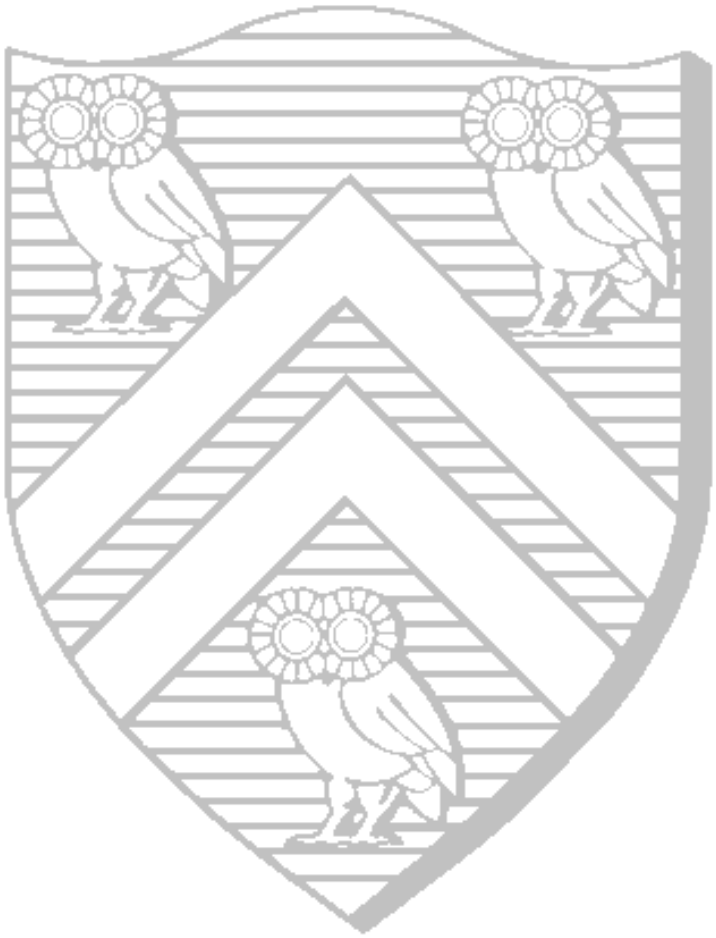
Algorithms and Architectures for Channel Estimation in Wireless CDMA Communication Systems

Chaitali Sengupta



Thesis: Doctor of Philosophy
Electrical and Computer Engineering
Rice University, Houston, Texas (December 1998)

RICE UNIVERSITY

Algorithms and Architectures for Channel Estimation in Wireless CDMA Communication Systems

by

Chaitali Sengupta

A Thesis Submitted
in Partial Fulfillment of the
Requirements for the Degree

Doctor of Philosophy

Approved, Thesis Committee:

Joseph R. Cavallaro, Chair
Associate Professor of Electrical and
Computer Engineering

Behnaam Aazhang
Professor of Electrical and Computer
Engineering

Devika Subramanian
Associate Professor of Computer Science.

Houston, Texas

December, 1998

ABSTRACT

Algorithms and Architectures for
Channel Estimation in
Wireless CDMA Communication Systems

by

Chaitali Sengupta

Wireless cellular communication is witnessing a rapid growth in markets, technology, and range of services. An attractive approach for economical, spectrally efficient, and high quality digital cellular and personal communication services is the use of code division multiple access (CDMA) technology. The estimation of channel delays along with channel attenuation and phases of different users constitutes the first stage in the detection process at the receiving base station in a CDMA communication system. This stage, called channel parameter estimation, forms the bottleneck for the detection of users' bitstreams; both in terms of accuracy as well as execution time. In this thesis, we develop new algorithms and architectures to solve the CDMA channel estimation problem.

We have first developed a framework that facilitates a computationally efficient solution to the combined problem of channel estimation and detection in a scenario involving multiple users, multiple paths, and multiple sensors at the receiver. The channel estimation approaches presented in this thesis, consist of two categories: (1) maximum likelihood based schemes, and (2) signal and noise subspace based schemes.

The maximum likelihood approach is used to solve the complex multidimensional

problem of channel estimation in the presence of multipath effects and concurrently using an antenna array at the base station receiver. Once the composite channel impulse response of each user is estimated, it is directly used in the detection process instead of first extracting the individual channel parameters, such as path delays and attenuation factors. This technique benefits from better performance as well as lower computational cost. Further, implementation issues of this algorithm, such as complexity reduction and fixed point error behaviour have also been addressed.

Our contribution to the subspace-based solution includes extension of the basic algorithm to tracking of the channel parameters in a time varying environment. We have also applied algorithmic optimizations to reduce the computation required for the algorithm and developed architectural enhancements to improve the execution time, such as parallel processing and implementation on fixed point hardware.

Acknowledgments

I take this opportunity to thank my advisor, Dr. Cavallaro, for his constant support, encouragement, and guidance, throughout my stay at Rice. I also thank Dr. Aazhang, for being a constant source of ideas and guidance, and for helping me believe in my work. Thanks also to Dr. Subramanian, for serving on my committee, and for all her constructive suggestions.

I also wish to thank all the members of the CDMA research group, especially Raghu and Suman, for helping me to organize my thoughts; and Ari Hottinen and Jorma Lilleberg of Nokia, Inc. for their advice and help.

I am grateful to Mandy and Nora for making my life as a graduate student so smooth and for taking care of so many different aspects of the graduate program. There are many others who have made my time at Rice truly enjoyable - Martin, Partha, Vijay, and Hazim deserve special thanks for putting up with me as their office-mates! Special thanks also to Scott King, for helping me out with a constant supply of magazines, and to Ara for his support and encouragement.

I would like to thank my family - Boromama-mamima, Chotomama-mamima, for making the last few years so memorable. Also, Dadu, Dida, Dada, Didi, and Dimma - they are my ultimate source of inspiration. Thanks also to Baba and Ma, for being so supportive - they are the best in-laws one could wish for. My heartfelt love and gratitude to Baba, Ma, and Somu, for their unwavering belief in me - I wouldn't have made it without them.

And finally, I want to thank Sudipta, for his love, patience, his desire that I succeed, and for sharing the joys as well as the disappointments. Thanks for helping me believe in myself, and for making the impossible look possible.

The work documented in this thesis has been supported by Nokia Inc., by the Texas Advanced Technology Program under grants 1995-#003604-049 and 1997-#003604-044, and by NSF under grants NCR 9506681 and CDA-9617383.

Contents

Illustrations

Tables

To Baba,

For showing us how to live.

"I have had my invitation to this world's festival, and thus my life has been blessed. My eyes have seen and my ears have heard. It was my part at this feast to play upon my instrument, and I have done all I could."

- Rabindranath Tagore.

Chapter 1

Introduction

The design and implementation of a high-speed, high-quality, wireless link between two mobile terminals, located anywhere in the world is the challenge being faced by the communications research community today. The popularity of cellular phones, paging, personal communication services (PCS) and other emerging mobile services on the horizon demonstrates a great demand for such technologies. The very essence of exponential growth in this area is illustrated by the growth of cellular telephone subscribership in the United States shown in Figure 1.1, based on data published in [1]. The worldwide growth in the number of subscribers over the last decade shows a similar trend [2], with projected annual handset sales of 300 million by the year 2002 [3].

In order to accommodate the demand for wireless communication services, new techniques that allow for efficient use of limited available frequency spectrum, increased system capacity and speed of communications as well as accuracy, are being developed. The Code Division Multiple Access (CDMA) protocol has recently emerged as one such commercially viable technique.

In a CDMA system, a communication channel with a given bandwidth is accessed by all the users simultaneously. The different mobile users are distinguished at the base station receiver by the unique *spreading code* assigned to the users to modulate their signals. Hence, the CDMA signal transmitted by any given user consists of that user's data which modulates the unique spreading code assigned to that user,

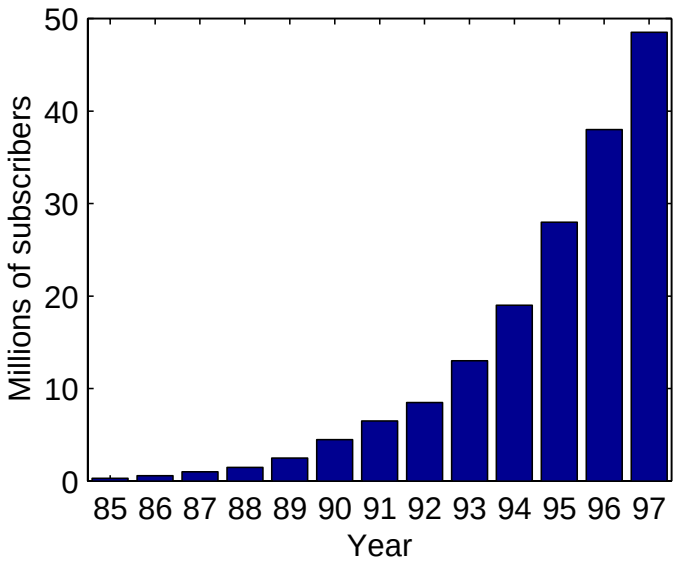


Figure 1.1 : Number of subscribers to cellular services in the U.S., tabulated as of June in each of the reported years.

which in turn modulates a carrier (the frequency of which is the same for all users), using any well-known modulation scheme such as *binary phase shift keying (BPSK)*. Figure 1.2 shows the modulation of the bits of the users by a spreading code. The low cross-correlation between the spreading codes of various users and the peaky auto-correlation property of each code provide the basis for detection of the transmitted symbols of each user at the receiver.

Wireless systems involve two radio links: the *reverse link* or the *uplink* from the mobile to the base station, and the *forward link* or the *downlink* from the base station to the mobile. In this thesis, we study the channel parameter estimation problem, described later, primarily in the reverse link.

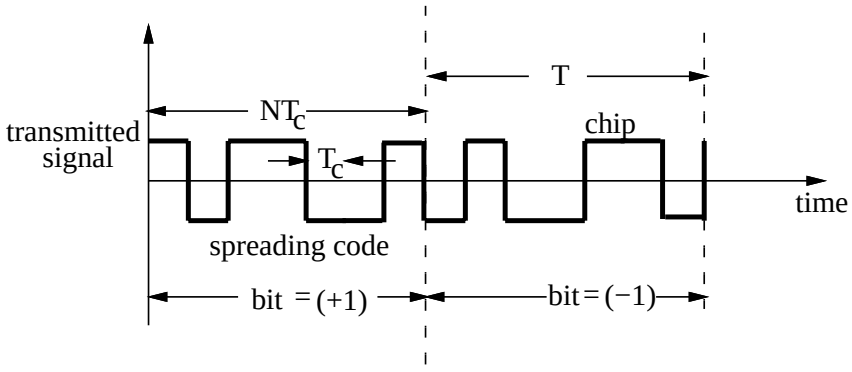


Figure 1.2 : Spreading in a direct sequence CDMA system. The transmitted signal consists of 2 bits +1 and -1. Each bit is multiplied by a spreading code {+1,-1,+1,+1,-1,-1,+1} consisting of 7 chips. T is the bit period, T_c is the chip period, and N is the number of chips per bit.

1.1 The channel estimation problem

The wireless channel in mobile radio poses a great challenge as a medium for reliable high speed communications. When a radio signal is transmitted through a wireless channel, (Figure 1.3) it suffers various types of distortions, described in greater detail in the next chapter (Section 2.2). Hence, the receiver obtains a linear superposition of the signals transmitted by all the users, attenuated by arbitrary factors and delayed by an arbitrary amount. In addition, due to scattering and reflections from various obstacles between the transmitter and the receiver, multiple copies of the same signal reach the receiver.

The reverse link is inherently asynchronous in nature, i.e., different signals arrive at the receiver with different relative time-offsets [4] with respect to an arbitrary timing reference at the receiver as shown in Figure 1.4.

The received signal is converted from passband to baseband, demodulated, digitized, and then processed in baseband to detect and decode the information bits

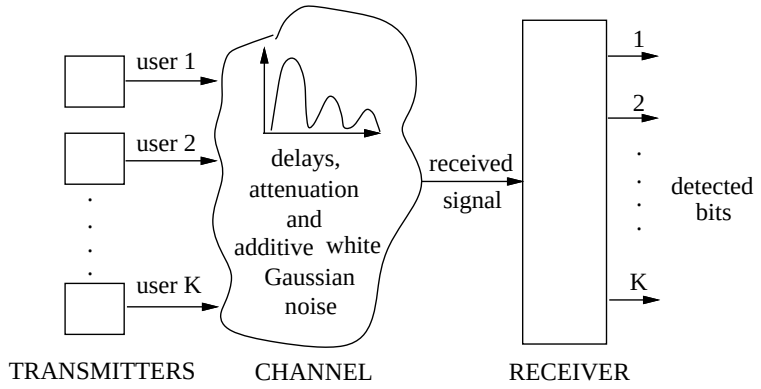


Figure 1.3 : A wireless transmission system

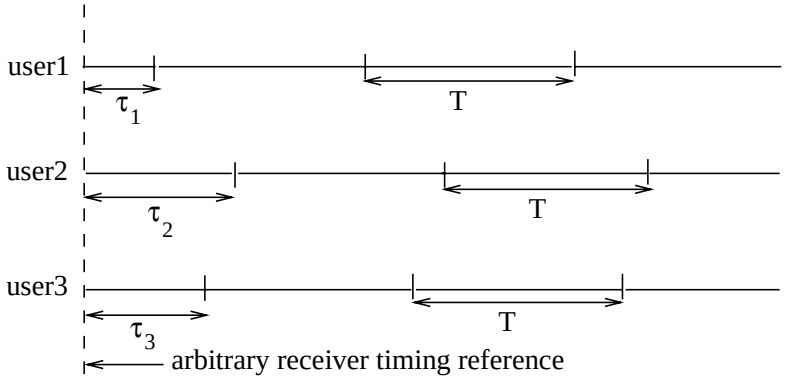


Figure 1.4 : Asynchronous nature of reverse link transmission

(Section 2.4.2). The detection of a particular user's transmitted bits involves the correlation of the received waveform with a copy of the corresponding spreading code at the receiver. Accurate correlation necessitates an accurate estimate of the user's timing offset shown in Figure 1.4. The necessity of an accurate estimate of this timing offset has been illustrated in Figure 1.5.

Once an initial estimate of each user's delay is calculated, the next task is to track these delays over the subsequent incoming bits. The two stage process of acquiring the users' timings from the baseband digital received signal, and tracking them as they

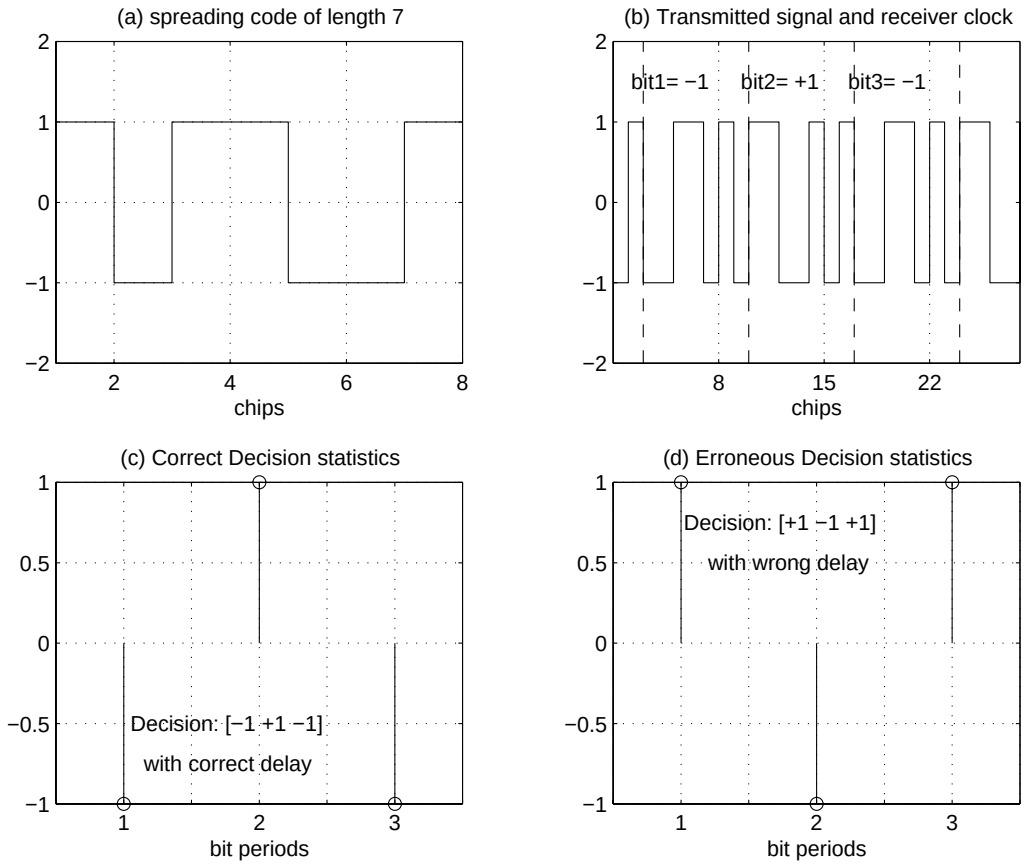


Figure 1.5 : Demonstrating the necessity of delay estimation for detection: Plot (b) shows the transmitted signal of a user with spreading code (a). The dotted lines in (b) are the bit intervals and the dashed lines mark the receiver clock with a delay of 2 chips. Plot (c) shows the correlator or the code-matched filter output with accurate delay estimate, which results in correct detection of the bits. Plot(d) shows that an incorrect delay estimate (estimate off by one chip) causes wrong detection.

change is called synchronization. In addition to the delays of the different propagation paths of the different users, certain advanced detection schemes also require estimates for the complex amplitudes of each path. Taken together, the estimation of all these parameters constitute the channel estimation problem.

The problem of channel estimation in the downlink or forward link is slightly different than the reverse link. Due to the use of a common transmitter clock at the

base station, the signals of each of the users is inherently synchronous with respect to each other and are subject to the same channel distortions. However the synchronous transmitted signals have a common offset with respect to the arbitrary receiver clock. The problem of channel estimation in this case involves estimation of this single offset as well as the common signal attenuation faced by the transmitted signals for all the users.

In the presence of a number of interfering users in the channel, channel estimation is not a trivial task, due to the *near-far* effect. The near-far problem arises when the signals from the different users arrive at the receiver with widely varying power levels. So a signal with lower power cannot be detected correctly due to the presence of other signals with higher powers. The near-far problem has been shown to severely degrade the performance of standard single user techniques (e.g., matched filters, correlators, etc.) in conventional CDMA systems, as such systems treat the interfering users as noise. In the conventional sliding correlator [5], used for channel estimation, the received signal is correlated with locally generated copies of the code at the receiver with delays spaced every one-half (or smaller fraction) of a chip. The delay corresponding to the largest output is the estimated delay. The process is illustrated in Figure 1.6. The success of the scheme depends on the peaky autocorrelation and low crosscorrelation properties of the spreading code.

Since the interfering users are treated as additive white Gaussian noise by the sliding correlator, this channel estimator is very susceptible to high interference from other users (called multiple access interference or MAI). Figure 1.7 illustrates how the near-far effect degrades the performance of a sliding correlator based timing estimator. Conventional CDMA systems try to limit the near-far problem with power control. However, even a small amount of the near-far effect can drastically de-

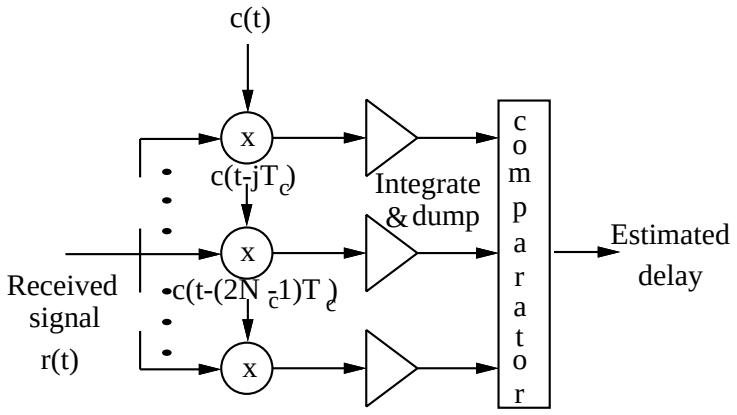


Figure 1.6 : Sliding correlator

grade the performance of conventional receivers. For many years this was thought to be an inherent limitation of CDMA until Verdú developed the optimum multiuser detector [6]. Verdú's work was followed by many suboptimal schemes of lower computational complexity [7–9], all of which are near-far resistant. However, these methods deal only with detection and assume that the timing (and in some cases amplitudes) of the different signals are known.

Most of the initial work done on timing acquisition for CDMA systems focused on jointly estimating the necessary parameters for all users [10–14]. While these techniques produce excellent results, they can be computationally intense since they involve solving a multidimensional optimization problem for a large number of parameters.

Subsequently, various other algorithms were proposed which focus on solving this multidimensional problem in an efficient manner. Two categories of algorithms which have been featured prominently in the literature because of various advantages in terms of performance and ease of computation are: (1) subspace based techniques,

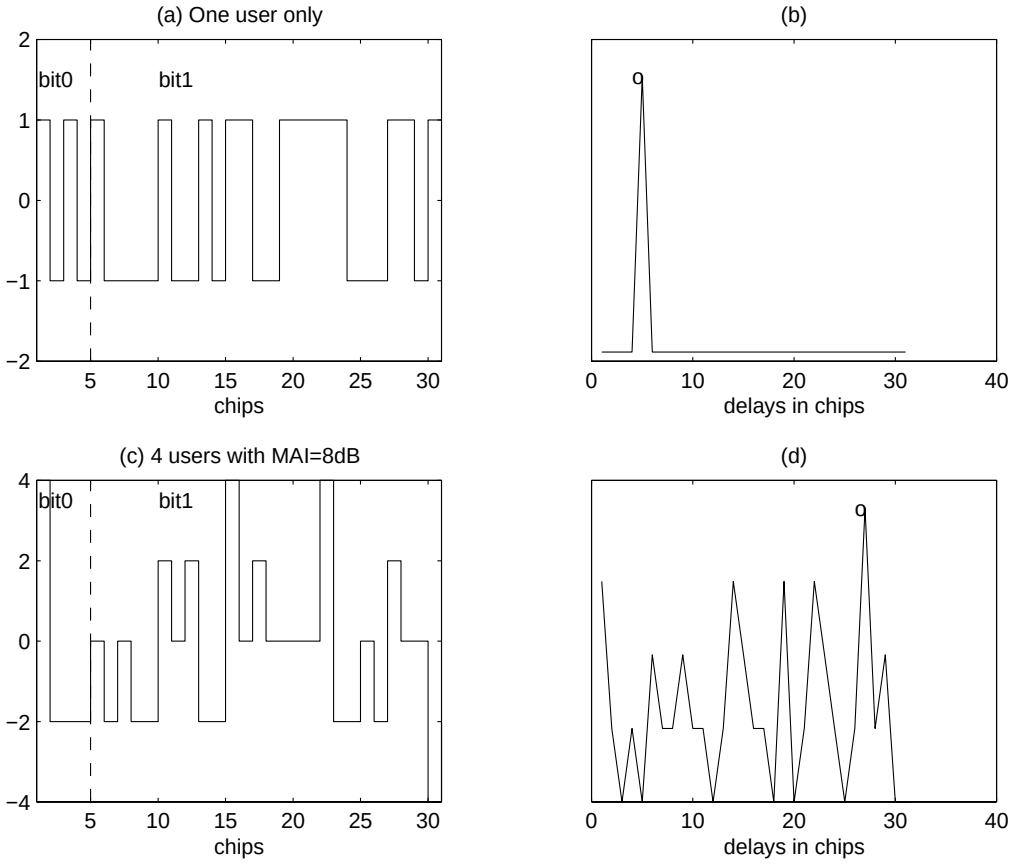


Figure 1.7 : Demonstrating the effect of multiple access interference on the sliding correlator: Plot (a) shows the transmitted signal with one user and Plot(c) shows the transmitted signal for 4 users (3 interfering users with power levels 8dB higher than the user of interest). Plot (b) shows the correlator output for the signal (a) where the delay of 5 chips shows up correctly. In Plot(d), the correlator gives an erroneous delay estimate due to the high MAI.

and (2) maximum likelihood techniques. The subspace based techniques [15–17] exploit the signal or noise subspace of the received signal and do not require any knowledge of the spreading codes of the interfering users or of a known sequence of bits, called a *preamble*. However such algorithms require the estimation of the subspace which is a computationally expensive process. Moreover, the number of users that can be handled by this algorithm is limited by the dimension of the signal subspace. The

maximum likelihood (ML) algorithms [18–20] find the maximum-likelihood estimate of the channel parameters using a preamble. These algorithms are more computationally efficient and can handle a large number of users. The algorithms in these two categories, described in greater detail in later chapters, deal with a variety of situations, such as, single path [18–20], multipath [17], single sensor or multiple sensors [19]. Some algorithms exploit the structure of the interfering users [17, 19] while others treat the multiple access interference as colored noise [20]. However, none of these techniques provides a solution for the entire problem, considering multiple paths, multiple sensors, and exploitation of the structure of the interfering users. In this thesis, we provide a solution to this more general problem.

1.2 Contributions of this thesis

We focus on *multiuser* techniques for channel parameter estimation which exploit the knowledge of the structure of the interfering users and which function in a multipath environment in the presence of multiple sensors. At the same time the proposed techniques are computationally efficient and perform well in the presence of widely varying power levels.

We propose a maximum likelihood technique [21, 22] that exploits the benefits of spatial diversity in the form of an antenna array at the receiver to gain an increase in performance of the system. It has been shown before [19] that the use of a number of sensors at the receiver dramatically improves the performance of channel estimators as well as linear detectors. However the problem of using multiple sensors to detect multipath channel parameters, is mathematically complex, as it involves the estimation of a large number of parameters. In this thesis, we first develop a system model that efficiently captures the effect of multipath and multiple sensors and then pro-

pose an algorithm which solves the channel estimation problem in a computationally viable manner.

The thesis also includes the idea of joint synchronization and detection, related to maximum likelihood channel estimation and linear detectors [23]. Here the focus is on capturing and estimating the effect of the channel in the fewest possible steps, in a form directly usable by the detector. This approach not only saves computation but also results in an increase in performance.

Our contributions to the subspace-based solution include extension of the basic multipath synchronization algorithm [17] to tracking of the parameters in a time varying environment [24]. Along with development of the algorithm, we will also consider techniques to improve the execution time of the algorithm with a view to achieving efficient real-time execution. These techniques include parallel processing [25], implementation on fixed point hardware [26, 27], and algorithmic optimizations to reduce computation [24].

1.3 Thesis overview

In this introductory chapter, we have defined the problem addressed by this thesis and demonstrated the need for efficient solutions to the channel estimation problem. We have also outlined the key contributions of this thesis.

In the next chapter, we present the background for the subject material of this thesis. We first describe the structure of wireless cellular communication systems and the different basic techniques on which such systems are based. Next, we discuss the current state of evolution of cellular systems and the acceptance of CDMA as the multiple access technique in a majority of the next generation systems. We also describe the wireless propagation environment and the key design considerations for

a CDMA system.

Chapter 3 develops the CDMA system model used in the rest of the thesis. The key contribution of this chapter is the incorporation of the effect of both multiple paths and multiple sensors in the model, in an efficient manner.

Chapter 4 presents a maximum likelihood based channel estimation algorithm that has been developed in the multipath and multiple sensors scenario. It also includes the idea of joint synchronization and detection, and discusses certain key issues in the implementation of the proposed algorithm.

Chapter 5 presents a subspace-based algorithm for tracking the channel parameters in a time varying environment. It also deals with various aspects of implementing the algorithm.

Finally, we present our conclusions and define some future directions related to the work presented in this thesis.

Chapter 2

Background

2.1 Wireless cellular systems

In order to accommodate the demand for wireless communication services, efficient use of limited available frequency spectrum is imperative. Cellular systems exploit the loss in power of a transmitted signal, with distance, to reuse a *communication channel* at another spatially separated location. Here, a communication channel is defined by either a frequency band (FDMA), a time slot (TDMA), or a unique code (CDMA). FDMA, TDMA, and CDMA are known as the basic multiple access techniques and will be explained in greater detail later.

The entire coverage area is divided into smaller regions called *cells*, each capable of supporting a subset of all the users subscribing to the cellular system. The size and distribution of the cells [28] are dictated by the coverage area of the *base station*, subscriber density and projected demand within a particular region. Each mobile uses a separate, temporary channel to communicate with the nearest base station. The base station controls operations within a cell and is responsible for serving calls to and from users located in the respective cell. As mobile users travel from cell to cell, their transmissions are *handed off* between cells in order to maintain seamless service. The base stations are connected to the *mobile telephone switching office* (MTSO) that serves as a controller to a group of base stations and as an interface with the fixed *public switching telephone network* (PSTN) or the *integrated services digital network*

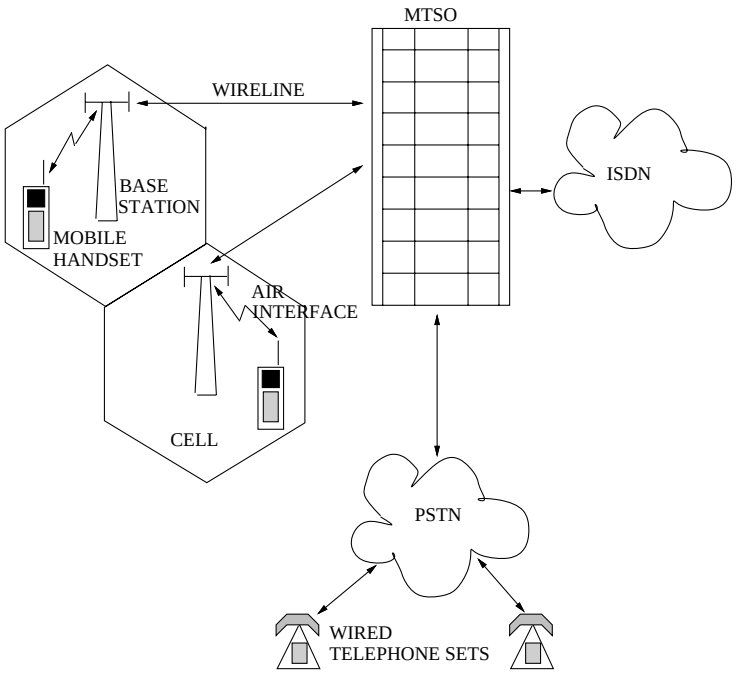


Figure 2.1 : A cellular telephone network

(ISDN) [28]. An illustration of a cellular network [29, 30] is shown in Figure 2.1

Users in a cellular system can be separated using one of three *multiple access schemes* (Figure 2.2): frequency division multiple access (FDMA), time division multiple access (TDMA) and code division multiple access (CDMA). In FDMA, all users transmit simultaneously, but use disjoint frequency bands. In TDMA, all users occupy the same RF (radio frequency) bandwidth, but transmit sequentially in time. The number of users that can be supported is determined by the number of slots available, either in frequency or time. Both the above systems are inherently inflexible in that the number of frequency or time slots (or channels) allocated is fixed *a priori*; this wastes resources when the number of users is less than the number of channels [31].

When users are allowed to transmit simultaneously and occupy the same band-

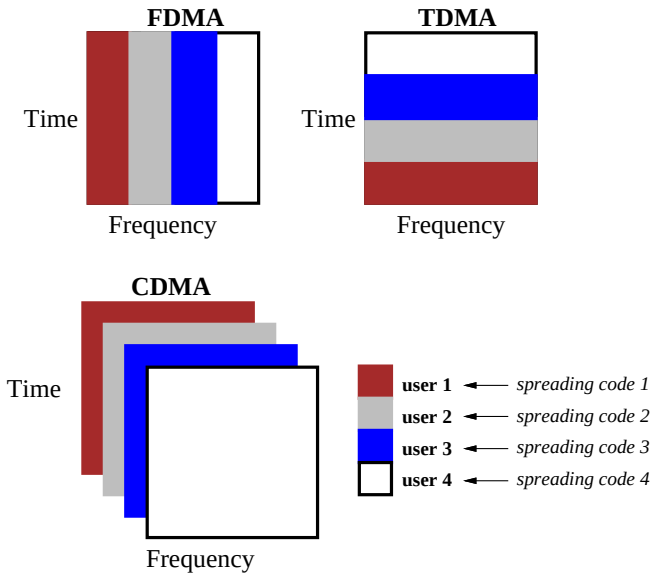


Figure 2.2 : Multiple access schemes

width, some other means of separating the users should be used - Code Division Multiple Access (CDMA) provides such a scheme. All the users occupy the entire time-bandwidth plane with the overall system performance degrading gracefully as the number of users increases. CDMA falls under a class of techniques called *spread spectrum* techniques [5, 32]. The term spread spectrum implies that the data stream is *modulated* or coded so that the overall transmission rate is much higher than the maximum needed to transmit the actual data. Each user's data bits are modulated by a coded waveform that can only be detected at the receiver which also knows the coding function. The two common modulation techniques for spread spectrum are direct sequence (DS) and frequency hopping (FH). In frequency hopping, the carrier is caused to shift frequency according to a pattern. This pattern is also known at the receiver and allows for detection of the signal. Direct sequence CDMA (DS-CDMA) signals are generated by multiplying the desired signal by a larger-bandwidth *spread-*

ing sequence, also called spreading codes (Figure 1.2). Again, the spreading codes are known at the receiver and this allows detection of the data bits.

In order to maximally differentiate between the users, the codes assigned to the users should be mutually orthogonal. In practice, this is not realistic due to the asynchronous manner of operation that CDMA supports, i.e., the bit intervals of the different users do not have to coincide. This makes the design of the spreading codes and also the CDMA system more difficult and recently a large part of the research in this field has focused on this issue [32,33].

Historically, the initial applictions of spread spectrum systems have been in military communications. This is because the spread spectrum nature of the signal results in a number of benefits [5], such as antijamming properties and low probability of intercept. In recent years, there has been a steadily growing interest in using CDMA [34] in mobile radio networks. Certain properties of the spread spectrum waveforms give CDMA some distinct advantages over FDMA or TDMA [35] for cellular systems. The two basic problems which the cellular radio designer faces are multipath fading [36] of the radio link and interference. Direct sequence techniques offer fine resolution in time and can therefore be used to enhance overall performance by diversity combining of the multipath in a rake receiver [36]. Due to their wide bandwidth, spread spectrum signals are also useful in mitigating interference.

The frequency *reuse factor* is another major factor favoring CDMA over either TDMA or FDMA. In the later techniques, the bandwidth used in a particular cell cannot be used in any immediately adjacent cells to prevent excessive interference. For example, in the analog AMPS system [37], a reuse factor of seven is employed. However with DS-CDMA it is possible to have a reuse factor of one, thus alleviating the problem of frequency planning. CDMA also provides a natural way to exploit

the *voice activity factor*, that is the percentage of time during a two-way telephone conversation, each channel is actually used. This bursty nature of the source can be utilized to increase overall system capacity.

Due to all these advantages, DS-CDMA has recently emerged as a commercially viable strategy for cellular wireless communications. In this thesis, we will restrict our discussion of CDMA to DS-CDMA.

As the wireless communications industry has evolved, several standards based on a combination of the various multiple access schemes just discussed have been proposed and deployed for both cellular (849-894 MHz) as well as PCS (1850-1990 MHz) services. This evolution has continued starting from the analog AMPS system to the third generation (3G) systems currently being developed, as the number of cellular subscribers has increased and the demand for higher communication speed, lower cost, size, and power, and higher capacity have continued to push the limits of these existing standards.

2.1.1 Cellular standards

In the U.S., cellular telephone service is provided mostly by the AMPS (Advanced Mobile Phone Service) system [37]. Analog voice signals are frequency modulated onto carriers in the 800MHz band (824-849 for the downlink and 869-894 for uplink). Each user channel is allocated a 30kHz bandwidth. Only a certain set of carrier frequencies are available in a given cell and neighboring cells must use a different set to reduce interference. However the system capacity is only one call per 210kHz. The demand for more efficient cellular services has pushed the limit of this technology and resulted in a number of 'second generation' digital systems, both in the U. S. and in Europe. Table 2.1 shows the increase in data transmission rates [3] as the standards

have evolved. A brief description of some of these systems follows.

System	Generation	Data transmission
GSM (TDMA)	2nd	9.6 kbps
IS-54/IS-136 (TDMA)	2nd	9.6 kbps
IS-95 (CDMA)	2nd	9.6, 14.4 kbps
UMTS (W-CDMA)	3rd	16kbps to 2Mbps

Table 2.1 : Digital wireless data transmission rates

Global System for Mobile Communication (GSM):

The European standard for cellular telephony, GSM [38], was first introduced in 1992. The bands allocated to GSM are 890-915 MHz for downlink and 935-960 MHz for uplink. The band in either direction is divided into 124 frequency channels, each with carriers spaced 200 kHz apart. Each cell site in GSM has a fixed number of two way channels ranging from 1 to usually not more than 15. A FDMA/TDMA channel structure is used. Each 200 kHz FDMA channel uses an aggregate bit rate of 270.833 kbps. The 270 kbps data stream in each FDMA channel is divided into 8 TDMA time slots called *logical channels*. Each slot is 577 μs which corresponds to a transmission time of 156.25 bits though only 148 bits are transmitted in each slot. The remaining time is a guard time to prevent overlapping of signals. The logical channels are organized into a hierarchical frame structure that provides each mobile terminal with a two-way traffic channel and a separate two-way control channel.

TDMA Digital Cellular Standard (IS-54):

As with GSM, the IS-54 radio channel structure is a combination of FDMA and TDMA. The designated frequency channels are the same as that for AMPS, with

carriers spaced 30 kHz apart. This was done to allow conversion of individual AMPS channels to IS-54 channels. Each 30 kHz digital channel has a transmission rate of 48.6 kbps. The 40 ms frame is composed of six such 6.67 ms time slots corresponding to 324 bits each. IS-54 provided a three-fold increase in the system capacity over AMPS, and has been enhanced (IS-54B) to include various other services including authentication (which uses sophisticated encryption technology to combat fraud), caller ID, voice mail indication, etc. One of the most important design constraints in developing the TDMA standard was compatibility and coexistence with AMPS - it was deemed important by carriers that their migration from AMPS to digital cellular be easy to manage on an incremental basis. The IS-54B TDMA standard provides for dual-mode operation, using AMPS control channels with a Digital Traffic Channel, and also supporting analog voice channels. This is also frequently referred to as D-AMPS, or Digital AMPS. Currently, this standard has evolved into the IS-136 D-AMPS system specification, with enhancements, such as an improved speech coder and mobility management techniques.

CDMA Digital Cellular (IS-95):

The North American IS-95 standard [39] specifies the common air interface for a system which uses CDMA technology. In contrast to the IS-54 standard, which uses the same set of carrier frequencies as used by the AMPS system, the CDMA system uses signals with 1.2288 MHz spreading bandwidth, equivalent to 41 AMPS channels. The forward and reverse links use separate frequencies, spaced by 45MHz. In IS-95, the basic user information rate is 9600 bps, spread to a chip rate of 1.2288 Mchip/s (128 times) using a combination of techniques such as, coding, interleaving and spreading. In the forward link, the user data stream is encoded using a rate 1/2 convolutional

code, interleaved, spread by one of 64 orthogonal spreading sequences (Walsh functions) and finally scrambled using a pseudo-random sequence of length 2^{15}. A pilot channel on the forward link allows coherent detection.

The reverse link is handled quite differently as each received signal arrives at the base station via different propagation channels. The user data stream is first convolutionally coded at rate 1/3, interleaved, mapped and then spread fourfold. The mobile's transmit power is tightly controlled, to avoid the *near-far* problem that arises from varying power levels with which the different signals reach the receiver.

At both base station and mobile, rake receivers are used to resolve and combine multipath components. Also, IS-95 allows 'soft' handoff when a mobile moves from one cell to another. The handoff is not abrupt, but rather it is a prolonged call state during which there is communication via two or more base stations. The multi-way communication diversity improves the link performance during the handoff. The diversity gain partially compensates for the large path loss at the cell boundary.

Third Generation Systems:

Emerging requirements for higher data rates and better spectrum efficiency [34,40] are the primary challenges facing the developers of third generation wireless systems. The worldwide effort towards this end is embodied in the development of the IMT-2000 (International Mobile Telecommunications by 2000) standards of ITU (International Telecommunications Union). At the time of the writing of this thesis the exact details of the standard are yet to be finalized. However the focus is to provide very high speed mobile multimedia services: 2Mbps indoor, 384 kbps pedestrian, and 144 kbps vehicular.

To meet the IMT-2000 requirements, several standards are being proposed and

developed by different industrial communities to serve the needs of different markets in the U.S., Europe and Japan. The most relevant feature to us, as CDMA researchers, is the acceptance of CDMA, in one form or the other, in a majority of these proposals.

In the early part of 1998, agreement was reached on the radio interface for the third generation European mobile system [40, 41], UMTS (Universal Mobile Telecommunications System), which has been proposed as the successor to GSM. UMTS has decided to adopt W-CDMA (wideband CDMA) in the paired band and TD-CDMA in the unpaired band. Detailed information on the European system is available at the European Telecommunications Standards Institute (ETSI)'s Special Mobile Group (SMG) website (http://www.etsi.org/smg/smg.htm).

In Japan, wideband CDMA [42] has been accepted as the standard by the Japanese standardization body, ARIB (Association of Radio Industries and Businesses). In the United States, the standard cdma2000 [43] for next generation systems, is an evolution of the IS-95, CDMA based standard.

In this thesis, we develop channel estimation techniques which will improve the performance of future generation systems. It may be noted here, that we assume a short spreading sequence system, that is, the period of the spreading code is the same as the bit period, as opposed to using spreading codes with periods spanning several bits. Short and long code CDMA are sometimes denoted D-CDMA (deterministic CDMA) and R-CDMA (random CDMA) [4, 44, 45]. In a long code system, the correlation between the users changes from bit to bit and the MAI therefore appears to be random in time, causing the performance for different users to be similar and determined by the average interference level. So such systems are suitable for implementation using conventional correlator based single user techniques. The advantage of the conventional correlator based detection schemes is its low complexity and ease

of implementation. These factors have led to the acceptance of long codes for commercial systems such as IS-95 and the first phase of the European third generation system.

However advanced receiver techniques such as interference cancellation and multiuser detection [4] and channel estimation [17, 20] are more easily applied to short code systems to get increased performance, due to the cyclostationarity of the interference. It has been shown in [45] that a short code system with multiuser detection always outperforms a long code system even when the long code system employs some type of interference cancellation. The drawback of such advanced techniques is the high computational complexity. However, the proposal for the European third generation system standard [41] has the provision for exploiting the gains from multi-user detection using short codes. The research community is attempting to prove that the added costs of multiuser techniques are justified by an increase in performance. The goal of this thesis is to devise new multiuser techniques for channel estimation which result in increased performance as well as a reduction in additional computational complexity.

2.2 Wireless channels

The wireless channel in mobile radio poses a great challenge as a medium for reliable high speed communications. When a radio signal is transmitted through a wireless channel, the wave propagates through a physical medium and interacts with physical objects and structures, such as buildings, hills, trees, moving vehicles, etc [29]. The propagation of radio waves through such an environment is a complicated process that involves diffraction, refraction, and multiple reflections. Also, the speed of the mobile impacts how rapidly the received signal level varies as the mobile terminal moves in

space. Modeling all these phenomenon effectively has been one of the most challenging tasks related to wireless system design. Typically it is necessary to utilize statistical models that reasonably approximate the environment, based on measurements made in the field.

A typical mobile radio communication scenario in an urban area usually involves an elevated fixed base-station antenna (or multiple antennas), a mobile handest, a line-of-sight (LOS) propagation path followed by many reflected paths due to the presence of natural and man-made objects between the mobile and the base station. Figure 2.3 illustrates such an environment. The different propagation paths (LOS as well as reflected paths) change with the movement of the mobile unit or the movement of its surroundings [29, 30].

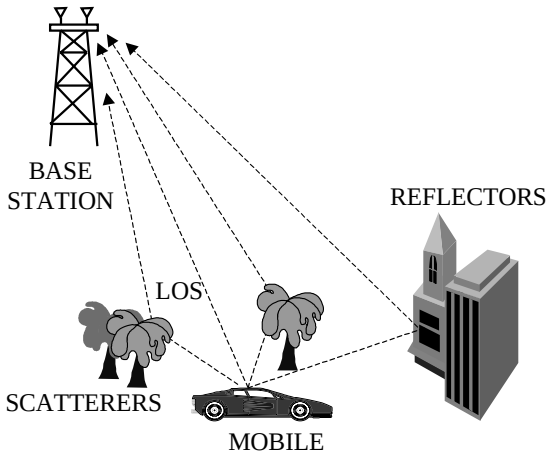


Figure 2.3 : The wireless propagation environment

Radio propagation models usually focus on predicting the average signal strength based upon the separation between the transmitter and the receiver, and also the rapid fluctuations in the instantaneous signal level that may be observed over short distances. The variation of the average signal strength over large distances (typically

several hundreds of meters) is called *large scale path loss*. The rapid fluctuations over short travel distances (typically a few wavelengths) is called *small scale fading*.

2.2.1 Large scale path loss

Both theoretical analysis and experimental measurements indicate that the average large scale path loss ($\bar{PL}$) or decrease in signal strength at the receiver is proportional to some power of the distance between the transmitter and the receiver [29]:

$$\bar{PL}(d) \propto \left(\frac{d}{d_0}\right)^n \tag{2.1}$$

or

$$\bar{PL}(d)[dB] = \bar{PL}(d_0) + 10nlog\left(\frac{d}{d_0}\right) \tag{2.2}$$

where d is the separation between the transmitter and receiver, d_0 is a reference distance which is determined from measurements close to the transmitter, and n is the path loss exponent. The path loss exponent determines the rate at which the path loss increases with the separation d, and its value depends on the specific propagation environment.

The path loss model stated above does not consider the fact that the dynamics of the wireless environment may be quite different at two different locations with the same transmitter-receiver separation. In order to predict the instantaneous signal level, the following model is widely used: the path loss $PL(d)$ at a location is randomly distributed about the average value:

$$PL(d)[dB] = \bar{PL}(d_0) + 10nlog\left(\frac{d}{d_0}\right) + X_\sigma \tag{2.3}$$

where X_σ is a zero-mean Gaussian distributed random variable with standard deviation σ. This effect is known as *log-normal shadowing*.

2.2.2 Small scale fading

Small-scale fading refers to the rapid variations in the amplitude of the received signal, in a wireless environment, over short distances or time intervals, such that the effect of large-scale path loss may be ignored. There are a number of physical factors causing these rapid fluctuations [29,36]:

- Multipath propagation: Due to the presence of a number of reflectors and scatterers between the transmitter and the receiver, in most instances, there may exist more than one path between the transmitter and receiver. These paths may add up either constructively or destructively due to the randomly time varying delays, phases,and attenuation of the various paths. So the amplitude of the composite received signal consisting of various path components may vary over time and give rise to the phenomenon of fading. The parameter of interest when dealing with multiple paths is the *delay spread*. The maximum delay spread is defined as the time delay during which the multipath energy falls to a pre-specified level below the maximum.

- Doppler spread: The relative motion between the base station and mobile as well as the movements of the surrounding objects results in random frequency modulation due to different Doppler shifts of each multipath component. The Doppler shift will be positive or negative depending on the direction of relative motion between the mobile and the base station. The parameter of interest here is the *Doppler spread* or the spectral broadening caused by this phenomenon. Again, it is defined as the range of frequencies over which the received Doppler spectrum is non-zero or above a certain threshold.

- Transmitted signal bandwidth or symbol period: The relative rate of change of

the transmitted signal, given by the symbol period, and the channel characteristics defines the type of fading the channel faces. This issue is elaborated upon in the next sections.

Depending upon the relationship between the signal parameters (bandwidth, symbol period) and the channel parameters (delay spread and Doppler spread), different transmitted signals will undergo different types of small scale fading in different environments.

Flat fading and frequency selective fading:

If the mobile radio channel has a constant gain and linear phase response over a bandwidth which is greater than the signal bandwidth, the received signal undergoes flat fading and in the opposite case it is said to undergo frequency selective fading [29].

In flat fading, the delay spread is much less than the symbol period and hence the spectral characteristics of the transmitted signal are preserved at the receiver. However the strength of the received signal varies with time. In frequency selective fading, the received signal includes multiple copies of the transmitted waveforms, attenuated and delayed in time, and hence the received signal is distorted. Modeling frequency selective fading is more difficult as each multipath signal should be modeled and the channel should be modeled as a linear filter. A typical model is the Rayleigh fading [36, 46] model which considers the channel impulse response to be made up of a number of delta functions which independently fade and have sufficient time delay between them to induce frequency selective fading.

Fast and slow fading:

Depending upon the relative rate of change of the transmitted signal and the channel characteristics, a channel may be fast fading or slow fading [29]. In a fast fading channel the channel impulse response varies rapidly within the symbol duration, that is, the Doppler spread is large relative to the symbol bandwidth. In slow fading, the channel may be assumed to be static over several symbol periods.

2.2.3 Channel related assumptions in this thesis

In this thesis, the channel is modeled as consisting of various multipath components for each user. The impulse response of the multipath channel consisting of P paths, is modeled as:

$$h(t) = \sum_{p=1}^{P} w_p \delta(t - \tau_p) \tag{2.4}$$

where, w_p is the complex attenuation and τ_p is the delay of the p^{th} path. Using this model for the multipath channel impulse response, and $\nu(t)$ as the additive white Gaussian background noise (AWGN), the received signal for a transmitted signal $s(t)$ is:

$$r(t) = \sum_{p=1}^{P} w_p s(t - \tau_p) + \nu(t). \tag{2.5}$$

In general, the channel impulse response will be time varying $h(t, \tau)$. Moreover, each distinct multipath component p in (2.4) may actually be a sum of a number of closely spaced paths due to a number of scatterers. Hence, the time varying complex channel attenuation is $w_p(t, \tau)$ and can be modeled as a zero-mean complex valued Gaussian process [36], when the number of such paths are large. This implies that the envelope $|w_p(t, \tau)|$ at any instant t is Rayleigh distributed.

In this thesis, each multipath component is delayed and attenuated by a different amount. The complex attenuation is either constant over several symbol periods or is slowly time varying according to the Jakes model [46] for Rayleigh fading (Figure2.4). Hence in this thesis, our model includes frequency selective fading and slow fading.

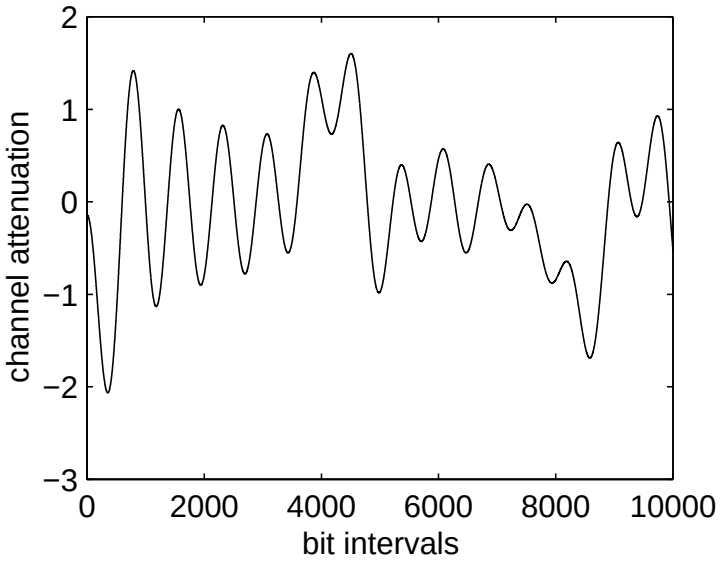


Figure 2.4 : Fading channel attenuation using the Jakes model for a system with 1800 MHz carrier frequency, mobile speed of 50mph, and a bit rate of 100K bits per second.

The scope of the channel estimation problem addressed in this thesis is restricted to short time intervals and hence large scale path loss effects are ignored.

2.3 Antenna array at the base station receiver

Also, in some of the work presented in this thesis (Chapter 4), an antenna array at the receiver is assumed. The front end of the receiver consists of an antenna array of M sensors arranged in a specific geometry [31, 47]. We assume that the direction of propagation of the received signal is approximately the same at each sensor. Each

array element has a different response to the various multipath components of the received signal, depending upon the direction of arrival of the signal and the array geometry. Figure 2.5 shows the schematic for a specific array geometry.

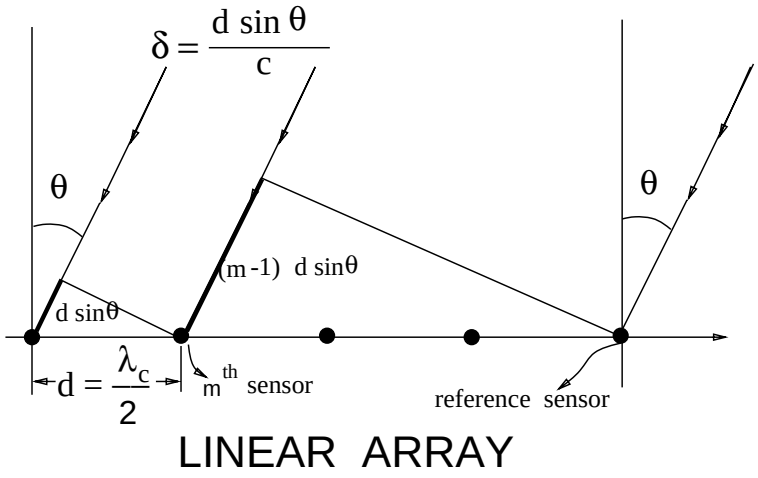


Figure 2.5 : A 5-element uniform linear array with intersensor spacing d. The angle of arrival is θ, c is the speed of light in free space, λ_c is the wavelength, and δ is the difference in arrival times of the signal at two consecutive sensors.

An *a priori* assumption on array geometry is not crucial to the development of the algorithms in this thesis. However we do assume that the envelope characteristics do not vary across the array.

Multipath fading caused by the wireless channel greatly degrades the performance of mobile communication systems. One technique that has been used in practice to combat fading is *diversity*. Diversity implies the exploitation of multiple copies of the transmitted signal at the receiver. The mobile wireless channel provides inherent diversity in the form of multiple paths which is exploited in current CDMA systems [32] through the use of rake receivers [36]. Also for next generation systems [41], the exploitation of spatial diversity, in the form of an antenna array at the base station is currently being investigated. Figure 2.6 shows the benefits to be obtained

by exploiting diversity in a fading environment. In the figure, BER refers to the bit error rate, that is the fraction of the transmitted bits that are detected erroneously at the receiver.

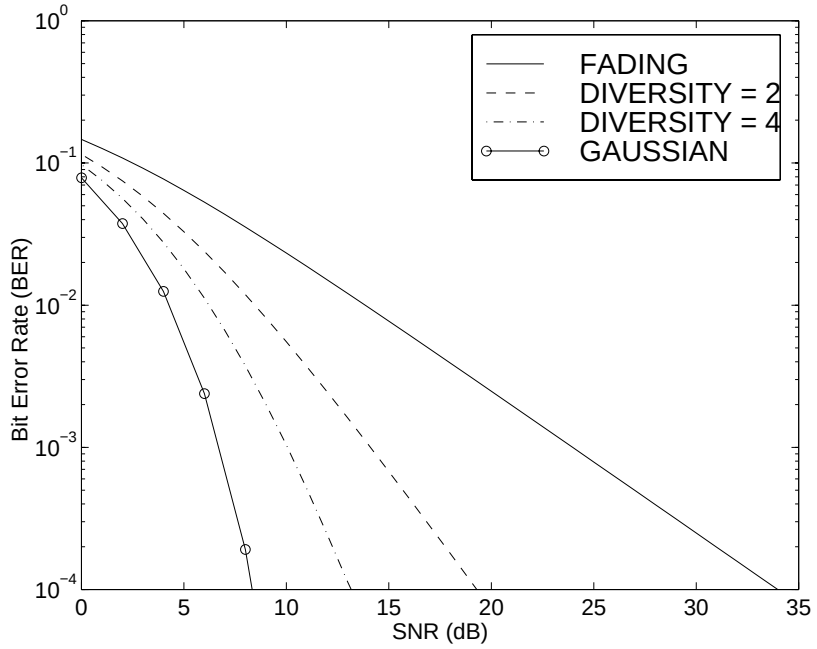


Figure 2.6 : Gains from exploiting diversity: 'FADING': BER in the presence of fading and AWGN without exploiting diversity, 'DIVERSITY=2' and 'DIVERSITY=4': BER exploiting 2 and 4 diversity components, and 'GAUSSIAN': BER in the absence of fading in an AWGN channel.

In order to exploit such diversity, the multipath channel seen at each sensor needs to be estimated. In the next chapter, we will develop a DS-CDMA system model that incorporates the effect of multiple paths as well as multiple sensors at the reciver. In Chapter 4, we will use this model to address the problem of channel estimation in a multipath, multiple sensor system.

2.4 Implementation aspects of CDMA systems

In this section, we will identify the key issues which influence the design of wireless communication systems. In addition we will present details about the hardware structures typically used in such systems.

2.4.1 Key design considerations

Limitations on the power and size of the mobile unit, and the bit-error-rate requirements of various data sources are the major considerations in the implementation of any wireless communication system. For example, the state-of-the-art Qualcomm CDMA (IS-95) Q-phone [3] has a size of 102×25 mm, weighs 156g, and the battery supports a talk time of $2\frac{1}{2}$ hrs. These figures are constantly improving in each newer generation of products.

Today, the weight of a wireless handset is dominated by the battery. Also, the bulk of the power consumption in a cellular terminal is in the power amplifier in the radio [48]. However, as we move toward an era of micro and pico area cells, transmit power will drop dramatically from the current 1-watt range. One of the most straightforward techniques for reducing power is to lower the voltage. Older phones generally operate at 5 volts. New phones typically use 3 volts, with sub-2-volt systems on the horizon. Since power varies as the square of voltage, this represents a dramatic reduction in power consumption. However, this does require both the digital processing unit (DSPs) and the analog circuitry to operate at lower voltages (http://www.ti.com/sc/docs/wireless/97/issues.htm).

There are other techniques for reducing power consumption as well. Systems can be partitioned for better power management by allowing circuitry to be turned off when it is not needed. In addition, various algorithms can be implemented so that

they can be executed with minimum power. Another solution is to perform the major part of the computations in the base stations which are at fixed sites, or to actually increase the processing at the base station with a view to decreasing the processing in the mobile handset [49]. Clock rates and even the design of software can affect power consumption. Finally, at the integrated circuit level, transistor designs and semiconductor processes also influence the power consumption.

The data rates and bandwidths set the throughput requirements of the system [50]. For instance, for the GSM system, described earlier in this chapter, the symbol period can be calculated to be $3.7\mu s$. On the other hand, for the third generation cellular system using CDMA, the proposed chip rate is 4.096Mcps (chip period of $0.2441\mu s$) with variable spreading gain, and hence variable bit rates upto 2Mbps. The bit or chip interval dictates the required speed of execution of the algorithms defining the system. The latency requirements of the system is determined by the data source. For example, in speech communications, there would be a limit on the maximum tolerable latency. On the other hand, data communications have more flexible latency requirements.

The bit-error-rate requirements of the system also depend on the data source and application. For voice communication, a typical cell may be required to support 40 users at 9600bps with a bit-error-rate of 10^{-3}. However, for data communications, the bit rate would be higher and the required bit-error-rate lower.

The continual evolution of wireless standards and the design of faster and more efficient algorithms to meet the existing standards, requires flexibility and reconfigurability in the design of such systems. Also, radios may have to handle multiple frequency bands and even multiple modulation schemes to allow operation under various standards. This will drive the need for integrated, writable, non-volatile memories

in order to shorten design cycles.

At the receiver end, there is a need for continuous collection of received data samples prior to actual processing [48]. This collected data will have to be buffered while actual processing occurs on previously collected data samples. Hence some hardware support such as direct memory access (DMA) is essential for efficient operation.

In addition to the mobile handset, base stations are an essential element of wireless communications, and must provide greater capacity, better quality, and high data rates - all at a lower cost. At the same time, base stations are evolving from enormous macrostations with vast transmitting power to minute, special-purpose picostations with smaller transmitting power but higher processing loads.

A key requirement of base station hardware is the on-chip integration of large amounts of memory. Base stations require substantial blocks of RAM in order to handle rapid changes of data as callers enter and depart cell areas. They also require more processing power to handle a large number of users whereas the mobile handles only one user. Other requirements include low integrated circuit (IC) power consumption to reduce the need for cooling fans and space and to minimize the need for backup power. Base stations also need ICs with multiple advanced peripherals, like enhanced buffered serial ports, for efficient data input-output. Support for multiple DSPs in the same system requires the processors to share memory, buses and peripherals for minimal board space and cost (http://www.ti.com/sc/docs/wireless/97/oem.htm).

Finally, designs of wireless systems should provide for efficient field testing in mobile environments. Large amounts of data need to be collected and analyzed. This is done through rapid movement of data to external devices capable of massive data processing (e,g. the mobile station monitor for testing IS-95 physical layer functionality [48]).

2.4.2 Structure of a wireless transceiver

The specific nature of wireless communication channels (passbands around high operating frequencies) dictates that almost all transceivers for wireless communications consist of three main parts [51, 52]: a front-end which performs the frequency conversion from passband to baseband or vice versa, a modulator/demodulator, and a baseband processing unit. This structure is shown in Figure 2.7. The design and implementation of these three units are most often considered separately. We briefly describe the design issues in each of these three units with respect to the receiver. Similar considerations apply to the transmitter.

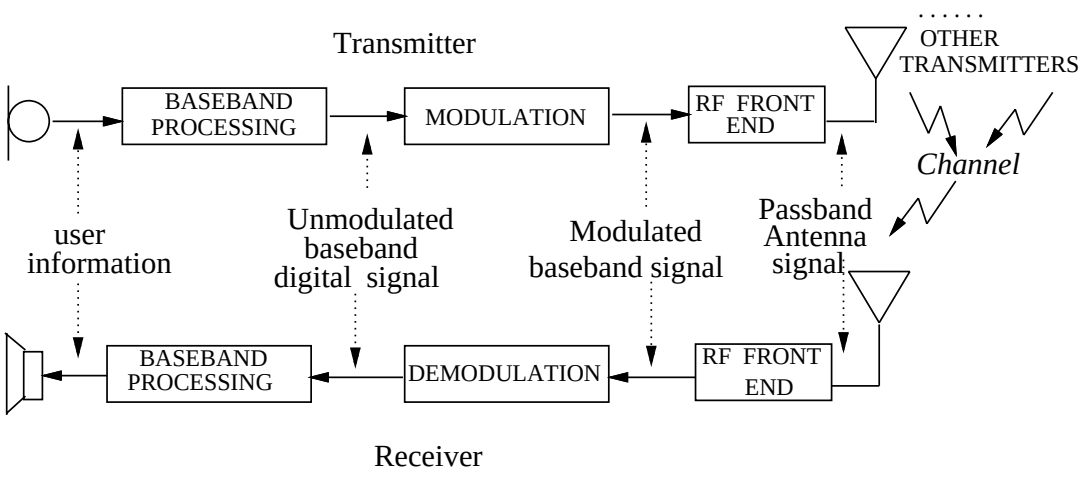


Figure 2.7 : The basic configuration of a transmitter-receiver for wireless communications.

The receiver front-end shown in Figure 2.8 brings the antenna signal down from its high operating frequency to a suitable lower frequency on which the demodulation can be done elegantly. The antenna signal consists of a very broad spectrum of many different information channels and noise sources. The wanted signal is, in its modulated form, only a small part of this broad spectrum. This requires from

the front-end a high operating frequency and a high input dynamic range. These specifications dictate that the front-end is always realized as an analog circuit [53] which mainly consists of mixers, filters and variable gain amplifiers which, in successive stages, filter the antenna signal and downconvert it to lower frequencies. In some curent digital systems [51], the final downconversion from IF to baseband in the frontend is also performed in the digital domain. The front-end has no influence on the form of the wanted signal but only determines the center frequency at which it is given to the demodulator.

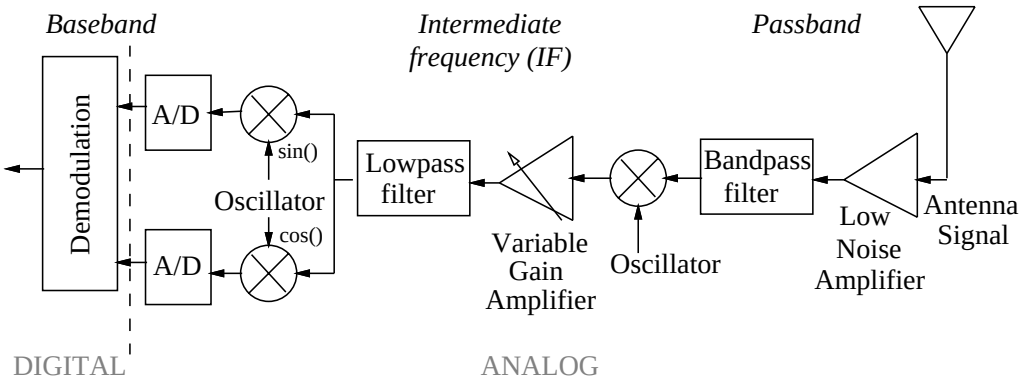


Figure 2.8 : The analog RF frontend.

The form of the output of the front-end depends on the design of the demodulator. In most of the current digital systems, the output of the front-end are digital, baseband, I (in-phase) and Q (quadrature) signals and the demodulation is performed in a DSP, in the digital domain (Figure 2.8). The I and Q components form the real and imaginary part of the digital signal. In other systems, the demodulation is also performed in the analog domain and the analog-to-digital (A/D) converter comes after the demodulator [52]. Other than this consideration, the design of the demodulator depends upon the modulation technique used such as BPSK, QPSK, etc.

The baseband, demodulated, complex, digital signal is next processed by the baseband processing unit. In current digital systems, in most cases, all processing in this unit is totally in the digital domain, implemented using a combination of digital-signal-processors (DSP) and custom ASICs (application specific integrated circuits) [53]. In a typical, CDMA transceiver, the basic building blocks of the baseband processing unit are shown in Figure 2.9. The individual blocks in the baseband

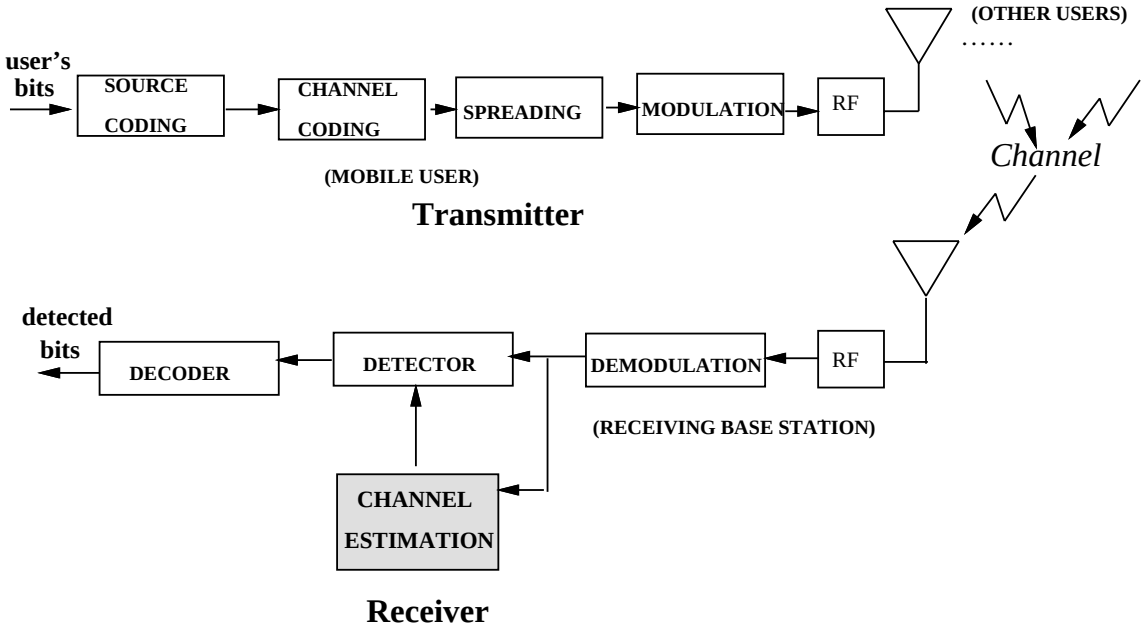


Figure 2.9 : The structure of the baseband processing unit in the transmitter and receiver

processing unit will be described in greater detail in the next chapter.

The first two units discussed above do not change drastically from standard to standard - the basic design principles of these two units can most often be considered separately from the wireless techniques and algorithms used, which primarily dictate the design and implementation of the baseband processing unit. Large strides have been made in recent years regarding the design of the first two units using low power

analog and mixed-signal CMOS technology, but this is beyond the scope of this thesis. The work in this thesis is concentrated on the processing involved in the channel estimation module within the baseband processing unit shown in Figure 2.9.

2.4.3 Architectures for wireless transceivers

The key design issues in wireless sytems design discussed in Section 2.4.1 clearly point to a low power, fast, as well as programmable VLSI device for such systems. This role has been filled by digital signal processors (DSPs), especially for the second generation digital standards, such as GSM and IS-136. It is expected that DSPs will continue to play a central role in the design of hardware for third generation systems.

DSPs are specialized processors designed for implementation of extensive numerical computations for digital signal processing functions through resident firmware / software, that ensure execution time predictibilty and hard real-time constraints [54]. Current high-end microprocessors are not optimized for specialized DSP functions. They are also expensive devices with significant power consumption. On the other hand, current microcontrollers are more suited for embedded systems [55] such as automobile electronics and home appliances. However microcontrollers are more suitable than DSPs for control instructions such as branching. For this reason, in a typical cellular phone design, it is usual to dedicate master and control functionality to microcontrollers [48]. However DSPs with their low cost, low power consumption, and ability to handle numeric tasks are the most suitable device for physical layer baseband functions such as coding/decoding, detection and channel estimation.

DSPs use extensive pipelining, on-chip memories, parallel functional units, separate program and data buses (Harvard architecture [54]), and specialized instructions such as single cycle multiply-accumulate and bit reversed addressing. DSPs are either

floating or fixed point arithmetic devices. Fixed point devices are cheaper, faster, and consume less power. Hence wireless systems, especially handsets, tend to use fixed point devices. Fixed point arithmetic requires programmers to pay attention to precision, scaling, and overflows due to quantizations effects. This issue has been addressed in more detail for algorithms developed in this thesis (Sections 4.6 and 5.7).

In spite of the rapid progress in the design of DSPs, digital baseband platforms for wireless systems continue to be a mix of microcontrollers and DSPs as well as custom ASICs. The choice between DSPs and ASICs is driven by constraints on execution time, cost, and size. For example, IS-136 specifications make it very suitable for implementation on DSPs (e.g. Texas Instrument (TI)'s TMS320C54x based Digital Baseband Platform [48], http://www.ti.com/sc/graphics/wireless/97/baseband/fig3.gif). On the other hand, the higher chip rate for IS-95 requires that a considerable portion of the baseband functionality be implemented in an ASIC [48].

The higher processing requirement of base stations are also being handled by DSPs optimized for such operations. Examples include Lucent's DSP 16210 [56] and TI's TMS320C6x [57] family of DSPs.

Chapter 3

Development of the Extended DS-CDMA System Model

In Section 2.2, the key features of the wireless channel were described, namely, multiple propagation paths, Doppler shifts, and the various forms of small scale fading. In this chapter, we will develop a system model that incorporates the structure of the transmitted signal in a CDMA system, the effects of the channel, and the structure of the received signal at each sensor of an antenna array at the receiver. The key contribution of this chapter is the extension of the channel model widely used in the literature [17,29] to incorporate the effect of both multiple paths and multiple sensors in an efficient manner.

We assume a K-user direct sequence CDMA system with BPSK (Binary Phase Shift Keying) modulation with each transmitted signal selected from a binary alphabet and limited to $[0, T]$, where T is the symbol period. Each user transmits a zero mean stationary bit sequence with i.i.d. components.

3.1 Continuous time received signal

The complex baseband representation of the k^{th} user's transmitted signal is given by

$$s_k(t) = \sqrt{P_k} \sum_{i=1}^{L} b_{k,i} c_k(t - iT) \,, \tag{3.1}$$

where P_k is the transmitted power, $b_{k,i} \in \{+1, \ -1\}$ is the i^{th} transmitted bit and $c_k(t)$ is the spreading waveform. The parameter L denotes the number of bits being

considered. The spreading or code waveform is composed of N chips and if we assume BPSK for the spreading modulation we have $c_k(t) = \sum_{n=0}^{N-1} c_{k,n}\Pi(t - nT_c)$, where $c_{k,n} \in \{+1, -1\}$ and the chip pulse waveform $\Pi(t)$ is a rectangular pulse of duration T_c. We will assume that the extent of the spreading code is one bit period and hence we have $T = NT_c$.

The frontend of the receiver consists of an antenna array of M sensors arranged in a specific geometry. The corresponding array response vector, which is the response to a propagating plane wave impinging on the array at an angle θ, with the vertical is :

$$[\rho^{(1)}(\theta), \cdots, \rho^{(m)}(\theta), \cdots, \rho^{(M)}(\theta)]. \tag{3.2}$$

and is determined by the geometry of the array. Here $\rho^{(m)}(\theta)$ is the response of the m^{th} sensor to a wave with angle of arrival θ. In this thesis, no *a priori* assumption is made on the geometry of the array. However, it is assumed that the time taken by the signal to traverse the physical array is much smaller than the inverse of the message bandwidth and hence, the envelope characteristics of the signal do not vary across the array (Figure 3.1).

We assume that the channel for each user consists of P distinct and resolvable propagation paths [17]. The model does not assume chip synchronicity, that is the delays of the signals, introduced by the channel are not integral multiples of chips (Figure 3.2).

The impulse response $h_k(t)$ of the channel seen by user k is given by:

$$h_k(t) = \sum_{p=1}^{P} w_{k,p}\delta(t - \tau_{k,p}) \tag{3.3}$$

where, $w_{k,p}$ is the complex amplitude with which the p^{th} path of the k^{th} user is received and includes contributions from the channel attenuation and the phase offset

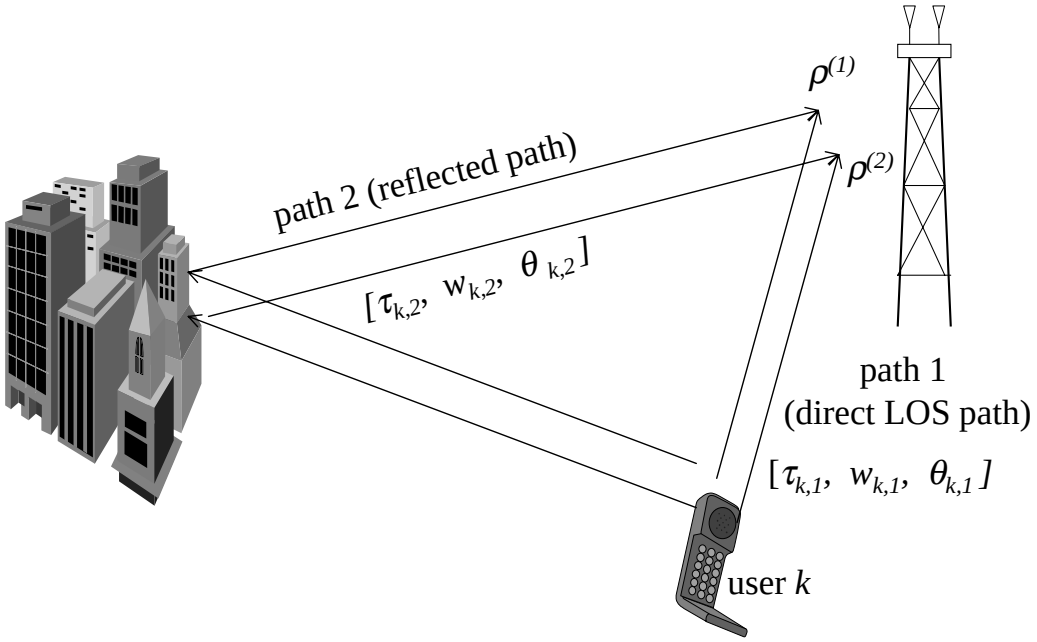


Figure 3.1 : Two propagation paths (a direct line-of-sight path and a reflected path) from a user k, received by 2 sensors at the base station receiver. Each path p, has a different delay $\tau_{k,p}$, amplitude $w_{k,p}$, and direction of arrival $\theta_{k,p}$. Each sensor m, has a different response $\rho^{(m)}(\theta)$.

and $\tau_{k,p}$ is the relative delay with respect to a reference at the receiver. The channel parameters are assumed to be unknown but constant in the time taken to estimate them. The multipath spread of the channel T_m is the maximum difference between propagation delays, that is, $T_m = \max_{p,p'} |\tau_{k,p} - \tau_{k,p'}|$. In our model, we assume that the multipath spread is less than half the symbol period ($T_m < \frac{1}{2}T$).

Accordingly, the received signal at the base station is a superposition of multiple copies of attenuated and delayed signals transmitted by all the K users. Therefore,

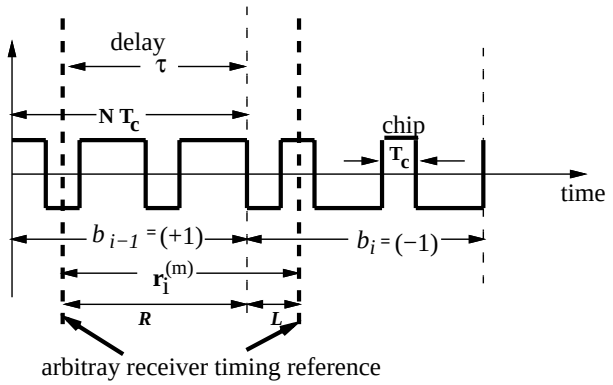


Figure 3.2 : System model - received signal.

the signal at the m^{th} sensor is given by:

$$r^{(m)}(t) = \sum_{k=1}^{K} \sum_{p=1}^{P} w_{k,p} s_k(t - \tau_{k,p}) \rho^{(m)}(\theta_{k,p}) + \nu^{(m)}(t) \tag{3.4}$$

where $\theta_{k,p}$ is the direction of arrival of the p^{th} path of user k with respect to an axis in the plane of the array.

The additive noise is assumed to have some correlation across the array. We use the following model for the noise [19]: $\nu^{(m)}(t) = \beta\eta(t) + (1-\beta)\eta^{(m)}(t)$. The component $\eta(t)$ is completely correlated across the array and can arise from other-cell interference - this component is taken to be the same at each sensor; $\eta^{(m)}(t)$ is the noise element independent from sensor to sensor, arising from a combination of receiver noise and the uncorrelated contributions from other-cell transmissions ($\beta \in [0, 1)$). The noise components $\eta(t)$ and $\eta^{(m)}(t)$ are uncorrelated random processes and each is assumed to be white and Gaussian with zero-mean and double-sided spectral density of $\mathcal{N}_0/2$. Modeling the other-cell interference as Gaussian is valid in light of the number of such interferers and the applicability of the central limit theorem.

3.2 Discretized received signal

The continuous time signal at each sensor is then discretized [17, 31] by sampling the output of a chip-matched filter, at the chip rate. The chip-matched filtering is a simple integrate and dump operation, over a time interval equal to the chip period:

$$r^{(m)}[n] = \frac{1}{T_c} \int_{nT_c}^{(n+1)T_c} r^{(m)}(t)dt \tag{3.5}$$

Observation vectors $\mathbf{r}_i^{(m)}$ are formed by collecting together N successive outputs $r^{(m)}[n]$. Hence, each observation vector corresponds to a time interval equal to the bit period. Since each spreading vector is periodic with period N, $r^{(m)}[n]$ is wide sense cyclostationary, and the observation vectors $\mathbf{r}_i^{(m)}$ are wide sense stationary. The observation vector, $\mathbf{r}_i^{(m)} \in \mathbb{C}^N$, at time i is

$$\mathbf{r}_i^{(m)} = \left[r^{(m)}[iN],\ r^{(m)}[iN+1],\ \ldots,\ r^{(m)}[iN+N-1]\right]^\top. \tag{3.6}$$

The system is asynchronous and the receiver has an arbitrary timing reference which will not be aligned to actual transmitted bit boundaries. Hence, each observation vector can be viewed as a linear combination of $2K$ *signal vectors* - 2 components from each user due to the past and current bits as shown in Figure 3.2. In Figure 3.2, the received vector $\mathbf{r}_i^{(m)}$ overlaps the two adjacent transmitted bits b_{i-1} and b_i. So the contribution of the spreading code of the user to vector $\mathbf{r}_i$ appears in two parts - the right part (shown as R) of bit b_{i-1} and the left part (shown as L) of bit b_i. Expressing $\mathbf{r}_i^{(m)}$ in terms of its various components:

$$\mathbf{r}_i^{(m)} = \mathbf{A}^{(m)}\mathbf{b}_i + \boldsymbol{\nu}_i^{(m)},\quad \boldsymbol{\nu}_i^{(m)} \sim \mathcal{N}(\mathbf{0}, \mathbf{K}^{(m)}). \tag{3.7}$$

The $2K$ length vector $\mathbf{b}_i$ is of the form : $\mathbf{b}_i = [b_{1,i-1}, b_{1,i}, \ldots, b_{K,i-1}, b_{K,i}]'$, where $b_{k,i}$ is the i^{th} bit of the k^{th} user. The matrix $\mathbf{A}^{(m)}$ is a function of the spreading codes of all the users and the channel parameters.

3.3 Multipath channel model for each sensor

In this section, we will develop the structure of matrix $\mathbf{A}^{(m)}$ to efficiently model the channel as seen at each sensor. Let us consider the columns of the matrix $\mathbf{A}^{(m)}$. As explained in Figure 3.2, the contribution of the spreading code of the user to vector $\mathbf{r}_i^{(m)}$ appears in two parts - R and L. Figures 3.3 and 3.4 shows the various components of the received observation vector and the contributions of one path of one user to the matrix $\mathbf{A}^{(m)}$. Figure 3.3 shows the ideal case when the received signal is chip synchronous, that is the delay is an integral multiple of a chip. In this case, the observation vector is a linear combination of two vectors - one corresponding to the right part of bit i and the other corresponding to the left part of bit $i+1$. The weights of the linear combination is the same in both cases and consists of the channel attenuation and the array response. Figure 3.4 shows the actual case of non-integar delay when the signal is not assumed to be chip synchronous. Here, each of the left and right parts are linear combinations of two other vectors. The two vectors are required, to incorporate the fact that $\mathbf{r}_i^{(m)}$ does not start from a chip boundary. The weights for the linear combination of the two vectors that contribute to the right part of bit i are γ and $1-\gamma$ where γ is the fractional part of the delay.

Hence, matrix $\mathbf{A}^{(m)}$ has columns corresponding to the two parts: denoted by the superscripts R and L:

$$\mathbf{A}^{(m)} = [\mathbf{a}_1^R \quad \mathbf{a}_1^L \quad \cdots \quad \mathbf{a}_K^R \quad \mathbf{a}_K^L]. \tag{3.8}$$

Each of the columns $\mathbf{a}_k^R$ and $\mathbf{a}_k^L$ for each user k, are functions of the corresponding delays ($\tau_{k,1}$ to $\tau_{k,P}$), attenuation factors ($w_{k,1}$ to $w_{k,P}$), and array responses ($\rho_{k,1}^{(m)}$ to $\rho_{k,P}^{(m)}$). Here, $\rho_{k,p}^{(m)} = \rho^{(m)}(\theta_{k,p})$.

If T_c is the chip period, let $\tau_{k,p}/T_c = q + \gamma$, $q \in \{0, 1, \cdots, N-1\}, \gamma \in [0, 1)$, we

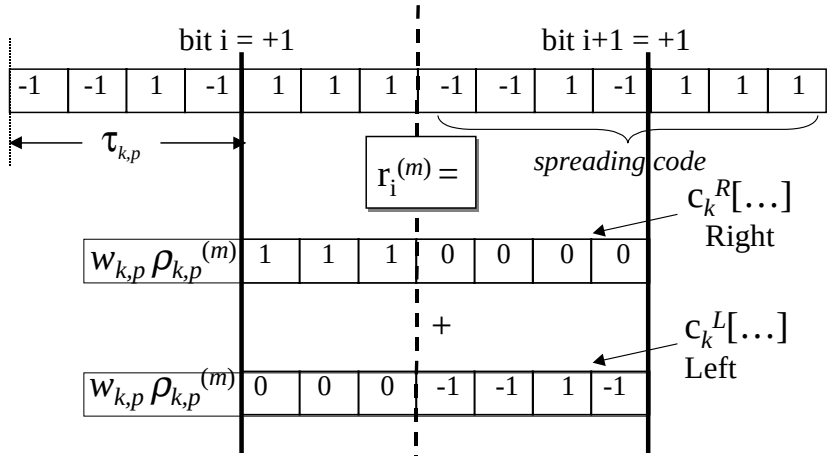


Figure 3.3 : Contributions of the spreading code, channel parameters and transmitted bits to the received signal - integer delay case.

have [17] (Figure 3.4):

$$\mathbf{a}_k^R = \sum_{p=1}^{P} [w_{k,p}\{(1-\gamma)\mathbf{c}_k^R[q] + \gamma \mathbf{c}_k^R[q+1]\}\rho_{k,p}^{(m)}]$$

$$\mathbf{a}_k^L = \sum_{p=1}^{P} [w_{k,p}\{(1-\gamma)\mathbf{c}_k^L[q] + \gamma \mathbf{c}_k^L[q+1]\}\rho_{k,p}^{(m)}] \tag{3.9}$$

where $\mathbf{c}_K^R[q]$ and $\mathbf{c}_K^L[q]$ are the spreading codes shifted by integer (multiples of chips) delays.

$$\mathbf{c}_k^R[q] = [c_{k,N-q} \cdots c_{k,N-1}\, 0 \cdots 0]'$$

$$\mathbf{c}_k^L[q] = [0 \cdots 0\, c_{k,0} \cdots c_{k,N-q-1}]' \tag{3.10}$$

The matrix $\mathbf{A}^{(m)}$ is a function of the spreading codes of the users which are known at the receiver as well as the channel parameters which are unknown at the receiver. Extracting the channel parameters from the received signal vectors using the model

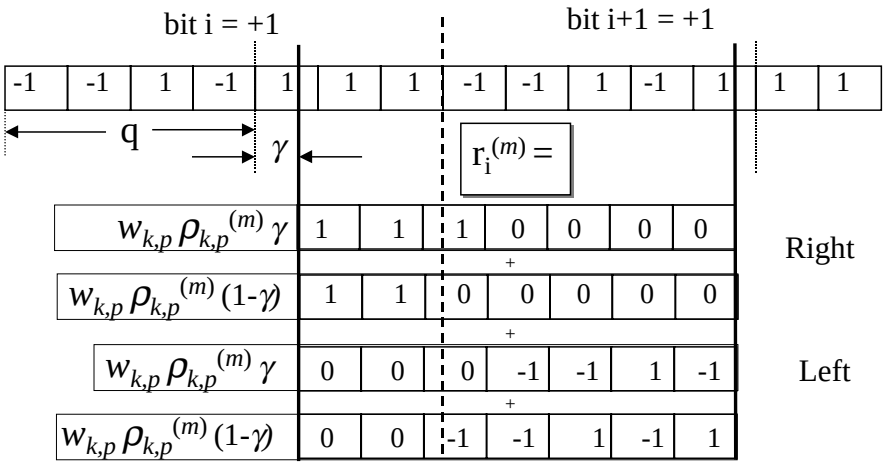


Figure 3.4 : Contributions of the spreading code, channel parameters and transmitted bits to the received signal - non-integer delay case.

(3.7) is difficult as the matrix $\mathbf{A}^{(m)}$ is a complex non-linear function of the channel parameters. Hence we will process $\mathbf{A}^{(m)}$ such that the known spreading codes and the unknown channel parameters are separated into two matrices. This will facilitate the process of estimation of the unknown channel parameters.

We note that (3.8) and (3.9) show that the columns of $\mathbf{A}^{(m)}$ are linear combinations of $\mathbf{c}_k[q]$'s. Let $\mathbf{U}_k^R \in \mathbb{C}^{N\times N}$ and $\mathbf{U}_k^L \in \mathbb{C}^{N\times N}$ be matrices formed from the spreading codes, delayed by all possible integer delays. Let $\mathbf{z}_k^{(m)} \in \mathbb{C}^{N\times 1}$ be the composite channel impulse response vector corresponding to the m^{th} sensor, consisting of the weight vectors in the linear combinations of the $\mathbf{c}_K[q]$'s, expressed in (3.9). Hence, we can write $\mathbf{U}_k^R$, $\mathbf{U}_k^L$, and $\mathbf{z}_k^{(m)}$ as

$$\mathbf{U}_k^{(R)} = [\mathbf{c}_k^{(R)}[0] \cdots \mathbf{c}_k^{(R)}[N-1]],$$

$$\mathbf{U}_k^{(L)} = [\mathbf{c}_k^{(L)}[0] \cdots \mathbf{c}_k^{(L)}[N-1]], \tag{3.11}$$

$$\mathbf{z}_k^{(m)} = \mathbf{p}_k^{(m)} \circ \mathbf{h}_k, \tag{3.12}$$

where,

$$\mathbf{h}_k = \begin{bmatrix} 0 & \\ \vdots & \\ w_{k,1}(1-\gamma_{k,1}) & \leftarrow \ q_{k,1}^{th} element \\ w_{k,1}\gamma_{k,1} & \leftarrow \ (q_{k,1}+1)^{th} element \\ \vdots & \\ w_{k,P}(1-\gamma_{k,P}) & \leftarrow \ q_{k,P}^{th} element \\ w_{k,P}\gamma_{k,P} & \leftarrow \ (q_{k,P}+1)^{th} element \\ \vdots & \\ 0 & \end{bmatrix} \tag{3.13}$$

and where,

$$\mathbf{p}_k^{(m)} = \begin{bmatrix} 0 & \\ \vdots & \\ \rho_{k,1}^{(m)} & \leftarrow \ q_{k,1}^{th} element \\ \rho_{k,1}^{(m)} & \leftarrow \ (q_{k,1}+1)^{th} element \\ \vdots & \\ \rho_{k,P}^{(m)} & \leftarrow \ q_{k,P}^{th} element \\ \rho_{k,P}^{(m)} & \leftarrow \ (q_{k,P}+1)^{th} element \\ \vdots & \\ 0 & \end{bmatrix} \tag{3.14}$$

Here the operator '$\circ$' denotes the element-wise multiplication of the two vector operands. Now, the columns of $\mathbf{A}^{(m)}$ can be expressed in terms of $\mathbf{U}_k^R$, $\mathbf{U}_k^L$, and $\mathbf{z}_k^{(m)}$

as

$$\mathbf{a}_k^R = \mathbf{U}_k^R \mathbf{z}_k^{(m)}, \quad \mathbf{a}_k^L = \mathbf{U}_k^L \mathbf{z}_k^{(m)}. \tag{3.15}$$

We can now rewrite matrix $\mathbf{A}^{(m)}$ as

$$\mathbf{A}^{(m)} = [\mathbf{U}_1^R \mathbf{z}_1^{(m)} \;\; \mathbf{U}_1^L \mathbf{z}_1^{(m)} \cdots \mathbf{U}_K^R \mathbf{z}_K^{(m)} \;\; \mathbf{U}_K^L \mathbf{z}_K^{(m)}]. \tag{3.16}$$

Now (3.7) is expressed as $\mathbf{A}^{(m)} = [\mathbf{U}_1^R \mathbf{z}_1^{(m)} \;\; \mathbf{U}_1^L \mathbf{z}_1^{(m)} \cdots \mathbf{U}_K^R \mathbf{z}_K^{(m)} \;\; \mathbf{U}_K^L \mathbf{z}_K^{(m)}]\mathbf{b}_i + \boldsymbol{\nu}_i^{(m)}$. In this section, we presented a model that captures the effect of the multipath channel, at each sensor. In the next section, we will combine the observations at each sensor in order to develop a composite model that captures multipath as well as multiple sensors.

3.4 Multipath channel model for multiple sensors

The observation vector $\mathbf{r}_i$, of length MN, across the array, is formed by concatenation of the observations at each sensor. This composite vector $\mathbf{r}_i = [\mathbf{r}_i^{(1)'}, \cdots \mathbf{r}_i^{(k)'}, \cdots \mathbf{r}_i^{(K)'}]'$ can be expressed as:

$$\mathbf{r}_i = \begin{bmatrix} \mathbf{U}_1^R \mathbf{z}_1^{(1)} & \mathbf{U}_1^L \mathbf{z}_1^{(1)} \cdots \mathbf{U}_K^R \mathbf{z}_K^{(1)} & \mathbf{U}_K^L \mathbf{z}_K^{(1)} \\ & \vdots & \\ \mathbf{U}_1^R \mathbf{z}_1^{(M)} & \mathbf{U}_1^L \mathbf{z}_1^{(M)} \cdots \mathbf{U}_K^L \mathbf{z}_K^{(M)} & \mathbf{U}_K^L \mathbf{z}_K^{(M)} \end{bmatrix} \mathbf{b}_i + \boldsymbol{\nu}_i. \tag{3.17}$$

where the noise vector $\boldsymbol{\nu}_i$ is formed from the components $\boldsymbol{\nu}_i^{(m)}$ and is assumed to have an unknown covariance of $\mathbf{K}$.

Let $\mathcal{U}_k^R \in \mathbb{C}^{MN \times MN}$ and $\mathcal{U}_k^L \in \mathbb{C}^{MN \times MN}$ be matrices defined as follows

$$\mathcal{U}_k^R = I_M \otimes \mathbf{U}_k^R, \qquad \mathcal{U}_k^L = I_M \otimes \mathbf{U}_k^L. \tag{3.18}$$

Here the operator '$\otimes$' represents the Kronecker product of two matrices. Also let $\mathbf{z}_k \in \mathbb{C}^{MN\times 1}$ be a vector defined as

$$\mathbf{z}_k = [\mathbf{z}_k^{(1)'}, \cdots, \mathbf{z}_k^{(m)'}, \cdots, \mathbf{z}_k^{(M)'}]'. \tag{3.19}$$

Using these matrices defined above, the columns of the combined code and channel matrix, corresponding to user k, in the expression for $\mathbf{r}_i$ (3.17) can be written as $\mathcal{U}_k^R\mathbf{z}_k$ and $\mathcal{U}_k^L\mathbf{z}_k$. This helps us to rewrite the expression for $\mathbf{r}_i$ as

$$\mathbf{r}_i = [\mathcal{U}_1^R\mathbf{z}_1 \quad \mathcal{U}_1^L\mathbf{z}_1 \cdots \mathcal{U}_K^R\mathbf{z}_K \quad \mathcal{U}_K^L\mathbf{z}_K]\mathbf{b}_i + \boldsymbol{\nu}_i$$

$$\mathbf{r}_i = \mathcal{U}\mathbf{Z}\mathbf{b}_i + \boldsymbol{\nu}_i \tag{3.20}$$

where, $\mathcal{U} = [\mathcal{U}_1^R \;\; \mathcal{U}_1^L \cdots \mathcal{U}_k^R \;\; \mathcal{U}_k^L \cdots \mathcal{U}_K^R \;\; \mathcal{U}_K^L]$ and $\mathbf{Z} = diag(\mathbf{z}_1, \mathbf{z}_1, \cdots, \mathbf{z}_K, \mathbf{z}_K)$.

The advantage to be gained from expressing $\mathbf{r}_i$ as (3.20) is that it allows easy modeling of multiple propagation paths without increasing the size of any of the matrices involved. Increasing the sizes of the matrices involved is directly related to an increase in the computational load of the algorithm. An alternative method to model multiple paths within (3.7), is to treat the multiple paths as different 'virtual users' and have two additional columns in matrix $\mathbf{A}$ for each extra path of each user. Hence, as the number of paths increases the size of matrix $\mathbf{A}$ also increases. However in (3.20), the size of matrices $\mathcal{U}$ and $\mathbf{Z}$ does not increase, as P increases; instead matrix $\mathbf{Z}$ becomes more dense. In effect, we discretize the multipath delay axis of the impulse response [29] into N equal time delay segments, each corresponding to a chip interval. Any number of multipath signals received within the n^{th} bin are represented by a single resolvable multipath component.

The model in (3.20) efficiently captures the effect of multipath and multiple sensors. Increasing the number of paths does not increase the sizes of any of the matrices

involved and the observations at all the sensors are captured in a single received vector. Moreover, the matrix $\mathcal{U}$ is a function of the known spreading codes and $\mathbf{Z}$ has all the unknown parameters of all the paths of all the users from all the sensors.

In the next chapter we will describe an algorithm based on the maximum likelihood technique in which $\mathbf{Z}$ will be estimated from the observations $\mathbf{r}_i$'s, the known sequence of transmitted bits (e.g., a preamble at the beginning of each frame), and the knowledge of the spreading codes of the different users.

In Chapter 5, we will use the same model developed in this chapter, but for the single sensor case ($M = 1$). Hence when expressing the received signal we will drop the superscript (m) from (3.7) for convenience.

$$\mathbf{r}_i = \mathbf{A}\mathbf{b}_i + \boldsymbol{\nu}_i, \qquad i = 1, 2, \cdots . \tag{3.21}$$

In Chapter 5, we will use this model for a subspace based channel tracking algorithm. The subspace based algorithm requires the knowledge of the spreading codes of the different users but does not require the transmission of a preamble.

Chapter 4

Maximum Likelihood Channel Estimation

In this chapter, we address the problem of channel estimation in the complex scenario involving multiple users, multiple paths and multiple sensors at the receiver. We will use the system model developed in Chapter 3 and develop a ML based technique for the estimation of the channel impulse response vector $\mathbf{z}_k$ for each user. The algorithm is based upon the ML technique for angle-of-arrival estimation presented in [58] and for single path, multiple sensor CDMA channel estimation presented in [19].

We use the ML technique to develop a novel channel estimation algorithm that estimates the composite channel impulse response vector for each user in the presence of multiple propagation paths using multiple sensors. We solve the complex problem involving both multiple paths as well as multiple sensors in a computationally efficient manner, without increasing the size of the matrices involved as the number of propagation paths increases.

4.1 The channel estimation algorithm

When an observation vector $\mathbf{r}_i$ depends on a parameter vector $\boldsymbol{\xi}$, that is either deterministic but unknown or whose a priori statistics are unknown, the maximum likelihood estimate of the parameter $\boldsymbol{\xi}$ is often used. The maximum likelihood estimate of the parameter is given by [47]

$$\hat{\boldsymbol{\xi}}_{ML} = \arg \max_{\boldsymbol{\xi}} p(\mathbf{r}_i|\boldsymbol{\xi}). \tag{4.1}$$

In our problem, $\mathbf{r}_i$ is a function of the channel vectors $\mathbf{z}_k$, the noise covariance matrix $\mathbf{K}$, which is assumed unknown and the transmitted bits $\mathbf{b}_i$. (The variable $\mathbf{K}$ used to denote the noise covariance matrix should not be confused with the number of users K.) We assume that the bits $\mathbf{b}_i$ are known since, otherwise, the maximization of the likelihood function as a function of all the above unknowns is an ill-posed problem. This is accomplished in the acquisition phase by requiring that all the users transmit training sequences or in the tracking phase by the receiver operating in a decision-directed mode.

Given L observations $\mathbf{r}_1\,, \mathbf{r}_2\,, \ldots\,, \mathbf{r}_L$, these are conditionally independent given $\mathbf{b} = [\mathbf{b}_0'\,, \mathbf{b}_1', \ldots\,, \mathbf{b}_L']'$, the transmitted bits. We form their joint conditional probability density function as

$$p(\mathbf{r}_1\,, \mathbf{r}_2\,, \ldots \mathbf{r}_L \mid \mathcal{U}, \mathbf{Z}, \mathbf{b}, \mathbf{K}) = \frac{1}{\pi^{MNL}|\mathbf{K}|^L}\ \exp\left\{ -\sum_{i=1}^{L} (\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)^H \mathbf{K}^{-1} (\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i) \right\} \tag{4.2}$$

where $|\cdot|$ represents the determinant operator. The corresponding log-likelihood function Λ is

$$\Lambda = -\ln|\mathbf{K}| - \ \mathrm{tr}\left\{ \frac{1}{L} \sum_{i=1}^{L} (\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)^H \mathbf{K}^{-1} (\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i) \right\}, \tag{4.3}$$

where $\mathrm{tr}(\cdot)$ represents the trace operator.

As the first step, maximization of the log-likelihood function is to be carried out with respect to $\mathbf{K}$. The maximization is achieved by the following value of $\mathbf{K}$ [47]:

$$\widehat{\mathbf{K}}(\mathcal{U}, \mathbf{Z}) = \frac{1}{L} \sum_{i=1}^{L} (\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)(\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)^H. \tag{4.4}$$

Substituting this into (4.3), in the next step, we find that we need to maximize $-\ln|\widehat{\mathbf{K}}|$ or minimize $|\widehat{\mathbf{K}}|$ over all $\{\mathbf{Z}\}$. The cost function that we need to minimize

with respect to $\mathbf{Z}$ is now:

$$\mathcal{L} = |\widehat{\mathbf{K}}| = |\frac{1}{L}\sum_{i=1}^{L}(\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)(\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)^H|. \tag{4.5}$$

Details regarding derivation of the cost function (4.5) by maximizing the log likelihood function (4.3) with respect to $\mathbf{K}$ is outlined in Appendix 4.8.

Direct minimization of (4.5) with respect to $\mathbf{Z}$ is rather intractable and hence, it is carried out indirectly in two steps [19, 58]:

(i) Capture the effect of all the unknowns in a single complex matrix $\mathcal{Y} \triangleq \mathcal{U}\mathbf{Z}$. Form the unconstrained ML estimate of $\mathcal{Y}$, given by $\widehat{\mathcal{Y}}$.

(ii) Obtain the estimates $\hat{\mathbf{z}_k}$ by minimizing the weighted least squares fit between $\mathcal{Y} = \mathcal{U}\mathbf{Z}$ and its unstructured estimate, $\widehat{\mathcal{Y}}$.

4.1.1 Step 1: Covariance approximation

Following the algorithm development in [19, 58], $\widehat{\mathcal{Y}}$ which denotes the unconstrained ML estimate of $\mathcal{Y}$, can be derived by minimizing the cost function (4.5), with respect to $\mathcal{Y}$.

Such a minimization yields expressions for $\widehat{\mathcal{Y}}$ and $\widehat{\mathbf{K}}(\mathcal{Y})$ in terms of certain sample correlation matrices as

$$\widehat{\mathcal{Y}} = \widehat{\mathbf{R}}_{rb}\widehat{\mathbf{R}}_{bb}^{-1}, \tag{4.6}$$

$$\widehat{\mathbf{K}}(\widehat{\mathcal{Y}}) = \widehat{\mathbf{R}}_{rr} - \widehat{\mathcal{Y}}\widehat{\mathbf{R}}_{rb}^H. \tag{4.7}$$

The sample correlation matrices used in (4.6) and (4.7) are defined as

$$\widehat{\mathbf{R}}_{rr} = \frac{1}{L}\sum_{i=1}^{L}\mathbf{r}_i\mathbf{r}_i^H, \qquad \widehat{\mathbf{R}}_{br} = \frac{1}{L}\sum_{i=1}^{L}\mathbf{b}_i\mathbf{r}_i^H, \qquad \widehat{\mathbf{R}}_{bb} = \frac{1}{L}\sum_{i=1}^{L}\mathbf{b}_i\mathbf{b}_i'. \tag{4.8}$$

Thus the unconstrained estimates are derived completely from the received observations and a knowledge of the training sequences. Details regarding derivation of expressions for $\widehat{\mathcal{Y}}$ and $\widehat{\mathbf{K}}(\mathcal{Y})$ by minimizing the cost function (4.5), are outlined in Appendix 4.8.

4.1.2 Step 2: Channel impulse response estimation

The expressions for $\widehat{\mathcal{Y}}$ and $\widehat{\mathbf{K}}$ are substituted into the cost function. The modified cost function is then optimized with respect to $\mathbf{Z}$. This process yields an elegant expression for the estimate of the channel impulse response of each user [19, 58] as follows

$$\hat{\mathbf{z}}_k = \arg\min_{\mathbf{z}_k} \left[(\mathbf{y}_{2k-1} - \hat{\mathbf{y}}_{2k-1})^H \widehat{\mathbf{K}}^{-1} (\mathbf{y}_{2k-1} - \hat{\mathbf{y}}_{2k-1}) + (\mathbf{y}_{2k} - \hat{\mathbf{y}}_{2k})^H \widehat{\mathbf{K}}^{-1} (\mathbf{y}_{2k} - \hat{\mathbf{y}}_{2k}) \right]. \tag{4.9}$$

For details regarding derivation of (4.9), please see Appendix 4.8.

For each user we have contributions for right and left signal vectors, i.e., the $2k^{th}$ and $(2k-1)^{th}$ columns of $\mathcal{Y}$. Let us recall that the columns of $\mathcal{Y}$ are nothing else but the columns of matrix $\mathcal{U}\mathbf{Z}$ as expressed in (3.20): $\mathbf{y}_{2k-1} = \mathcal{U}_k^R \mathbf{z}_k, \;\; \mathbf{y}_{2k} = \mathcal{U}_k^L \mathbf{z}_k$ Hence, the estimate, $\widehat{\mathbf{z}_k}$, of $\mathbf{z}_k$ is :

$$\arg\min_{\mathbf{z}_k} \left[(\mathcal{U}_k^R \mathbf{z}_k - \hat{\mathbf{y}}_{2k-1})^H \widehat{\mathbf{K}}^{-1} (\mathcal{U}_k^R \mathbf{z}_k - \hat{\mathbf{y}}_{2k-1}) + (\mathcal{U}_k^L \mathbf{z}_k - \hat{\mathbf{y}}_{2k})^H \widehat{\mathbf{K}}^{-1} (\mathcal{U}_k^L \mathbf{z}_k - \hat{\mathbf{y}}_{2k}) \right]. \tag{4.10}$$

Solving this optimization problem using Lemma 1 in Appendix 4.8, we obtain an expression for $\hat{\mathbf{z}}_k$ as follows

$$\hat{\mathbf{z}}_k^H = (\hat{\mathbf{y}}_{2k-1}^H \widehat{\mathbf{K}}^{-1} \mathcal{U}_k^R + \hat{\mathbf{y}}_{2k}^H \widehat{\mathbf{K}}^{-1} \mathcal{U}_k^L)(\mathcal{U}_k^{R'} \widehat{\mathbf{K}}^{-1} \mathcal{U}_k^R + \mathcal{U}_k^{L'} \widehat{\mathbf{K}}^{-1} \mathcal{U}_k^L)^{-1}. \tag{4.11}$$

So, we have a closed form expression for the channel impulse response vector $\hat{\mathbf{z}}_k$ for each user. In the next section we will show how this estimated channel impulse

response vector can be used directly in the detection process, expoiting both multipath and spatial diversity.

4.1.3 Extraction of channel parameters

Most existing literature [17,19,20] on channel estimation techniques focus on estimating individual parameters for each user, such as delay and attenuation factors for one or multiple paths. For our model, in the *single sensor* scenario, extraction of individual parameters from the estimated channel impulse response (4.11) is a relatively easy task [21]. In Appendix 4.8 we provide an algorithm, following [17], for extraction of individual channel parameters (delays and amplitudes) from the estimated channel impulse response vector, *with a single sensor*.

However when multiple sensors are used, and the various multipath components arrive at the receiver from different directions, the problem of fitting the estimate $\hat{\mathbf{z}_k}$ to the parametric channel model is much more complex. To simplify the computation, we propose to use the estimate $\hat{\mathbf{z}_k}$ directly in detection algorithms. Using this approach it will be possible to exploit the benefits of spatial diversity from using multiple sensors, without having to extract the individual delays, amplitudes, and directions of arrival of all the paths of each user.

In the next section we show how the estimated composite multipath channel impulse response from multiple sensors can be used directly in the detection process in an efficient manner. This will not only avoid the computation involved in extracting individual parameters but also avoid any error arising from the extraction process. Also, this technique facilitates the design of a base station receiver which will estimate the channel and detect the information bits while exploiting and accounting for multiple paths and multiple sensors. Moreover, unlike the conventional approach [32],

the chip matched filter output used for channel estimation can be directly used for detection, instead of realigning the received signal according to the delays of each users and then performing the code matched filter for detection. This will save further computation from the overall channel estimation and detection process.

4.2 Use of the channel impulse response vector in detection

In this section, we explore the use of the composite channel impulse response, from the multiple sensors, in the conventional detector and in a linear multiuser detector. We propose to use the channel impulse response vector directly in the detection algorithms, instead of first extracting individual parameters. We restrict ourselves to *single-shot* detectors; however, the gains from using multiple sensors in this case can also be extended to sequence detection.

The conventional detector for the received signal at a single sensor, in a single path environment, is a bank of K code-matched filters. For the k^{th} user, the matched filter matched to the single path is

$$y_k^{(1)}(i) = \int_{iT}^{(i+1)T} \mathrm{w}_k r^{(1)}(t) c_k(t - iT - \tau_k) dt, \tag{4.12}$$

where $\mathbf{y}_k^{(1)}(i)$ is the matched filter output at a single sensor for the k^{th} user at the i^{th} time step. The outputs of the matched filters yield *soft* estimates of the transmitted bits. The final *hard* decisions are the sign of these *soft* estimates.

The discretized version of the received signal at a single sensor, for a single path for each user, is expressed as $\mathbf{r}_i^{(1)} = \mathbf{A}^{(1)}\mathbf{b}_i + \boldsymbol{\nu}_i^{(1)}$, where the superscript (1) always denotes the instance $M = 1$. Using this discretized signal model, the equation for the matched filter output, for all the users, is expressed in vector form as :

$$\mathbf{y}_i^{(1)} = (\mathbf{A})^{(1)^H} \mathbf{r}_i^{(1)}. \tag{4.13}$$

It follows from Chapter 3 that, in a single path environment, and for a single sensor ($M = 1$), $\mathbf{A}^{(1)} = (\mathcal{U}\mathbf{Z})^{(1)}$. Using this expression for $\mathbf{A}^{(1)}$ we can rewrite the above equation in terms of the channel impulse response matrix:

$$\mathbf{y}_i^{(1)} = (\mathcal{U}\mathbf{Z})^{(1)^H} \mathbf{r}_i^{(1)}. \tag{4.14}$$

It can be shown quite easily that (4.14) continues to hold for the matched filter output in the multipath case, when the received signal at each sensor is given by (3.4). As in Chapter 3, $\mathbf{Z}$ incorporates the information about all the paths of all the users.

In this model, equation (4.14) is similar to the notion of a *received effective signature waveform*, described in [59,60]. The received effective signature waveform is the spreading code of a user modified with the corresponding channel parameters, which is exactly the result of $(\mathcal{U}\mathbf{Z})$, for all the users.

Extending the same idea to multiple sensors, the combined output of a matched filter receiver at each of the sensors, is given by:

$$\mathbf{y}_i = (\mathcal{U}\mathbf{Z})^H \mathbf{r}_i. \tag{4.15}$$

where $\mathbf{y}_i \in \mathbb{C}^{2K}$. Since all the delay as well as the angle of arrival information is captured in $\mathbf{Z}$, and $\mathbf{z}_k^{(m)} = \mathbf{p}_k^{(m)} \circ \mathbf{h}_k$, (4.15) can be interpreted as a code-matched filter operation at each sensor, followed by an 'angle-matched filter' or conventional beamformer. Thus the compound matched filters at each sensor are matched to the users' delays as well as directions of arrival. This matched filter would be optimum for the multisensor case for a single user only, in an additive white Gaussian noise (AWGN) channel.

The idea of using the composite channel impulse response vector in detection can also be applied to linear multiuser detectors, with multiple sensors [31]. Linear

multiuser detectors apply a linear mapping G, to the soft output of the matched filters to reduce the multiple access interference seen by each user.

Therefore, the vector of decision statistics for all the users, from a linear *single-shot* detector, $G \in \mathbb{C}^{2K \times 2K}$, is given by :

$$\mathbf{x}_i = G\mathbf{y}_i = G(\mathcal{U}\mathbf{Z})^H \mathbf{r}_i. \tag{4.16}$$

The required bit estimates are the hard decision: $\hat{\mathbf{b}}_i = sign(\mathbf{x}_i)$. According to our system model, $\mathbf{b}_i$ contains consecutive bits of each user. Hence the decision statistic for each user will consist of two *soft* estimates for each bit of the user. The two *soft* estimates can be combined such that the output signal-to-interference ratio is maximized [31], to obtain a *hard* decision for each bit.

The decorrelating detector, G_{dec} provides the maximum likelihood linear estimate of the received bits, given the output of the code-matched filters [7]. Hence, it follows that:

$$G_{dec} = ((\mathcal{U}\mathbf{Z})^H(\mathcal{U}\mathbf{Z}))^{-1}. \tag{4.17}$$

The benefits of incorporating multiple sensors in the decorrelating detector has been analyzed in [31] in the single path case. In a following section, we will report results of simulations that show the performance gains obtained from such a detector, using the composite channel impulse response vector, in the multipath case.

4.3 Joint channel estimation and detection

The detection scheme presented in Section 4.2 where the estimated channel impulse response is directly used in detection is a single-shot technique. Single-shot implies that the detector deals with one symbol period at a time and does not account for

the fact that two consecutive symbols of each interferer overlaps with each symbol of the desired user. Performance gains may be expected if this asynchronous nature [23] of the system can be accounted for by detecting a block of bits for all the users simultaneously.

In this section we will present a scheme for such multi-shot detection [61]. Let us recall from Chapter 3, the model for the received signal at an antenna array is given by:

$$\mathbf{r}_i = [\mathcal{U}_1^R \mathbf{z}_1 \quad \mathcal{U}_1^L \mathbf{z}_1 \cdots \mathcal{U}_K^R \mathbf{z}_K \quad \mathcal{U}_K^L \mathbf{z}_K]\mathbf{b}_i + \boldsymbol{\nu}_i$$

$$\mathbf{r}_i = \mathcal{U}\mathbf{Z}\mathbf{b}_i + \boldsymbol{\nu}_i, \tag{4.18}$$

where, $\mathbf{b}_i = [b_{1,i-1}, b_{1,i}, \ldots, b_{K,i-1}, b_{K,i}]'$. Rearranging the matrix and vector elements in (4.18), we can have:

$$\mathbf{r}_i = \begin{bmatrix} \mathcal{U}_1^R \mathbf{z}_1 & \cdots & \mathcal{U}_K^R \mathbf{z}_K & \mathcal{U}_1^L \mathbf{z}_1 & \cdots & \mathcal{U}_K^L \mathbf{z}_K \end{bmatrix} \begin{bmatrix} b_{1,i-1} \\ \cdots \\ b_{K,i-1} \\ b_{1,i} \\ \cdots \\ b_{K,i} \end{bmatrix} + \boldsymbol{\nu}_i. \tag{4.19}$$

In order to do multi-shot detection, the above model should be extended to include multiple bits. Let us assume that we will consider D bits at a time ($i = 1, 2, \cdots, D$). So, we form the multi-shot received vector $\mathbf{r}$ of length MND by concatenating D $\mathbf{r}_i$-s ($i = 1, 2, \cdots, D$). The vector $\mathbf{r}$ can be written as:

$$\begin{bmatrix} \mathcal{U}_1^R\mathbf{z}_1 \cdots \mathcal{U}_K^R\mathbf{z}_K & \mathcal{U}_1^L\mathbf{z}_1 \cdots \mathcal{U}_K^L\mathbf{z}_K & 0 & 0 & 0 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ 0 & 0 & 0 & \mathcal{U}_1^R\mathbf{z}_1 \cdots \mathcal{U}_K^R\mathbf{z}_K & \mathcal{U}_1^L\mathbf{z}_1 \cdots \mathcal{U}_K^L\mathbf{z}_K \end{bmatrix} \begin{bmatrix} b_{1,1} \\ \cdots \\ b_{K,1} \\ b_{1,2} \\ \cdots \\ b_{K,2} \\ \cdots \\ b_{1,D} \\ \cdots \\ b_{K,D} \end{bmatrix}$$

$$+\boldsymbol{\nu}. \tag{4.20}$$

Using this model for the received signal, equations (4.15) for the matched filter and (4.16) for linear detectors may be applied to $\mathbf{r}$ instead of to $\mathbf{r}_i$ to achieve multi-shot detection. It may be noted that, it has been shown [62] that in practice, a window size D of about 11 is sufficient to achieve good performance in such systems. In addition, the estimate of the first and last bit in the window may be ignored by using a sliding window, in order to avoid *edge effects*. The edge effect refers to the error in the estimation of the first and last bits as the D length window ignores the effect of the 0^{th} bit and $(D+1)^{th}$ bit. Also, the first matrix on the left hand side of (4.20) has a block Toeplitz structure and can benefit from the techniques presented in [63] to reduce the computational complexity of linear multiuser detectors.

The detection scheme presented above incorporates the benefits of multiple paths, multiple sensors as well as multi-shot detection. In addition, since the estimated

channel impulse response vector $\hat{\mathbf{z}_k}$ is directly used in the detection process (Sections 4.1.3 and 4.2) instead of first extracting the individual parameters, we avoid any error that would arise from the parameter extraction process.

Closer inspection of (4.20) reveals that we can carry this process one step further. The expression for the received signal and hence the detectors consist of the term $\mathcal{U}_k^R\mathbf{z}_k$ or $\mathcal{U}_k^L\mathbf{z}_k$, instead of $\mathbf{z}_k$ in isolation. Actually, the terms $\mathcal{U}_k^R\mathbf{z}_k$ or $\mathcal{U}_k^L\mathbf{z}_k$ are the spreading code of a user modified with the corresponding channel parameters or the *received effective signature waveform* [59, 60]. These two are the terms of primary interest to the detector and not the individual channel parameters or even the composite response $\mathbf{z}_k$.

So, instead of estimating individual parameters or the composite response $\mathbf{z}_k$, it is sufficient if the channel estimation step estimates $\mathcal{U}_k^R\mathbf{z}_k$ and $\mathcal{U}_k^L\mathbf{z}_k$. It may be recalled from Section 4.1, these two terms are the columns of matrix $\widehat{\mathcal{Y}}$, the estimation of which is the first step (4.6) in the maximum likelihood algorithm developed in Section 4.1.

Using the developed algorithm to estimate $\widehat{\mathcal{Y}}$ is quite straightforward, and saves computation when compared to estimating $\mathbf{z}_k$ or the individual parameters. The savings in computation comes from reducing the ML channel estimation algorithm to computation of the sample correlation matrices followed by computation of the effective spreading code (Section 4.1.1). The performance will also improve as the error from the estimation of $\mathbf{z}_k$ from $\mathcal{U}\mathbf{Z}$ and extraction of the individual parameters can be avoided. This is shown in Section 4.5.2.

4.4 Computational complexity

The computational complexity of the various steps of the ML channel estimation algorithm are :

- Covariance approximation
  - Calculation of sample correlation matrices in (4.8) - $O(M^2N^2)$, $O(MNK)$, $O(K^2)$
  - Calculation of $\widehat{\mathcal{Y}}$, $\widehat{\mathbf{K}}(\widehat{\mathcal{Y}})$ in (4.6) and (4.7) - $O(K^3)$, $O(MNK^2)$, $O(KM^2N^2)$
- Channel impulse response estimation in (4.11) - $O(M^3N^3)$
- Extraction of parameters, for single sensor case (Section 4.1.3) - $O(N)$

From the list of the complexity of all the steps in the algorithm, shown above, it is evident that the overall complexity of the algorithm is $O(M^3N^3)$. However the complexity will be reduced further if $\widehat{\mathbf{K}}$ is assumed to be identity [21] (the noise being additive white Gaussian) instead of explicitly calculating the covariance. This whiteness assumption is practical not only because thermal noise is often assumed white Gaussian but also because the residual noise due to other-cell interference can be considered spatially and temporally white.

Other than eliminating the computation of $\widehat{\mathbf{K}}$, this whiteness will also reduce the complexity of the channel impulse response estimation step to $O(M^2N^2)$. The inverse of $\widehat{\mathbf{R}}_{bb}$ can be estimated using matrix inversion update algorithms ($O(K^2)$), using the Sherman-Morrison-Woodbury formula [64], or can be pre-calculated as the preamble is a known sequence of bits. After these computation-saving approximations, the dominant computation will be the matrix-vector multiplication with complexity $O(M^2N^2)$ which can benefit greatly from well-known parallelization techniques on a number of processors [64]. It has been shown that the parallelization of matrix-vector multiplication is scalable with the number of processors used and a speedup of a factor of C can be obtained by using C processors.

Using the joint channel estimation and detection scheme (Section 4.3) it is possible to avoid the computation to estimate $\mathbf{z}_k$ and to extract the parameters from $\mathbf{z}_k$. The resulting reduction in computation is demonstrated in Figure 4.1.

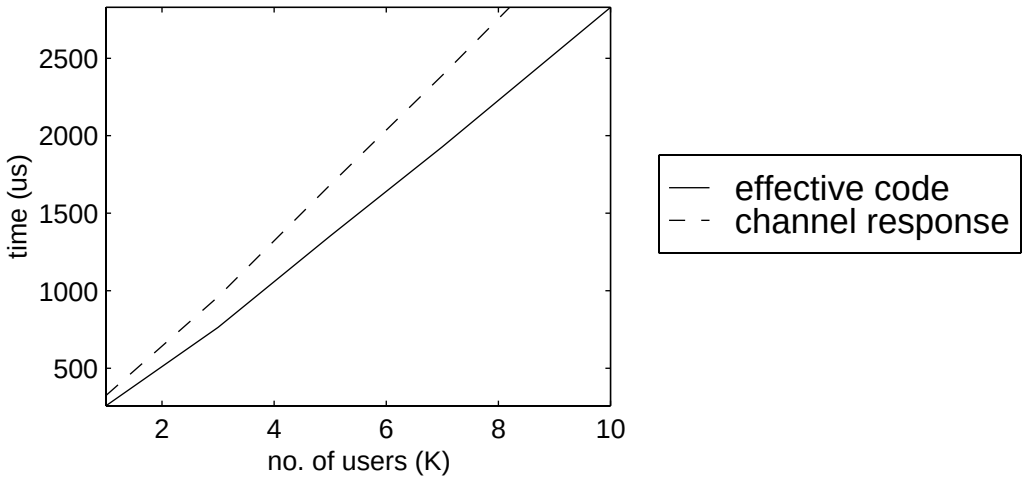


Figure 4.1 : Computation reduction using the joint scheme: Execution time, on the TI DSP TMS320C67 simulator, to estimate the effective spreading code $\mathcal{U}_k\mathbf{z}_k$ and to estimate the channel response $\mathbf{z}_k$. $N = 31, MAI = 20dB, SNR = 10dB, L = 150, P = 1, M = 1$.

4.5 Simulation results

In this section, we describe the simulations that we conducted to evaluate the performance of the proposed estimators. Gold codes of length $N = 31$ were used as spreading codes in all the simulations. The delays of all the users were assumed uniformly distributed in $[1, 31)$ chips. The multiple access interference presented by each interferer, which is the ratio of the interferer's and desired user's received energies, was uniformly distributed in $[0, MAI]$dB. The number of paths for each user is denoted by P, the number of observations by L, the number of users by K, and the signal-to-noise ratio of the background noise at each sensor by SNR. The number

of sensors in the antenna array at the receiver is denoted by M and the parameter β defines the extent of the correlation across the array, as explained in the previous chapter.

4.5.1 Comparison with other estimators - single sensor case

We compare the proposed algorithm with previously proposed channel parameter estimation algorithms for a single sensor in a multipath environment. The single sensor scenario has been chosen as the competing techniques used in this experiment do not address the problem of channel estimation in the presence of multiple paths as well as multiple sensors. The performance of our algorithm in this complex scenario is presented later.

In this experiment, channel parameter estimation is performed using the proposed ML algorithm, a subspace-based algorithm presented in [17], and the conventional sliding correlator [5]. The channel estimates from the various sources as well as the exact value of the channel parameters used in the simulations are then passed to a matched filter receiver to detect the bits. The bit error rates obtained from this experiment are shown in Figure 4.2. The proposed algorithm performs consistently better than the competing strategies. However, the subspace based algorithm does not require a preamble but is more computationally complex and suffers from being dimension-limited, in terms of the number of users it can handle.

Several forms of the ML technique have been proposed in the literature. These algorithms work in different situations such as single path and single sensor [18, 20] or single path and multiple sensors [19, 65, 66]. Out of these, [20, 65, 66] fall under a distinct class which can be easily separated from our proposed algorithm. All these three algorithms model all the interfering users as colored noise, thus reducing

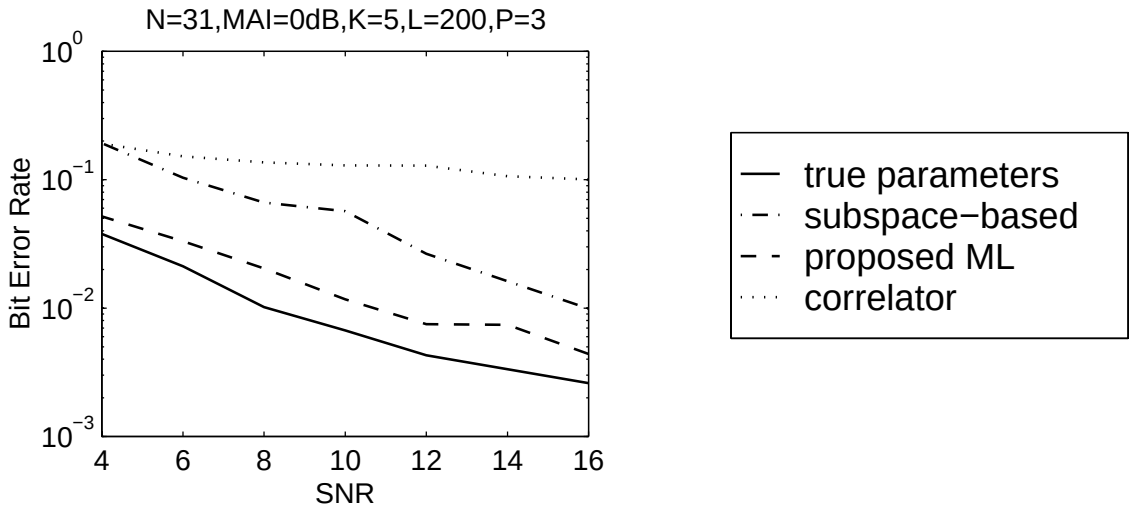


Figure 4.2 : Multipath, single sensor case: BER using a matched filter receiver and multipath channel parameters using true parameters or perfect channel knowledge, estimates from a subspace-based algorithm, estimates from the proposed ML algorithm, and estimates using a sliding correlator.

the size of the spreading code matrix and training sequence vector and hence the required computation. However, since all the interfering users are modeled as colored noise, it will not be possible to reduce computation by assuming white noise, as is possible in our proposed algorithm (Section 4.4). Due to the use of colored noise to model interfering users, these algorithms are more suitable for the downlink where the spreading codes of the interfering users are unknown.

On the other hand, the algorithm in [65] uses a different structure for the observation vectors while forming the joint conditional density functions. This structure exploits the spatial diversity provided by the multiple sensors in such a way that a very short training sequence (about 20 bits) can be used to achieve acquisition.

We have compared (Figure 4.3) the proposed algorithm to the ML scheme proposed in [20], called large scale maximum likelihood algorithm (LSML) which works in the single path and single sensor case. The figure shows the root mean square

error (RMSE) in the estimate of the delay of the weakest user after acquisition has occurred. Acquisition is defined as $Pr[|\tau - \hat{\tau}| < 0.5T_c]$. Again, the proposed algorithm performs better, mainly because the LSML algorithm models all the interfering users as colored noise whereas the proposed algorithm explicitly takes into account the structure of the interfering signals. In addition, the proposed algorithm works in the presence of multipath and multiple sensors, unlike LSML. Both algorithms require a preamble of about 30-65 bits, for the given system parameters. However, the proposed ML algorithm performs much better than LSML when the preamble size is very small.

However it should be noted that none of these competing techniques address the problem of channel estimation in the presence of multiple paths as well as multiple sensors. The performance of our algorithm in this complex scenario is presented later.

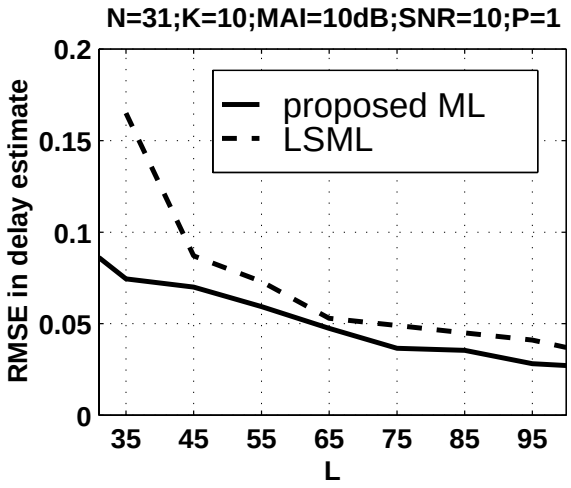


Figure 4.3 : Single sensor, single path case: Proposed algorithm vs. LSML [17]

4.5.2 Use of the channel impulse response vector in detection

In this section, we illustrate the performance benefits to be obtained from using the estimated channel impulse reponse vector, directly in the detection step, instead of first extracting individual parameters.

Figure 4.4 shows the bit error rate (BER) obtained from the matched filter receiver as the SNR is increased. The results for the single sensor case show that the performance actually improves if the channel impulse response is directly used in the detector as in (4.14) (plot labeled 'z directly, M=1') instead of first extracting the individual parameters from the composite channel impulse response and then using them as in (4.12) (plot labeled 'parameters, M=1'). The technique outlined in Appendix 4.8 was used to extract the individual delays and amplitudes of all the paths of all the users from the estimated channel impulse response of each user.

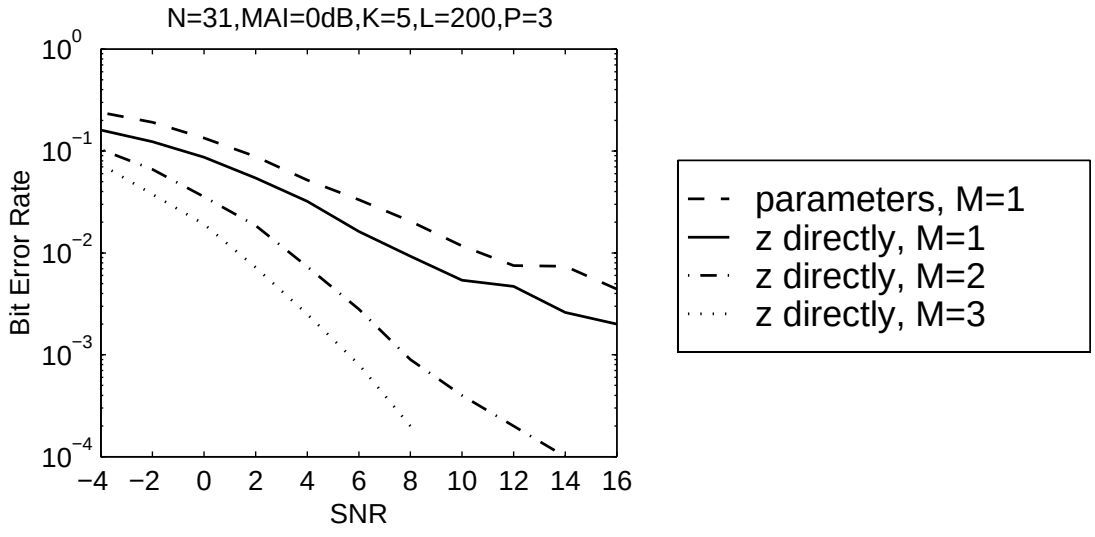


Figure 4.4 : Multipath, multiple sensors (M) case: BER from a matched filter receiver, using composite channel impulse response ($\hat{\mathbf{z}_k}$) directly, compared to the case where the individual parameters are extracted from $\hat{\mathbf{z}_k}$ and then used in the matched filters. Also the direct $\hat{\mathbf{z}_k}$ method has been used with multiple sensors (M). $\beta = 0.7$.

Figure 4.4 also shows that the performance of the ML algorithm followed by matched filter detection improves dramatically as the number of sensors are increased. In the experiments described in this section and the next, the direction of arrivals were assumed to be uniformly distributed in $[-60^\circ, 60^\circ]$, corresponding to one sector of 120° in a cell. The values of the rest of the system parameters are specified along with the graphs.

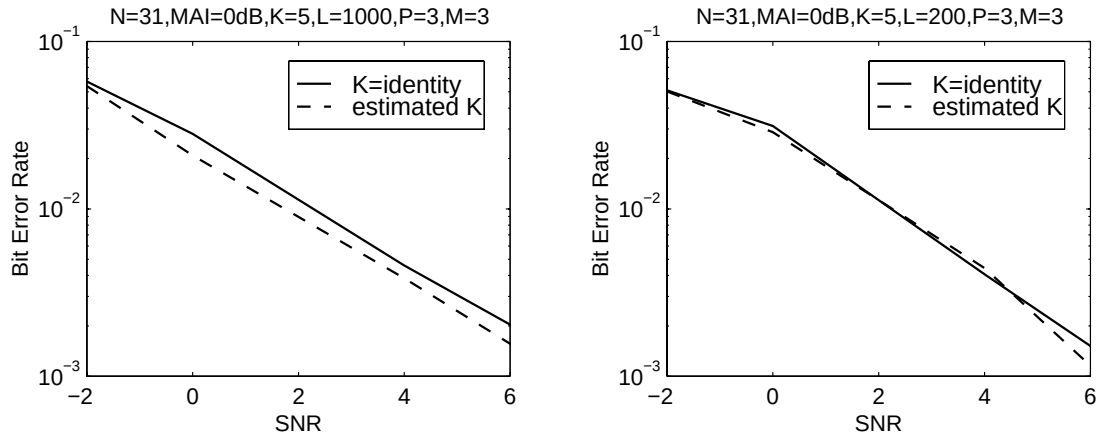
4.5.3 White noise assumption

As mentioned in Chapter 3, the additive noise is assumed to have some correlation across the array, with a component $\eta(t)$ that is completely correlated across the array and $\eta^{(m)}(t)$ is the component that is independent from sensor to sensor.

The maximum likelihood algorithm developed in this chapter, provides a mechanism for estimating the covariance $\widehat{\mathbf{K}}$ of this non-white noise. However, as discussed in Section 4.4, estimation of this covariance is one of the most computationally complex steps in the algorithm. It was suggested that to reduce complexity, $\widehat{\mathbf{K}}$ may be assumed to be identity. This implies that the algorithm will assume that the noise is additive white Gaussian, even though in reality, it will have some correlation across the array.

In this section, we present a simulation study (Figure 4.5), which shows the performance loss that will result from this assumption. The parameter β which defines the extent of the correlation across the array was specified to be quite high ($\beta = 0.9999$). A high value of β, that is a high correlation of the noise across the array, will make the noise non-white. Under this circumstance, assuming the noise to be white may lead to degradation in performance. However Figure 4.5 shows that the performance loss due to the whiteness assumption is very small. Also, the performance loss is

evident only if a large window size is used ($L = 1000$ in Figure 4.5(a)), so that the noise covariance can be estimated with greater accuracy.



(a) (b)

Figure 4.5 : Multipath, multiple sensors case: BER from a matched filter receiver, using estimated noise covariance and noise covariance equal to identity. $M = 3$, $\beta = 0.9999$.

4.5.4 Performance in multiple sensors and multipath case

In this section, we focus on detailed evaluation of the channel estimation and linear detection strategy developed in this chapter. Multiple propagation paths are considered and both the detection and channel estimation steps exploit the benefits of multiple sensors.

Figure 4.6 shows the performance of the decorrelating detector using multiple sensors and compares it to the conventional or the matched filter detector, as the different system parameters are varied. For both the decorrelating detector and the matched filter detector, channel parameter estimates from the proposed maximum likelihood algorithm under the same system configuration have been used.

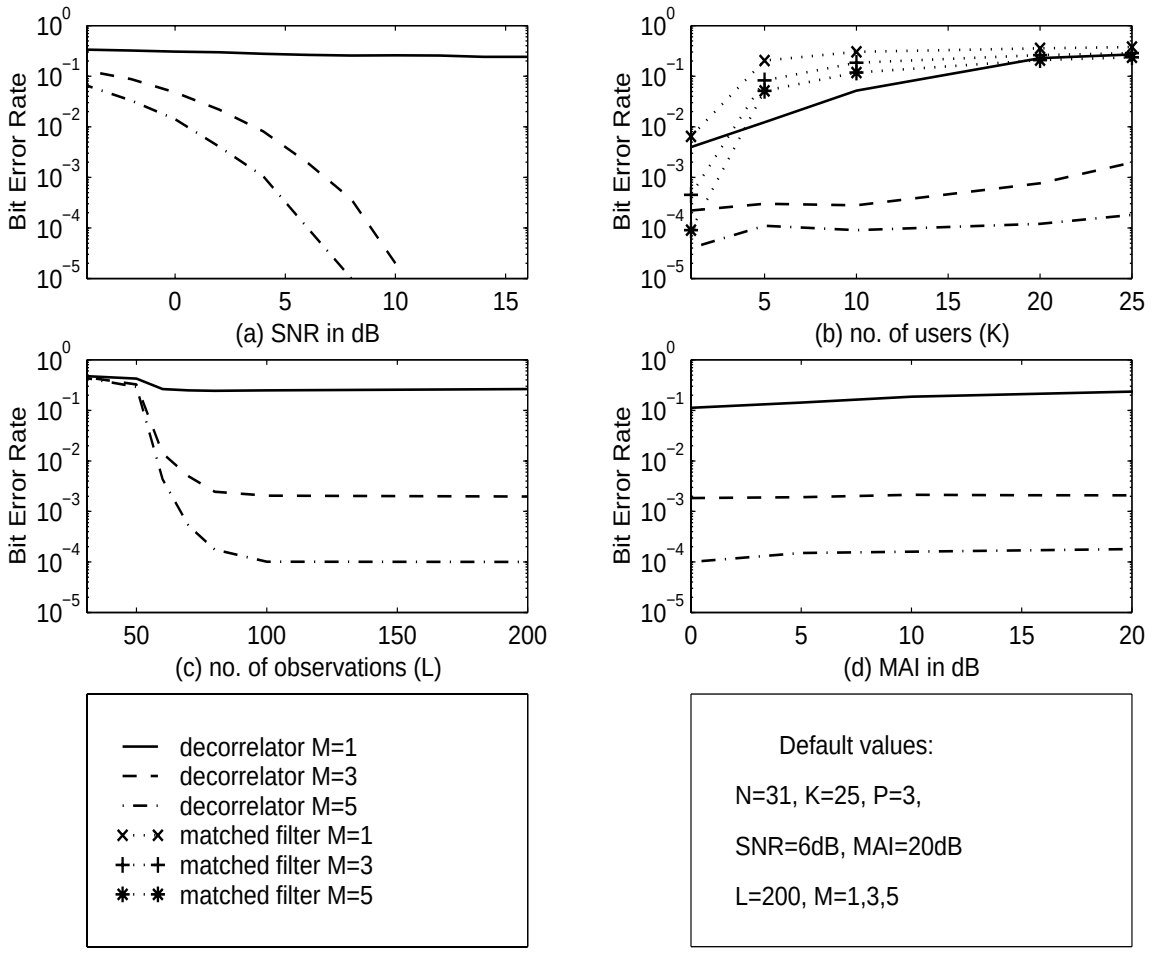


Figure 4.6 : Multipath, multiple sensors case: BER from a matched filter receiver and a *single-shot* decorrelating detector using estimated composite channel impulse response ($\hat{\mathbf{z}_k}$) and multiple sensors (M) with $\beta = 0.7$.

In Figure 4.6, the default set of system parameters are such that, the system is heavily loaded (25 users with 3 paths per user) and the interference and noise is quite high (multiple access interference is 20dB and SNR is 6dB). Figure 4.6(a), with SNR in the x-axis, shows that for the heavily loaded situation, the single-shot decorrelating detector performs quite poorly with only one sensor. However, by increasing the number of sensors, the performance improves dramatically to achieve acceptable bit-error-rates at relatively low SNRs.

It has been shown [31] that the performance of the decorrelating detector, with one sensor, drops significantly as the number of users is increased (Figure 4.6(b)), with no. of users in the x-axis). Geometrically speaking, since the decorrelating detector attempts to orthogonalize the users, with more than $N/2$ users in the system, they cannot be orthogonalized all at once and hence the performance degradation. Again, the use of even one extra sensor significantly improves the performance in the presence of a large number of users. The same plot shows that the matched filter detector performs worse than the decorrelator because of the high multiple access interference. Even then, the matched filter detector performs better as the number of senors are increased, especially when the number of users and hence multiple access interference is small.

Figure 4.6(c), with L in the x-axis, shows that a relatively short preamble (less than 100 bits) can be used to achieve optimum performance. It may be noted that for the proposed algorithm to work, at least MN observations are required. This is because, for $L < MN$, the covariance matrix, which has dimension $MN \times MN$, is rank deficient. This observation is substantiated by this plot. In this experiment, the noise is assumed to be white. The improved performance at very high values of L when the noise is not assumed to be white has been shown in Figure 4.5.

The last plot Figure 4.6(d) demonstrates the near-far resistant capability of the algorithm, which improves with increasing the number of sensors. It may be noted that the slight increase in BER at high MAI is due to the use of the *single-shot* decorrelator.

The experiment in this section demonstrates the performance of the ML algorithm for the multiuser, multipath, multiple sensors case, which is one of the primary contributions of this thesis. As expected, the use of spatial diversity offers dramatic

improvement in performance for a wide range of system parameters.

4.5.5 Performance in the presence of fading

Figure 4.7 shows the performance of the decorrelator directly using the estimated channel impulse response vector from the ML algorithm, in a fading situation. The fading statistics during the preamble as well as the transmission of data bits that are detected remains the same. The bit error rate statistics are collected over 5000 bits, after the preamble has been transmitted. Figure 4.7(a) shows the gain in performance in this time varying situation when multiple sensors are used. Figure 4.7(b) illustrates an interesting behavior in the time varying situation. The figure shows the BER variation with the size of the preamble for the ML channel estimation. Unlike the time invariant case, best performance is achieved at a certain preamble size which is a function of the system SNR. There is a degradation in performance for preamble sizes smaller as well as higher than the best size.

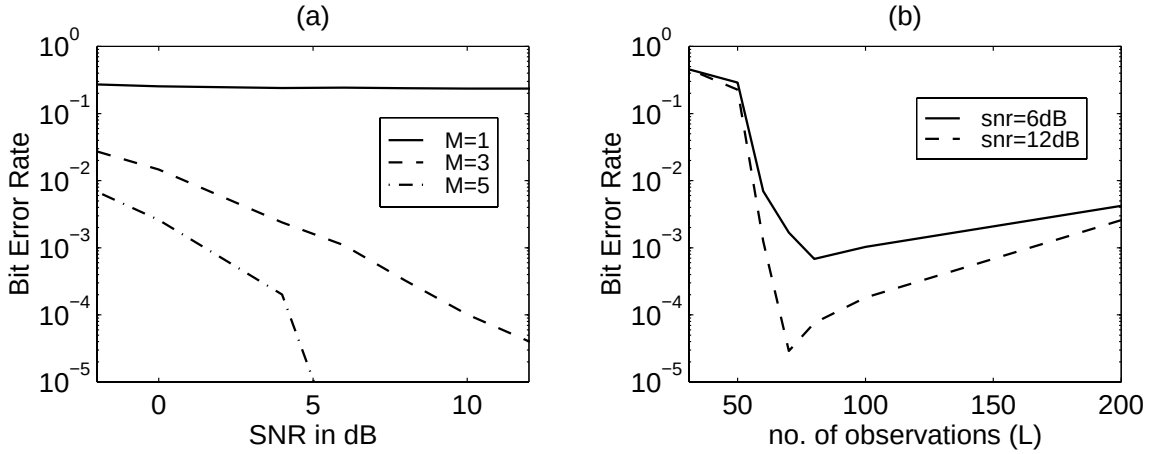


Figure 4.7 : Multipath, multi-sensor case: BER from single shot decorrelator in the presence of fading. System parameters: 1800 MHz carrier frequency, mobile speed of 50mph, and a bit rate of 100K bits per second. Default: $N = 31, K = 25, P = 3, MAI = 20dB, L = 100, M = 3$.

This can be intuitively explained by taking note of the fact that the channel attenuation and hence the channel impulse response is continuously varying. So on one hand, the preamble needs to be as small as possible such that the variation of the channel within the preamble window is small. On the other hand, due to the presence of the background noise, a large enough window is required to 'average' out the noise. This is also the reason for a larger optimum preamble size for a lower SNR situation as shown in Figure 4.7(b).

4.5.6 Performance of joint channel estimation and detection

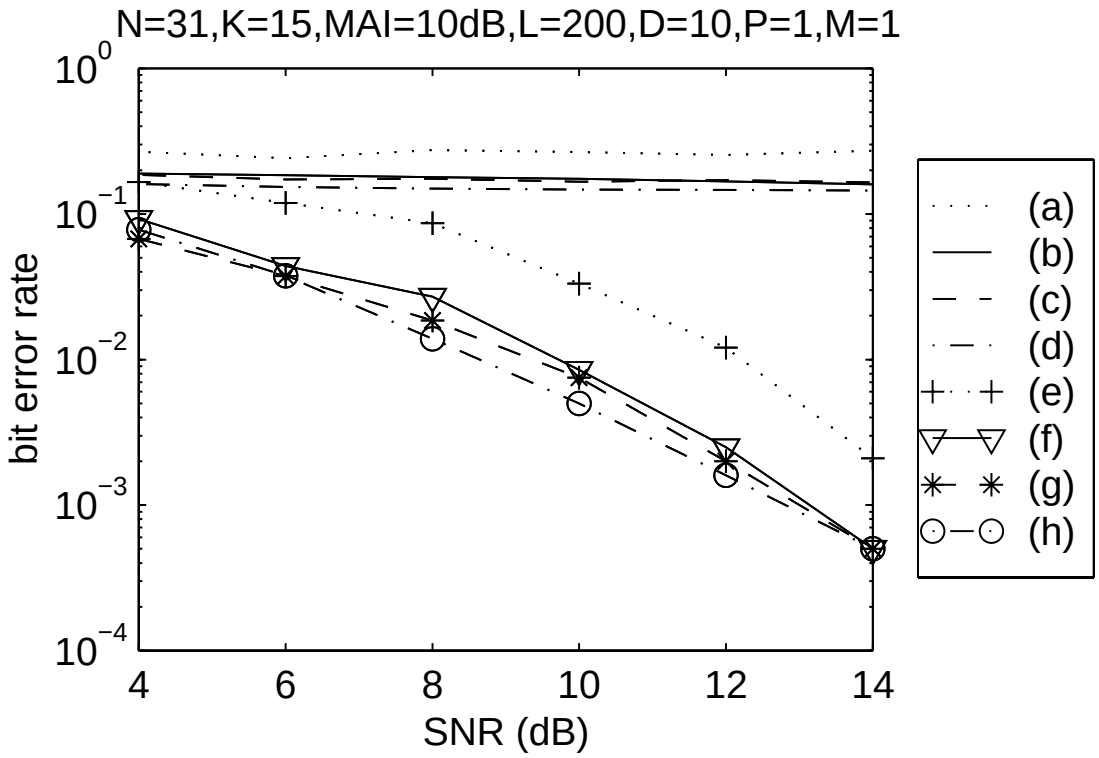


Figure 4.8 : Single path, single sensor case: BER from multishot matched filter and decorrelator for (a) matched filter with extracted parameters (b) matched filter with $\mathbf{z}_k$ estimation (c) matched filter with only $\widehat{\mathcal{Y}}$ estimation (d) matched filter with true parameters (e) decorrelator with extracted parameters (f) decorrelator with $\mathbf{z}_k$ estimation (g) decorrelator with only $\widehat{\mathcal{Y}}$ estimation (h) decorrelator with true parameters.

Figure 4.8 shows the performance of the multi-shot decorrelator detector and matched filter detector with various versions of the channel estimation process. The figure shows that the multi-shot detector performs much better than the single-shot case (Figure 4.6), especially at high SNRs. Also the performance improves as we go from extracting individual parameters to estimating $\mathbf{z}_k$ to estimating only $\widehat{\mathcal{Y}}$.

In this section we demonstrated the gains from using the effective spreading code in multiuser detection. This approach to joint channel estimation and detection has been a significant contribution of this thesis.

4.6 Fixed-point implementation

In this section, we study the fixed-point error behavior of a particular case of the proposed maximum likelihood algorithm when applied to the downlink, that is, the receiver in the handset. When designing the handset receiver, the necessity for reduced hardware size, power, and cost motivates the use of fixed-point arithmetic hardware.

The mobile handset receives a superposition of the signals for all the users transmitted by the base station. As the base station uses a common transmit clock for the signals of all the users, these signals are synchronous with respect to each other. However, the channel will introduce an arbitrary delay and arbitrary attenuation factor into this composite signal. This delay and attenuation factor must be estimated by the receiver for efficient detection of the user's bits. In this section, for the downlink, the signals for all the users are assumed to have the same power. However, in practice, this may not always be true as the base station can deliberately transmit the signals of the various users with different power levels in certain scenarios.

Algorithms used in the handset receiver should be essentially 'blind'. That is, the

algorithms should assume that the receiver has knowledge of its own spreading code only, and not of any of the other users. This requirement is necessary not only to simplify the computation but also as one method to ensure security.

Fixed-point hardware [26] has several advantages, as mentioned above. However, due to the fixed wordlength of the hardware, a fixed-point implementation will also incur errors [67] due to quantization and overflow. In this section, we evaluate the error pattern of a blind version [20] of the maximum likelihood channel estimation algorithm, when implemented on fixed-point hardware.

The proposed maximum likelihood algorithm is adapted to the 'blind' assumption, by reducing the number of users to only 1 ($K = 1$ in the algorithm). The rest of the users are modeled along with the additive noise as colored Gaussian noise [20], whose covariance is estimated as $\hat{\mathbf{K}}$. It should be noted that the assumption of white noise evaluated in Section 4.5.3 is no longer applicable in this case. Before going into the fixed-point behavior of the algorithm, we compare the complexity of the algorithm to the conventional correlator for the single-sensor, single path case.

We will use the parameters shown in Table 4.1 to estimate the operation counts. The operation counts are listed in Table 4.2 and are derived from a close analysis of the various steps in the channel estimation algorithms.

The estimate in Table 4.2 refers to the number of operations required to get each user's channel estimate. In addition, for the maximum likelihood technique, one would have to estimate the covariance of the colored noise whenever the interference characteristics change. For example, this calculation, may be done once for every frame, with the assumption that the interference structure remains constant for the frame duration. This would also be consistent with the typical design in which the preamble is sent at the beginning of each frame being transmitted. This extra com-

Table 4.1 : Notation for system parameters

Symbol	Definition
K	No. of users
$N(=T/T_c)$	Spreading code size
L	Window size for channel estimation algorithms
fT_c	Granularity of sliding correlator

Table 4.2 : Operation count of downlink channel estimation algorithms

Processing	Add/Sub	Mult	Div	Sqrt
Maximum Likelihood	$2NL + 10N + N^2$	$2NL + 10N + N^2$		1
Sliding Correlator	N^2fL	N^2fL		

putation to estimate the colored noise covariance will require $N^2L + 4L + 2N^2 + 2N^3$ operations. However, without this extra computation, the maximum likelihood algorithm requires less computation than the correlator. This complexity makes the ML algorithm an attractive alternative to the correlator for next generation systems, in light of its better performance as shown below. Also, the complexity can be further reduced if the idea of joint channel estimation and detection, discussed in Section 4.3, is also applied here.

Next, we evaluate the error due to the fixed-point implementation of the 'blind' version of the algorithm. Any fixed-point number can be represented with a fixed l_i bits for the integer part and l bits for the fractional part (Figure 4.9). The dynamic

range of the problem determines l_i. For our problem, appropriate normalization of the data can eliminate the need for the integer part, l_i. However l is determined by the precision requirement of the algorithm and requires more careful analysis. In this section, we estimate l through extensive fixed-point simulations of the algorithms. The fixed-point simulation was performed using a simulator that has been written in C++, using object-oriented programming techniques.

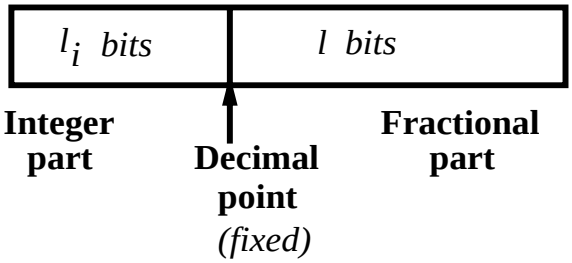


Figure 4.9 : Fixed-point representation

Figure 4.10 shows the performance of the channel estimation algorithms in terms of probability of acquisition (defined in Section 4.3) as the fixed-point wordlength is varied. We conclude that the algorithm requires 16 bits wordlength to reach its performance limit, for the chosen set of system parameters. This result is very encouraging as it implies that this algorithm can be implemented in the handset using state-of-the-art low power DSPs. One such low power fixed-point DSP is the Texas Instruments TMS320C54x processor [68] with 16 bit multiplier and 40-bit adders and accumulators.

4.7 Summary

We have developed a maximum likelihood algorithm for multipath channel estimation for a set of transmitting users in the reverse link of a wireless CDMA communication

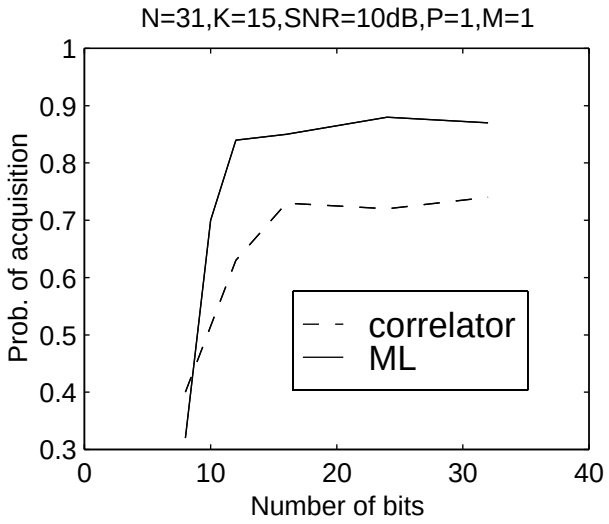


Figure 4.10 : Fixed-point performance of channel estimation algorithms for downlink. $(N = 31, K = 15, SNR = 10dB, P = 1, M = 1)$

system, when an antenna array is used at the base station receiver. The algorithm elegantly decomposes the multiuser problem into a series of single-user ones and estimates a composite channel impulse response encapsulating all the channel parameters. This estimated channel impulse response was then directly used in the detection process.

The additive noise in the system is assumed to be zero-mean, Gaussian but no assumption is made on its covariance, which is estimated within the algorithm. Our simulations verify that the algorithm is near-far resistant. Also, the estimators are not dimensionally limited; making it ideal for acquisition of a large number of users. Furthermore, the preamble required is not prohibitively large. Our simulations show that considerable performance benefits are obtained by using multiple sensors at the receiver, in the presence of multipath effects and multiple access interference. We have also shown that it is possible and indeed beneficial to use the estimated received

effective spreading code directly in detection, both in terms of performance as well as reduced computation load. The development of the maximum likelihood algorithm for the multipath, multiple sensors case and the use of the effective spreading code to obtain dramatic gains in terms of performance as well as reduced computation have been one of the major contributions of this thesis.

We have evaluated the computational complexity of various steps in the algorithm for both uplink and downlink. Table 4.3 summarizes the total work for all the users in the acquisition phase while the preamble is transmitted, and the time frame in which it needs to be performed. The correlation matrices can be calculated by processing each observation vector as it comes in. Hence Table 4.3 shows the work required per bit of preamble, in order to estimate the correlation matrices. The next three steps for acquisition - estimation of $\widehat{\mathcal{Y}}$, estimation of $\mathbf{z}_k$, and extraction of delays and attenuation factors from $\mathbf{z}_k$ - can be performed only after the entire preamble has been transmitted. We have shown earlier in this chapter, that estimation of the colored noise covariance is not necessary for the uplink. For the downlink, the colored noise estimation can be done only once per frame, assuming that the noise statistics do not change over the time period corresponding to one frame.

The maximum likelihood algorithm can be extended to tracking, in a decision directed mode. In this mode, a block of detected data bits are fed back into the system in order to update the channel estimates. In this case, estimation of the correlation matrices would require the same work per bit as shown in Table 4.3. For the next 3 steps, the work that has been shown in Table 4.3 to be done per preamble, will have to be done after collecting the same number of data bits as the size of the preamble. In addition, the inversion of the data correlation matrix will have to be performed in the case of tracking. This will require an additional work of complexity

Table 4.3 : Complexity of the maximum likelihood algorithm

Processing	Frequency of execution	Uplink	Downlink
Estimate correlation matrices	per bit of preamble	$O(MNK)$, $O(K^2)$	$O(MN)$
Estimate $\widehat{\mathcal{Y}}$	once per preamble	$O(MNK^2)$	$O(MN)$
Estimate $\mathbf{z}_k$	once per preamble	$O(M^2N^2K)$	$O(M^2N^2)$
Extract parameters	once per preamble	$O(MNK)$	$O(MN)$
Estimate colored noise covariance	once per frame	not required	$O(M^2N^2L)$+$O(M^3N^3)$

$O(K^2)$ per bit. However the noise covariance estimate in the downlink need not be updated again for the duration of the entire frame.

Lastly, we have evaluated the fixed-point wordlength requirement of the algorithm, when adapted to the 'blind' assumption of the downlink. We concluded that the fixed-point error behavior will allow implementation of the algorithm on a 16-bit DSP processor for future generation mobile handsets.

In the next chapter, we will investigate the application of another technique: subspace-based estimation, to the problem of CDMA channel estimation. This technique searches for the signal vector (that is, the effective spreading code) that minimises the projection of the signal vector onto the estimated noise subspace of the received vectors. This algorithm will not require a preamble and is essentially 'blind'. This makes the algorithm more suitable for a scenario when the spreading codes of the interfering users are not known at the receiver. However it is computationally complex and is dimension-limited in terms of the number of users it can handle [31].

Our contribution in the next chapter is the extension of the basic subspace-based acquisition algorithm [17] to the time varying scenario and development of techniques to handle the computational load of the algorithm.

4.8 Appendix

Derivation of (4.5):

The log-likelihood function (4.3) is:

$$\Lambda = -\ln|\mathbf{K}| - \ \mathrm{tr}\left\{\frac{1}{L}\sum_{i=1}^{L}(\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)^H\mathbf{K}^{-1}(\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)\right\}$$

$$= -\ln|\mathbf{K}| - \ \mathrm{tr}\left\{\mathbf{K}^{-1}\frac{1}{L}\sum_{i=1}^{L}(\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)(\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)^H\right\}. \tag{4.21}$$

The above equality follows from the following property of the trace: $\mathrm{tr}\{\mathbf{ABC}\} = \mathrm{tr}\{\mathbf{CAB}\} = \mathrm{tr}\{\mathbf{BCA}\}$.

As the first step, maximization of the log-likelihood function is to be carried out with respect to $\mathbf{K}$. We use the following fact [47] to find the maximum with respect to $\mathbf{K}$:

Fact: If $\mathbf{B}$ is a positive definite Hermitian matrix, then

$f(\mathbf{K}) = -\ln|\mathbf{K}| - \mathrm{tr}\{\mathbf{K}^{-1}\mathbf{B}\}$ is maximised by $\widehat{\mathbf{K}} = \mathbf{B}$.

Hence we have,

$$\widehat{\mathbf{K}}(\mathcal{U}, \mathbf{Z}) = \frac{1}{L}\sum_{i=1}^{L}(\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)(\mathbf{r}_i - \mathcal{U}\mathbf{Z}\mathbf{b}_i)^H. \tag{4.22}$$

Substituting this value of $\widehat{\mathbf{K}}$ into (4.21), we obtain the cost function (4.5).

Derivation of (4.6) and (4.7):

In terms of the sample correlation matrices, $\widehat{\mathbf{K}}(\mathcal{U}, \mathbf{Z})$ or $\widehat{\mathbf{K}}(\mathcal{Y})$ from (4.4) can be written as

$$\widehat{\mathbf{K}}(\mathcal{Y}) = \widehat{\mathbf{R}}_{rr} - \mathcal{Y}\widehat{\mathbf{R}}_{rb}^H - \widehat{\mathbf{R}}_{rb}\mathcal{Y}^H + \mathcal{Y}\widehat{\mathbf{R}}_{bb}\mathcal{Y}^H. \tag{4.23}$$

Minimizing cost function $\mathcal{L}$ in (4.5), $\widehat{\mathcal{Y}}$, the unconstrained ML estimate of $\mathcal{Y}$ is

$$\widehat{\mathcal{Y}} = \arg\min_{y} |\widehat{\mathbf{K}}(\mathcal{Y})| = \arg\min_{y} |\widehat{\mathbf{R}}_{rr} - \mathcal{Y}\widehat{\mathbf{R}}_{rb}^{H} - \widehat{\mathbf{R}}_{rb}\mathcal{Y}^{H} + \mathcal{Y}\widehat{\mathbf{R}}_{bb}\mathcal{Y}^{H}|. \tag{4.24}$$

We use the following lemma [31], stated without proof, to obtain an expression for $\widehat{\mathcal{Y}}$:

Lemma 1: If the matrices $\mathbf{P}$, $\mathbf{C} \in \mathbb{C}^{A\times B}$, and $\mathbf{R} \in \mathbb{C}^{A\times A}$, where A and B are some integers, then for any positive definite Hermitian matrix $Q \in \mathbb{C}^{B\times B}$,

$$\arg\min_{\mathbf{C}} \left|\mathbf{R} - \mathbf{P}\mathbf{C}^{H} - \mathbf{C}\mathbf{P}^{H} + \mathbf{C}\mathbf{Q}\mathbf{C}^{H}\right| = \mathbf{P}\mathbf{Q}^{-1}. \tag{4.25}$$

Applying this Lemma to (4.24), $\widehat{\mathcal{Y}}$ can be expressed as

$$\widehat{\mathcal{Y}} = \widehat{\mathbf{R}}_{rb}\widehat{\mathbf{R}}_{bb}^{-1}. \tag{4.26}$$

Substituting back into the equation for $\widehat{\mathbf{K}}$,

$$\widehat{\mathbf{K}}(\widehat{\mathcal{Y}}) = \widehat{\mathbf{R}}_{rr} - \widehat{\mathcal{Y}}\widehat{\mathbf{R}}_{rb}^{H}. \tag{4.27}$$

Derivation of (4.9):

Using expressions for $\widehat{\mathcal{Y}}$ and $\widehat{\mathbf{K}}$ in (4.23), our cost function (4.5) now becomes

$$\begin{aligned}
\mathcal{L} &= |\widehat{\mathbf{K}}(\mathcal{Y})| \\
&= |\widehat{\mathbf{R}}_{rr} - \mathcal{Y}\widehat{\mathbf{R}}_{rb}^{H} - \widehat{\mathbf{R}}_{rb}\mathcal{Y}^{H} + \mathcal{Y}\widehat{\mathbf{R}}_{bb}\mathcal{Y}^{H}| \\
&= \left|\widehat{\mathbf{K}}\right| \cdot \left|\mathbf{I} + \widehat{\mathbf{K}}^{-1}(\mathcal{Y} - \widehat{\mathcal{Y}})\widehat{\mathbf{R}}_{bb}(\mathcal{Y} - \widehat{\mathcal{Y}})^{H}\right| .
\end{aligned} \tag{4.28}$$

We note that minimizing $\left|\mathbf{I} + \widehat{\mathbf{K}}^{-1}(\mathcal{Y} - \widehat{\mathcal{Y}})\widehat{\mathbf{R}}_{bb}(\mathcal{Y} - \widehat{\mathcal{Y}})^{H}\right|$ is equivalent to minimizing $\ln\left|\mathbf{I} + \widehat{\mathbf{K}}^{-1}(\mathcal{Y} - \widehat{\mathcal{Y}})\widehat{\mathbf{R}}_{bb}(\mathcal{Y} - \widehat{\mathcal{Y}})^{H}\right|$. Using properties of eigenvalues and norms of matrices, it can be shown [31] that:

$$\min_{\mathbf{Z}} \ln\left|\mathbf{I} + \widehat{\mathbf{K}}^{-1}(\mathcal{Y} - \widehat{\mathcal{Y}})\widehat{\mathbf{R}}_{bb}(\mathcal{Y} - \widehat{\mathcal{Y}})^{H}\right| \xrightarrow{\equiv} \min_{\mathbf{Z}} \mathrm{tr}\{\widehat{\mathbf{R}}_{bb}(\mathcal{Y} - \widehat{\mathcal{Y}})^{H}\widehat{\mathbf{K}}^{-1}(\mathcal{Y} - \widehat{\mathcal{Y}})\}, \tag{4.29}$$

where the symbol $\stackrel{\equiv}{\longrightarrow}$ denotes "is asymptotically equivalent to" (the reader is reminded of the scalar case where $\ln(1+x) \approx x$ for small x). This greatly simplifies the problem since it replaces the nonlinear functional $|\cdot|$ by a linear one $\mathrm{tr}\{\cdot\}$. The problem now reduces to

$$\widehat{\mathbf{Z}} = \arg\min_{\mathbf{Z}} \mathrm{tr}\left\{\widehat{\mathbf{R}}_{bb}(\mathcal{Y}-\widehat{\mathcal{Y}})^H \widehat{\mathbf{K}}^{-1}(\mathcal{Y}-\widehat{\mathcal{Y}})\right\}. \tag{4.30}$$

Since the received signals of the different users are "uncorrelated", $\widehat{\mathbf{R}}_{bb}$ is diagonal for large L. This diagonal form helps to separate the estimation process of each user and yields the expression (4.9) for the estimate of the k^{th} user's channel impulse response.

Extraction of parameters in single sensor case:

In the single sensor case, the estimated channel impulse response vector $\widehat{\mathbf{z}}_k$ is the same as component $\widehat{\mathbf{h}}_k$ only. We perform a least squares fit of $\widehat{\mathbf{h}}_k$ to our parametric channel model (3.13) to extract the strongest P paths. For each pair of adjacent coefficients of $\widehat{\mathbf{h}}_k$, we obtain *local* values of amplitudes and delays, from the following optimization:

$$[\mathrm{w}_q, \gamma_q] = \arg \min_{w\in\mathbb{C},\gamma\in[0.1)} ||\widehat{h}_{k,q} - (1-\gamma)w||^2 + ||\widehat{h}_{k,q+1} - \gamma w||^2. \tag{4.31}$$

We then search for the *global* maxima to obtain the strongest path :

$$\begin{aligned} \hat{q} &= \arg \max_{q\in\{0,\cdots,N-1\}} |w_q| \\ \hat{\tau} = (\hat{q}+\gamma_{\hat{q}})T_c, &\qquad \hat{w} = w_{\hat{q}}. \end{aligned} \tag{4.32}$$

The estimated path is subtracted from $\widehat{\mathbf{h}}_k$ and the process is repeated to find the next strongest path, until either a specified number of paths have been identified or $|\hat{w}|$ falls below some predetermined significant level.

Chapter 5

Subspace-based Tracking of Channel Parameters

Subspace-based techniques [16,17] are an attractive alternative to the maximum likelihood technique for solving the channel estimation problem, as they provide a method for decomposing a multidimensional parameter search into a series of one dimensional optimization problems. Such an algorithm exploits the signal subspace spanned by certain observation vectors, in order to estimate the unknown channel parameters. Subspace-based algorithms for channel estimation have been shown to be near-far resistant and effective in the presence of multiple propagation paths. In addition they do not require any training sequences or the knowledge of the spreading codes of the interfering users. Hence, the subspace based schemes are suitable for environments where transmitting a preamble is not feasible or in the downlink where the mobile receiver does not have the knowledge of the spreading codes of the interfering users.

The performance of the subspace-based channel estimation algorithms depends, to a large extent, on the speed and accuracy of the subspace estimation process, especially when the parameter (and hence the signal subspace) is time varying. The tool typically used to estimate the signal and noise subspaces of the received signal vectors is the Singular Value Decomposition (SVD) of an observation matrix formed from received data vectors. However the SVD has high computational complexity ($O(N^3)$ for an $N \times N$ matrix), involving orthogonal rotations that require costly operations such as divisions and square-roots. The subspace estimation is a crucial step in this algorithm and it is necessary to update the estimate in response to any

time variations in the channel.

In this chapter, we explore several issues related to performance as well as computational complexity of a subspace-based synchronization technique, when tracking channel parameters in a CDMA system in the presence of multipath fading. We will first explore tracking issues when the multipath channel delays remain constant over several bits while the channel attenuation varies due to fading. This analysis will help us determine the effect of tracking mechanisms on parameters which vary at different rates. We will also analyze situations when the channel delays as well as the attenuation factors vary simultaneously. In this case the delay is assumed to change in the form of discrete jumps, which would occur if any one of the propagation paths suddenly disappears due to a change in the suroundings. On the other hand the channel attenuation varies continuously due to multipath fading.

Recently, several multiuser detection schemes, based on signal subspace estimation, have also been proposed [59, 60, 69]. Under this approach, it has been shown that both the decorrelating detector and the minimum-mean-square-error (MMSE) detector can be obtained blindly, that is, only with prior knowledge of the spreading code of the user of interest. However, the performance of these schemes, especially in time-varying situations, also depend largely on the accuracy and complexity of the subspace estimation algorithm. Hence, the study of subspace tracking algorithms and their application to CDMA systems, are also relevant to these subspace-based blind detection schemes.

5.1 Channel estimation as a subspace-based algorithm

In this chapter, we will only consider the single sensor scenario. We will use the system model presented in Chapter 3, with the number of sensors $M = 1$. Hence

when expressing the received signal we will drop the superscript (m) from (3.7) for convenience.

$$\mathbf{r}_i = \mathbf{A}\mathbf{b}_i + \boldsymbol{\nu}_i, \qquad i = 1, 2, \cdots . \tag{5.1}$$

We may recall that the columns of matrix A are known functions of the users' spreading code and the unknown channel parameters (τ and w). The unknown parameters, τ, are the relative time delays of each path of each user, and w are the complex valued attenuation factors of each path of each user.

Given observation vectors $\mathbf{r}_i$, for all i, the first step is the estimation of the noise subspace. The subspace spanned by the columns of $\mathbf{A}$ is called the *signal subspace* and the orthogonal subspace is termed *noise subspace*. This basis may be obtained by computing an Eigenvalue Decomposition (EVD) of the correlation matrix, R, of the observation vectors:

$$R = \mathcal{E}[\mathbf{r}_i \mathbf{r}_i'], \tag{5.2}$$

where $\mathcal{E}[.]$ denotes the expected value and $'$ denotes the conjugate transpose operator. In practice, the correlation matrix is estimated as a time average of $\mathbf{r}_i \mathbf{r}_i'$ over several observations. This average may be performed in a variety of ways, in order to take into account the time variations of the channel parameters. Hence several formulations of the problem are possible and in effect result in different algorithms, which in turn may be implemented in a variety of different ways. One of these formulations is the *sliding window formulation*, where the last W successive observations are used at each time step i, to estimate the correlation matrix as:

$$\hat{R}_i = \frac{1}{W} \sum_{j=i-W+1}^{i} \mathbf{r}_j \mathbf{r}_j' = \frac{1}{W} Y_i' Y_i. \tag{5.3}$$

The matrix Y_i is an observation matrix, which is defined as follows for the sliding window formulation:

$$Y_i = \begin{bmatrix} \mathbf{r}'_{i-W+1} \\ \mathbf{r}'_{i-W+2} \\ \cdots \\ \mathbf{r}'_i \end{bmatrix}. \tag{5.4}$$

Another formulation is the *exponential window formulation* described as :

$$Y_i = \begin{bmatrix} \beta^{i-1}\mathbf{r}'_0 \\ \vdots \\ \beta\mathbf{r}'_{i-1} \\ \mathbf{r}'_i \end{bmatrix} = \begin{bmatrix} \beta Y_{i-1} \\ \mathbf{r}'_i \end{bmatrix}, \tag{5.5}$$

where $0 < \beta \leq 1$ is a "forgetting factor". (The variable β used here to denote the "forgetting factor" should not be confused with the variable β used in the previous chapters to denote noise correlation across the antenna array at the receiver.) The choice of the windowing scheme and the respective window parameters depends on the rate of variation of the signal and noise subspace and the time available to estimate the subspace.

In practice, the EVD of the estimated correlation matrix, $\hat{R}_t$, is computed using the SVD of the observation matrix, Y_i. This obviates the need for the matrix multiply operation required to compute the correlation matrix and helps achieve higher numerical accuracy.

The SVD of Y_i is defined as:

$$Y_i = U_i \Sigma_i V'_i, \tag{5.6}$$

where, U and V are unitary matrices and Σ is a diagonal matrix of singular values. The basis for the noise subspace, Vn, is determined as the last $N - r$ columns of the

matrix V, where r is the rank of matrix A. In this thesis, we will assume that r is known; however a variety of techniques have been proposed for estimating the model order (e.g. [70]). It may be noted that the rank of matrix A is determined by the number of users K, and the number of propagation paths for each user, P.

Once an orthogonal basis for the noise subspace, Vn, has been determined, the unknown time delays and attenuation factors are estimated by solving non-linear optimization problems which may again be posed in numerous ways. One of these formulations has been proposed in [17], based on the well-known *Multiple Signal Classification (MUSIC)* [71] algorithm. In this formulation, the different users' time delays and attenuation factors are estimated independently of each other. The formulation essentially estimates the channel impulse response (3.3) which is related to the users' time delays and attenuation factors as :

$$h_k(t) = \sum_{p=1}^{P} w_{k,p}\delta(t - \tau_{k,p}). \tag{5.7}$$

The channel impulse response for each user is estimated as that value chosen from a set of all feasible impulse responses, (as determined by our *a priori* channel model), which minimizes the ℓ_2-norm of the projection of the user's signal vectors into the estimated noise subspace.

The limitation of this method is that the number of users is limited by the size of the spreading code ($K \leq N/J$, where J is the number of signal components per user in the input vector). However this problem can be alleviated by sampling the received signal faster than the chip rate, although at the cost of increased complexity.

5.2 Tracking a time varying subspace

When the channel parameters vary with time, the performance of the subspace-based synchronization algorithm depends largely upon the ability of the subspace estimator to adapt to and track the time variations in the underlying signal or noise subspace. Studies show that the structure of the data matrix formed from the received observation vectors [72, 73] is one of the most important factors in the performance of any subspace tracker. In this section and the next, we focus on this aspect and evaluate the performance of several matrix structures.

Following the ideas presented in [74], we first analyze the tracking behavior of any subspace estimator in a time varying situation. Let the matrix Vn_i^{exact} denote the actual matrix that defines the noise subspace at time i, and the corresponding noise subspace be called $\mathcal{R}(Vn_i^{exact})$. However, at time i, any subspace tracking mechanism gives an estimate of Vn_i^{exact}, denoted by Vn_i, and the corresponding noise subspace is called $\mathcal{R}(Vn_i)$.

The function $dist(\mathcal{R}(V_1), \mathcal{R}(V_2))$ is a measure of the distance between the two subspaces specified as its arguments. The distance between two subspaces is defined as [64] :

$$dist(\mathcal{R}(V_1), \mathcal{R}(V_2)) = \|\mathcal{P}_1 - \mathcal{P}_2\|_2, \tag{5.8}$$

where $\mathcal{P}_j$, $(j = 1, 2)$, is the orthogonal projection onto the subspace $\mathcal{R}(V_j)$. Here, the columns of V_j are an orthonormal basis for the subspace $\mathcal{R}(V_j)$. Hence, it follows that $\mathcal{P}_j = V_j V_j'$.

The following quantities - tracking error (TE), and time variation (TV) are defined using the notion of distance between noise subspaces. The tracking error, TE_t, at

time step t is defined as :

$$TE_i = dist(\mathcal{R}(Vn_i^{exact}), \mathcal{R}(Vn_i)). \tag{5.9}$$

The time variation ($TV_{[i-W]\rightarrow[i]}$) in the received vectors from time step $i-W$ to time step i, is defined as :

$$TV_{[i-W]\rightarrow[i]} = dist(\mathcal{R}(Vn_{i-W}^{exact}), \mathcal{R}(Vn_i^{exact})). \tag{5.10}$$

At time step $i-W$, we have a tracking error of TE_{i-W}. If there was no change in the observation matrix over the next W steps, the tracking error would decrease by some reduction factor, RF_w, after w time steps. However, the observation matrix does change due to actual change of input data. So, we have the following inequality:

$$TE_i \leq \frac{TE_{i-W}}{RF_W} + TV_{[i-W]\rightarrow[i]}. \tag{5.11}$$

The best window size, W, for any subspace tracking mechanism is dictated by the two terms on the right hand side of (5.11). The first term will be reduced by the use of a larger window size since in the presence of noise, RF_W will be larger for larger W. On the other hand, the second term will be reduced by the use of a smaller window size, so that the variation of the incoming data over the window is small. Thus the optimum window size is determined by a tradeoff between these two terms. This analysis will help us explain the results of our experiments in the next section.

5.3 Choice of window and window parameters

In this section, we present our evaluation of the two different windowing methods on the incoming data and the respective window parameters, through simulation studies of the tracking behavior of the subspace-based algorithm in a multipath fading

environment. All the experiments described in this chapter use Gold codes of length 31 as spreading codes. The values used for multiple access interference (MAI), signal-to-noise ratio (SNR) of the additive background noise, number of users (K), number of paths for each user (P), and the fading parameters are listed against the results of each experiment.

5.3.1 Behavior of sliding window

In the presence of fading, the attenuation factors vary with time, causing variations in the subspace. In this case, the best window size is dictated both by the SNR and the localized time variation in the subspace being estimated. If the window size is too small, the noise cannot be 'averaged out' successfully, whereas if the window size is too large, the subspace estimation process cannot respond to the variation in time as it is biased by old data points. The dependence of the window size W on the SNR as well as the time variation of the subspace is again explained by (5.11). In this case, both terms in the right hand side of (5.11), influence the choice of the best window size, but in opposite ways.

In the experiments in this section, time variation is introduced through Rayleigh fading simulated according to the Jakes model [46]. The experiments have been performed with a Doppler frequency of $1.04 \times 10^{-3}/T_b$, where T_b is the bit period. This corresponds to a carrier frequency of 900 MHz, bit rate of 40 Kbps, and a relative velocity between transmitter and receiver of 50km/h. In these experiments, the attenuation factor of each path of each user varies with time due to fading. However, the time delay of each path of each user is fixed. All statistics related to tracking are collected for the first path of the weakest user, over 2000 bits.

The effect of time variation in a channel parameter on the required window size

is shown in Figure 5.1, which shows the variation in the estimate of the attenuation factor of the first path of the weakest user with window size. The error curve attains a minima at a particular window size - which depends upon the fading characteristics, number of users in the system and the SNR.

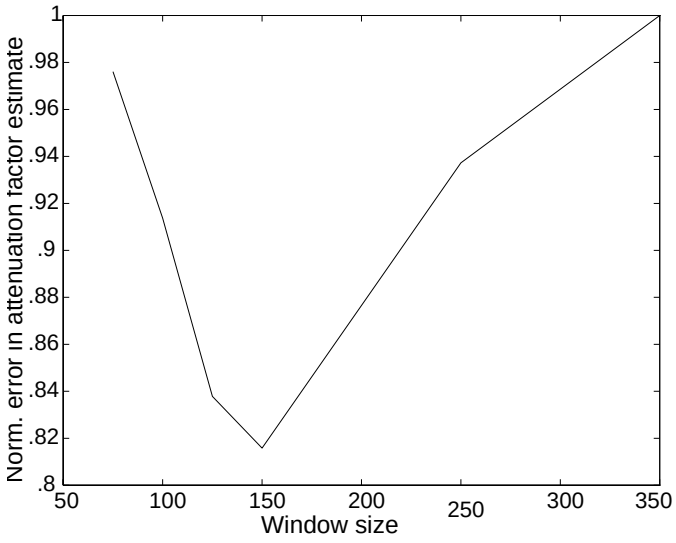


Figure 5.1 : Error in attenuation factor estimate of first path of weakest user vs. window size (sliding window), $K = 3$, $P = 2$, Rayleigh fading with Doppler frequency of $1.04 \times 10^{-3}/T_b$, time varying attenuation factor and fixed delay of each path of each user, $SNR = 15dB$, $MAI = 20dB$.

However, since the delays of each path of each user is fixed, the error in the delay estimate decreases monotonically with increasing window size (Figure 5.2). Here, successful tracking at any time step, is defined as $|\tau_k - \hat{\tau}_k| \leq T_c/2$, where τ_k is the actual delay, $\hat{\tau}_k$ is the estimate of the delay and T_c is the chip period. The probability of tracking is calculated over 2000 bits. Figure 5.2 shows that the choice of the best window size is influenced both by SNR and the number of users K in the system. The error in the delay estimate from a sliding correlator, (used in conventional CDMA systems and computationally the least expensive), has been provided for reference.

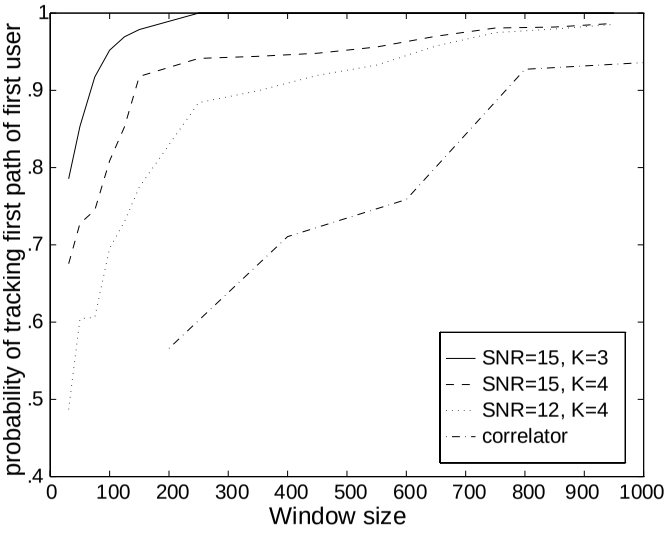


Figure 5.2 : Probability of tracking first path of weakest user vs. window size (sliding window). $P = 2$, Rayleigh fading with Doppler frequency of $1.04 \times 10^{-3}/T_b$, time varying attenuation factor and fixed delay of each path of each user, $MAI = 20dB$. The sliding correlator has been evaluated at $SNR = 15dB$ and $K = 4$. All SNR values are in dB.

However, for the sliding correlator, all users were assumed to have equal power (as would be the case with strict power control). This is because, the sliding correlator, unlike the subspace-based estimator is not near-far resistant and requires power control. The fading parameters used were the same.

These experiments show how a parameter which remains fixed over time, as well as, a time varying parameter, is effected by the window size of a sliding window. Also, the best window size for estimating two different parameters may be different depending on how fast each parameter is varying. Hence there is a tradeoff involved in the choice of the window size, based on how the inaccuracy in the estimation of a parameter effects the bit error rate of the receiver using these estimated parameters. If a particular parameter is more important to the receiver for achieving lower bit error rates, the choice of the window size would lean towards the best window size

required for the estimation of that parameter.

5.3.2 Behavior of exponential window

In this section we will demonstrate the effect of the 'forgetting factor' of an exponential window in tracking the time delay of a user. In an exponential windowing scheme, the window size W, and the forgetting factor β, are related as follows - given a "small" threshold b, the W^{th} power of β is equal to b, that is $\beta^W = b$. Here the threshold b is such that, when the weighting factor for a particular data vector is less than b, it does not make any significant contribution to the subspace estimate and can be ignored. Hence, a smaller β, corresponds to a smaller window size and vice versa. This implies that the probability of tracking should improve with increasing values of the forgetting factor β. This is shown in Figure 5.3.

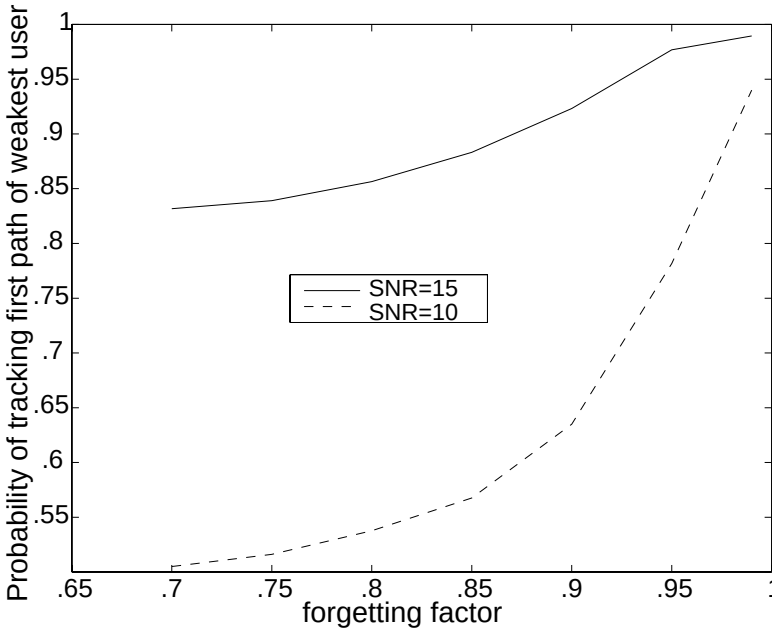


Figure 5.3 : Probability of tracking first path of weakest user vs. β (exponential window). $P = 2$, Rayleigh fading with Doppler frequency of $1.04 \times 10^{-3}/T_b$, time varying attenuation factor and fixed delay of each path of each user, $MAI = 20dB$. $SNR = 15dB$ and $10dB$, $K = 3$.

5.3.3 Comparison of windows for delay and amplitude tracking

We will first discuss some examples which illustrate the behavior of both windowing schemes in the same situation. In all the examples in this section, the following parameters were used: $P = 2$, $K = 3$, $SNR = 15dB$, and Rayleigh fading with Doppler frequency of $1.04 \times 10^{-3}/T_b$. The attenuation factor of each path of each user varies with time due to fading. However, the time delay of each path of each user is fixed, except for a discrete jump after 1000 bits. All statistics related to tracking are collected for the first path of the weakest user, over 2000 bits.

Figure 5.4 shows the performance of a sliding window with a 'large' window size. The delay which is not varying continuously is estimated perfectly. It even tracks the sudden jump in the delay at 1000 bits, within an interval of about 30 time steps. However it performs extremely poorly when tracking the continuously varying attenuation factor.

Figure 5.5 shows the performance of an exponential window with two different values of the forgetting factor, β. In both cases, the delay estimates are more noisy than that from the sliding window, (with β=0.80 doing worse than β=0.95). However, the exponential window tracks the attenuation factor better than the sliding window. The figures also show that a lower value of β enables a better tracking of a time varying parameter, even though the estimate is more noisy. The results of this experiment are summarized in Table 5.1.

The tradeoffs involved in the choice of windowing scheme may be summarized as follows: if a parameter varies continuously, an exponential window will perform better than a slidng window. Also, an exponential window with smaller forgetting factor will track larger variations in the parameter better at the cost of larger estimate variance during stationary periods. Similarly, a sliding window with smaller window

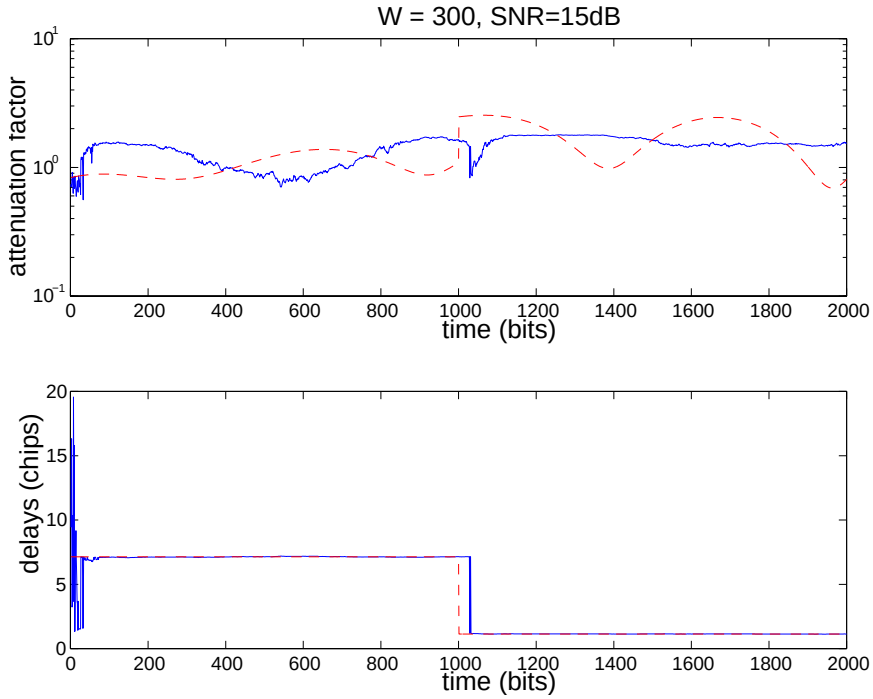


Figure 5.4 : Attenuation factor and delay tracking, using a sliding window of size of 300. The dotted line is the actual parameter and the solid line is the estimate.

size will track variations better than a larger window size. On the other hand, a sliding window with smaller window size or an exponential window with smaller forgetting factor will require less system memory. It is possible, that in order to achieve good tracking performance for two parameters (e.g. delay and attenuation factor) which are varying at different rates, one may need to execute the same subspace tracking algorithm but with different window structures.

In this section, we focused on the performance of the subspace estimator in a time varying environment. We demonstrated that the structure of the observation matrix is an important factor in the effectiveness of the subspace estimator and evaluated two of the window structures. We found that, provided the window structure and window parameter is chosen correctly, the subspace-based method performs very well

window type	parameter	P_t	E_a
sliding	$W = 300$	0.9861	1.0000
exponential	$\beta = 0.95$	0.9768	0.6850
exponential	$\beta = 0.80$	0.8564	0.6517

Table 5.1 : Comparitive performance of windowing schemes: P_t denotes probability of tracking first path of weakest user and E_a is the normalized error in estimating the attenuation factor of first path of weakest user.

while tracking fading multipath channel parameters.

However, a subspace-based method for tracking channel parameters involves executing a computationally complex ($O(N^3)$) subspace estimation algorithm (typically, the SVD) at each time interval. Given existing processors and data rates, this may seem to be an almost impossible task. Hence, it is natural to look for some approximations which will reduce the computational load, possibly at the cost of some loss in performance. In the next section, we evaluate the performance of several computation reducing approximate techniques for the SVD when tracking CDMA channel parameters.

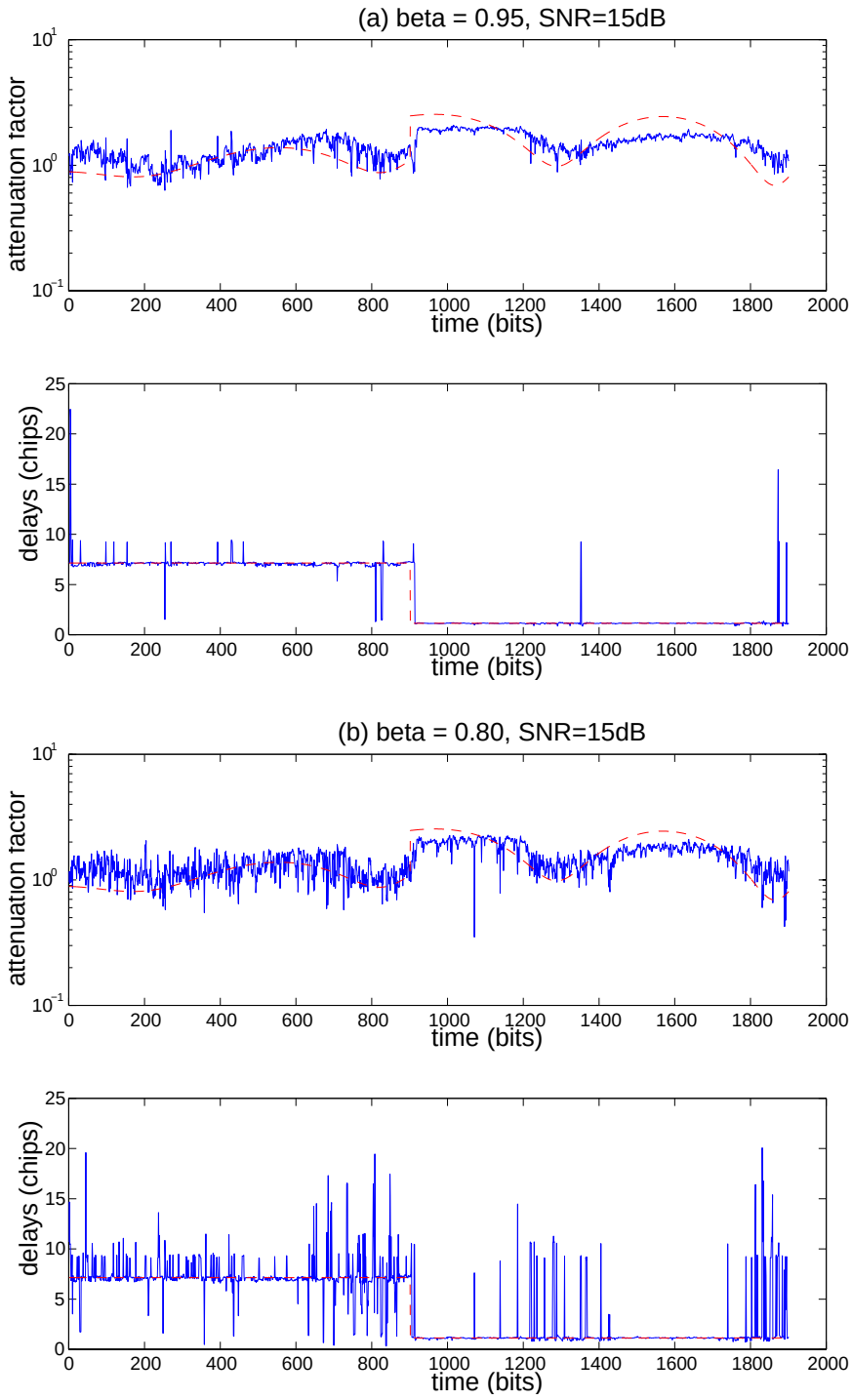


Figure 5.5 : Attenuation factor and delay tracking, using exponential windows, with (a) β=0.95 and (b) β=0.80. The dotted line is the actual parameter and the solid line is the estimate.

5.4 Approximate techniques to reduce computation

Several groups of researchers have addressed the problem of reducing the computational complexity of the subspace estimation process. One such computation reducing technique is the use of updating algorithms which efficiently use the subspace estimation at time step $i-1$ and the new data vector which arrives at time step i, to obtain the subspace estimate for time step i. Two schemes which estimate both the signal and noise subspace using updating algorithms are : Stewart's URV updating algorithm [75] and an SVD updating algorithm by Moonen et. al. [74], which updates the SVD by interlaced QR triangularizations and Jacobi rotations. Both these methods are of complexity $O(N^2)$. The algorithm proposed by Moonen et. al. was also proposed independently in [76]. A detailed treatment of such SVD and SVD updating schemes can be found in [50].

In this chapter we focus on the use of Jacobi rotation based SVD update schemes and possible approximations to these schemes to reduce the complexity further at the cost of accuracy. The reason for choosing this class of algorithms is because of the high possibility of parallelization of the algorithms [74, 77] and their efficiency in tracking the noise subspace as required by our parameter estimation algorithm of choice. Several techniques, (such as SVD updating [74,77,78], approximate orthogonal rotations [79]), have been proposed in the literature to reduce the complexity of the SVD when tracking a slowly changing subspace. We explore the use of such techniques in tracking the signal subspace in the presence of multipath fading. We also propose and develop a new technique to reduce the SVD workload, in which only a fraction of the available signal samples are used for the subspace estimation.

5.4.1 Approximate SVD updating techniques

Typically, approximate SVD updating algorithms [74–76] can be implemented in time which is at least an order of complexity less than a full SVD. These techniques suggest that one or less than one sweep of the orthogonalization procedure embedded within the SVD algorithms can be performed at each update, instead of the $O(log(n))$ sweeps required for full convergence of the complete SVD algorithm, where n is the number of columns of the matrix which is being decomposed. It has been shown, via simulations [76], for other subspace-based applications such as frequency estimation using MUSIC, that a 'good' enough estimate of the signal subspace is obtained by performing only one sweep per update, that is, one sweep per time step. In this section, we present results from our experiments to evaluate the performance of a Hestenes method based SVD updating scheme [77], for sliding window, when tracking channel parameters in the presence of fading. In the updating scheme, only one sweep of the orthogonalization procedure is performed per update.

Figure 5.6 shows the root-mean-square-error (RMSE) in the delay estimates when full convergence is achieved at each update (5 sweeps for $N = 31$) and the error when only 1 sweep is used, along with the sliding window size used. The SVD update technique used here is that developed later in this thesis [77]. Details of this technique can be found in Section 5.5. The figure shows that, when only 1 sweep is used at each update, we obtain almost as good performance as when full convergence is achieved at each update, especially at larger window sizes. This technique, reduces the computational load from $O(N^3)$ to $O(N^2)$ at a very small cost in terms of reduced accuracy.

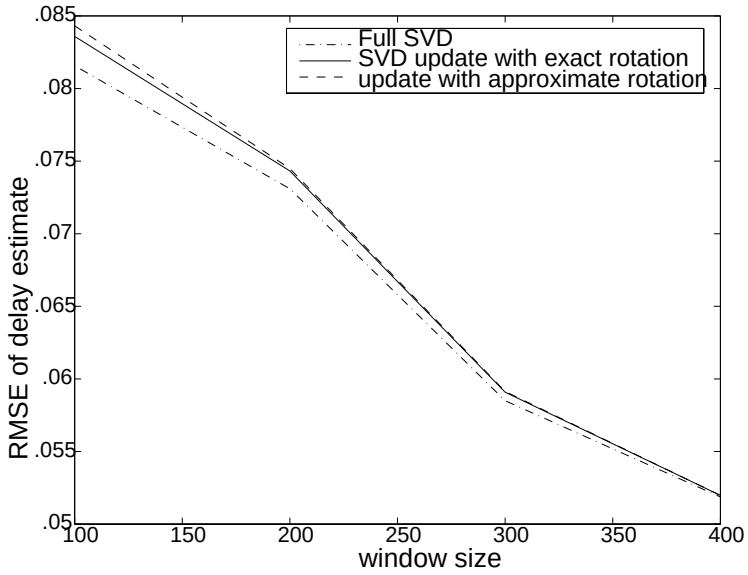


Figure 5.6 : RMSE of delay estimate when tracking first path of weakest user vs. window size (sliding window). $P = 2$, Rayleigh fading with Doppler frequency of $1.04 \times 10^{-3}/T_b$, time varying attenuation factor and fixed delay of each path of each user, $MAI = 20dB$, $SNR = 10dB$, $K = 3$, $N = 31$.

5.4.2 Approximate orthogonal rotations in the SVD

In addition to the SVD updating schemes mentioned in the previous section, several techniques have been proposed to reduce the computation required in the orthogonalization procedure embedded within the SVD algorithms. This is done by approximating the exact Jacobi rotations by less accurate but faster approximate orthogonal rotations [79]. It has been shown that in practice, these approximate rotations provide a 'good' enough estimate of the signal subspace. Figure 5.6 shows the loss in accuracy of the delay estimate by using one of the approximations mentioned in [79]. The approximation can be described as follows.

Let us suppose we are trying to calculate the SVD of matrix Y, using Hestenes method which applies Jacobi rotations to orthogonalize the columns of matrix Y.

In Hestenes method for SVD, starting with any given data matrix Y, an orthogonal matrix V is built such that YV has orthogonal columns. To construct V, we take $Y^{(0)} = Y$ and iterate :

$$Y^{(j-1)} = Y^{(j)} Q^{(j)}, \tag{5.12}$$

(where j represents a *sweep* and $Q^{(j)}$ is orthogonal), until some $Y^{(j)}$ has orthogonal columns. Typically, $O(log(n))$ sweeps are required for convergence, where n is the number of columns of matrix Y. $Q^{(j)}$ is chosen to be a product of $n(n-1)/2$ Jacobi rotations :

$$Q^{(j)} = \prod_{k=1}^{n(n-1)/2} Q_k^{(j)}. \tag{5.13}$$

Every possible pair (r, s), $1 \leq r < s \leq n$, is associated with one of the rotations $Q_k^{(j)}$. Let us denote the rotation associated with columns (r, s) by Q_{rs}.

The exact Jacobi rotation Q_{rs} is constructed by replacing 4 elements of an $n \times n$ identity matrix: (q_{jk} is the $(j,k)^{th}$ element of matrix Q_{rs}):

$$\begin{bmatrix} q_{rr} & q_{rs} \\ q_{sr} & q_{ss} \end{bmatrix} = \begin{bmatrix} -s & c \\ c & s \end{bmatrix}, \quad c = cos\phi, \quad s = sin\phi, \tag{5.14}$$

$$t_{ex} = tan\phi = \frac{sign(\sigma)}{|\sigma| + sqrt(1+\sigma^2)}, \tag{5.15}$$

$$\sigma = \frac{||Y_r||^2 - ||Y_s||^2}{Y_r' Y_s}, \tag{5.16}$$

$$c = \frac{1}{sqrt(1+t_{ex}^2)}, \qquad s = t_{ex} c. \tag{5.17}$$

where, Y_r denotes the r^{th} column of matrix Y, $||.||$ denotes vector norm, and Y_r' is the conjugate transpose of Y_r.

In the approximate rotation used in our simulations [79], $t = tan\phi$ is calculated as follows:

$$t_{ap} = tan\phi = \frac{1}{2\sigma}, \tag{5.18}$$

instead of as equation 5.15. This approximate rotation is less costly to compute but less accurate, especially for smaller values of $|\sigma|$. The decrease in computational load when the approximate rotation is used instead of the exact rotation arises from the removal of the square-root, which is an expensive operation on most computer systems. This is because the square-root operation is not one of the basic arithmetic operations on most computer systems, and is done either in software, which requires multiple processor cycles, or on special custom hardware. However, Figure 5.6 shows that this approximation can be used (with only 1 SVD sweep) with very little loss of accuracy, for our application.

5.4.3 Subspace estimate with data skipping

In this section, we form the subspace estimate, without using the data vectors corresponding to every bit observed. Only a fraction of the data vectors available are used. If the channel is slowly varying with time, this should give as good an estimate as the subspace estimate using all the data vectors, while more time would be available between updates for computation. In this way, we gain more time to perform each update.

Figure 5.7 shows the results in the presence of Rayleigh fading with a Doppler frequency of $1.04 \times 10^{-3}/T_b$. In this figure, the error is plotted against number of bits observed. Please recall from Chapter 3 that every data vector corresponds to one bit interval. The parameter *load* defines the size of the data matrix or the number of

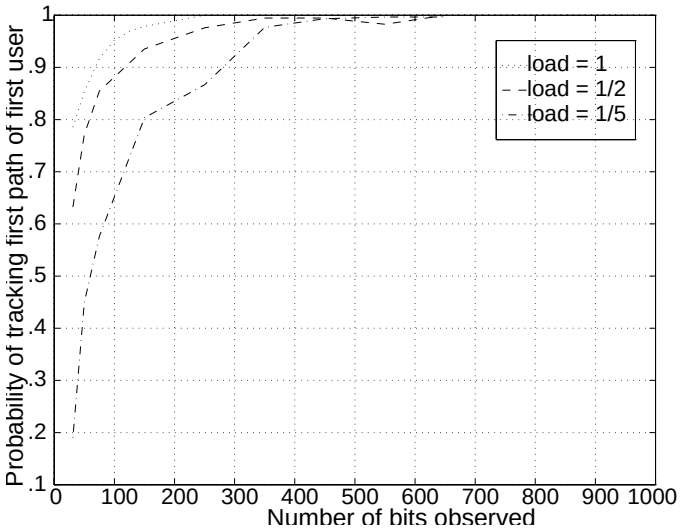


Figure 5.7 : Probability of tracking first path of weakest user vs. number of bits observed, $SNR = 15dB$, $K = 3$, $P = 2$, $MAI = 20dB$, Doppler frequency is $1.04 \times 10^{-3}/T_b$.

data vectors used to form the observation matrix. Hence, $load = 1/2$ implies that the data vector corresponding to every other bit has been used to form the data matrix. Similarly, $load = 1/5$, implies that every fifth data vector has been used to form the data matrix.

Figure 5.7 shows that when $load = 1/2$, the same performance is obtained as when all data vectors are used ($load = 1$), but after observing a larger number of bits. However the number of bits that need to be observed to obtain the same performance with $load = 1/2$, is less than twice that are required when all the data vectors are used. For example, to obtain a probability of tracking of 0.9, either we need to observe 65 bits and use $load = 1$ (every bit observed) or we need to observe 115 bits and use $load = 1/2$ (every other bit observed). However, when $load = 1$, we need to update every bit interval, but with $load = 1/2$, we need to update every other bit interval. Also, the size of the data matrix with $load = 1/2$ is half of that

with $load = 1$. Using this technique, we can gain more time in which to perform each update, but at the cost of collecting data over more bit intervals or at the cost of some loss in performance.

5.5 SVD updating on a fixed sized linear array of processors

In Section 5.4.1, the idea of SVD updating was introduced. The updating approach helps in the reduction of computational load per update by spreading the computation over time. In this section, we present a new SVD updating algorithm [77]. The application of this algorithm to CDMA channel estimation has already been shown in Section 5.4.1, Figure 5.6. The primary aim of developing this new SVD updating algorithm is to design a scheme which is suitable for a software solution on a fixed sized array of off-the-shelf DSPs, as opposed to the necessity of a systolic solution consisting of a large number of custom processors [74].

It may be noted that the proposed SVD updating algorithm is not only applicable to the subspace based channel estimation algorithm but also to any other application which can employ the technique, such as, direction of arrival, frequency estimation, etc.

We propose a variation of the Jacobi rotation based SVD update scheme for the two previously discussed window structures on the incoming data, and an efficient parallelization scheme. The proposed SVD updating scheme is especially suitable for implementation on a fixed and possibly under-sized linear array of off-the-shelf processors, to achieve speed ups close to the number of processors used and with a finite or sliding window structure.

It is evident that with a combination of the two types of windows (exponential and sliding) and by varying the choice of the window parameters, various windowing

formulations are possible that will track the subspace with varying degrees of speed and accuracy. However, from the implementation point of view the sliding and exponential windowing schemes have different requirements in terms of the structure of the algorithm. We will deal with the two types separately in the description of our algorithm.

As the problem needs to be solved in real time, it is natural to look for a parallel algorithm so that computation time can be reduced by distributing the work among a number of processing units. We also propose a parallel scheme for our SVD updating algorithm that can be implemented on a fixed sized linear array of off-the-shelf processors.

5.5.1 Algorithm description

Our algorithm is based on Hestenes method [80] for computing the SVD. In the Hestenes method, starting with any given matrix Y, an orthogonal matrix V is built such that YV has orthogonal columns. Thus, $YV = U\Sigma$, where U has orthonormal columns and Σ is non-negative and diagonal. The SVD of Y is $Y = U\Sigma V'$. Details of the construction of V have been presented before in Section 5.4.2.

Let us denote $U\Sigma$ by the single matrix X. Thus given matrix Y, Hestenes method yields matrices X and V such that

$$Y = XV'. \tag{5.19}$$

U and Σ can be obtained from X as follows - the singular values are equal to the norms of the columns of matrix X, and U can be obtained by normalizing the columns of X. However, since most subspace-based applications require V only, this step is not necessary.

Exponential window:

In the exponential window case, we can relate matrix Y_i and Y_{i-1} as:

$$Y_i = \begin{bmatrix} \beta Y_{i-1} \\ \mathbf{r}_i' \end{bmatrix} \tag{5.20}$$

Using equation 5.20 and the SVD of Y_{i-1} from (5.19), yields,

$$Y_i = \begin{bmatrix} \beta X_{i-1} \\ y_i' V_{i-1} \end{bmatrix} V_{i-1}'. \tag{5.21}$$

Let us denote the first matrix on the right hand side of equation 5.21 by Z_i. The columns of Z_i are orthogonalized using j sweeps of Hestenes method [80], such that $Z_i = X_i \tilde{V}_i'$. So we get the desired update of the SVD as:

$$Y_i = X_i \tilde{V}_i' V_{i-1}' = X_i V_i', \tag{5.22}$$

where $V_i = V_{i-1}\tilde{V}_i$.

Simulations of application of this method to the frequency estimation problem (refer Section 5.5.2) show that a "good" enough approximation is obtained after just one *sweep* of Hestenes method, for each update. Here a *sweep* is defined as the process of orthogonalizing each pair of columns, once, in any order.

Sliding window:

In the sliding window case, we can relate Y_i and Y_{i-1} as :

$$\begin{bmatrix} 0 \\ Y_i \end{bmatrix} = \begin{bmatrix} Y_{i-1} \\ \mathbf{r}_i' \end{bmatrix} - \begin{bmatrix} \mathbf{r}_{i-W}' \\ 0 \end{bmatrix} \tag{5.23}$$

In our proposed scheme for sliding window, we basically update the SVD of the matrix on the left hand side of equation 5.23. In the rest of this section, we shall denote this matrix (matrix Y_i with a zero row appended at top) as Y_i.

Again, from time step $i-1$, we have the SVD of matrix Y_{i-1}. The updating algorithm is based on the following equations:

$$Y_i = \begin{bmatrix} Y_{i-1} \\ \mathbf{r}'_i \end{bmatrix} - \begin{bmatrix} \mathbf{r}'_{i-L} \\ 0 \end{bmatrix} = \begin{bmatrix} X_{i-1}V'_{i-1} \\ \mathbf{r}'_i \end{bmatrix} - \begin{bmatrix} \mathbf{r}'_{i-W} \\ 0 \end{bmatrix} \tag{5.24}$$

$$= \begin{bmatrix} \begin{bmatrix} X_{i-1} \\ \mathbf{r}'_i V_{i-1} \end{bmatrix} - \begin{bmatrix} \mathbf{r}'_{i-W} V_{i-1} \\ 0 \end{bmatrix} \end{bmatrix} V'_{i-1}. \tag{5.25}$$

The columns of the first matrix on the right hand side of equation 5.25 are orthogonalized using j sweeps of Hestenes method [80]. So we get :

$$Y_i = X_i \tilde{V}'_i V'_{i-1} = X_i V'_i, \tag{5.26}$$

where $V_i = V_{i-1}\tilde{V}_i$. Again, simulations of application of this method to the frequency estimation problem show that a "good" enough approximation is obtained after just one *sweep* of Hestenes method, for each update. This scheme requires an extra matrix-vector multiplication which is not required in the exponential window case, which however, requires the multiplication with β.

It should be noted that in addition to the Hestenes method based SVD updating algorithm with exponential and sliding windows, the proposed update scheme and its corresponding parallelization described later, can also be applied to the "spherical noise subspace" methods [78]. The "spherical noise subspace" method reduces the complexity of any SVD update scheme by forcing the noise subspace to be spherical, which makes it easy to deflate the problem.

5.5.2 Evaluation of tracking capabilities of proposed scheme

In Section 5.4.1, Figure 5.6, we have already shown the applicability of the proposed SVD updating scheme to the channel estimation problem.

In order to further show the tracking capabilities of our algorithm, we will use another classical example - spectral estimation. The specific example is that used by Ferzali and Proakis in [76]. In this example, a covariance eigenstructure based algorithm is used in a spectral estimation application, where the problem is to track the variations in the instantaneous frequencies of a signal $s(n)$ composed of multiple, superimposed, time varying sinusoidal components. The received signal $r(n)$ is the sum of the k sinusoids contaminated with additive zero mean Gaussian white noise $\eta(n)$. For this specific example, n denotes time, and m is the size of the observation vector, such that

$$r(n) = \sum_{i=1}^{k} A_i sin(\omega_i n) + \eta(n). \tag{5.27}$$

The data matrix Y_n is constructed from $r(n)$. The eigenvector based estimation algorithm used was MUSIC [81].

Please note that in this chapter, m denotes the size of the data vector for the generic subspace-based approach and should not be confused with the m^{th} sensor used in the previous chapters. When the subspace-based approach is applied to CDMA channel estimation, the size of the data vector m takes on a value of N, the size of the spreading code.

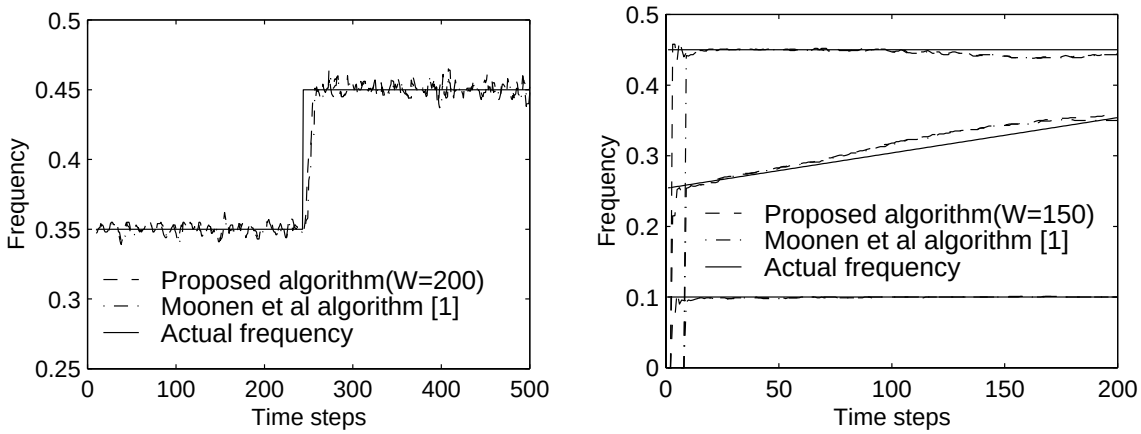
In the first experiment, we track an abruptly changing subspace. A sudden jump in frequency is introduced. This changes the subspace suddenly. The first 250 data vectors were generated from

$$r_1(n) = 2cos(2\pi \times .35t) + \eta(n). \tag{5.28}$$

Another 250 data vectors were formed from

$$r_2(n) = 2cos(2\pi \times .45t) + \eta(n), \tag{5.29}$$

where the frequency of the only sinusoid has jumped from .35 to .45 Hz. Data vectors of size m were constructed. Figure 5.8.a shows that our proposed scheme and Moonen et al's method [74] perform very similarly with only 1 sweep of their respective SVD algorithms for the same values of β.



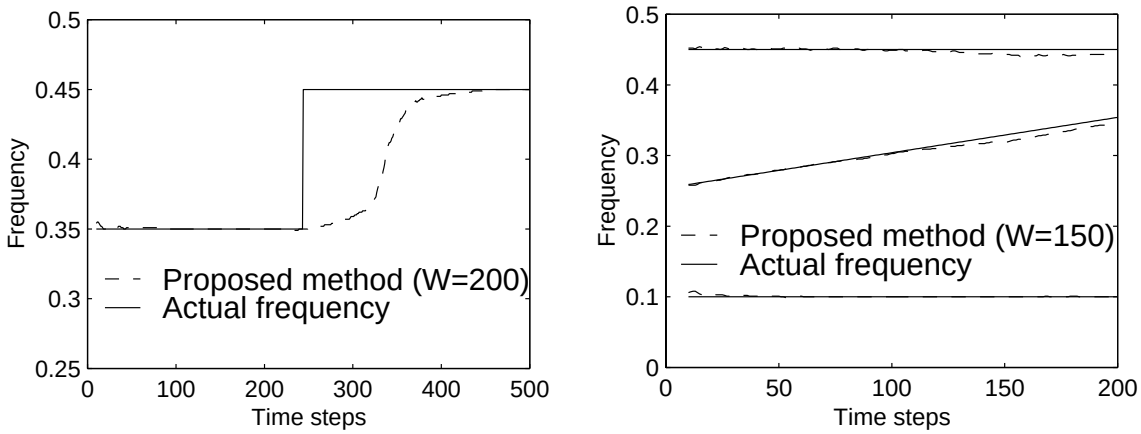
(a) (b)

Figure 5.8 : Application of SVD updating to the frequency estimation problem with exponential window. SNR = 10dB (a) Tracking an abruptly changing subspace, $\beta = 0.9$, $m = 7$. (b) Tracking a slowly changing subspace, $\beta = 0.9935$, $m = 9$.

In the next example, we track a slowly but continuously changing frequency. The data vectors were generated from the received signal

$$r_3(n) = 2cos(2\pi \times .10n) + 2cos(2\pi \times (.25 + (inc \times n))n) + 2cos(2\pi \times .45n) + \eta(n). \tag{5.30}$$

The gradual change in the second frequency is defined by inc and the value of inc used in our experiments was 0.1/200, that is, inc changes by 0.1 over 200 time steps. Again, both schemes perform very similarly and successfully track all 3 frequencies (Figure 5.8.b).



(a) (b)

Figure 5.9 : Application of SVD updating to the frequency estimation problem with sliding window. SNR = 10dB (a) Tracking an abruptly changing subspace, $w = 200$, $m = 7$. (b) Tracking a slowly changing subspace, $w = 150$, $m = 9$.

Moonen et al's SVD algorithm is constructed by combining QR updating with a Jacobi type SVD diagonalization procedure (Kogbetliantz's algorithm, modified for triangular matrices). As the Hestenes method for SVD is a modified version of Jacobi's method for SVD, we can expect that both algorithms compared in this section will perform very similarly, when tracking a subspace. This statement is substantiated by our simulation results.

Figure 5.9 shows the performance with sliding window. As discussed earlier this windowing scheme performs better during stationary periods but takes more time to recover after an abrupt change in the subspace.

We have shown the ability of the proposed scheme to track a time varying subspace and its applicability to the classical frequency estimation problem as well as the CDMA synchronization problem. In the next section, we will discuss our scheme for parallelizing the algorithm on a fixed sized array of off-the-shelf processors.

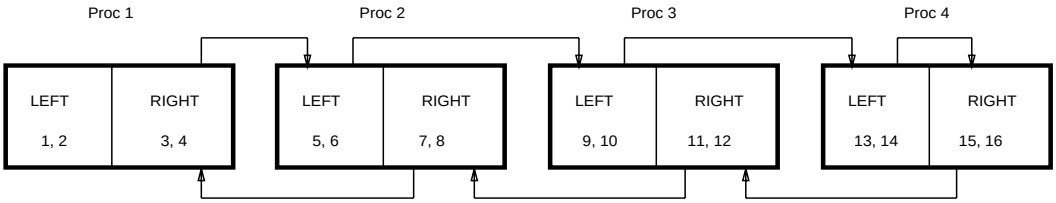


Figure 5.10 : The linear array of processors used for the SVD updating, P = 4, m = 16. The arrows indicate connections between processors.

5.5.3 Parallelization scheme for SVD updating

To date, the most powerful parallel scheme for SVD updating is the systolic array proposed by Moonen, Van Dooren and Vandewalle [74] for Moonen et al's SVD updating algorithm [82]. The data dependence structure of Moonen et al's two dimensional systolic algorithm is such that at each update, the computation using each column depends on the results of the computation using all the columns to its left. Also, computation using each row depends on the results of the computation using all rows above it. Thus, the data cannot be easily partitioned onto a fixed number of processors. A *fine grained parallelization* scheme is required [74], which will depend on pipelining of the problem on a large number of custom-built processing units. The number of processing units required is proportional to the size of the problem (defined by $m/2$).

However, in our algorithm, at each update, the computation using each pair of columns is independent of the computation using all other columns. Hence, it can be easily partitioned onto a linear array of processors of size less than (or equal to) the size of the problem, to get speedups close to the number of processors used.

Our parallelization scheme for the SVD updating problem is based on the solution of singular value problems on an under-sized linear array proposed by Schreiber [83],

using the Hestenes method [80] for the SVD. In the rest of this section, we will restrict our discussion to the exponential window case, though the sliding window case follows closely.

The linear array of C processors (with $C \leq m/2$) used for our scheme is shown in Figure 5.10. Again, m is the size of the data vector and has the value of N or the size of the spreading code for the channel estimation algorithm. Let us recall, that at the beginning of each time step, i, we have the matrices X_{i-1} and V_{i-1} from the SVD update of the previous step. The number of columns of each of these matrices is equal to the size of the data vectors, m. Figure 5.10 shows the initial distribution, (at the beginning of each time step) of the columns of both the matrices on the C processors. The data memory of each processor is conceptually divided into 2 parts - LEFT and RIGHT memory. Initially, processor j holds columns $(j-1) \times \frac{m}{C} + 1$ to $j \times \frac{m}{C}$ of X_{i-1} and V_{i-1} in its data memory. The columns are distributed equally between its LEFT and RIGHT memory as shown in Figure 5.10. This implies LEFT and RIGHT memory holds $m/2C$ columns each.

At each time step, the data vector $\mathbf{r}_i$ is broadcast to all the processors by a host processor. The broadcasting of $\mathbf{r}_i$ can be done elegantly by pipelining the broadcasting of $\mathbf{r}_{i+1}$ with the computation of time step i, if $\mathbf{r}_{i+1}$ is already available. Once, $\mathbf{r}_i$ is available to all the processors, each processor multiplies its columns of X_{i-1} with β and the columns of V_{i-1} by $\mathbf{r}_i'$ (refer equation 5.21). Thus we form the first matrix in the right hand side of equation 5.21, that is, matrix Z_i. We then orthogonalize the columns of Z_i, as described below.

The array performs one *sweep* of Hestenes SVD algorithm. In each *sweep*, first, each processor orthogonalizes all possible pairs of columns in its LEFT and RIGHT memory. These pairs are formed by taking both columns either from the LEFT or

RIGHT memory but not one from each. The rest of the sweep is performed in $(2C-1)$ *cycles*.

In each *cycle*, each processor orthogonalizes all possible pairs of columns formed by taking one column from its LEFT memory and one from its RIGHT memory. Then, the processors exchange the contents of their LEFT and RIGHT memories along the arrows shown in Figure 5.10. The connections between the processors are such that in each *sweep* (that is, in $(2C-1)$ *cycles*) all possible pairs of columns of Z_i are orthogonalized exactly once [83]. The transformations applied to Z_i are also applied to V_{i-1} to get V_i.

Let the time required to apply one single rotation ($Q_k^{(j)}$ in equation 5.13) be t_r. Also let t_c be the time required to transfer m/C columns from one processor to another. Then we can calculate the speedup as :

$$Speedup = \frac{Sequential\ \ execution\ \ time}{Parallel\ \ execution\ \ time} = \frac{(m(m-1)/2)t_r}{(m(m-1)/2C)t_r + (2C-1)t_c} \quad (5.31)$$

That is, the speedup obtained scales with the number of processors, especially for larger values of m. For larger values of m the first term in the denominator of equation 5.31 dominates over the second and the speedup is close to C. This is illustrated in Figure 5.11(b) which shows that the fraction of the total processing time spent on communication between processors in the parallel implementation decreases, as m increases. Consequently, the speedup increases and gets closer to C, as m increases.

5.5.4 Advantages of proposed algorithm

The proposed algorithm and parallelization scheme can be implemented on a fixed sized array of off-the-shelf DSPs (Digital Signal Processors) thus avoiding the high design time and cost as well as inflexibility due to custom processors. Our scheme uses

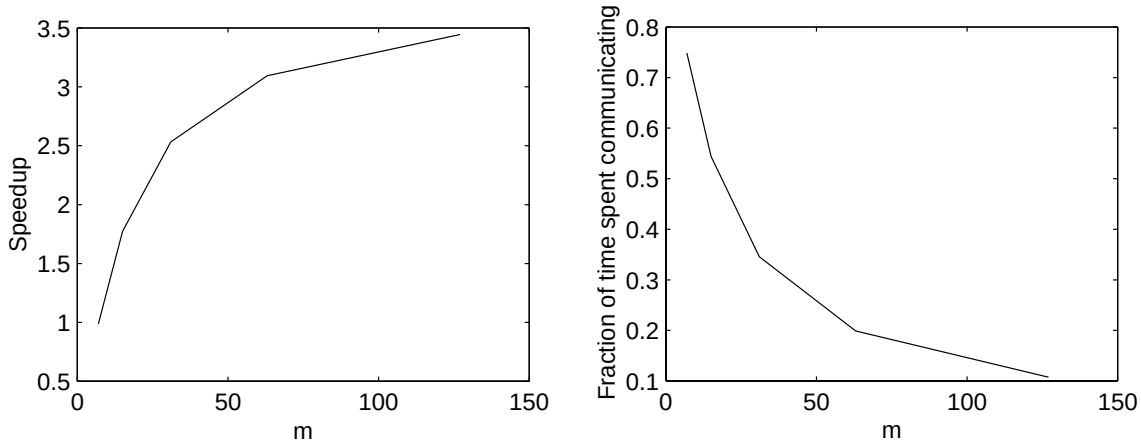


Figure 5.11 : Parallelization on a linear array of 4 Texas Instruments TMS320C40 DSP chips, sliding window, L=150. (a) Speedup vs. size of matrix, m (b) Fraction of processing time spent on communication between processors in the parallel implementation.

coarse grained parallelism, that is, communication between processors is interleaved with computation involving $\frac{m}{C}$ columns of matrix X, instead of only a few elements. Thus the array does not spend a large amount of time loading and unloading data. Hence, it is very suitable for implementation on an array of typical DSPs like the Texas Instruments TMS320C40, which has single-cycle functional units, but requires several cycles to transfer data across the communication ports.

A software solution on off-the-shelf DSPs would be more flexible with regards to changing parameters like β, m and the size of the signal subspace according to the time-variance of the data vectors. Also, the subspace-based algorithm which uses the results of the SVD can be implemented on the same DSPs. This would reduce the physical size of the hardware and would be of great advantage in applications like subspace-based CDMA synchronization [25], where this hardware would be a part of a base station. Also, as this application requires the SVD update of matrices with large

values of m [25], the proposed parallelization scheme would be particularly useful in this case.

The parallelization scheme can be implemented on a scalable number of processors. Even if C is much less than the problem size, (that is $m/2$), this scheme will yield speedups close to C and processor utilization close to 100%. Hence, with a processor array of given fixed size, we can update the SVD of arbitrarily large matrices.

5.6 Parallelization of subspace based channel estimation

In this section, we present the results of parallelization of the subspace based channel estimation algorithm [25] in the time-invariant case. Since the subspace-based algorithm require quite a lot of computation, it is convenient to partition the algorithm into smaller parts. An intuitive and convenient way to handle these algorithms is to split them into a frontend subspace decomposition and a backend non-linear optimization problem. Other factors which guide such a treatment is that the subspace tracking problem is a well-defined linear algebra problem that has been studied extensively in the literature. The problem is characterized by regular computation ideally suited for parallel arrays. The frontend computation is almost completely independent of the algorithms used in the backend and often independent of the overall application.

The backend computation on the other hand is marked by irregular computation, where the execution times may not be predictable. Often a closed form solution is not feasible due to the multimodal nature of the objective functions, and an extensive search may be needed to solve the problem. The backend problem formulation is much more dependent on the overall problem. The above separation also allows development of several alternative designs for the frontend and backend, which can

then be mixed and matched at will with very few restrictions. A complete treatment of these alternatives can be found in [50].

The goal in this section is to prove the feasibility of coarse-grained parallelization of the subspace based channel estimation algorithm for reasons listed in Section 5.5.4. We will show that our scheme achieves high processor utilization on an array of DSPs which makes it suitable for implementation on off-the-shelf DSPs instead of custom hardware. In order to evaluate the parallelization of the subspace based algorithm, the TI Parallel Processing Development System (PPDS) was chosen, as the PPDS and its related software gave a quick and efficient means of testing out the algorithms on an existing parallel system. The hardware platform used, consists of the TMS320C40 PPDS connected to a Sun workstation through a XDS510WS emulator.

The TMS320C40 is a parallel processing DSP. In addition to a 32MHz CPU it contains 6 communication ports and a multichannel DMA [84] controller. The on-chip communication ports allow direct processor-to-processor communication, and the DMA unit provides concurrent I/O by running parallel to the CPU. The TMS320C40 PPDS has four on-board TMS320C40s, each supported by a local bus with $64K \times 32$-bit words of zero wait-state static RAM [85]. The 4 processors are fully connected and can hence be used to implement almost any parallel configuration. All these features make the TMS320C40 PPDS very suitable for our purpose. As this scheme could be used in a receiving base station, parallelization upto 4 processors is a reasonable hardware environment.

As discussed in [86], the TMS320C40 has single-cycle functional units, but, several cycles are required to transfer data across the six communication ports. Thus, the PPDS is not particularly suited for fine grained parallel processing. This motivated the choice of the parallelization technique for both the frontend and the backend. If

the PPDS were configured as a 2×2 array or a pipelined array handling only a few matrix elements or a single column at a time, the communication overhead compared to the computation time would be large. Hence, processor utilization would be low. Instead, the PPDS was configured as a linear array of 4 processors and the columns of the data matrix were distributed among the 4 processors. Each processor works independently on its own set of columns and then exchanges columns with other processors. Thus, this scheme implements coarse grained parallelism which gives speedups close to 4 and hence, high processor utilization.

5.6.1 Implementation of SVD on the PPDS

For the frontend, the Hestenes method, which is a non-recursive procedure was chosen. Such an algorithm should be used when the singular values and singular vectors are being computed from scratch. Once the SVD is obtained using such a non-recursive procedure, a recursive algorithm may be used later to update the decomposition whenever new data arrives.

The Hestenes method for SVD was parallelized based on [83] and as described in Section 5.5.3, but for the time-invariant case. The 4 processors on the PPDS are connected as shown in Figure 5.10. The number of processors remains fixed even though the problem size, that is the size of the input matrix, varies. The partitioning scheme in the algorithm takes care of distributing the problem on the fixed and under-sized linear array. The size of the input matrix which is decomposed, is $N \times L$, where N is the spreading code size and L is the block size or the number of data bits.

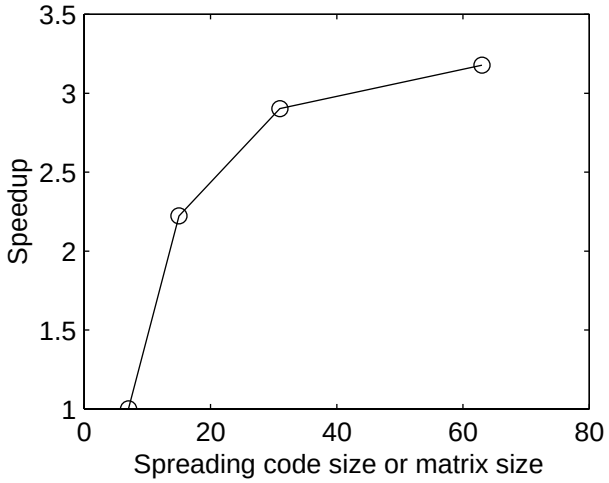


Figure 5.12 : Frontend - speedup vs. spreading code size (N).

Performance measurements :

The performance of the implementation was tested using a random number generator to generate the transmitted data for each of the K users. Each user's spreading code was generated using length N (spreading code size) Gold codes. The use of the Gold codes as spreading codes restricts N to $2^k - 1$, k being an integer. So, while executing the frontend, N was varied from 7 to 63 and the blocksize L is equal to N. That is, an $N \times N$ matrix was decomposed.

Any parallel implementation incurs two main cost components - computation time, which includes execution time for all arithmetic/logic operations and communication time, which includes time for data movement between processors. The most commonly used criteria used to evaluate performance of a parallel algorithm is speedup, defined as $S_p = T_s/T_p$, where T_s is the time taken when the algorithm executes sequentially on one processor and T_p is the parallel run time. Typically, the maximum speedup that can be obtained using p processors is p. However, due to factors like

memory access conflicts and communication delays, the speedup in most real implementations is less than p.

The speedup figures are reported (Figure 5.12) for the frontend as the input matrix size is increased (that is as the spreading code size is increased). The speedup increases as the size of the matrix increases. This is to be expected as the communication overhead decreases when compared to the computation time as the matrix size increases. As N increases, the speedup increases to 3.2, that is, a maximum processor utilization of $3.2/4 \times 100\% = 80\%$.

In this implementation, the data resides in the off-chip local memory. Code optimizations such as prefetching data into the on-chip RAM while DMA data transfer occurs between processors should considerably improve performance.

5.6.2 Implementation of the backend optimizer on the PPDS

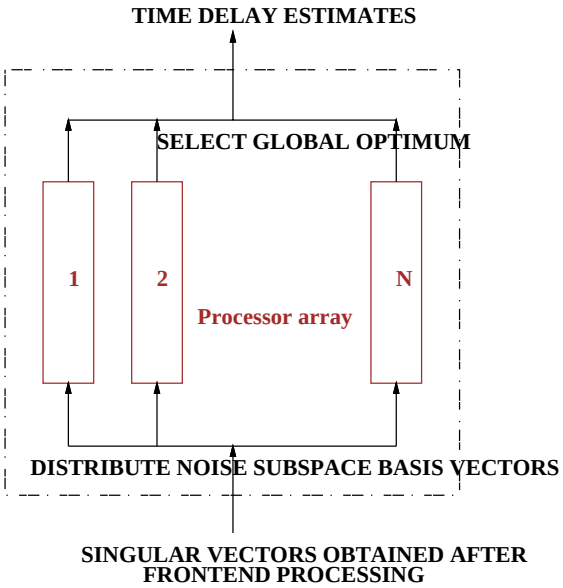


Figure 5.13 : Parallel architecture to implement the backend optimizer for each user.

The formulation of the backend as a parallel algorithm using N processors is described in Figure 5.13 [25, 50]. The backend was also implemented on the PPDS with the 4 processors configured as a linear array. The processing corresponding to the columns of the parallel array shown in Figure 5.13 were distributed among the 4 processors of the PPDS. The architecture consists mainly of a linear array for the matrix multiplication for the projection of the signal vectors onto the basis for the noise subspace. The same array is also used to extract the delays from the estimated channel impulse response vector [17].

Performance measurements :

Two separate experiments were carried out in this section. In the first, (Figure 5.14) the dependence of speedup of the backend on the spreading code size was analyzed with the number of users being fixed at 2. In the second, (Figure 5.15) speedup as well as execution time was measured as the number of users K was varied while the spreading code size was fixed at $N = 31$. The variation of execution time with K helps illustrate the inherent parallelism in the backend and how that can be exploited. The speedup plots with N and K, show the high processor utilization of the scheme, especially at large matrix sizes.

In the backend, the work for each user, was divided among the 4 processors. However, since the estimation of the delay of each user is independent of the delays of all other users, an obvious approach towards parallelization would be to perform the delay estimate of each user on separate sets of 4 processors. These experiments were done on one PPDS, with only 4 processors, that is, the work for each user was divided among the 4 processors, but the K users were handled sequentially. However, the execution times for estimating delays of all K users on 4 processors

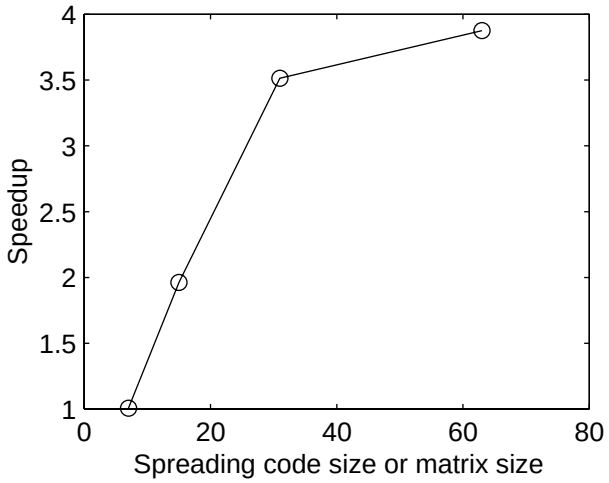
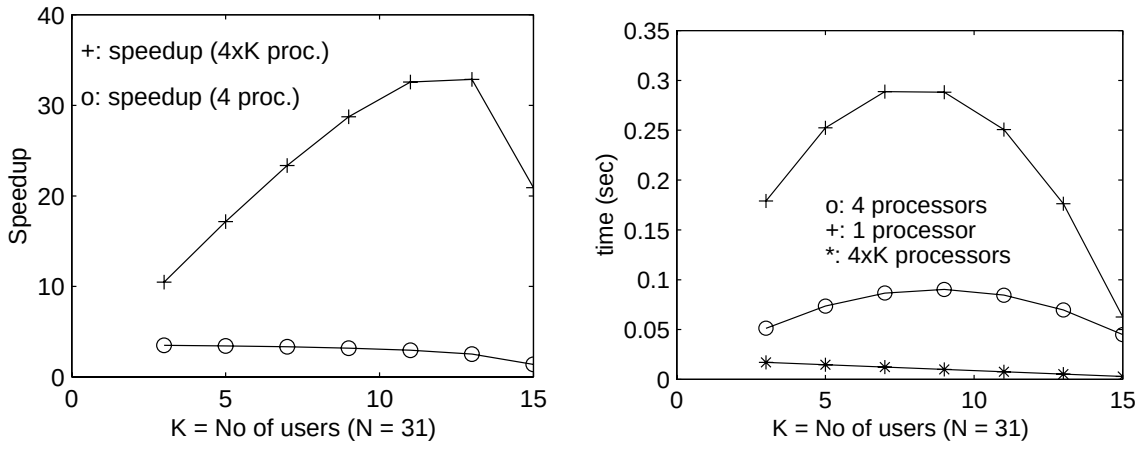


Figure 5.14 : Backend - speedup vs. spreading code size (N).

can easily be extrapolated to execution times for estimating delays of all K users on $4 \times K$ processors. That is, the delay of each user is estimated on separate sets of 4 processors. This extrapolation is done using the relation : $t_{4\times K} = t_4/4$, where t_4 is the execution time of delay estimation of K users on 4 processors and $t_{4\times K}$ is the execution time of delay estimation of K users on $4 \times K$ processors. Figure 5.15 shows the variation of speedup and execution time with 4 as well as $4K$ processors with K.

An important point to be noted from Figure 5.15(b), is that the execution time for the backend, on 1 or 4 processors, increases as K increases, until $K = N - 2K$. As K increases further the execution time for the backend decreases. This is because, the size of the noise subspace decreases as K increases. So, for each user, there is less work to do in the estimation steps, as K increases. On the other hand, since the 4 processors handle the K users sequentially, there are more delays to be estimated as K increases. However, with $4 \times K$ processors, the K users are handled in parallel, so the execution time continuously decreases as K is increased.



(a) (b)

Figure 5.15 : Backend - speedup and execution times vs. number of users (K).

In this section, we have presented a coarse grained parallelization scheme for the subspace based algorithm. We implemented the scheme on the TI PPDS to demonstrate that high processor utilization can be achieved by the scheme on an array of off-the-shelf DSPs.

5.7 Fixed point error analysis

In a practical implementation of a communication system, fixed point hardware is an attractive alternative in terms of increased speed, reduced power consumption and reduced hardware cost. In this section, we analyze the error pattern of the subspace based algorithm when implemented on fixed point hardware.

We can recall from the previous chapter that any fixed point number can be represented with a fixed l_i bits for the integer part and l bits for the fractional part (i.e. l_i bits before the binary point and l bits after). The dynamic range of the problem determines l_i. For our problem, appropriate normalization of the data can

eliminate the need for the integer part, l_i. However l is determined by the precision requirement of the algorithm and requires more careful analysis. In this section, we will show the dependence of the fixed point error in each step on the various system parameters.

The most computationally complex operation in the subspace based method is the SVD of Y and hence it also gives rise to the dominant term in the expression for error due to fixed point implementation of the method. We will assume that the Frobenius-norm of the matrix Y is bounded by unity. Using Hestenes method for SVD [77, 80], we get : $Y = XV'$, where $X = U\Sigma$, and $Y = U\Sigma V'$ is the SVD of Y. The unitary matrices, U and V, contain the left and right singular vectors and Σ is the diagonal matrix of singular values.

For a fixed point implementation, instead of X and V, we get inexact matrices $\tilde{X}$ and $\tilde{V}$, where

$$\tilde{X} = X + R, \qquad \tilde{V} = V + Q, \tag{5.32}$$

and where R and Q are error matrices. It can be shown, following the fixed point error analysis in [67], that the F-norm of the matrices R and Q are bounded by :

$$C_1 N^{\frac{5}{2}} L w 2^{-l} \tag{5.33}$$

where C_1 is a constant, L is the number of observation vectors in the matrix Y, and w is the number of SVD sweeps.

Modeling the error due to fixed point implementation as added noise to the data matrix Y :

$$\tilde{Y} = Y + N_o = \tilde{X}\tilde{V}' = (X + R)(V + Q)'. \tag{5.34}$$

$$N_o = XQ' + RV', \tag{5.35}$$

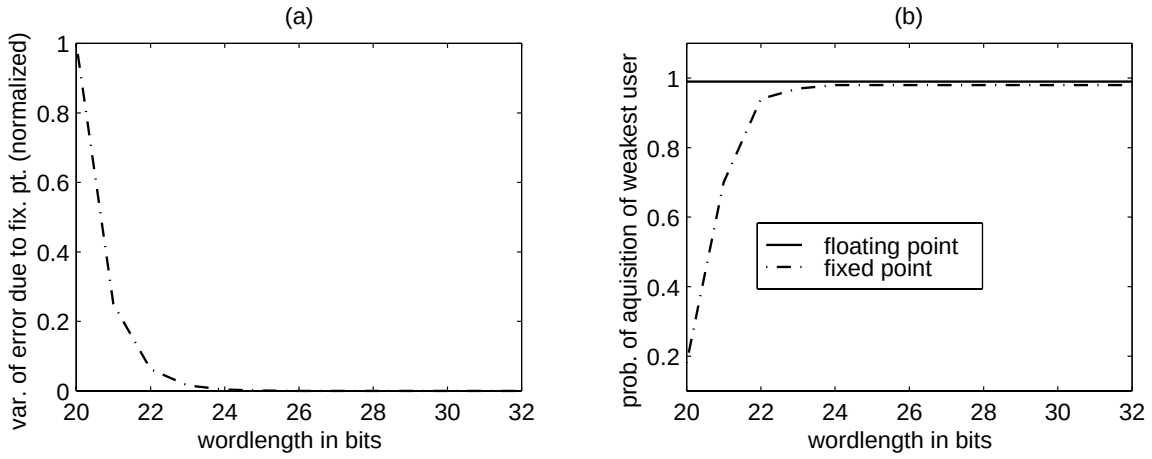


Figure 5.16 : (a) Variance of the error (normalized) due to fixed point implementation of the SVD (analytic). (b) Probability of synchronization for fixed point and floating point implementations (simulated).

ignoring second order terms.

A similar analysis shows that the norm of the error arising from the actual parameter estimation steps, using the subspace decomposition, is bounded by

$$C_2 N 2^{-l} \tag{5.36}$$

where C_2 is a constant. Obviously this error is much lower compared to the error due to the SVD.

In Figure 5.16(a), we plot the variance of the added fixed point 'noise' against the number of bits used in the fractional part for the fixed point implementation for a system with N=31, $K = 5$, $SNR = 8db$, $MAI = 20db$, $L = 200$, and with fixed channel parameters without any multipath fading. For the same system parameters specified above, the subspace estimate obtained from a fixed point simulation of the SVD was used to estimate the delays of the users. Figure 5.16(b) shows the probability of synchronization of the weakest user versus the number of bits used. Synchronization is defined as the estimation of the delays to within half a chip of the

actual delays.

The plots show that the error from the fixed point calculation becomes 'small' and the probability of synchronization becomes acceptable after about 24 bits. The conclusion drawn from the totally simulated probability of synchronization plot and the analytic expression for the added error due to fixed point calculations is the same: it will be possible to use 24-32 bit fixed point hardware for the synchronization step. The expressions for the fixed point error (5.33) and (5.36) show that the size of the spreading code is the most important factor in the error accumulation in the subspace based synchronization method. Especially for the SVD, the size of the spreading code is directly related to the number of orthogonalization operations that have to be executed, and hence the parameter N dominates the error expression.

5.8 Summary

In this chapter we evaluated the performance of a subspace-based channel parameter tracking algorithm. We discussed the tradeoffs involved in choosing a windowing scheme and its parameters and showed that the performance of the parameter estimation algorithm depends heavily on the efficiency of the subspace estimation. We showed that the subspace-based algorithm tracks time varying channel parameters very well, provided the data window structure and window parameter is chosen appropriately. Also, since the subspace tracking algorithm is computationally complex, we explored the applicability of several approximate techniques. Such approximations can be used to reduce the computational complexity from $O(N^3)$ to $O(N^2)$, to remove square-root operations, and to reduce the size of the data matrix by skipping over some data vectors, at a very small cost in terms of loss in performance.

We also studied two implementation issues related to the subspace based schemes

- development of an SVD updating algorithm and its parallelization scheme and fixed point error analysis of the overall channel estimation algorithm. Finally we presented a parallelization of the entire algorithm on a linear array of DSPs.

The performance analysis of the parallel implementation shows good processor utilization resulting in speedups close to 3.8 using 4 processors and an appropriate data size. This analysis shows that the chosen parallelization schemes are suitable for implementation on an array of DSPs.

Chapter 6

Conclusion and Future Work

In this thesis we have addressed the problem of channel estimation for CDMA communication systems. We focus on *multiuser* techniques for channel parameter estimation which exploit the knowledge of the structure of the interfering users and work in a multipath environment in the presence of multiple sensors.

The contributions of this thesis may be summarized as follows. First, we developed an efficient system model which elegantly captures the effect of multiple paths, multiple sensors and also allows multi-shot detection. Second, we developed a maximum likelihood algorithm for channel estimation in the complex scenario involving multiple paths and multiple sensors at the receiver. It may be noted that none of the previously proposed techniques for channel estimation work efficiently in this complex scenario. We also focused on the interaction between detection and channel estimation and developed a scheme to capture and estimate the effect of the channel in the fewest possible steps, such that computation is reduced and performance is improved.

Third, we extended a subspace based channel estimation algorithm to the time varying case and studied the effects of the size and shape of the tracking window. We also applied various techniques to reduce the computational load for this problem.

Lastly, we have focused on implementation issues for the channel estimation problem related to both the ML and the subspace-based technique, such as, complexity reduction through algorithmic modifications, parallelization, and fixed point error analysis.

6.1 Future work

In certain scenarios, such as emergency call location, it is desirable to estimate the angle-of-arrival of the signal of each user. For this reason, in the future, the ML algorithm may be extended to include an efficient technique to extract the angle-of-arrival from the estimated channel impulse response. Such a technique will require the assumption of some particular antenna array structure, such as a uniform linear array or circular array.

The next step in the development of the ML algorithm is the extension of the algorithm to the time varying scenario. Once the channel is estimated using the known preamble, it will be necessary to track the channel, in a decision directed mode. The decision directed mode refers to the feedback of the detected signals to update the channel estimate.

Also, there is another important problem that arises out of the tracking scenario for both channel estimation algorithms. That is, determination of an effective measure of time variation and an algorithm which will allow us to vary the window type or parameter and to choose between the various approximate techniques according to the rate of variation of the channel.

The idea of joint synchronization and detection introduced in Chapter 4 opens up the possibility of development of a number of joint schemes based on different channel estimation and linear detection algorithms. Such schemes may be targeted to various scenarios - multiuser techniques for uplink or blind techniques for downlink.

Finally, the ideas presented in this thesis can be tailored to future cellular standards to increase the physical layer performance of such systems. Wideband CDMA has been accepted as the multiple access standard in the third generation (3G) cellular systems to be deployed around 2005. Standardization efforts for such a system

are underway in Europe, Japan, and the United States. The application of maximum likelihood or subspace-based channel estimation or joint synchronization and detection to the design of the *advanced receiver structures* [41] mentioned in the context of 3G systems will be an interesting follow-up to this thesis.

Bibliography

[1] M. J. Riezenman, "Communications," *IEEE, Spectrum, Technology 1998, Analysis and forecast issue*, vol. 35, no. 1, pp. 29–36, Jan. 1998.

[2] P. J. Walsh, "Wireless designers heed IP call," *Portable Design*, pp. 51–52, April 1998, PennWell Publishing Company, Tulsa, OK, http://www.portabledesign.com.

[3] M. McMahan, "Wireless telephony at the crossroads," *Portable Design*, pp. 53–55, April 1998, PennWell Publishing Company, Tulsa, OK, http://www.portabledesign.com.

[4] S. Verdu, *Multiuser detection*, Cambridge University Press, 1st edition, 1998, New York, NY.

[5] R. L. Pickholtz, D. L. Schilling, and L. B. Milstein, "Theory of spread-spectrum communications - A tutorial," *IEEE Trans. Communications*, vol. COM-30, no. 5, pp. 855–884, May 1982.

[6] S. Verdú, "Minimum probability of error for asynchronous Gaussian multiple-access channels," *IEEE Trans. Information Theory*, vol. IT-32, pp. 85–96, Jan. 1986.

[7] R. Lupas and S. Verdú, "Linear multiuser detectors for synchronous code-division multiple access channels," *IEEE Trans. Information Theory*, vol. IT-35, pp. 123–136, Jan. 1989.

[8] M. K. Varanasi and B. Aazhang, "Multistage detection in asynchronous code-divison multiple access communications," *IEEE Trans. Communications*, vol. COM-38, pp. 509–519, Apr. 1990.

[9] U. Madhow and M. L. Honig, "MMSE interference suppression for direct-sequence spread spectrum CDMA," *IEEE Trans. Communications*, vol. COM-42, no. 12, pp. 3178–3188, Dec. 1994.

[10] S. Y. Miller and S.C. Schwartz, "Paramater estimation for asynchronous multiuser communication," *Conference on Information Sciences and Systems*, pp. 294–299, March 1989, Baltimore, MD.

[11] M. K. Varanasi and S. Vasudevan, "Multiuser detectors for synchronous CDMA communication over non-selective Rician fading channels," *IEEE Trans. Communications*, vol. 42, pp. 711–722, Feb. 1994.

[12] Z. Xie, C. K. Rushforth, R.T. Short, and T. K. Moon, "Joint signal detection and parameter estimation in multiuser communications," *IEEE Trans. Communications*, vol. 41, no. 7, pp. 1208–1216, 1993.

[13] A. Radović and B. Aazhang, "Iterative algorithms for joint data detection and delay estimation for code division multiple access communication systems," in *Proc. of the 31st Annual Allerton Conference on Communication, Control, and Computing*, Allerton House, Monticello, IL, 1993, pp. 1–10.

[14] Y. Steinberg and H. V. Poor, "Sequential amplitude estimation in multiuser communications," *IEEE Trans. Information Theory*, vol. IT-40, no. 1, pp. 11–20, Jan. 1994.

[15] E.G. Ström, S. Parkvall, S.L. Miller, and B.E. Ottersten, "Propagation delay estimation in asynchronous Direct Sequence Code Division Multiple Access systems," *IEEE Trans. Communications*, vol. 44, no. 1, pp. 84–93, Jan. 1996.

[16] E. G. Strom, S. Parkvall, S. L. Miller, and B. E. Ottersten, "DS-CDMA synchronization in time-varying fading channels," *IEEE Journal on Selected Areas in Communication*, pp. 1636–1642, Oct. 1996.

[17] S. E. Bensley and B. Aazhang, "Subspace-based channel estimation for code division multiple access communication systems," *IEEE Trans. Communications*, vol. 44, no. 8, pp. 1009–1020, August 1996.

[18] S. E. Bensley and B. Aazhang, "Maximum likelihood synchronization of a single user for CDMA communication systems," *IEEE Trans. Communications*, vol. 46, no. 3, pp. 392–399, March 1998.

[19] R. Madyastha and B. Aazhang, "Antenna arrays for joint maximum likelihood parameter estimation in CDMA systems," *Conference on Information Sciences and Systems*, vol. II, pp. 984–988, March 1997, Baltimore, MD.

[20] D. Zheng, J. Li, S. L. Miller, and E. G. Strom, "An efficient code-timing estimator for DS-CDMA signals," *IEEE Trans. Signal Processing*, vol. 45, no. 1, pp. 82–89, Jan. 1997.

[21] C. Sengupta, A. Hottinen, J. R. Cavallaro, and B. Aazhang, "Maximum likelihood multipath channel parameter estimation in CDMA systems," *Conference on Information Sciences and Systems*, March 1998, Princeton, NJ.

[22] C. Sengupta, J. R. Cavallaro, and B. Aazhang, "Maximum likelihood multipath channel parameter estimation in CDMA systems using antenna arrays," *IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, pp. 1406–1410, September 1998, Boston, MA.

[23] S. Moshavi, "Multi-user detection for DS-CDMA communications," *IEEE Communications Magazine*, pp. 124–136, October 1996.

[24] C. Sengupta, J. R. Cavallaro, and B. Aazhang, "Subspace-based tracking of multipath channel parameters for CDMA systems," *European Trans. Telecommunications, Special issue on CDMA techniques for wireless communication systems*, vol. 9, no. 5, pp. 439–447, Sep.-Oct. 1998.

[25] C. Sengupta, K. Kota, and J. R. Cavallaro, "Parallel algorithms and architectures for subspace based channel estimation for CDMA communication systems," *SPIE, Advanced Signal Processing Algorithms, Architectures, and Implementations VI*, vol. 2846, pp. 412–423, 1996, Denver, CO.

[26] C. Sengupta, S. Das, J. R. Cavallaro, and B. Aazhang, "Fixed point error analysis of multiuser detection and synchronization algorithms for CDMA communication systems.," *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 3249–3252, May 1998, Seattle, WA.

[27] S. Das, C. Sengupta, and J. R. Cavallaro, "Hardware design issues for a mobile unit for next generation CDMA communication systems," *SPIE, Advanced Signal Processing Algorithms, Architectures, and Implementations VIII*, pp. 476–487, July 1998, San Diego, CA.

[28] R. Steele, J. Whitehead, and Wong W. C, "System aspects of cellular radio," *IEEE Communications Magazine*, vol. 33, no. 1, pp. 80–86, Jan. 1995.

[29] T. S. Rappaport, *Wireless communications - Principles and Practice*, Prentice-Hall, Inc., Upper Saddle River, NJ, 1996.

[30] A. F. Naguib, *Adaptive antenna arrays for CDMA wireless networks*, Ph.D. thesis, Department of Electrical Engineering, Stanford University, Stanford, CA, Aug. 1996.

[31] R. Madyastha, *Antenna arrays for wireless CDMA communications systems*, Ph.D. thesis, ECE Dept., Rice University, Houston, TX, 1997.

[32] Andrew J. Viterbi, *CDMA: Principles of spread spectrum communication*, Addison-Wesley Wireless Communications Series. Addison-Wesley, Reading, MA, 1995.

[33] D. V. Sarwate and M. B. Pursley, "Crosscorrelation properties of pseudorandom and related sequences," *Proeedings of the IEEE*, vol. 68, no. 5, pp. 593–619, May 1980.

[34] T. Ojanpera, "Perspectives on 3G system development," *IEEE International Symposium on Personal, Indoor, and Mobile Radio Communications (PIMRC)*, September 1998, Boston, MA.

[35] R. Kohno, R. Meidan, and L. B. Milstein, "Spread spectrum access methods for wireless communications," *IEEE Communications Magazine*, vol. 33, no. 1, pp. 58–67, Jan. 1995.

[36] J. G. Proakis, *Digital communications*, McGraw-Hill, New York, NY, 3rd edition, 1995.

[37] W. C. Lee, *Mobile cellular communications*, McGraw-Hill, New York, NY, 1995.

[38] M. Rahnema, "Overview of the GSM system and protocol architecture," *IEEE Communications Magazine*, vol. 31, no. 4, pp. 92–100, Apr. 1993.

[39] Telecommunications Industry Association, *TIA/EIA/IS-95 Mobile Station-Base Station Compatibility Standard for Dual-Mode Wideband Spread Spectrum Cellular System*, July 1993, http://www.tiaonline.org/.

[40] E. Berruto, M. Gudmundson, R. Menolascino, W. Mohr, and M. Pizarroso, "Research activities on UMTS radio interface, network architectures, and planning," *IEEE Communications Magazine*, vol. 36, no. 2, pp. 82–95, Feb. 1998.

[41] European Telecommunications Standards Institute, "The ETSI UMTS terrestial radio access (UTRA) ITU-R RTT candidate submission," *http://www.etsi.org/smg/UTRA/utra.pdf*, May/June 1998.

[42] Association of Radio Industries and Businesses, "Japan's proposal for candidate radio transmission technology on IMT-2000 : W-CDMA," *http://www.itu.int/imt/2-radio-dev/proposals/*, June 1998.

[43] Telecommunications Industry Association, "The cdma2000 ITU-R RTT candidate submission," *http://www.t1.org/index/0506.htm/8p110690.pdf*, April 1998.

[44] S. Vembu and S. Verdu, "Two different philosophies in CDMA - a comparision," *IEEE Vehicular Technology Coneference*, pp. 869–873, April 1996, Atlanta, GA.

[45] S. Parkvall, "Variability of user performance in cellular DS-CDMA - long vs short spreading sequences," *Submitted to IEEE Trans. Communications*, June 1998.

[46] W. C. Jakes, "Chapter 1: Multipath interference," *Microwave Mobile Communications. Ed. W. C. Jakes*, pp. 11–77, 1974, 1993, Wiley, New York.

[47] D. H. Johnson and D. E. Dudgeon, *Array Signal Processing: Concepts and Techniques*, Prentice Hall, Englewood Cliffs, NJ, 1993.

[48] Z. Kostic and S. Seetharaman, "Digital signal processors in cellular radio communications," *IEEE Communications Magazine*, vol. 35, no. 12, pp. 22–34, Dec. 1997.

[49] A. Dharap and M. Brandt-Pearce, "Transmitter-based multiuser interference rejection for synchronous CDMA systems in multipath environment," *Conference on Information Sciences and Systems*, March 1998, Princeton, NJ.

[50] K. Kota, *Parallel algorithms and architectures for subspace-based signal processing*, Ph.D. thesis, ECE Dept., Rice University, Houston, TX, 1996.

[51] J. Crols and M. Steyaert, *CMOS wireless transceiver design*, Kluwer Academic Publishers, Dordrecht, Netherlands, 1997.

[52] S. Sheng, L. Lynn, J. Peroulas, K. Stone, I. O'Donnell, and R. Brodersen, "A low power CMOS chipset for spread spectrum communications," *IEEE International Solid-State Circuits Conference*, pp. 346–347, 1996, San Francisco, CA.

[53] A. Burt, "3G applications challenge chip makers," *EE Times*, p. 112, March 30 1998, CMP Media Inc., Manhasset, NY, http://www.eet.com.

[54] J. Eyre and J. Bier, "DSP processors hit the mainstream," *IEEE Computer*, pp. 51–59, Aug. 1998.

[55] M. Schlett, "Trends in embedded microprocessor design," *IEEE Computer*, pp. 44–49, Aug. 1998.

[56] Lucent Technologies, *DSP16210 Digital Signal Processor*, Sept. 1997.

[57] Texas Instruments Incorporated, *TMS320C62X/C67X programmer's guide*, Feb. 1998.

[58] J. Li, B. Halder, P. Stoica, and M. Viberg, "Computationally efficient angle estimation for signals with known waveforms," *IEEE Trans. Signal Processing*, vol. 43, no. 9, pp. 2154–2163, Sept. 1995.

[59] X. Wang and H. V. Poor, "Blind adaptive interference suppression for CDMA communications based on eigenspace tracking," *Conference on Information Sciences and Systems*, pp. 468–473, Mar. 1997, Baltimore, MD.

[60] X. Wang and H. V. Poor, "Subspace-based blind adaptive joint interference suppression and channel estimation in multipath CDMA channels," *IEEE International Conference on Universal Personal Communications (ICUPC)*, pp. 460–464, Oct. 1997, San Diego, CA.

[61] C. Sengupta, S. Das, J. R. Cavallaro, and B. Aazhang, "Joint multiuser channel estimation and detection for CDMA systems," Tech. Rep. 9808, ECE Department, Rice University, MS 366, Houston, TX 77005, Nov. 1998.

[62] M. Juntti and B. Aazhang, "Linear Finite Memory-length Multiuser Detectors," in *Proc. of Communication Theory Miniconference at IEEE Global Telecommunication Conference, Singapore*, Nov. 1995, pp. 126–130.

[63] S. Das, J. R. Cavallaro, and B. Aazhang, "Computationally Efficient Multiuser Detectors," in *8th IEEE International Symposium on Personal,Indoor, and Mobile Radio Communications(PIMRC)*, 1997, pp. 62–67, Helsinki, Finland.

[64] G. H. Golub and C. F. Van Loan, *Matrix computations*, The Johns Hopkins University Press, Baltimore, MD, 2nd edition, 1989.

[65] Z-S Liu, J. Li, and S. L. Miller, "A receiver diversity based code-timing estimator for asynchronous DS-CDMA systems," *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 3245–3248, May 1998, Seattle, WA.

[66] Z-S Liu, J. Li, and S. L. Miller, "An efficient code-timing estimator for receiver diversity DS-CDMA systems," *IEEE Trans. Communications*, vol. 46, no. 6, pp. 826–835, June 1998.

[67] J. H. Wilkinson, *The algebraic eigenvalue problem*, Clarendon Press, Oxford, 1965.

[68] Texas Instruments Incorporated, *TMS320C5000 DSP family functional overview*, Sept. 1998.

[69] X. Wang and H. V. Poor, "Blind equalization and multiuser detection for high rate CDMA communications," *Sixth Communication Mini-Conference in conjunction with the Global Telecommunications Conference (GLOBECOM)*, pp. 113–117, Nov. 1997, Phoenix, AZ.

[70] M. Wax and T. Kailath, "Detection of signals by information theoretic criteria," *IEEE Trans. on Acoustics, Speech and Signal processing*, vol. ASSP-33, pp. 387–392, Apr. 1985.

[71] R. O. Schmidt, *A signal subspace approach to multiple emitter location and spectral estimation*, Ph.D. thesis, Stanford Univ., Stanford, CA, 1981.

[72] Y. Ding and R. J. Vaccaro, "Determination of data matrix dimensions for subspace-based parameter estimation algorithms," *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 2547–2550, May 1996, Atlanta, GA.

[73] F. Lorenzelli and K. Yao, "Jacobi SVD algorithms for tracking of nonstationary signals," *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, pp. 3183–3186, May 1995, Detroit, MI.

[74] M. Moonen, P. Van Dooren, and J. Vandewalle, "A systolic array for SVD updating," *SIAM Journal Matrix Anal. Appl.*, vol. 14, pp. 353–371, 1993.

[75] G. W. Stewart, "An updating algorithm for subspace tracking," *IEEE Trans. Signal Processing*, vol. 40, pp. 1535–1541, 1992.

[76] W. Ferzali and J. G. Proakis, "Adaptive SVD algorithm with application to narrowband signal tracking," *SVD and Signal Processing. Ed. R. Vaccaro*, vol. II, pp. 149–159, 1991, Elsevier, New York, NY.

[77] C. Sengupta, J. R. Cavallaro, and B. Aazhang, "Solving the SVD updating problem for subspace tracking on a fixed sized linear array of processors," *International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. 5, pp. 4137–4140, Apr 1997.

[78] A. Kavcic and B. Yang, "A new efficient subspace tracking algorithm based on SVD," *Proceedings of the International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*, vol. IV, pp. 485–488, April 1994, Adelaide, Australia.

[79] J. Gotze, "On the parallel implementation of Jacobi and Kogbetliantz algorithms," *SIAM Journal Sci. Stat. Comput.*, vol. 15, no. 6, pp. 1331–1348, 1994.

[80] M. R. Hestenes, "Inversion of matrices by biorthogonalization and related results," *Journal SIAM*, vol. 6, pp. 51–90, 1958.

[81] R. O. Schmidt, "Multiple emitter location and signal parameter estimation," *IEEE Trans. on Antennas and Propagation*, vol. AP-34, no. 3, pp. 276–279, Mar. 1986.

[82] M. Moonen, P. Van Dooren, and J. Vandewalle, "An SVD updating algorithm for subspace tracking," *SIAM Journal Matrix Anal. Appl.*, vol. 13, pp. 1015–1038, 1992.

[83] R. Schreiber, "Solving eigenvalue and singular value problems on an undersized systolic array," *SIAM Journal Sci. Stat. Comput.*, vol. 7, no. 2, pp. 441–451, Apr. 1986.

[84] Texas Instruments, *Application Guide, Parallel processing with the TMS320C4x*, 1994.

[85] Texas Instruments, *Technical Reference, TMS320C4x Parallel Processing Development System*, 1993.

[86] B. A. Curtis and V. K. Madisetti, "Rapid prototyping on the Georgia Tech digital signal multiprocessor," *IEEE Trans. Signal Processing*, vol. 42, no. 3, pp. 649–661, Mar. 1994.