T]^T$ where $E_{ijk}(1 \leq k \leq N_s)$ is a N x 1 vector. Using the above definitions and $EF = I$, we can show the following

$$E_{ij}^T F = e_{(i-1)(M+N)+j} \qquad \begin{aligned} 1 &\leq i \leq N_t \\ 1 &\leq j \leq M+N \end{aligned} \tag{3.38}$$

where e_k (dim. N_sN x 1) is the unit vector with kth index equal to 1. From Equation 3.38, we can recover $Y((i-1)(M+N)+j)$ using E_{ij}. This implies we can recover $y_i(n-j+1)$ with E_{ij}. In other words, E_{ij} is the deconvolution filter

for $y_i(n)$ where E_{ijk} is the filter coefficients for sensor k. In order to recover all $y_i(n)(i = 1, \ldots, N_t)$, we need to have N_t deconvolution filters E_{ij}. The index j determines the latency of the recovered $y_i(n)$ and we only need to estimate one set of E_{ij} for each i.

Each E_{ij} can be estimated individually through a constrained minimization problem. The cost function $J_{E_{ij}}$ is given by

$$J_{E_{ij}} = ||E_{ij}F - e_{(i-1)(M+N)+j}||_2 \tag{3.39}$$

subject to the unity norm constraint on E_{ij}. $||\cdot||_2$ is the Euclidean norm. This minimization problem can be solved using the gradient method:

$$E_{ij}(k) = E_{ij}(k-1) - \mu\nabla J_{E_{ij}}(k) \tag{3.40}$$

where $\nabla J_{E_{ij}}(k) = 2F(k)F^T(k)E_{ij}(k-1) - 2F(k)e_{(i-1)(M+N)+j}$.

3.2.2 Adaptive Source Separation Algorithm

The deconvolution algorithm in the previous section recovers Y from received signal X. The procedure for recovering source signal D from Y is called source separation. We note outline an adaptive algorithm for source separation. In the problem of instantaneous signal mixtures, a set of measured signals $y_i(k)(1 \leq i \leq N_t)$ is assumed to be generated from a set of unknown stochastic, independent sources $d_i(k)(1 \leq i \leq N_t)$ as

$$y(k) = Cd(k) \tag{3.41}$$

where $y(k) = [y_1(k), \ldots, y_{N_t}(k)]^T$, $d(k) = [d_1(k), \ldots, d_{N_t}(k)]^T$, and C is an unknown matrix of N_t^2 mixing coefficients $\{c_{ij}\}$. We want to process the measured sensor signals using a linear single-layer network $W(k)$.

$$z(k) = W(k)y(k) \tag{3.42}$$

where $W(k)$ is an N_t x N_t-dimensional weight matrix. We want to adjust $W(k)$ such that

$$\lim_{k \to \infty} W(k)C = PI \tag{3.43}$$

where P is an N_t x N_t-dimensional permutation matrix with a single unity entry in any of its rows or columns, and I is the identity matrix.

Douglas [46] analyzes a class of adaptive networks for blind decorrelation of instantaneous signal mixtures using the second order statistics. The class of adaptive decorrelation algorithms outlined by Douglas can be represented by

$$W(k+1) = W(k) + \eta(k)G(k) \tag{3.44}$$

where $G(k)$ is a matrix that depends on $z(k)$ and $W(k)$, and $\eta(k)$ is a step-size sequence. In particular, we choose $G(k) = (I - z(k)z^T(k))W(k)$ for our simulation. This algorithm uses the variable step-size sequence $\eta(k)$ for convergence

rate control. In order to have the fastest convergence without causing instability, the following rule for step size should be used.

$$\eta(k) = \begin{cases} \dfrac{0.5\eta_0(k)}{\sqrt{\beta_z(k)}} & \text{if } 0 < \beta_z(k) \leq 3 + 2\sqrt{2} \\ \dfrac{\eta_0(k)}{\beta_z(k) - 1} & \text{if } \beta_z(k) > 3 + 2\sqrt{2} \end{cases} \tag{3.45}$$

where $0 < \eta_0(k) < 0.1$ and $\beta_z(k)$ is $tr(R_{zz}(k))$. $R_{zz}(k)$ is the correlation matrix for $z(k)$.

The cost function for this type of adaptive algorithm is given by

$$\rho(k) = ||I - W(k)R_{yy}W^T(k)||_F^2 \tag{3.46}$$

where $||\cdot||_F$ denotes the matrix Frobenius norm. Without loss of generality, we assume all sources are equal power and $R_{dd} = I$. Then we have $R_{yy} = CC^T$ for our signal model. However, it can shown that minimizing Equation 3.46 is not a sufficient condition for satisfying Equation 3.43. It can be shown with the following example.

Given W that satisfied Equation 3.46, define $W_1 = U^T W$ where U is an orthonormal matrix, it can shown that

$$||I - W_1 R_{yy} W_1^T||_F^2 = ||I - U^T P I I^T P^T U||_F^2 = 0 \tag{3.47}$$

As a result, Douglas' algorithm is only able to decorrelate a set of instantaneous signal mixtures. It is not capable of separating a set of instantaneous signal mixtures. The recovered signal from Douglas' algorithm is still off from the original source signals by an orthonormal matrix U. It can be shown that one set of second order statistics is not sufficient to uniquely separate sources [28]. However, it is possible to uniquely separate sources if there is more than one set of second order statistics, *i.e.*, time-delay second order statistics. If all sources d_i are colored sources and satisfy the following uncorrelated property stated in Equation 3.7, we can recover the unknown orthonormal matrix E using time-delay second order statistics.

We decorrelate $y(k)$ using Douglas' algorithm to generate $z(k)$. The relationship between $z(k)$ and $d(k)$ is

$$z(k) = U^T(k)d(k) \tag{3.48}$$

where $U(k)$ is a orthonormal matrix.

U can be estimated by minimizing the following cost function J_U

$$J_U(k) = \frac{1}{2}\{\sum_{i=1}^{N_t} \log(s_i(k)s_i(k-\tau)) - \log|s(k)s(k-\tau)|\} \tag{3.49}$$

where $s(k) = U(k)z(k)$. $s(k)$ is the estimate for source signal $d(k)$.

The minimization problem can be solved using the gradient method:

$$U'(k) = U(k-1) - \mu \nabla J_U(k) \tag{3.50}$$

$$\nabla J_U(k) = U^T(k-1)\{(diag(s(k)s(k-\tau)^T))^{-1}(s(k)s(k-\tau)^T - (diag(s(k)s(k-\tau)^T))^{-1}\}$$

where $U(k)$=GS-orthogonalization of the columns of $U'(k)$.

3.2.3 Simulation Results

We evaluate the performance of the adaptive algorithm through simulations. Two sets of simulations are performed to evaluate the performance of the algorithm. In the first set of simulations, we show the blind sources separation capability of the algorithm on instantaneous signal mixtures. Two simulation cases are run and the results are shown in Figures 3.12 and 3.13. One simulation is for a 2 x 2 mixture matrix and another one is for a 3 x 3 mixture matrix.

The input sources are generated by passing white Gaussian sources through filter G_i defined in Equation 3.27.

We evaluate the performance of the algorithm by computing the average similarity between sources inputs and recovered inputs.

As shown in the figures, the algorithm can separate the signal mixtures. In the 3 x 3 matrix case, the recovered signals are off from the source signals by a permutation matrix.

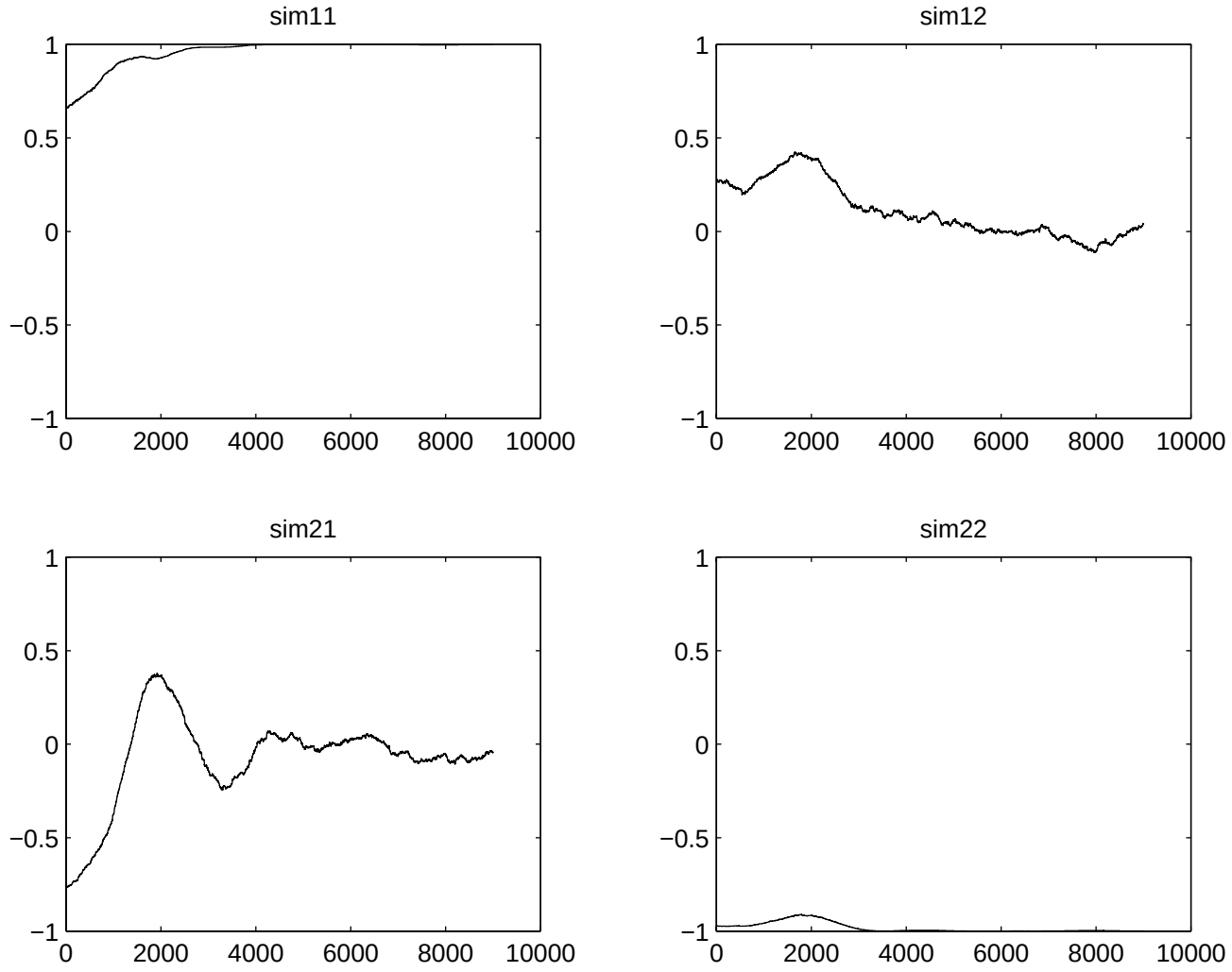

Figure 3.12: Average similarity between data vs. iterations (2 sources mixture)

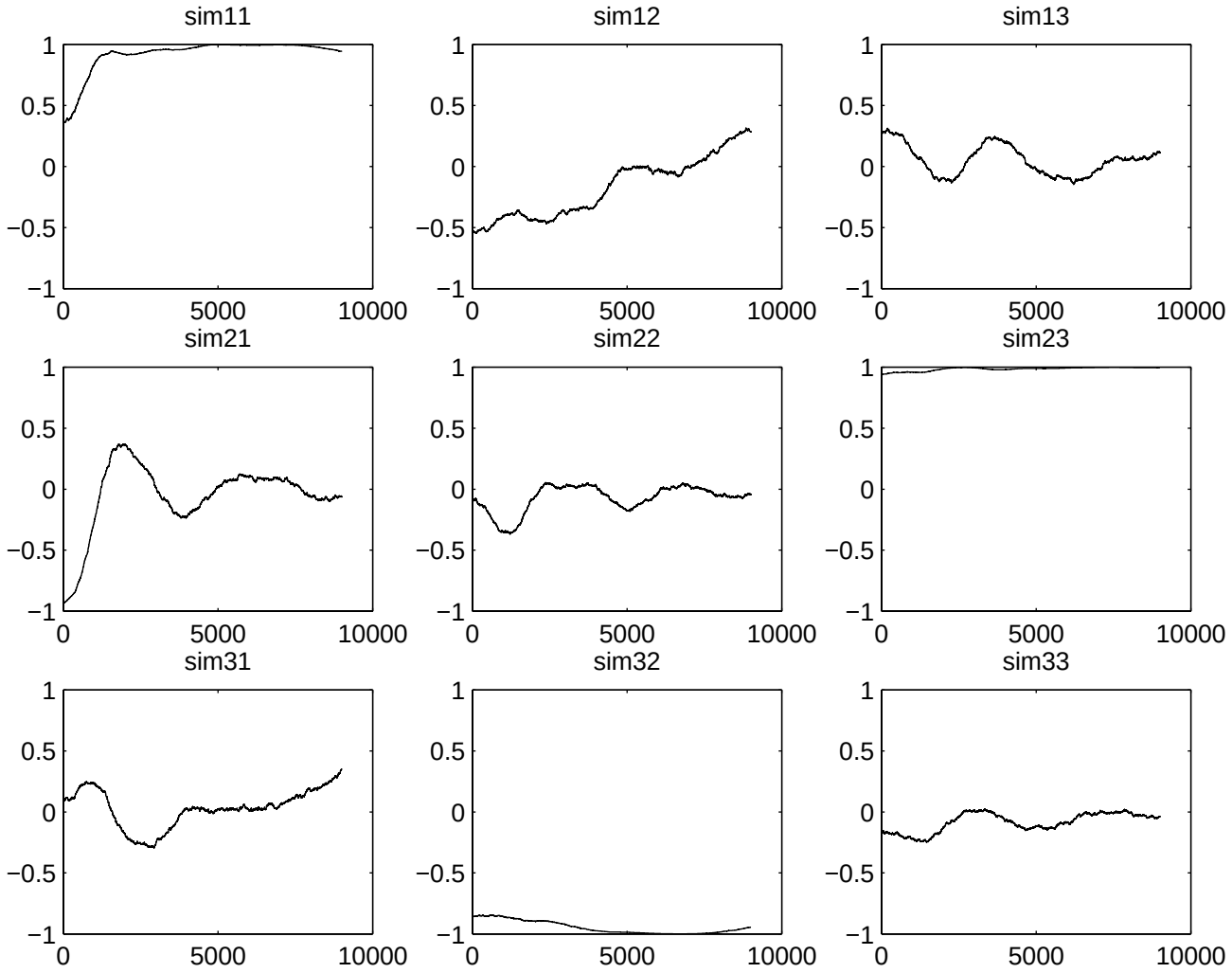

Figure 3.13: Average similarity between data vs. iterations (3 sources mixture)

In the second set of simulations, we investigate the performance of the algorithm for convolutive signal mixtures. Two simulation case are run and the results are shown in Figure 3.14 and 3.15. The following table summarizes the simulation setup:

Simulation Case	N_t	N_s	M	N
1	2	5	4	6
2	3	5	4	18

N_t, number of inputs sources

N_s, number of sensors

M, maximum order of each channel

N, number of taps for the equalizer

The input sources are generated by passing white Gaussian sources through filter G_i defined in Equation 3.27. The coefficients of H_{ij} are produced by a Gaussian number generator.

As shown in the figures, the adaptive algorithm is capable of adjusting filter coefficients for blind deconvolution and source separation. However, we notice the convergence time is quite large and it gets longer as the number of input sources increases. In the case of 2 sources and 5 sensors, it takes about 2500 iterations to converge. For the case of 3 sources and 5 sensors, the number of iterations for convergence is about 25000. That is a ten times increase in computational

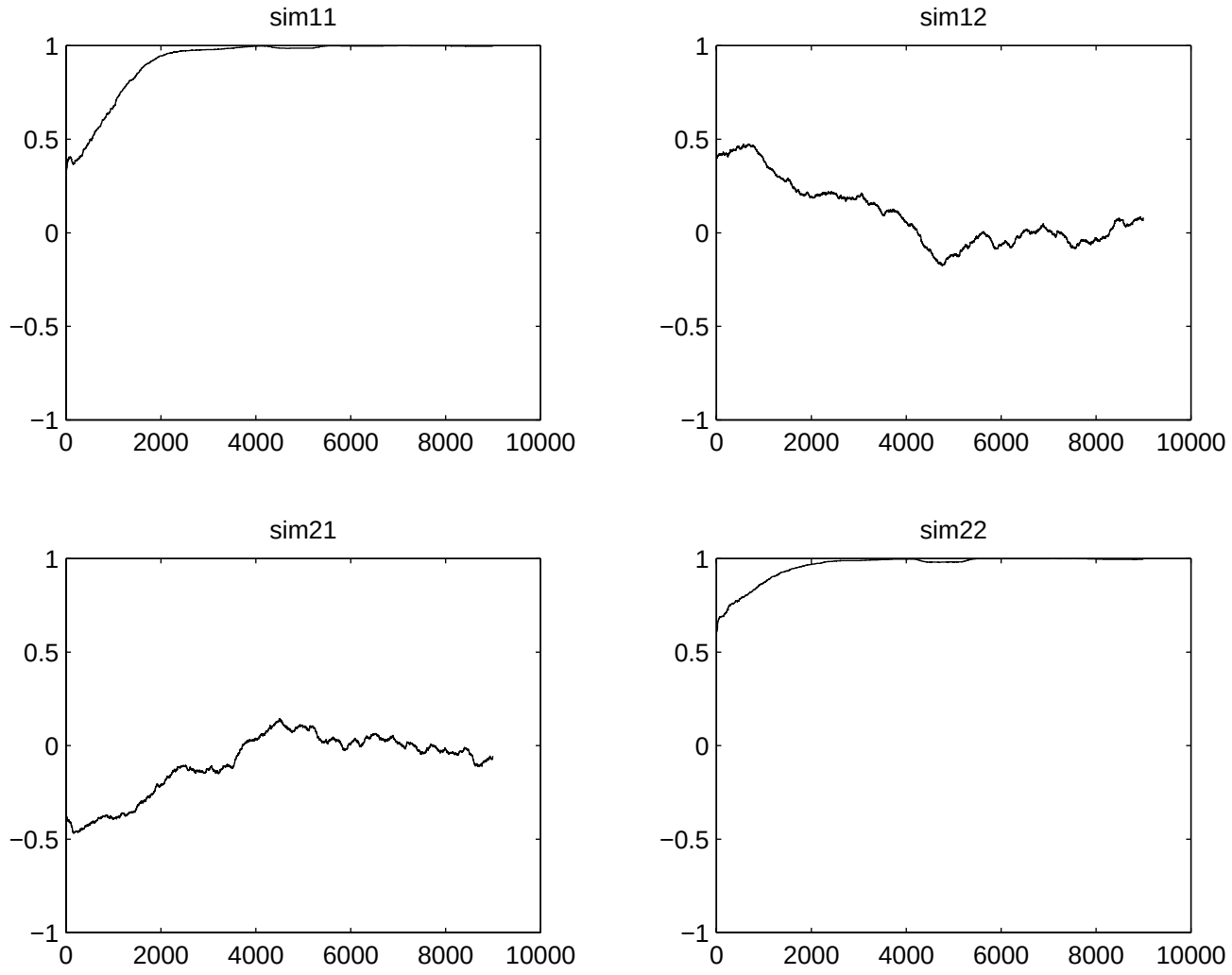

Figure 3.14: Average similarity between data vs. iterations (2 sources and 5 sensors)

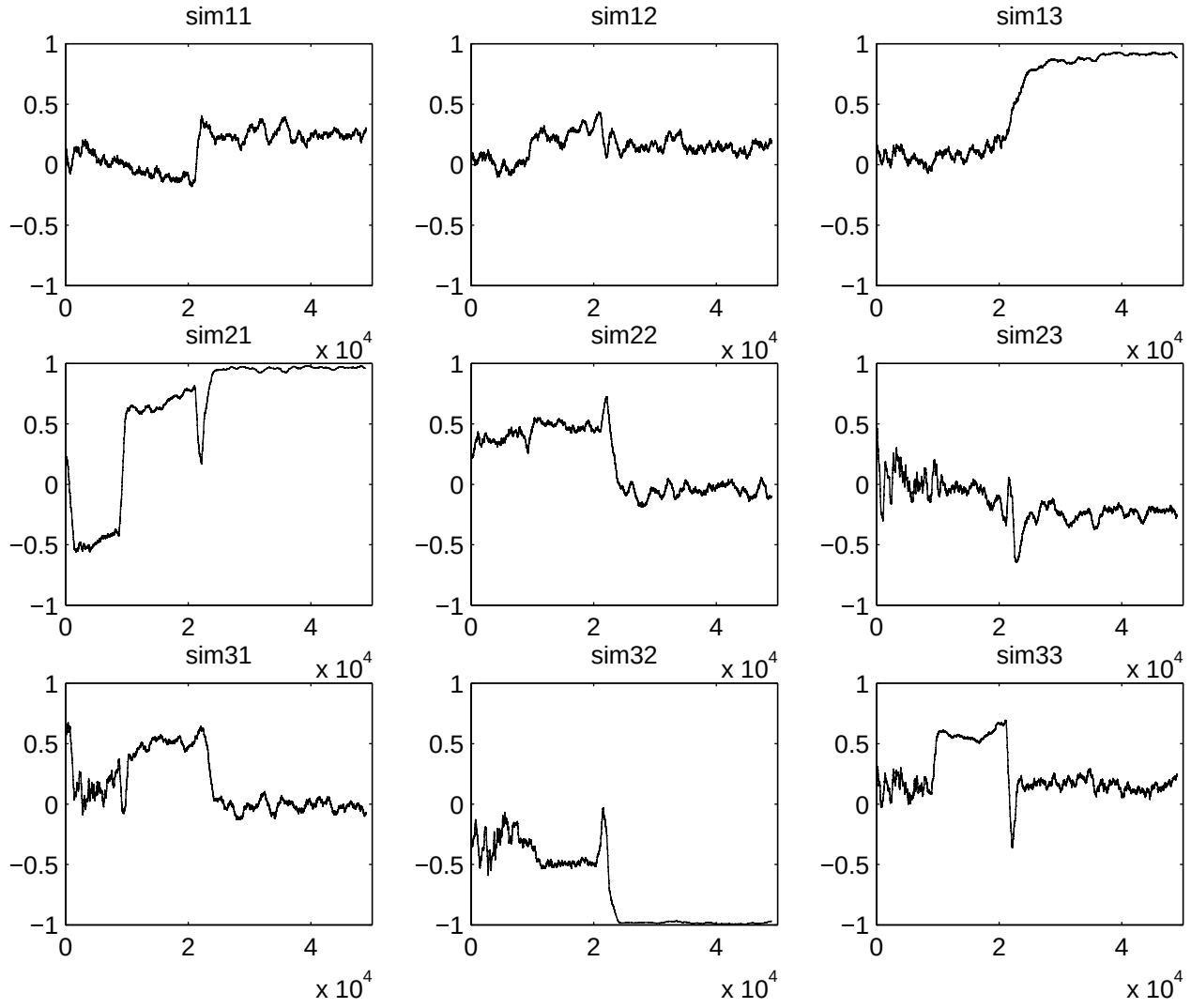

Figure 3.15: Average similarity between data vs. iterations (3 sources and 5 sensors)

complexity.

3.3 Summary

In this chapter, we presented a structure for the source separation and deconvolution processor to be used in our target identification processor. This subprocessor has two main functions. First, it performs the source separation for the mixture of multiple sources. As a result, the subsequent recognition processor is only required to perform identification on a single target type instead of multiple target types together. This greatly reduces the complexity of the subsequent processor and improves the overall performance of the system. Second, it performs the deconvolution to reduce signal degradation caused by multipath propagation. This further improves the robustness of our system. The new algorithm we proposed for this subprocessor has the least number of constraints on the unknown signals we are trying to recover. This increases the coverage of our sensor system. After the source separation and deconvolution processor, our sensor data has been converted from a mixture of many signals into a single source signal. In the following chapters, we perform further processing on this single source signal for target identification.

Chapter 4

Feature Extraction Processor

The second subprocessor module to be considered is the feature extraction processor. The feature extraction processor converts the time series waveform to some type of parametric representation (generally at a considerably lower information rate) for further analysis and processing. The aim of feature extraction is to reduce the computational complexity of subsequent processing. There are a wide range of possibilities for parametrically representing the time series waveform. These include the short time energy, zero crossing rates, and level crossing rates.

In the area of speech recognition, the most important parametric representation for time series waveforms (speech) has been the short time spectral envelope. The short time spectral envelope representation breaks the time series waveform into frames. Spectral analysis is performed for each frame. Each frame is then

represented by a vector (generally much shorter length than the frame) containing all the spectral information of the frame. Because of its success in speech recognition, we are going to use short time spectral envelope representation for our feature extraction processor.

There are two dominant methods of spectral analysis - namely, the filter-bank spectrum analysis model, and the linear predictive coding (LPC) spectral analysis model. However, the LPC spectral analysis has been the popular choice for the front-end processor in the application of speech recognition, for a number of reasons. First, LPC is an analytically tractable model. The method of LPC is mathematically precise and is simple and straightforward to implement in either software or hardware. The computation involved in LPC processing is considerably less than that required for an all-digital implementation of the filter-bank approach. Furthermore, experience in speech recognition has shown that the performance of speech recognizers, based on LPC front ends, is comparable to or better than that of recognizers based on filter-bank front ends. As a result, LPC spectral analysis is chosen for our feature processor.

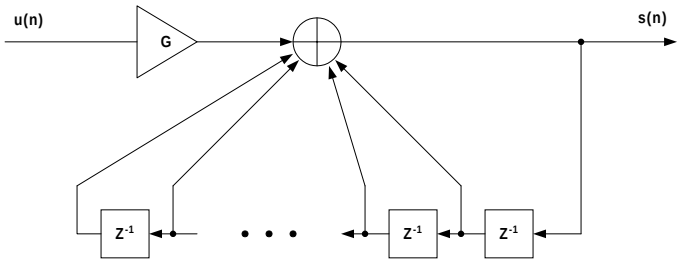

Figure 4.1: Autoregressive Model Block Diagram

4.1 LPC Spectral Analysis

The basic idea behind the LPC model is that a given time series waveform sample at time n, $s(n)$, can be approximated as a linear combination of the past p time series samples as shown in Figure 4.1, such that

$$s(n) \approx a_1 s(n-1) + a_2 s(n-2) + \ldots + a_p s(n-p) \tag{4.1}$$

where the coefficients a_1, a_2, ..., a_p are assumed constant over one frame. The above relationship can be written as an equality by including an excitation term, $Gu(n)$, giving:

$$s(n) = \sum_{i=1}^{p} a_i s(n-i) + Gu(n) \tag{4.2}$$

where $u(n)$ is a normalized excitation and G is the gain of the excitation. In this model, the time series waveform $s(n)$ is generated by a normalized excitation

source, $u(n)$, being scaled by the gain, G, passing through an all-pole system.

The basic problem of linear prediction analysis is to determine the set of predictor coefficients, $\{a_k\}$, directly from the time-series waveform. One of the methods is the autocorrelation method. Consider a time series waveform segment, $s_n(m)(0 \leq m \leq N-1)$. We can derive the following relationship from Equation 4.2 under the minimum mean-squared error criterion.

$$\sum_{k=1}^{p} r_n(|i-k|)a_k = r_n(i) \qquad 1 \leq i \leq p \tag{4.3}$$

where $r_n(k)$ is the autocorrelation coefficient and it is defined as:

$$r_n(i-k) = \sum_{m=0}^{N-1-(i-k)} s_n(m)s_n(m+i-k) \tag{4.4}$$

Equation 4.3 can also be expressed in matrix form as

$$\begin{bmatrix} r_n(0) & r_n(1) & r_n(2) & \dots & r_n(p-1) \\ r_n(1) & r_n(0) & r_n(1) & \dots & r_n(p-2) \\ r_n(2) & r_n(1) & r_n(0) & \dots & r_n(p-3) \\ \vdots & \vdots & \vdots & & \vdots \\ r_n(p-1) & r_n(p-2) & r_n(p-3) & \dots & r_n(0) \end{bmatrix} \begin{bmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_p \end{bmatrix} = \begin{bmatrix} r_n(1) \\ r_n(2) \\ r_n(3) \\ \vdots \\ r_n(p) \end{bmatrix} \tag{4.5}$$

Once the LPC coefficients $\{a_k\}$ have been determined, each time series frame can be replaced by a vector of LPC coefficients. These vectors of LPC coefficients can then be processed further for target recognition. In the application of speech

recognition, the cepstral coefficients are also used for spectral representation. The experience in speech recognition has shown that the set of cepstral coefficients is a more robust and reliable feature set than the LPC coefficients. These LPC cepstral coefficients, $c(m)$, can be derived directly from the LPC coefficients $a(m)$. As a result, the final output of our feature extraction processor is chosen to be the LPC cepstral coefficients.

The cepstral coefficients are the coefficients of the Fourier transform representation of the log magnitude spectrum. For a power spectrum (magnitude-squared Fourier transform) $\mathcal{S}(\omega)$, which is symmetric with respect to $\omega = 0$ and is periodic for a sampled data sequence, the Fourier series representation of log $\mathcal{S}(\omega)$ can be expressed as

$$log\mathcal{S}(\omega) = \sum_{n=-\infty}^{\infty} c_n \exp^{-jn\omega} \tag{4.6}$$

where $c_n = c_{-n}$ are the cepstral coefficients and they are real. Given a pair of spectra $\mathcal{S}(\omega)$ and $\mathcal{S}'(\omega)$, we can relate the $\mathcal{L}_2$ cepstral distance of the spectra to the rms log spectral distance using Parseval's theorem:

$$\begin{aligned} d_2^2 &= \int_{-\pi}^{\pi} |log\mathcal{S}(\omega) - log\mathcal{S}'(\omega)|^2 d\omega/2\pi \\ &= \sum_{n=-\infty}^{\infty} (c_n - c'_n)^2 \end{aligned} \tag{4.7}$$

where c_n and c'_n are the cepstral coefficients of $\mathcal{S}(\omega)$ and $\mathcal{S}'(\omega)$ respectively.

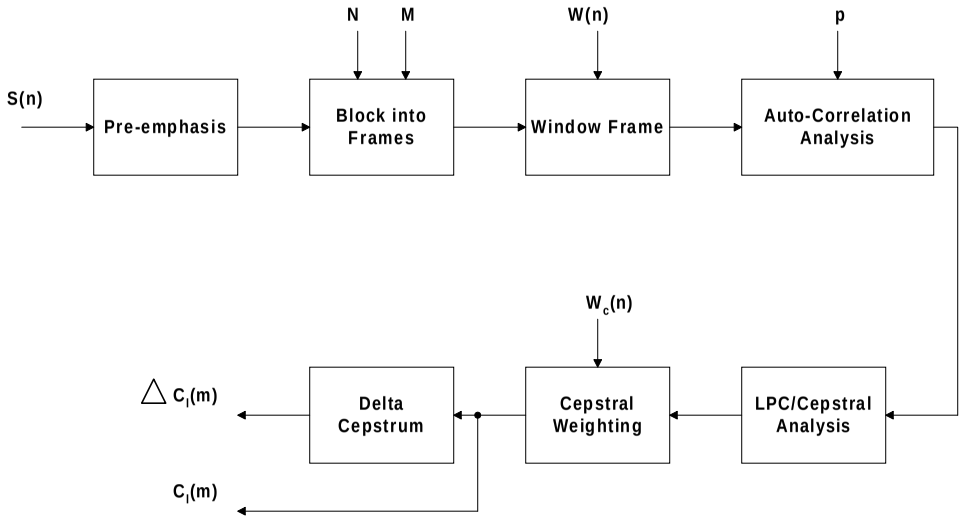

Figure 4.2: LPC Processor Block Diagram

Rabiner [8] presents a detailed coverage of LPC spectral analysis in his book. A summary of the LPC spectral analysis is presented in this section. A block diagram for the LPC spectral analysis is shown in Fig. 4.2. The overall system is a block processing model in continuous samples are processed and feature vectors O_t are generated. The steps in the processing are as follows:

1) *Preemphasis:* The digitized speech signal, $s(n)$, is put through a low-order digital system, to spectrally flatten the signal and to make it less susceptible to finite precision effects later in the signal processing. In our system, a fixed first-order system is used for the preemphasis network:

$$\mathcal{H}(z) = 1 - az^{-1} \tag{4.8}$$

Since most real life signals have most of their energies concentrated in the low

end of the spectrum, the high frequency components of their signals which may be critical to recognition are not readily visible. The procedure of preemphasis can make the high frequency spectral features more visible by placing more weight on them.

2) *Blocking into Frames:* Short time spectral analysis is done on a frame to frame basis. In this step the preemphasized speech signal is blocked into frames of N samples, with adjacent frames being separated by M samples. In our application, M is chosen to be smaller than N so the resulting LPC spectral estimates will be correlated from frame to frame. As a result, LPC spectral estimates from frame to frame will be quite smooth.

3) *Frame Windowing:* In order to minimize the signal discontinuities at the beginning and end of each frame, we use windowing to taper the signal to zero at the beginning and end of each frame. Given data $x_l(n)$ in frame l and window $w(n)$, the result of windowing is the signal

$$\tilde{\mathrm{x}}_l(n) = x_l(n)w(n) \qquad 0 \leq n \leq N-1 \tag{4.9}$$

For our application, the Hamming window is used and it has the form

$$w(n) = 0.54 - 0.46cos(\frac{2\pi n}{N-1}) \qquad 0 \leq n \leq N-1 \tag{4.10}$$

4) *Autocorrelation Analysis:* Each windowed set of samples is autocorrelated

to give a set of $(p+1)$ coefficients, where p is the order of the desired LPC analysis.

$$r_l(m) = \sum_{n=0}^{N-1-m} \tilde{\mathrm{x}}_l(n)\tilde{\mathrm{x}}_l(n+m) \qquad m = 0, 1, \ldots, p \tag{4.11}$$

5) *LPC/Cepstral Analysis:* For each frame, a vector of LPC coefficients is computed from the autocorrelation vector using a Durbin recursion method. The Durbin's method is given as the following algorithm:

$$\begin{aligned}
E^{(0)} &= r(0) && (4.12)\\
k_i &= \left\{ r(i) - \sum_{j=1}^{L-1} \alpha_j^{(i-1)} r(|i-j|) \right\} / E^{i-1} && 1 \le i \le p \quad (4.13)\\
\alpha_i^i &= k_i && (4.14)\\
\alpha_j^i &= \alpha_j^{i-1} - k_i \alpha_{i-j}^{(i-1)} && (4.15)\\
E^{(i)} &= (1 - k_i^2) E^{(i-1)} && (4.16)
\end{aligned}$$

The above set of equations are solved recursively for $i = 1, 2, \ldots, p$, and the final solution is given as

$$a_m = \text{LPC coefficients} = \alpha_m^{(p)} \qquad 1 \leq\leq p \tag{4.17}$$

The LPC coefficients are used to derived the LPC cepstral coefficients $c(m)$. In general, a cepstral representation with $Q > p$ coefficients is used. The conversion procedure is:

$$c_0 = ln\sigma^2 \tag{4.18}$$

$$c_m = a_m + \sum_{k=1}^{m-1}(\frac{k}{m})c_k a_{m-k} \qquad 1 \leq m \leq p \tag{4.19}$$

$$c_m = \sum_{k=1}^{m-1}(\frac{k}{m})c_k a_{m-k} \qquad m > p \tag{4.20}$$

where σ^2 is the gain term in the LPC model. Note that while c_0 corresponds to the energy of the model, this term is not used in the recognition since the comparison is done on the spectral envelope following the speech recognition algorithm. However, c_0 can be useful for the target recognition application. First, since c_0 is the energy of each signature frame, it can be used for target tracking. Second, as we shall see in the next chapter, since our pattern recognition processor is based on a statistical model, the probability measure assigned to each signature frame should be weighted by c_0. By doing so, we should generate better classification decisions.

6) *Cepstral Weighting:* The cepstral weighting is one additional step to condition the Q-coefficient cepstral vector. Its aim is to minimize the sensitivity of the low-order cepstral coefficients to overall spectral slope and the sensitivity of the higher-order cepstral coefficients to noise. In order to do that, a bandpass lifter $W_c(m)$ is chosen and it has the form

$$W_c(m) = 1 + \frac{Q}{2}sin(\frac{\pi m}{Q}) \qquad 1 \leq m \leq Q \tag{4.21}$$

to give

$$\hat{c}_l(m) = c_l(m) * W_c(m) \tag{4.22}$$

This bandpass lifter de-emphasizes c_m around $m = 1$ (low-order) and around $m = Q$ (high order).

7) *Delta Cepstrum:* The cepstral representation can be improved by adding the temporal cepstral derivative. The time derivative of the sequence of weighted cepstral vectors is approximated by a first-order orthogonal polynomial over a finite length window of $(2K+1)$ frames, centered around the current vector. The cepstral derivative is computed as

$$\Delta\hat{c}_l(m) = \left[\sum_{k=-K}^{K} k\hat{c}_l(m+k)\right] * \mu \qquad\qquad 1 \leq m \leq Q \tag{4.23}$$

where μ is a gain term chosen to make the variances of $\hat{c}_l(m)$ and $\Delta\hat{c}_l(m)$ equal.

The signature vector O_l used for recognition and training is the concatenation of the weighted cepstral vector, and the corresponding weighted delta cepstrum vector, *i.e.*,

$$O_l = \{\hat{c}_l(m), \Delta\hat{c}_l(m)\} \tag{4.24}$$

4.2 Simulation

We illustrate the operation of LPC spectral analysis with simulations using real life data. Acoustic signatures for four different types of vehicles are shown in Figure 4.3. Vehicle signatures are shown in the clockwise direction: 1) heavy tracked vehicle, 2) heavy wheeled vehicle, 3) light wheeled vehicle, and 4) light tracked vehicle.

LPC spectral analysis is performed for these acoustic signatures using the following parameters:

N, window size = 3000 samples

M, window spacing = 1000 samples

p, number of autocorrelation coefficients used = 6

Q, number of cepstral coefficients generated = 9

K, window for cepstral derivative calculation = 2

The resulted cepstral vector and delta cepstrum vector are shown in Figures 4.4 and 4.5. The horizontal axis corresponds to the frame number. While there is no simple way to distinguish between different vehicles using raw acoustic data or cepstrum data, we can reduce the amount of data by several orders of magnitude using LPC spectral analysis.

4.3 Summary

In this chapter, we outlined the architecture for the feature extraction processor. It is capable of reducing the information rate with minimal loss of information. Consider, for example, 1-kHz sampled sensor data with 16-bit amplitudes. A raw signal information rate of 16,000 bps is required to store the sensor data in uncompressed format. Consider the feature extraction processor with $Q = 9$ and 1 spectral vector per second. If we represent each signature vector to 16-bit precision, the information rate is then 2 x 9 x 16 bps, or 288 bps - about a 60-to-1 reduction over the uncompressed signal.

We want to point out, however, LPC spectral analysis is not the only choice for parametric representation. As stated before, there is also filter bank spectral analysis. The key drawback of filter bank approach is the higher complexity in hardware implementation. Scholl [53] [54] proposes using the wavelet transform for target classification. The wavelet transform can be categorized as the filter bank spectral analysis in the general sense. However, since wavelet transforms can be efficiently implemented using a multi-rate filter bank structure, it can be a very attractive alternative to LPC spectral analysis. In particular, wavelet transforms are very popular for the analysis of nonstationary signals. The analysis of nonstationary signals often involves a compromise between how well transitions or discontinuities can be located, and how finely long-term behavior can be identified. A typical example is the choice of window length in the short-time Fourier

transform. In wavelet analysis one looks at the signal at different resolutions: a rough approximation shows how the signal might look if stationary, while at detailed level (when using a small window) discontinuities become apparent. This multiresolution view of signal analysis is the essence of the wavelet transform. However, there are some open issues for wavelet transforms in our application. The most important one is the type of mother wavelet that should be used for our applications. Haar wavelet has the shortest time duration. This class of wavelet bases is ideal for short time signals as it provides the best localization in the time domain. The drawback is poor resolution in the frequency domain. On the other end of the comparison, there is the Fourier wavelet which offers the best resolution in the frequency domain and performs poorly in time localization. There are also some other classes of wavelet basis that have performances bounded by these two, *i.e.*, Meyer, Batle, and Lemarie. The proper choice of wavelet basis for our feature extraction is yet to be researched.

At this point, the time series sensor data has been converted into a sequence of signature vectors. The final identification of this sequence of signature vectors is addressed in the next chapter.

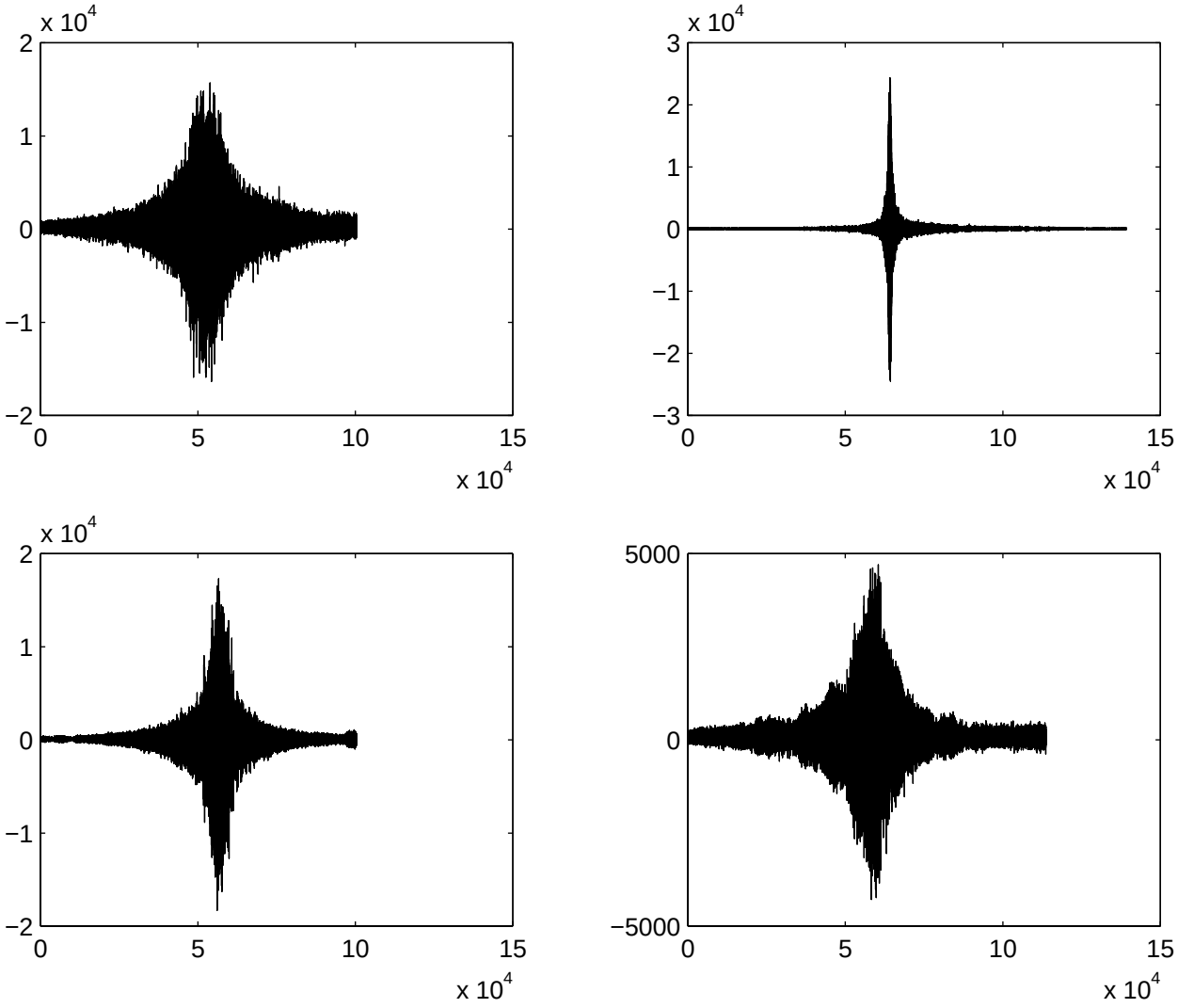

Figure 4.3: Acoustic Signatures

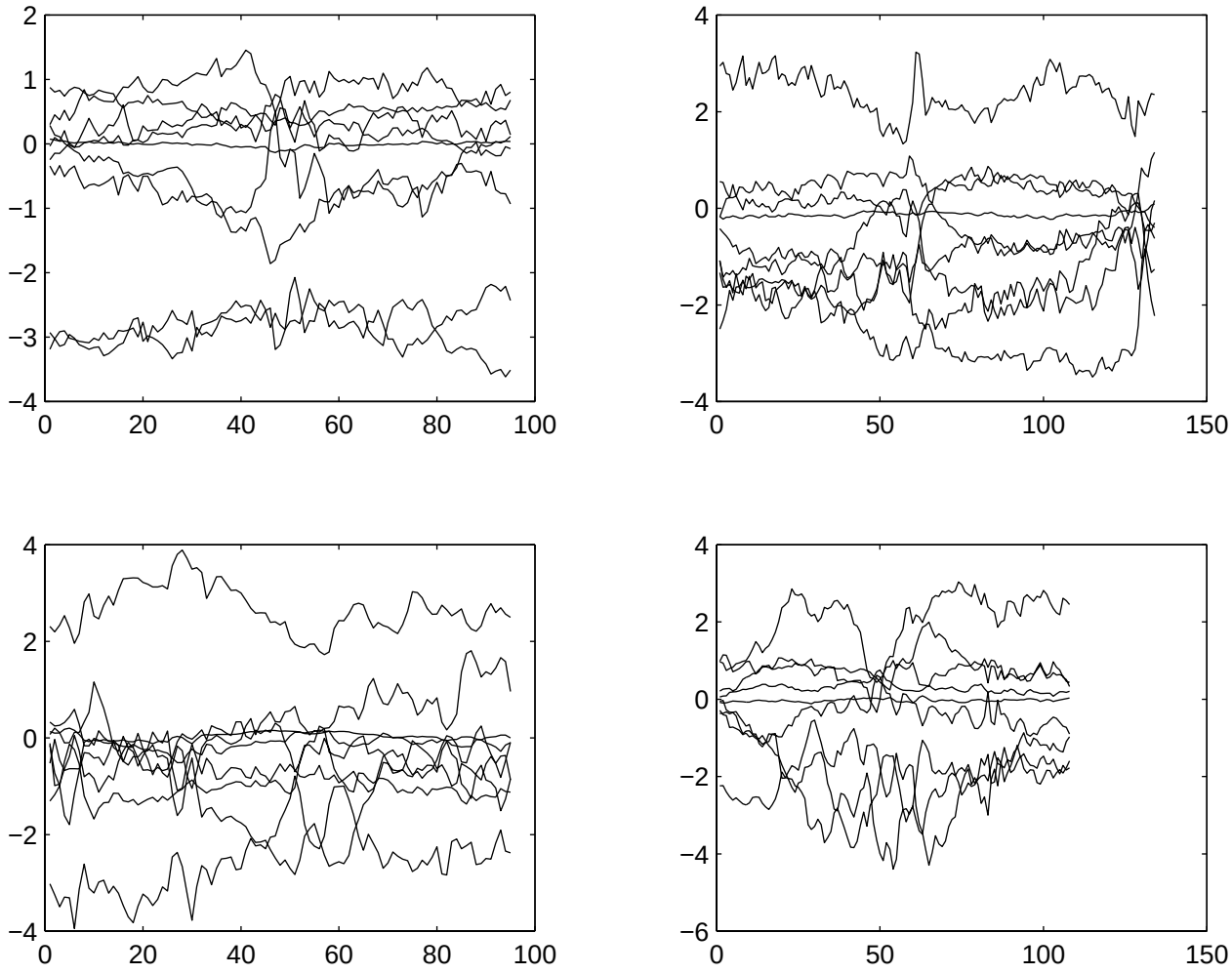

Figure 4.4: Cepstral Vectors

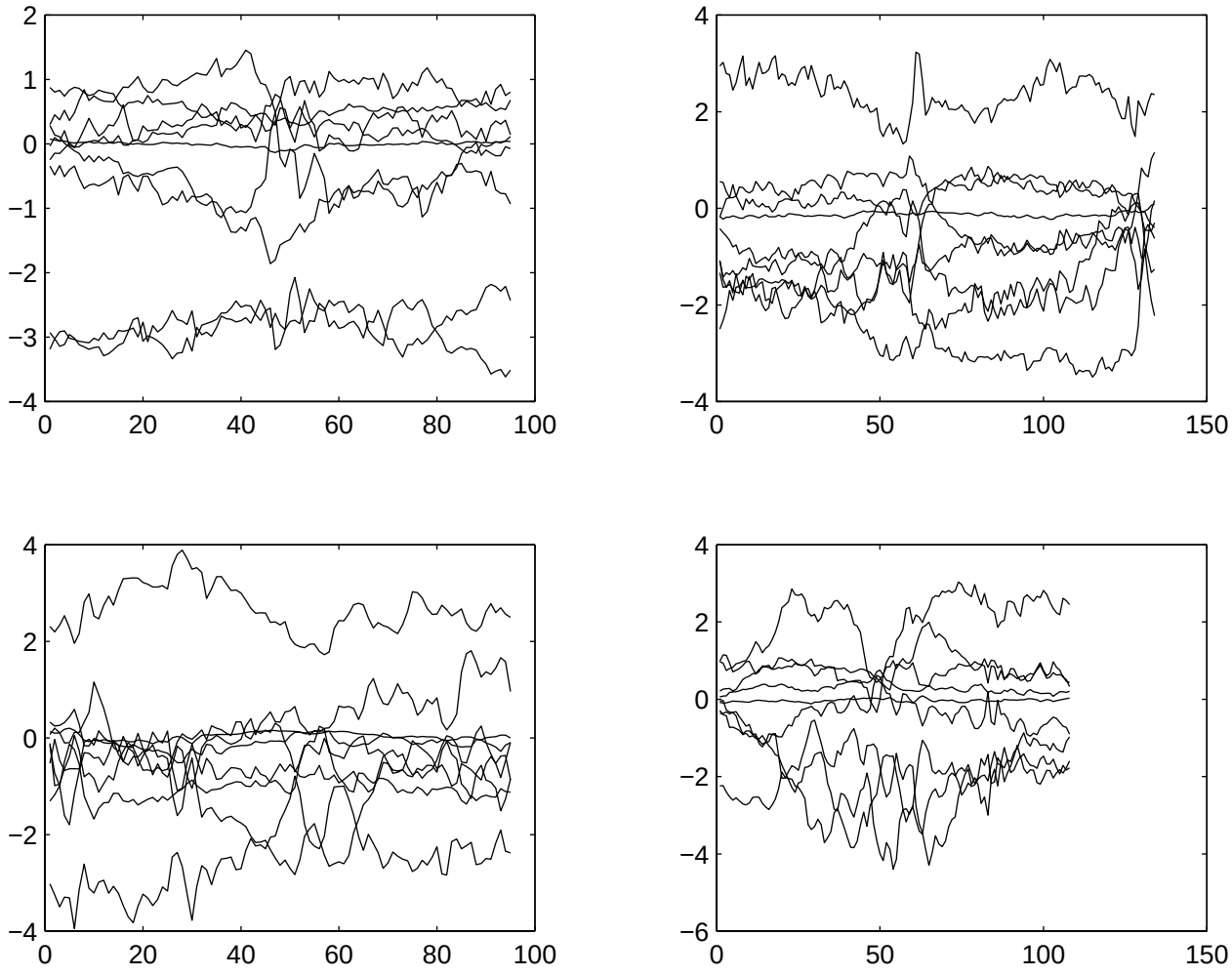

Figure 4.5: Delta Cepstrum Vectors

Chapter 5

Pattern Recognition Processor

The signature vectors (cepstral coefficients and cepstral derivatives) can be classified individually by comparing them with some reference patterns. However, it is rare to have an event consisting of one signature vector. Most real life events consist of many signature vectors. One simple extension to the individual vector comparison test can be the template method [8]. In our pattern recognition problem, we have a signature pattern O, as the concatenation of spectral frames over the duration of the event, such that

$$O = O_1, O_2, O_3, \ldots, O_t, \tag{5.1}$$

where each O_i is the spectral vector of the target at time i, and t is the total number of frames. These spectral vectors are obtained from the front-end spectral analysis discussed in Chapter 4. In a similar manner we define a set of reference

patterns, $\{R^1, R^2, \ldots, R^V\}$ where each reference pattern, R_j, is also a sequence of spectral frames, such that

$$R_j = r_1^j, r_2^j, \ldots, r_J^j. \tag{5.2}$$

The goal of the template method is to determine the dissimilarity or distance of O to each of the $R^j, 1 \leq j \leq V$, in order to identify the reference pattern that has the minimum dissimilarity, and to associate our test pattern to this pattern. The key idea in the template method is to derive typical sequences of waveform frames for an event via some averaging procedure, and to rely on the use of local spectral distance measures to compare patterns. Another key idea is to use some form of dynamic programming to temporally align patterns to account for differences in target movements across targets. The methodology of the template approach is well developed and provides good recognition performance for a variety of practical applications, *i.e.*, speech recognition.

The template approach, however, is not based on the ideas of statistical signal modeling in a strict sense. Even though statistical techniques have been widely used in clustering to create reference patterns, the template approach is best classified as a simplified, non-parametric method in which a multiplicity of reference sequences are used to characterize the variation among different events. As a result, statistical signal characterization inherent in the template representation is only implicit and often inadequate. Consider, for example, the use of cepstral

distortion measure as the local distance for template matching. The Euclidean distance form of the cepstral distance measure suggests that the reference vector can be viewed as the *mean* of some assumed distribution. The simple form of the sufficient statistic neglects the second-order statistics - i.e., *covariance,* which can be of particular significance in statistical modeling. On the other hand, the variation in real life events can be quite large. In order to model our targets accurately, a large set of reference patterns is required. It is very costly to build a library of a large number of patterns. Furthermore, it is computationally expensive to search through the library to identify an event. Clearly, we should use a more elaborate and analytical statistical method. As a result, we introduce the concept of a signal model in this chapter. We choose the hidden Markov model (HMM) for its good success in speech recognition problems.

Rabiner provides a very detailed analysis in his book [8] on the subject of HMM and its application in speech recognition. In this section, we borrow heavily from Rabiner's analysis and apply the HMM signal model to our target recognition problem.

We want to illustrate a modeling example before addressing the details of the pattern recognition processor architecture. An event of a tracked vehicle passing by a sensor consists of a set of time series frames. Each frame is represented by a signature vector. In our HMM model, this sequence of signatures is generated by a state sequence. The physical interpretation for each state corresponds to

the maneuver of the tracked vehicle. In the case of the three states model, three states are: 1) tracked vehicle moving toward the sensor, 2) tracked vehicle moving away from the sensor, and 3) tracked vehicle is tangent (appears to be motionless) to the sensor. At each state, the tracked vehicle can generate a set of possible signature vectors. Since the underlying state sequence is not known, that is the reason the model is a hidden state model. The state transition of the unknown state sequence is assumed to be a Markov chain. While this Markov Chain assumption is still open to justification, it is a well understood model and can be readily used in our application. Furthermore, our simulations using the HMM model yield very good classification performance.

5.1 Hidden Markov Model

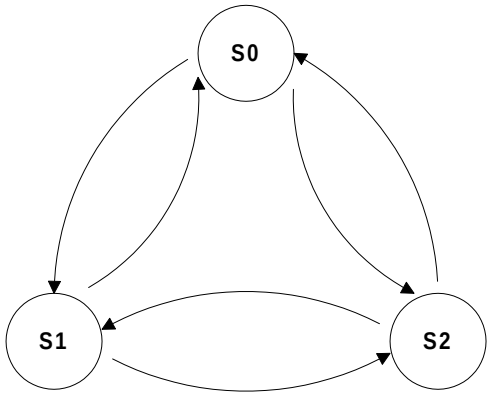

Figure 5.1: Hidden Markov Model State Diagram

HMM characterizes a system of N states. The state transition process is a

Markov process, that is, the probability of transition is only a function of the current state. However, this state sequence can not be observed. Hence, this signal model is called a hidden Markov model. A signature vector will be generated at each state following some probability distribution, and these signature vectors are our observations. Each HMM is characterized by three parameters: A, B, and π. For convenience, we use the compact notation $\lambda = (A, B, \pi)$.

a) The state transition probability $A = \{a_{ij}\}$ where

$$a_{ij} = P\left[q_{t+1} = S_j | q_t = S_i\right] \qquad 1 \leq i, j \leq N \tag{5.3}$$

b) The observation signature probability distribution in state j, $B = \{b_j(k)\}$, where

$$b_j(k) = P\left[v_k \text{ at t} | q_t = S_j\right] \qquad \begin{aligned} 1 &\leq j \leq N \\ 1 &\leq k \leq M \end{aligned} \tag{5.4}$$

The above definition assumes the signature vectors are characterized as discrete symbols chosen from a finite alphabet. However, the signature vectors can be continuous signals. While it is possible to quantize these continuous signals via a codebook, there may be a serious degradation associated with such quantization. As a result, it may be advantageous to use a HMM with continuous signature vector densities. However, the penalty we pay for using continuous

signature vector densities is higher computational complexity.

A common representation is a finite mixture of the form

$$b_j(O) = \sum_{m=1}^{M} c_{jm}\Pi[O, \mu_{jm}, U_{jm}], \qquad 1 \leq j \leq N \tag{5.5}$$

where O is the vector being modeled, c_{jm} is the mixture coefficient for the mth mixture in state j and Π is a density with mean vector μ_{jm} and covariance matrix U_{jm} for the mth mixture component in state j. In general, a Gaussian density is used for Π.

c) The initial state distribution $\pi = \{\pi_i\}$ where

$$\pi_i = P\left[q_1 = S_i\right] \qquad 1 \leq i \leq N \tag{5.6}$$

Given the form of HMM stated, there are two problems of interest that must be solved for the model to be useful in our application. These problems are the following:

Problem 1: Given the signature vector sequence $O = O_1 O_2 \ldots O_T$, and a model characterized by $\lambda = (A, B, \pi)$, how do we efficiently compute $P(O|\lambda)$, the probability of the observation sequence, given the model. This is the *classification* problem. That is, given a sequence of observations and several HMM models, we want to calculate the conditional probability of each model and classify the event using some criterion, *e.g.* Bayes Criterion.

Problem 2: Given the observation sequence and the event, how do we adjust

the model parameters $\lambda = (A, B, \pi)$ to maximize $P(O|\lambda)$? This is the *training* problem. HMM based pattern recognition is not a blind algorithm. Training is essential for successful classification.

Rabiner reviews algorithms for these two problems in his tutorial paper [36]. For the *classification* problem, it can be solved using either a forward or backward procedure. A summary of the forward procedure is as follows:

Consider the forward variable $\alpha_t(i)$ defined as

$$\alpha_t(i) = P(O_1 O_2 \cdots O_t, q_t = S_i|\lambda) \tag{5.7}$$

i.e., the probability of the partial observation sequence, $O_1 O_2 \ldots O_t$, (until time t) and state S_i at time t, given the model λ. We can solve for $\alpha_t(i)$ inductively, as follows:

1) Initialization:

$$\alpha_1(i) = \pi_i b_i(O_1) \qquad\qquad 1 \leq i \leq N \tag{5.8}$$

2) Induction:

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}) \qquad\qquad \begin{aligned} 1 &\leq t \leq T-1 \\ 1 &\leq j \leq N \end{aligned} \tag{5.9}$$

3) Termination:

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{5.10}$$

On the other hand, the backward procedure is based on the backward variable $\beta_t(i)$ defined as

$$\beta_t(i) = P(O_{t+1}O_{t+2}\cdots O_T|q_t = S_i, \lambda) \tag{5.11}$$

i.e., the probability of the partial observation sequence from $t+1$ to the end, given state S_i at time t and the model λ. Again, $\beta_t(i)$ can be solved inductively as follows:

1) Initialization:

$$\beta_T(i) = 1 \qquad\qquad 1 \le i \le N \tag{5.12}$$

2) Induction:

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j) \tag{5.13}$$
$$t = T-1, T-2, \cdots, 1, 1 \le i \le N$$

3) Termination:

$$P(O|\lambda) = \sum_{i=1}^{N} \beta_1(i) b_i(O_1) \pi_i \tag{5.14}$$

The computational complexities of the forward procedure and the backward procedure are comparable. Either one can be used to calculate the probability of a signature vector sequence. Given a sequence of signature vectors $\{O\}$ and a set of HMM characterized by $\{\lambda_l\}$, classification is done by choosing the HMM with the highest probability measure.

$$\text{Event Class} = \text{argmax}_l P(O|\lambda_l) \tag{5.15}$$

For the *training* problem, an iterative procedure known as the Baum-Welch method is chosen. In order to describe the procedure for reestimation (iterative update and improvement) of HMM parameters, we need to define

$$\gamma_t(i) = P(q_t = S_i|O, \lambda) \tag{5.16}$$

i.e., the probability of being in state S_i at time t, given the observation sequence O, and the model λ. In addition, we define

$$\xi_t(i, j) = P(q_t = S_i, q_{t+1} = S_j|O, \lambda) \tag{5.17}$$

i.e., the probability of being in state S_i at time t, and state S_j at time t+1, given the model and the observation sequence. Both $\gamma_t(i)$ and $\xi_t(i, j)$ can be expressed in forward and backward variables,

$$\gamma_t(i) = \frac{\alpha_t(i)\beta_t(i)}{\sum_{i=1}^{N} \alpha_t(i)\beta_t(i)} \tag{5.18}$$

$$\xi_t(i,j) = \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{P(O|\lambda)} \tag{5.19}$$

Furthermore, we can relate $\lambda_t(i)$ and $\xi_t(i,j)$ with the following expression:

$$\gamma_t(i) = \sum_{j=1}^{N} \xi_t(i,j) \tag{5.20}$$

Using the above definitions (and the concept of counting event occurrences) we can get a method of reestimation of the parameters of an HMM.

$$\bar{\pi}_i = \text{expected number of times in state } S_i \text{ at time } (t=1) = \gamma_1(i) \tag{5.21}$$

$$\begin{aligned} \bar{a}_{ij} &= \frac{\text{expected number of transitions from state } S_i \text{ to state } S_j}{\text{expected number of transitions from state } S_i} \\ &= \frac{\sum_{t=1}^{T-1} \xi_t(i,j)}{\sum_{t=1}^{T-1} \gamma_t(i)} \end{aligned} \tag{5.22}$$

$$\begin{aligned} \bar{b}_j(k) &= \frac{\text{expected number of times in state } S_j \text{ and observing symbol } v_k}{\text{expected number of times in state } S_j} \\ &= \frac{\sum_{t=1\ s.t.\ O_t=v_k}^{T} \gamma_t(j)}{\sum_{t=1}^{T} \gamma_t(j)} \end{aligned} \tag{5.23}$$

The above reestimation equations are for a HMM with finite observation alphabet. For a HMM using a continuous observation density, we need to reestimate the coefficients of the mixture density, namely, c_{jk}, μ_{jk}, and U_{jk}.

$$\bar{c}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k)}{\sum_{t=1}^{T} \sum_{k=1}^{M} \gamma_t(j,k)} \tag{5.24}$$

$$\bar{\mu}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot O_t}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{5.25}$$

$$\bar{U}_{jk} = \frac{\sum_{t=1}^{T} \gamma_t(j,k) \cdot (O_t - \mu_{jk})(O_t - \mu_{jk})'}{\sum_{t=1}^{T} \gamma_t(j,k)} \tag{5.26}$$

where $\gamma_t(j,k)$ is the probability of being in state j at time t with the kth mixture component accounting for O_t, *i.e.*,

$$\gamma_t(j,k) = \left[\frac{\alpha_t(j)\beta_t(j)}{\sum_{j=1}^{N} \alpha_t(j)\beta_t(j)} \right] \left[\frac{c_{jk}\Pi(O_t, \mu_{jk}, U_{jk})}{b_j(O)} \right] \tag{5.27}$$

The term $\gamma_t(j,k)$ is the generalization of $\gamma_t(j)$ in Eq. 5.16. The reestimations for a_{ij} and π_i are identical to those used for a finite alphabet.

In theory, the reestimation equations should give values of the HMM parameters which correspond to a local maximum of the likelihood function. A key question is therefore how do we choose initial estimates of the HMM parameters so that the local maximum is the global maximum of the likelihood function. Experiences have shown that uniform initial estimates of the π and A parameters is adequate for giving useful reestimates of these parameters in almost all cases. However, for the B parameters, experience has shown that good initial estimates

are essential for reestimation process. There are many ways for obtaining this initial estimates. A estimation procedure called *Segmental k-Means Segmentation* is used for our application.

The *Segmental k-Means Segmentation* procedure can be illustrated by the flow chart shown in Figure 5.2 [36]. We assume we have a training set of observations (the same as is required for parameter reestimation), and an initial estimate of all model parameters. However, unlike the one required for reestimation, the initial model estimate can be chosen randomly, or on the basis of any available model which is appropriate to the data.

Following model initialization, the set of training observation sequences is segmented into states, based on the current model λ. This segmentation is achieved by finding the optimum state sequence, via the Viterbi algorithm, and then backtracking along the optimal path. The result of segmenting each of the training sequences is, for each of the N states, a maximum likelihood estimate of the set of the observations that occur within each state S_i according to the current model.

In the case where we are using discrete symbol densities, each of the observation vectors within a state is coded using the M-codeword codebook, and the updated estimate of the $b_j(k)$ parameters is

$$\hat{b}_j(k) = \text{number of vectors with codebook index } k \text{ in state } j \text{ divided by the number of vectors in state } j.$$

In the case where we are using continuous observation densities, a segmental K-means procedure is used to cluster the observation vectors within each state S_j into a set of M clusters (using a Euclidean distortion measure), where each clusters represents one of the M mixtures of the $b_j(O_t)$ density. From the clustering, an updated set of model parameters is derived as follows:

$\hat{c}_{jm} =$ number of vectors classified in cluster m of state j divided the numbers of vectors in state j

$\hat{\mu}_{jm} =$ sample mean of the vectors classified in cluster m of state j

$\hat{U}_{jm} =$ sample covariance matrix of the vectors classified in cluster m of state j

Based on this state segmentation, updated estimates of the a_{ij} coefficients can be obtained by counting the number of transitions from state i to j and dividing it by the number of transitions from state i to any state (including itself).

An updated model $\hat{\lambda}$ is obtained from the new model parameters and the formal reestimation procedure is used to reestimate all model parameters. The resulting model is then compared to the previous model (by computing a distance score that reflects the statistical similarity of the HMMs). If the model distance score exceeds a threshold, then the old model λ is replaced by the new (reestimated) model $\bar{\lambda}$, and the overall training loop is repeated. If the model distance score falls below the threshold, then model convergence is assumed and the final model parameters are saved.

5.2 Simulation Results

We conclude our discussion on HMM based pattern recognition processor by giving a set of performance results (in terms of probability of correct classification and probability of error classification) on the task of recognizing a set of vehicles based on data recorded by a microphone.

The set of vehicle data is provided by US Army Research Laboratory. There are four types of ground vehicles in this data set, namely, 1) heavy tracked vehicle (4 vehicles), 2) light tracked vehicle (1 vehicle), 3) heavy wheeled vehicle (2 vehicles), and 4) light wheeled vehicle (2 vehicles). The set of test data is gathered from four different test sites:

1) **Site a**: Desert like similar to conditions in the desert southwest in the summer (typically dry and very hot).

2) **Site b**: Arctic like similar to conditions in Alaska in the winter (typically cold, windy, with and without snow cover).

3) **Site c**: Normal, similar to conditions in Mid-Atlantic states in the spring and fall (typically mild, with and without humidity).

4) **Site d**: Arctic like similar to conditions in Alaska in the winter but different from Site b (typically cold, windy, with and without snow cover).

The ground vehicles were traveling at constant speeds from one direction toward the microphone passing the closest point of approach (CPA) and then away from the microphone. The CPA to the microphone varies from 25 m to

100 m. The speed of the ground vehicle varied from 5 km/hr to 40 km/hr. The microphone data is low-pass filtered at 400 Hz via a 6th order filter to prevent spectral aliasing and high-pass filtered at 25 Hz via a 1st order filter to reduce wind noise. The data is digitized by a 16-bit A/D at the rate of 1025.641 Hz.

In the first set of simulations, each microphone data set is treated as an isolated measurement. There is no source separation and deconvolution preprocessing in this set of simulations. Approximately half of the vehicle runs are used for training to build the HMM model. The training set for each vehicle consists of about half the vehicle runs at four different sites.

Vehicle Type	Number of Vehicle Runs	Number of Vehicle Runs used for Training
Heavy Tracked Vehicle	339	189
Light Tracked Vehicle	69	36
Heavy Wheeled Vehicle	100	52
Light Wheeled Vehicle	105	57

All vehicle run data is first processed by an event detection processor to extract the proper data segmentation for classification. The event detection processor estimates the background noise floor and extracts data segments that exceed the preset SNR ratio. The event detection processor ensures the classification is performed on data with high SNR. All selected data are run through a LPC

based feature extraction processor with the following parameters:

p, number of autocorrelation coefficients used = 6

Q, number of cepstral coefficients generated = 9

K, window for cepstral derivative calculation = 2

The window size is chosen such that the frequency spectrum of the sensor data is relatively stationary within the window. Two window sizes are chosen for our simulations. The window spacing is chosen to be one third of the window size following the speech signal processing guideline. The number of autocorrelation coefficients used, the number of cepstral coefficients generated, and the window for cepstral derivative calculation are also chosen from experiences in speech signal processing. After the feature extraction processor, the signature vectors in the training data set are used to build HMM models with the following parameters (HMM with continuous signature vector density is used):

N_s, number of states in the model = 3

N_m, number of mixtures for each state = 3

After HMM models are built, all vehicle runs are classified via the Bayes criterion. The classification results are summarized as follows:

N, window size = 3000 samples

M, window spacing = 1000 samples

Vehicle Type	Number of Vehicle Runs correctly Classified	Probability of Detection
Heavy Tracked Vehicle	330	95.6%
Light Tracked Vehicle	65	92.8%
Heavy Wheeled Vehicle	97	97.0%
Light Wheeled Vehicle	100	94.3%

N, window size = 4500 samples

M, window spacing = 1500 samples

Vehicle Type	Number of Vehicle Runs correctly Classified	Probability of Detection
Heavy Tracked Vehicle	330	97.9%
Light Tracked Vehicle	65	86.6%
Heavy Wheeled Vehicle	97	97.0%
Light Wheeled Vehicle	100	93.2%

In this simulation, we notice that the performance of our classifier varies as a function of the window size. In particular, the classification performance improves for heavy vehicles using longer window size. On the other hand, the classification performance degrades for light vehicles using longer window size. In general, signatures recorded for heavy vehicles are slow changing while signatures recorded for light vehicles are quite spontaneous. As a result, the classifier using

longer window size yields a better result for heavy vehicles than light vehicles. However, the optimal window size for mixtures of targets is an open problem that requires further research.

In the second set of simulation, we consider three microphones in the array jointly. In this case, the microphone array data is processed by all three sub-processors in our architecture. The acoustic data is first processed by the source separation and deconvolution processor. The pre-processed data is then fed into the feature extraction processor and the pattern recognition processor for further processing. The following parameters are used for our simulation:

Source Separation and Deconvolution Processor

N_d, number of taps used in the deconvolution filter = 6

N_o, order of the system model = 2

N_t, number of sources = 1

Feature Extraction Processor

N, window size = 3000

M, window spacing = 1000

p, number of autocorrelation coefficients used = 6

Q, number of cepstral coefficients generated = 9

K, window for cepstral derivative calculation = 2

Pattern Recognition Processor

N_s, number of states in the model = 3

N_m, number of mixtures for each state = 3

The classification performance of our system is summarized as follows:

Vehicle Type	Number of Vehicle Runs correctly Classified	Probability of Detection
Heavy Tracked Vehicle	330	96.46%
Light Tracked Vehicle	65	91.30%
Heavy Wheeled Vehicle	97	97.06%
Light Wheeled Vehicle	100	91.43%

In this simulation, we demonstrate the capability of our target identification processor. As shown by our simulation results, we improve the detection performance for heavy vehicles by adding the source separation and deconvolution processor. On the other hand, the detection performance for light vehicles degrades slightly compared to the case without the source separation and deconvolution processor. The source separation and deconvolution processing in our simulation is performed on each frame of size 3000 samples. As we have pointed out previously, acoustic data from light vehicles change quite rapidly. The statistical estimates may not be good enough since the window size is too big. On the other hand, the blind deconvolution algorithm requires a large data set for accurate estimate in general. In order not to increase frame size, we have to increase the

number of sensors in the array. We would expect a more dramatic improvement for seismic data with deconvolution, since the acoustic waves undergo relatively little dispersion. Unfortunately, we lack access to large array data sets to be able to rigorously test this.

5.3 Summary

In conclusion, the HMM based pattern recognition processor has performed quite well in target classification as shown by our simulations. From the simulation, we demonstrate the performance of target identification processor in the event of single target. Furthermore, we show that the cepstral coefficient based feature extraction processor is a good choice for our application. While the preliminary result using the HMM based processor shows encouraging results, the applicability of the HMM model to the general target recognition problem is yet to be demonstrated with a larger set of observation data.

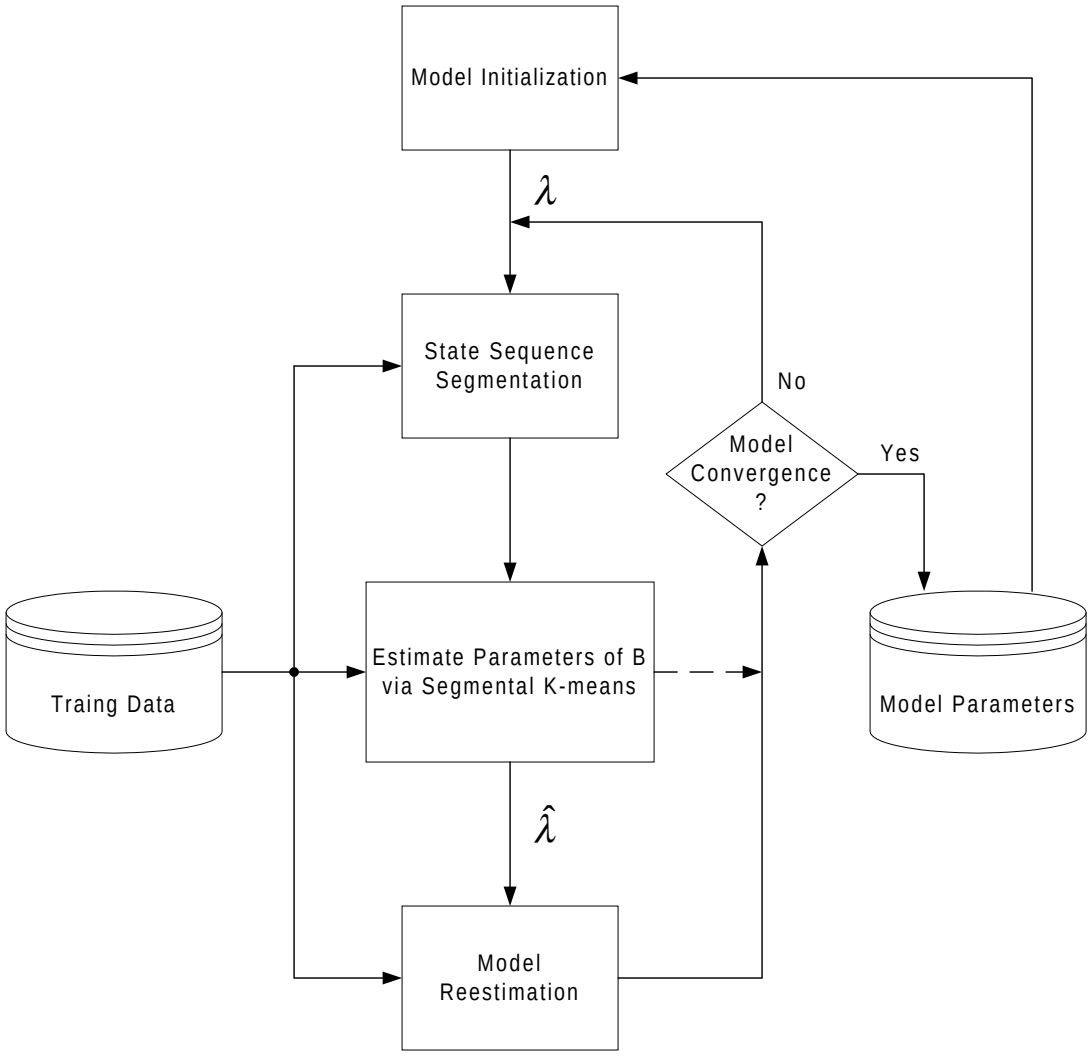

Figure 5.2: The segmental k-means training procedure used to estimate parameter values for the optimal continuous mixture density fit to a finite number of observation sequences

Chapter 6

Conclusion

Remote sensing has applications in many different areas. Target identification is one key objective that we want to achieve in a remote sensing system. In this research, we propose a multi-processor architecture for target identification in the wireless sensor network. This multi-processor architecture facilitates the design of this subsystem since each processor can be designed and optimized individually. In particular, a new source separation and deconvolution algorithm is proposed for the source separation and deconvolution processor in Chapter 3. The performance of our algorithm has been verified by Monte Carlo simulations. While this algorithm is proposed for the source separation in the target detection, it can be applied to many other fields, *e.g.*, communication. In the communication system, channel equalization is normally started with a training sequence. With the blind equalization algorithm, we can start equalization without the training

sequence which saves bandwidth for the system. Source separation can be useful for the communication system as well. With the source separation algorithm, we can separate multiple sources in the same channel which increases the effective bandwidth of the system. Further research is in progress on applications of blind source separation and deconvolution algorithm to communication systems.

The particular system explored in Chapter 3 is a system of a known number of targets and system order. However, this may not be the case in a real life application. As a result, the number of targets and system order need to be estimated. The performance of our algorithm using this estimated information must be investigated for real life deployment.

In Chapter 4 and 5, we presented the architectures for the feature extraction processor and the pattern recognition processor. Algorithms for the feature extraction processor and the pattern recognition processor are heavily influenced by algorithms used in speech recognition. Using the field measurement data, we are able to achieve excellent classification results with our target identification processor under a single target scenario. The single target classification limitation is caused by the lack of measurement data of multiple targets. The multiple targets classification problem will be researched further in the future. While these algorithms give us good results for target identification, they still may not be optimal. More studies are needed in that area. In particular, the wavelet transform can be a potential algorithm for feature extraction. The key problem of the

wavelet transform based feature extractor is searching for the optimal mother wavelet. The hidden Markov model is chosen for pattern recognition processor because of deep understanding of the model. However, there are some inherent limitations of this type of statistical model for our problem. A major limitation is the assumption that successive observations are independent, so that we can then apply the Markov chain model. More research on the pattern recognition processor can include searching for a general model for the hidden state transition process.

As the semiconductor technology advances, we can build more sensors and more complex processors in a sensor node. However, we do not want to overwhelm ourselves with the huge amount of information gathered by these sensors. We want to learn particular things about the physical world, and ignore the rest. The target identification processor we are studying is one of the many processes that convert information to intelligence. Complicated information analysis can be simplified or eliminated. Consequently, more resources can be devoted to achieve better decision making.

Chapter 7

Appendix A - Proof of Lemma 1

Lemma 1: The N_t unit-norm eigenvectors associated with the smallest N_t eigenvalues of matrix Q form the basis for $h_j, j = 1, \ldots, N_t$.

We shall prove this Lemma through proving a series of claims.

Claim 1: Let H be a full-column rank matrix and $H = [K_1, \ldots, K_{N_t}]$, then $\{K_i, i = 1, \ldots, N_t\}$ is a set of linearly independent matrices.

Proof: By contradiction, if $\{K_i, i = 1, \ldots, N_t\}$ is not a set of linearly independent matrices, then there exist K_i such that

$$K_i = \sum_{k \neq i} a_k K_k \tag{7.1}$$

This implies columns of K_i can be expressed as a linear combination of the other columns. The same thing can be said about H. Thus, H is not a full-column rank matrix, which is a contradiction.

Claim 2: Let $\{K_i, i = 1, \ldots, N_t\}$ be a set of linearly independent matrices where $K_i = [H_{1i}^T, \ldots, H_{N_s i}^T]^T$, and $h_i = [h_{1i}^{(0)}, \ldots, h_{1i}^{(M)}, \ldots, h_{N_s i}^{(0)}, \ldots, h_{N_s i}^{(M)}]^T$ (dim. $N_s(M+1)$ x 1); $\{h_i, i = 1, \ldots, N_t\}$ is a set of linearly independent vectors.

Proof: By contradiction. Similar to the proof of Claim 1.

Claim 3: Let $\{h_i, i = 1, \ldots, N_t\}$ be a set of linearly independent vectors and $\{r_j, j = 1, \ldots, N_t\}$ be the unit-norm basis for $\{h_i\}$. If $q(h_i) = h_i^T Q h_i = 0$ for all i, then q$(r_j) = r_j^T Q r_j = 0$ for all j.

Proof: since $\{r_j\}$ is the basis for $\{h_i\}$, any r_j can be expressed as a linear combination of $\{h_i\}$, that is:

$$\begin{aligned} r_j &= \sum_{i=1}^{N_t} a_{ji} h_i \\ q(r_j) &= q\left(\sum_{i=1}^{N_t} a_{ji} h_i\right) = \sum_{i=1}^{N_t} a_{ji} q(h_i) = 0 \end{aligned} \tag{7.2}$$

Searching for N_t orthonormal vectors $\{r_j\}$ such that $q(r_j) = r_j^T Q r_j = 0$ is equivalent to solving the N_t eigenvectors in the null subspace of Q. However, in the presence of noise, the searching process is equivalent to solving the N_t eigenvectors associated with the smallest N_t eigenvalues of Q. This complete the proof.

Bibliography

[1] T. Yu, D. Chen, G.J. Pottie, and K. Yao, "Blind Decorrelation and Deconvolution Algorithm for Multiple-Input Multiple-Output System I: theorem derivation", Proceedings of SPIE, vol. 3807, 1999.

[2] T. Yu, D. Chen G.J. Pottie, and K. Yao, "Blind Decorrelation and Deconvolution Algorithm for Multiple-Input Multiple-Output System II: simulation results and performance analysis", Proceedings of SPIE, vol. 3807, 1999.

[3] K. Aki and P.G. Richards, *Quantitative seismology: theory and methods.* San Francisco:W. H. Freeman, 1980.

[4] J.E. White, *Seismic waves: radiation, transmission, and attenuation.* New York:McGraw-Hill, 1965.

[5] P. Malischewsky, *Surface waves and discontinuities.* Amsterdam:Elsevier, 1987.

[6] V.I. Keilis-Borok, *Seismic surface waves in a laterally inhomogeneous earth.* Dordrecht:Kluwer Academic Publishers, 1989.

[7] J.M. Mendel, *Maximum-likelihood deconvolution: a journey into model-based signal processing.* New York:Springer-Verlag, 1990.

[8] L. Rabiner and B. Juang, *Fundamentals of speech recognition.* Englewood Cliffs:Prentice Hall PTR, 1993.

[9] D.P. Bertsekas, *Constrained optimization and Lagrange multiplier methods.* New York:Academic Press, 1982.

[10] M. Vetterli and J. Kovacevic, *Wavelets and subband coding.* Englewood Cliffs:Prentice Hall PTR, 1995.

[11] J.M. Mendel, *Optimal seismic deconvolution: an estimation-based approach.* New York:Academic Publishers, 1983.

[12] L. Ljung, *System Identification: Theory for the User.* Englewood Cliffs, NJ:Prentice-Hall, 1987.

[13] A.D. Whalen, *Detection of Signals in Noise.* Academic Press, Inc., 1971

[14] Z. Chair and P.K. Varshney, "Optimal data fusion in multiple sensor detection systems," *IEEE Trans. Aerospace and Electronic Systems*, vol. 22, pp. 98-101, 1986.

[15] E.W. Libby and P.S. Maybeck, "Sequence comparison techniques for multisensor data fusion and target recognition," *IEEE Trans. Aerospace and Electronic Systems*, vol. 32, pp. 52-65, 1996.

[16] Z.Q. Luo, J.N. Tsitsiklis, "Data fusion with minimal communication," *IEEE Trans. Inform. Theory*, vol. 40, pp. 1551-1563, 1994.

[17] E. Weinstein, A.V. Oppenheim, M. Feder, and J.R. Buck, "Iterative and sequential algorithms for multisensor signal enhancement," *IEEE Trans. Signal Processing*, vol. 42, pp. 846-859, 1994.

[18] M.C. Dogan and J.M. Mendel, "Blind deconvolution (equalization): Some new results," *Signal Processing*, vol. 53, pp. 109-116, 1996.

[19] M.C. Dogan and J.M. Mendel, "Applications of cumulants to array processing. II. Non-Gaussian noise suppression," *IEEE Trans. Signals Processing*, vol. 43, pp. 1663-1676, 1995.

[20] M.C. Dogan and J.M. Mendel, "Applications of cumulants to array processing. I. Aperture extension and array calibration," *IEEE Trans. Signals Processing*, vol. 43, pp. 1200-1216, 1995.

[21] M.C. Dogan and J.M. Mendel, "Cumulant-based blind optimum beamforming," *IEEE Trans. Aerospace and Electronic Systems*, vol. 30, pp. 722-741, 1994.

[22] M.C. Dogan and J.M. Mendel, "Single sensor detection and classification of multiple sources by higher-order spectra," *IEE Proceedings F (Radar and Signal Processing)*, vol. 140, pp. 350-355, 1993.

[23] E. Moulines, P. Duhamel, J. Cardoso, and S. Mayrargue, "Subspace Methods for the Blind Identification of Multichannel FIR Filters," *IEEE Trans. Signal Processing*, vol. 43, pp. 516-525, 1995.

[24] M. Gureli and C. Nikias, "EVAM: An Eigenvector-Based Algorithm for Multichannel Blind Deconvolution of Input Colored Signals," *IEEE Trans. Signal Processing*, vol. 43, pp. 134-149, 1995.

[25] M. Kristensson and B. Ottersten, "A Statistical Approach to Subspace Based Blind Identification," *IEEE Trans. Signal Processing*, vol. 46, pp. 1612-1623, 1998.

[26] K. Abed-Meraim, J. Cardoso, A. Gorokhov, P. Loubation, and E. Moulines, "On Subspace Methods for Blind Identification of Single-Input Multiple-Output FIR Systems," *IEEE Trans. Signal Processing*, vol. 45, pp. 42-55, 1997

[27] E. Weinstein, M. Feder, and A. Oppenheim, "Multi-Channel Signal Separation by Decorrelation," *IEEE Trans. Speech and Audio Processing*, vol. 1, pp. 405-413, 1993.

[28] D. Yellin and E. Weinstein, "Criteria for Multichannel Signal Separation," *IEEE Trans. Signal Processing*, vol. 42, pp. 2158-2168, 1994.

[29] D. Yellin and E. Weinstein, "Multichannel Signal Separation: Methods and Analysis," *IEEE Trans. Signal Processing*, vol. 44. pp. 106-118, 1996.

[30] L. Molgedey and H.G. Schuster, "Separation of a Mixture of Independent Signals Using Time Delayed Correlations," *Physical Review Letters*, vol. 72, pp. 3634-3637, 1994.

[31] O. Shalvi and E. Weinstein, "Super-Exponential Methods for Blind Deconvolution," *IEEE Trans. Inform. Theory*, vol. 39, pp. 504-519, 1993.

[32] L. Tong, G. Xu, and T. Kailath, "Blind Identification and Equalization Based on Second-Order Statistics: A Time Domain Approach," *IEEE Trans. Inform. Theory*, vol. 40, pp. 340-349, 1994.

[33] J. Yang and M. Kaveh, "Adaptive Eigensubspace Algorithms for Direction or Frequency Estimation and Tracking," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 36, pp. 241-251, 1988.

[34] Y. Hua, "Fast Maximum Likelihood for Blind Identification of Multiple FIR Channels," *IEEE Trans. Signal Processing*, vol. 44, pp. 661-672, 1996.

[35] B. Yang, "Projection Approximation Subspace Tracking," *IEEE Trans. Signal Processing*, vol. 43, pp. 95-107, 1995.

[36] L. Rabiner, "A Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition," *Proc. IEEE*, vol. 77, pp. 257-286, 1989.

[37] B. Juang and L. Rabiner, "The Segmental K-Means Algorithm for Estimating Parameters of Hidden Markov Models," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 38, pp. 1639-1641, 1990.

[38] A.V. Oppenheim, E. Weinstein, K.C. Zangi, M. Feder, and D. Gauger, "Single-Sensor Active Noise Cancellation," *IEEE Trans. Speech, Audio Processing*, vol. 2, pp. 285-290, 1994.

[39] M.J. Levin, "Estimation of a System Pulse Transfer Function in the Presence of Noise," *IEEE Trans. Automatic Control*, pp. 229-235, 1964.

[40] R. Roy and T. Kailath, "ESPRIT - Estimation of Signal Parameters Via Rotational Invariance Techniques," *IEEE Trans. Acoust., Speech, Signal Processing*, vol. 37, pp. 984-995, 1989.

[41] E. Gonen and J.M. Mendel, "Applications of Cumulants to Array Processing - Part III: Blind Beamforming for Coherent Signals," *IEEE Trans. Signal Processing*, vol. 45, pp. 2252-2264, 1997.

[42] E. Gonen, J.M. Mendel, and M.C. Dogan, "Applications of Cumulants to Array Processing - Part IV: Direction Finding in Coherent Signals Case," *IEEE Trans. Signal Processing*, vol. 45, pp. 2265-2276, 1997.

[43] J.M. Mendel, "Tutorial on Higher-Order Statistics (Spectra) in Signal Processing and System Theory: Theoretical Results and Some Applications," *Proc. IEEE*, vol. 79, pp. 278-305, 1991.

[44] S. Haykin, B.W. Currie, and S.B. Kesler, "Maximum-Entropy Spectral Analysis of Radar Clutter," *Proc. IEEE*, vol. 70, pp. 953-962, 1982.

[45] T.K. Moon, "The Expectation-Maximization Algorithm," *IEEE Signal Processing Magazine*, pp. 47-60, 1996.

[46] S. Douglas and A. Cichocki, "Neural Networks for Blind Decorrelation of Signals", *IEEE Trans. Signal Processing*, vol. 45, pp. 2829-2842, 1997.

[47] H. Wu and J. Principe, "A Unifying Criterion for Blind Source Separation and Decorrelation: Simultaneous Diagonalization of Correlation Matrices", *Proceedings of the 1997 IEEE Signal Processing Society Workshop*, pp. 496-505, 1997.

[48] K. Matsuoka and M. Kawamoto, "A Neural Net for Blind Separation of Nonstationary Signal Sources", *1994 IEEE International Conference on Neural Networks*, pp. 221-232, 1994.

[49] K. Diamantaras, "Asymmetric PCA Neural Networks for Adaptive Blind Source Separation", *Proceedings of the 1998 IEEE Signal Processing Society Workshop*, pp. 103-112, 1998.

[50] A. Gorokhov and P. Loubaton, "Subspace-Based Techniques for Blind Separation of Convolutive Mixtures with Temporally Correlated Sources", *IEEE Trans. Circuits and Systems-I: Fundamental Theory and Applications*, vol. 44, pp. 813-820, 1997.

[51] K. Abed-Meraim and Y. Hua, "Blind Identification of Multi-Input Multi-Output System Using Minimum Noise Subspace", *IEEE Trans. Signal Processing*, vol. 45, pp. 254-258, 1997.

[52] K. Abed-Meraim, P. Loubaton, and E. Moulines, "A Subspace Algorithm for Certain Blind Identification Problems", *IEEE Trans. Information Theory*, vol. 43, pp. 499-511, 1997.

[53] J.F. Scholl, J.R. Agre, L.P. Clare, and M.C. Gill "A Low Power Impulse Signal Classifier Using the Haar Wavelet Transform", *Proceedings of the SPIE International Symposium on Enabling Technologies for Law Enforcement and Security*, 1998.

[54] J.F. Scholl, J.R. Agre, and L.P. Clare "Wavelet Packet Based Target Classification Schemes", *Proceedings fo the 1998 Meeting of the IRIS Specialty Group on Acoustic and Seismic Sensing*, 1998.

[55] J.R. Lundien and H. Nikodem "A Mathematical Model for Predicting Microseismic Signals in Terrain Materials", *Tehnical Report M-73-4*, U.S. Army Engineer Waterways Experiment Station, 1973.