BLIND ESTIMATION OF FIR CHANNELS USING SPATIAL SEPARATION

Y M SASIRI S YAPA

NATIONAL UNIVERSITY OF SINGAPORE

2004

BLIND ESTIMATION OF FIR CHANNELS USING SPATIAL SEPARATION

Y M SASIRI S YAPA

(*BSc. Eng., University of Moratuwa, Sri Lanka*)

A THESIS SUBMITTED

FOR THE DEGREE OF MASTER OF ENGINEERING

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2004

Acknowledgement

I would like to take this opportunity to express my warmest thanks to many who have contributed to the production of this thesis. Without their support, this thesis could not have been written.

I am deeply indebted to my supervisors Dr. A. Rahim Leyamn and Prof. Tjhung Tjeng Thiang, whose help, stimulating suggestions, supervision, creative advice and encouragement helped ignite and refine the ideas that is this thesis.

My appreciation also goes to my parents and family, who were always there for me, and supported me in all my decisions.

I would also like to thank the Electrical and Computer Engineering Department at NUS and the A*STAR Institute for Infocomm Laboratories for giving me the opportunity, and providing a congenial environment conducive to my research.

Lastly, but not least I would like to thank all my friends who made my stay in Singapore enjoyable.

Contents

List of Figures

List of Tables

Abbreviations

VLF:	Very Low Frequency
SHF:	Super High Frequency
LOS:	Line Of Sight
T-R:	Transmitter - Receiver separation
ISI:	Inter Symbol Interference
DFE:	Decision Feedback Equalizer
TDL:	Tap Delay Line
ZF:	Zero Forcing
MMSE:	Minimum Mean Square Error
GSM:	Global System Mobile
HOS:	Higher Order Statistics
SOS:	Second Order Statistics
SISO:	Single Input Single Output
HMM:	Hidden Markov Model
SIMO:	Single Input Multiple Output
FA:	Finite Alphabet
BPSK:	Binary Phase Shift Keying
QPSK:	Quadrature Phase Shift Keying
QAM:	Quadrature Amplitude Modulation
SNR:	Signal to Noise Ratio
CR:	Cross Relation method
LSS:	Least Squares Smoothing
PAM:	Pulse Amplitude Modulation
DSPK:	Differential Phase Shift Keying
ILSP:	Iterative Least Square with Projection algorithm
VA:	Viterbi Algorithm
EBSD:	Explicit Blind Sequence Detection
IBSD:	Implicit Blind Sequence Detection

ML:	Maximum Likelihood
MAP:	Maximum A Posterior
VA:	Viterbi Algorithm
EBSD:	Explicit Blind Sequence Detection
LSE:	Least Significant Elements
CBHS:	Cost Based Heuristic Search
CEDS:	Channel Estimation By Difference Sets
CETI:	Channel Estimation by Twin Indexing
SDSE:	Sequence Driven Symbol Estimation
MIMO:	Multiple Input Multiple Output

Notations

Basic Elements:

$\mathbf{M}$	A Matrix
$\mathbf{v}$	A Vector
a	A Scalar
S	A Set
$func()$	A Function
$\delta()$	The Delta Function
$\Gamma()$	The Gamma Function Function

Notations Used

$\mathbf{M}'$ or $\mathbf{v}'$	Transposition
$\mathbf{M}^{\dagger}$	Inverse or pseudo inverse of a matrix
$\{a\}$	An element
$*$	Convolution operator
$abs[]$	Absolute value
$sgn[]$	Signum function
$sum[]$	Summation
$max[]$	Maximum
$min[]$	Minimum
$Diag[\mathbf{v}]$	A Matrix with $\mathbf{v}$ as diagonal
$[\mathbf{v}]_i$	The i^{th} element of $\mathbf{v}$
$[\mathbf{M}]_{ij}$	The element at the indices (i,j) of the matrix $\mathbf{M}$
$\|\mathbf{v}\|$	Magnitude of the vector$\mathbf{v}$
$func_e[]$	Element by element operator of the function $func[]$

Summary

Mobile communication has become one of the fastest growing technologies of the twenty first century. However, inherent properties of the wireless media place fundamental limitations on the capacity of such mobile systems. One of the main problems faced in wireless communication is Inter Symbol Interference (ISI). Traditionally, ISI has been compensated using adaptive equalizers with training data. However, recent demand for high bandwidth has made these algorithms obsolete with more efficient blind algorithms taking their place.

In this thesis, we present a new class of deterministic blind algorithms. Instead of using only the channel structure, algorithms presented in this thesis utilize data structures that are created by the Finite Alphabet (FA) property as transmitted data is impinged onto a mobile channel. In this thesis, we examine both direct sequence estimation and blind channel estimation based on the data structures created by the FA property. We begin our thesis by first introducing and examining the structure of the data that is created. This, we label as *spatial* data in our thesis. Then, we proceed to outline two spatial tools, the *Primary* and *Secondary* clustering algorithms that are used for processing the spatial data described above.

We first present the State Driven Sequence Estimation (SDSE) algorithm,

which we have implemented for blind sequence detection. This algorithm uses the spatial structure to derive a state transition table, which when complemented by actual time data can be used to extract transmitted symbols within a sign ambiguity. Later, we present two channel estimation algorithms. Both, the Channel Estimation by Difference Sets (CEDS) and Channel Estimation by Twin Indices (CETI) utilize vectors that are generated from the spatial structure. However, the manner they utilize these vectors differ, resulting in different behaviors in the two algorithms.

Lastly we conclude our thesis, extending our work with subtle modifications thereby enabling it to include complex transmitter constellations and Multiple Input Multiple Output systems into its repertoire.

Chapter 1

Introduction

1.1 The mobile media

Wireless communication has become one of the fastest growing technologies of the twenty first century. Starting from the late 19th century, when Marconi began experimenting with the transmission and reception of "Hertzian Waves", wireless systems have evolved to become a technology capable of providing instantaneous high bandwidth links to mobile users. The current research thrust on wireless systems is concentrated on the last two aspects mentioned above: To provide a higher bandwidth to a more mobile user. The mobile media is an important consideration in designing wireless systems. Inherent properties of the wireless media place fundamental limitations on the capacity of mobile systems. The characteristics of the mobile channel are affected by the environment it encompasses. The environment results in creating a multitude of propagation modes. These modes vary from direct line of sight (LOS) to a mixture of scattered, reflected

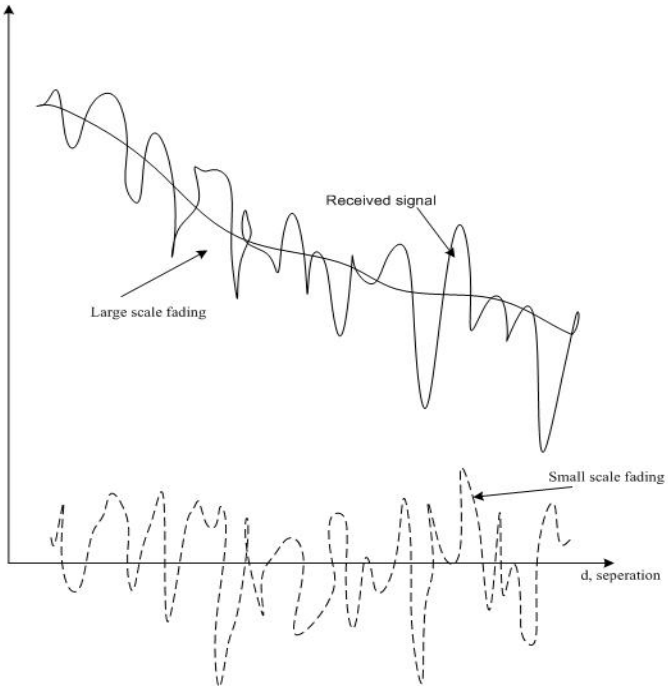

Figure 1.1: Multipath propagation

and diffracted modes depending on the clutter present within the channel. This lends to the random nature of the mobile channel, and consequently its difficulty in being modeled. Characterization of the wireless channel has been traditionally separated into two categories [1]. They are, *Large scale fading* that predicts the average signal strength for an arbitrary transmitter receiver (T-R) separation, and *small scale fading* that characterizes the rapid random fluctuations of signal strength over distances comparable to its wavelength. This is illustrated in Fig 1.1 where the T-R separation is denoted by d. Large scale fading is due to the nature of radio waves, and their modes of propagation with respect to the environment. The main components that factor into *Large scale fading* are,

- Free space path loss given by

$$PL(dB) = -10\log_{10}\left[\frac{G_t G_r \lambda^2}{(4\pi d)^2}\right] \tag{1.1}$$

  G_t and G_r are transmitter and receiver gains respectively, while λ is the the carrier wavelength.

- Ground reflections

- Diffraction due to edges such as buildings and mountains

- Scattering due to objects within the media.

In the real world, these four components interact to produce complex fading characteristics. However, with the advent of radio, television and microwave links, modeling of large scale fading became a necessity. This pushed open the door for empirical modeling, and the models proposed by Okumura [2], Hata [3] and Walfisch & Bertoni [4] provides the means to predict average signal strength across many terrains with reasonable accuracy.

1.1.1 Small scale fading and the multipath model

Small scale fading is due to the rapid, random, fluctuations of the amplitude, phase, and frequency, of a received radio signal over a time period, or distance comparable to its wavelength. It is primarily due to objects like cars, buildings and trees that clutter the mobile media. These objects cause transmitted rays with slightly different angles of departure to undergo different perturbations on

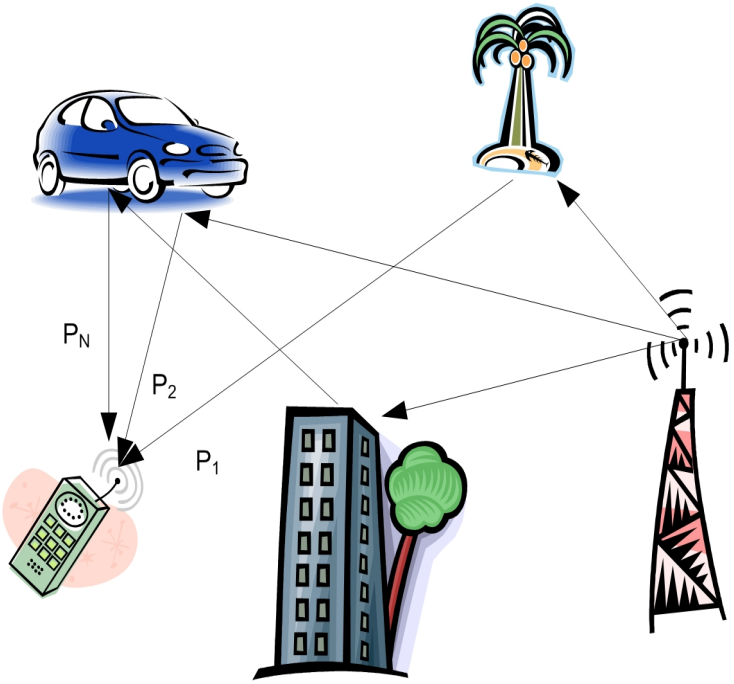

Figure 1.2: Multipath propagation

each surface they reflect, scatter, or diffract on. This results in the signals being almost completely uncorrelated by the time they incident on the receiver antenna. Furthermore, the change of the environment; swaying of trees, rain, humidity, etc, creates additional complexities by inducing temporal variations in the signals. Both effects, temporal and spatial randomness, limit the capacity of wireless systems.

Consider the multipath channel shown in Fig. 1.2. It consists of P paths, where each path $p \in \{1, ..., P\}$, is defined by its respective path length $\{\gamma_p\}$, and its attenuation coefficient $\{a_p\}$. Let $s(t)$ be the transmitted signal at time index t. Then, for a narrow band transmission, the superposition of the multipath signals

can be written using the real operator $\Re$,

$$\tilde{y}(t,\bar{\gamma}) = \sum_{p=1}^{P} \Re\Big\{ a_p s(t-\gamma_p/c) exp\left(j2\pi[f_c t - \gamma_p/\lambda_c]\right) \Big\} \tag{1.2}$$

where λ_c and f_c are the wavelength and frequency of the carrier respectively. In the equation, the speed of light is denoted by c and the time index by t. The mean path length traversed $\bar{\gamma}$, is defined by

$$\bar{\gamma} = \frac{1}{P} \sum_{p=1}^{P} \gamma_p \tag{1.3}$$

Defining $\tau_p = \gamma_p/c$, Eqn. (1.2) reduces to the more familiar form:

$$\tilde{y}(t) = \Re \left\{ \left[\sum_{p=1}^{P} a_p s(t-\tau_p) exp(-j2\pi f_c \tau_p) \right] exp(j2\pi f_c t) \right\} \tag{1.4}$$

Then, under the assumptions of both a *time invariant* channel, and the existence of a *large number of multipaths*, the received baseband signal can be modeled by the integral,

$$y(t) = \int_{-\infty}^{+\infty} h(\tau) s(t-\tau) d\tau \tag{1.5}$$

where $h(\tau) = a(\tau) exp(-j2\pi f_c \tau)$. Here, $a(\tau)$ is the continuous-time form of a_p. Eqn. (1.5) reveals that the channel under these assumptions operate in a similar manner to a linear filter with an impulse response of $h(\tau)$. For a discrete system

this integral further simplifies to,

$$y(nT) = \sum_{l=0}^{L} h(lT)s(nT - lT) \tag{1.6}$$

when the output $r(t)$ is sampled every Ts and given that the channel has a finite impulse response of $L + 1$ symbols. This, with a slight abuse of notation can be written in the simpler form,

$$y(n) = \sum_{l=0}^{N} h_l s_{n-l} \tag{1.7}$$

where $h_l \triangleq h(lT)$ and $s_n \triangleq s(nT)$ for the n^{th} transmitted symbol.

The underlying assumption of time invariance holds in high speed communication systems. This is because there, the data packets are relatively shorter in duration with respect to the *coherence time* of the channel. The *coherence time* of a channel is the time which the impulse response of the media is highly correlated. The assumption of a finite channel length has also been verified by practical measurements. These experiments show that the bulk of the energy of a received symbol is concentrated in a finite time frame from the reception of the first ray.

Eqn. (1.5) suggests that the mobile channel can be mathematically modeled as a linear filter under the above two assumptions. However, modern wireless communication systems are primarily based on digital transmissions. Thus, Eqn. (1.6) provides a more accurate portrayal of the mobile media. This mathematical

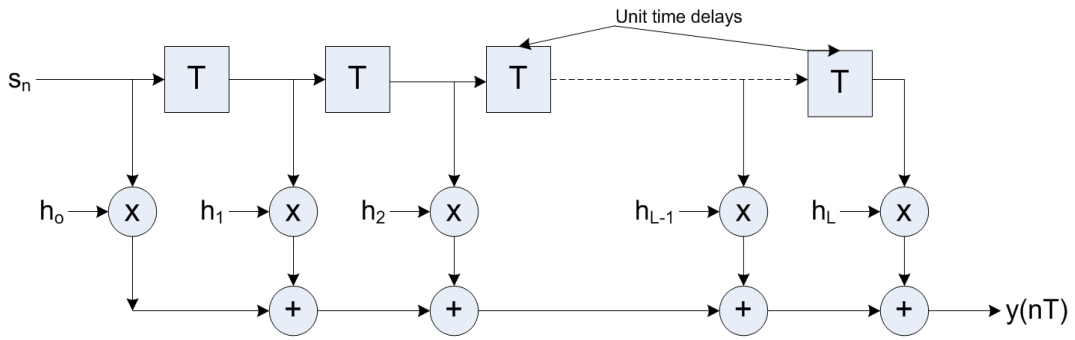

Figure 1.3: FIR structure of multipath channels

structure represents a Finite Impulse Response (FIR) transversal filter, and this is illustrated in Fig 1.3.

1.1.2 Inter Symbol Interference

The FIR structure evident in Fig 1.3 indicates that mobile channels create delayed and attenuated replicas for each symbol that is transmitted through the media. Thus, what incidents on the receiver is not only the transmitted symbol, but a superimposition of all the delayed signals that the media creates. This has the effect of smearing the symbol in time as shown in the first graph of Fig 1.4. Time-dispersion of the channel causes received symbols to trail for more than its allocated time period. Thus, components of one symbol begin to affect the received signal of adjacent symbols. This effect is known as Inter Symbol Interference (ISI). It corrupts the received signal, thereby preventing accurate reconstruction of the transmitted symbols. Fig 1.4 illustrates how time dispersion ultimately results in a received signal that has little or no resemblance to the

transmitted symbols. In such cases, accurate reconstruction of the transmitted symbol sequence is almost impossible without additional processing.

Time-dispersion in mobile channels is quantified using the *rms delay spread* parameter, σ_τ. This parameter is empirically derived using the power delay profile of a given channel. For channels that are Wide Sense Stationary with Uncorrelated Scattering (WSSUS) the power delay profile, $p(t)$ can be derived from the channel parameters [1] as,

$$p(t) = 0.5|h(t)|^2 \tag{1.8}$$

The rms delay spread is the square root of the second central moment of the power delay profile and it is defined as

$$\sigma_\tau \triangleq \sqrt{\bar{\tau^2} - \bar{\tau}^2} \tag{1.9}$$

where

$$\bar{\tau} = \frac{\sum_k p(\tau_k)\tau_k}{\sum_k p(\tau_k)} \tag{1.10}$$

$$\bar{\tau^2} = \frac{\sum_k p(\tau_k){\tau_k}^2}{\sum_k p(\tau_k)} \tag{1.11}$$

and $k \in \{0, ..., \infty\}$. Viewing from the frequency domain, the rms delay spread transforms into a *coherence bandwidth*. The physical interpretation of the coherence bandwidth, B_c is framed by a high correlation between of the two channels seen from two frequencies separated by less than B_c.

Although as mentioned previously, the channel distorts the received signal

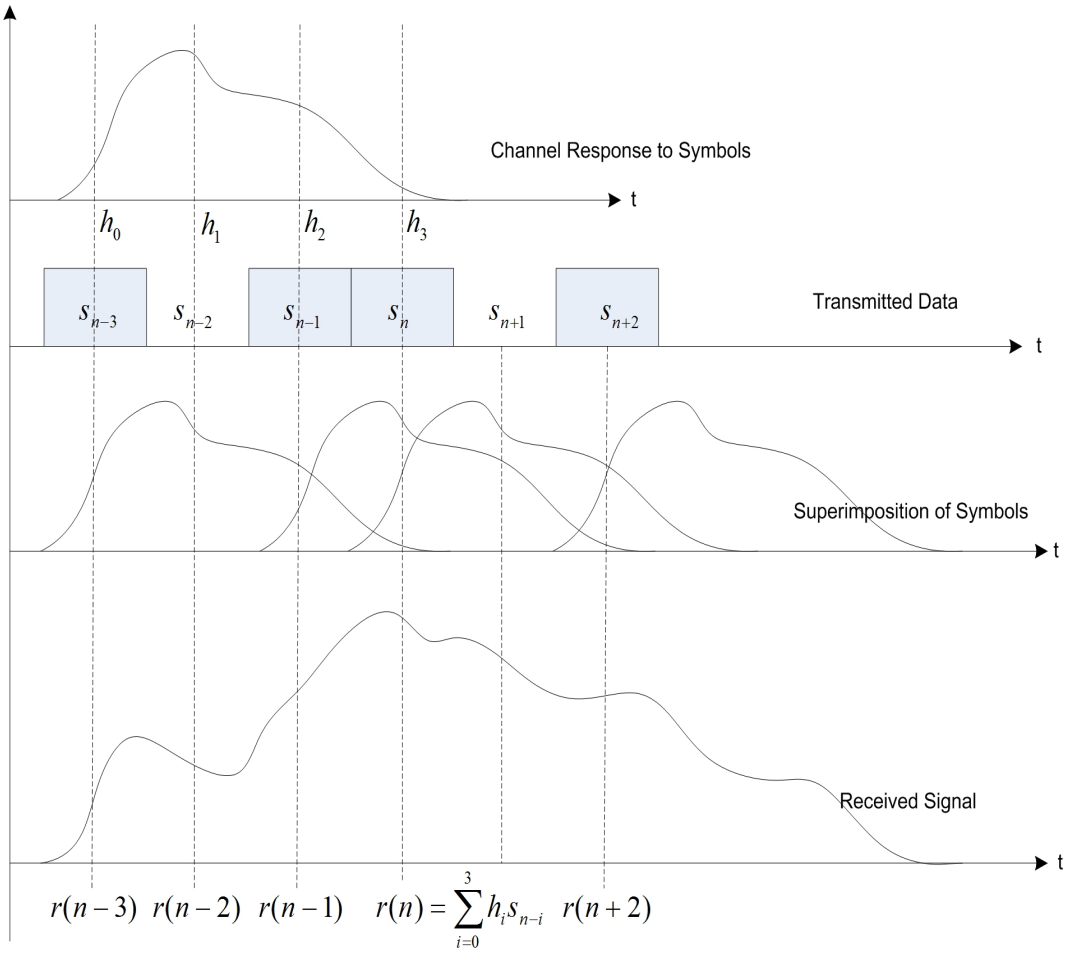

Figure 1.4: Smearing of received signal by ISI

to almost beyond recognition, there are tools available in communications to overcome and undo such distortions inserted by the media. They are,

Diversity

Diversity is a tool that is used to compensate for fading where the signal level drops to below the threshold of receptability in a receiver. It hinges on the premise that if more than one replica of a signal is received on uncorrelated channels, then the probability that all signals will fade simultaneously decreases rapidly with the number of received signals.

A number of methods exist to provide identical signals that arrive through uncorrelated channels.

- Spatial diversity - Here, the receiver antennae must be separated physically by more than half a wavelength to minimize channel correlation.
- Time diversity - For time diversity, the transmissions must be separated by more than the *coherence time* of the channel.
- Frequency diversity - In this case, transmission frequencies should differ by more than the *coherence bandwidth*.
- Polarization diversity - This form of diversity depends on the fact that the properties of mobile channels are dependant on the plane of polarization of the transmitted carrier.

These schemes provide the means to enhance the received signal so that the depth and duration of fades is appreciably reduced.

Channel Coding

Channel coding adds redundant data bits onto the transmitted symbol sequence so that even if a few bits are lost during fading, they can still be estimated or detected using the additional bits embedded onto the transmission. However, coupling additional bits onto the transmitted sequence reduces the raw data transmission rate.

Channel decoding generally takes place after detection . Thus, it is essentially a *post detection scheme*. Within channel coding, there are three main techniques that is widely used in mobile communications. Application of the type of coding depends on the requirements of the communication link. These factors include the bi-directionality of the link, the nature of the communication system: whether it is broadcast, multicast or unicast, and the bandwidth reduction that is tolerable. The three families of channel coding available are,

- Block codes
- Convolution codes and
- Turbo codes

Channel coding is generally independent of modulation schemes. However, with the advent of Orthogonal Frequency Division Multiplexing (OFDM), new space-time coding techniques that combines antenna or space diversity, coding and modulation have been proposed. These schemes offer high coding gains without any bandwidth expansion.

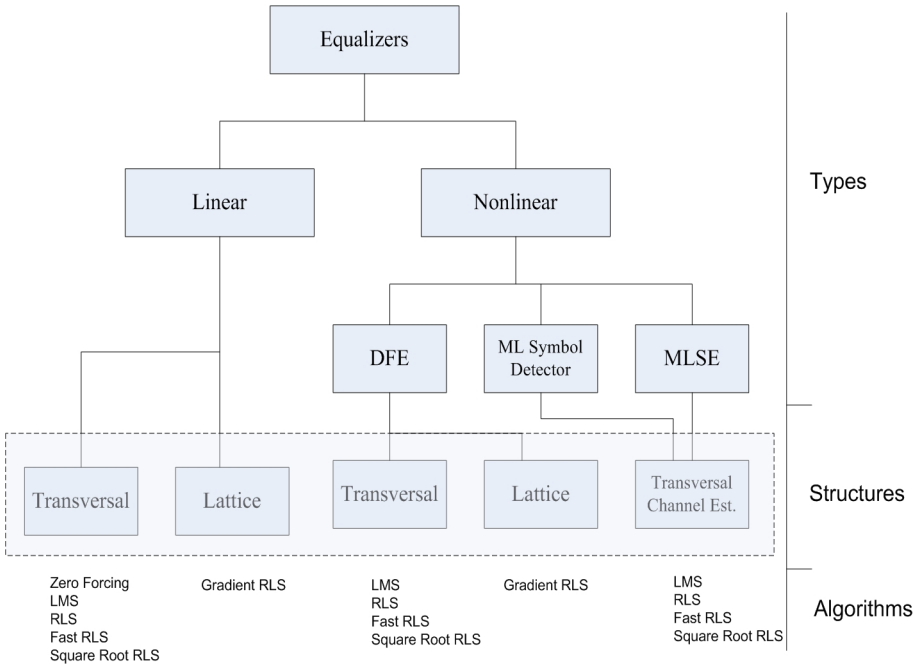

Figure 1.5: Filter structures and algorithms used for ISI cancelation

Equalization

Equalization compensates ISI that is generated by multipath, time-dispersive channels. In a broad sense, any signal processing technique that helps reduce ISI can be labeled as a equalizer. However, since mobile channel are time variant, these algorithms must be adaptive. Most of the equalization algorithms used today break equalizers into two components. A filter structure that is capable of modeling the inverse of a given mobile channel, and an adaptive component that estimates the filter taps to provide the best filter to compensate for the mobile channel.

Of the filter structures used, the most commonly used is the Linear Transversal Filter. The linear filter is essentially a tap delay line as shown in Fig 1.3.

Another popular filter structure is the Decision Feedback Equalizer (DFE). In contrast to the previous filter, the DFE filter has a non-linear structure. In addition to the filter structures, there need to be conditions or schemes that can be used to adjust the filter taps. The two widely used schemes in this area are the Zero Forcing (ZF) and the Least Mean Square (LMS) schemes. In the case of the ZF equalizer, the weights are chosen such that all but one of the combined channel and equalizer coefficients are zero. This however can create noise enhancement. The LMS equalizer on the other hand minimizes both ISI and noise. Such, it is a more optimum filter. However, both filters need the channel coefficient vector $\mathbf{h} = [h_0, h_1, h_2, ..., h_L]'$, to derive the optimized filter taps. A summary of the filter types, their implantation structures and the algorithms that can be used in adjusting the filter taps is illustrated in Fig 1.5

1.2 Blind Estimation

Traditionally, training sequences have been used for estimating channel parameters. In these algorithms, known bit patterns, $\breve{s}_n$ are transmitted. The receiver then adaptively adjusts the tap weight vector $\mathbf{f} \triangleq [f_0, f_1, ..., f_L]$ using schemes such as ZF or LMS to minimize the error signal $\breve{e}_n$. This is illustrated in the Fig. 1.6

However, in face of higher signaling and bandwidth requirements, training sequences are fast becoming a non viable option. For example, in GSM, training sequences use up to about 20% of the available channel [5]. Moreover, as the sig-

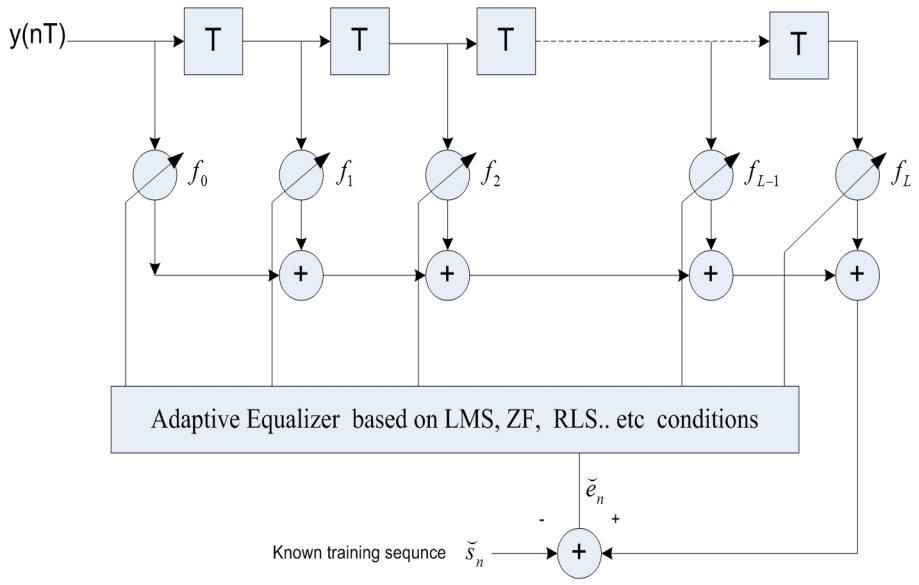

Figure 1.6: A linear trasversal adaptive filter structure

naling rate increases, the portion of the bandwidth used up by training sequences tends to increase. Another detrimental aspect of training sequences is that they cannot be used for estimating time varying channels. This is because they function under the assumption of a static channel to extract channel parameters from the training data. Moreover, where they can be used, strict synchronization restrictions have to be followed. Furthermore, even in slowly varying channels, training sequences become ineffectual when the channels undergo severe fading.

On the other hand, blind algorithms presents a bandwidth efficient alternative. Using information already embedded on the data stream, these algorithms are able to extract channel parameters at a higher computational cost. Starting from the seminal work of Sato [6] in 1975, blind algorithms have spread to include several different classes. They all however have key features that make them useful in both military and commercial high bandwidth applications.

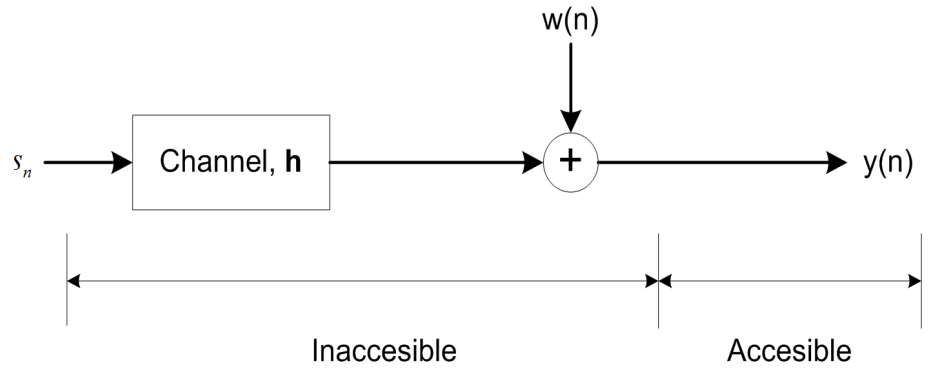

Figure 1.7: Schematic of the blind estimation problem

- *No training sequences required*, therefore conserve bandwidth and are harder to jam and hack into.
- *Robust to severe fading*, therefore ensures lower outages where signal levels fall below the receiver's threshold.
- *Capable of being used in estimating time varying channels*

However, they do come with their own inherent problems.

- *Computationally more expensive.*
- *Convergence to local minima* due to the non linear nature of estimation.

1.2.1 The blind estimation problem

The blind estimation problem is aptly described by Fig. 1.7. The essence of blind estimation is to extract the channel parameters **h**, and the source symbols $s(n)$, using only the channel output $y(n)$. Though distinguishing the channel

from the source may at first seem intractable, it can be done by exploiting the deterministic and statistical structures embedded by the channel and input. Define $\mathbf{h} \triangleq [h_0, h_1, h_2, ..., h_L]$ to be the channel vector. Let $\mathbf{s}_n \triangleq [s_n, s_{n-1}, s_{n-2}, ..., s_{n-L}]'$ be the transmitted symbol vector and w_k the noise element at time $t = nT$. Then the received signal element at time index n is given by,

$$y_n = \sum_{i=0}^{L} h_i s_{n-i} + w_n \tag{1.12}$$

In mathematical terms, the goal of blind estimation is to estimate either $\mathbf{h}$ or $\mathbf{s}$ given *only* the output vector $\mathbf{y}(n) \triangleq [y_n, y_{n-1}, y_{n-2}, ...]'$ and *prior* knowledge of statistical and deterministic structures of the input or channel or both.

Depending on the information they utilize, blind algorithms can be categorized into two main classes [7]. They are the *statistical* and *deterministic algorithms*. An important technique, the Maximum Likelihood (ML) estimators fall under both categories. ML estimators are optimal for large data sets, and under certain regularity conditions, the asymptotic variance of ML estimators approach the Cramer Rao Bound (CRB) [7]. These estimators have the added advantage of being able to be derived in a systematic manner. However, unlike subspace methods, they do not lend to closed form solutions. Numerous ML estimators have been proposed in literature. They can be found varying from the Deterministic ML approaches like IQML and TSML proposed by Hua [8] and Slock [9] to Statistical ML approaches like the Expectation-Maximization (EM) approach proposed in [10, 11]. In contrast, Single Input Single Output (SISO)

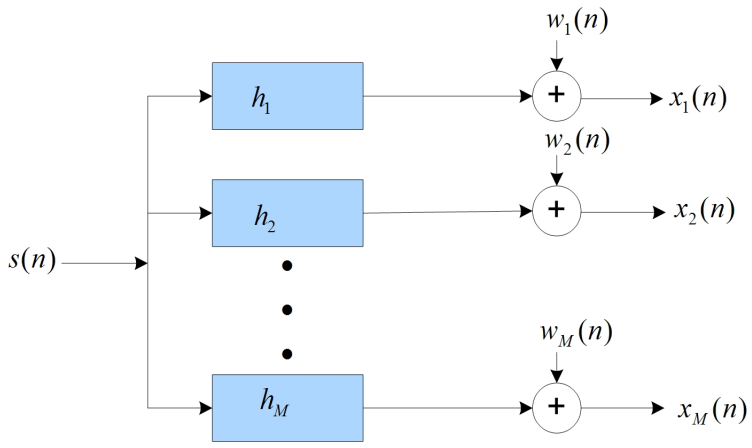

Figure 1.8: The Single Input Multiple Output channel model

systems rely primarily on statistical data gained from higher than second order statistics. This is because in absence of the multiple output structure, phase information needed to clean symbol or channel parameters can only be read from Higher Order Statistics.

Except ML estimators, most modern blind algorithms additionally require channel diversity. They use diversity in either spatial or temporal forms to transform the blind identification problem onto the Single Input Multiple Output (SIMO) platform [12]. The SIMO platform used by these algorithms is illustrated in Fig. 1.8.

1.2.2 Statistical and deterministic algorithms

Statistical algorithms assume the input **s**, to be random with predefined statistical properties. Generally, zero mean, independent and white distributions of known variances are assumed for both noise and **s** in this class of algorithms. Moreover, these algorithms require an accurate estimate of the channel length

(length of the channel impulse response) for reliable estimation.

The earliest blind algorithms were primarily based on Higher Order Statistics (HOS). This was primarily due to research then being concentrated on Single Input Single Output (SISO) channels. The SISO platform yields phase information only in higher than second order statistics. Thus, HOS was needed for estimation. On the other hand, Second Order Statistic (SOS) based algorithms extract phase information using the multichannel SIMO platform. This makes them more restrictive as HOS algorithms are able to perform without any channel diversity. Furthermore, HOS algorithms show asymptotic insensitivity to additive Gaussian noise that corrupts the received signals. This is useful in noisy environments. However, the HOS algorithms suffer from higher computational costs in constructing higher order cumulants. Furthermore, they require a larger data set for the estimates to stabilize compared to SOS algorithms. One important fact is HOS are the primary source of data for estimating channels on the SISO platform.

Generally, HOS algorithms can be categorized into three main classes. They are: the Hidden Markov Model (HMM) based algorithms, the Polyspectra methods and the Bussgang methods. The HMM [13, 14] algorithms provide estimates of channels driven by Finite Alphabet (FA) inputs using Markovian channel sequence information the FA property creates. This is viable in digital communications, where fixed constellations such as BPSK, QPSK and 16 QAM are used for data transmission. However, HMM algorithms require large memory and computational resources. Furthermore, they have a possibility of converging to

local minima. Polyspectra methods [15, 16, 17] on the other hand use higher order spectra. Using either the bispectrum (third order spectral cumulant) or the trispectrum (fourth order spectral cumulant) [18] they extract information needed to estimate channel parameters. The bispectrum however is not used much in communications. This is due to the fact that most communication systems use data that have pdf's symmetric around 0. This practice keeps energy requirements low, a prime concern in most communication systems. Thus these signals would contain no third order statistics and the bispectrum would be essentially useless. Another category of HOS algorithms, the Bussgang methods [20] do not explicitly use HOS. Instead, they minimize a cost function that implicitly contains HOS information. Bussgang algorithms are generally of an adaptive nature. These algorithms range from Sato [6] through Godard [19] to the stop and go algorithm of Picchi [20]. However, like HMM algorithms, both the Polyspectra and Bussgang methods may at times converge to local minima.

SOS algorithms are generally based on subspace decomposition. In one category, the cyclic spectra or cyclic statistics provides a key to identifying channels [21, 22]. However, in addition to the cyclic statistics, these algorithms require the FIR multichannel SIMO structure for estimation. SOS algorithms are generally more robust to noise than equivalent deterministic algorithms. However, convergence of source statistics is required for their optimum performance. Another category of statistical SOS algorithms that exist in literature are the Filtering Transform algorithms [23, 24]. These algorithms utilize a two-step, closed form approach to first estimate a filtering matrix $\boldsymbol{F}(\mathbf{h})$, and then derive the channel

parameters from the estimated matrix. However, this algorithm does not take advantage of the channel structure (structure of the filtering matrix, $\boldsymbol{F}(\mathbf{h})$ in this case). Furthermore, the accuracy of the estimate in the first step becomes a limiting factor in the accuracy of the estimate of the final result. However, when a large number of channels are available, using filter matrices for identification may have computational advantages. A third category of SOS algorithms falls under the generic banner of linear prediction. Introduced first by Slock [9, 25], they have an added advantage of being robust against over determination of the channel length. This is important as estimating the channel length may turn problematic in noisy environments.

Deterministic algorithms on the other hand do not assume any statistical structures to be present in the input. They are generally capable of finite sample convergence. That is, in absence of noise, the algorithms are capable of producing exact channel estimates using a finite number of samples. Statistical algorithms on the other hand need convergence of statistics for estimation. This makes deterministic algorithms more effective in regions of high SNR. In addition, its dependence on relatively shorter data sets makes it ideal for use in fading channels. Moreover, as it does not depend on source statistics, it can be used in a wider range of equalizing applications. However, deterministic algorithms suffer faster deterioration as the conditions within the media come close to violating its identifiability conditions. Secondly, they may at times require restrictions on the input sequence. This may complicate the identifiability conditions and is discussed by Hua [26] and Xu [27].

Deterministic algorithms in general exploit information structures that are present in either the multichannel SIMO platform or those generated by the FA property. These algorithms can be categorized basically into subspace and non subspace algorithms. The subspace algorithms can be further categorized based on the information structures they utilize. The Cross Relation (CR) approach which was independently discovered by Liu [28], Gurreli and Nikias [29], Baccala and Roy [30] and Robinson [31] exploits the multichannel structure. It performs effectively in regions of high SNR using a relatively short data set. However, the CR algorithm shows a relatively higher sensitivity to channel length over-estimation. Another algorithm, the Noise Subspace(NS) algorithm proposed by Moulines [32] exploits the structure of the filtering matrix. It forces the signal space to have a block Teoplitz structure, which is orthogonal to the noise subspace. The NS algorithm is strongly related to the CR algorithm [33] as they only differ in their parameterizations of noise and signal subspaces. Though, it is relatively more complex than the CR method, it appears to provide better estimates under most conditions. Recently another deterministic subspace method has been proposed by Tong and Zhao based on the Least Squares Smoothing (LSS) of the observation process [34, 35, 36]. This algorithm uses the isomorphic relationships between the inputs and the outputs of a channel. Using these relationships, the algorithm converts the blind estimation problem into a linear LSS problem. This makes the LSS algorithms capable of having adaptive implementations. Furthermore, some derivatives like the Joint Order Detection and Channel Estimation by LSS (J-LSS) algorithm, needs only an upper bound of the channel

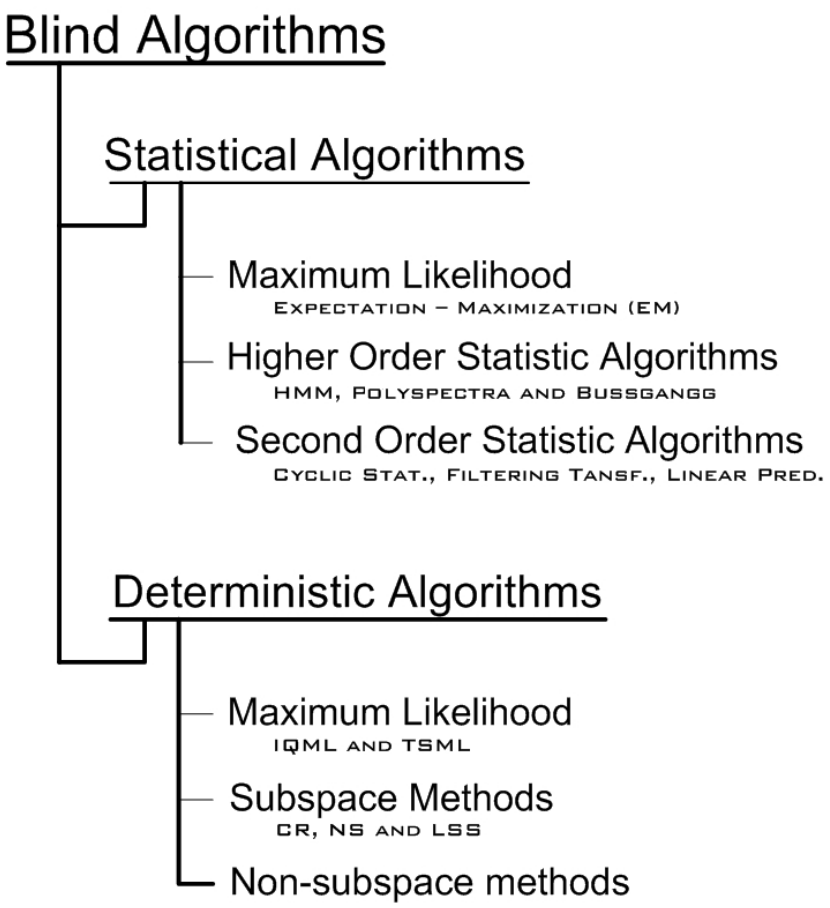

Figure 1.9: Classification of blind estimation algorithms

length to produce reliable estimates. A summary of the discussion presented is illustrated in Fig. 1.9.

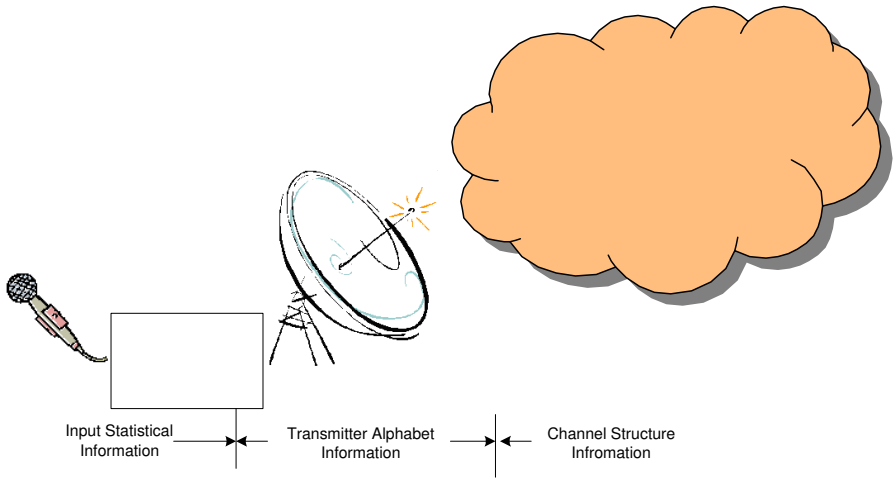

Figure 1.10: The embedding of data used for blind estimation

1.3 Finite alphabet algorithms

Besides the two traditional sources of information, there exists another category that is impinged onto the data stream at the moment of transmission. The convolution of a FIR channel matrix with a transmitter constellation creates a series of useful information that can be broadly categorized as Finite Alphabet (FA) data. Thus, FA data contains not only channel information, but within, it contains information that can be used to extract the transmitted symbol sequence. For example, most algorithms use prior knowledge of the transmitter constellation to ensure that the received symbols fall into one of the known elements within the constellation. A more definitive description of the structures that are used by our algorithms are presented in Chapter 2. Another distinction of the FA data with respect to the other two arises from its usage. In contrast to either statistical or algebraic channel structures that are traditionally confined to their respective algorithms, FA data can be used to supplement either algorithm

or used on its own. Such, algorithms that were originally categorized under statistical or deterministic categories can at times contain FA dependencies. The manner additional data that is used for blind estimation is embedded onto the transmitted signal is illustrated in Fig. 1.10.

Interestingly, the first blind algorithms to appear in literature can be categorized under this category. Both Sato and Godard used the FA property in penalizing the deviation of the equalizer output from either the binary states in Pulse Amplitude Modulation (PAM) [6], or the constant modulus condition in Quadrature Amplitude Modulation (QAM) [19]. In recent development, the Viterbi Algorithm (VA) has became a prominent tool in FA algorithms. This was precipitated by Forney in establishing that the VA can be used to compute the maximum likelihood estimate of the transmitted signal, provided that the multipath channel is known [37, 38]. Coupled with FA data, this has enabled the VA to form a nucleus for sequence estimation algorithms.

Numerous algorithms have developed on this theme. Tong in [39] outlines a novel algorithm that not only uses FA data, but also uses statistical and algebraic channel structure information. The algorithm uses the Mahalanobis-transform on the SOS subspace, and then the VA to search through the labels that are created. However, Tong's algorithms uses a SOS front end. Such, limitations of SOS are inherently transferred to this algorithm. Firstly, the statistical structure on the input data has to be assumed, and secondly, the phase ambiguity of SOS manifested as a sign ambiguity in the extracted symbol sequence. Furthermore, convergence of statistics becomes essential for optimal performance. However,

due to the statistical nature of this algorithm, it is more robust to noise than an equivalent deterministic algorithm. A low cost alternative to Tong's algorithm has been put forward by T Li and Z Ding [40]. Taking advantage of the structure of differentially encoded data, [40] outlines a scheme capable of reducing the states in the VA by at least half. However this method is valid only for Differential Phase Shift Keying (DPSK) signals. Extending Tong's work to the multi user platform, Gunther and Swindlehurst have proposed a novel source separation scheme using the shift structure present in the block Teoplitz structured input, together with the relationships of input and output subspaces [41]. However, rather than being statistical, this algorithm is more deterministic in nature. Van der Veen *et al* in [41] has outlined another technique for using FA data in source separation. Instead of subspace relationships, Van der Veen uses the Iterative Least Square with Projection (ILSP) algorithm to infuse FA structure onto the input. This provides a noise robust output with an added advantage that the algorithm can operate independent of the observed channel length.

In addition to the Viterbi Algorithm, the algebraic structure of the channel can also be used for both channel and sequence estimation. In [42], Manton and Hua outline a scheme that refines channel estimates by transforming the blind problem into a minimization problem. The FA structure provides the set of the discrete number of points to search for the minima. However, for optimal performance the algorithm requires a close initial estimate. This maybe problematic in noisy environments. Another pseudo deterministic algorithm for sequence estimation has been proposed by Yellin and Porat [43]. They utilize the FA property

to curb the exhaustive search for symbols needed to satisfy the Time Delay Line (TDL) equations,

$$\sum_{k=0}^{N} s_{n_i-k} h_k = y(n_i) \qquad i \in [1, M] \tag{1.13}$$

Being a deterministic algorithm, it converges to exact channel estimates in absence of noise. But on the other hand, it shows a higher sensitivity to noise. Moreover, the schemes assumes the existence of a correct symbol sequence satisfying its identifiability conditions with a probability of 1. In spite of these disadvantages, the algorithm has an in built insensitivity to order overestimation. Even though it is vulnerable to order underestimation, this makes the algorithm more viable in practical applications. Another approach to estimation using closed form forward and time reversal equations of the channel and symbol least squares estimates is outlined in [44]. With regard to performance, the algorithm is sensitive to the initial estimates it generates. The initial estimates play a crucial role on its global convergence capability. Sato in [5] suggests another algorithm for sequence detection in the form of Implicit and Explicit Blind Sequence Detection (IBSD/EBSD). It uses the short time average of squared error, together with the *Maximum a Posterior* (MAP) and the ML algorithms to generate Trellis labels. Then, [5] uses the VA to estimate the symbol sequence.

A more interesting algorithm from the point of this thesis was proposed by Daneshgaran [45]. In this thesis, the FA property is used in context of the clustering that occurs in the received vector set of a Single Input M Output (SIMO) channel. The received vector set of a SIMO channel describes points in

a M-dimensional space, thus algorithms using this category of information are described as spatial algorithms in this paper. Presence of noise causes the received vector set to deviate, forming clusters around theoretical centers. Daneshgaran [46, 45], using the Linde-Buzo-Gray (LBG) clustering algorithm with a novel initiation scheme, was able to present a methodology for extracting the clusters and then using them as labels for the VA algorithm.

1.4 Motivation and Thesis outline

In the domain of blind estimation, explicit use of FA data is a recent phenomenon. Admittedly, it was used in both statistical and deterministic algorithms starting from 1975 when Sato [6] first published his seminal work on blind estimation. However, these works used the FA data implicitly, using the Bussgang algorithms to fuse FA data with HOS for estimation. On the otherhand, explicit use of the FA data is more recent. Table 1.1 shows the distribution of blind algorithms categorized by the sources of information utilized for estimation. In this table, deterministic algorithms is split into two categories to form the general deterministic algorithms and the spatial algorithms. The algorithms in the general category rely primarily on the channel structure whereas spatial algorithms rely more on data structures created by the FA property.

Clearly the domain of spatial algorithms is largely unexplored, and thus it holds promise of alternate estimation algorithms with different behavior patterns to both statistical and deterministic algorithms. This thesis is largely motivated

Information Source	Statistical Algorithms	Deterministic Algorithms	Spatial Algorithms
Algebraic Channel Structure	*Low*	*High*	*Low*
Statistical Data Structure	*High*	*Low*	*Low*
Finite Alphabet Structure	*Medium*	*Medium*	*Low*

Table 1.1: Distribution density of blind algorithms, categorywise

to researching this promising area: *To use spatial data created by the FA property for blind estimation of channel and symbol parameters.*

This thesis consists of five chapters.

Chapter 1: The current chapter is intended as a primer to the background of blind estimation in mobile channels. Properties of the channel and how it affects mobile transmissions is analyzed in this section. Then, the need for blind estimation is explained followed by an in-depth discussion and analysis of currently available blind algorithms. Finally, FA algorithms are explored in the context of the current research thrust.

Chapter 2 is primarily concentrated on building spatial tool and information structures we will be using later in our algorithms. We begin by introducing the basic Multiple Output platform which forms one of the bases in our algorithms. In addition, the FA property and how it embeds channel and input symbol information onto the received vector, thereby creating spatial clusters is explained. An introduction into the spatial tools used in our algorithms is presented. The two main tools introduce here are the *primary* and *secondary* clustering algorithms. These tools enable us to handle spatial data in myriad of ways. Lastly,

we introduce the deterministic indices. These are mathematical structures that form the core of the Channel Estimation by Twin Indexing algorithm.

Chapter 3 Introduces the first of our spatial algorithms, the State Driven Sequence Estimation(SDSE) scheme. Indepth working of the theoretical algorithm is first presented and then followed a discussion of the errors that can plague it in noisy environments. Modifications needed to overcome these limitations are then presented, and finally, the performance of the algorithm is presented with an indepth discussion into its behavior.

Chapter 4 begins the presentation of the channel estimation schemes. Here, we introduce the two channel estimation algorithms, Channel Estimation by Difference Sets (CEDS) and Channel Estimation by Twin Indexing (CETI). These algorithms are explained in detail and followed by a procedural presentation that makes the algorithms easy to understand. Next, we present an auxiliary algorithm that helps overcome spatial algorithms inherent blindness to time ordering. This algorithm resolves the sign and permutation ambiguities inherent in the output of CEDS and CETI algorithms. Lastly, the performance of the CETI and CEDS algorithms are analyzed individually and with respect to each other and then followed by an indepth discussion into their behavior.

Chapter 5 first presents modifications that can be incorporated into our primary algorithms to extend their utility. These modifications enable our algorithms to work on new platforms ranging from complex transmitter constellations to Multiple Input Multiple Output (MIMO) systems. Next we introduce avenues open that can help enhance spatial algorithms. Finally, in the last section we

conclude our thesis, presenting the crux of our work.

Chapter 2

Spatial Structures and Tools

2.1 Introduction

In this chapter, we introduce the multiple output platform on which data structures we call *spatial structures* are formed when data is transmitted using a finite alphabet. Next, we introduce tools capable of processing the above spatial structures. These *spatial tools* form the foundation on which our blind estimation algorithms are developed. Lastly, we introduce a derivative of the spatial structure, *the Deterministic Indices*, which are formed when spatial structures are projected onto a one dimensional axis.

2.2 The Multiple Output Channel

Consider the single input, multipath communication channel shown in Fig 1.2 in *Chapter 1*. The received signal at the i^{th} receiver, $x_i(t)$ is the superimposition

of P multipath rays. Under the static channel assumption, the impulse response of such a channel, $c_i(t)$ can be described by,

$$c_i(t) = \sum_{m=1}^{P} \alpha_{im}\delta(t - \tau_{im}) \tag{2.1}$$

where $\{\alpha_{im}\}$ are zero mean Gaussian distributed reflection coefficients, and $\{\tau_{im}\}$, the randomly distributed path delays of the P multipath signals. Under these conditions, the received baseband signal takes the form,

$$x_i(t) \triangleq \sum_{n=-\infty}^{\infty} s_n h_i(t - nT) + w_i(t) \tag{2.2}$$

where $\{s_n\}$ is the transmitted symbol sequence and $h_i(t) \triangleq p(t) * c_i(t)$, the convolution of $p(t)$ with $c_i(t)$. In the equation, $p(t)$ describes the impulse response of the pulse shaping filter while T denotes the symbol period. The symbol $w_i(t)$, represents the band limited noise component present in mobile channels. The above equations are general and hold true for any power delay profile. For a set of discrete inputs, the received signal at time index nT may be further simplified to,

$$x_i(n) \triangleq \sum_{l=0}^{L} h_{il} s_{n-l} + w_i(n) \tag{2.3}$$

assuming the channel has an impulse response limited to $(L+1)$ symbol durations. The number of multipaths, P and the channel length, $L + 1$ are not directly related. Instead $L + 1$ is the time duration beyond which the composite channel response becomes trivial. On the other hand, P is the number of multipaths that

sum up to produce each of the $\{h_{il}\}$ coefficients. In Eqn. (2.3), $h_{il} \triangleq h_i(t - lT)$ denotes the discrete channel response coefficients. Stacking the output of M sensors, we can then obtain the equivalent Single Input Multiple Output (SIMO) channel model. This we write as,

$$\mathbf{x}(n) \triangleq \mathbf{H}\mathbf{s}(n) + \mathbf{w}(n) \tag{2.4}$$

where $\mathbf{x}(n) \triangleq [x_1(n), ..., x_M(n)]'$, $\mathbf{w}(n) \triangleq [w_1(n), ..., w_M(n)]'$ and $\mathbf{s}(n) \triangleq [s_n, ..., s_{n-L}]'$ are the received, noise and symbol vectors respectively. The channel matrix is denoted by $\mathbf{H} \triangleq [\mathbf{h}_1, \mathbf{h}_2, ..., \mathbf{h}_M]'$ with row vectors defined by $\mathbf{h}_m = [h_{m0}, h_{m1}, ..., h_{mL}]$. The objective of this thesis is to recover the channel and symbol parameters L, $\mathbf{H}$ and $\mathbf{s}$ using spatial data embedded in $\mathbf{x}$, under the following the key assumptions:

a) The channel is stationary for the time duration needed to collect data for estimation.

> This assumption is critical as all algorithms presented operate in a batch mode where a set of received data vectors is processed simultaneously. The time a channel is required to be stationary depends on the algorithm utilized. It is directly correlated to the smallest data set the algorithm utilizes to estimate channel parameters to the required accuracy.

b) The noise $\{w_i\}$ is zero mean, and statistically independent of the transmitted symbol sequence.

This condition is required for the clustering phase in our algorithm. It is only under this condition that the spatial structure can be extracted from a received vector set corrupted by noise.

c) The transmitter symbols are independent and chosen from a finite alphabet. For the purpose of this thesis $s_i \in \{1, -1\}$, and we define this alphabet $C_B \triangleq \{1, -1\}$.

This is a necessary condition for the creation of spatial structures in mobile channels. In our algorithms, any finite *transmitter constellation* or *alphabet* can be used. However, to produce a simple and clear presentation, we have in our thesis limited the constellations used to binary systems

d) The channel matrix $\mathbf{H}$ is full column rank. i.e. $M \geq L + 1$

This assumption does not originate from spatial algorithms. Instead, it is a requirement created by the use of an auxiliary algorithm to correct sign and permutation ambiguities of the channel matrix $\mathbf{H}$ extracted. These ambiguities result from *time-blindness*, inherent in spatial algorithms.

Let $\mathbf{M}$, $\mathbf{v}$ and s denote a Matrix, a Vector and a Scalar respectively. Then, in developing our thesis, we shall use the notations, $\mathbf{M}'$ and $\mathbf{M}^{\dagger}$, to denote the matrix operators, transpose and inverse (the pseudo-inverse when the matrix is not square). Furthermore, $\langle\,\rangle_i$ will denote the expectation operator over the index

i and $\sim$ will be used to associate a state to its spatial vector. The functions $abs[]$, $sgn[]$, $sum[]$, $max[]$ and $min[]$ are similarly defined, denoting the absolute value, the signum, the summation and the maxima and minima respectively.

2.3 The spatial structure and clustering

Consider the SIMO system described by Eqn. (2.4). Under noiseless conditions, the received vector $\mathbf{x}$ can be represented using its noiseless counterpart $\mathbf{y}$. This takes the form,

$$\mathbf{y}(n) = \mathbf{H}\mathbf{s}(n) \tag{2.5}$$

Then, under assumptions *(a)* and *(c)*, the vector set containing all received noiseless vectors,

$$Y \triangleq \Big\{\mathbf{y}|\mathbf{y} = \mathbf{y}(i) \quad i \in \{1, ..., N\}\Big\} \tag{2.6}$$

is finite with at most T^{L+1} elements. In the equation, N is the length of the sampled time duration, i.e number of received vectors, while T represents the number of symbols in the transmitter constellation. Each element of Y, $\mathbf{y} \in Y$ describes a point in an M-dimensional space. Thus the set Y describes a lattice in M-dimensional space. This M-dimensional lattice can be thought as a state diagram, where each element represents a unique state. Under this model, the output vector $\mathbf{y}(n)$ then can be seen transiting between the states in response to the input symbol s_n. This duality between the state diagram and its M-dimensional vector representation forms the basis of our spatial algorithms. In

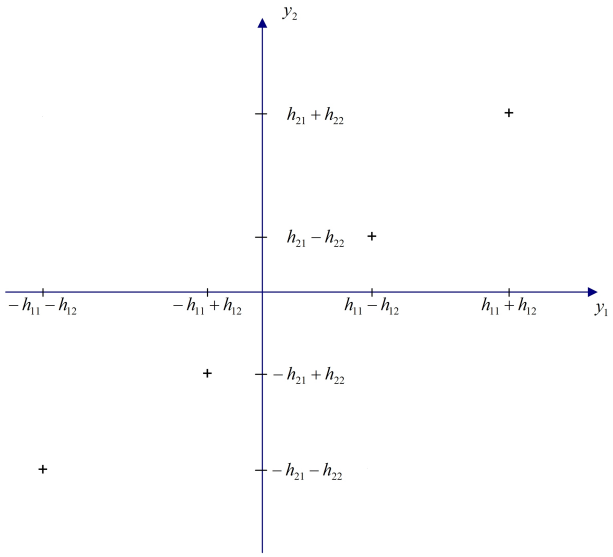

Figure 2.1: 2D structure of a vector space created by channel of $L = 2$

this thesis, we associate a given state S_k to its respective M dimensional *spatial vector* $\mathbf{y}(k)$ by,

$$S_k \sim \mathbf{y}(k) \tag{2.7}$$

Fig 2.1 shows a two-dimensional structure that is created in the received vector set of a two-sensor system in a channel of length $L = 2$. Noise corrupts this lattice like structure, dispersing the received vectors inside hyper-spheres of radii proportional to the noise power and origins defined by the noiseless vector set Y. This is shown in Fig 2.2. If noise is normally distributed, the pdf of the square of the cluster radii, U follows a central chi-squared distribution,

$$f_U(u) = \frac{1}{N_o^M 2^{M/2}\Gamma(M/2)} u^{(D/2-1)} exp(-u/2N_o) \tag{2.8}$$

where $2N_o$ denotes the noise variance and M denotes the number of sensors in

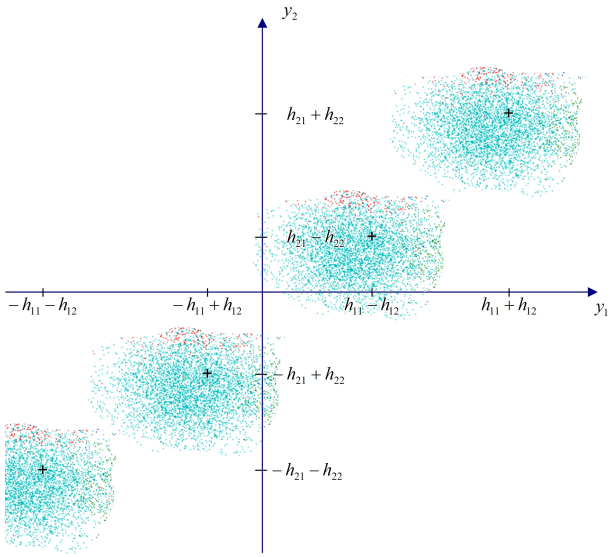

Figure 2.2: 2D structure corrupted by noise

the receiver. In Gamma function in Eqn. (2.8) is given by,

$$\Gamma(p) \triangleq \int_0^\infty t^{p-1}e^{-t}dt \tag{2.9}$$

Using clustering algorithms, it is possible to extract an estimate of the noiseless structure from the dispersed vectors [45]. Clustering is essentially a tool or algorithm that groups data using a defining characteristic that is unique to each group. In our case, the clustered groups are the points in the M-dimensional lattice structure Y.

Clustering algorithms have wide utility, especially in the field of *Machine Learning*. They are widely used in applications ranging from pattern recognition to data compression. Depending on the manner they approach the clustering problem, these algorithms can be classified into two main groups. They

are the *Parametric Clustering* [52] and *Non-Parametric Clustering* [53] classes. Parametric clustering attempts to minimize a cost function. Built on an optimization structure, these algorithms encompass statistical algorithms such as *Expectation-Maximization* to fuzzy implementations like *C Means Fuzzy Clustering* [50]. Non-Parametric algorithms on the other hand uses dissimilarities between clusters formed at a given iteration to either merge them together or split them apart. These algorithms, also called *Hierarchial Algorithms* do not need to make any assumption on the distribution of the data vectors they are processing. However, they have larger memory requirements and are more prone to errors when clustering regions overlap.

From a clustering point of view, the ability to separate the spatial structure, Y in a noisy environment depends on the ratio,

$$\rho = \frac{\text{Vol. of the noise hyper-spheres for a given containment probability}}{\text{Vol. of the hyper-sphere containing all noiseless channel outputs}} \quad (2.10)$$

This is explained by the illustration in Fig 2.3, which shows how signal and noise hyper-spheres are defined. Note that while the signal hyper-sphere encompasses all lattice points, the noise hyper-sphere is defined within the concept of a containment probability. The containment probability defines the expected percentage of points that should lie within the noise hyper-sphere. As refreshed in [45], the equations

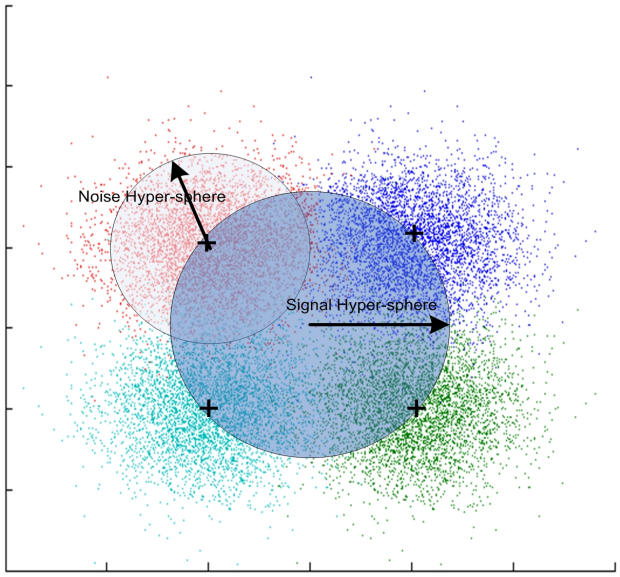

Figure 2.3: Signal and noise hyper-spheres

$$V(r,M) \;=\; \begin{cases} \dfrac{\pi^{M/2} r^M}{(M/2)!} & \text{for even } M \\ \quad or & \\ \dfrac{2^M \pi^{\frac{M-1}{2}} (\frac{M-1}{2})! r^M}{M!} & \text{for odd } M \end{cases} \tag{2.11}$$

define the volume of any M-dimensional hyper-sphere of radius r. Now, given a set of SIMO channels of length $L+1$, the radius squared of the noiseless channel outputs will have the form,

$$R_i^2 = \sum_{m=1}^{M} \left[\sum_{l=0}^{L} a_{mli} h_{ml} \right]^2 \tag{2.12}$$

where $a_{mli} \in \{+1, -1\}$ are randomly distributed transmitter symbols that each form one of the unique elements, i of the lattice Y. Then, assuming normalized

channels, i.e. $||\mathbf{h}_j|| = 1$ $j \in \{1, M\}$, the expected value of the radius square $\bar{R}^2$ can be expressed as

$$\bar{R}^2 = \sum_{m=1}^{M}\sum_{l=0}^{L}\left\langle a_{mli}^2\right\rangle_i [h_{ml}]^2 + \sum_{m=1}^{M}\sum_{l=0}^{L}\sum_{j=0,j\neq l}^{L}\left\langle a_{mli}a_{mji}\right\rangle_i [h_{ml}h_{mj}] \quad (2.13)$$

which under assumption *(c)* simplifies to,

$$\begin{aligned}\bar{R}^2 &= \sum_{m=1}^{M}\sum_{l=1}^{L}[h_{ml}]^2 \\ &= M \end{aligned} \quad (2.14)$$

On the other hand, the volume of the noise hyper-spheres can only be defined within the concept of a containment probability, p_c.

2.4 The spatial tools and contention clustering

Given the nature of the blind estimation problem, i.e. the lack of knowledge of channel parameters L and $\mathbf{H}$, contention based clustering emerges as a viable tool as it only requires an estimate of the noise power. The clustering algorithm used in our thesis was adapted from Danshgaran's derivative of the LBG algorithm [45]. It relies on the fact, given that noise power is within acceptable limits, then data points belonging to a given cluster are situated spatially closer to one another than to data points belonging to a different cluster. This can be seen in

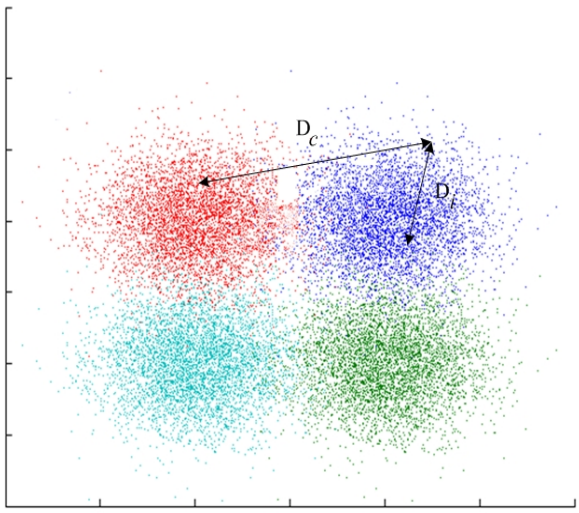

Figure 2.4: Separation criteria for clustering algorithms

Fig 2.4 where,

$$\langle D_c \rangle > \langle D_i \rangle \tag{2.15}$$

Spatial tools form the core of the blind estimation algorithms we will be presenting in this thesis. They form the handle which we will be using to manipulate spatial data for symbol and channel parameter extraction. The first algorithm we will be presenting is the *Primary* clustering algorithm. It is a de-noising tool that uses first order statistics and spatial knowledge (Eqn. 2.15) to extract the noiseless spatial structure Y (Eqn. 2.5) from the noise contaminated input vector set. The *Secondary* clustering algorithm although similarly structured serves another purpose. It provides the means to extract identical vectors corrupted by noise using the population of the vectors as a key. This tool can extract the channel vectors from the spatial data they are embedded in.

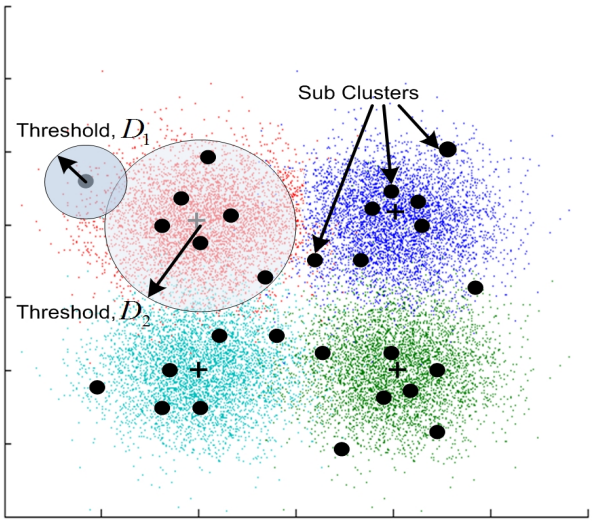

Figure 2.5: Sub clustering in the two-step primary clustering algorithm

2.4.1 The Primary Clustering algorithm

The clustering algorithm used in our simulations is based on a two-step approximation structure. The first step creates sub-clusters between the number of vectors in Y, and the number of final states, 2^L. The second phase then coalesces all sub-clusters belonging to a given cluster to one point. This sub-clustering effect is illustrated in Fig 2.5. The two step approach described above is more robust in regions of low Signal to Noise Ratios (SNR), and hence finds application in our proposed algorithms.

The two phases of the above algorithm relies on two thresholds for extracting and clustering vectors. The first, D_1 is used for extracting sub-clusters as illustrated in Fig 2.5. The second, D_2 then fuses all sub-clusters belonging to a single cluster into one point. The thresholds D_1 and D_2 in our thesis were obtained heuristically for normally distributed channel and noise parameters. Closed form

derivation of D_1 and D_2 for optimum cluster detection is complex and was not attempted in this thesis. The heuristic algorithms used in this thesis is presented at the end of the current discussion on clustering algorithms. However, though the two step approach has the distinct advantage of robustness, it may suffer a small loss of accuracy in the position of the extracted cluster centers.

Another threshold, $P_{MAX} = 0.8N/2^{L+1}$ is used in the second clustering phase to limit the number of vectors coalesced per cluster. This is 80% of the expected populations for each cluster. Limiting the population in this manner minimizes the probability of sub-clusters belonging to different clusters from fusing into other clusters. The 80% numeric was obtained through Monte-Carlo iterations by using different values to yield the best approximate to the number of clusters that should be ideally created.

To begin deriving our algorithm, we shall first define $\tilde{Y}$ to be the set of extracted cluster vectors initially containing $C = 0$ elements. The clustering algorithm can then be described as follows:

i) Scan the received vectors sequentially, comparing the Euclidean squared distance, d of each received vector to the established C cluster centers, $\tilde{\mathbf{y}}(m) \quad m \in \{1, .., C\}$ using,

$$d_{min} = \min_{m \in \{1,...,C\}} \sum_{i=1}^{M} [x_i(n) - \tilde{y_i}(m)]^2 \tag{2.16}$$

ii) If $d_{min} > D_1$, add the data vector as a new cluster center. Otherwise merge it to the closest center m weighted by the number of points already merged

into it. This yields the sub-clusters described above and ends the first phase of the clustering algorithm.

iii) Sort the sub-clusters, $\tilde{\mathbf{y}} \in \tilde{Y}$ by the number of data points fused into each center.

iv) Beginning from the least populated sub-cluster, for each center, j compute distances l_{jk} to all other sub-clusters, $k \in \{1, ..., C\}$.

$$l_{jk} = \sum_{i=1}^{M} [\tilde{y}_i(j) - \tilde{y}_i(k)]^2 \tag{2.17}$$

v) Find the closest center, k_S satisfying the conditions,

1. $l_{jk_S} < D_2$
2. $P_j + P_{k_S} < P_{MAX}$

vi) If k_S exists, merge the centers j and k_S weighted by their populations. Otherwise go back to *iv)*.

The resultant set of vectors, $\tilde{Y}$ will be an approximate to the noiseless lattice structure Y. In this algorithm, P_i denotes the population of the sub-cluster i.

Simulation platform used for estimating clustering thresholds and the channel length

The above results were obtained using the SIMO channel model. The channel was modeled as a stochastic SIMO model, with impulse parameters modeled as zero mean Gaussian processes having unit variances. Channel coefficients

and noise are assumed identically and independently distributed, and in this simulation, noise was modeled as a zero mean Gaussian process. The Monte Carlo trials were conducted using a data set of $N = 2000$ samples per iteration. Furthermore, the source symbols were generated from a alphabet of $\{+1, -1\}$ with equal probability. Results from 30 Monte Carlo iterations were compiled to obtain information indices. 30 iterations was chosen as then, 1st order statistics stabilized within acceptable ranges to extract mean values for the indices.

Derivation of the clustering thresholds

The distance thresholds D_1 and D_2 were empirically calculated using the clustering algorithm in an adaptive mode. In this step, Monte-Carlo iterations were carried out for each M-SNR pair, while gradually increasing the threshold distance till the number of estimated centers converged around twice the expected number of clusters, 2^{L+2} for D_1 in the first phase and to expected number of clusters, 2^{L+1} for D_2 in the second phase. Using this two-step approach, it is possible to push the collapse of states in low SNR regions lower than what a single step approach would yield. This could be due to the fact that as the SNR deteriorates, the cluster radii expand to encompass more than one cluster. Thus, a single threshold could collapse multiple clusters into a single point. However, when using two thresholds smaller sub-clusters are first created. These are more densely populated near the theoretical centers. Thus, a smaller secondary threshold can be used to collapse these sub-clusters to a single point without collapsing surrounding clusters into it. This is illustrated in Fig. 2.6 which shows how

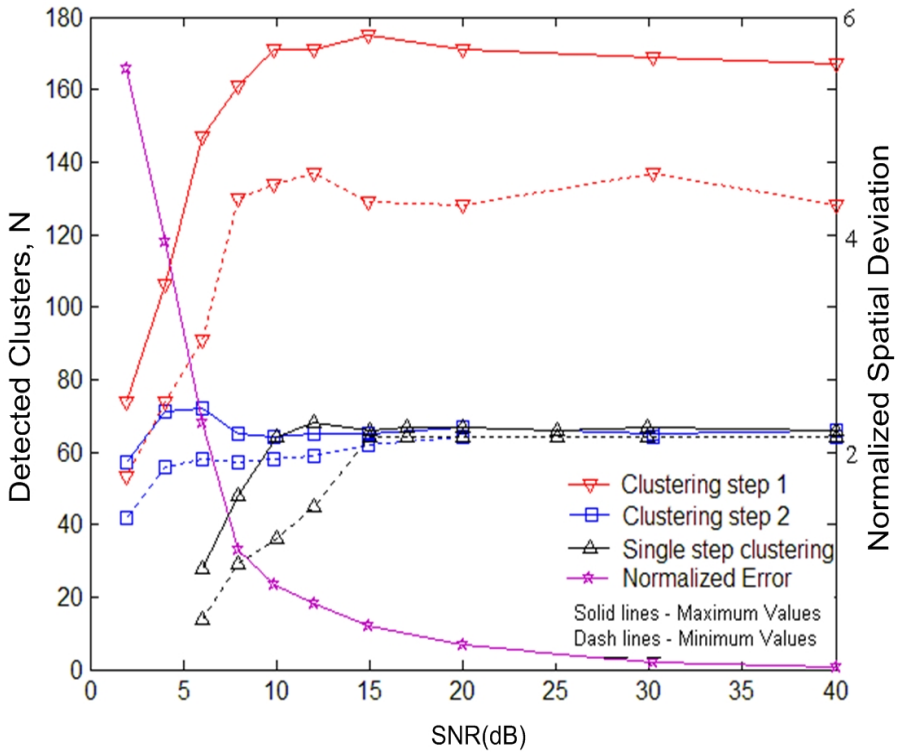

Figure 2.6: Cluster extraction

single-step and 2-step clustering algorithms behave in noisy environments. In the figure, normalized error shows the average spatial deviation of the estimated centers from the actual centers. In the series of figures 2.6, 2.7 and 2.8 the solid and dotted lines outline the maxima and minima respectively in the output set of Monte-Carlo iterations for each SNR value.

The above results were obtained using the simulation platform described above. A channel length of $L = 6$ was selected and simulations were carried out for $M = 8, 12, 16$ and 24 receivers. Then, the mean value of the of the clustering thresholds D_1 and D_2 were tabulated for use in spatial algorithms.

Channel length estimation

Using elementary curve fitting on all results obtained above through Monte-Carlo iterations, we were able to derive empirical relationships for the two thresholds using the parameters N_o, L and M. An interesting outcome of the modeling was the independence of the first threshold, $D_1 = N_o(2M + 5)$ from the channel length, L. This is also indicated in the results from the Monte-Carlo trials. This implies that the number of centers estimated by the first step of our clustering algorithm can also be used as a rough *blind estimator of the channel length*. Thus, though channel length was assumed known in the previous section, it can in fact be estimated using the PCA. However, it is important to keep in mind that this step requires knowledge of the SNR. Fig. 2.7 illustrates the channel length estimate extracted using the primary clustering algorithm. For this simulation, the above simulation platform was used with $M = 16$ multipaths.

From figure 2.7, it is clear that the clustering algorithm does not guarantee in converging to the exact number of states. In regions of high SNR, the algorithm may converge to a slightly higher estimate. This is because the threshold distances used to extract related spatial clusters are short in these areas. Thus, the algorithm may converge to two points instead of one. On the other hand, in regions of low SNR, the converse is true. Here, the noise radii may exceed cluster separation distances violating Eqn. (2.13). This results in the thresholds distances exceeding cluster separation distances. As such, two clusters or more may at times converge to a single point.

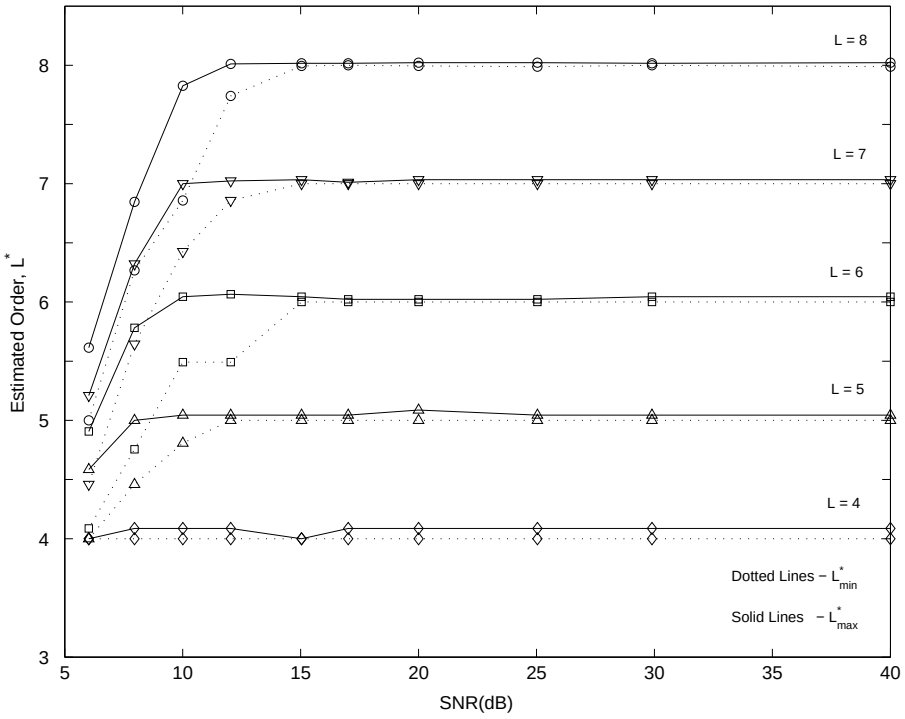

Figure 2.7: Order estimation using clustering algorithms

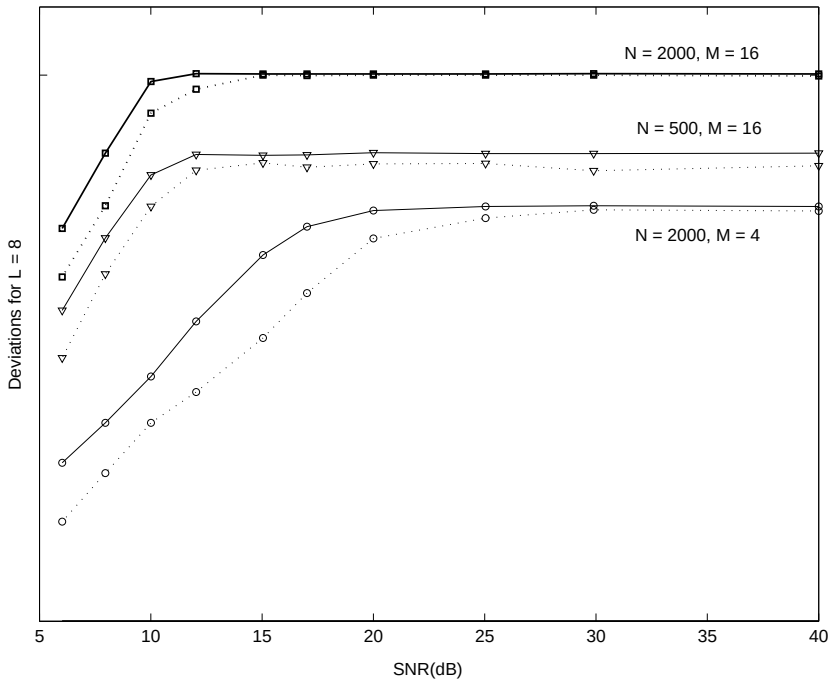

Figure 2.8: Factors affecting order estimation

The clustering algorithm described above is dependant on both the number of sensors, M and the size of the data set, N for estimation. Fig. 2.8 shows the dependency of the algorithm to these two factors. However, it is evident that while a reduction in the data set N creates a wider range for the estimates, a reduction in the number of sensors M is more detrimental. To add to the clarity of the figure, the following point needs to be stressed. The three graphs in Fig. 2.8 should ideally be superimposed on one another. However, an exploded view is given to better present the influence of the factors N and M in the algorithm.

However, the use of these thresholds D_1 and D_2 create a dependency on the knowledge of the SNR. This can be circumvented by putting the algorithm in a learning mode where it gradually increases the estimated noise parameter till the number of extracted clusters stabilizes to the theoretical value 2^{L+1}. One important fact this outlines is that clustering algorithms do not require the channel length as an input parameter. Knowing the SNR the system is operating in is sufficient for successful clustering. This in turn implies that the spatial algorithms developed in this thesis will be immune from the need to know the channel length L. Infact, the spatial algorithms can measure the rough channel length as shown above.

2.4.2 Secondary clustering

In addition to the primary need to separate the received data vectors into spatial clusters, we need an additional clustering tool that enables us to extract *vector families*. That is, given a vector family F_v of the vector $\mathbf{v}$ having a popu-

lation of P_v,

$$F_v = \left\{ \mathbf{f}_i | \mathbf{f}_i = \mathbf{v} + \mathbf{n}_i \quad i \in \{1, ..., P_v\} \right\} \tag{2.18}$$

we need to extract the estimates $\tilde{\mathbf{v}} \approx \mathbf{v}$ and $\tilde{P}_v \approx P_v$. The noise source $\mathbf{n}_i$ is assumed to be a zero mean source. The algorithm used for this purpose is basically a derivative of our primary clustering algorithm. It is limited to the steps *i)* to *iii)*, with a subtle variation in the derivation of D_1.

The threshold distance D_1 is empirically calculated using the deviation of the extracted vector population against the theoretical population. The theoretical population is extracted from a pilot output which is uncontaminated by noise. Two instances of the algorithm, one using noiseless data and the other in an adaptive form are run side by side across the entire M-SNR spectrum used in our thesis. At each M-SNR, the adaptive algorithm iteratively increases the threshold distance D_1 till the extracted populations falls within 2-5% of the theoretical populations. The expected values of D_1 across the twin indices of M and SNR are then tabulated to be used to separate and extract vector families.

2.5 1-D derivatives of the spatial structure

The data present in the multidimensional spatial structure is more than adequate for estimation purposes. In addition, *1-D* projections of this structure suffices for channel estimation. This is of immense value in creating practical algorithms. The resulting reduction in computational cost makes FA algorithms more attractive. Moreover, due to the superimposition of data when projected

onto the *1-D* axis, these algorithms require relatively shorter data sets for estimation. However, these advantages come with inherent limitations. The *1-D* algorithms suffer in low SNR regions as it does not have redundant information to increase the accuracy of its estimate.

In this thesis, we present a 1-Dimension derivative, the *Deterministic Indices* that is formed by the projection of the M-dimensional spatial structure onto a single axis. In the preceding section, an introduction into the structure of Deterministic Indices is presented, and later in Chapter 4, we will lay out an algorithm that uses this structure for estimating the channel matrix $\mathbf{H}$.

2.5.1 The Deterministic Indices

Consider the Single Input Single Output (SISO) channel model described in Eqn. (2.3). Under noiseless conditions it takes the simpler form,

$$\begin{aligned} y_i(n) = & \sum_{l=0}^{L} h_{il} s_{n-l} \\ = & \quad \mathbf{h}_i \mathbf{s}(n) \end{aligned} \tag{2.19}$$

where $y_i(n)$ is the received signal at time index nT. Define $Z \triangleq \{\mathbf{z} | \mathbf{z} = \mathbf{s}(i) \;\; i \in \{1, ..., N\}\}$ to be the set of all possible source vectors. Then, the set of the absolute value of the output, $|y_i|$ can be shown to be given by

$$O_i \triangleq \{o_i | o_i = abs\left([h_{i1}, h_{i2}, ..., h_{iL}]\mathbf{s}_l\right) \;\; \mathbf{s}_l \in Z\} \tag{2.20}$$

This is a projection of the M-dimensional structure onto the i^{th} axis. Under assumption *(c)*, the set O_i is finite. Moreover, the three largest elements of O_i can be proved to have deterministic forms. Let $o_{i\alpha}$ be the largest element of the set O_i. i.e. $o_{i\alpha} = max(O_i)$. Under *(c)*, $o_{i\alpha}$ satisfies

$$o_{i\alpha} = \sum_{l=0}^{L} |h_{il}| \tag{2.21}$$

which is formed by the symbol vector, $\mathbf{s}_\alpha = [sgn(h_{i0}), sgn(h_{i1}), ..., sgn(h_{iL})]'$. Similarly, the two next largest elements of O_i; $o_{i\beta}$ and $o_{i\gamma}$ can be shown to be formed by the source vectors

$$\mathbf{s}_\beta = [sgn(h_{i0}), ...\text{-}sgn(h_{iu}), ...sgn(h_{iL})]'$$

$$\mathbf{s}_\gamma = [sgn(h_{i0}), ...\text{-}sgn(h_{iv}), ...sgn(h_{iL})]'$$

and given by,

$$o_{i\beta} = \sum_{l=0, l\neq u}^{L} |h_{il}| - |h_{iu}|, \tag{2.22}$$

$$o_{i\gamma} = \sum_{l=0, l\neq v}^{L} |h_{il}| - |h_{iv}|, \tag{2.23}$$

where $|h_{iu}|$ and $|h_{iv}|$ are the two smallest elements of $\mathbf{h}_i$. All other elements of O_i are dependant on the distribution of the elements of the channel vector, $\mathbf{h}_i$ and thus have no deterministic form.

Similar to the M dimensional structure, noise corrupts these projections,

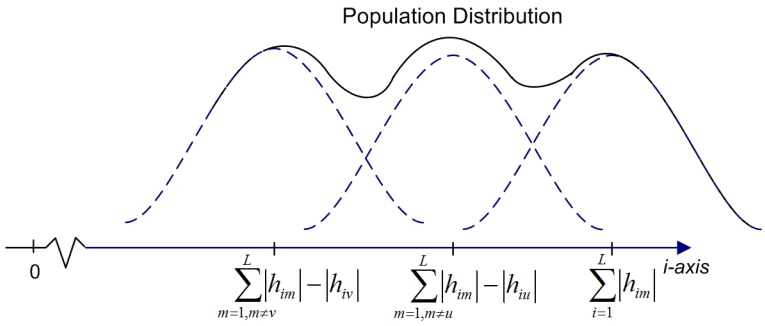

Figure 2.9: Linear projections and population distribution in noise

dispersing them about their theoretical origins. One way to estimate the theoretical centers is to use clustering algorithms. However, a 1-D axis provides little or no foothold for the clustering algorithms to work on. Therefore, instead of using only the given 1-D data, we can use all other dimensions inexorably linked to the 1-D structure in M space. i.e use the vectors $\mathbf{y} \in \mathbf{Y}$ such that $y_i = o_{i\alpha}$ for clustering.

Noise however mars such straight forward relationships. In practice we do not know the elements; $o_{i\alpha}, o_{i\beta}$ and $o_{i\gamma}$. However, taking into account that they are the three largest elements, we can pre filter, and extract $3 \bullet N/2^L$ vectors corresponding to the largest values of $[\mathbf{x}(n)]_i$ into another subset F_i. $3 \bullet N/2^L$ is the expected population for the three clusters. Statistics then imply that in this subset, the vectors with the deterministic elements at the i^{th} position, $\mathbf{f}_{i\alpha}, \mathbf{f}_{i\beta}$ and $\mathbf{f}_{i\gamma}$ will be more densely populated than other vector families. This provides the key for identifying the vectors and is illustrated in Fig. 2.9. Then, using the secondary clustering algorithm, it is possible to extract the vectors $\mathbf{f}_{i\alpha}, \mathbf{f}_{i\beta}$ and $\mathbf{f}_{i\gamma}$.

2.6 Summary

In this chapter, we presented the core channel platform and the assumptions that help formulate the algorithms proposed in this thesis. Another important base, the tools for handling spatial data was also presented. These tools form the core of spatial data processing. Furthermore, we explored the structure of the spatial data, and introduced the mathematical structures that from the last base of the mainstream algorithms. Thus, this chapter will be heavily referenced in the preceding chapters.

Chapter 3

Blind Sequence Detection

3.1 Introduction

The objective of this chapter is to show that spatial algorithms can be used as tools for blind sequence detection. In this respect, we introduce the State Driven Sequence Estimation (SDSE) algorithm. We begin this chapter with an introduction into the working of this algorithm. Here, mathematical structures that help in extracting the symbol sequence will be explained. Following, we present the algorithm in a structured form that simplifies understanding. Next, we highlight problems that may arise during realization of the algorithm and then proceed to outline solutions that minimize and ameliorates these effects. The gains achieved in overcoming realization problems are then demonstrated and followed finally by an indepth presentation and discussion into the performance of the SDSE algorithm.

3.2 State Driven Sequence Estimation (SDSE)

The ultimate aim of blind algorithms is to estimate the transmitted data sequence within an acceptable confidence. The methodology used in achieving the above goal can vary from algorithm to algorithm. However, in general, blind algorithms have two traditional approaches to solving this problem. One approach is to estimate the transmitted data sequence directly. The other is to estimate the transmitted data sequence by first estimating the channel parameters, and then utilizing these parameters to estimate the transmitted symbol sequence.

In this thesis, we attempt to show that spatial algorithms are capable of approaching the blind estimation problem in both directions mentioned previously. The main advantage of direct symbol estimation lies in the fact that it skips over the intermediate step of estimating channel parameters. This may make some of the direct estimation algorithms more computationally attractive. Furthermore, direct symbol estimation can be more accurate in estimating the transmitted symbol sequence. This is because accuracy maybe compromised when estimating symbols via channel parameters.

In this chapter, we will be focusing on the State Driven Sequence Estimation (SDSE) algorithm. It is a direct symbol estimator, bypassing intermediate estimation of channel parameters. In this chapter, we begin by providing the basis for formulation of this algorithm. Consider the spatial structure introduced in the *clustering* subsection of *Chapter 2*. Under noiseless conditions, the output vector of a multiple-output system alternates between the elements of the set Y.

This creates an allusion of a finite state machine, with the output transitioning between the states in response to the input $\mathbf{s}(n)$. For a binary system such as BPSK, each state has two possible inputs and two possible outputs. This Two-Input Two-Output (TITO) basis creates the information structures we need for estimation in the SDSE algorithm. This state like pseudo structure is illustrated in Fig 3.1. To continue formulation of the SDSE algorithm, let us assume that the received vector at time index nT_s corresponds to the state S_n. i.e. $S(n) = S_n$ where $S(n)$ is a time indexed state array as shown in Table 3.1. Thus we can write,

$$\begin{aligned} S(n) &= S_n \\ S_n &\sim \mathbf{y}(n) \\ \therefore S(n) &\sim \mathbf{H}[s_n, s_{n-1}, ..., s_{n-L}]' \end{aligned} \tag{3.1}$$

using the notation as defined in *Chapter 2* that links a *spatial vector* to a *state*. The *spatial vector*,

$$\mathbf{y}(n) \triangleq \mathbf{H}[s_n, s_{n-1}, ..., s_{n-L}]'$$

consists of two components. They are the *channel matrix*, $\mathbf{H}$ and the *source vector* segment, $[s_n, s_{n-1}, ..., s_{n-L}]'$. The behavior of the *source vector* segment plays a crucial role in our algorithm. Now under assumption *(c)*, the next transmitted symbol can be denoted as $s_{n+1} \in \{+1, -1\}$. Assumption *(c)* in *Chapter 2*

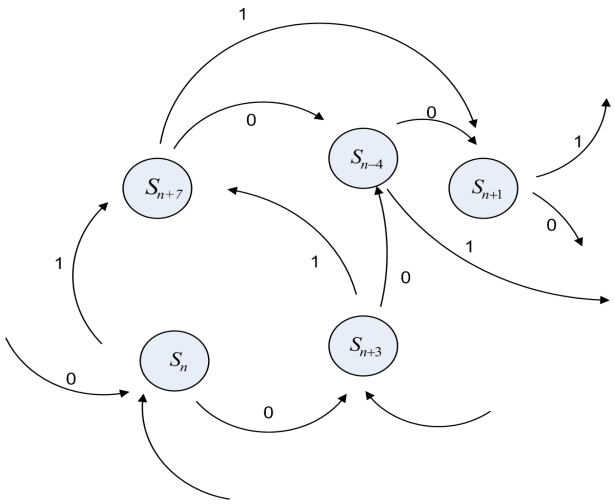

Figure 3.1: Typical state transition diagram

essentially states that the transmitter constellation used in this thesis is limited to $\{+1, -1\}$. Entry of s_{n+1} into the FIR channel forces the oldest symbol, s_{n-L} out of the channel's memory. Thus, the *source vector* segment of the next spatial vector consists of either a positive or negative realization of s_{n+1} minus the oldest symbol s_{n-L}. This results in either,

$$S(n+1) = \begin{cases} S^{+}(n) \sim & \mathbf{H}[+s_{n+1}, s_n, ..., s_{n-L-1}]' \\ & or \\ S^{-}(n) \sim & \mathbf{H}[-s_{n+1}, s_n, ..., s_{n-L-1}]' \end{cases} \tag{3.2}$$

depending on the transmitted symbol. The physical realization of a new symbol entering the channel is illustrated in Fig 3.2. Then with reference to the state diagram in Fig 3.1, the states $S^{+}(n)$ and $S^{-}(n)$ become the two possible transitions out of $S(n)$.

To extract the mathematical structures hidden behind the spatial states, let

State at Time Index	s(0)	s(1)	...	s(n-1)	s(n)
Time Index	0	T	...	(n-1)T	nT
State	S_0	S_8	...	S_k	S_n

Table 3.1: Time Indexed state array

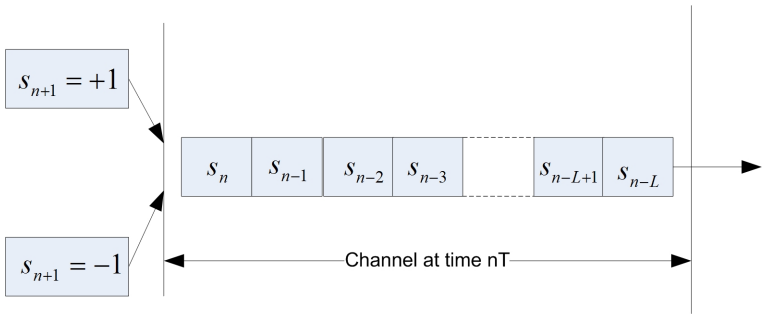

Figure 3.2: Typical state transition diagram

us now define $\mathbf{d}_{n+1}$ to be the difference vector of the two states, $S^{+}(n)$ and $S^{-}(n)$ as shown below:

$$\begin{aligned} \mathbf{d}_{n+1} &\sim 0.5\left[S^{+}(n) - S^{-}(n)\right] \\ &= 0.5\mathbf{H}\Big[[+s_{n+1}, s_n, ..., s_{n-L-1}] - [-s_{n+1}, s_n, ..., s_{n-L-1}]\Big] \\ &= s_{n+1}[h_{10}, h_{20}, ..., h_{M0}]' \end{aligned} \tag{3.3}$$

Of the two components that make up $\mathbf{d}_{n+1}$, the channel component, $\acute{\mathbf{h}}_0 = [h_{10}, h_{20}, ..., h_{M0}]$ is independent of the time index n. s_{n+1} on the other hand modulates the direction of $\mathbf{d}_{n+1}$ and thus holds the key to estimation. To complete formulating the SDSE algorithm, we shall undertake the two assumptions

stated below in addition to the general assumptions stated in *Chapter 2*

e) The length of the data set is sufficient for completely filling up the state transition table.

For complete extraction of the transmitted data sequence, the state transition diagram shown in Fig 3.1 must be 100% linked. This inturn assures a complete state transition table. An incomplete state table makes the algorithm blind when transmitted symbol sequences resulting in the vacant states are generated.

f) Of the two possible output states from $S(0)$, one is assumed to be caused by $\{+1\}$ and $\mathbf{d}_1$ is defined to point in its direction. This is equivalent to assuming $s_0 = +1$.

This assumption underscores the sign ambiguity that is inherent in the extracted symbol sequence. As the sign ambiguity cannot be resolved, the sign of the difference vector generated in the first iteration is assumed positive. However, the algorithm is equally capable of assuming the converse. This however, generates a symbol sequence out of phase by 180 degrees.

These assumptions are vital, as only under them do conditions exist where the spatial structures contain sufficient information to completely resolve the transmitted symbol sequence. To estimate the symbol s_{n+1}, we first need to identify which of the two output states of $S(n)$ corresponds to the transmission of a $\{+1\}$.

This can be established under assumption *(f)*, which defines the reference vector $\mathbf{d}_1$. Under these conditions, the sign of the symbol transmitted can be reasoned thus: If the two difference vectors $\mathbf{d}_{n+1}$ and $\mathbf{d}_1$ point in the same direction, then the state transition, $s(n) \rightarrow S^{+}(n)$ is due to $\{+1\}$. Otherwise the state $S^{-}(n)$ contains the positive transition. This knowledge when coupled with time information in the time indexed state array, i.e.

$$s(n) \rightarrow s(n+1) \triangleq s(n) \rightarrow S^{+}(n) \tag{3.4}$$

$$s(n) \rightarrow s(n+1) \triangleq s(n) \rightarrow S^{-}(n) \tag{3.5}$$

what of the two equations, (3.4) or (3.5) holds true, can be used to estimate the symbol $\tilde{s}_{n+1} \approx s_{n+1}$. This is illustrated in Table 3.2.

In the next section we attempt to outline our algorithm explicitly. There, we attempt to structure the formulation presented above into logical steps. These logical steps enables us to present the algorithm in a clear procedural form that can be easily grasped.

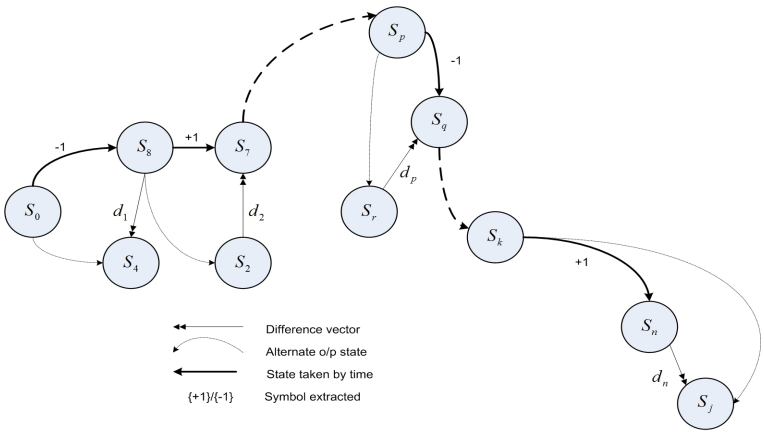

Figure 3.3: Visualization of the decoding process

Current State S(n)	Next State $S^+(n)$	Next State $S^-(n)$	Difference Vector	Phase wrt. to $\mathbf{d}_0$	$S_{+1}(n)$ {+1}State wrt. $\mathbf{d}_0$	$S(n+1)$	Symbol s_n
S_0	S_4	S_8	$\mathbf{d}_0$	0	S_4	S_8	-1
S_8	S_2	S_7	$\mathbf{d}_1$	180	S_7	S_7	1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
S_p	S_q	S_r	$\mathbf{d}_p$	180	S_r	S_q	-1
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
S_k	S_n	S_j	$\mathbf{d}_n$	0	S_n	S_n	1

Table 3.2: State Transition Table and symbol extraction

3.3 The core SDSE algorithm

i) Use the primary clustering algorithm to extract estimates of the spatial structure Y, to $\tilde{Y}$ from the data vectors $\mathbf{x}(n)$ $n \in \{1, ..N\}$.

ii) Assign each unique estimated center, $\tilde{\mathbf{y}}_i$ to a state S_i.

iii) Assign each data vector, $\mathbf{x}(n)$ to its closest state S_j using spatial separation between the vectors $\mathbf{x}(n)$ to each of the *spatial vectors* $\tilde{\mathbf{y}}_i$ $i \in \{1, .., C\}$.

$$\min_{j \in (1,C)} || \tilde{\mathbf{y}}_i - \mathbf{x}(n) || \tag{3.6}$$

iv) Add the state to the time indexed *spatial vector* $S(n)$.

v) Following $S(n)$, build the state transition table as given by Table 3.2.

vi) Create the difference vectors $\mathbf{d}_n$ and find which of the states, S^+ or S^- corresponds to the transmission of a $\{+1\}$. (Note: S^+ and S^- are chosen randomly).

vii) Following $S(n)$ and using the table, estimate the transmitted bit sequence as follows.

- if $S(n+1) = S_{+1}(n)$ then $s_{n+1} = +1$
- if $S(n+1) \neq S_{+1}(n)$ then $s_{n+1} = -1$

3.4 Issues when implementing SDSE

The SDSE algorithm described in the previous section was developed without factoring the effect of noise. As such, the two independent channel components, channel noise and distribution of parameters in the channel matrix can at times cause the algorithm to fail. In this section, we outline problems that may arise when using the theoretical SDSE algorithm, along with modifications that have been added to make it robust to face each of these problems. The end result is a modified SDSE algorithm that is more robust to both noise and channel parameter distributions.

3.4.1 Sign ambiguity

The extracted symbol sequence depends on the premise of the direction of the reference vector $\mathbf{d}_1$. Consequently, the extracted data sequence contains a sign ambiguity. This is inline with our assumption *(f)* and cannot be resolved using the FA data used in our algorithm.

3.4.2 Dependency on the channel matrix

The difference vectors $\mathbf{d}_1$ and $\mathbf{d}_n$ form an essential component of the estimation algorithm. The capability of the algorithm is limited to how well it can estimate the directions of $\mathbf{d}_1$ and $\mathbf{d}_n$ relative to one another. Estimation of the relative direction between the two vectors becomes problematic when the magnitude of the channel component, $|\acute{\mathbf{h}}_0|$ comes onto the same magnitude order as

the noise power. This can happen either when

- Operating in low SNR regions or when
- Extreme instances of the channel matrix occur, where the magnitude of leading column $|\acute{\mathbf{h}}_0|$ drops such that $|\acute{\mathbf{h}}_0| \approx \sigma^2$

However, even under such conditions preventing or minimizing the probability of an erroneous estimate of the relative direction between $\mathbf{d}_1$ and $\mathbf{d}_n$ is possible. First, we can improve the estimate of the vector $\mathbf{d}_1$ by extracting the expected value of it across the entire table.

$$\mathbf{d}_1 = \left\langle \frac{(\mathbf{y}_i^+ - \mathbf{y}_i^-)[\mathbf{d}_1]_j}{[\mathbf{y}_i^+ - \mathbf{y}_i^-]_j} \right\rangle_i \tag{3.7}$$

Here, $\mathbf{y}_i^+ \sim S_i^+$ and $\mathbf{y}_i^- \sim S_i^-$ are the *spatial vectors* corresponding to the two output states of a given i^{th} row in Table 3.2. The index j represent the largest element of the vector $\mathbf{d}_1$ and the notation $[\mathbf{d}_1]_j$ denotes the j^{th} element of the vector $\mathbf{d}_1$. In other words, this module firsts corrects the direction of the difference vectors generated in Table 3.2 using the largest element of $\mathbf{d}$, and then extracts the expected vector from the set $\mathbf{d}_1 ... \mathbf{d}_n$. Corruption of the largest element of $\mathbf{d}$ is least probable in noisy conditions, thus it becomes a robust index to estimate the difference vectors direction.

Another way to minimize the probability of extreme channel matrix configurations from occurring is to increase the number of sensors. Increasing M increases the dimensions of $\mathbf{d}_1$ and $\mathbf{d}_n$. Thus, it decreases the probability of

$|\acute{\mathbf{h}}_0| \approx \sigma^2$ from occurring as the number of elements making up $\mathbf{d}_1$ increases. If h_{i0} $i \in \{1, ..., M\}$ has a distribution of $\wp(i)$, this can be mathematically described by

$$P\left(\left[\sum_{i=1}^{M} h_{i0}^2 \right]^{1/2} > \sigma^2 \right) < P\left(\left[\sum_{i=1}^{M+m} h_{i0}^2 \right]^{1/2} > \sigma^2 \right) \quad m > 0 \tag{3.8}$$

$P(A)$ here denotes the probability of the event A happening.

A more practical solution to increasing the number of sensors is to do a backward estimation. That is, instead of working from state $S(n)$ to $S(n+1)$ using the two output states, it is possible to estimate the symbols from the transition of $S(n+1)$ to $S(n)$ using the two input states. The advantage gained is that the channel component of the difference vector changes from $\acute{\mathbf{h}}_0$ to $\acute{\mathbf{h}}_L = [h_{1L}, h_{2L}, ..., h_{ML}]$. Furthermore, the probability that both $\acute{\mathbf{h}}_0$ and $\acute{\mathbf{h}}_L$ will fade to the noise level is much less probable than that of either one of them fading.

$$P\left(|\acute{\mathbf{h}}_L| \approx \sigma^2 \cap |\acute{\mathbf{h}}_0| \approx \sigma^2 \right) \ll P\left(|\acute{\mathbf{h}}_L| \approx \sigma^2 \right) \tag{3.9}$$

3.4.3 Dependency on the TITO structure

The accuracy of the SDSE algorithm depends on the state transition table that it derives by scanning the input data vectors. However, clustering may not be perfect due to noise. One result of sub-optimum clustering is the creation of *split* states. This happens when the second phase of the clustering algorithm is not able to reconcile all sub-clusters that are parts of a single cluster into one point. These states are located spatially close to one another. However, they violate the

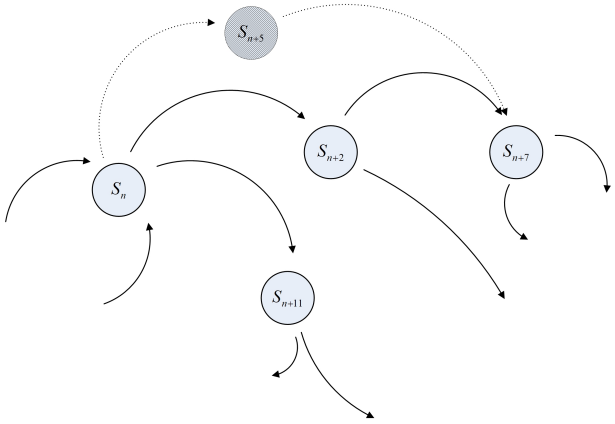

Figure 3.4: A Single input single output state

TITO structure of the state diagram. One undesirable result this causes is the creation of single output states. This is shown in Fig. 3.4. The algorithm relies on each state having two output states to first generate a difference vector, and then use the difference vector to estimate which of the two states corresponds to the transition of a $\{+1\}$. This however is not possible with single output states. The solution to resolving single output states is to walk back through the state diagram and reach a common root. On each step down, all possible paths from that state are checked against the path taken down to see if there is an alternate path that is spatially close using the cost function,

$$C_p = \frac{1}{m} \sum_{i=n-m}^{n} \mathbf{y}_i - \tilde{\mathbf{y}}_{i,p} \tag{3.10}$$

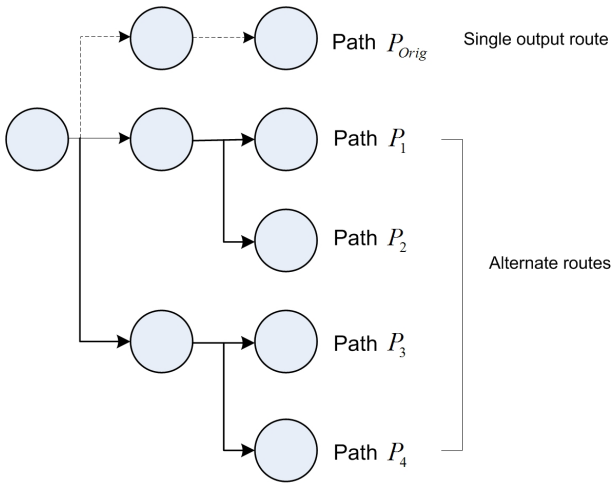

Figure 3.5: Alternate route search

that measures the spatial deviation of the new route from the original. Here, p is the path being evaluated, and the series $[\mathbf{y}_{n-m}, .., \mathbf{y}_n]$ forms the state vectors of the path climbed down. The path being evaluated is described by the vector series $[\tilde{\mathbf{y}}_{n-m,p}, .., \tilde{\mathbf{y}}_{n,p}]$ and m denotes the number of states walked back. A simple example to this is illustrated in Fig 3.5. On arrival of an optimal root, the alternate route is taken and all SISO states on the route are merged to their TITO counterparts. Fig. 3.4, shows such a case where the states S_{n+2} and S_{n+5} are in reality split states. Another error that can occur is the creation of multiple output states. This is especially true in low SNR regions, where noise causes some data vectors to be erroneously assigned to wrong states. These states have 4 or more outputs. Thus, deriving the difference vector becomes problematic. There is no clear solution to this error. It can however be reduced. First, when the probability of an erroneous transition is lower, we can utilize the usage of

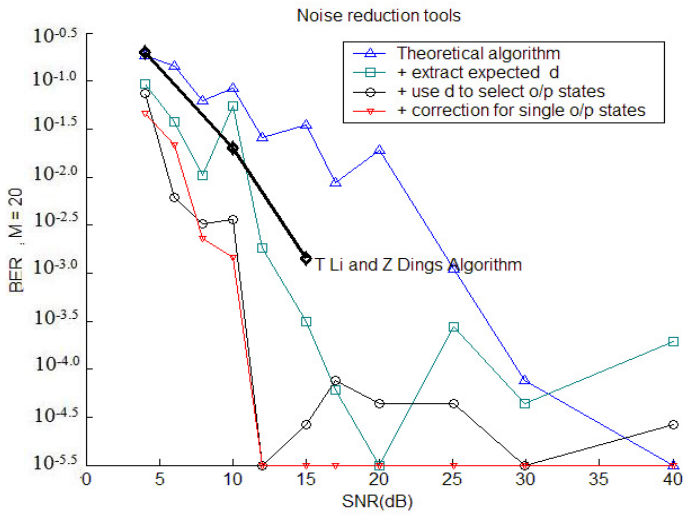

Figure 3.6: SDSE algorithm with correction modules

the output states to limit our search for the two correct states. Secondly, as all real difference vectors vary only in direction, finding which two states generate a difference vector that is parallel to the difference vectors generated by other TITO states gives promising candidates. By using these two restrictions, it is possible to decrease the error probability in low SNR regions.

3.5 Results and discussion

The channel model we used in our simulations was a stochastic SIMO model, with impulse parameters modeled as zero mean Gaussian processes having unit variances. Channel coefficients and noise are assumed identically and independently distributed, and in this simulation noise was modeled as a zero mean Gaussian process. For comparison purposes, we benchmark different aspects of our algorithm against the reference system described below. For the reference system, a channel length of $L = 6$ was selected with $M = 12$ receivers, and the results were then obtained using a data set of $N = 2000$ samples per iteration. The results obtained were then averaged over 30 Monte-Carlo iterations. 30 Monte-Carlo iterations enables us to see the stochastic behavior of the algorithm. The SDSE algorithm has a step like estimating capability, thus the benefits of using a larger Monte-Carlo set is negligible.

In this section, we shall first examine the SDSE algorithm in its theoretical form and subsequently proceed to outline the effects of various recovery modules that have been later incorporated into it. In the later second part, we shall examine and evaluate the performance of the SDSE algorithm with respect to the sample size, N, the number of multipaths, M and lastly the channel length L.

We shall begin our analysis by examining the makeup of the SDSE algorithm. Fig. 3.6 outlines the increment in performance generated by modules that have been integrated later into the SDSE algorithm. The theoretical algorithm outlined

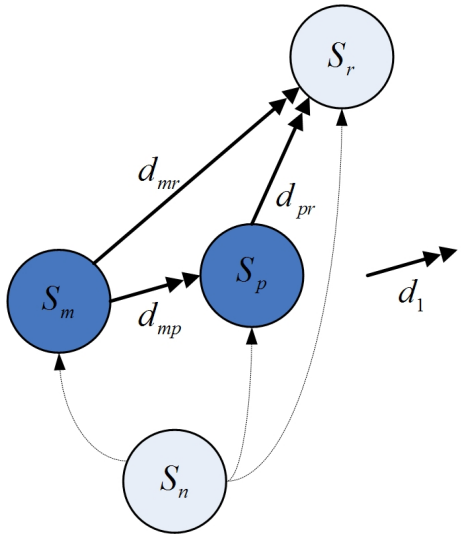

Figure 3.7: Selecting output states with $\mathbf{d}_1$

in section 3.2 does not perform on par with modern algorithms such as T Li and Z Ding's Reduced State VA[40]. Furthermore, it can be seen from Fig 3.6 that different modules have impact on different regions within the SNR spectrum. The first two modules, *expectation of the difference vector column in Table 3.2 to refine* $\boldsymbol{d}_1$ and *using* $\boldsymbol{d}_1$ *to select output states when multiple output states are available*, do not completely eliminate errors in high SNR regions. But instead, they account for a significant portion of the reduction of BER in regions of low and moderate SNRs. The functionality of the second module, *using* $\boldsymbol{d}_1$ *to select output states* is illustrated in Fig 3.7. On the other hand, correcting for single output states while effective in moderate to high SNR regions is almost ineffective in low SNR regions. This outlines the nature of the two problems, creation of single output states, and creation of multiple output states, that occur during the clustering process. Typically single output states occur in high SNR regions whereas multiple output states occur in low SNR regions.

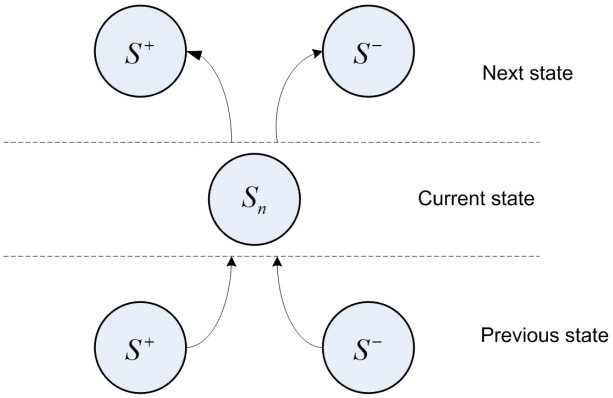

Figure 3.8: The symmetry of the state diagram

Another correction that can be implemented in the SDSE algorithm is a backward estimation process. This is introduced in *Section 3.3* as an alternative to minimizing the probability of the channel column integrated into the difference vector from fading to the noise level. The current estimation algorithm utilizes the two output states of each state to estimate the next state i.e. $S(n) \rightarrow S(n+1)$. However, due to symmetry of the state diagram, it is possible to use the two input states to estimate the previous state i.e $S(n+1) \rightarrow S(n)$. Then, performing an averaging or majority rules decision, it will be possible to increase the accuracy of our estimates. Furthermore, the symbol sequences the forward and backward algorithms disagree upon contains the erroneous symbols. This knowledge can then be used to deploy more computationally expensive algorithms to extract them. Fig 3.8 illustrates the symmetrical nature of the state diagram. This is the feature that enables the algorithm to be implemented in both forward and

backward directions. However, this correction has not been incorporated in our results as we try simultaneously to minimize the computational cost.

The performance of the SDSE algorithm with respect to the number of multipath data it utilizes is illustrated in Fig. 3.9. Here, all simulations are based on the parameters $N = 2000$ and $L = 6$. The SDSE algorithm shows a similar sensitivity as the order estimation algorithm (Fig. 2.5) to regions of low SNR. This is because the rapid deterioration of states in low SNR regions leave the state transition table incomplete, and the algorithm blind to certain bit sequences. On the other hand, extra states that occur in high SNR regions have no effect on the algorithm. This is due to the fact that these states represent redundant data. Another important aspect of the SDSE algorithm is its step-like estimating capability, whereby data can be recovered with almost no errors within one region. This can be due to the algorithms dependency on estimating the relative direction between two difference vectors. In regions of low SNR, the estimation can be erroneous leading to a deterioration in BER. On the otherhand, in high SNR, corruption of the difference vectors is minimal and it poses no threat to the determination of the relative direction between the two vectors. Another important aspect the figure illustrates is the capability of the algorithm to increase the accuracy of its estimates by increasing the number of sensors, M. This is true especially in low SNR regions. However, the gain achieved is of diminishing nature. This is evident from the results which show that beyond $M = 28$, no perceivable gain is achieved. The use of a large number of multi-paths is characteristic of both this algorithm and other spatial algorithms introduced in this

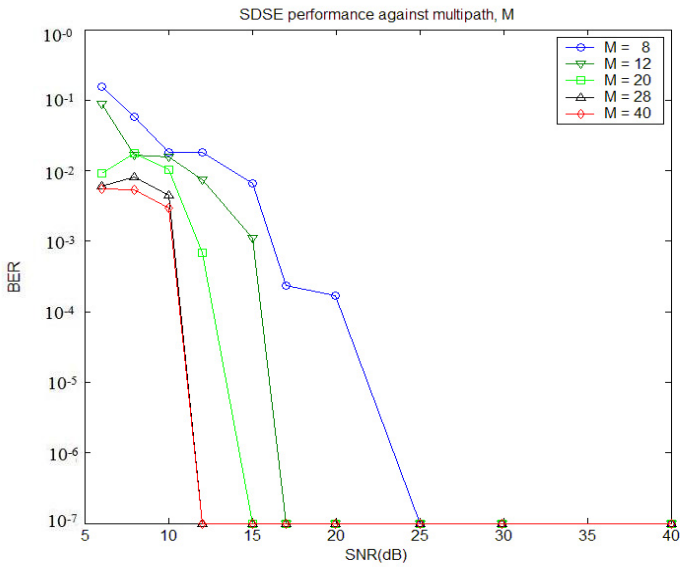

Figure 3.9: Performance of the SDSE algorithm

thesis. This however can be obtained by sampling the received signal at higher rates to create virtual channels. Thus, the physical limitation on the number of receivers is eased.

In Fig 3.10, we present the effect channel length has on the SDSE algorithm. The two parameters, channel length, L and the data set size, N are linked tightly to the SDSE algorithm by assumption*(e)*. Assumption*(e)* simply states that the state transition table needs to be complete for extracting the transmitted symbol sequence without errors. On examining Figs 3.10 and 3.11, it is clear that the SDSE algorithm displays non recoverable errors either on increasing L or decreasing N. Increasing the channel length results in an exponential increase in the number of states. Thus, the current data set no longer contain sufficient data for completing the new transition table. Reducing the size of the data set again creates a similar situation. Then, instead of the states increasing, the available

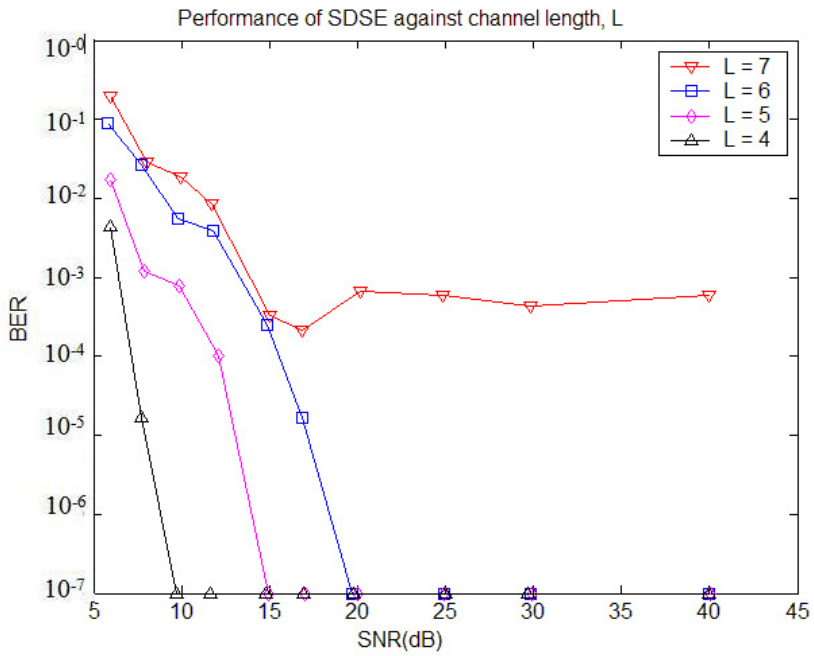

Figure 3.10: The effect of the channel length, L on SDSE

data to fill the states decreases. Both from the point of view of the algorithm are equivalent and leads to unrecoverable BER even in high SNRs. The simulations in Fig 3.10 were derived using the parameters $N = 2000$ and $M = 12$ whereas Fig 3.11 was derived using the parameters $L = 6$ and $M = 12$. One important fact that is noticeable in the SDSE algorithm is that knowledge of the channel length L does not form a necessity condition for extraction. This is because the SDSE algorithm relies on the *Primary* clustering algorithm to do spatial data processing. As mentioned in *Chapter 2*, the primary clustering algorithm does not require an estimate of the channel length to perform. Instead, it can give a rough estimate of the channel length if required. On the other hand, the bulk of the computational cost of the SDSE algorithm lies with the clustering process. This in turn is dependant on the number of states in the spatial structure, 2^L.

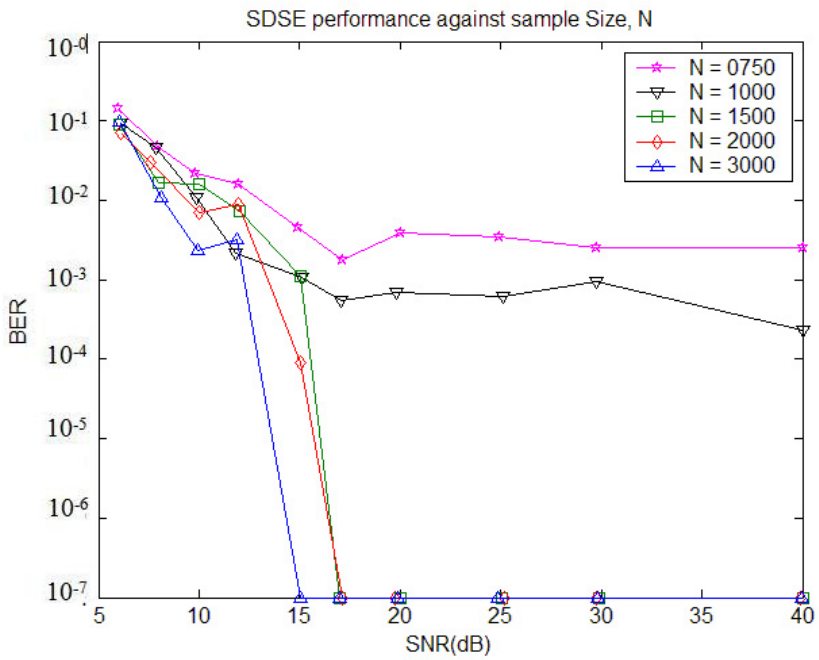

Figure 3.11: The effect of the data set size, N on SDSE

Thus, computational cost of the SDSE is roughly proportional to this value.

3.6 Summary

In this chapter, we presented a blind sequence estimation scheme based on spatial data. We began by presenting the mathematical structures that generate the state information used, and then proceed to formulate the algorithm in a procedural form. The effect of noise on the algorithm is then analyzed and following, we present an indepth analysis and discussion into the behavior of the SDSE algorithm.

In the next chapter, we present spatial algorithms belonging to another category in blind estimation. Instead of direct sequence estimation, the two algo-

rithms presented in *Chapter 4* estimate the channel parameters of the SIMO platform. The Channel Estimation by Difference Sets (CEDS) and Channel Estimation by Twin Indexing (CETI) are sibling algorithms derived from the mathematical structures found in the difference set of the spatial vector set Y.

Chapter 4

Blind Channel Estimation

4.1 Introduction

As outlined in *Chapter 3*, estimation of channel parameters is one of the key approaches to extracting transmitted data. This chapter is intended to prove that the spatial algorithms introduced in this thesis are capable of estimating SIMO FIR channels. In this chapter, we first focus on the Channel Estimation by Difference Sets (CEDS) algorithm. After formulating the CEDS algorithm in a procedural from, we then introduce the Channel Estimation by Twin Indexing (CETI) algorithm. Next, we introduce limitations of using purely spatial data and then proceed to outline how time information can be incorporated into the algorithms. Furthermore, we outline additional finite alphabet resources that are available, and finally provide an indepth analysis and discussion into the behavior of both the CEDS and CETI algorithms

4.2 Channel Estimation by Difference Sets (CEDS)

In *Chapter 3*, we examined the use of spatial algorithms for direct estimation of transmitted symbols. In this section, we present an algorithm from the opposing branch, an algorithm that first estimates the channel parameters. We begin by introducing mathematical structures, that form the basis of the CEDS algorithm. To begin understanding the structures embedded in the spatial data, we first need to define a subset of the difference vectors.

Definition: Let $\mathbf{d}$ be the difference of two spatial vectors as described in *Chapter 3*, Eqn. (3.3). Then, $\mathbf{d}$ is defined as an *elemental vector* of order p, $\mathbf{e}_p$, *if and only if* the spatial vectors generating the difference vector differ only in the p^{th} bit position of their respective *source vector* segments. That is if,

$$\begin{aligned} \mathbf{d} &\sim 0.5(S_b - S_c) \quad \text{AND} \\ S_b &\sim \mathbf{H}[a_L, ..., a_p, ..., a_0]' \\ S_c &\sim \mathbf{H}[a_L, ..., -1 \times a_p, ..., a_0]' \end{aligned}$$

then,

$$\begin{aligned} \mathbf{e}_p &\triangleq \mathbf{d} \\ &= a_p[h_{1p}, h_{2p}, ..., h_{Mp}]' \end{aligned} \tag{4.1}$$

where $\{a_i\} \in \{-1, +1\}$.

Using the above definition as an example, consider the state S_b. Changing the

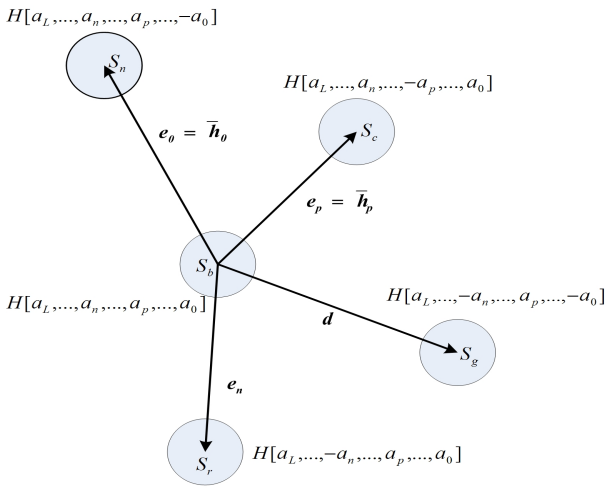

Figure 4.1: Elemental vector structure

sign of any one symbol, a_p in its *source vector* segment results in forming another spatial vector corresponding to a different state, S_c. Moreover, the difference vector generated between the two states will be an *elemental vector* of order p as defined above. This structure creates the basis for our estimation algorithm to operate on. If difference vectors were to be calculated with respect to a given state S_b, then at least one vector will be an *elemental vector* of order p. Thus, for a channel of length $L+1$, $L+1$ unique *elemental vectors* exist. This is shown in Fig 4.1. The figure illustrates spatial states that form *elemental vectors* when *difference vectors* are computed. Consequently if *difference vectors* were to be generated for the complete set of states,

$$V \triangleq \{v|v = S_i \;\; i \in \{1, ..., 2^{L+1}\}\} \tag{4.2}$$

each of the 2^{L+1} *spatial vectors* associated with each state will contribute $L+1$

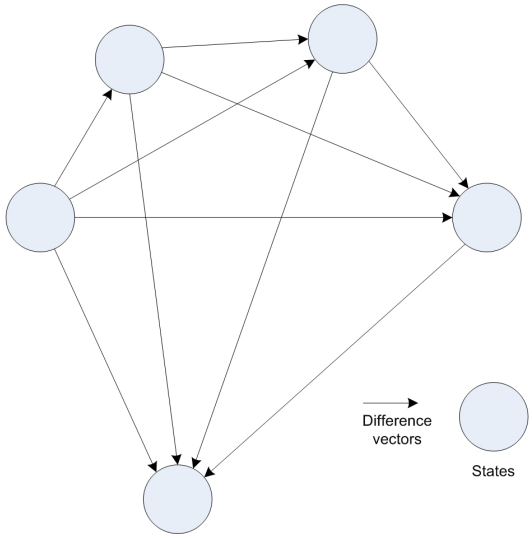

Figure 4.2: Elemental vector structure

elemental vectors. In other words, 2^{L+1} copies of each of the unique $L+1$ *elemental vectors* will exist in the difference vector set,

$$D = \{\mathbf{d} | \mathbf{d} = 0.5(\mathbf{y}_i - \mathbf{y}_j) \quad \{i, j\} \in \{1, ..., 2^{L+1}\} \quad i \neq j\} \tag{4.3}$$

where $\mathbf{y}_i \sim S_i$ and $\mathbf{d} \sim 0.5(S_i - S_j) \Rightarrow \mathbf{d} = 0.5(\mathbf{y}_i - \mathbf{y}_j)$. The generation of the difference set is illustrated in Fig 4.2. The arrows in the figure indicate unique difference vectors.

However, vectors that are not *elemental vectors*, i.e. vectors generated by states differing by more than one bit in their source vector segments will be less populous. For a difference vector resulting from q bit differences in the *source vector* segments, the maximum number of identical vectors that can be created is upper bound by

$$N_q = 2^{L-q+2} \tag{4.4}$$

The *elemental vector* families in D will be more populous, and this provides the key to their identification and consequent extraction by clustering algorithms. Of the *elemental vectors*, Eqn. (4.1) indicates that they are in fact channel coefficients. More precisely, they are columns of $\mathbf{H}$. Thus, the extracted vector set would be essentially the channel matrix, albeit having sign and order permutation ambiguities in the columns.

The ambiguities results from not knowing the time order of the channel vectors extracted. Sign and permutation ambiguities can be resolved later using time data in a post processing step. One implementation of a post processing algorithm is described under the "Correcting CEDS" sub-section. Let the matrix thus extracted be denoted by $\tilde{\mathbf{H}}$. We can now summarize our algorithm as follows:

4.2.1 The CEDS algorithm

i) Use the primary clustering algorithm to extract an estimate of Y, to $\hat{Y}$ from the input data vectors $\mathbf{x}(n) \;\; n \in \{1, ..., N\}$

ii) Generate the difference vector set, D from the estimated vector set $\hat{Y}$.

$$D \triangleq \{\tilde{\mathbf{d}} | \tilde{\mathbf{d}} = 0.5(\hat{\mathbf{y}}_i - \hat{\mathbf{y}}_j) \;\; i, j \in \{1, ..., 2^{L+1}\}, \; i \neq j\} \tag{4.5}$$

iii) Use population relationships to extract elemental vectors by applying the secondary clustering algorithm to D.

iv) Use post processing to correct sign and permutation ambiguities.

4.3 Channel Estimation by Twin Indexing (CETI)

In this section, we will be introducing another channel estimation scheme that relies exclusively on spatial data. The CETI algorithm is computationally cheaper than the CEDS algorithm. However, it doesn not supplant the CEDS algorithm. This is due to the fact that the two algorithms were designed to work in different environments. The CEDS algorithm is useful in estimating slowly varying channels in lower SNRs. The CETI on the otherhand can handle faster fading channels. However, it admittedly requires a higher SNR to work. This, we quantify in the *results and discussion* section. There, we show that the CETI algorithm relies on a substantially smaller data set compared to the CEDS algorithm. This inturn implies that the channel has to be relatively stationary for a shorter time compared to the channel CEDS requires. On the other hand the CEDS algorithm outperforms the CETI, even more prominently in low SNR regions. This is evident from Fig 4.9.

We begin the introduction into the CETI algorithm by introducing the mathematical concepts and structures that power it. In *Chapter2*, *Section 2.5*, under the sub section *The Deterministic Indices*, we introduced three elements $o_{i\alpha}, o_{i\beta}$ and $o_{i\gamma}$ that are formed when the multi dimensional spatial structure is projected onto a 1-dimensional axis. By using spatial algorithms, vectors $\mathbf{f}_{i\alpha}$, $\mathbf{f}_{i\beta}$ and $\mathbf{f}_{i\gamma}$ containing $o_{i\alpha}, o_{i\beta}$ and $o_{i\gamma}$ can be easily extracted. These vectors and indices form the core of the CETI estimation algorithm. To begin formulating the CETI

algorithm, consider the noiseless SIMO model,

$$\mathbf{y}(n)_{M\times 1} = \mathbf{H}_{M\times(L+1)}\mathbf{s}(n)_{(L+1)\times 1} \tag{4.6}$$

We can use the deterministic elements $o_{i\alpha}, o_{i\beta}$ and $o_{i\gamma}$ with respect to any of the M SISO sub-channels contained within the SIMO channel for estimation. To begin, consider the i^{th} SISO sub-channel. Equations (2.21), (2.22) and (2.23) indicate that the source vectors, $\mathbf{s}_\alpha$, $\mathbf{s}_\beta$ and $\mathbf{s}_\gamma$ generate the i^{th} element of the vectors $\mathbf{f}_{i\alpha}, \mathbf{f}_{i\beta}$ and $\mathbf{f}_{i\gamma}$. Since we are working within the SIMO model, these source vectors contribute to generating all other elements contained within the vectors $\mathbf{f}_{i\alpha}, \mathbf{f}_{i\beta}$ and $\mathbf{f}_{i\gamma}$. i.e.,

$$\mathbf{f}_{i\alpha} = \mathbf{H}[sgn(h_{i1}), ..., sgn(h_{iL})]' \tag{4.7}$$

$$\mathbf{f}_{i\beta} = \mathbf{H}[sgn(h_{i1}), ..., -sgn(h_{iv}), ..., sgn(h_{iL})]' \tag{4.8}$$

$$\mathbf{f}_{i\gamma} = \mathbf{H}[sgn(h_{i1}), ..., -sgn(h_{iu}), ..., sgn(h_{iL})]' \tag{4.9}$$

Of the vectors $\mathbf{f}_{i\alpha}, \mathbf{f}_{i\beta}$ and $\mathbf{f}_{i\gamma}$ extracted by the secondary clustering algorithm, probability implies that the vector corresponding to the element $o_{i\alpha}$ will be more populous. This can be seen from the Fig 4.3. If the indexes $o_{i\alpha}, o_{i\beta}$ and $o_{i\gamma}$ have the distributions of $\wp(o_{i\alpha})$, $\wp(o_{i\beta})$ and $\wp(o_{i\gamma})$,

$$\int_T^\infty \wp(o_{i\alpha}) > \int_T^\infty \wp(o_{i\beta}) \tag{4.10}$$

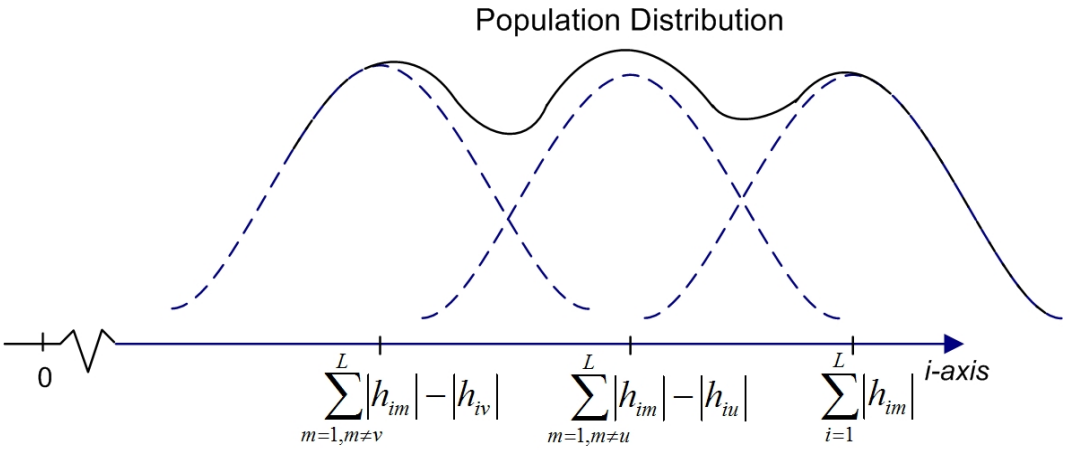

Figure 4.3: Elemental vector structure

By using this information, $\mathbf{f}_{i\alpha}$ can be identified. Now, define the *difference vectors*,

$$\begin{aligned} \mathbf{d}_{i\alpha\beta} &\triangleq 0.5(\mathbf{f}_{i\alpha} - \mathbf{f}_{i\beta}) \\ &= sgn(h_{iv})[h_{1v}, h_{2v}, ..., h_{Mv}]' \qquad (4.11) \\ \mathbf{d}_{i\alpha\gamma} &\triangleq 0.5(\mathbf{f}_{i\alpha} - \mathbf{f}_{i\gamma}) \\ &= sgn(h_{iu})[h_{1u}, h_{2u}, ..., h_{Mu}]' \qquad (4.12) \end{aligned}$$

As can be seen, the two *difference vectors* contain two unique columns of $\mathbf{H}$ indexed by the two smallest elements of the i^{th} sub-channel. We shall call these two elements the *Least Significant Elements (LSE)* in our thesis. To complete the formulation of the CETI algorithm, we shall undertake the additional assumption stated below in addition to the general assumptions stated in *Chapter 2*

e) Sufficient multipath channels have been chosen to ensure complete extrac-

tion of all channel columns,

> The CETI algorithm can only extract a channel column if in that channel column there exists an element that is one of the two smallest element of a channel row. Furthermore, for the secondary clustering algorithm to be able to extract a channel column, it must be more populous compared to erroneous vectors. These two conditions place a lower bound to the number of multipaths that is needed for estimation.

holds true in addition to the general assumptions stated at the beginning of this thesis. If M multipaths are used, and the secondary clustering algorithm requires a minimum population of Q vectors, the probability of extracting a channel column can be described by,

$$\begin{aligned} P_E &\triangleq 1 - P\Big(\text{Less than } Q \text{ LSEs found}\Big) \\ &= 1 - \sum_{r=0}^{Q-1} {}^{M}C_r [2/L]^r [1-2/L]^{M-r} \end{aligned} \tag{4.13}$$

where the probability of a LSE occurring is $2/L$. ${}^{M}C_r$, here is the binomial coefficient given by,

$${}^{M}C_r = \frac{M!}{(M-r)!r!} \tag{4.14}$$

and the probability of finding exactly r LSE is given by

$$P(r) = {}^{M}C_r [2/L]^r [1-2/L]^{M-r} \tag{4.15}$$

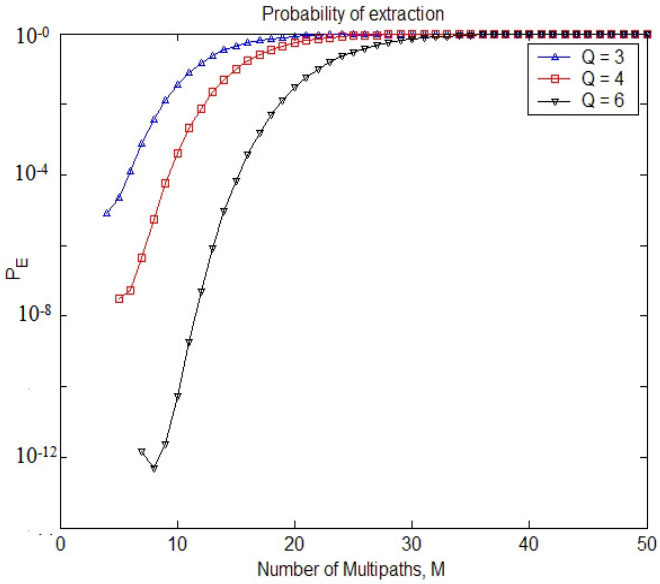

Figure 4.4: Probability of extraction of channel columns

.

Choosing a large number of multipaths has two distinct advantages. First, as the smallest elements of each of the sub-channels are randomly distributed, this ensures the probability that all columns will be indexed at least once approaches unity. Secondly, it increases the probability of the channel columns being indexed more than once. This is illustrated in Fig 4.4. Thus, column vectors will be more populous than any erroneous vectors that occur. This population relationship forms the basis for identification of the channel columns. They can then be extracted using the secondary clustering algorithm.

Table 4.1 shows the structure of the channel highlighting structures that the CETI algorithm utilizes. The coefficients in italic (h_{ij}) indicate general channel coefficients while values in Sans Serif (h_{ij}) indicate the two smallest coefficients in each channel. Thus, in each row, the CETI algorithm is able to extract the

Channel Columns						Extractable Columns
$\bar{\mathbf{h}}_1$	$\bar{\mathbf{h}}_2$	$\bar{\mathbf{h}}_3$	$\bar{\mathbf{h}}_4$	$\cdots$	$\bar{\mathbf{h}}_L$	
h_{11}	h_{12}	h_{13}	h_{14}	$\cdots$	h_{1L}	$\bar{\mathbf{h}}_3,\ \bar{\mathbf{h}}_4$
h_{21}	h_{22}	h_{23}	h_{24}	$\cdots$	h_{2L}	$\bar{\mathbf{h}}_1,\ \bar{\mathbf{h}}_3$
h_{31}	h_{32}	h_{33}	h_{34}	$\cdots$	h_{3L}	$\bar{\mathbf{h}}_2,\ \bar{\mathbf{h}}_L$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
h_{M1}	h_{M2}	h_{M3}	h_{M4}	$\cdots$	h_{ML}	$\bar{\mathbf{h}}_1,\ \bar{\mathbf{h}}_4$
n_1	n_2	n_3	n_4	$\cdots$	n_L	Nos. minimums

Table 4.1: Twin indexing through channel coefficients

channel columns indexed by the elements printed in Sans Serif. In the first row, for example, h_{13} and h_{14} appear in Sans Serif. Thus, when processing the 1st row the channel columns $\bar{\mathbf{h}}_3$ and $\bar{\mathbf{h}}_4$ are extractable. The last row indicates the number of times the given channel column has been extracted. Using the table as a reference, the CETI algorithm outlined in the next section can be easily understood.

4.3.1 The CETI algorithm

i) Scanning the data $\mathbf{x}(n)$, extract vectors having the largest $3N/2^L$ elements at the i^{th} position, $[\mathbf{x}(n)]_i$ into the subset F_i.

This step extracts the vectors corresponding to the largest three elements, $o_{i\alpha}, o_{i\beta}$ and $o_{i\gamma}$.

ii) Use the secondary clustering algorithm to extract the vectors $\mathbf{f}_{i\alpha}, \mathbf{f}_{i\beta}$ and $\mathbf{f}_{i\gamma}$ from F_i.

iii) Identify the more populous vector, $\mathbf{f}_{i\alpha}$.

iv) Create the two difference vectors $\mathbf{d}_{i\alpha\beta}$ and $\mathbf{d}_{i\alpha\gamma}$

v) Add the difference vectors to the set D, and repeat steps *(a)* - *(d)* till all SISO sub-channels have been processed.

$$D = \{\mathbf{d} | \mathbf{d} = \{\mathbf{d}_{i\alpha\beta}, \mathbf{d}_{i\alpha\gamma}\} \;\; i \in \{1, ..., M\}\} \tag{4.16}$$

vi) Use the secondary clustering algorithm in context of D and extract $L+1$ vectors corresponding to the most populous families.

vii) Correct sign and permutation ambiguities using the post-processing algorithm as outlined in the next sub-section.

4.4 Improving and correcting CEDS and CETI

In this subsection, we attempt to rectify the spatial algorithms main drawback, its blindness to time. This is done by integrating a module that can incorporate time data and resolve the ambiguities resulting from its absense. However, the module presented is not a part of the spatial algorithm and more refined schemes may be available to the same end without the additional constraints this algorithm imposes. Then, we proceed to present another module that has been developed and integrated into the CEDS and CETI algorithms. This module uses finite alphabet data in a distinctively different manner to examine the accuracy of the extracted channel columns.

4.4.1 Sign and Permutation Correction

In comparison to the SDSE algorithm introduced in *Chapter 3*, both the CETI and CEDS algorithms rely purely on spatial data. The SDSE algorithm incorporates time data when building the time indexed state array. Time is intangible in the spatial domain. Thus, the spatial algorithms presented in this chapter are blind to ordering of data in the time domain. In our algorithms, time blindness manifests as sign and permutation ambiguities. These result from not being able to know the order or sign of the channel columns extracted with respect to the channel matrix $\mathbf{H}$. The sign ambiguity is a the factor, $\{+1, -1\}$ each column need to be multiplied before fitting onto $\mathbf{H}$. However, these ambiguities can be resolved by reprocessing the extracted information with time rich data.

To begin, let us assume $\tilde{\mathbf{H}}$ to be the extracted channel matrix containing both sign and permutation ambiguities. We can now define,

$$\mathbf{v}(n) \triangleq \tilde{\mathbf{H}}^{\dagger}\mathbf{x}(n) \tag{4.17}$$

$$= \mathbf{P}\mathbf{s}_n \tag{4.18}$$

a pseudo input vector $\mathbf{v}(n) = [v_0(n), ..., v_L(n)]'$. This step is the primary reason for the inclusion of assumption*(d)*. Other algorithms may exist, that can resolve the time dependant ambiguity without adding such restrictions. Under noiseless conditions, these ambiguities can be represented using a permutation matrix $\mathbf{P}$. The matrix $\mathbf{P} = [\mathbf{p}_1, ..., \mathbf{p}_M]'$ is defined with rows $\mathbf{p}_i = [0, ..., p_{m_i}, ...0]$, where $m_i \in \{0, ..., L\}$ is the position of the non-zero element $p_{m_i} \in \{\text{-}1, +1\}$. By integrating the structure of $\mathbf{P}$, $\mathbf{v}(n)$ can be expanded to,

$$\mathbf{v}(n) = [p_{m_0}s_{n-m_0}, p_{m_1}s_{n-m_1}, ..., p_{m_L}s_{n-m_L}]' \tag{4.19}$$

Now consider the relative vector defined as,

$$\mathbf{r}_j(n) \triangleq \mathbf{v}(n)./v_j(n+1) \tag{4.20}$$

$$= \left[\frac{p_{m_0}s_{n-m_0}}{p_{m_j}s_{n-m_j+1}}, ..., \frac{p_{m_L}s_{n-m_L}}{p_{m_j}s_{n-m_j+1}}\right] \tag{4.21}$$

Both $\mathbf{v}(n)$ and $\mathbf{v}(n+1)$ share $L/(L+1)$ elements. Thus, the vector $\mathbf{r}_j(n)$ can be used to extract information of the elements that are not shared commonly by

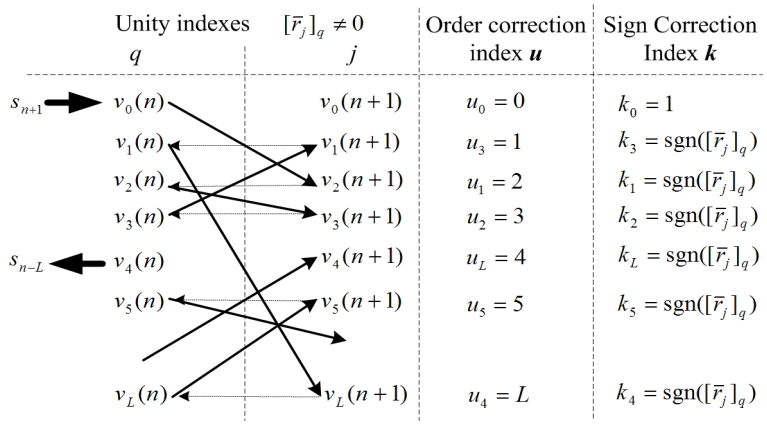

Figure 4.5: Symbol transition decoding for permutation correction

$\mathbf{v}(n)$ and $\mathbf{v}(n+1)$. Now consider the elements of $\mathbf{r}_j(n)$. From Eqn. (4.17), it is evident they can be represented by,

$$[\mathbf{r}_j(n)]_q = \begin{cases} \frac{p_{m_q} s_{n-m_q}}{p_{m_j} s_{n-m_j+1}} & m_q \neq m_j + 1 \\ \frac{p_{m_q}}{p_{m_j}} & m_q = m_j + 1 \end{cases} \tag{4.22}$$

for $q \in \{0, ..., L\}$. This is the structure that lets us isolate the element in $\mathbf{v}(n)$ that is not shared by $\mathbf{v}(n+1)$. Furthermore, it yields information as how symbols rearrange themselves as they transfer from $\mathbf{v}(n) \rightarrow \mathbf{v}(n+1)$. This is the inherent permutation. Now, consider the expected value of $\mathbf{r}_j(n)$ across the time index $n \in \{1, ..., N\}$.

$$\bar{\mathbf{r}}_j = \Big\langle \mathbf{r}_j(n) \Big\rangle_n \tag{4.23}$$

Note that the element $[\mathbf{r}_j(n)]_q$ when $m_q = m_j + 1$ is only dependant on the permutation matrix coefficients. This implies that a common element, $[\mathbf{v}(n)]_q = [\mathbf{v}(n+1)]_j$ exists. Thus $\bar{\mathbf{r}}_j$ will contain one non-zero element provided $m_q = m_j + 1$

is satisfied. Moreover, since $\mathbf{v}(n)$ and $\mathbf{v}(n+1)$ share L common elements, the above condition holds for L instances of $\bar{\mathbf{r}}_j \quad j \in \{0,..,L\}$. The j and q indices where non-zero elements occur impart useful information. It states that a symbol at the q^{th} position in $\mathbf{v}(n)$ is transformed to the j^{th} position in $\mathbf{v}(n+1)$. When all transformations are known, the permutation matrix can be corrected.

We begin by constructing the transition diagram shown in Fig. 4.5. In the figure, solid arrows point from the q index to the j index at the occurrence of each non-zero element. The direction of the arrows show how a symbol is affected by the channel columns with respect to time. Once complete, correcting indices $\mathbf{u}$ and $\mathbf{k}$ can be extracted as shown in Fig. 4.5. The time order of the columns can be extracted from how the arrows index the j column. e.g. $v_0 \rightarrow v_2 \rightarrow v_1 \rightarrow v_L \rightarrow v_5 \rightarrow ... \rightarrow v_4$. This creates the order for the correcting index $\mathbf{u} = [u_0, u_1, u_2, ..., u_l]$. The sign of $[\mathbf{r}_j(n)]_q$ at each instance forms the elements of the sign correcting index $\mathbf{k}$. However, when $s_{n-m_j+1} = s_{n+1}$ the condition $m_q = m_j + 1$ cannot be met, and the vector $\bar{\mathbf{r}}_j$ will be identically $\mathbf{0}$. This occur at the entry of a symbol into the channel, as then, the new symbol is not shared between $\mathbf{v}(n)$ and $\mathbf{v}(n+1)$. It also serves as the starting point to the correcting indices. With the two correcting indices, the correcting matrix can be formulated as the diagonal matrix $\mathbf{C}_T$ given by,

$$\mathbf{C}_T = Diag\left[u_0k_0, u_1k_0k_1, u_2k_0k_1k_2, ..., u_L\prod_{i=0}^{L} k_i\right] \tag{4.24}$$

4.4.2 Cost based Heuristic search (CBHS)

The population based vector identification criteria used in the separation of channel vectors for both CEDS and CETI algorithms provide accurate results in regions of moderate and good SNR. But in lower SNR regions, the population statistics become unreliable and the extraction of channel vectors become problematic. Under such conditions, the finite alphabet data provides another mechanism to test the extracted vectors for accuracy. Using the knowledge that the extracted vectors are in fact channel columns, it is possible to derive a function to verify that the extracted vectors do infact constitute a valid channel matrix. To begin, let $\mathbf{v}(n)$ be a pseudo estimate of the source vector as defined in Eqn. (4.13). All elements of $\mathbf{v}(n)$ should ideally belong to either $\{+1\}$ or $\{-1\}$. This is the structure we use to generate our cost function. To begin, define the matrix

$$\mathbf{V} \triangleq [\mathbf{v}(n), \mathbf{v}(n+1), ..., \mathbf{v}(n+K)] \tag{4.25}$$

where K is a stacking factor. The cost function is then described by,

$$C(\texttt{M,L,N,SNR}) = sum\Big\{ sum\Big[square_e\Big(abs_e(V) - ones(K+1, L+1)\Big)\Big]\Big\} \tag{4.26}$$

and is used to give a measure of the "goodness" of the estimated channel matrix. It essentially measures the deviation of symbols extracted using this channel matrix against the transmitter alphabet. Ideally, symbols extracted should belong to the transmitter alphabet. The cumulated deviation of the symbols from the

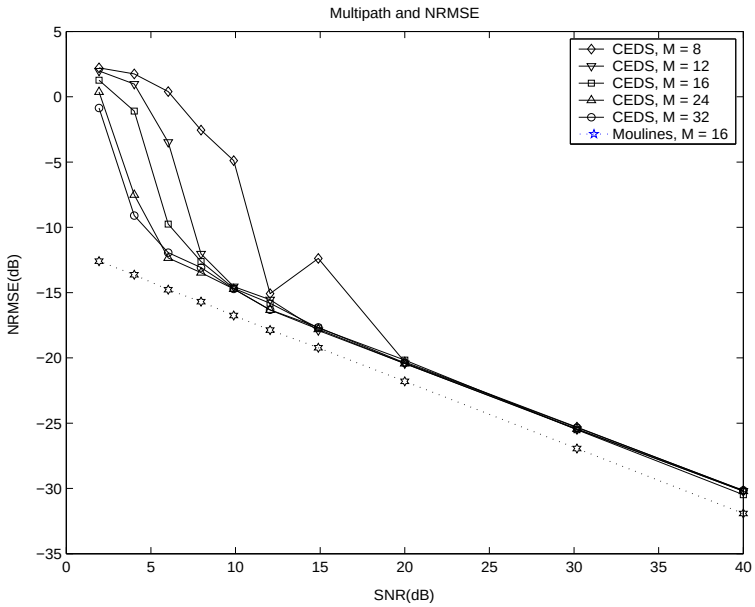

Figure 4.6: Performance of the CEDS algorithm

transmitter alphabet forms the cost. The cost function depends on the three parameters N, L and M in addition to noise. In the equation above, the function $ones(K, L)$ defines a matrix of dimension $K \times L$ having $\{+1\}$ as elements. Furthermore, the operators $abs_e(\mathbf{M})$ and $square_e(\mathbf{M})$ are used to denote element by element operators that perform the absolute and squaring operations respectively on the matrix $\mathbf{M}$. Using Monte-Carlo iterations over all M-SNR regions while keeping N and L constant, it is possible to extract the mean and range of the cost function at each point. Then, using these tabulated values as a reference, it is possible to identify a badly extracted $\tilde{\mathbf{H}}$ matrix as then, the cost function will exceeds the tabulated range. These erroneous matrices can then be corrected using $C(\texttt{M,L,NSNR})$ as a benchmark and applying the following algorithm,

- Extract N_v vector families having a population greater than N_1/k_{POP} from the set D used in the CETI and CEDS algorithms into the set $J = [\mathbf{j}_1, ..., \mathbf{j}_{N_v}]$. $N_1 = 2^{L+1}$ by Eqn. (4.4).

  This essentially extracts additional vectors when extracting channel columns in both CEDS and CETI algorithms. When the cost function indicates that the most populous vectors are not channel columns, the additional vectors can then be searched to find channel columns

- Rearrange the columns of J to the increasing order of the population of the vectors. Set $\tilde{\mathbf{H}} = [\tilde{\mathbf{h}}_0, ..., \tilde{\mathbf{h}}_L]$ to the most populous vectors in J and estimate its cost.

  The population of an extracted vector is an indicator to the probability of it being a channel column. Thus, searching for channel columns first among the more populous vectors saves time.

- Use the algorithm outlined by the pseudo code

  The algorithm described below attempts estimate the cost of all permutations of possible channel matrices from the most probable to the least probable. It saves the matrix if the computed cost is lower than a given percentage of the previous cost. Not included here, but more practical, is an exit mechanism that can be used

to exit the exhaustive search when an acceptable channel matrix is found. This can be done by simply checking the cost to see if it falls within the tabulated ranges and exiting.

`Check every column in` $\tilde{\mathbf{H}}$

For $i = 0$ to L

`Against all vectors extracted`

For $j = 1$ to N_v

`Replace the` i^{th} `column with the vector in the` j^{th} `position`

$\breve{\mathbf{H}} = \tilde{\mathbf{H}}$; $\breve{\mathbf{H}}[i, :] = \mathbf{j}_j$;

`Re-estimate into` C_{NEW}

`If the new cost <` $1/k_C$ `of the old cost, save channel data`

If $C_{NEW} < C_{OLD}/k_C$

$C_{OLD} = C_{NEW}$; $\acute{\mathbf{H}} = \breve{\mathbf{H}}$;

End

End

End

The constants k_{POP} and k_C were estimated empirically using Monte-Carlo iterations to minimize the errors of extracted $\mathbf{H}$ matrices.

4.5 Results and Discussion

The channel model we used in our simulations was a stochastic SIMO model, with impulse parameters modeled as zero mean Gaussian processes having unit variances. Channel coefficients and noise are assumed identically and independently distributed, and in this simulation noise was modeled as a zero mean Gaussian process. For comparison purposes, we benchmark different aspects of our algorithm against the reference system described below. For the reference system, a channel length of $L = 6$ was selected with $M = 16$ receivers, and the results obtained using a data set of $N = 2000$ samples per iteration. This system was chosen so as to be a common denominator for both CEDS and CETI algorithms. The source symbols were generated from a alphabet of $\{+1, -1\}$ with equal probability. Finally the results obtained were then averaged over 30 Monte-Carlo iterations. In our simulations, 30 Monte Carlo iterations proved sufficient to present the stochastic performance of the algorithms.

The performance of the CEDS algorithm in noisy environments is illustrated in Fig 4.6. It shows the deviation of the performance of the CEDS algorithm with respect to the number of multipaths utilized. In comparison to the SDSE algorithm outlined in Chapter 3, the CEDS algorithm appears to be more robust in regions of low SNRs. From Fig 3.9, it is evident that the SDSE algorithm develops unrecoverable errors when the SNR falls below 10dB. On the other hand, for the same number of multipaths, the CEDS maintains its asymptotic performance well below 5dB. This maybe because while the SDSE algorithm is dependant on a

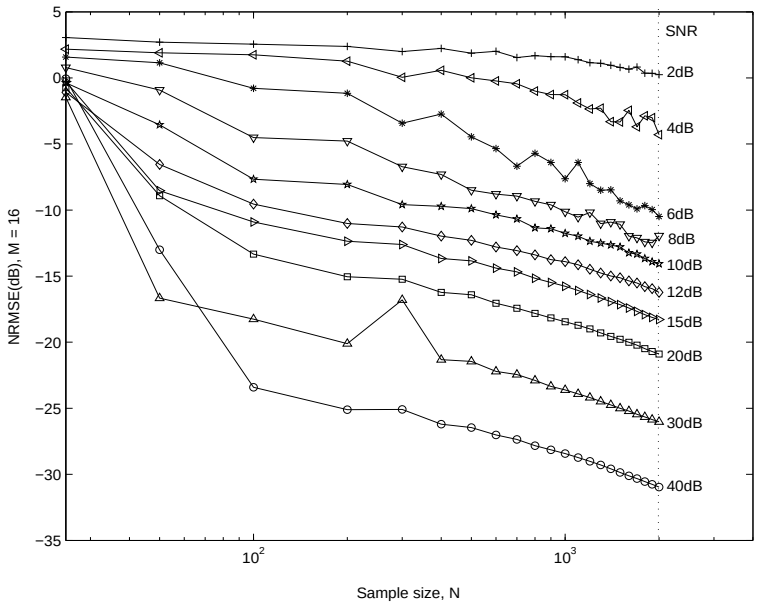

Figure 4.7: The CEDS algorithm as a function of the data set, N

derived state transition table, the CEDS algorithm relies on relative populations of vector families within a difference vector set. The transition table depends on the accuracy of assigning received vector data to extracted states. On the other hand, the secondary clustering algorithm used in the CEDS relies on the relatively population of channel columns in the difference vector set D. Noise can corrupt received vectors, so that they maybe spatially located closer to erroneous states. This induces a corruption in the state table. On the otherhand, the effect of noise on the population statistics of the set D is less felt. For one, elemental vectors have the largest population, exceeding the next population strata by a factor of 2. Such, even if a few vectors are corrupted beyond recognition, the relative population relationships still hold. In addition, as with the SDSE algorithm, increasing the number of sensors allows the algorithm to perform in lower

SNR regions with a lower NRMSE. NRMSE in our thesis is defined as

$$NRMSE = \frac{1}{M} \frac{\left[\sum_{i=1}^{M} |\mathbf{h}_i - \tilde{\mathbf{h}}_i|\right]^{1/2}}{\left[\sum_{i=1}^{M} |\mathbf{h}_i|\right]^{1/2}} \tag{4.27}$$

where $\mathbf{h}_i$ and $\tilde{\mathbf{h}}_i$ are the channel and estimated channel rows respectively. Also shown in Fig 4.6 is Moulines' subspace algorithm[33]. Moulines' algorithms is based on subspace processing of data and as a deterministic algorithm provides a good platform to compare our results. It should be noted that though Moulines' algorithm has an advantage of about 3dB, the rate of deterioration of accuracy of the channel estimates is almost identical. Furthermore, it should be kept in mind that the performance statistics presented in this thesis are dependant on the clustering algorithm used. Clustering algorithms were not the thrust of our research, therefore other clustering algorithms such as fuzzy clustering and neural network based clustering when incorporated into our algorithm, may help exceed its current limitation. However, the current algorithm utilizes data that Moulines algorithm ignores. Thus, it will be possible to create more robust algorithms by creating hybrids incorporating spatial data estimation along with time data.

Fig 4.7 illustrates another aspect of the CEDS algorithm: its dependance on the size of the data set used for estimation. While the CEDS algorithm performs acceptably above a sample size of about 100 for a channel of $L = 6$, the performance deteriorates rapidly when using smaller data sets. This is more felt in higher SNRs, and in Fig 4.7 we can see the performance of the simulations above an SNR of 15dB beginning to collapse as they pass the $N = 100$ boundary.

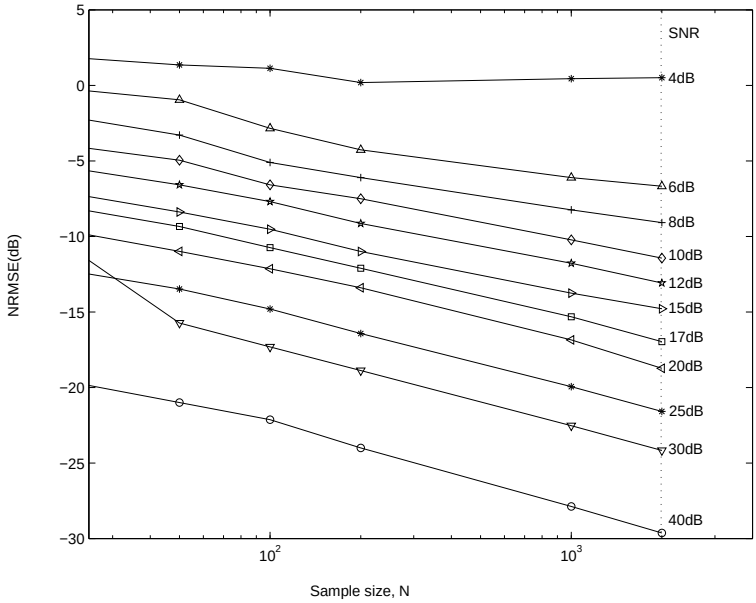

Figure 4.8: The CETI algorithm's reliance on the data set size, N

The CETI algorithm on the other hand is more resilient to smaller data sets. Fig 4.8 shows the performance of the CETI algorithm as a function of the data set used. It shows the CETI algorithm maintaining its asymptotic performance well below 100 samples to almost 20.

The projection of the spatial structure onto a single axis condenses input data vectors. Thus, the CETI algorithm is able to perform acceptably with even a data set size of 20 samples per estimate. Such a small data set is not capable of generating the complete spatial structure, which in the simulations described above has 64 points in M dimensional space. However, the CETI condenses available data, generating the information outlined in Fig 4.3. The short data length requirement is a very useful feature. It makes the CETI algorithm capable of estimating time varying channels having a small quasi stationary window, which in our simulation model maybe as small as 20 symbols.

However, the CETI algorithm relies heavily on multipath data to infuse information. Such, the algorithm needs a large number of sensors ($> 2L$) before it is able to extract all channel columns. Even so, the CETI algorithm utilizes only a portion of the data set for estimation. This implies a lower computational complexity in the execution of the algorithm. Fig. 4.9 outlines the detrimental effect this has on estimation. It shows the deterioration of the CETI algorithm as compared to the CEDS algorithm. Coupled to the algorithm's poor performance in regions of low SNR is its reliance on the smallest elements of each SISO channel. These coefficients are more vulnerable to noise. Thus, in regions of high SNR the percentage corruption of these elements would be low. On the other hand, in noisy environments, the magnitude of noise may even surpass the coefficients. This results in the algorithm losing its accuracy. However, the CETI algorithm is proposed to be used in medium and high SNR regions. In these regions, it can operate on extremely small data sets with lower computational requirements than the CEDS. Also shown in Fig 4.9 is Tongs' SOS algorithm[12]. From it, we can draw the same conclusion. The CETI outperforms the SOS algorithm in high SNRs. This maybe due to the finite sample convergence property, FA algorithms share with their deterministic brethren. However, as the SNR deteriorates, the SOS algorithm proves superior. This however, is not relatively large when compared to the CEDS algorithm. We can see both CEDS and Tong's algorithm beginning to deteriorate more rapidly on passing the 10dB threshold, and yet they are still within 3-4dB of one another.

Fig 4.10 shows the effect of using the Cost Based Heuristic Search (CBHS)

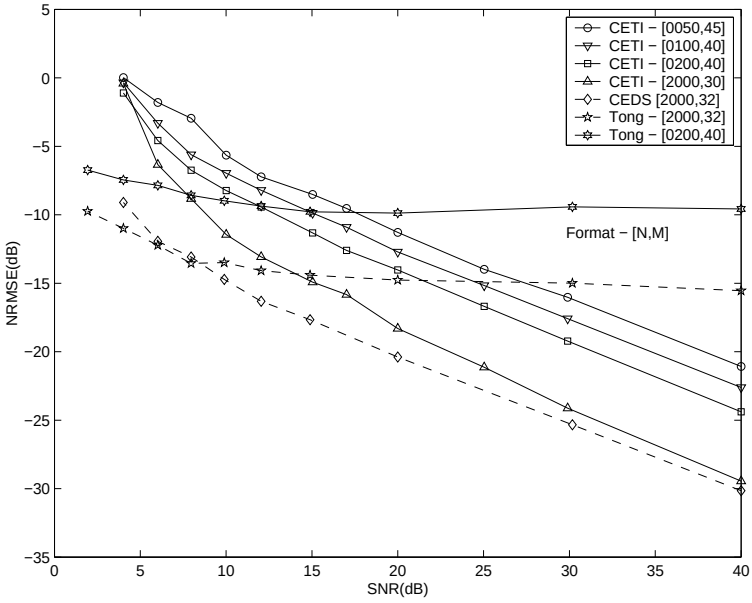

Figure 4.9: The CETI algorithm

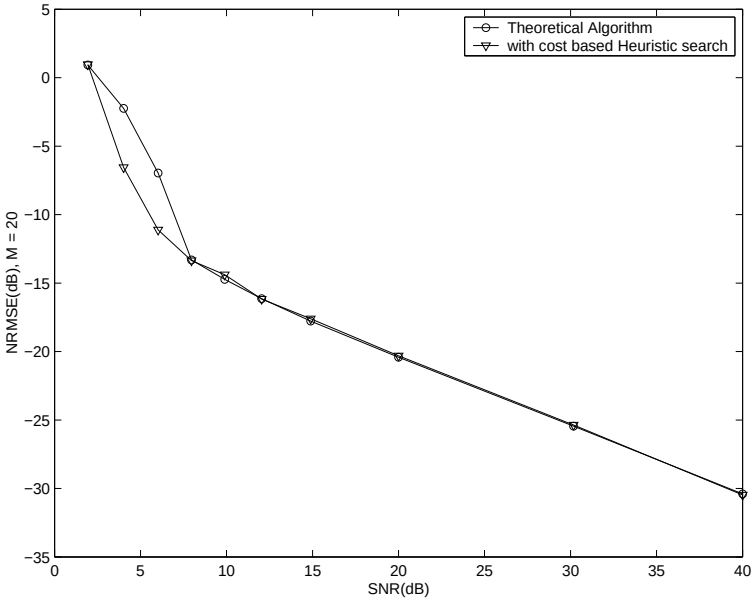

Figure 4.10: the CBHS module

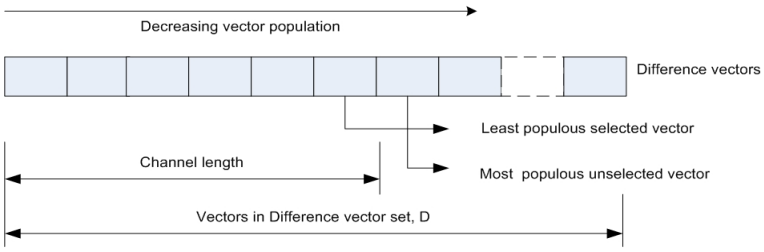

Figure 4.11: Difference vector set structure

module on the CEDS algorithm. While the increase in accuracy is significantly lower than for the SDSE algorithm, it does make the algorithm operate more efficiently in lower SNR regions. Not outlined in the error recovery section, but utilized in the algorithm is a pseudo adaptive component that dynamically increases the threshold distance D_1 of the secondary clustering algorithm to ensure proper vector extraction. The distance threshold D_1 is dynamically increased if the population difference between the least populous vector extracted and the most populous vector not extracted differs by less than two. The structure described is illustrated in Fig 4.11, and this helps eliminate erroneous sub clustering that may occur even in regions of moderate and high SNRs.

4.6 Summary

In this chapter, we presented channel estimation algorithms relying on spatial data as the primary source of information. The two algorithms presented, Channel Estimation by Difference Sets (CEDS) and the Channel Estimation by Twin Indexing (CETI) process spatial data in two unique methods. Thus, the

algorithms differ in their strengths and weaknesses with respect to one another. The CEDS algorithm is able to derive the channel coefficients with admittedly higher accuracy than the CETI algorithm. Yet, the CETI algorithm cannot be ruled out as inadequate. Its reliance on a smaller data set coupled with its lower computational costs makes it attractive in high SNR regions.

In the next chapter, we shall extend our algorithms, relaxing some of the key assumptions taken in the beginning of this thesis. These assumptions are not inherent to the algorithm, but have been included to give a more readable and simplified thesis. In this section, we will be discussing small alterations that can enhance the utility of our algorithms, enabling them to add complex transmitter constellations and MIMO systems to their repertoire. Next, we proceed to chart out new avenues that can be explored in this field, and finally conclude this thesis by presenting the crux of our work.

Chapter 5

Future work and Conclusion

One limitation of the spatial algorithms as yet presented is their dependence on the SIMO channel. Furthermore, the algorithms presented make exhaustive use of the binary constellation as limited by assumption *(b)* in *Chapter 2*. However, these two factors are not inherent limitations of the spatial algorithms. In the first subsection, *Extending spatial algorithms*, we outline how these algorithms can be modified not only to process complex constellations, but also to process and extract data in Multiple Input Multiple Output (MIMO) platforms. Next, under *Future Work*, we describe additional directions for improving spatial algorithms. These modifications will enable more accurate and cost effective spatial algorithms to be formulated. Finally, we summarize our thesis and present the crux of our results in the last and concluding subsection.

5.1 Extending spatial algorithms

5.1.1 *T*-element Transmitter Constellations

The constellation used by a transmitter to impinge data onto a channel plays a major role in creating the spatial structures our algorithms utilizes for estimation. The spatial structures result from the convolution of a transmitter constellation with impulse parameters of the channel involved. In our thesis, to simplify the mathematics and data structures used in presenting our algorithms, we have limited the transmitter constellations to binary systems. Thus, our thesis rests on the premise of a transmitter limited to a constellation of

$$C_B = \{+1, -1\} \tag{5.1}$$

as specified in assumption *(b)* in *Chapter 2*. This however, is not an inherent limitation of the algorithms. However, using a larger transmitter alphabet is not without disadvantages. The number of states created in the received vector set Y depends on both the channel length and the number of elements in the transmitter constellation. An increase in either would result in a respective increase in the number of spatial states. The relationship between the number of states created N_S, to the channel length L, and the number of elements in the transmitter constellation, T is described by,

$$N_S = T^L \tag{5.2}$$

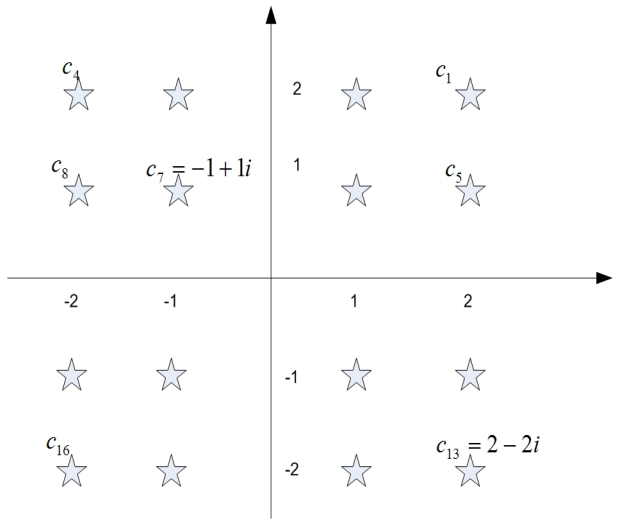

Figure 5.1: A 16 - element symmetric transmitter constellation, C_{16}

The number of states, N_S has a direct correlation to the computational requirements of the clustering algorithms. The exponential relationship between T and N_s makes spatial algorithms expensive where large constellations or long channels are involved. However, taking into fact that most practical constellations use complex elements and are symmetric about the real and imaginary axes, a simplification that reduces the number of states generated can be implemented. We begin by first defining the transmitter constellation as,

$$C_T \triangleq \{c_i | c_i = a_i + jb_i \; i \in \{1, ..., T\}\} \tag{5.3}$$

which, in case it is symmetric is also bound by

$$\{a_i, b_i\} \in \{\pm 1, ..., \pm(T^{1/2} - 1)\} \tag{5.4}$$

T here is the number of elements in the constellation. Fig 5.1 illustrates a sixteen element symmetric constellation which is similarly structured to 16 QAM. Using the constellation notation, a transmitted symbol at time index n can then be written as,

$$s_n = a_n + jb_n, \quad \{a_n + jb_n\} \in C_T \tag{5.5}$$

Integrating the structure of the complex input symbols as defined in Eqn. (5.4) onto the SIMO channel platform, we can expand Eqn. (2.4) from *Chapter 2* to

$$\mathbf{x}(n) = \mathbf{H} \begin{bmatrix} a_n + jb_n \\ a_{n-1} + jb_{n-1} \\ \vdots \\ a_{n-L} + jb_{n-L} \end{bmatrix} + \mathbf{w}_{R(n)} + j\mathbf{w}_{I(n)} \tag{5.6}$$

where $\mathbf{w}_{R(n)}$ and $\mathbf{w}_{I(n)}$ denotes the real and imaginary components of noise. Then, separating the real and imaginary components of the transmitter symbols, we can further simplify Eqn. (5.6) to,

$$\mathbf{x}(n) = \mathbf{H}\mathbf{a}(n) + j\mathbf{H}\mathbf{b}(n) + \mathbf{w}_{R(n)} + j\mathbf{w}_{I(n)} \tag{5.7}$$

where $\mathbf{a}(n) = [a_n, a_{n-1}, ..., a_{n-L}]'$ and $\mathbf{b}(n) = [b_n, b_{n-1}, ..., b_{n-L}]'$. Now, by separating the real and imaginary components from Eqn. (5.6), we can create the two parallel SIMO systems,

$$\mathbf{x}_R(n) = \mathbf{H}\mathbf{a}(n) + \mathbf{w}_{R(n)} \tag{5.8}$$

$$\mathbf{x}_I(n) = \mathbf{H}\mathbf{b}(n) + \mathbf{w}_{I(n)} \tag{5.9}$$

Instead of solving Eqn. (2.4), we can now estimate the channel by solving either Eqns. (5.7) or (5.8) or both. The creation of two parallel SIMO systems is graphically illustrated in Fig 5.2. The CEDS and CETI algorithms described in this thesis need only one of the two equations described above to estimate channel parameters. This is because they rely on the channel matrix and *elemental vectors* created between the spatial states. The SDSE algorithm however requires both equations. This is due to the fact that it has to identify each state uniquely before sequence estimation can begin. Such unique information can only be extracted when both Eqns.(5.7) and (5.8) are solved

The advantage to using Eqns. (5.7) and/or (5.8) instead of Eqn. (2.4) is that both (5.7) and (5.8) are based on pseudo constellations. The constellation Eqns. (5.7) and (5.8) describe contain only the components of C_T projected onto the real and imaginary axes respectively. Thus, they are lower in complexity compared to C_T. This inturn implies a lower computational burden in the clustering process. Let the pseudo constellations described by Eqns. (5.7) and (5.8) be denoted by C_A and C_B. For a symmetric constellation as shown in Fig 5.1,

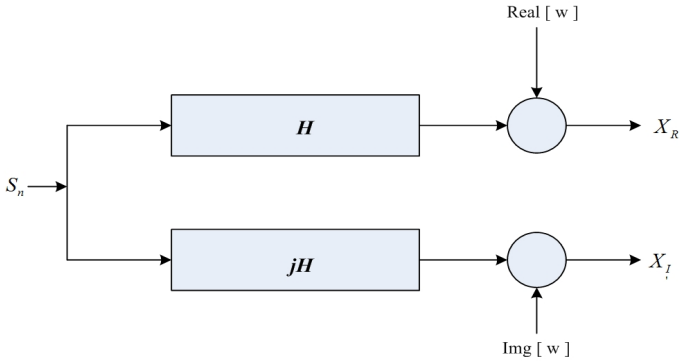

Figure 5.2: The complex channel

$$\{C_A, C_B\} = \{\pm 1, ..., \pm(T^{1/2} - 1)\} \tag{5.10}$$

Most practical constellations used for data transmission fall into this category. The savings in computational cost this simplification entitles makes spatial algorithms a more viable option for practical implementations.

5.1.2 Extending spatial algorithms to MIMO channels

Spatial information provides the basis for estimation in our algorithms. In this aspect, the multiple output platform we base our algorithms is vital. Only on such a platform can we capture the spatial structures needed for estimation. However, since only the multiple output structure is needed, it follows that in addition to estimating SIMO channels, our algorithms are capable of handling MIMO channels. To begin extending our algorithm, consider the SIMO model described by Eqn. (2.4)in *Chapter 2.* Let the source in Eqn. (2.4) be denoted by the subscript i. This helps identify the given source in a multiple input system.

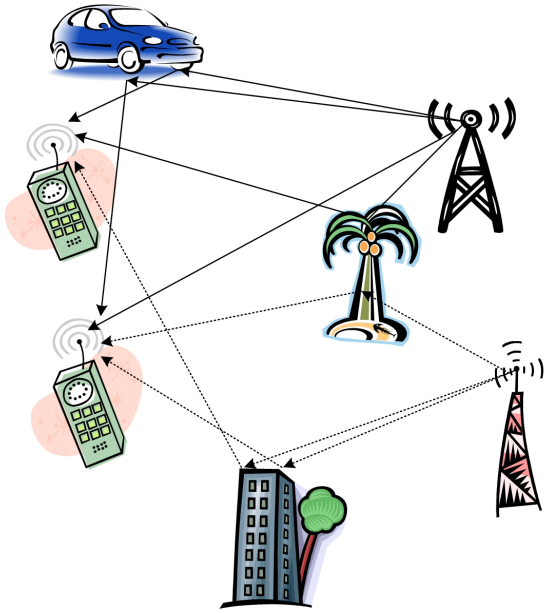

Figure 5.3: The Multiple input multiple output channel

We can then write a more appropriate representation of Eqn. (2.4) as,

$$\mathbf{x}_i(n) = \mathbf{H}_{i_{M\times L}}\mathbf{s}_{i_{L\times 1}}(n) + \mathbf{w}_{i_{M\times 1}}(n) \tag{5.11}$$

where the i subscript declares that the system is bound to the source i. A MIMO channel can be thought of as a collection of S such SIMO channels. This is illustrated in the simplified MIMO channel shown in Fig 5.3. In the figure, dotted lines indicate the propagation of one source while the solid lines indicate the propagation of another. For such systems, the received signal at any sensor is a superimposition of the data received from all sources. i.e.,

$$\mathbf{x}(n) = \sum_{i=1}^{S} \mathbf{H}_{i_{M\times L}}\mathbf{s}_{i_{L\times 1}}(n) + \mathbf{W}_{M\times 1}(n) \tag{5.12}$$

Expanding Eqn. (5.11), we can arrange it in the form,

$$\mathbf{x}(n) = [\mathbf{H}_1, \mathbf{H}_2, ...\mathbf{H}_S] \begin{bmatrix} \mathbf{s}_1(n) \\ \mathbf{s}_2(n) \\ \vdots \\ \mathbf{s}_S(n) \end{bmatrix} + \mathbf{W}_{M\times 1}(n) \tag{5.13}$$

illustrating the data structure spatial algorithms grasp when fed with input vectors from a MIMO system. We can further simplify Eqn. (5.12) to obtain the generic form,

$$\mathbf{x}(n) = \mathit{H}_{M\times \bar{L}} \mathit{S}_{n_{\bar{L}\times 1}} + \mathit{W}_{n_{M\times 1}} \tag{5.14}$$

where $\mathit{H} = [\mathbf{H}_1, \mathbf{H}_2, ...\mathbf{H}_S]$, $\mathit{S}_n = [\mathbf{s}_1(n), \mathbf{s}_2(n), ..., \mathbf{s}_S(n)]'$, $\mathit{W}_n = W(n)$ and,

$$\bar{L} = \sum_{i=1}^{S} L_i \tag{5.15}$$

This is identical to equations (2.4), (5.7) or (5.8), except the fact that the effective channel length has increased due to the concatenation of the channels into a monolithic whole. The mathematical structure given by Eqn. (5.13) indicates that the spatial algorithms will see the MIMO channel as a SIMO channel, albeit having a longer channel length. This is because spatial algorithms utilize the unique output vectors Eqn. (5.13) generates in absence of noise. The spatial vector set for the above MIMO system can be described by,

$$Y = \{\mathbf{y} | \mathbf{y} = \mathit{HS}_i \; i \in \{1, ..., N\}\} \tag{5.16}$$

This has exactly the same structure as Eqn. (2.5), which is used to describe SIMO channels in *Chapter 2*. Thus, estimation of MIMO channels comes easily as an extension to our algorithms. Though spatial algorithms see no distinction between SIMO and MIMO channels, increase of the channel length results in an exponential increase in the computational cost. This is due to the fact that the number of states in a MIMO or SIMO channel increases exponentially with the channel length.

Fig 5.4 shows the performance of the CETI algorithm modified to compute MIMO channels. In this simulation, the channel was modeled as a stochastic SIMO model, with impulse parameters modeled as zero mean Gaussian processes having unit variances. Channel coefficients and noise are assumed identically and independently distributed, and in this simulation noise was modeled as a zero mean Gaussian process. Two SIMO systems with $L = 6$ and $L = 4$ were superimposed to produce the MIMO channel.

One fact that should be kept in mind when estimating MIMO channels in this manner is spatial algorithms blindness to time order. The CETI algorithm extracts the MIMO channel as a concatenated SIMO channel having 10 columns. Time blindness causes the channel columns of all SIMO channels to be mixed randomly, resulting in a complex permutation. By using the permutation recovery module explained in *Chapter 4*, permutation of the columns within the SIMO channels can be resolved. However, this algorithm is not able to solve the permutation of the SIMO channels with the monolithic concatenated channel. Further studies will need to be carried out in this area. This final permutation is shown

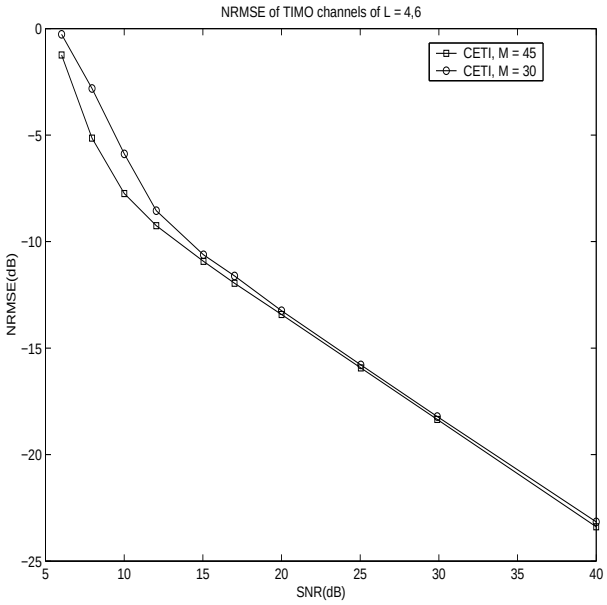

Figure 5.4: Extracting a Two Input Two Output channel using CETI

in Fig 5.5

5.2 Future Work in spatial algorithms

The algorithms described in this thesis utilizes blocks of input data for estimating channel and symbol parameters. This however, does not imply that the algorithms presented cannot be converted to have adaptive implementations. This is especially true with regard to the CEDS and CETI algorithms. In *Chapter 4*, Figs 4.7 and 4.8 outline pseudo adaptive implementations of the above two algorithms using incrementing blocks of data. Completely adaptive implementations has not been a goal in our current research. However, such an implementation would have lower computational requirements. Such, it would make spatial algorithms more practical.

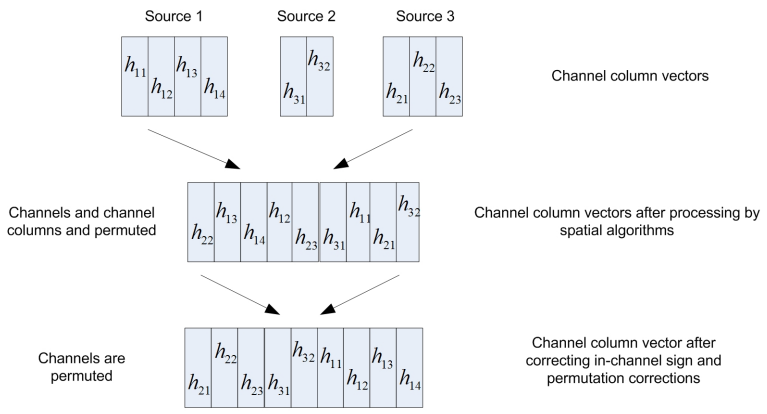

Figure 5.5: Permutation in extracting MIMO channels

Secondly, though spatial tools (e.g., primary and secondary clustering algorithms), form an essential part of our algorithms, they themselves have not been a focus in our study. However, using better spatial tools may result in generating better estimates for both channel and symbol parameters. Modern clustering algorithms based on fuzzy [50] and neural [51] technologies may have higher extraction capability compared to our algorithm. This is the capabilities of the clustering algorithm to extract the required vectors from the input vector set. This is one reason we do not attempt to benchmark our results. The estimation algorithms we present are platforms that can be used together with spatial tools to extract channel or symbol parameters. Such, they are dependant on the capability of spatial tools. Spatial tools are a topic in themselves and is an area we need to explore to utilize the full power of the algorithms presented in this thesis.

One problem with the spatial algorithms we present is that they are blind to time order. This is because time does not exist in the spatial domain. Thus, even

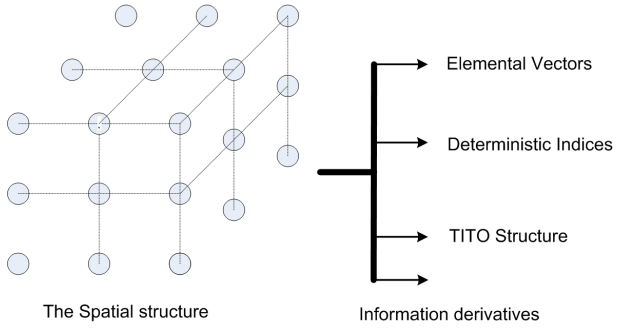

Figure 5.6: Derivatives of the spatial structure

though spatial algorithms can extract columns of the channel matrix **H**, they cannot extract data pertaining to their position or sign within **H**. To provide completeness, we have included an auxiliary algorithm that can correct both ordering and sign errors in the columns extracted. However, this algorithm does not belong to the spatial algorithm family. Better methodologies for correcting these errors are a topic that needs to be researched into. This will help ease assumption*(d)*, which is included only to validate the auxiliary algorithm. This in-turn will increase the scope of our algorithms.

Another important aspect that needs to be considered and researched into is the availability of other derivatives of information from the spatial structure. In our algorithms, we have utilized *elemental vectors*, *deterministic indices* and the *TITO structure* for channel and symbol estimation. This is illustrated in Fig 5.6. It is possible that other derivatives of information may exist and this is an area

that needs to be researched extensively. For example, the finite alphabet property used in the Cost Based Heuristic Search module in *Chapter 4* maybe incorporated into spatial algorithms to result in better performance. Furthermore, hybrid algorithms that use more than one data source of *statistical*, *deterministic* or *finite alphabet data* needs to be explored. Using two sources may help to create better algorithms that uses each source to cover weakness of the others it utilizes. This may create algorithms that are both robust and practical.

5.3 Conclusion

In our thesis, we present a family of algorithms utilizing spatial structures created by finite alphabet transmitters for estimating the channel parameters $\mathbf{H}$, L and the transmitted symbol sequence $\mathbf{s}$, under the assumption of a static or slow varying channel.

We begin our thesis with first an introduction into the mobile media, its properties and how it impedes communication systems. Then, we explore the solutions available for overcoming ISI in the form of blind algorithms. A discussion on the two main categories of blind estimation algorithms, *statistic algorithms* and *deterministic algorithms* is presented. Then, a sub-category of deterministic algorithms known as finite alphabet algorithms which utilizes the structure of the transmitter constellation in lieu of the channel structure is presented. We then proceed to outline the motivation behind our thesis, to create a framework for developing a class of finite alphabet algorithms, algorithms that utilize spatial structures and can estimate the channel parameters $\mathbf{H}$, $\mathbf{s}$ and L. These algorithms can be developed in the future to perform on par and even exceed traditional algorithms.

Next, we introduce fundamental mathematical concepts that power our algorithms. We begin by introducing the channel platform used to formulate our algorithms and outline the basic assumptions which all algorithms in this thesis are based on. Furthermore, in this section we introduce two spatial tools: the *primary* and *secondary* clustering algorithms, that we have developed to handle

spatial data. The primary clustering algorithm can extract the spatial structure from the received vector set of a multiple output platform. The secondary clustering algorithm is more subtle in nature. It is used to extract vector families corrupted by noise from a vector admixture using the vectors relative population as a key. We end this chapter with an introduction into the deterministic indices that form the core of the Channel Estimation by Twin Indexing (CETI) scheme.

In the third chapter, we introduce the first of our spatial algorithms, the State Driven Sequence Estimation (SDSE) algorithm. Indepth working of the theoretical algorithm is first presented and then followed a discussion of the errors that can plague it in noisy environments. Modifications needed to overcome these limitations are then presented, and finally, the performance of the algorithm is presented with an indepth discussion into its behavior.

Following, we introduce the two channel estimation schemes, Channel Estimation by Difference Sets (CEDS) and Channel Estimation by Twin Indexing (CETI). These algorithms are explained in detail and followed by a procedural presentation that makes the algorithms easy to understand. Next we present an auxiliary algorithm that helps overcome spatial algorithms inherent blindness to time ordering. This algorithm resolves the sign and permutation ambiguities inherent in the output of the CEDS and CETI algorithms. Lastly, the performance of the CETI and CEDS algorithms are analyzed individually and with respect to each other and later followed by an indepth discussion into their behavior.

The assumptions stated at the beginning of our thesis limit the utility of our algorithms. First, we can extend our work from the binary constellations used to

presnet our algorithms to examine complex constellations that exist in the real world. The mathematical models we have used is analyzed explaining how to incorporate complex transmitter constellations into it. Furthermore, an advantage of using symmetric constellations is highlighted within this framework. This is of advantage as most practical constellations used in communications are symmetric and this simplification induces a reduction in computation requirements. Then we extend our work from the SIMO platform onto the MIMO platform. The MIMO platform also shares the same multiple output feature, enabling our algorithm to migrate into this domain easily. Following, we discuss new avenues for research that will enhance the accuracy while simultaneously decreasing the computational requirements of the algorithms.

The algorithms described in this thesis are essentially a category of deterministic algorithms. Thus, they have the advantage of having finite sample convergence which makes them perform better in SNRs above 15dB. In addition, algorithms like the CETI can utilize extremely small data sets for estimation. This has two distinct advantages. The reduction in computational cost is one. The second results from using a shorter data set. This property makes the algorithm more resilient to fading conditions. The smaller the data set the algorithm utilizes, the better apt it is to facing faster fading channels. This gives the CETI algorithm a distinct practical advantage.

However, all spatial algorithms introduced in this thesis have one main drawback. They rely extensively on the multiple output platform. Such, the number of inputs available has a direct impact on the ability of the algorithms to extract

either the spatial structure or its derivative, the deterministic indices. In either case, the performance of the algorithms drop sharply if the number of multipaths are not sufficient. This is more noticeable in the CETI algorithm as it only uses a fraction of the input data available for estimation.

But taken from the other end, increasing the number of sensors allows the algorithm to perform better even in low SNR regions. A scheme that dynamically alters the number of inputs depending on the current SNR will be beneficial as it can reduce the computational cost in moderate SNRs and still functioning aptly in regions of low SNRs. This will help create a more viable algorithm.

Lastly, it must be noted that the assumptions stated at the beginning of this thesis do not necessarily represent limitations of the spatial algorithms introduced here. The assumption of a binary constellation stems from the need for a simplified presentation. Extending our work to more complex constellations is described in the previous section. Another assumption stems from the auxiliary algorithm we have bundled in our thesis for correcting sign and permutation errors. This algorithm needs the channel matrix to be full column rank. This assumption is not an inherent component of the spatial algorithms. Such, alternate algorithms that can correct these errors may help ease this restriction.

Bibliography

[1] T S Rappaport, "Wireless Communications: Principles and Practice", Second Edition, Prentice Hall, Upper Saddle River, NJ 2002.

[2] T Okumura, E Ohmori and F Fukuda, "Field Strength and its Variability in VHF and UHF Land Mobile Service", *Review of Electrical Communication Laboratory*, Vol. 16, No. 9-10, pp. 825-873, Oct. 1968.

[3] Hata, Masaharu, "Empricial Formula for Propagation Loss in Land Mobile Radio Service", *IEEE Trans. on Vehicular Technology*, Vol. VTC-29, No. 3, pp. 317-325, Aug 1980.

[4] J. Walfisch, H.L. Bertoni, "A theoretical model of UHF propagation in urban environments" *IEEE Trans. Antennas and Propagation,* Vol. 36, Issue: 12 , pp. 1788 - 1796, Dec. 1988

[5] Y. Sato, "A Blind Sequence Detection and its Application in Digital Mobile Communicatiom," *IEEE Jour. on Selected Areas in Commun.*, Vol. 13, Issue: 1, pp. 49 - 58 Jan. 1995

[6] Y. Sato, "A method of self-recovering equalization for multilevel amplitude modulation," *IEEE Trans. Commun.*, Vol. 6, pp. 679-682, 1975.

[7] L. Tong, S. Perreau, "Multichannel blind identification: from subspace to maximum likelihood methods, "*Proc. IEEE* , Vol. 86 , Issue: 10, pp. 1951 - 1968, Oct. 1998

[8] Y. Hua, "Fast maximum likelihood for blind identification of multiple FIR channels," in *Proc. 28th Asilomar Conf. Signals, Systems, and Computers*,Vol 1, pp 414-419, Pacific Grove, CA, Nov. 1994.

[9] D. Slock, "Blind fractionally-spaced equalization, perfect reconstruction filterbanks, and multilinear prediction," in *Proc. IEEE ICASSP Conf. on Acoust,. Speech, Signal Proc.* Vol. IV, pp 585-588, April 1994.

[10] L. E. Baum, T. Petrie, G. Soules, and N. Weiss, "A maximization technique occuring in the statistical analysis of probabilistic functions of Markov chains," *Ann. Math. Stat.*, vol. 41, pp. 164-171, 1970.

[11] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum Likelihood from Incomplete Data via the EM Algorithm," *Journal of Royal Statistical Society B*, vol. 39, no. 1, pp. 1-38, 1977.

[12] L Tong, Guanghan Xu, T Kailath, "Blind identification and equalization based on second order statistics: A Time domain approach", IEEE Trans. on Info. Theory, Vol. 40, No. 2, pp. 340−349, Mar 1994.

[13] K. Abed-Meraim and E. Moulines, "A maximum likelihood solution to blind identification of multichannel FIR filters," *Proc. EUSIPCO*, Edinburgh, Scotland, Vol. 2, pp. 1011-1014, 1994.

[14] D.M. Titterington, "Recursive parameter estimation using incomplete data," *J. Roy. Stat. Soc. B*, vol. 46, no. 2, pp. 257-267, 1984.

[15] G. Giannakis and J. Mendel, "Identification of nonminimum phase systems using higher-order statistics," *IEEE Trans. Acoust., Speech, Signal Processing*, Vol. 37, pp. 360-377, 1989.

[16] Cadzow, "Blind deconvolution via cumulant extrema," *IEEE Signal Processing Mag.*, Vol. 13, pp. 24-42, 1996.

[17] J. A. Cadzow and X. Li, "Blind deconvolution," *Digital Signal Processing*, Vol. 5, pp. 3-20, 1995.

[18] D. Hatzinakos and C. Nikias, "Blind equalization using a trispectrum based algorithm," *IEEE Trans. Commun.*, Vol. 39, pp. 669-682, May 1991.

[19] D. Godard, "Self-recovering equalization and carrier-tracking in two-dimensional data communication systems," *IEEE Trans. Commun.*, Vol. 28, pp. 1867-1875, Nov. 1980.

[20] G. Picchi and G. Prati, "Blind equalization and carrier recovery using a 'stop-and-go' decision-directed algorithm," *IEEE Trans. Commun*, Vol. 35, pp. 877-887, Sept. 1987.

[21] L. Tong, G. Xu, B. Hassibi, and T. Kailath, "Blind channel identification based on second-order statistics: a frequency-domain approach," *IEEE Trans. Inform. Theory*, vol. 41, pp. 329-334, Jan. 1995.

[22] G.B. Giannakis, "Linear cyclic correlation approach for blind identification of FIR channels," in *Conf Rec. 28th IEEE Asilomar Conf. on Signals, Systems, and Computers*, Pacific Grove, CA, pp. 420-423, Nov. 1994.

[23] L. Tong, G. Xu, and T. Kailath, "A new approach to blind identification and equalization of multipath channels, *25th Asilomar Conf. Signals, Systems, and Computers*, Vol. 2, pp 856-560, Nov. 1991

[24] 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, Mar. 1994.

[25] D. Slock and C. B. Padias, "Further results on blind identification and equalization of multiple FIR channels," in *IEEE Proc. Intl. Conf. Acoustics, Speech, Signal Processing*, Detroit, MI, pp. 1964-1967, Apr. 1995.

[26]]G. Xu, H. Liu, L. Tong, and T. Kailath, "A least-squares approach to blind channel identification," *IEEE Trans. Signal Processing,* Vol. 43, pp. 2982-2993, Dec. 1995.

[27] Y. Hua and M. Wax, "Strict identifiability of multiple FIR channels driven by an unknown arbitrary sequence," *IEEE Trans. Signal Processing*, vol. 44, pp. 756-759, Mar. 1996.

[28] H. Liu, G. Xu, and L. Tong, "A deterministic approach to blind equalization", in *Conf. Rec. 1994 IEEE ICASSP Conf. on Acoust., Speech, and Signal Proc.* Vol.4, pp 581-584, April 1994.

[29] M. L. Gurelli and C. L. Nikias, "EVAM: An eigenvector-based deconvolution of input colored signals," *IEEE Trans. Signal Processing*, vol. 43, pp. 134-149, Jan 1995.

[30] L. A. Baccala and S. Roy, "Time-domain channel identification algorithms," *in Proc. 26th Conf. Information Science Systems*, Princeton, NJ, pp. 863-867, Mar. 1994.

[31] E. A. Robinson, "T. Tomographic deconvolution of echograms," in *Communications, Computation, Control and Signal Processing: A Tribute to Thomas Kailath*, A. Paularj, V. Roychowdhury and C. Schaper, Eds. Norwell, MA,: Kluwer, 1997.

[32] E. Moulines, P. Duhamel, J. F. Cardoso, and S. Mayrargue, "Subspace-methods for the blind identification of multichannel FIR filters," *IEEE Trans. on Acoust., Speech, and Signal Proc.,* Vol: 43, Issue: 2, pp. 516 - 525, Feb. 1995

[33] K. Abed-Meraim, P. Loubaton, and E. Moulines, "A subspace algorithm for certain blind identification problems," *IEEE Trans. Inform. Theory*, vol. 43, pp. 499-511, Mar. 1997.

[34] L. Tong and Q. Zhao , "Blind channel estimation by least squares smoothing," in *Proc, 1998 Intl. Conf. Acoustics, Speach and Signal Processing*, Vol. 5, pp. 2121 - 2124, Sept. 1998

[35] L. Tong and Q. Zhao, , "Joint order detection and channel estimation by least squares smoothing," *IEEE Trans. Signal Processing,* Vol. 47, pp. 2345-2355, Dec. 1999.

[36] Q. Zhao and L. Tong, "Adaptive blind channel estimation by least squares smoothing,", *IEEE Transactions on Signal Processing*, Vol. 47 , Issue: 11, pp. 3000 - 3012, Nov. 1999.

[37] G. Forney, "Maximum-likelihood sequence estimation of digital sequences in the presence of intersymbol interference", *IEEE Trans. Information Theory*, Vol. 18, Issue: 3, pp. 363 - 378, May 1972.

[38] G. Forney, "The Viterbi algorithm", *IEEE Proc.*, Vol 61, pp. 268-278, Mar 1972.

[39] L. Tong, "Blind sequence estimation" *IEEE Trans. on Communications,* Vol. 43 , Issue: 12 , pp. 2986 - 2994 Dec. 1995

[40] Tongtong Li, Zhi Ding, "A reduced-state Viterbi algorithm for blind sequence estimation of DPSK sources", *Proc. IEEE Conf. Global Telecomm.*, Rio de Janeireo, Brazil, Vol: 4, pp. 2167−2171, Sept. 1999

[41] J. Gunther, A. Swindlehurst, "Blind sequential symbol estimation of co-channel finite alphabet signals", *Conf. Rec. of 13 Asilomar Conf. on . Signals, Systems and Computers*, Vol. 2, pp. 823 - 827 Nov. 1996.

[42] J.H. Manton, Yingbo Hua, "A randomised algorithm for improving source and channel estimates by exploiting the finite alphabet property", *Conf. Rec. 34th Asilomar Conf. on Signals, Systems and Computers*, Vol. 2, pp. 1582 - 1585, Nov. 2000

[43] D. Yellin, B. Porat, "Blind identification of FIR systems excited by discrete-alphabet inputs," *IEEE Trans. Signal Proc.,* Vol. 41, Issue: 3 pp. 1331 - 1339, March 1993.

[44] Chong-Meng Samson See, "A novel approach to data-efficient blind channel identification and equalization", *Proc. IEEE Int. Conf. on Information, Communications and Signal Processing*, pp. 1213 - 1215 Vol.2, Sept. 1997

[45] F. Daneshgaran and M. Laddomada, "Multiscale LBG Clustering for SIMO Identification", *Proc. IEEE Int. Conf. on Comm.*,New York, NY ,pp. 84−88, May 2002.

[46] F. Daneshgaran, M. Mondin, F. Dovis, M.S. Roden, "Blind estimation of output labels of SIMO channels based on a novel clustering algorithm", *IEEE Ltrs. on Commun.* , Vol. 2 , Issue: 11, pp. 307 - 309 Nov. 1998

[47] A.J. van der Veen, S. Talwar, A. Paulraj, "Blind identification of FIR channels carrying multiple finite alphabet signals", *Proc. IEEE Acoustics, Conf.*

on Speech, and Signal Processing , Detroit, USA, Vol.2, pp. 1213−1216 May 1995.

[48] A.J. Van der Veen, S. Talwar, A.Paulraj, "Blind estimation of multiple digital signals transmitted over FIR channels", *IEEE Signal Processing Letters,* Vol. 2, Issue: 5, pp. 99 - 102, May 1995.

[49] K. Abed-Meraim, Wanzhi Qiu, Yingbo Hu, "Blind system identification", *Proceedings of the IEEE* , Vol. 85, Issue: 8, pp. 1310 - 1322, Aug. 1997.

[50] S.Nascimento, B. Mirkin, F. Moura-Pires, "A fuzzy clustering model of data and fuzzy c-means", *Proc. 9th IEEE Int. Conf. on Fuzzy Systems*, San Antonio, TX USA, Vol. 1, pp. 302 - 307, May 2000.

[51] M.Sato-Ilic, "Non-metric neural clustering", *Proc. 6th IEEE Int. Conf. on Neural Information Processing*, Perth, WA Australia, Vol. 1, pp. 72 - 77, Nov. 1999.

[52] C.Saint-Jean, C. Freicot, " A robust semi-supervised EM-based clustering algorithm with a reject option", *Proc. 16th IEEE Int. Conf. on Pattern Recognition*, Vol. 3, pp. 399-402, Aug 2002.

[53] Pauwels, E.J. Frederix, G., Cluster-based segmentation of natural scenes, *Proc. 7th IEEE Intl. Conf. on PComputer Vision*, Vol. 2, pp 997-1002, Sept 1999.