TIME DEPENDENT ADAPTIVE FILTERS FOR
INTERFERENCE CANCELLATION IN CDMA SYSTEMS

by

Richard D. Holley

Thesis submitted to the Bradley Department of Electrical Engineering
and the Faculty of the Virginia Polytechnic Institute and State University
in partial fulfillment of the requirements for the degree of

Master of Science

in

Electrical Engineering

APPROVED:

Dr. Theodore S. Rappaport

Dr. Brian D. Woerner

Dr. Jeffrey H. Reed
(Principal Adviser)

January, 1994
Blackburn, Virginia
Abstract

Interference is a major problem in modern wireless communications systems. No longer are background noise and average power loss the limiting factors in system capacity — corruption of the available spectrum by multiple access and nearby interference provides the upper limit to system capacity. If the exponential growth of commercial wireless communications is to continue, systems must effectively deal with the increasingly crowded and corrupted spectrum.

Direct Sequence Spread Spectrum modulation (DS-SS) combined with Time Dependent Processing represents a valid approach to meeting the needs of future communications systems. Traditionally, the exploitation of cyclostationarity in digital communications signals has been reserved for the hostile communication environments faced by the military. However, the advent of cost-effective, high-speed DSP chips and associated processing hardware have made Time Dependent Processing a viable commercial technology.

This thesis presents several forms of the Time Dependent Adaptive Filter (TDAF) which are able to fully exploit the cyclostationarity and high degree of spectral correlation in certain DS-SS signals. It is shown that these optimal TDAFs are able to combat interference from noise, multipath, signals with dissimilar modulation, and signals with similar modulation (multiple access interference). Performance gains are achieved without a knowledge of the specific type of interference and depend solely on the high degree of spectral correlation in DS-SS signals. It is shown that properly designed DS-SS CDMA systems that utilize the TDAF can achieve spectral efficiencies which are within 10% of FDM/TDM systems. Furthermore, these systems retain the benefits of wideband modulation and universal frequency reuse traditionally associated with CDMA systems. The net result is a tremendous increase in system user capacity and signal reception quality.
© Copyright 1993
by
Richard D. Holley
Acknowledgements

To Laura, for her patience and understanding during the past several months.
To my mother and father, for their encouragement and support during this endeavor.
To Dr. Reed, for his insights, direction, and input in this thesis.
To my Creator, for the gifts which He has entrusted to me and His sustaining power.
Contents

Abstract ii

Acknowledgements iv

1 Introduction 1
   1.1 Sources of Interference in the Mobile Channel 2
   1.2 Current Techniques for Interference Rejection in CDMA Systems 3

2 Cyclostationarity in DS-SS 6
   2.1 Theoretical Background 6
   2.2 Spectral Correlation in DS-SS Modulation 8

3 Optimal Time Dependent Filters 20
   3.1 Optimal Time Dependent Filter Structures 20
   3.2 Training Sequence Adaptation Algorithms for TDAF Processing 26
      3.2.1 LMS TDAF Implementation 26
      3.2.2 Modified LMS TDAF Implementations 30
      3.2.3 RLS TDAF Implementations 32
   3.3 Blind Adaptation Algorithms for TDAF Processing 43
      3.3.1 The DFE-TDAF Algorithm 43
      3.3.2 The SDFE-TDAF Algorithm 46
      3.3.3 The EDFE-TDAF Algorithm 52

4 Simulated Performance of the TDAF for DS-SS 59
   4.1 Performance of the TDAF in Conventional Interference 63
      4.1.1 Interference from Background Noise 64
      4.1.2 Interference from Multipath Propagation 67
      4.1.3 Interference from Dissimilar Modulation 78
List of Tables

1 Circular RLS TDAF Algorithm — Version 1 ......................... 44
2 Circular RLS TDAF Algorithm — Version 2 ......................... 45
3 The DFE-TDAF Algorithm ............................................. 47
4 The SDFE-TDAF Algorithm ............................................ 51
5 The EDFE-TDAF Algorithm ............................................ 58
# List of Figures

<table>
<thead>
<tr>
<th>Figure</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>DS-SS Modulation</td>
<td>9</td>
</tr>
<tr>
<td>2</td>
<td>Fundamental Periodicities in DS-SS</td>
<td>10</td>
</tr>
<tr>
<td>3</td>
<td>Fundamental Periodicities in DS-SS, $\frac{S}{M} = 1$</td>
<td>12</td>
</tr>
<tr>
<td>4</td>
<td>Typical 32-Chip Spreading Waveform</td>
<td>15</td>
</tr>
<tr>
<td>5</td>
<td>Magnitude Spectrum of 32-Chip Spreading Waveform</td>
<td>16</td>
</tr>
<tr>
<td>6</td>
<td>Baseband Power Spectrum of Data, $d(t)$</td>
<td>17</td>
</tr>
<tr>
<td>7</td>
<td>Baseband Power Spectrum of Spreading Sequence, $c(t)$</td>
<td>18</td>
</tr>
<tr>
<td>8</td>
<td>Baseband Power Spectrum of DS-SS Signal, $\frac{M}{S} = 1$</td>
<td>19</td>
</tr>
<tr>
<td>9</td>
<td>Optimal Time Dependent Time Invariant Filter Structure</td>
<td>22</td>
</tr>
<tr>
<td>10</td>
<td>Optimal Time Dependent Adaptive Filter Structure</td>
<td>23</td>
</tr>
<tr>
<td>11</td>
<td>FFT Based Optimal Time Dependent Adaptive Filter Structure</td>
<td>25</td>
</tr>
<tr>
<td>12</td>
<td>FFT Based Time Independent Adaptive Filter Structure</td>
<td>27</td>
</tr>
<tr>
<td>13</td>
<td>Constellation of Ideal BPSK at Baseband</td>
<td>49</td>
</tr>
<tr>
<td>14</td>
<td>Scale Factor, $\Psi_k$, for BPSK Baseband Modulation</td>
<td>50</td>
</tr>
<tr>
<td>15</td>
<td>Scale Factor for EDFE-TDAF ($p = 0.20$)</td>
<td>53</td>
</tr>
<tr>
<td>16</td>
<td>Comparison of Scale Factors for EDFE-TDAF ($p = 0.01, 0.20$)</td>
<td>54</td>
</tr>
<tr>
<td>17</td>
<td>Scale Factor $\Psi_k$ versus the BPSK Constellation Plane</td>
<td>55</td>
</tr>
<tr>
<td>18</td>
<td>Scale Factor $\Psi_k$ versus the QPSK Constellation Plane</td>
<td>56</td>
</tr>
<tr>
<td>19</td>
<td>Comparison of Scale Factors for SDFE-TDAF and EDFE-TDAF</td>
<td>57</td>
</tr>
<tr>
<td>20</td>
<td>Addition of DFE-TDAF to Typical CDMA Base Station</td>
<td>61</td>
</tr>
<tr>
<td>21</td>
<td>MSE for Pre-Despread DS-SS Signal after TDAF vs. Received SWNR</td>
<td>65</td>
</tr>
<tr>
<td>22</td>
<td>Pre-Despread SNR Before and After TDAF Processing of DS-SS Signal</td>
<td>66</td>
</tr>
<tr>
<td>23</td>
<td>Convergence Rate of RLS-TDAF for DS-SS</td>
<td>68</td>
</tr>
<tr>
<td>24</td>
<td>Power Spectrum of DS-SS Signal Corrupted by Urban Multipath</td>
<td>70</td>
</tr>
<tr>
<td>25</td>
<td>Constellation of Pre-Despread DS-SS Signal Corrupted by Urban Multipath</td>
<td>71</td>
</tr>
<tr>
<td>26</td>
<td>Power Spectrum of DS-SS Signal After TDAF Processing</td>
<td>72</td>
</tr>
</tbody>
</table>
27 Constellation of Pre-Despread DS-SS Signal Corrupted by Urban Multipath
After TDAF Processing ........................................ 73
28 Power Spectrum of DS-SS Signal Corrupted by Suburban Multipath ...... 74
29 Constellation of Pre-Despread DS-SS Signal Corrupted by Suburban Multipath 75
30 Power Spectrum of DS-SS Signal Corrupted by Suburban Multipath after TDAF Processing ........................................ 76
31 Constellation of Pre-Despread DS-SS Signal Corrupted by Suburban Multi-
path After TDAF Processing ........................................ 77
32 Power Spectrum of Uncorrupted DS-SS Signal ........................................ 79
33 Power Spectrum of Environment (DS-SS and Noise) ......................... 80
34 Power Spectrum of DS-SS Signal with 20 FM-AMPS Interferers (SWNR = 0dB) ........................................ 81
35 Constellation Diagram for M.F. Output without TDAF Processing ........ 82
36 Constellation Diagram for M.F. Output after TDAF Processing .......... 83
37 Output Power Spectrum after TDAF Processing ........................................ 84
38 Power Spectrum after Spectral Notching ........................................ 85
39 Constellation Diagram for M.F. Output with Spectral Notching .......... 86
40 Constellation Diagram for M.F. Output with Spectral Notching and TDAF Processing ........................................ 87
41 MSE for Conventional Processing (SWNR = 0,10,20,30dB) .................. 89
42 MSE for DFE-TDAF Processing (SWNR = 0,10,20,30dB) .................. 90
43 Theoretical Performance of Conventional Correlation/Matched Filter for CDMA 91
44 BER for Conventional Processing (SWNR = 0,10,20,30dB) .................. 92
45 BER for DFE-TDAF Processing (SWNR = 0,10,20,30dB) .................. 93
46 BER for Worst User in Each Group, Conventional (SWNR = 0,10,20,30dB) 94
47 BER for Worst User in Each Group, DFE-TDAF (SWNR = 0,10,20,30dB) 95
48 Average BER for Each User (SWNR = 30dB, # Users = 64) .................. 96
49 DFE-TDAF Catastrophic Failure Rate (SWNR = 0,10,20,30dB) ........ 97
50 Constellation Diagrams for All 48 Users (Overlaid) — Conventional Processing 98
51 Constellation Diagrams for All 48 Users (Overlaid) — TDAF Processing . 99
52 Constellation Diagrams for All 48 Users (Overlaid) — TDAF Processing . 100
53 Constellation Diagrams for 47 of 48 Users (Overlaid) — TDAF Processing . 101
54 Constellation Diagrams for 18 of 48 Users (Overlaid) — TDAF Processing . 102
55 MSE for Conventional Processing ........................................ 103
56 MSE for SDPE-TDAF Processing ........................................ 104
Chapter 1

Introduction

For many years sophisticated interference cancellation and mitigation techniques have been reserved for military communications and other hostile communication environments. Traditionally, governing bodies such as the Federal Communications Commission have allocated specific frequency bands for commercial use. This allocation has served to minimize the number of inter-system and intra-system interference sources. However, the current lack of available RF spectrum combined with modern multiple-access techniques has created the need for effective interference rejection techniques in commercial systems. For many systems, interference rejection techniques represent the only viable solution for increasing user capacity.

Code Division Multiple Access (CDMA) systems present a challenging environment for interference mitigation. In addition to the traditional sources of interference such as noise and channel distortion, CDMA systems suffer from the interference of many users transmitting a similar modulation format within the same frequency band. This *multiple access interference* severely limits the user capacity and spectral efficiency of CDMA systems. Additionally, certain CDMA “overlay” systems which have recently been proposed [SLG93] must combat interference from dissimilar modulation formats (FM-AMPS) as well as similar modulation formats. These systems rely primarily on the processing gain of the spread spectrum modulation to overcome the many sources of interference; thus, user capacity becomes a direct function of the available processing gain.

Several techniques have been presented in recent years to combat interference in CDMA systems [Ver84, Ver86, Ver89, VA88, VA90, Age93, MPG92, Koh91, KKK+93]. These techniques are primarily directed towards the cancellation of multiple access interference in CDMA systems. The most common approach consists of a joint estimation and subtraction process which relies heavily on knowledge of the other users in the system. The *Time
Dependent Adaptive Filter (TDAF), when applied to CDMA systems, represents a distinct change from recent CDMA interference cancellation techniques. First, time dependent filtering is able to mitigate many different forms of interference in the mobile channel — not just multiple access interference. Second, time-dependent filtering requires no knowledge of the other users in the system in order to produce marked increases in user capacity.

This thesis presents several time dependent filtering algorithms which have been optimized for interference mitigation in CDMA systems. The performance of the most suitable time-dependent filtering algorithms is determined for several types of interference. The results show that the TDAF can provide impressive interference cancellation and mitigation for properly designed CDMA systems.

1.1 Sources of Interference in the Mobile Channel

Wireless communications which utilize CDMA must deal with many sources of potential interference. First, mean transmitted power levels must be adjusted to provide an adequate signal to white noise ratio (SWNR) at both the base station and mobile receiver. Traditionally such background noise has been characterized as Gaussian or thermal noise. While most narrowband communications systems are able to ignore background noise, wideband CDMA systems are often designed to operate at pre-despread SWNR's of 0dB or less [oDCC92]. Thus, any effective interference mitigation technique must be able to minimize the interference from background noise.

Second, the multipath nature of the mobile channel is highly time-varying and extremely dependent on the surrounding topography. Thus, any successful interference rejection technique must be able to compensate for the multipath nature of the channel. In general, this forces useful interference rejection techniques to be adaptive and able to accurately track the changing environment of the mobile user.

Third, effective interference rejection techniques must be able to minimize the effects of multiple access interference. This problem represents one of the most challenging in wireless CDMA systems. CDMA is a natural modulation choice for the mobile channel. It's wide bandwidth helps to combat frequency selective fading — a problem with most every narrowband wireless system. Furthermore, the universal frequency reuse of CDMA systems minimizes the need for frequency reuse planning and increases the user capacity of cellular systems. However, these benefits are not without a price. The overlaying of many users in the same spectrum eliminates the utility of traditional time-independent filters which remove energy from all users. Additionally, while the properties of a single user in a CDMA system can often be accurately determined, it is extremely difficult to determine the
relationship between each user in a CDMA system to every other user. Thus, interference rejection techniques for CDMA systems must be able to accommodate time-varying user loads and differing received power levels.

Finally, there exist several forms of interference which may be broadly classified as interference from dissimilar modulation. These forms of interference include narrowband signals (such as FM-AMPS and TDMA signals), terrestrial microwave links, paging and data relay services, and signals from unlicensed users. For many CDMA systems, these sources of interference are minimal due to FCC frequency allocations; however, there exist proposals which attempt to “overlay” CDMA signals to the existing FM-AMPS cellular standard [SLG93]. In these cases, the processing gain of the spread spectrum modulation is insufficient to combat the high level of interference from conventional users. Thus, some form of interference rejection must be employed.

While many other forms of interference exist in wireless CDMA systems, these four represent the major categories of concern. To effectively combat all four categories of interference is to create a system with enormous user capacity and reception quality. A brief review of available techniques reveals that no single technique currently exists for combating all forms of interference.

1.2 Current Techniques for Interference Rejection in CDMA Systems

Many techniques have been developed in the past several years for combating interference in CDMA systems. Of particular interest to this thesis are techniques which operate on the signal after reception. Recently, researchers [LR93, Age93] have shown that significant interference rejection can be achieved through the use of adaptive antenna arrays. Other researchers have shown that modifications to traditional transmission and receiver structures can significantly improve the performance of CDMA systems [MI86]. However, in most instances adaptive array techniques only improve the performance of techniques which operate on the receiver output. Thus, a baseline comparison of post-receiver processing techniques is sufficient, since array processing will tend to enhance the performance of all post-receiver techniques (although, perhaps, to different degrees).

The rejection of background noise has been achieved in most commercial CDMA systems. Qualcomm Inc. and Interdigital Inc. have developed robust open and closed loop power control schemes which control the transmitted power at both the mobile and base transmitters [oDCC92] [SLG93]. This intricate power control was originally developed to
combat the "near-far" problem of CDMA systems, but has the side benefit of maintaining a near-constant received SWNR across users. Thus, the primary technique for combating background noise is the effective management of transmitter power levels.

In order to combat multipath interference, Qualcomm has chosen to utilize a RAKE receiver structure [oDCC92]. The basic concept of the RAKE receiver is well understood and is generally credited to the foundational work of Price and Green [PJ58]. Modern advances in digital signal processing (DSP) coupled with exponential increases in computer speeds have made the RAKE receiver a cost effective structure. The RAKE receiver actually utilizes multipath interference to form an improved estimate of the spread spectrum signal that was transmitted [Tur80]. The RAKE receiver utilized by Qualcomm identifies up to three delayed signal components in the channel, time-aligns the primary signal and the delayed (multipath) versions of the signal, and combines the three components to achieve an improved signal to noise ratio (SNR) and signal to interference ratio (SIR) [oDCC92]. Thus, the RAKE receiver utilizes the inherent time diversity of the mobile channel to mitigate channel interference. Unfortunately, the performance of the RAKE receiver is limited by the time resolution of the chip interval. For Qualcomm's system this limitation means that the RAKE receiver cannot improve performance for channels in which the average multipath delay is shorter than roughly one-quarter microsecond [Li93]. The system proposed by Interdigital, however, incorporates a 10MHz chip rate, and is theoretically able to exploit the shorter path delays (i.e. hundreds of nanoseconds) which are representative of most highly urban and in-building environments. However, a RAKE receiver structure is not currently specified for the Interdigital system.

In the past few years, several interesting techniques for combating multiple access interference have been developed. Of particular interest is the work of Verdu in the determination of the optimal receiver for multiple access interference in CDMA systems [Ver84, Ver86]. Verdu and other researchers have shown that the traditional correlation/matched filter receiver used in spread spectrum receivers is optimal only against Gaussian noise. For multiple access interference, the correlation/matched filter receiver is suboptimal. The optimal receiver used by Verdu views the received CDMA signal as a combination of symbol states in a dynamic constellation. The optimal receiver is therefore similar to a Viterbi decoder. Unfortunately, the complexity of this optimal receiver grows exponentially with the number of users [Ver89]. For systems with more than eight users, the required computation count becomes impractical, and for most real-world systems the required computation count cannot be met with current technology.
CHAPTER 1. INTRODUCTION

In light of the computational complexity of the optimal CDMA receiver, several researchers have developed suboptimal structures for dealing with multiple access interference in CDMA systems. Varanasi and Aazhang [VA88, VA90] have developed a multi-stage estimation and subtraction technique which attempts to minimize multiple access interference. The receiver establishes the time-displacement and relative received power levels between the signals for all users, forms an estimate of the data for the strongest users, and iteratively removes the strongest signals to improve performance. Other researchers have developed similar structures [MPG92, Koh91, KKK+93], but each suffers from two major problems. First, the structures assume that the time-displacements and relative power levels between all users of the system can be established. Second, most of the structures require a relatively high degree of power variance among users to be effective. For systems in which the power levels between users are well balanced, the determination of the strongest users and their corresponding data becomes difficult. This, in turn, leads to incorrect estimates of the data from the stronger users, which results in a doubling of the undesired signal power in subsequent stages. Thus, the utility of the multi-stage detector in real-world systems remains to be seen.

A promising area of future research is the use of neural networks for multiple access interference mitigation in CDMA systems. Recent simulations have shown that certain neural network structures can approximate the behavior of the optimal CDMA receiver without the exponential increase in complexity [MP93, APO92, Aue94]. In many respects, the entire field of neural networks is still in its infancy, especially in applications for communications systems. Thus, the degree of foundational work and proven feasibility required by most commercial systems is not currently available.

The exploitation of spectral correlation and cyclostationarity represents a promising arena for interference cancellation in CDMA systems. Agee has shown that CDMA receivers which exploit the spectral correlation in spread spectrum signals can achieve multiple access efficiencies approaching that of FDM/TDM systems [Age93]. In this thesis, it is shown that the exploitation of spectral correlation can add orders of magnitude of improvement to traditional CDMA architectures. Furthermore, the fundamental theory behind cyclostationary processing is well understood, and the algorithms form the basis of many commercial and military products. Based on these facts, it seems only reasonable that cyclostationary processing, as applied to spread spectrum signals, represents one of the best possibilities for interference mitigation in modern CDMA systems.
Chapter 2

Cyclostationarity in Direct Sequence Spread Spectrum

2.1 Theoretical Background

In order to realize any benefit from time dependent filtering, either the desired signal(s) or interfering signal(s) must exhibit the property of cyclostationarity. Cyclostationarity is a term which is used to describe the repetitive or cyclic nature of the statistics associated with digital communications signals. The traditional autocorrelation of a stochastic process \( x(t) \) is given by

\[
R_{xx}(t_1, t_2) = \mathbb{E} (x(t_1)x^*(t_2))
\]  

(1)

where \( \mathbb{E}(\bullet) \) denotes the expectation operator. If the autocorrelation function can be rewritten strictly as a function of the time difference, \( \tau = t_1 - t_2 \), then the process is insensitive to the specific instance of sampling, and the autocorrelation function may be rewritten as

\[
R_{xx}(\tau) = \mathbb{E} \left( x \left( t + \frac{\tau}{2} \right) x^* \left( t - \frac{\tau}{2} \right) \right)
\]  

(2)

For such processes, the signal statistics are often described as stationary. Suppose, however, that the autocorrelation is a periodic function of both the sample instant and the time difference such that

\[
R_{xx}(t, \tau) = \mathbb{E} \left( x \left( t + \frac{\tau}{2} \right) x^* \left( t - \frac{\tau}{2} \right) \right)
\]  

(3)

and

\[
R_{xx}(t, \tau) = R_{xx}(t + T_0, \tau)
\]  

(4)

For this case, the process is best described as cyclostationary with fundamental periodicity \( T_0 \) [Gar85b, Gar87a, GIC87]. For most digital modulation schemes, \( T_0 \) represents the period
of a baud or half the period of the carrier [Gar87b, GIC87, Gar87a]. It is useful to define
the set of cycle harmonics which are integrally related to the fundamental periodicity, $T_0$,
as
$$\alpha = \frac{n}{T_0}$$
(5)
where $n \in \{-l, -l+1, \ldots, 0, \ldots, u-1, u\}$ is an integer number. Using this notation, the
coefficients for the harmonic components of the cyclic autocorrelation function are given by
[Gar85b]
$$R_{xx}^\alpha(\tau) = \frac{1}{T_0} \int_{-\frac{T_0}{2}}^{\frac{T_0}{2}} R_{xx}(t, \tau) e^{-j2\pi \alpha t} dt$$
(6)
Furthermore, if the range of $n$ represents the complete set of harmonics which are integrally
related to the fundamental periodicity (i.e. $n \in \{-\infty, \infty\}$), then the total autocorrelation
may be expressed in terms of it's Fourier series expansion
$$R_{xx}(t, \tau) = \sum_{n=-\infty}^{\infty} R_{xx}^\alpha(\tau) e^{j2\pi \alpha t}$$
(7)
For many signals, there exist multiple fundamental periodicities or periodicities which are
non-harmonically related [Gar87a]. Such signals are referred to as almost cyclostationary,
and the total autocorrelation can be expressed as the sum of the Fourier components from
all fundamental periodicities or cycle frequencies
$$R_{xx}(t, \tau) = \sum_{\alpha \in \vec{\alpha}} R_{xx}^\alpha(\tau) e^{j2\pi \alpha t}$$
(8)
In this notation, $\vec{\alpha}$ represents the vector set of fundamental cycle frequencies present in the
signal,
$$\vec{\alpha} \in \{\alpha_0, \alpha_1, \ldots, \alpha_M\}$$
(9)
which correspond to the set of fundamental periodicities
$$T_i \in \{T_0, T_1, \ldots, T_M\}$$
(10)
The cyclic autocorrelation coefficients in Equation 8 are therefore given by
$$R_{xx}^\alpha(\tau) = \lim_{T \to \infty} \frac{1}{T} \int_{-T}^{T} R_{xx}(t, \tau) e^{-j2\pi \alpha t} dt$$
(11)
In similar fashion to traditional spectral analysis, the cyclic autocorrelation is useful
in the determination of the cyclic spectrum. Taking the Fourier transform of the cyclic
autocorrelation in Equation 11 yields the cyclic spectral density
$$S_{xx}^\alpha(f) = \int_{-\infty}^{\infty} R_{xx}^\alpha(\tau) e^{-j2\pi f \tau} d\tau$$
(12)
Similarly, the time dependent spectral density is given in [Gar87a] as

$$S_{xx}(t, f) = \int_{-\infty}^{\infty} R_{xx}(t, \tau)e^{-j2\pi ft}d\tau$$  \hspace{1cm} (13)

These quantities provide a measure of the spectral correlation between frequency components in a signal. In particular, the cyclic spectral density, $S_{xx}^\alpha(f)$, provides a measure of the correlation between frequency components at $f - \frac{\alpha}{2}$ and $f + \frac{\alpha}{2}$. This fact is most clearly seen by taking the Fourier transform of a finite window of data,

$$X_W(t, f) = \int_{t-\frac{W}{2}}^{t+\frac{W}{2}} x(u)e^{-j2\pi fu}du$$  \hspace{1cm} (14)

and using the linearity property of the Fourier transform to rewrite Equation 12 as

$$S_{xx}^\alpha(f) = \lim_{W \to \infty} \lim_{T \to \infty} \frac{1}{T} \int_{-\frac{T}{2}}^{\frac{T}{2}} \frac{1}{W} E\left\{X_W\left(t, f + \frac{\alpha}{2}\right)X_W^*\left(t, f - \frac{\alpha}{2}\right)\right\} dt$$  \hspace{1cm} (15)

In this form, it is clear that the cyclic spectral density provides a clear measure of both the location and degree of spectral correlation in a cyclostationary process. Thus, the task at hand is to identify and quantize the degree of spectral correlation present in DS-SS signals so that such correlation may be exploited.

### 2.2 Spectral Correlation in DS-SS Modulation

In its most basic form, direct sequence spread spectrum can be viewed as the multiplication of a low-rate cyclostationary process (the input data signal, $d(t)$) by a high-rate cyclostationary process (the spreading or chip signal, $c(t)$). This is clearly seen in Figure 1. Intuitively, it is reasonable to assume that the fundamental periodicities present in the transmitted DS-SS signal are the result of some combination of the periodicities present in the generating signals, $d(t)$ and $c(t)$. In his dissertation on spectral correlation, Chen [Che88] has shown that there exist three fundamental periodicities in a direct sequence spread spectrum signal:

- the fundamental chip rate periodicity, $T_{chip}$,
- the fundamental data rate periodicity, $T_{data}$,
- and the fundamental code repetition rate periodicity, $T_{code}$. 

These periodicities result from the individual fundamental periodicities in the data and spreading signals, \( d(t) \) and \( c(t) \). Their relative periods are shown in Figure 2. The proof of the existence of these periodicities is involved, and is beyond the scope of this thesis; however, the results derived by Chen in [Che88] provide the necessary fundamental framework to establish the locations of spectral correlation in DS-SS signals.

A high degree of spectral correlation can be achieved in a DS-SS signal by carefully choosing the three fundamental periodicities. Traditionally, the data rate periodicity is chosen as an integer multiple of the chip rate periodicity. Although not required, this choice insures minimal spectral “splashing” and maximum in-band power utilization [Dix84]. Such a “choice” results naturally in systems where the baud clock for \( d(t) \) is derived from the chip rate clock.

Similarly, the code repetition rate can be designed to provide a high degree of spectral correlation. The utility of the code rate periodicity has often been overlooked in the design of traditional spread spectrum systems. In fact, many systems attempt to eliminate this periodicity through the use of non-repetitive spreading sequences (i.e. military systems) or spreading sequences with extremely long repetition rates [oDCC92]. However, if the code repetition period is designed to be an integer multiple of the chip rate periodicity, then the degree of spectral correlation is greatly enhanced.

\[1\text{Spectral “splashing” occurs in DS-SS signals which have data rates and chip rates that are not integrally related. For such systems, there exist instances in which the multiplication of the data sequence by the chip sequence produces narrow, impulsive components whose spectrum “splashes” or occupies excessive bandwidth.}\]
Figure 2: Fundamental Periodicities in Direct Sequence Spread Spectrum
CHAPTER 2. CYCLOSTATIONARITY IN DS-SS

When both the data and code repetition rates are designed to be integrally related to the chip rate, then the number of fundamental periodicities in the DS-SS signal is reduced to one. Specifically, if $S$ is defined as the number of chips per baud and $M$ is defined as the number of chips per code sequence repetition, then the fundamental cycle frequencies become functions of a single fundamental periodicity, $T_{\text{chip}}$.

\[
\begin{align*}
\alpha_{\text{chip}} &= \frac{1}{T_{\text{chip}}} \\
\alpha_{\text{data}} &= \frac{1}{T_{\text{data}}} = \frac{1}{ST_{\text{chip}}} \\
\alpha_{\text{code}} &= \frac{1}{T_{\text{code}}} = \frac{1}{MT_{\text{chip}}} 
\end{align*}
\] (16)

In traditional notation, the parameter $S$ is known as the spreading or processing gain of the DS-SS signal.

In order to best exploit spectral correlation in DS-SS, the data rate and code repetition rate must be integrally related — i.e. either the ratio $\frac{S}{M}$ or the ratio $\frac{M}{S}$ must be an integer [Che88]. An example which satisfies this criterion is pictured in Figure 3 for $\frac{S}{M} = 1$. This form of DS-SS is commonly referred to as “code on pulse” or “modulation on symbol” [Age93]. In order to identify the locations of spectral correlation in DS-SS signals, Chen [Che88] has derived expressions for the cyclic spectral density of signals employing DS-SS modulation.

The basic mathematical description of DS-SS modulation is simply

\[x(t) = c(t)d(t)\] (17)

where $d(t)$ represents the input data signal, $c(t)$ represents the spreading waveform, and $x(t)$ represents the DS-SS signal. For simplicity, it is assumed that the spreading waveform consists of real rectangular pulses described by

\[q(t) = \begin{cases} 
1 & -\frac{1}{2} \leq t \leq \frac{1}{2} \\
0 & \text{otherwise}
\end{cases}\] (18)

which are modulated or scaled by a binary sequence, $b_m \in \{-1, 1\}$. Thus, the spreading waveform can be written as

\[c(t) = \sum_{m=-\infty}^{\infty} b_m q\left(\frac{t}{T_{\text{chip}}} - m\right)\] (19)

In order to maximize the degree of spectral correlation and to insure complete usage of the available spectrum, the ratio of the code repetition period to the period of a data baud is chosen to be an integer ($\frac{M}{S} =$ integer). If the inverse ratio is an integer, then there may exist spectral “gaps” in the DS-SS signal due to the high code repetition rate. It is
Figure 3: Fundamental Periodicities in DS-SS, $\frac{\tilde{M}}{M} = 1$
evident from Equation 16 that decreasing the fundamental period of the code repetition rate, \( T_{code} = MT_{chip} \), is tantamount to increasing the fundamental code repetition cycle frequency, \( \alpha_{code} \). Thus, it is assumed that the code repetition period is at least as long as a data baud and perhaps some integer multiple, \( N \), of a data baud period in length. Under this condition, the binary spreading sequence will repeat with period

\[
T_{code} = M \cdot T_{chip} = NS \cdot T_{chip}
\]

such that

\[
b_{m+NS} = b_m
\]

With this stipulation, Chen [Che88] has derived the expression for the cyclic autocorrelation of \( x(t) \) as

\[
R_x^\alpha(\tau) = \sum_{n,m=-\infty}^{\infty} C_n C_m^* R_{dd}^{\alpha, (n-m)/NST_{chip}}(\tau) e^{-j \pi(n+m)/2NST_{chip}}
\]

In this equation, \( C_n \) represents a scaled version of the Fourier coefficients to the chip sequence waveform:

\[
C_n = \frac{1}{NST_{chip}} \int_{-NST_{chip}/2}^{NST_{chip}/2} c(t) e^{-j \frac{2\pi n t}{NST_{chip}}} dt
\]

\[
= \frac{1}{NST_{chip}} \int_{-NST_{chip}/2}^{NST_{chip}/2} b_m q \left( \frac{t}{T_{chip}} - m \right) e^{-j \frac{2\pi n t}{NST_{chip}}} dt
\]

\[
= B_n \frac{\sin \left( \frac{\pi n}{NS} \right)}{\pi n}
\]

where

\[
B_n = \sum_{m=0}^{NS-1} b_m e^{-j \frac{2\pi nm}{NS}}
\]

Taking the Fourier transform of Equation 22 gives the desired expression for the cyclic spectral density of DS-SS modulation

\[
S_x^\alpha(f) = \sum_{n,m=-\infty}^{\infty} C_n C_m^* S_{dd}^{\alpha, (n-m)/NST_{chip}} \left( f + \frac{n + m}{2NST_{chip}} \right)
\]

\[
= \sum_{n,i=-\infty}^{\infty} C_n C_m^* S_{dd}^{\alpha, \frac{f}{NST_{chip}}} \left( f + \frac{n}{NST_{chip}} - \frac{l}{2NST_{chip}} \right)
\]

In this equation, \( S_{dd}^{\beta}(f) \) represents the cyclic spectral density of the baseband message signal, \( d(t) \). Since it is assumed that polar binary signalling is used for \( d(t) \), the baseband
CHAPTER 2. CYCLOSTATIONARITY IN DS-SS

message modulation format is simply *binary phase shift keying* (BPSK). If the conjugate spectral symmetry of BPSK is ignored, then the fundamental periodicity of \( d(t) \) is the baud rate, \( T_{\text{data}} \) [GIC87, Gar87a]. Thus, the useful set of cycle frequencies for the data signal consists of

\[
\beta = \frac{m}{T_{\text{data}}} = \frac{m}{ST_{\text{chip}}} \quad m \in \{\text{integer}\} \tag{29}
\]

From Equations 29, 27 it is apparent that \( S_{xx}^\alpha(f) \) is non-zero only when \( \alpha \) satisfies

\[
\alpha = \frac{l}{NST_{\text{chip}}} + \frac{m}{ST_{\text{chip}}} \quad l, m \in \{\text{integer}\} \tag{30}
\]

or

\[
\alpha = \frac{l + mN}{NST_{\text{chip}}} = \frac{p}{NST_{\text{chip}}} \quad p \in \{\text{integer}\} \tag{31}
\]

Thus, for DS-SS modulation where the code repetition rate is integrally related to the data rate (and both are functions of the chip rate), there exists a non-zero degree of spectral correlation. Specifically, if the code repetition rate is equal to the data baud rate (modulation on symbol), then \( N = 1 \) and the fundamental cycle frequencies are fully represented by the integer harmonics of the fundamental code rate cycle frequency (or alternatively the fundamental data rate cycle frequency).

Technically, the fundamental code repetition cycle frequency \( \alpha_{\text{code}} \) is a function of the fundamental chip cycle frequency \( \alpha_{\text{chip}} \). (That is to say the code repetition cycle frequency represents a sub-harmonic of the chip repetition cycle frequency.) This result is of limited practical use, however, since the chip rate periodicities produce spectral correlation between sidelobes of a DS-SS signal. In most real-world systems, the 1st null-to-null bandwidth of the wideband DS-SS signal is the only portion of the signal which is transmitted. Sidelobes of the DS-SS signal are routinely filtered before transmission to improve the spectral efficiency. Therefore, the fundamental cycle frequencies of interest consist of the data rate and code repetition rate cycle frequencies only.

These results are of extreme importance, for it opens the door to many unconventional design techniques for DS-SS systems. The traditional discussion of DS-SS modulation views the spreading process as a time domain procedure where the low-rate data sequence, \( d(t) \), is multiplied by a high-rate chip sequence, \( c(t) \), to produce a a high-rate (i.e. wideband) modulation. Figure 4 depicts several cycles of a simple 32-chip binary spreading waveform, \( c(t) \). For this example, the code repetition rate is \( T_{\text{code}} = MT_{\text{chip}} = 32 \cdot T_{\text{chip}} \). Since the code repeats in a periodic fashion, the spreading waveform may be described by the time domain representations of Equations 19 and 21. However, the repetitive nature of the spreading code implies that a Fourier series representation is also possible, where the Fourier series coefficients are given by Equation 26 with \( M = NS = 32 \). Figure 5 shows
Figure 4: Several Unmodulated Cycles of a Typical 32-Chip Spreading Waveform
Figure 5: Magnitude Spectrum (Main Lobe) of 32-Chip Spreading Waveform

the magnitude spectrum of the periodic spreading waveform. Notice that the periodic spreading waveform can be viewed in the frequency domain as a series of harmonically related sinusoidal tones. The spacing of the tones corresponds to the fundamental code repetition rate cycle frequency, $\alpha_{code}$, and the magnitudes or weights of the tones are given by the Fourier coefficients of the waveform.

These observations lead to an interesting interpretation of DS-SS modulation. Let Figure 6 represent the power spectrum for a BPSK signal with relatively low data rate. Figure 7 shows the power spectrum of a spreading waveform which has a high code repetition rate ($\frac{N}{S} = 1$). Basically, one can view DS-SS modulation in the frequency domain as a translation of the baseband data signal to multiple carriers. The spacing of the carriers is determined by the fundamental code repetition cycle frequency, and the magnitudes of the carriers are determined by the Fourier coefficients of the spreading waveform. Thus, the transmitted signal appears as Figure 8. Demodulation of DS-SS becomes the retranslation of the message signal from multiple locations in the spectrum to baseband. For demodulation, the phase of each carrier is given as the conjugate of the translating carrier, while the
Figure 6: Baseband Power Spectrum of Data, $d(t)$

magnitude of each carrier remains the same.

Thus, it is seen that DS-SS modulation exhibits a high degree of spectral correlation, provided the data rate, code repetition rate, and chip rate are integrally related. Furthermore, the degree of spectral correlation can be enhanced by choosing the code repetition rate to be an integer multiple of the data rate. These facts allow DS-SS systems to be intentionally designed for cyclostationary exploitation by *time dependent adaptive filters*. 
Figure 7: Baseband Power Spectrum of Spreading Sequence, $c(t)$
Figure 8: Baseband Power Spectrum of DS-SS Signal, $\frac{M}{S} = 1$
Chapter 3

Optimal Time Dependent Filters for CDMA Interference Rejection

Chapter 2 presented justification for the presence and location of spectral correlation in DS-SS signals. For a properly designed data rate, code repetition rate, and chip rate, DS-SS signals exhibit a high degree of statistical cyclostationarity. The goal is to develop structures which can exploit this property of cyclostationarity in order to mitigate the effects of undesired interference.

3.1 Optimal Time Dependent Filter Structures

J. H. Reed [Ree87, Ree90] and W. A. Gardner [Gar93] have developed optimal filtering structures which exploit the spectral correlation of signals having cyclostationary statistics. The basic approach of these structures is to convert a single cyclostationary process into a collection of spectrally correlated processes. Optimal filtering is then accomplished through the optimal combining of the spectrally correlated processes. Reed [Ree87] has developed optimal filter structures based on the Time Series Representation (TSR) and Fourier Series Representation (FSR) of a cyclostationary process.

The TSR of a cyclostationary process has certain benefits over the FSR when large numbers of fundamental periodicities are present in the signal. Optimal filtering structures based on the TSR exploit all of the available periodicities and require no frequency domain transformations. Unfortunately, optimal filtering structures based on the TSR generally require large numbers of synchronous samplers and phase locked circuits. For wideband signals and high-rate processes (such as DS-SS), the timing constraints required for filter structures based on the TSR are difficult to achieve. For this reason, optimal filter structures
CHAPTER 3. OPTIMAL TIME DEPENDENT FILTERS

based on the TSR will not be presented. The interested reader is referred to the dissertation by Reed [Ree87].

The basic idea of the Optimal Time Dependent Filter (OTDF) is to minimize the time-averaged difference between a desired frequency component and a linear combination of correlated frequency components. By filtering correlated frequency components of the corrupted input signal and combining these components appropriately, a better estimate of the uncorrupted signal can be obtained.

The derivation of the OTDF begins with a finite window Fourier transform of the input data \( X_T(t, f) \) [Ree87],

\[
X_T(t, f) = \int_{t-T}^{t+T} x(\tau)e^{-j2\pi f\tau} d\tau,
\]

(32)

where \( T \) is the window length. Assuming that an uncorrupted version of the transmitted signal, \( Y_T(t, f) \), is available (i.e. — a training or desired signal), the objective is to determine the filters \( G_n(f) \) which minimize the mean square error (MSE):

\[
\text{minimize} \quad \text{MSE} = \lim_{T \to \infty} \frac{1}{T} \int_{-T/2}^{T/2} \left[ Y_T(t, f) - \sum_{n=0}^{M-1} G_n(f)X_T(t, f - \alpha_n) \right]^2 dt
\]

(33)

In this equation, \( \alpha_n \) represents the range of fundamental cycle frequencies available to the filter. Equation 33 is satisfied when the derivative with respect to \( G_n(f) \) is set equal to 0:

\[
0 = \lim_{T \to \infty} \frac{1}{T} \int_{-T/2}^{T/2} \left[ Y_T(t, f) - \sum_{n=0}^{M-1} G_n(f)X_T(t, f - \alpha_n) \right] X_T^*(t, f - \beta) dt
\]

(34)

When \( T \) is allowed to become large, the solution to the above equation is [Ree87]

\[
\sum_{n=0}^{M-1} G_n(f)\hat{S}_zx^{\alpha_n} \left( f - \frac{\alpha_n + \beta}{2} \right) = \hat{S}_y^{\beta} \left( f - \frac{\beta}{2} \right)
\]

(35)

where \( \hat{S}_zx^{\alpha_n}(f) \) and \( \hat{S}_y^{\beta}(f) \) represent finite time approximations of their respective cyclic spectral densities. It is clear from Equation 35 that the OTDF consists of a bank of filters \( G_n(f) \), each of which filters a frequency shifted version of the input signal. The output of the OTDF is the linear combination of the individual filter outputs.

The basic structure of the OTDF is given in [Ree87, Gar93] and is shown in Figure 9. This structure is a time dependent time invariant (TDTI) structure in that the optimal filter for each branch is optimized for the Wiener solution in that branch alone and is incapable of adapting to changes in the input process. The time dependent time varying (TDTV)
form of this structure depicted in Figure 10 is sufficient for a small number of exploitable periodicities.  

Optimal filter structures which utilize the FSR (such as the structures in Figure 9,10) are actually special cases of structures which utilize the TSR. In particular, the FSR structure can be obtained from the TSR structure by replacing banks of periodic synchronized

---

1For the remainder of this thesis the following terms will be used to describe filter structures:

- *time independent* will denote filtering structures which are not configured to exploit the time dependent nature of the cyclic autocorrelation function (and therefore cyclic spectral density) of a cyclostationary signal,

- *time dependent* will denote filtering structures which are able to exploit the cyclostationary nature of a given process,

- *time invariant* will denote filtering structures which are not capable of tracking or adapting to changes in the input process,

- *time varying* will denote filtering structures which are capable of adapting the weights or impulse responses via some error minimization criterion, in order to track changes in the input process.
Figure 10: Optimal Time Dependent Adaptive Structure
samplers with their Fourier series representations [Ree87]. Thus, the TSR and FSR structures become identical when the entire Fourier series representation is used for the FSR filter structure. Fortunately, most of the benefit of optimal time dependent filtering can be achieved through the use of only a few periodicities, and the use of the FSR in place of the TSR is well justified [Ree87].

It is apparent that the structure of Figure 10 is impractical for exploiting the plethora of spectral correlations in most DS-SS signals. For DS-SS modulation with a high code repetition rate, the structure in Figure 10 would require too many phase locked complex oscillators and filter banks to be computationally efficient. In order to minimize the overall computational count, the Discrete Fourier Transform (DFT) can be used in place of the phase locked complex oscillators and filters to achieve the necessary frequency translation and lowpass filtering of spectrally correlated components. If the sampling rate is chosen to be a multiple of the chip rate, and the aforementioned constraints on code repetition and data rates are observed, then an appropriate DFT size can be chosen to exploit the spectral correlation. Recall that the definition of the DFT is given by

$$X(k) = \sum_{n=0}^{N-1} x(n)e^{-j2\pi \frac{kn}{N}}$$

(36)

One can think of the $N$-point DFT as a process whereby the various harmonic components in a signal are translated to baseband and lowpass filtered with a simple moving average or FIR filter having all ones as its filter coefficients. In certain instances, the size of the DFT can be chosen to exploit several fast implementation algorithms such as the Fast Fourier Transform and Winograd Transforms. These implementations offer a substantial computational savings over the DFT for large transform sizes. Thus, the preferred structure for optimal time dependent filtering of DS-SS signals is shown in Figure 11. In particular, great computational savings can be realized when the chip rate, $f_{\text{chip}}$, code repetition rate, $f_{\text{code}}$, and sample rate, $f_s$, are chosen such that the number of required points in the DFT is a power of two, $N = 2^k$. This criterion allows the more efficient FFT to replace the DFT operation. Alternatively, the use of prime factor ratios can be used to reduce the computational count of the transform process if the Winograd transforms are used.
Figure 11: FFT Based Optimal Time Dependent Adaptive Filter Structure Showing the Estimation of One Output Bin
3.2 Training Sequence Adaptation Algorithms for TDAF Processing

This section presents an LMS frequency domain implementation of the FSR Time Dependent Adaptive Filter (TDAF). For large filter lengths the implementation offers substantial computational savings and often exhibits faster convergence rates than its corresponding time domain algorithm [Ree87, NPN83, Yas85, MG81, BA81]. Moreover, the parallel/pipeline nature of the frequency domain TDAF architecture allows wide band TDAFs to be efficiently implemented with currently available hardware by reducing the computational count per branch.

3.2.1 LMS TDAF Implementation

The Circular Time Independent Adaptive Filter (TIAF)

The basic structure of the frequency domain TDAF employs circular convolution and is therefore called the Circular FSR TDAF [Ree87]. Reed and Ferrara [Ree87, Fer85] have developed TDAF structures which utilize the overlap-save method to achieve a linear convolution in the frequency domain; however, the utility of these structures for DS-SS processing is limited. In both instances, the linear convolution implementation of the TDAF results in a doubling of the input and output FFT size and corresponding computations. Thus, the circular implementation of the TDAF is preferred for DS-SS processing.

The time independent time varying form of the algorithm, the Circular TIAF, begins with an \( N \) - point FFT as shown in Figure 12 [Ree87]. Each frequency bin of the initial FFT, \( X_k(f_i) \), is multiplied by a corresponding complex weight, \( W_k(f_i) \), to yield the output estimate \( \hat{Y}_k(f_i) \). If the LMS algorithm is used to update the weight vector for each bin, the output estimate for the \( k \)th block can be written in scalar form as

\[
\hat{Y}_k(f_i) = W_k(f_i)X_k(f_i) \quad i = 0, 1, \ldots, N - 1
\]  

or in vector form as

\[
\hat{Y}_k = W_k \otimes X_k
\]  

where \( \otimes \) denotes element-by-element vector multiplication. The scalar form of the weight adaptation equation is

\[
W_{k+1}(f_i) = W_k(f_i) + \mu E_k(f_i)X_k^*(f_i),
\]  

where the LMS error criterion is

\[
E_k(f_i) = Y_k(f_i) - \hat{Y}_k(f_i)
\]
Figure 12: FFT Based Time Independent Adaptive Filter Structure
or
\[ E_k = Y_k - \hat{Y}_k \] (41)
in vector form. In this algorithm, \( Y_k(f_i) \) represents the training or desired response, and \( \mu \) is the LMS adaptation constant governing both the rate of convergence and amount of misadjustment. To achieve the optimum convergence rate, \( \mu \) must be dynamically adjusted according to some power estimating function. In general, the fastest stable convergence occurs when \( \mu \) is made inversely proportional to the power in a given input FFT bin. For the LMS algorithm in Equations 37 through 39, \( \mu \) is assumed to be a constant that is small enough to insure the stable convergence of all the filter weights. This approach minimizes the computational burden and alleviates potential problems with local instabilities across bins. The Circular TIAF structure has been known for over a decade, and its convergence properties and limitations are well understood [DMW78, Fer80, Shy92]. The Circular TIAF can be modified for optimal time dependent filtering in order to process cyclostationary, or almost cyclostationary signals with minimal changes.

The Circular FSR TDAF

A circular FSR TDAF can be implemented in the frequency domain by applying the steepest-descent LMS optimization procedure to the cost function in Equation 33. One realization of this frequency domain Circular FSR TDAF is given in Figure 11. Like the time domain FSR TDAF, this filter uses a weighting of frequency shifted versions of the input data signal to estimate the desired signal.

Alternatively, the Circular FSR TDAF can be viewed as a multi-channel adaptive filter operating on correlated frequency bins separated by the signal’s statistical periodicities, \( \alpha_n \) [Ree87]. Note that in the DFT implementation, the sampling rate for the input data signal must be an integer multiple of the desired periodicity. For TDAF processing of DS-SS signals, this criterion is equivalent to choosing the sample rate to be in integer multiple of the code rate cycle frequency, \( \alpha_{code} \). If the sampling rate of the input data is not an integer multiple of the desired periodicity, then some form of decimation/interpolation and re-sampling must be performed. (This effect is often described in terms of the “off-bin-center” problem). Furthermore, it is assumed that the input signal has been tuned to 0 Hz in complex baseband form, thus setting the fundamental carrier cycle frequency to zero \( (\alpha_0 = 0 \text{ Hz}) \). The implementation algorithm for the Circular FSR TDAF is given in [Ree87] as
\[ \hat{Y}_k(f_i) = \sum_{n=-l}^{u} W_{k,n}(f_i)X_{k,n}(f_i + \alpha_n) \] (42)
where the weight vector elements are given by

\[
W_{k+1,n}(f_i) = W_{k,n}(f_i) + \mu E_k(f_i) X_{k,n}^*(f_i + \alpha_n) \\
\alpha_n \in \{-l, -l + 1, \ldots, u - 1, u\},
\]

and the error signal is

\[
E_k(f_i) = Y_k(f_i) - \hat{Y}_k(f_i)
\]

In Equations 42 through 44, \( f_i \) refers to the frequency corresponding to the \( i \)th bin of the \( N \)-point FFT. The subscripts \( k \) and \( n \) refer to the \( k \)th block and \( n \)th harmonic of the fundamental cycle frequency, respectively. The range of \( n \) is defined to be \( \{-l, u\} \), where \( l \) and \( u \) are constrained such that \( X_{k,-l}(f_i + \alpha_{-l}) \) and \( X_{k,u}(f_i + \alpha_u) \) lie within the possible FFT bins. To exploit conjugate spectral correlation, conjugated versions of the bins must also be filtered, thus doubling the number of weight vectors.

The Circular FSR TDAF, while being intuitively satisfying, is not an exact implementation of the time domain LMS algorithm. In particular, the circular convolution employed by this frequency domain filter is not equivalent to the linear convolution of the time domain transversal LMS adaptive filter. F. Reed and Feintuch [RF81] have shown that under certain conditions, the wrap around effect of circular convolution introduces "correlated algorithm noise" which can prevent the weight vector from converging to the minimal MSE solution. Thus, in certain cases the Circular FSR TDAF may converge to a sub-optimal solution. Specifically, if the delay between the input data and the desired response is significant (roughly greater than one-fourth of the FFT block size in samples), the circular wrap produces undesired correlation of the input data with itself, and the Circular FSR TDAF will not achieve the optimal MSE. Repeated simulations have shown no significant distortion due to the circular convolution when the filter length is sufficiently long and the delay between the input signal and desired response is minimal.

In some instances linear convolution may be desired, and at least two implementations have been developed for linear convolution in the DFT-TDAF. E. Ferrara [Fer85] has proposed the use of polyphase pre-filter structures before and after the DFT blocks in order to minimize the spectral leakage associated with the discrete Fourier transform, thus precluding circular convolution. This technique essentially converts the DFT bins into a bank of contiguous bandpass filters with minimal sidelobe leakage. In the telecommunications industry, this technique is used to implement a transmultiplexer [Bel74, Cop82, SG87]. The major drawback of this solution is the additional transport delay introduced by both the polyphase prefilter to the input DFT and the phase compensation postfilter to the output DFT\(^{-1}\). In general, for an \( n \)-tap polyphase prefilter and postfilter, the transport delay
is proportional to \( nN \), where \( N \) is the DFT block size. In practice, a three or four tap \((n = 3, 4)\) polyphase filter per bin is sufficient to reduce the sidelobe leakage by 45dB [Hol93]. Over 60dB rejection can be obtained with a five tap per bin prefilter and postfilter.

Another alternative to circular convolution is to utilize the overlap-save algorithm which implements linear convolution in the frequency domain. However, this technique requires a doubling of the DFT size, and a corresponding increase in the computation count for the weight update equations. When a large number spectral correlations exist, the transmultiplexer approach provides a more reasonable computation increase.

An equally important issue to computation count is the suitability of implementing the proposed adaptive filters in parallel or pipeline architectures. All of the proposed adaptive filters can be implemented with such architectures. As stated in previous sections, the time-domain TSR TDAF has the advantage of using all the harmonics of the fundamental frequencies without any additional computation; however, any reduction in computational burden may be offset by the timing and synchronization requirements of the filter structure. The frequency-domain adaptive filters require substantially fewer computations for large filter lengths, and have the added advantage of easier implementation utilizing currently available high-speed FFT chips.

### 3.2.2 Modified LMS TDAF Implementations

One of the major drawbacks of the LMS algorithm is its notoriously slow convergence rate for a given degree of misadjustment in the final weights. In fact, the most difficult problem in applying the LMS algorithm is in the determination of an appropriate value for the convergence parameter, \( \mu \). In general, \( \mu \) is dependent on the statistics of the input signal — quantities which are rarely known a priori. The result is that \( \mu \) is usually chosen artificially low in order to insure filter stability, and the algorithm converges, albeit with small misadjustment, at a "snails pace." It can be shown that the maximum stable value of \( \mu \) is determined by the maximum eigenvalue of the input signal autocorrelation matrix. This result is of limited practical use since the computation of the eigenvalues associated with a large filter length (such as that required for DS-SS processing) is cumbersome and computationally intensive. However, Treichler [TJL87] and other researchers, have derived upper bounds for \( \mu \) based on the eigenvalue concept, and the resulting algorithms are referred to as the Normalized LMS and Accelerated LMS algorithms.
The Normalized LMS TDAF

In the time-domain transversal LMS filter, \( \mu \) is chosen inversely proportional to the total power of the current input signal samples. However, one can think of the FFT operation as a decomposition of a time-domain signal into its orthogonal (or almost orthogonal) frequency components.\footnote{For cyclostationary signals and properly spaced DFT bins, the decomposition produces correlated but independent — not orthogonal — components.} Thus, frequency-domain LMS filters can achieve very fast convergence rates by allowing \( \mu \) to be unique for each FFT bin. Therefore, an \( N \)-point FFT contains \( N \) values for \( \mu \). Obviously, this change requires more memory and a few more computations, but the increased convergence rate often justifies the added expense. For each FFT bin, the corresponding values of \( \mu \) are computed according to:

\[
\mu_k(f_i) = \frac{\Phi}{\gamma + X_k(f_i) \cdot X_k^*(f_i)}
\]  

(45)

The parameter \( \Phi \) determines the amount of misadjustment which can be tolerated in the weights and is constrained to the range \( 0 \leq \Phi \leq 2 \) to insure mean stability of the weights [TJL87]. In practice, \( \Phi \) is chosen to be slightly less than 1 for moderate misadjustment and less than 0.1 for slight misadjustment. \( \gamma \) is a small (several orders of magnitude smaller than the minimum expected bin power) positive constant which insures that the algorithm remains stable even during a temporary loss of signal. The second denominator term simply represents the power in the \( f_i \)’th bin of the input signal FFT.

The Accelerated LMS TDAF

The Accelerated LMS algorithm is similar to the Normalized LMS algorithm with a recursive power estimation. This algorithm updates \( \mu_k(f_i) \) using

\[
\mu_k(f_i) = \frac{\Phi}{\gamma + P_k(f_i)},
\]  

(46)

where the bin powers are recursively updated by

\[
P_k(f_i) = \psi P_{k-1}(f_i) + (1 - \psi) [X_k(f_i) \cdot X_k^*(f_i)]
\]  

(47)

and the recursive weighting factor, \( \psi \), is chosen to minimize the effects of instantaneous bin power fluctuation:

\[
0.5 \leq \psi < 1.0
\]  

(48)

In practice \( \psi \) is usually chosen to be \( \psi \approx 0.9 \). Higher values lead to slower convergence and lower final misadjustment. Repeated simulations have shown that the Accelerated
LMS algorithm is less sensitive to instantaneous fluctuations in signal power, and outperforms the Normalized LMS algorithm in environments highly corrupted by noise or CDMA interference.

The benefits of the Normalized LMS and Accelerated LMS algorithms over the traditional LMS algorithm easily justify the added computational burden. The algorithms replace the guessing or estimation often involved in choosing $\mu$ with the choice of a single misadjustment parameter $\Phi$, and insure a near-optimal convergence rate. The incorporation of a bin-dependent $\mu$ into the LMS TDAF algorithm of the previous section is a trivial matter, and yields useful algorithms which are easily realized in hardware. In light of the minor changes, the derivation of the Circular TDAF algorithm with bin-dependent $\mu$ is not presented. A simple substitution of Equation 45 or Equations 46 and 47 into Equation 43 yields the desired result.

3.2.3 RLS TDAF Implementations

The previous sections of this thesis have used the LMS algorithm to define the necessary error criterion for weight adaptation of the TDAF. While the LMS algorithm requires minimal computation and hardware, it is often slow to converge. This property results directly from the dependence of the LMS algorithm on the eigenvalue spread in the autocorrelation matrix for the input data. At least two solutions to this problem were proposed — the Normalized and Accelerated LMS algorithms. These algorithms attempt to decrease the convergence time by redefining the adaptation constant, $\mu$, as a time varying quantity. By defining $\mu_i$ inversely proportional to the signal power, the effect of the eigenvalue spread of the input autocorrelation matrix is decreased, and the convergence rate of the LMS algorithm is improved. However, even the Normalized LMS algorithm converges at an unacceptably slow rate when the misadjustment parameter is chosen very small in order to achieve a low final misadjustment. Also, the LMS algorithm produces an essentially time-invariant solution once the weights converge due to the "large memory" of the weight adaptation equation and slow tracking. This feature makes the LMS algorithm unattractive for environments where the interference or signal statistics change over short periods of time. In particular, the LMS algorithm, when applied to TDAF structures configured for DS-SS signal processing, converges too slowly to track the dynamic mobile channel. Therefore, the utility of gradient-based TDAF structures is limited to fixed CDMA applications such as wireless local area networks.

Instead of utilizing gradient search approximations to the least squares problem, many algorithms attempt to provide an exact solution. Such is the case with the recursive least
squares (RLS) algorithm. In the ideal case, the RLS algorithm forms an exact solution to the least squares problem at each iteration and converges immediately — i.e., the RLS defines precisely the right step size and direction to retain optimallity for each input block. In practice, the error introduced by the initialization process of the RLS algorithm produces an approximate solution which converges extremely fast. When compared to the LMS algorithm, the RLS algorithm is very attractive due to its fast convergence, relative insensitivity to the eigenvalue spread, and lower steady-state mean squared error [Hay86, Hay91]. The major drawbacks of the RLS algorithm are its increased computation count and storage requirements.

The Time Domain RLS Algorithm

The time domain version of the RLS algorithm as described in [TJL87] begins with the definition of the input vector

$$x(k) = \begin{bmatrix} x(k) & x(k-1) & \cdots & x(k-N+1) \end{bmatrix}^T$$

(49)

and the weight vector

$$w^0(k) = \begin{bmatrix} w_0(k) & w_1(k) & \cdots & w_{N-1}(k) \end{bmatrix}^T$$

(50)

where $k$ represents the current sample index and $N$ is the order of a transversal RLS filter. The superscript ‘o’ in $w^o(k)$ denotes that the weight vector is optimal at time $k$. This distinction is important, since the RLS algorithm uses the sub-optimal weight vector at time $k - 1$ to form an exact solution for the optimal weight vector at time $k$. Thus, the weights at time $k$ are actually optimal for the data at time $k - 1$. In order to obtain an optimal filtering of the input data, two separate filterings are required during each iteration of the RLS algorithm. The suboptimal filtering is typically called the a priori solution, and will be denoted $\hat{y}^o(k)$. The optimal filtering is called the a posteriori solution and will be denoted $\hat{y}^p(k)$. In practice, the two solutions are very similar after the weights converge, so the second or “optimal” filtering of the data is often omitted from the algorithm to reduce computation. If the increased computation count can be tolerated, it is useful to perform both filterings in order to monitor algorithm convergence. This technique is especially useful when no training sequence is available and the desired response is generated from a “blind” algorithm.

The time-domain RLS algorithm begins with the initialization of the weight vector, $w^0(0)$, and the inverse autocorrelation matrix, $r_0^{-1}(0)$,

$$w^o(k-1) = \begin{bmatrix} 0 & 0 & \cdots & 0 \end{bmatrix}^T$$

(51)
\[ r_0^{-1}(k - 1) = \eta \cdot I \]

where \( \eta \) is a large positive constant and \( I \) represents the \( N \times N \) identity matrix. In order to insure that the initial value of the inverse correlation matrix has no significant effect on the algorithm, \( \eta \) should be chosen several orders of magnitude higher than the maximum expected power of the input signal. Alternatively, \( r_0^{-1}(k - 1) \) can be determined from the inversion of the autocorrelation matrix formed from \( N \) points of data taken previous to time \( k \), but this procedure is rarely used since it requires the computation of \( N^2 \) autocorrelation elements and the inversion of an \( N \times N \) matrix. The small error introduced by the initialization in Equation 52 is quickly removed after the first or second iteration [Hay91].

After initialization, the RLS algorithm can be described in eight steps. First, the a priori output is computed from the matrix product of the transposed input vector (which contains the most recent sample) and the previous optimal weight vector:

\[ \hat{y}^a(k) = w^o(k - 1)x(k) \]

Next, the a priori error is computed from the current desired response and the a priori output as

\[ e(k) = y(k) - \hat{y}^a(k) \]

Third, the gain vector \( z(k) \) is computed from the vector product of the conjugate transpose of the current input vector and the previous optimal inverse correlation matrix.

\[ z(k) = x^H(k)r_0^{-1}(k - 1) \]

In order to form a normalized gain vector (traditionally called the Kalman gain vector), the normalized error power constant \( q \) is computed

\[ q = z(k)x(k) \]

from which the gain constant \( v \) is found

\[ v = \frac{1}{1 + q} \]

Thus, the normalized gain vector becomes

\[ \hat{z}_k = v \cdot z^H(k) \]

and the new optimal weight vector for the above data is computed as

\[ w^o(k) = w^o(k - 1) + e(k) \cdot \hat{z}^*(k) \]
The corresponding inverse correlation matrix becomes

$$r_o^{-1}(k) = r_o^{-1}(k - 1) - \hat{z}(k)\hat{z}(k)$$  (60)

Equations 53 through 60 and the initial conditions of 51, 52 form the time domain version of the RLS algorithm. Several versions of the algorithm have been proposed, but the version presented above is computationally efficient in that no unneeded variable is computed and no needed variable is computed twice [TJL87].

Unfortunately, Haykin [Hay91] has shown that the RLS algorithm presented above may become unstable. The source of the potential instability rests indirectly in Equation 58. Ideally, the inverse of the autocorrelation matrix, $r_o^{-1}(k)$, is conjugate symmetric (Hermetian). However, it is possible for the autocorrelation matrix to become very nearly singular for certain input data. This singularity results in a poorly conditioned autocorrelation matrix for the input data, the inverse of which is dominated by numerical truncation and rounding errors. The result is that asymmetry can be introduced into the inverse autocorrelation matrix. A simple exercise in vector algebra shows that any asymmetry in $r_o^{-1}(k)$ is simply amplified by the recursive nature of Equations 58 and 60. Haykin [Hay91] has derived a simple modification to the traditional RLS algorithm which insures the stability of the algorithm. The solution involves replacing Equation 58 with its definition:

$$\hat{z}_k = v \cdot [x^H(k)r_o^{-1}(k - 1)]^H$$  (61)

Using the vector identity, $[AB]^H = B^HA$, Equation 61 can be re-written as

$$\hat{z}_k = v \cdot r_o^{-1H}(k - 1)x(k)$$  (62)

which yields the same result as the previous algorithm with an increase in the total computational count. However, if we assume that the autocorrelation matrix is properly conditioned, then the inverse autocorrelation matrix is Hermetian, and Equation 62 can be re-written

$$\hat{z}_k = v \cdot r_o^{-1}(k - 1)x(k)$$  (63)

Notice that when Equation 63 is substituted for Equation 58 in the previous RLS algorithm, any asymmetry in the inverse autocorrelation matrix is removed by the inverse autocorrelation update of Equation 60. Thus, stability of the RLS algorithm can be guaranteed if the additional computation of Equation 63 can be tolerated. Technically, the substitution of Equation 63 for Equation 58 only guarantees marginal stability; however, the inclusion of exponential weighting in the algorithm produces unconditional stability.

As presented, the RLS algorithm is an efficient solution to the least squares problem when the statistics of the input data are well behaved over the entire transmission $0 \leq k \leq L$.  

CHAPTER 3. OPTIMAL TIME DEPENDENT FILTERS

In this form, the RLS algorithm is known to converge in approximately $2N$ iterations or less, where $N$ represents the number of taps in the filter. If the input data changes character within the transmission, then the infinite averaging interval of the RLS algorithm will not allow the filter weights to adapt accordingly [Str91]. One solution to this problem is to choose an averaging interval which is slightly shorter than the interval of data consistency. This approach gives rise to the RLS algorithm with exponential weighting.

The Time Domain RLS with Exponential Weighting

If the “windowed” autocorrelation matrix is defined as

$$r_o(k) = \sum_{n=k-N-1}^{k-1} \rho^{k-1-n} x(n)x^H(n)$$

then the contribution of older data is reduced when $\rho < 1$. In this form $\rho$ is called the “exponential” or “forgetting” factor. This exponential weighting emphasizes the effects of the most recent data on the autocorrelation matrix and eventually eliminates the effects of the older data. This feature allows the RLS algorithm to track non-stationary input signals or changing interference levels, so long as $\rho$ is chosen to emphasize only the consistent intervals of data [Str91]. The RLS algorithm of Section 3.2.3 is easily modified to include exponential weighting by substituting

$$\tau = \frac{1}{\rho + q}$$

for Equation 57, and scaling the inverse correlation matrix of Equation 60 by $\frac{1}{\rho}$ to form

$$r_o^{-1}(k) = \left[ \frac{1}{\rho} \right] \left[ r_o^{-1}(k-1) - \hat{z}(k)z(k) \right]$$

In practice, the exponential factor $\rho$ is chosen to be a positive constant slightly less than 1. Exponential factors of 0.5 or less generally produce excellent intermediate tracking results but poor LS solutions to long periods of stationary data. An excellent rule of thumb is to choose $\rho$ such that the data after $2N$ iterations has less than 1% of the effect of present data on the autocorrelation matrix. In other words, choose $\rho$ such that

$$[\rho]^{2N} \leq 0.01$$

This choice of $\rho$ produces one of the best trade-off between intermediate tracking and global convergence.
The Circular RLS TIAF with Exponential Weighting

Perhaps the greatest problem with the time-domain RLS algorithm is the large computational count it requires. The Circular RLS TIAF alleviates much of this computation by processing the data in approximately orthogonal blocks, which substantially reduces the number of required algorithm iterations; in fact, the Circular RLS TIAF is typically able to converge after only a few iterations. The RLS algorithm in Section 3.2.3 is composed of linear operations; thus, the Circular RLS TIAF can be derived from the time-domain RLS through the use of Fourier transformations and complex notation. Recall that the FFT of input data essentially decomposes the input signal into orthogonal frequency components. If the transformed input vector elements are defined as

\[ X_k(f_i) = \text{i'th bin of } \text{FFT} \left[ x(k) \ x(k-1) \ \cdots \ x(k-N+1) \right]^T, \]  

(68) 

(\(k\) represents the \(k\)'th \(N\)-point block and \(i\) represents the \(i\)'th frequency bin of the FFT) then the frequency bins are orthogonal and independent in time. (e.g. – The bins are independent to each other, and no single bin can be better estimated from a linear combination of previous values of that bin.) Thus, the weight adaptation for each bin can be controlled by a single-tap version of the RLS algorithm. For the single-tap case, the RLS algorithm reduces to eight independent scalar equations per bin. The block diagram for the Circular RLS TIAF is identical to Figure 12, but the following procedure is used for the weight adaptation algorithm.

The Circular RLS algorithm begins with the initialization of the frequency domain weight vector and the inverse point correlation vector (since there is only one tap per bin):

\[ W_{k-1}^0 = \left[ \begin{array}{ccc} 0 & 0 & \cdots & 0 \end{array} \right]^T \]  

(69) 

\[ R_{k-1}^{-1} = \eta \cdot \left[ \begin{array}{ccc} 1 & 1 & \cdots & 1 \end{array} \right]^T \]  

(70) 

Again, \(\eta\) is chosen to be a large positive constant in order to minimize its effect on the weight vector initially. Next, the a priori frequency-domain output is computed as

\[ \hat{Y}_k^a = W_{k-1}^0 \otimes X_k \]  

(71) 

where the symbol \(\otimes\) denotes element-by-element multiplication of the vectors — i.e.,

\[ \hat{Y}_k^a(f_i) = W_{k-1}^0(f_i)X_k(f_i). \]  

(72) 

Next, the bins of the a priori frequency domain error vector are computed from the difference of the transformed desired response bins and the a priori output bins

\[ E_k(f_i) = Y_k(f_i) - \hat{Y}_k^a(f_i) \]  

(73)
The frequency-domain gain vector is then computed from the element-by-element product of the previous optimal inverse point correlation vector and the current frequency-domain input vector
\[ Z_k = X_k^* \otimes R_{k-1}^{-1} \] (74)
Again, to normalize the frequency-domain gain vector, the normalization constants for each bin are computed using
\[ Q_k = Z_k \otimes X_k \] (75)
from which the gain constants for each bin become
\[ V_k = \left[ \frac{1}{\rho + Q_k(f_0)} \left( \frac{1}{\rho + Q_k(f_1)} \right) \cdots \left( \frac{1}{\rho + Q_k(f_{N-1})} \right) \right]^T. \] (76)
The elements of the normalized gain vector are then computed using the computationally efficient form of the RLS algorithm
\[ \tilde{Z}_k = V_k \otimes Z_k^* \] (77)
since there can be no asymmetry in a point or element. Finally, the new optimal frequency-domain weight vector is updated according to
\[ W_k^o = W_{k-1}^o + E_k \otimes \tilde{Z}_k^* \] (78)
and the new inverse point correlation vector is computed from
\[ R_{k-1}^{-1} = \left[ \frac{1}{\rho} \right] \left[ R_{k-1}^{-1} - \tilde{Z}_k \otimes Z_k \right] \] (79)
at which point the time-domain solution is obtained from
\[ \begin{bmatrix} \hat{y}(k) & \hat{y}(k-1) & \cdots & \hat{y}(k-N+1) \end{bmatrix}^T = \text{FFT}^{-1} \left[ \hat{Y}_k \right] \] (80)
Although the initial computational count is high, the Circular RLS TIAF can easily be justified by its rapid convergence. As previously stated, the RLS algorithm typically converges at least an order of magnitude earlier than the LMS algorithm, and it converges to the optimal solution of the Least Squares problem. The scalar equations for each bin in the above algorithm do not begin to exploit the full power of the RLS algorithm. Traditionally, the RLS algorithm has been used to adaptively combine correlated signals in multi-array sensors and delay lines. Thus, the power of the RLS algorithm can be utilized for time-dependent filtering to adaptively combine spectrally correlated portions of the signal spectrum in the following manner.
The Circular RLS TDAF with Exponential Weighting

Again, the block diagram for the Circular RLS TDAF is identical to the Circular TDAF of Figure 11 with a change in the weight adaptation algorithm. The basic goal is to minimize the least squares error between a desired frequency component and a linear combination of correlated frequency components. As in the case of the preceding TDAF's, it is assumed that the sample rate of the input data signal is an integer multiple of the desired periodicity. As in Equation 68, the transformed input vector is given by:  

\[ X_k(f_i) = i^{th} \text{ bin of FFT } \left[ x(k) \ x(k-1) \ \cdots \ x(k-N+1) \right]^T \]  

(81)

Again, the range of \( n \) is defined to be \( n \in \{-l, u\} \) so that \( X_k(f_i + \alpha_{-l}) \) and \( X_k(f_i + \alpha_u) \) lie within the possible FFT bins. The weight vectors and inverse spectral correlation matrix of the Circular RLS TDAF are initialized according to

\[ \sum_{n=-l}^{u} W_{k-1,n} = \left[ \begin{array}{ccc} 0 & 0 & \cdots \end{array} \right]^T \]  

(82)

\[ R_{k-1}^{-1} = \eta \cdot I \]  

(83)

where \( \eta \) is chosen as a large positive constant and \( I \) represents the \( N \times N \) identity matrix. If we assume that all available periodicities will be used by the algorithm to form the output bin estimates (which increases performance, yet increases the total computation count very little in the RLS algorithm), then it becomes convenient to define the weight vectors of Equation 82 as rows of a single \( N \times N \) matrix:

\[ W_k^o = \begin{bmatrix} 
W_{0,0}^o & W_{0,1}^o & W_{0,2}^o & \cdots & W_{0,N-1}^o \\
W_{1,0}^o & W_{1,1}^o & W_{1,2}^o & \cdots & W_{1,N-1}^o \\
\vdots & \vdots & \vdots & & \vdots \\
W_{N-1,0}^o & W_{N-1,1}^o & W_{N-1,2}^o & \cdots & W_{N-1,N-1}^o 
\end{bmatrix} \]  

(84)

\[ ^3 \text{If conjugate spectral correlation is to be exploited, then the length of the input vector doubles to } 2N \text{ where the first } N \text{ elements are given by} \]

\[ X_{1k} = \text{FFT } \left[ x(k) \ x(k-1) \ \cdots \ x(k-N+1) \right] \]

and the remaining \( N \) elements are given by the conjugate transform

\[ X_{2k} = \text{FFT}^* \left[ x(k) \ x(k-1) \ \cdots \ x(k-N+1) \right] \]

such that the input vector \( X_k \) becomes

\[ X_k = \left[ X_{1k} \ X_{2k} \right]^T. \]
where the weights for the estimation of the zero'th output frequency bin, $\hat{Y}_k(f_0)$, are contained in the first row of $W_k^0$. For the processing of DS-SS signals, where the bin centers are spaced at the fundamental code repetition cycle frequency, optimal performance can only be achieved by using the weight matrix of Equation 84 in place of a few select weight vectors. When this approach is used, the initialization of the weight matrix simply becomes

$$W_{k-1}^0 = 0 \cdot I.$$  (85)

The elements of the a priori frequency-domain output bins are computed in scalar form as

$$\hat{Y}_k^a(f_i) = \sum_{u=-i}^{u} W_{k-1,n}^0(f_i) \cdot X_{k,n}(f_i + \alpha_n)$$  (86)

or in matrix form as the product of the weight matrix and the input FFT bins

$$\hat{Y}_k^a = W_{k-1}^0 X_k.$$  (87)

In this form, each bin of $\hat{Y}_k^a$ is formed from a linear combination of all the input bins. Unless the frequency separation between the input signal FFT bins represents the fundamental periodicity of the input signal (i.e. the code repetition cycle frequency, $\alpha_{code}$), it is obvious that certain weights in the $N \times N$ weight matrix must be zero for the matrix product of Equation 87 to be equivalent to the scalar sum in Equation 86. This difference is addressed in the weight adaptation equation to follow. After forming the a priori output estimate, the frequency-domain error vector is computed as

$$E_k = Y_k - \hat{Y}_k^a$$  (88)

or in scalar form

$$E_k(f_i) = Y_k(f_i) - \hat{Y}_k^a(f_i)$$  (89)

The frequency-domain gain vector is then computed based on the vector product of the inverse spectral correlation matrix and the current frequency-domain input vector,

$$Z_k = X_k^H R_k^{-1}$$  (90)

at which point the normalization constant is computed from the sum of the products of the conjugated input bin values and their respective gain-vector components

$$Q = \sum_{i=0}^{N-1} X_k(f_i) \cdot Z_k(f_i)$$  (91)

This operation is more compactly written as the matrix product

$$Q = Z_k X_k$$  (92)
From the normalization constant, the corresponding gain constant can be computed as
\[ V = \frac{1}{\rho + Q} \]  
(93)
and the normalized gain vector computed as
\[ \tilde{Z}_k = V \cdot Z_k \]  
(94)
Since the spectral correlation matrix is no longer a point vector, it is wise to incorporate Haykin's modification to the RLS algorithm in order to insure stability of the algorithm. This is easily achieved by replacing Equation 94 with
\[ \tilde{Z}_k = V \cdot R_{k-1}^{-1} X_k \]  
(95)
With the normalized gain vector computed, the new optimal weight vectors for each output bin can be computed. Recall that in scalar form, the weight vectors for each periodicity are computed by summing the previous weight with the scalar product of the error bin and the conjugated normalized gain vector:
\[ W_{k,n}^o = W_{k-1,n}^o + E_k(f_i) \cdot \tilde{Z}_k^* \quad n \in \{-l, -l + 1, \cdots, 0, \cdots, u - 1, u\} \]  
(96)
This process is equivalent to forming an $N \times N$ matrix where each row of the matrix equals the conjugate of the normalized gain vector $\tilde{Z}_k^*$, multiplying each row of the new matrix by a scalar (the value of the $i$'th error bin), and adding the result to the previous weight vector:
\[ W_k^o = W_{k-1}^o + E_k \tilde{Z}_k^H \]  
(97)
Technically, Equations 96 and 97 are only equivalent when all available periodicities are used and the bin spacing of the input FFT equals the fundamental code repetition cycle frequency. In order to make the two processes equivalent for all cases, the weights in the matrix of Equation 97 that do not correspond to spectrally correlated bins must be zeroed. This operation is easily accomplished by introducing a "mask vector", $A$, such that the locations of desired periodicities are set equal to 1 in the mask and all other elements are zero. For example, if we assume that spectral correlation exists between every third bin of
of the input FFT, the \( N \times N \) mask vector would take the form

\[
A = \begin{bmatrix}
1 & 0 & 0 & 1 & 0 & 0 & 1 & \cdots & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
\vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \ddots & \vdots \\
1 & 0 & 0 & 1 & 0 & 0 & 1 & \cdots & 0 \\
0 & 1 & 0 & 0 & 1 & 0 & 0 & \cdots & 0 \\
0 & 0 & 1 & 0 & 0 & 1 & 0 & \cdots & 1 \\
\end{bmatrix}
\]  \hspace{1cm} (98)

and the constrained weight vector for the next iteration would become

\[
W_k^o = A \otimes W_k^o
\]  \hspace{1cm} (99)

where once again \( \otimes \) denotes element-by-element multiplication. In practice, this masking procedure has been found to be useless unless the adaptive filter is configured to implement a Spectral Correlation Discriminator (SCD) [Ree87]. The RLS algorithm, if implemented without the masking procedure, quickly realizes that uncorrelated bins are unnecessary for forming the estimate of a particular output bin, and automatically sets the appropriate weights to zero. In fact, eliminating the masking procedure may actually help minimize the effects of circular convolution and uncorrelated interference by allowing certain uncorrelated bins to be slightly weighted with inverse phasing to cancel the unwanted components of “correlated algorithm noise” in the filter output [RF81].

The final step of the Circular RLS TDAF algorithm simply updates the inverse spectral correlation matrix using the normalized and unnormalized gain vectors

\[
R_k^{-1} = \frac{1}{\rho} \left[ R_{k-1}^{-1} - \tilde{Z}_k Z_k \right]
\]  \hspace{1cm} (100)

after which the time-domain solution is obtained from

\[
\begin{bmatrix}
\hat{y}^a(k) & \hat{y}^a(k-1) & \cdots & \hat{y}^a(k-N+1)
\end{bmatrix}^T = \text{FFT}^{-1} \left[ \hat{Y}_k^a \right]
\]  \hspace{1cm} (101)

The Circular RLS TDAF, while requiring substantial computation, offers impressive performance and rapid convergence with minimum steady-state misadjustment. Additionally, the “forgetting factor” incorporated in the algorithm allows the Circular RLS TDAF to track changes in the interference environment — a critical requirement for the mobile channel. The vector forms of the Circular RLS TDAF algorithm are summarized in Tables 1
and 2. It is highly recommended that Version 2 be implemented whenever finite arithmetic or highly correlated input data is processed in order to insure algorithm stability.

A version of the RLS TDAF using linear convolution in the frequency domain will not be presented for several reasons. First, as previously mentioned, the Circular RLS TDAF appears to compensate for many of the effects of circular convolution. Second, the use of linear convolution requires the input FFT size to be doubled, which increases the number of weights from $N^2$ to $(2N)^2$. A similar increase in computation count follows, since the inverse spectral correlation matrix also increases in size to $(2N)^2$ and all other vectors increase in length by a factor of two. Such an increase in computation count renders the RLS TDAF algorithm unrealizable for many practical systems. If the effect of linear convolution is desired, then the use of a 4-tap polyphase pre-filter and postfilter to the input and output FFTs, respectively, is recommended. Such prefiltering simply adds $8N$ multiplications to the overall computation count of the algorithm — a far cry from the tremendous increase in computation resulting from a doubling of the input vector size.

3.3 Blind Adaptation Algorithms for TDAF Processing

Section 3.2 introduced several adaptation algorithms for the optimal time dependent filtering structure. Each of the adaptation algorithms attempts to minimize the MSE between a desired signal and the OTDF output. Unfortunately, the “desired signal”, $y(k)$, represents a training sequence which spans the duration of the transmitted message. Such training sequences are unrealistic for information bearing systems. In order to adaptively track changes in the input process, some form of non-training sequence directed adaptation criterion must be employed.

Several algorithms have been developed for the blind adaptation of time dependent filters. These algorithms primarily include Bussgang techniques such as the Constant Modulus Algorithm [RMHA89] and Decision Feedback Equalization [HHS93]. Several other algorithms have been modified to accommodate time dependent structures [MRH89] [Ree87]; however, the Decision Feedback Equalization (DFE) structures represent the fastest converging and best tracking class of algorithms when baud synchronization can be achieved and the received bit error rate is less than one percent [HHS93].

3.3.1 The DFE-TDAF Algorithm

The incorporation of the DFE adaptation algorithm into the time dependent structure represents a natural extension of the work in Section 3.2. In fact, the training sequence
Table 1: Circular RLS TDAF Algorithm — Version 1

\[
W_{k-1}^o = 0 \cdot I \\
R_{k-1}^{-1} = \eta \cdot I \\
X_k(f_i) = i^{th} \text{ bin of } \text{FFT} \left[ \begin{array}{ccc} x(k) & x(k-1) & \cdots & x(k-N+1) \end{array} \right]^T \\
\hat{Y}_k = W_{k-1}^o X_k \\
E_k = Y_k - \hat{Y}_k \\
Z_k = X_k^H R_{k-1}^{-1} \\
Q = Z_k X_k \\
V = \frac{1}{\rho_k^2 Q} \\
\tilde{Z}_k = V \cdot Z_k^H \\
R_k^{-1} = \frac{1}{\hat{p}} \left[ R_{k-1}^{-1} - \tilde{Z}_k Z_k \right] \\
W_k^o = W_{k-1}^o + E_k \tilde{Z}_k^H \\
\left[ \begin{array}{ccc} \hat{y}^a(k) & \hat{y}^a(k-1) & \cdots & \hat{y}^a(k-N+1) \\
\hat{y}^p(k) & \hat{y}^p(k-1) & \cdots & \hat{y}^p(k-N+1) \end{array} \right]^T = \text{FFT}^{-1} \left[ \hat{Y}_k \right] \\
\left[ \begin{array}{ccc} \hat{y}^a(k) & \hat{y}^a(k-1) & \cdots & \hat{y}^a(k-N+1) \\
\hat{y}^p(k) & \hat{y}^p(k-1) & \cdots & \hat{y}^p(k-N+1) \end{array} \right]^T = \text{FFT}^{-1} \left[ W_k^o X_k \right]
\]
Table 2: Circular RLS TDAF Algorithm — Version 2

\[
W_{k-1}^o = 0 \cdot I
\]

\[
R_{k-1}^{-1} = \eta \cdot I
\]

\[
X_k(f_i) = i^{th} \text{ bin of FFT } \begin{bmatrix} x(k) & x(k - 1) & \cdots & x(k - N + 1) \end{bmatrix}^T
\]

\[
\hat{Y}_k = W_{k-1}^o X_k
\]

\[
E_k = Y_k - \hat{Y}_k
\]

\[
Z_k = X_k^H R_{k-1}^{-1}
\]

\[
Q = Z_k X_k
\]

\[
V = \frac{1}{\rho + Q}
\]

\[
\tilde{Z}_k = V \cdot R_{k-1}^{-1} X_k
\]

\[
R_k^{-1} = \frac{1}{\rho} \left[ R_{k-1}^{-1} - \tilde{Z}_k Z_k \right]
\]

\[
W_k^o = W_{k-1}^o + E_k \tilde{Z}_k^H
\]

\[
\begin{bmatrix}
\hat{y}^a(k) & \hat{y}^a(k - 1) & \cdots & \hat{y}^a(k - N + 1)
\end{bmatrix}^T = \text{FFT}^{-1} \left[ \hat{Y}_k \right]
\]

\[
\begin{bmatrix}
\hat{y}^p(k) & \hat{y}^p(k - 1) & \cdots & \hat{y}^p(k - N + 1)
\end{bmatrix}^T = \text{FFT}^{-1} \left[ W_k^o X_k \right]
\]
adaptation algorithm may be thought of as a DFE adaptation algorithm having perfect decisions and timing recovery. That is to say, decision directed adaptation is obtained by making a "best guess" of the transmitted symbol based on the filter output, \( \hat{y}(k) \), and using this "best guess" as the training or desired sequence, \( y(k) \). The best estimate of the correct symbol, \( s_c \), is chosen from the entire symbol set, \( s_c \in \{ s_m : \forall m \} \), to minimize the normalized distance, \( \Delta \), between the filter output and the nearest symbol state. Therefore, the goal is to find \( m \) for which

\[
\Delta = \| \hat{y}(k) - s_c \| \\
= \min_{\forall m} \| \hat{y}(k) - s_m \| \tag{102}
\]

Having selected the best symbol estimate, the error is determined by

\[
e(k) = s_c - \hat{y}(k) \\
= y(k) - \hat{y}(k) \tag{103}
\]

For DS-SS modulation, the valid symbol states after despreading consist of \( s_m \in \{-1, +1\} \). Assuming that the recovery of accurate symbol timing is achievable, the DFE-TDAF structure can be implemented with any of the preceding adaptation algorithms.

The basic block diagram of the RLS-DFE-TDAF (DFE-TDAF for short) configured for DS-SS interference cancellation is identical to the TDAF of Figure 11 with the combined RLS-DFE algorithm used for weight adaptation. The only change in the RLS adaptation algorithm is a substitution of the demodulated (respread) decision, \( S_c \), for the desired response, \( Y_k \). Thus, the demodulated output is remodulated and used as the training sequence for the algorithm. The modified algorithm for the DFE-TDAF is given in Table 3.

Unfortunately, the DFE-TDAF suffers from the well known "catastrophic failure" of traditional DFE algorithms [HHS93]. Although the RLS-DFE combination represents one of the fastest converging and best tracking combinations [HR91] for the mobile channel, it can fail to converge when the decision error rate grows sufficiently high (roughly greater than one percent). In order to combat this problem, R. Holley and J. Reed have developed a more robust version of the RLS-DFE which borrows heavily from the theory of neural networks.

### 3.3.2 The Square-Law DFE-TDAF Algorithm

Neural nets have two basic advantages over traditional filter adaptation algorithms and structures. First, the neural net can compensate for non-linear and non-minimal phase
CHAPTER 3. OPTIMAL TIME DEPENDENT FILTERS

Table 3: The DFE-TDAF Algorithm

\[ W_{k-1}^o = 0 \cdot I \]

\[ R_{k-1}^{-1} = \eta \cdot I \]

\[ X_k(f_i) = i' \text{th bin of FFT} \left[ x(k) \ x(k-1) \ \ldots \ x(k-N+1) \right]^T \]

\[ \hat{Y}_k = W_{k-1}^o X_k \]

\[ E_k = S_c - \hat{Y}_k \]

\[ Z_k = X_k^H R_{k-1}^{-1} \]

\[ Q = Z_k X_k \]

\[ V = \frac{1}{\rho + Q} \]

\[ \tilde{Z}_k = V \cdot R_{k-1}^{-1} X_k \]

\[ R_k^{-1} = \frac{1}{\rho} \left[ R_{k-1}^{-1} - \tilde{Z}_k Z_k \right] \]

\[ W_k^o = W_{k-1}^o + E_k \tilde{Z}_k^H \]

\[
\begin{bmatrix}
\hat{y}^a(k) & \hat{y}^a(k-1) & \ldots & \hat{y}^a(k-N+1)
\end{bmatrix}^T = \text{FFT}^{-1} \left[ \hat{Y}_k \right]
\]

\[
\begin{bmatrix}
\hat{y}^p(k) & \hat{y}^p(k-1) & \ldots & \hat{y}^p(k-N+1)
\end{bmatrix}^T = \text{FFT}^{-1} \left[ W_k^o X_k \right]
\]
distortions [RHVH93]. Second, certain forms of neural nets can update the weighting vectors based on the strength of a decision. It is the second feature which allows modifications of the DFE-TDAF to achieve impressive performance.

The update equation for the weight matrix in the RLS-TDAF algorithm is given in Equation 97 and is repeated below.

\[ W_k^o = W_{k-1}^o + E_k \tilde{Z}^H \]  \hspace{1cm} (104)

Recall that \( E_k \) represents the frequency domain error vector and \( \tilde{Z} \) represents the Kalman gain vector. Together these vectors form the exact least squares correction to \( W_{k-1}^o \) for the proper least squares solution to the entire data set. This update is optimal only when the error vector represents a known degree of error — such as when a training sequence is employed for \( y(k) \). For the DFE-TDAF, the desired response is provided by a hard decision on the filter output \( \hat{y}(k) \). This decision may or may not be correct; however, the DFE-TDAF, as presented, assumes that the decision is correct in the weight update equation. Thus, the weight vector is updated just as strongly for questionable decisions as for known decisions. It is this feature of the DFE which leads to “catastrophic failure.” In order to mitigate the effects of poor decisions, some degree of information regarding the strength of the decision must be incorporated into the algorithm. Neural nets accomplish this function by using sigmoidal or non-linear weighting criterion.

For DS-SS with BPSK message modulation, the ideal constellation diagram for the demodulated data is represented by the solid black dots in Figure 13. In this figure, \( d_1 \) and \( d_2 \) represent two different outputs from the BPSK demodulator. Notice that the error vectors for the \( d_1 \) output are essentially the same length. This implies that little confidence can be attached to a decision of either \( s_c = -1 \) or \( s_c = 1 \) for the DFE. However, the traditional RLS-DFE will weight the decisions for both \( d_1 \) and \( d_2 \) equally. In order to incorporate the strength of the decision into the RLS-DFE algorithm, Equation 104 must be modified with a scale factor applied to the update terms. Thus, the new weight update equation becomes

\[ W_k^o = W_{k-1}^o + \Psi_k \cdot E_k \tilde{Z}^H \]  \hspace{1cm} (105)

In this fashion, \( \Psi_k \) becomes a scalar quantity by which the least squares update is scaled. The traditional form of the RLS-DFE is realized by setting \( \Psi_k = 1 \). However, by defining \( \Psi_k \) as a function of the frequency domain error vector, a better choice can be obtained.

In the time domain, the most simple measure for \( \Psi_k \) is the value of the normalized power in the error. Since the time domain error output of the demodulator is given by

\[ e(k) = y(k) - \hat{y}(k) \]
Figure 13: Constellation of Ideal BPSK at Baseband

\[ s_c - \hat{y}(k) \]  

the power of the error output can easily be computed as

\[ \text{error power} = e(k)e^*(k) \]  

It is desirable, however, to compute the scale factor from the frequency domain error vector, \( \mathbf{E}_k \). In this case, the error power output becomes the \textit{average} power of the error vector. Additionally, the error vector must be scaled by \( \frac{1}{N} \) to account for the DFT scaling of frequency bins. Thus, we define the average squared error radius as

\[ T = \left[ \frac{1}{N^2} \right] \mathbf{E}_k^H \mathbf{E}_k \]  

where \( \mathbf{E}_k \) is given in Table 3. The scale factor may then be computed in conditional form as

\[ \Psi_k = \begin{cases} \frac{d_{\text{min}}}{2} - T & 0 \leq T < \frac{d_{\text{min}}}{2} \\ 0 & T \geq \frac{d_{\text{min}}}{2} \end{cases} \]  

or in equation form as

\[ \Psi_k = \frac{1}{2} \left[ \frac{d_{\text{min}}}{2} - T \right] \left[ 1 + \frac{d_{\text{min}}}{2 - T} \right] \]
Figure 14: Scale Factor, $\Psi_k$, for BPSK Baseband Modulation

For BPSK modulation, the minimum distance between constellation points is $d_{\text{min}} = 2$, and for QPSK modulation the minimum distance between constellation points is $d_{\text{min}} = \sqrt{2}$. A plot of the scale factor versus the real axis of the ideal BPSK constellation is shown in Figure 14. Since this modification to the RLS uses a square-law metric to scale the weight update matrix, it is called the Square-Law DFE-TDAF or SDFE-TDAF for short. Using the notation of previous sections, the SDFE-TDAF algorithm is summarized in Table 4.

There are several drawbacks to the SDFE-TDAF algorithm. First, the algorithm must employ an initial training sequence to adapt the overall gain of the weights so that the ideal constellation may be used during blind periods of adaptation. This is necessary to prevent an unusually large or small input signal level from precluding weight adaptation. Second, the SDFE-TDAF may experience a "freezing" of the weights if the input signal level changes drastically during the blind adaptation. This fact is clearly seen in the weight update equation. Should the normalized error power exceed $\frac{d_{\text{min}}}{2}$ or move to a "dead spot" in the ideal signal constellation, then the weights will stop adapting or "freeze" to the previous value. Finally, the SDFE-TDAF may experience a slower convergence rate than
Table 4: Square-Law DFE-TDAF Algorithm

\[ W_{k-1}^o = 0 \cdot I \]
\[ R_{k-1}^{-1} = \eta \cdot I \]
\[ X_k(f_i) = i^{th} \text{ bin of FFT} \left[ x(k) \ x(k-1) \ \cdots \ x(k-N+1) \right]^T \]
\[ \hat{Y}_k = W_{k-1}^o X_k \]
\[ E_k = S_c - \hat{Y}_k \]
\[ Z_k = X_k^H R_{k-1}^{-1} \]
\[ Q = Z_k X_k \]
\[ V = \frac{1}{\mu + Q} \]
\[ \tilde{Z}_k = V \cdot R_{k-1}^{-1} X_k \]
\[ R_k^{-1} = \frac{1}{\rho} \left[ R_{k-1}^{-1} - \tilde{Z}_k Z_k \right] \]
\[ T = \left[ \frac{1}{N^2} \right] E_k^H E_k \]
\[ \Psi_k = \frac{1}{2} \left[ \frac{d_{\text{min}}}{2} - T \right] \left[ 1 + \frac{\frac{d_{\text{min}}}{2} - T}{\frac{d_{\text{min}}}{2} - T} \right] \]
\[ W_k^o = W_{k-1}^o + \Psi_k \cdot E_k \tilde{Z}_k^H \]
\[
\begin{bmatrix}
\hat{y}^a(k) & \hat{y}^a(k-1) & \cdots & \hat{y}^a(k-N+1)
\end{bmatrix}^T = \text{FFT}^{-1} \left[ \hat{Y}_k \right]
\]
\[
\begin{bmatrix}
\hat{y}^p(k) & \hat{y}^p(k-1) & \cdots & \hat{y}^p(k-N+1)
\end{bmatrix}^T = \text{FFT}^{-1} \left[ W_k^o X_k \right]
\]
the DFE-TDAF algorithm, particularly in well behaved environments. However, repeated simulations have proven that the SDFE-TDAF is superior to the DFE-TDAF in highly dynamic environments such as the mobile channel. In particular, the SDFE-TDAF can converge under conditions in which the DFE-TDAF experiences catastrophic failure. In order to alleviate some of the SDFE-TDAF's problems, a scaling function commonly used in neural networks can be employed.

3.3.3 The Exponential-Law DFE-TDAF Algorithm

The source of the weight "freezing" in the SDFE-TDAF lies in the discontinuity at the decision boundary. Any filter output which exceeds the decision boundary produce no weight adaptation. For BPSK signalling, this is equivalent to having a total error power greater than \( \frac{d_{\text{min}}}{2} = 1 \). Even filter outputs which exceed the boundary value radius to the correct side (i.e. a filter output of \( d_i \geq |2| \) on the real axis for BPSK) produce no weight adaptation. Clearly, this feature is unacceptable for highly dynamic environments. By replacing the square-law function with a continuous function, better tracking and adaptation can be obtained.

The most common adaptation function in neural networks is the exponential or Gaussian function [RHVH93]. If the average normalized error power is defined by Equation 108, then the scale factor \( \Psi_k \) can be defined as

\[
\Psi_k = e^{-\left[ \frac{t}{R_k^2} \right]}
\]

(111)

where \( R_k^2 \) is the normalization radius given by

\[
R_k^2 = \frac{d_{\text{min}}^2}{4 \ln p}
\]

(112)

The constant \( p \) represents the degree of influence which a boundary value point from the filter should have on the weight update equation. For instance, \( p = 0.01 \) implies that a filter output which is located at an error radius of \( \frac{d_{\text{min}}}{2} \) should have only 1% of the influence of a filter output located directly on a valid constellation point. A graph of Equation 112 is given in Figure 15 for a BPSK constellation with \( p = 0.20 \). For comparison, Figure 16 shows the effect of reducing \( p \) from 20% to 1% on the scale factor, \( \Psi_k \). Since the scale factor now follows an exponential curve, the new algorithm is designated the Exponential-Law DFE-TDAF or EDFE-TDAF for short. Notice that the scale factor \( \Psi_k \) represents a sort of "hill" in the complex constellation plane. By scaling the least squares solution at each iteration, the EDFE-TDAF slowly moves the weight vector to the vertex of the "hill." Figures 17 and 18 show the scale factor \( \Psi_k \) as a function of location in the complex plane.
Figure 15: Scale Factor for Exponential-Law DFE-TDAF ($p = 0.20$)
Figure 16: Comparison of Scale Factors for Exponential DFE-TDAF ($p = 0.01, 0.20$)
Figure 17: Scale Factor $\Psi_k$ versus the BPSK Constellation Plane
Figure 18: Scale Factor $\Psi_k$ versus the QPSK Constellation Plane
The EDFE-TDAF retains many of the desirable properties of the SDFE-TDAF, while removing the possibility of weight "freezing." Since the scale function is never zero, the EDFE-TDAF retains the best features of both gradient and least squares based algorithms. This is clearly seen in Figure 19 which compares the scale factors for both SDFE-TDAF and EDFE-TDAF ($p = 0.20$) algorithms. The EDFE-TDAF algorithm is summarized in Table 5. For very dynamic signals, the EDFE-TDAF behaves as a gradient algorithm, slowly moving the weight vector towards the optimal solution. In addition, the EDFE-TDAF is less sensitive to impulsive or transient interference than the DFE-TDAF. Once convergence is achieved, the EDFE-TDAF behaves in a manner similar to the DFE-TDAF where the scale factor is approximately unity.
Table 5: Exponential-Law DFE-TDAF Algorithm

\[ W_{k-1}^o = 0 \cdot I \]

\[ R_{k-1}^{-1} = \eta \cdot I \]

\[ X_k(f_i) = i^{th} \text{ bin of } \text{FFT} \begin{bmatrix} x(k) & x(k-1) & \cdots & x(k-N+1) \end{bmatrix}^T \]

\[ \hat{Y}_k = W_{k-1}^o X_k \]

\[ E_k = S_c - \hat{Y}_k \]

\[ Z_k = X_k^H R_{k-1}^{-1} \]

\[ Q = Z_k X_k \]

\[ V = \frac{1}{\rho + Q} \]

\[ \tilde{Z}_k = V \cdot R_{k-1}^{-1} X_k \]

\[ R_k^{-1} = \frac{1}{\rho} \left[ R_{k-1}^{-1} - \tilde{Z}_k Z_k \right] \]

\[ T = \left[ \frac{1}{N^2} \right] E_k^H E_k \]

\[ \Psi_k = e^{-\left[ \frac{x}{\rho s} \right]} \]

\[ W_k^o = W_{k-1}^o + \Psi_k \cdot E_k \tilde{Z}_k^H \]

\[ \begin{bmatrix} \hat{y}^a(k) & \hat{y}^a(k-1) & \cdots & \hat{y}^a(k-N+1) \end{bmatrix}^T = \text{FFT}^{-1} \left[ \hat{Y}_k \right] \]

\[ \begin{bmatrix} \hat{y}^p(k) & \hat{y}^p(k-1) & \cdots & \hat{y}^p(k-N+1) \end{bmatrix}^T = \text{FFT}^{-1} \left[ W_k^o X_k \right] \]
Chapter 4

Simulated Performance of the TDAF for DS-SS

In this chapter the performance of TDAF for DS-SS signals is explored through computer simulation. The purpose of the simulations is not to examine every scenario facing CDMA systems, but to verify that the TDAF is capable of combating the most common forms of interference to CDMA systems — namely background noise, multipath propagation, dissimilar modulation, and multiple access interferences.

In order for TDAF algorithms to prove useful in the mobile channel, the combined structure, adaptation algorithm, and adaptation criterion must exhibit fast convergence, guaranteed stability (or recovery capability), and a low computational count. Several trade-offs were made in the choice of an optimal TDAF for DS-SS in the mobile channel.

The natural choice for the TDAF structure is the DFT-based realization pictured in Figure 11. The use of this structure assumes that the system sampling rate, $f_s$, and the code repetition rate, $f_{code}$, are chosen such that a low computation implementation (i.e. FFT, Winograd) may be utilized in place of the DFT.

In order to achieve fast convergence and good tracking characteristics, the DFE adaptation criterion is used. Unfortunately, the DFE has notoriously poor startup performance, and can fail to converge if the initial decisions are incorrect. Thus, a combined adaptation procedure which begins with a short training sequence and continues with the DFE is used in place of a pure DFE.

The choice of an adaptation algorithm is critical to both the convergence rate and overall computational count of the optimal TDAF for DS-SS signal processing. The LMS and other gradient based algorithms have a very low computational count, but converge at a very slow rate. Additionally, the gradient algorithms tend to have poor tracking performance. The
Normalized and Accelerated LMS algorithms exhibit a better convergence rate, but can exhibit a high degree of misadjustment after convergence. Since the algorithms developed in this thesis are primarily designed for interference cancellation at a base station, a significantly higher computation count can be tolerated. The RLS adaptation algorithm, while requiring a substantial computational count, represents the best choice for fast convergence and tracking. Thus, Haykin's second version of the RLS [Hay91] adaptation algorithm combined with Treichler's notation [TJL87] is used.

For CDMA mobile communications systems, the TDAF can be integrated with traditional base station hardware as a "black box" between the IF-to-baseband downconverter and DS-SS demodulator. This scenario is depicted in Figure 20. The major benefit of this structure is that the conventional hardware for DS-SS demodulation can be retained, with some additional signal processing hardware.

All simulations in this thesis were performed on Sun Microsystems SPARC 10/40 workstations using the MATLAB signal processing software and custom scripts. Samples of the software scripts are included in Appendix A. For all simulations, reasonable parameters were chosen to closely approximate real-world scenarios. In particular, the message modulation, \( d(t) \), is chosen to be BPSK with a bit rate of 16.384kHz. This number represents roughly twice the 8kHz bit rate required for the current voice coder used by the digital TDMA cellular standard [DTC90], and it should accommodate powerful error correction coding and signalling information.

The processing gain is chosen to be \( S = 64 \) — i.e. there are 64 chips per message band. In order to maximize the performance of the conventional DS-SS demodulator, the 64 chip signature sequences employed in the simulations have characteristics similar to 63 chip Gold codes. For a preferred pair of maximal length sequences, 63 chip Gold codes can be generated with a maximum cross correlation of 15 chips for any arbitrary phase shift between separate codes [Dix84]. However, for length 63 Gold code sequences there exist only two preferred generating polynomials. This means that the set of all possible signature sequences is limited to a total of 65 unique Gold codes. Such a low number of unique codes is clearly unacceptable for CDMA systems. By utilizing both preferred and non-preferred generating polynomials, the maximum expected cross correlation is increased to 23 chips, but the available set of signature sequences becomes very large (in excess of 250,000) [Dix84].

For the simulations in this chapter, a set of 128 unique signature sequences are used, each having a length of 64 chips. The maximum measured cross correlation between any two codes is 22, and the maximum autocorrelation sidelobe of any single code is 16. Thus, while the length 64 codes are not Gold codes in the strict sense, they have characteristics which
Figure 20: Addition of DFE-TDAF to Typical CDMA Base Station — Single User Demodulation Shown
are similar to length 63 Gold codes generated from preferred and non-preferred pairs of generating polynomials.

All signals in the computer simulations are represented in complex baseband form. For each DS-SS signal, the effective chip rate is

\[
f_{\text{chip}} = 64 \cdot f_{\text{data}} = 64 \cdot 16.384\text{kHz} = 1.048576\text{MHz}
\]

Thus, the total signal bandwidth is roughly 2.1MHz (1st null-to-null bandwidth of DS-SS signal). Since all signals are represented in analytic form, the complex sample rate of the simulations is \(f_s = 2.079152\text{MHz}\), which corresponds to 2 samples per real chip after demodulation.

The structure of the TDAF used for cyclostationary exploitation is identical to Figure 11. Since only the code rate periodicities of the DS-SS signals are exploited, and since the bin spacing equals the code rate cycle frequency, every frequency bin of the output FFT is formed from a weighted combination of every bin of the input FFT. In the notation of Chapter 3, this corresponds to a square weight matrix. For code-on-pulse or modulation-on-symbol (MOS), the code rate cycle frequencies for the DS-SS signals are spaced at

\[
\alpha_{\text{code}} = \alpha_{\text{data}} = \frac{1}{T_{\text{data}}} = 16.384\text{kHz}
\]

Thus, the number of points in the analysis and synthesis DFTs must be chosen such that the bin centers are separated by an amount equal to the code rate cycle frequency, \(\alpha_{\text{code}}\). Recall from the definition of the DFT that bin spacing is given in a normalized sense by

\[
\Delta f_i = \frac{1.0}{N}
\]

Thus, the number of points required for the input and output DFTs of Figure 11 is

\[
N = \frac{1.0}{f_{\text{code}}} (f_s) = \frac{1.0}{\frac{2.097152\text{MHz}}{16.384\text{kHz}}} = 128
\]

for these simulations. Notice that the sampling rate, chip rate, and code rate are integrally related for optimal time dependent processing. It is also useful to note that the required DFT size is a power of 2; thus, the computationally efficient FFT may be used in place of the DFT to exploit spectral correlation. In general, the code repetition rate should be
chosen so that the number of bins in the DFTs correspond to a number suitable for an efficient implementation algorithm.

The remainder of this chapter documents the performance of the TDAF for DS-SS in a variety of environments. In all simulations, a DFE-based adaptation criterion is used, preceded by a 20 symbol random training sequence used for weight vector initialization. The determination of an adequate number of training symbols is critical to the performance of the DFE-TDAF. For stationary environments, the use of a long training interval can allow the TDAF to achieve incredible results (up to 200% of FDM/TDM capacity); however, the mobile channel is rarely stationary. Even when the mobile user remains in a fixed location, the surrounding environment is constantly changing, especially in urban settings. However, the strongest influences on RF reflection, diffraction, and absorption tend to be large stationary objects such as buildings, mountains, water, and so forth. Thus, the primary factor in mobile environment changes is the movement of the mobile user. To a first order approximation, TDAF tracking of the dynamic channel is tantamount to tracking of the mobile user’s movement.

If we assume that the perceived environment of the mobile user is stationary for movements of one-eighth wavelength or less ($\frac{\lambda}{8}$), then mobile movement within a sphere of radius

$$\lambda = \frac{1}{8} \cdot \frac{c}{f_c} = 41.667 \times 10^{-3} \text{ meters}$$  \hspace{1cm} (117)

will have minor effects on the received signal parameters at $f_c = 900$MHz. Since the system sample rate is 2.097MHz and the data rate is 16.384kHz, there are 128 samples per data baud after despreading. If the maximum mobile velocity is 75mph (33meters/second), then the minimum duration of perceived mobile stationarity is $1.2626 \times 10^{-3}$ seconds. In the simulations, this time span corresponds to roughly 2648 samples or 20.6 data bauds. Thus, the minimum training interval for the TDAF is chosen to be 20 symbols.

Granted, the above calculations represent a simplistic model of the mobile channel; however, the calculations insure that the parameters of the DFE-TDAF are reasonable for accurate tracking in a highly dynamic environment. Again, the goal is to establish the ability (or lack of ability) of the TDAF to mitigate the various forms of interference encountered by CDMA systems. The degree to which specific systems meet this goal will need to be determined experimentally.

### 4.1 Performance of the TDAF in Conventional Interference

All CDMA systems which operate in the mobile channel experience the classic forms of interference. These include background noise, multipath propagation, and in-band interference
from surrounding systems. The degree of interference from a specific type of interference depends heavily on the surrounding topography, allowable transmission powers, and degree of spectrum regulation. In this section, the performance of the TDAF for DS-SS is examined in background noise, multipath, and in an environment with dissimilar modulation. TDAF mitigation of multiple access interference is given the most attention in this chapter, and the performance results for CDMA systems are presented in Section 4.2.

4.1.1 Interference from Background Noise

Traditionally, background noise has primarily presented a problem to deep space communications and certain sonar applications. However, most CDMA systems operate at a pre-despread SWNR ranging from the negative to the moderately positive. Reed [Ree87] has demonstrated the utility of the TDAF in environments dominated by Gaussian noise. Since Gaussian noise is a purely stationary process, it exhibits no spectral correlation [Gar87b]. Thus, cyclostationary processing is able to exploit the spectral correlation of the signal of interest (SOI) in order to extract it from Gaussian noise.

Figure 21 shows the mean squared error for the pre-despread DS-SS signal after TDAF processing. Figure 22 shows the pre-despread SWNR of the DS-SS signal versus the pre-despread SWNR of the signal after TDAF processing. The pre-despread SWNR is used in the simulations to provide a rough measure of the estimated improvement in system performance due to TDAF processing. It is the pre-despread SNR which is most useful in link budget calculations. Notice that the improvement in pre-despread SNR is approximately given by the processing gain of the DS-SS signal. This is both intuitively and theoretically satisfying since the TDAF has $N$ spectrally correlated sources to exploit for the estimation of each bin. Thus, the improvement for any single bin must be directly proportional to $N$.

These results are similar to the theoretical performance predicted by Reed [Ree87] for other forms of digital modulation such as BPSK. The basic justification for this performance rests in the theory of optimal time dependent filtering and is beyond the scope of this thesis; however, the linear dependence of input and output SNRs is typical of TDAF performance for other digital modulations. If conjugate spectral symmetry of the DS-SS signal is exploited, up to 3dB of additional improvement can be realized; however, this moderate gain would be at the expense of a tremendous increase in computation count. As previously mentioned, the exploitation of conjugate symmetry requires a doubling of the input vector size and a tremendous increase in the corresponding computation count of the algorithm.

The pre-despread SWNR is used in Figures 21,22 to provide a baseline measure of improvement in system performance. It is a simple matter to show that the post-despread
Figure 21: MSE for Pre-Despread DS-SS Signal after TDAF vs. Received SWNR
Figure 22: Pre-Despread SNR Before and After TDAF Processing of DS-SS Signal
SWNR after matched filtering will be equal to the pre-despread SWNR out of the TDAF plus the processing gain. Thus, for an actual processing gain of 18dB ($S' = 64$), TDAF processing increases the perceived processing gain of a single user to over 36dB. This increase in available processing gain allows the TDAF to reject many form of interference which traditional correlation/matched filter receivers cannot.

Figure 23 shows the convergence rate of the TDAF using the RLS adaptation algorithm ($\rho = 0.97$). For the plot, convergence rate is defined as the number of iterations required for the weight matrix, $W_k^2$ to converge to within 10% of the minimum MSE solution. This plot is critical in the design of a CDMA system which incorporates any of the RLS-TDAF based algorithms, for it aids in the determination of an adequate training interval and system SWNR set point. While the same degree of improvement from TDAF processing is experienced at all pre-despread SWNRs, the convergence and tracking rates are by no means insensitive to the pre-despread SNR of the input signal. The convergence rates of the RLS-TDAF are essentially equal for pre-despread SWNRs greater than 5dB. However, the presence of a stationary process such as Gaussian noise greatly increases the convergence time of the algorithm, especially when the DS-SS signal and interference noise powers are approximately equal. While the performance improvement gained by TDAF processing is independent of the received noise level, the convergence rate of the adaptive filter is a strong function of received SNR. From Figure 23, it is clear that a 20 symbol training interval requires a pre-despread SWNR of at least 5dB for the weights to fully converge during the training interval. Thus, the choice of an optimum system SWNR set point and training interval becomes a tradeoff between convergence/tracking rates and the available mobile transmitter power.

For most modern CDMA systems, the pre-despread SWNR for optimal convergence is easily achieved. Given the elaborate power control algorithms of current systems, obtaining a fixed signal to background noise level across most users represents a requirement already satisfied. Setting the minimum received SWNR to 5dB is a trivial adjustment to most systems; however, the effects of such an increase on adjacent cell propagation and mobile battery life prove more demanding. For specific systems, the tradeoffs between mobile transmit power, received SWNR, and convergence rate of the TDAF will need to be examined in greater detail.

### 4.1.2 Interference from Multipath Propagation

Perhaps the best justification for the use of DS-SS in the mobile channel rests in the multipath nature of the channel. Multipath propagation creates time delays between arriving
Figure 23: Convergence Rate of RLS-TDAF for DS-SS
wavefronts of the RF carrier. The result is a combination of constructive and destructive phasor components which make portions of signal spectrum appear to fade during transmission. For narrowband modulation schemes, this frequency selective fading can result in deep fades and signal dropouts during transmission [Lee82]. However, many DS-SS systems are able to exploit the time diversity and achieve interference rejection [oDCC92]. The TDAF is also able to exploit multipath propagation under certain circumstances.

The mathematical description of multipath propagation can be understood in terms of the idealized channel impulse response,

$$ h(t) = A_0 \delta(t) + A_1 \delta(t - \tau_1) + A_2 \delta(t - \tau_2) + \cdots $$

where $A_i$ represents the gain and $\tau_i$ represents the time delay of the $i$th path with respect to the receiver. If the transmitted signal is given by $x(t)$, then the received signal is given by

$$ r(t) = x(t) \circledast h(t) $$

$$ = x(t) \circ [A_0 \delta(t) + A_1 \delta(t - \tau_1) + A_2 \delta(t - \tau_2) + \cdots] $$

$$ = x(t) \circ A_0 \delta(t) + x(t) \circ A_1 \delta(t - \tau_1) + x(t) \circ A_2 \delta(t - \tau_2) + \cdots $$

where the \circ operator denotes convolution. Thus, the received signal consists of a sum of delayed and scaled versions of the transmitted signal. Taking the Fourier transform of both sides of Equation 119 and collecting terms yields

$$ R(f) = X(f) \times H(f) $$

$$ = X(f) \times \left[ A_0 e^{-j0} + A_1 e^{-j2\pi f\tau_1} + A_2 e^{-j2\pi f\tau_2} + \cdots \right] $$

The frequency domain representation of the multipath channel $H(f)$ is little more than a collection of rotating phasors. At certain frequencies the phasor components add coherently to form a better representation of those frequencies at the receiver. At other frequencies the phasors add destructively, and a poor representation results. If the transmitted signal contains a high degree of spectral correlation, and the frequency selective fades occur at a spacing greater than the interval of spectral correlation, then TDAF processing can correct the corrupted signal. Such is the case for DS-SS signals at 900MHz with 16kHz data rates for most mobile environments.

Figure 24 shows the power spectrum of a DS-SS signal with the parameters described in the introduction of this chapter that has been corrupted by heavy urban multipath. The SWNR in the main lobe of the DS-SS signal is roughly 10dB. The constellation diagram of the pre-despread signal in Figure 25 clearly shows the distortion from the delayed components. In this simulation, a three-ray model is used to describe the channel impulse.
In addition to the primary signal component, two delayed components with delays 0.91 and 2.40 microseconds, respectively, are received. The first of the delayed components has 7.50dB less power than the primary component, and the second delayed component has only 2.05dB less power than the primary component.

Figures 26 and 27 show the improvement in the pre-despread signal after RLS-TDAF processing. Notice that the TDAF utilizes the spectral components which are momentarily enhanced by the coherent phasor addition of the received signals to correct those momentarily corrupted. The effect is similar to the correction achieved by time diversity exploitation in the RAKE receiver and fractionally spaced adaptive equalizers.

Figures 28,29,30 and 31 show the performance of RLS-TDAF processing for a DS-SS signal corrupted by multipath. Again, a three-ray model is used for the simulation, but the channel impulse response is representative of a severe suburban multipath environment. Relative to the primary signal component, the delayed paths experience 1.80 and 7.40 microsecond delays with 6.02dB and 11.38dB attenuation, respectively. In both cases, TDAF processing is able to correct for the effects of multipath in the pre-despread signal.
Figure 25: Constellation of Pre-Despread DS-SS Signal Corrupted by Urban Multipath
Figure 26: Power Spectrum of DS-SS Signal After TDAF Processing
Figure 27: Constellation of Pre-Despread DS-SS Signal Corrupted by Urban Multipath After TDAF Processing
Figure 28: Power Spectrum of DS-SS Signal Corrupted by Suburban Multipath
Figure 29: Constellation of Pre-Despread DS-SS Signal Corrupted by Suburban Multipath
Figure 30: Power Spectrum of DS-SS Signal Corrupted by Suburban Multipath after TDAF Processing
Figure 31: Constellation of Pre-Despread DS-SS Signal Corrupted by Suburban Multipath After TDAF Processing
While TDAF processing is able to correct for multipath corruption in most cases, there exist several potential scenarios in which TDAF processing may fail to fully exploit the enhanced spectral components. In particular, if the carrier frequency is high enough for the frequency selective fades to occur at intervals smaller than the spacing of the fundamental code rate cycle frequency $\alpha_{\text{code}}$, then there exists no guarantee that an enhanced spectral component will be available for the correction. Also, and perhaps more importantly, the ability of the TDAF to track changes in the mobile environment is completely dependent upon the convergence rate of the filter. Attempts to exploit an excessive number of periodicities or conjugate spectral symmetry will increase the number of elements in the weight matrix to such a degree that even the convergence rate of the RLS-DFE algorithm may be insufficient to track the dynamic changes. In addition to the frequency selective fading, the convergence rate of the TDAF determines the ability of the TDAF to track and correct for changes in the overall received power of a particular user (i.e. correct for flat fading). If the received signal experiences either frequency selective or flat fading at a higher rate than the adaptation and tracking rate of the RLS-TDAF, then a substantial degradation in the processed signal can occur. However, for reasonable design parameters, TDAF processing represents a valid alternative to the RAKE receiver for diversity exploitation and to rapid transmitter power control.

4.1.3 Interference from Dissimilar Modulation

In the past several years, many researchers and companies have expressed interest in CDMA “overlay” systems for mobile communications. The basic idea of such systems is to transmit DS-SS signals at a low level so that narrowband signals such as FM-AMPS and digital TDMA in the same spectrum experience only a slight increase in the perceived noise floor. In particular, InterDigital Inc. [SLG93] has proposed such a system for New York city. Unfortunately, the system proposed by InterDigital Inc. simply applies a digital notch filter to any narrow band interference which is present in the received DS-SS signal spectrum. The result is that useful DS-SS signal energy is removed as well as the undesired narrowband signals. The system then depends on the processing gain of the DS-SS signal to compensate for the excised DS-SS signal energy.

TDAF processing of DS-SS signals allows correlated spectral energy to be utilized, even when that correlated spectral energy is corrupted by dissimilar interference. The energy from undesired signals is removed based on its lack of spectral correlation to the desired DS-SS signal components. Figure 32 shows a DS-SS signal with the parameters previously mentioned. The signal is represented in complex baseband form. Figure 33 shows the same
signal added to Gaussian noise for a pre-despread SNWR of 0dB, the combination of which is the noise floor for the environment in Figure 34. In Figure 34, twenty narrowband FM signals with parameters similar to FM-AMPS specifications (i.e. 30kHz channels, moderate deviation) have been added to the environment. Admittedly, the 30dB to 40dB carrier-to-noise-ratios (CNRs) for each of the twenty users is excessive, but the purpose of the simulation is to demonstrate the power of TDAF processing for DS-SS. Thus, the excessive power levels of the narrowband users represents a greater challenge to the TDAF processor.

The total signal to interference ratio for the DS-SS signal is -43.6dB while the total signal to white noise ratio for the pre-despread signal is 0dB. The processing gain of 18dB for the DS-SS signal is clearly insufficient to combat such high levels of interference. In fact, Figure 35 shows the constellation diagram after conventional DS-SS processing — correlation followed by a matched filter with perfect timing recovery. As expected, the constellation is representative of the spinning phases of the FM interferers. Figure 36 shows the constellation diagram for the matched filter output after TDAF processing, and
Figure 33: Power Spectrum of Environment (DS-SS and Noise)
Figure 34: Power Spectrum of DS-SS Signal with 20 FM-AMPS Interferers (SNWR = 0dB)
Figure 35: Constellation Diagram for Matched Filter Output without TDAF Processing

Figure 37 shows the signal spectrum after TDAF processing. Except for residual power from the DFT bin leakage, the FM interference has been completely removed. If optimal processing is desired, the analysis and synthesis DFT blocks in the FFT-TDAF structure must be replaced by a transmultiplexer or other analysis filter bank with better sidelobe and leakage performance [Fer85, Hol93].

If the transmultiplexer approach is not employed, then some form of notch filtering must preceded the DFT operation to reduce the bleed through between DFT bins. One technique for performing this spectral notching is to take a very high resolution DFT and zero the bins with excessive corruption. Figure 38 shows the power spectrum of the environment after spectral excision of the narrow band interferers. Since the 20 FM-AMPS signals occupy a total RF bandwidth of roughly 500kHz, the signal power in the DS-SS signal after notch filtering is reduced by roughly 25%. Figure 39 shows the constellation diagram for the matched filter output after spectral notching followed by conventional processing. Note that the reduced signal power is evident from the movement of the constellation centers off of the ideal \([-1, +1]\) constellation pattern. Figure 40 shows the output of the matched
Figure 36: Constellation Diagram for Matched Filter Output after TDAF Processing
Figure 37: Output Power Spectrum after TDAF Processing
filter after spectral notching and TDAF processing. In theory, the constellation diagrams for Figures 36, 40 would be the same if DFT bin leakage were not present in the FFT-TDAF structure employed.

In summary, the use of TDAF processing in DS-SS systems can eliminate many of the traditional forms of interference in the mobile environment. Furthermore, these gains are achieved through the exploitation of a knowledge of the parameters of the DS-SS signal of interest. No exploitation of a knowledge of the interference parameters is required to achieve dramatic results. In the next section, the performance of several blind TDAF filters will be examined in a multiple access environment.

4.2 Performance of the TDAF in CDMA Interference

Perhaps the best use of TDAF processing is in the mitigation of CDMA interference. Unlike most interference mitigation schemes for CDMA systems, TDAF processing can achieve
Figure 39: Constellation Diagram for Matched Filter Output with Spectral Notching
Figure 40: Constellation Diagram for Matched Filter Output with Spectral Notching and TDAF Processing
tremendous gains based on the knowledge of a single user. When implemented as in Figure 20, the performance of the TDAF for each user is independent of the performance of the TDAF for every other user. Furthermore, TDAF processing of the CDMA signals allows CDMA systems to approach the spectral efficiency of Frequency Division Multiplex (FDM) and Time Division Multiplex (TDM) systems.

In establishing the performance of the TDAF for multiple access interference, hundreds of Monte Carlo simulations were run in order to achieve statistically meaningful results. For the minimum BER categories, at least 50,000 data bauds were simulated for each user. The parameters for the DS-SS signals in the initial section of this chapter were used for each CDMA signal ($S = 64$, 16.384 kbits/sec data). The individual user signals were generated from random data, and the spreading sequences were representative of Gold codes. The SWNR specified for all simulations represents the pre-despread SWNR for each single user. The DS-SS signals are asynchronous with respect to other users and contain random phase offsets between users. Perfect timing recovery is assumed for both conventional processing and TDAF processing of the DS-SS signals. For each of the three blind TDAF structures, a 20 symbol random training sequence precedes blind adaptation.

The baseline measure of spectral efficiency is the traditional measure of information bits per second per Hz (bits/sec/Hz). For FDM/TDM systems using BPSK modulation with perfect Nyquist pulse shaping, the spectral efficiency is

$$\eta_{FDM/TDM} = \frac{0.5 \text{bits/second}}{1.0 \text{Hz}} = 0.5$$

(121)

Typical CDMA systems have spectral efficiencies on the order of 25% to 30% that of FDM/TDM systems for similar data signal modulation ($12.5% \leq \eta_{CDMA} \leq 15.0\%$) — slightly more for well designed spreading codes and power management [Dix84].

### 4.2.1 DFE-TDAF Performance in CDMA Interference

The basic performance measures for the evaluation of TDAF processing are Mean Squared Error (MSE), Bit Error Rate (BER), spectral efficiency, and constellation cluster variance. MSE is included in the evaluation to determine general trends in the BER. For all CDMA simulations, sufficient Monte Carlo simulations were performed to obtain an accurate measure of BER for $10^{-3}$ and greater. In order to observe trends below this BER, over 100,000 bits would need to be simulated per user for the low user categories. This degree of simulation was deemed excessive, and for the most part unenlightening. Although MSE and BER need not be closely related, experience has shown that each is useful for predicting trends in the other. Thus, while the behavior of the BER below $10^{-3}$ is questionable, the
behavior of the MSE provides insight as to the direction of performance in BER. In addition to traditional performance measures, parameters which are specific to the blind TDAF algorithms (such as catastrophic failure rate) are considered.

Figure 41 shows the average MSE across all users when conventional sequence correlation and match filtering is employed. The MSE is computed at the output of the matched filter (despread MSE). In the conventional approach, the MSE at the output of the filter is dominated by multiple access interference for pre-despread SWNRs of 0dB,10dB,20dB, and 30dB having any number of users greater than 10. With DFE-TDAF processing, the MSE is dominated by multiple access interference after 30 users for all SWNRs greater than 10dB (see Figure 42). ¹

Figure 44 shows the average BER using conventional processing. In order to verify the validity of the computer simulations, a plot of average BER versus the number of

¹The 0dB SWNR MSE curve for TDAF processing does not accurately reflect the true MSE. Most of the MSE for the 0dB SWNR curve occurs because the required number of iterations for convergence exceeds the 20 symbol training interval. The result is that the DFE adaptation begins before the weights have fully converged, and catastrophic failure results in some of the users. This effect is more clearly seen in the BER plots.
Figure 42: MSE for DFE-TDAF Processing (SNR = 0,10,20,30 dB)
simultaneous users was generated for a processing gain of $S = 64$ using an analytical model developed by Holtzman [Hol91]. Holtzman's model assumes well-balanced codes and perfect power control are employed. The results are shown in Figure 43. A comparison of the BER determined by simulation and the BER from the analytical model shows relatively good agreement. Again, for more than 10 users, the average BER for conventional processing is primarily a function of the multiple access interference. For an average BER of $10^{-3}$, a CDMA system using conventional processing can support only 17 users or 27% of the capacity of an FDM system.

Figure 45 shows the average BER with TDAF processing. Notice that the BER curve is approximately a step function at 40 to 50 users. This feature is indicative of the catastrophic failure associated with the DFE. In fact, the BER for 0dB SWNR represents a total lack of convergence for almost all user categories. Figures 46,47 demonstrate the catastrophic failure of the DFE algorithm even more clearly. These plots show the average BER for the users having the worst performance across simulations. Comparing Figure 44 with
Figure 44: BER for Conventional Processing (SNR = 0, 10, 20, 30 dB)
Figure 45: BER for DFE-TDAF Processing (SNR = 0,10,20,30dB)

Figure 46 we see that the performance of the worst user using conventional processing is essentially equal to the general performance of all users. This is expected since the codes are well-balanced. However, a comparison of Figure 45 with Figure 47 quickly reveals that there exists at least one catastrophic failure for almost every user category.

Figure 48 demonstrates the deceptive nature of the average BER curve for the DFE-TDAF. By plotting the average BER for each user in the 64 user category, it becomes evident that a wide variance exists between the BER for individual users. While the average BER curve of Figure 45 shows the BER across all users to be approximately 6.5%, Figure 48 shows that many users have an average BER less than $10^{-4}$. It is the users that experience a failure to converge or catastrophic failure who artificially raise the average BER across all users. Figure 49 shows the percentage of catastrophic failures versus number of users for several pre-despread SWNRs. Notice, however, that there are no catastrophic failures when there are 48 or fewer simultaneous users and the pre-despread SWNR is greater than 10dB. For comparison, Figures 50,51,52,53,54 show the constellation diagrams for the 48 user category. The constellation diagrams represent the output of the matched filters after
Figure 46: BER for Worst User in Each Group, Conventional (SNR = 0, 10, 20, 30 dB)
Figure 47: BER for Worst User in Each Group, DFE-TDAF (SNR = 0, 10, 20, 30 dB)
Figure 48: Average BER for Each User (SWNR = 30dB, # Users = 64)
Figure 49: DFE-TDAF Catastrophic Failure Rate (SNR = 0, 10, 20, 30 dB)
Conventional and TDAF processing and do not include users that experience a catastrophic failure or a failure to converge.

Obviously, DFE-TDAF processing can achieve tremendous gains in user capacity compared to conventional processing. However, the DFE-TDAF is subject to both catastrophic failures and convergence rate limitations which preclude its use for lower SWNRs. In order to improve the gains of TDAF processing, more robust algorithms such as the SDFE-TDAF or EDFE-TDAF are required.

4.2.2 SDFE-TDAF Performance in CDMA Interference

In order to improve the robustness of the DFE-TDAF, the Square-Law DFE-TDAF can be employed. The total computation count of the SDFE-TDAF is slightly more than the computation count for the DFE-TDAF, but the performance gains are significant. Figure 55 shows the MSE performance of conventional processing, and Figure 56 shows the MSE performance of the SDFE-TDAF. For fewer than 48 users, the SDFE-TDAF produces...
Figure 51: Constellation Diagrams for All 48 Users (Overlaid) — TDAF Processing
Figure 52: Constellation Diagrams for All 48 Users (Overlaid) — TDAF Processing
Figure 53: Constellation Diagrams for 47 of 48 Users (Overlaid) — TDAF Processing
Figure 54: Constellation Diagrams for 18 of 48 Users (Overlaid) — TDAF Processing
over three orders of magnitude improvement to conventional processing. Also, a quick comparison of Figures 42 and 56 reveals that the SDFE-TDAF is able to maintain performance even when the DFE-TDAF experiences catastrophic failure. This performance increase is even more dramatic in the BER plots of Figures 57, 58.

The clearest indicator, however, of the performance gain of the SDFE-TDAF is the plot of catastrophic failure rates given in Figure 59. The SDFE-TDAF is well-behaved in that fewer catastrophic failures occur at or below FDM capacity (64 users). Also, the onset and progression of catastrophic failures is much more linear than the DFE-TDAF. This reduction in catastrophic failures and stability of convergence allows the SDFE-TDAF to achieve a lower cluster variance among all users than the DFE-TDAF. Figure 60 is a plot of the constellation diagrams of the matched filter outputs for each member of the 48 user category. Comparison of Figures 50, 53 and 60 shows the dramatic decrease in cluster variance which the SDFE-TDAF maintains across all 48 users.

In order to test the performance of the SDFE-TDAF against variations in power levels, the power among users in each category were randomly assigned to match a desired power
Figure 56: MSE for SDFE-TDAF Processing
Figure 57: BER for Conventional Processing
Figure 58: BER for SDFE-TDAF Processing
Figure 59: SDFE-TDAF Catastrophic Failure Rate
Figure 66: Constellation Diagrams for All 48 Users (Overlaid) — SDFE-TDAF Processing
variance between users. In traditional terms, this variation is referred to as the "near-far" problem of CDMA systems. For conventional processing, the near-far problem translates directly into a loss in system user capacity. Figures 61,62 show the average MSE for conventional and SDFE-TDAF processing when the power variance among users is 0dB, 1dB, 2dB, and 3dB. Figures 63,64 show how the power variance among users affects the average BER for each user category. Notice that a 3dB variance in power reduces the capacity at BER = 10^{-3} from 17 users to 8 users for conventional processing. However, the 3dB variance has an almost imperceptible effect on SDFE-TDAF processing. Thus, SDFE-TDAF processing is quite effective in combating the near-far problem of CDMA systems.

Figure 65 is a plot of the catastrophic failure rate as a function of the number of users and power variance among users. Notice that the "near-far" problem has little effect on the performance of the SDFE-TDAF. Interestingly, small levels of power variation among users appear to produce a slight decrease in the mean catastrophic failure rate. Although

---

The term "power variance" indicates the variance on a log-normal distribution from the mean received power. For example, a power variance of 3dB among users implies that 99% of the users have a received power level which is within ±5.2dB of the average received power.
Figure 62: MSE for SDFE-TDAF Processing (Power Variance = 0,1,2,3 dB)
Figure 63: BER for Conventional Processing (Power Variance = 0,1,2,3 dB)
Figure 64: BER for SDFE-TDAF Processing (Power Variance = 0,1,2,3 dB)

Mean SNR = 10 dB

Average BER (across users) vs. # Simultaneous Users
Figure 65: SDFE-TDAF Catastrophic Failure Rate (Power Variance = 0,1,2,3 dB)

an analytical explanation for this phenomenon is difficult, the trend seems to stem from the fact that a better estimation is available for the more powerful users. Thus, while catastrophic failures may occur, they will tend to occur for the weak users and not across all users in the CDMA system.

Although dramatic performance gains can be achieved through the use of SDFE-TDAF processing, several potential drawbacks exist. In particular, the “freezing” phenomenon discussed in Chapter 3 cannot be avoided. In order to maintain the performance gains of the SDFE-TDAF and eliminate potential “freezing”, the EDFE-TDAF may be employed.

4.2.3 EDFE-TDAF Performance in CDMA Interference

The Exponential-Law DFE-TDAF (EDFE-TDAF) is by far the most robust of the DFE-based TDAF algorithms; yet, this performance gain is accompanied by only a moderate increase in computation count over the DFE-TDAF. As previously mentioned, the EDFE-TDAF is a blind technique in the truest sense. For sample-by-sample iteration (in place of block iteration), it requires no training sequence or baud timing recovery; however,
the EDFE-TDAF block implementation for DS-SS signals requires at least baud timing recovery. The major drawback of the EDFE-TDAF is that the convergence rate may be slower than the SDFE-TDAF convergence rate for large border update values \( (p \geq 10\%) \). This is because incorrect decisions will have a weighted value greater than the zero value of the SDFE-TDAF. In fact for \( p \approx 1.0 \), the behavior of the EDFE-TDAF is almost identical to the DFE-TDAF.

Figures 66,67,68,69 show the average MSE and BER performance of the EDFE-TDAF compared to conventional processing. For perfect power control (power variance = 0dB), the performance of EDFE-TDAF processing is roughly equivalent to SDFE-TDAF processing. Figure 70 shows the percentage of users experiencing catastrophic failure when the EDFE-TDAF is used. Comparing Figure 59 with Figure 70 it is apparent that the EDFE-TDAF performance is slightly better than the SDFE-TDAF performance in terms of convergence stability and the prevention of catastrophic failure. This performance, however, is highly dependent on the correct choice of the border update parameter, \( p \). By choosing \( p \) sufficiently low, the catastrophic failure rate of the EDFE-TDAF can be greatly
Figure 67: MSE for EDFE-TDAF Processing
Figure 68: BER for Conventional Processing
Figure 69: BER for EDFE-TDAF Processing
reduced; however, the corresponding convergence time is dramatically increased. Repeated simulations have shown that $1\% \leq p \leq 20\%$ represents the best range of choices. The optimal choice of $p$ depends on the volatility of the RF environment, the interval between training symbols, and the desired rate of convergence.

Figures 71, 72, 73, 74 show the average MSE and BER performance of both conventional and EDFE-TDAF processing when power variances of 0dB, 1dB, 2dB, and 3dB exist between users. The conventional curves are almost identical to the simulations of SDFE-TDAF performance, as expected. Also, the effect of power variation on EDFE-TDAF performance is almost imperceptible. Thus, the EDFE-TDAF is also able to effectively mitigate the "near-far" problem of CDMA systems, without significant knowledge of the other users in the system.

Figure 75 shows the percentage of catastrophic failures as a function of the number of users and power variance between users for the EDFE-TDAF. As expected, the EDFE-TDAF catastrophic failure performance is a weak function of the power variance among users.
Figure 71: MSE for Conventional Processing (Power Variance = 0, 1, 2, 3 dB)
Figure 72: MSE for EDFE-TDAF Processing (Power Variance = 0,1,2,3 dB)
Figure 73: BER for Conventional Processing (Power Variance = 0, 1, 2, 3 dB)
Figure 74: BER for EDFE-TDAF Processing (Power Variance = 0,1,2,3 dB)
Figure 75: EDFE-TDAF Catastrophic Failure Rate (Power Variance = 0, 1, 2, 3 dB)
For mobile CDMA systems, the EDFE-TDAF represents a powerful tool to mitigate severe interference problems. For properly chosen parameters, the EDFE-TDAF can provide marked increases in noise rejection, multipath cancellation, and multiple access interference cancellation. Furthermore, these gains are achieved with no knowledge of the type or level of interference — only a knowledge of the code repetition rate and spreading sequence of the desired signal is required.
Chapter 5

Conclusions

The purpose of this thesis is to present new algorithms for time dependent adaptive filtering and to explore the performance of these algorithms for CDMA systems. There can be little doubt that TDAF processing can dramatically improve CDMA system capacity and performance for properly designed systems. For the first time, CDMA systems can approach the same spectral efficiency as FDM systems while maintaining the advantages of wideband modulation and universal frequency reuse. Furthermore, these gains are achieved independently of any knowledge of the interfering signals. There remain, however, several issues which require further investigation.

5.1 Potential Drawbacks

Perhaps the greatest drawback of TDAF processing for CDMA signals is that a relatively high pre-despread SWNR must be maintained in order for the convergence and tracking properties of the filter to accommodate the dynamic mobile channel. Alternatively, a low pre-despread SWNR can be tolerated if the training interval for the TDAF is increased. Initial simulations with a 500 symbol training interval and 0dB SWNR produced user capacities in excess of 150% that of FDM/TDM systems. However, the mobile channel is almost never stationary for such a long interval of time. Thus, a high pre-despread SWNR currently represents the best alternative to insure filter convergence. This limitation may force the TDAF adaptation algorithms presented in this thesis to be applicable only in micro-cell systems, stationary environments (wireless LAN), or systems where battery life is not a limitation (automobiles).

Another drawback of the DFE-TDAF based algorithms applied to DS-SS signals is that the computational requirements may force the algorithms to be feasible only at base stations.
However, this drawback is of minimal significance since the majority of interference is seen by the base station. Base-to-mobile transmissions can achieve perfect power balancing and orthogonal signalling which eliminates much of the interference seen by mobile users.

TDAF processing also requires that the mobile transmitters have extremely accurate on-board clocking for the chip, data, and code rates. Any variation in the chip rate clock can result in "off-bin-centering" of the desired periodicities. In other words, the fundamental code repetition cycle frequency ($\alpha_{code}$) can deviate from the frequency separation between DFT bins. Currently, there exist few techniques to compensate for such variation at the base station [RH93]. For chip rates of 1MHz, the necessary timing constraints are easily realized. For chip rates of 10MHz or greater, the required stability may not be realizable with economical hardware in the mobile units.

5.2 Summary of Results

Time dependent processing of DS-SS signals represents a source of potential gain for many CDMA systems, though not all. The degree and location of spectral correlation in DS-SS signals is a strong function of the chip rate, data rate, code repetition rate, and the relationship between these rates. For many military DS-SS systems and specifically the QualComm system [oDCC92], efficient TDAF processing cannot be realized. However, by carefully designing the chip rate, data rate, and code repetition rate, TDAF processing in CDMA systems can achieve remarkable results.

Although TDAF processing is very effective against background noise, the convergence rate of the adaptive filter suffers a marked decrease, especially in low or negative SWNR environments. For adaptive tracking in the mobile channel, TDAF processing with the blind algorithms derived in this thesis requires a positive pre-despread SWNR (i.e. $\frac{E_b}{N_0} > 0dB$). This criterion presents a real limitation for many CDMA systems. The investigation of more robust and computationally efficient algorithms may alleviate this problem for future systems.

Given a positive pre-despread SWNR and reasonable design parameters, TDAF processing can accurately track and mitigate the effects of multipath propagation. In many instances, TDAF processing functions as a frequency domain RAKE receiver, exploiting enhanced spectral diversity much as the RAKE receiver exploits time diversity.

The performance of TDAF processing is very good in interference environments which are dominated by dissimilar modulation formats. This category represents the traditional usage of TDAF processing, and is the primary focus of many cyclostationary processors. In particular, the TDAF is able to mitigate a high degree of FM-AMPS type interference.
This feature makes the TDAF highly attractive for "overlay" systems.

Perhaps the most impressive performance of the TDAF rests in its ability to effectively cancel multiple access interference without a knowledge of either the number or power distribution of other users. Previous interference rejection schemes have relied heavily on a knowledge of the other users in a CDMA system. TDAF processing exceeds the performance of many such schemes without a knowledge of the other users [RHMS90]. Furthermore, the failure of TDAF processing to improve any particular user's signal has no effect on the other users of the system. EDFE-TDAF processing allows properly designed CDMA systems to achieve at least 75% the spectral efficiency of FDM/TDM systems, with 100% performance in certain instances. Also, this degree of spectral efficiency maintains the universal frequency reuse which FDM/TDM systems do not. In short, TDAF processing of DS-SS signals allows CDMA systems to meet and exceed the requirements of current and future mobile communications systems.

5.3 Future Research

The application of cyclostationary processing to DS-SS signals is a relatively new topic. As such, there are many areas for further investigation. Specifically, the investigation of optimal polyphase spreading codes, transmultiplexer performance gains, time-dependent adaptive arrays, and more robust blind adaptation algorithms offers tremendous possibilities for increasing system capacity.

The spreading codes in this thesis were formed from real-valued \{-1, +1\} sequences. Qualcomm has realized an enormous increase in system capacity by utilizing polyphase spreading sequences [oDCC92]. There is no reason to believe that such sequences could not also improve the performance of DS-SS systems with TDAF processing. In fact, the use of polyphase spreading sequences can theoretically allow properly designed CDMA systems with TDAF processing to exceed FDM/TDM capacity for a given baseband modulation.

Regardless of the gains achieved by post receiver processing, optimal CDMA system performance will not be achieved without the use of adaptive antenna array techniques. Reed [RH93] has developed a time dependent adaptive array structure which incorporates the benefits of both the TDAF and adaptive array. This structure holds great promise in increasing CDMA system capacity and should be further explored.

To summarize, TDAF processing of DS-SS signals allows CDMA systems to achieve near-optimal performance. When combined with adaptive antenna technology and optimal code design, TDAF processing has tremendous potential for increasing user capacity and reception quality in future wireless communications systems. In light of the results in this
thesis, the continued investigation of these techniques is highly recommended.
Appendix A

MATLAB Scripts

The following scripts were written for MATLAB Version 4.2 on Unix platforms. All simulations were done on Sun Microsystems SPARC 10/40 workstations. The following lines of code are representative of the many scripts used for generation, processing, and data analysis. The complete list of scripts would require several hundred pages, and would provide little (if any) additional information. The missing scripts differ only in the file name specifications and sequence of executable statements — not in content.

%/ This script computes the BER and MSE for each of the N CDMA users
%/ in a signal generated by "cdmasig.m". The particular TDAF used
%/ for interference cancellation is called as a function to
%/ conserve memory and disk space.
% load group4_1dB_8.mat
%/ ----- 1) Begin by defining some commonly used vectors to save time
%/ numtap = input('Enter the number of frequency domain taps ....... ');
%/ tlength = input('Enter the length of the training sequence (20) ... ');

129
\%
numtap = 128;
tlength = 20;
dfac = spbaud;
ntot = max(size(signal));
\%
BERc = zeros(nusers,1);
BERm = zeros(nusers,1);
MSEo = zeros(nusers,1);
MSEn = zeros(nusers,1);
cdata = zeros(ntot,1);
ss = zeros(ntot,1);
dmatcho = zeros(nbauds,nusers);
dmatchn = zeros(nbauds,nusers);
\%
\% echo off
\% format bank
\%
\%
for iter = 1:nusers,
\%
iter
    junk = clock;
junk(2:6)
\%
\%
\% ------ 2) Next, find the tuning offset for each user and return to 0 Hz.
\%
    if fo(iter) == 0
        retune = signal;
    else
        retune = signal.*exp(-j.*2.*pi.*((fo(iter)/fs).*linspace(0,(ntot-1),ntot).'));
    end
\%
\%
\% ------ 3) Re-generate the upsampled signature sequence
\%
x = ones(spchip,1)*sequen(1:1chip,iter).';
rcode = reshape(x,(spchip*1chip),1);
x = scode(1:(spchip*lchip))*ones(1,nbauds);
ss = reshape(x,(spchip*lchip*nbauds),1);

% ----- 4) Re-generate the training sequence
%
x = scode(1:(spchip*lchip))*baud(1:tlength,iter)';
cdata = reshape(x,(spchip*lchip*tlength),1);
%
clear('x');
%
% ----- 5) Call the Frequency-Domain RLS TDAF Version 2 Weight vector training
%
rho = 1;
numtrain = spbaud*tlength;
[Wtrain] = train_tdaf_function(numtap,rho,numtrain,signal,cdata);
%
% ----- 6) Call the FD RLS TDAF with DFE for interference cancellation
%
rho = (0.01)^(1/numtap);
[yarls] = ddfe_tdaf_function(numtap,rho,spbaud,signal,scode,Wtrain);
%
%
% ----- 7) Generate the despread signal with interference cancellation
%
dspread = ss.*yarls;
%
clear('yarls');
%
% ----- 8) Integrate & Dump traditional (match filter) to get Correlate/MF data bits
%
[dmachn(:,iter)] = intdump(dspread,spbaud);
%
clear('dspread');
%
% ----- 9) Finally, calculate BER and MSE of Correlate/MF data bits
%
sstart = tlength;
sstop = nbauds;
%
err = sign(baud(:,iter)) - sign(real(dmachn(:,iter)));
bit_err = sum(sign(abs(err(sstart:sstop))));
%
BERn(iter) = bit_err/(sstop - sstart);
%
MSEn(iter) = mean((baud(sstart:sstop,iter) - dmatchn(sstart:sstop,iter)).^2);
%
% ---- 10) Generate the traditional despread signal
%
dsspread = ss.*signal;
%
% ---- 11) Integrate & Dump traditional (match filter) to get Correlate/MF data bits
%
[dmatcho(:,iter)] = intdump(dspread,spbaud);
    dmatcho(:,iter) = dmatcho(:,iter).*exp(-j*(pi/180)*poff(iter));
%
clear('ss','dspread');
%
% ---- 12) Finally, calculate BER and MSE of Correlate/MF data bits
%
sstart = tlength;
sstop = nbauds;
%
err = sign(baud(:,iter)) - sign(real(dmatcho(:,iter)));
bit_err = sum(sign(abs(err(sstart:sstop))));
%
BERo(iter) = bit_err/(sstop - sstart);
%
MSEo(iter) = mean((baud(sstart:sstop,iter) - dmatcho(sstart:sstop,iter)).^2);
%
end
%
save data BERo BERn MSEo MSEn dmatcho dmatchn
%
quit

::: ::::::::
/home/u1/rholley/thesis_scripts/cdmasiggen.m
::: ::::::::
% This routine generates multiple CDMA signals using well designed
% signature sequences (+1,-1) and BPSK baseband modulation.
% The signals are represented in complex baseband form, and a
% constant frequency and phase offset residual from tuning
% is assumed. Also, the script assumes 2 samples/chip and an
% integer number of samples/baud.
% Finally, only the signal parameters for each user and the combined
% signal is saved in order to conserve disk space.
%
% ***** 1) Initialize
%
% nusers = input('Enter the number of users ...................... '); %
% lchip = input('Enter length of signature sequences (64) .... ');
% spbaud = input('Enter # of samples per baud (128) .............. ');
% nbauds = input('Enter # of bauds desired ....................... ');
% SNR = input('Enter received SNR (dB) of SS signals ........ ');
% Avar = input('Enter the power variance (dB) among users ... ');
% foff = input('Enter the frequency offset variance (Hz) .... ');
%
nusers = 8
lchip = 64
spbaud = 128
nbauds = 10000
SNR = 30
Avar = 0
foff = 0
%
% ***** 2) Define frequency (Hz), phase (degrees), and amplitude (voltage)
% offsets of the complex baseband signals.
%
% fo = fix(sqrt(foff).*randn(nusers,1));
% poff = fix(360.*rand(nusers,1));
xpoff = (pi/180).*poff;
%
% AdB = sqrt(Avar).*randn(nusers,1);
A = 10.^-(AdB./20);
% % **** 3) Calculate the total # of samples, chip rate (Hz), and % data rate (Hz) based on the overall sample rate %
ntot = spbaud*nbauds
spchip = 2
fs = round((9192)*256)
fcs = fs/spchip
fd = fs/spbaud
%
%
% **** 4) Get the random signature sequences from "gold_codes.mat"
%
load gold_codes
%
sequen = gcodes(1:1chip,1:nusers);
%
%
% **** 5) Generate the random data bits and store in "baud" matrix 
%
baud = sign(randn(nbauds,nusers));
baud = sign(baud + 0.1);
%
%
% ++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++++% % % Now generate the signal for each user and form the total signal % % % ===================================================================================== % %

signal = zeros(ntot,1);
ruser = zeros(ntot,1);
k = linspace(0,(ntot - 1),ntot)';
j = sqrt(-1);

% for iter = 1:nusers,
% iter
%
%% 6) Generate the upsampled signature sequence and data sequence
%%
x = ones(spchip,1)*sequen(1:lchip,iter.');
scode = reshape(x,(spchip*lchip),1);
%%
x = scode(1:(spchip*lchip))*baud(1:nbauds,iter').';
cdata = reshape(x,(spchip*lchip*nbauds),1);
%%
%% 7) Generate the complex baseband signal
%%
ruser = A(1).*cos((2*pi*(fo(iter)/fs)).*k + rpoFF(iter)).*cdata ...
+ j.*A(1).*sin((2*pi*(fo(iter)/fs)).*k + rpoFF(iter)).*cdata;
%%
signal = signal + ruser;
%
end


%---------------------------------------------------------------
%


snr = 10^-(SNR/10);
%
signal = signal + sqrt(0.5/snr).*(randn(ntot,1) + j.*randn(ntot,1));
%


% clear('faxis','i','j','k','noise','snr','sstart','sstop','taxis','cdata')
clear('xi','xr','ttot','rpoFF','groups','scode','x','ruser','gcodes')
%
save group1_SNR30_8
%
quit


/home/u1/rholley/thesis_scripts/dfetdaf_function.m


function [yarls] = dfetdaf_function(N,Rho,smblng,xrls,drls,Wtrain);
%% This function implements a Blind Circular RLS Frequency-Domain TDAF
%% minimum MSE receiver and DFE structure for weight adaptation using
%% the RLS algorithm to update the N^2 weights.
%% The function assumes #weights = #samples/baud so that a 2-symbol
%% alphabet can be used (+1,-1). It can easily be modified for larger
%% # of weights, so long as #weights = integer*(#samples/baud).
%% The function assumes that the signal has been synchronously sampled,
%% so that the symbol timing is known.
%%
n = max(size(xrls));

% yrls = zeros(n,1);
% Yrls = zeros(N,1);
Drls = zeros(N,1);
Yrls = zeros(N,1);
Erls = zeros(N,1);
Zhat = zeros(N,1);
Z = zeros(1,N);

% Err1 = zeros(N,1);
Err0 = zeros(N,1);
T1 = 0;
T0 = 0;

% Sym1 = fft( drls(1:mmblng));
Sym0 = fft(-drls(1:mmblng));

% W = zeros(N);

% R = 10000.*eye(N);

% Q = zeros(1);
V = zeros(1);

% W = Wtrain;

% for k = 1:(n/N), 
bstart = (k-1)*N + 1;
bench = (k-0)*N;
Xr1s = fft(xr1s(bstart:bend));
Yr1s = W*Xr1s;
Err1 = Sym1 - Yr1s;
Err0 = Sym0 - Yr1s;
T1 = mean(Err1.*conj(Err1));
T0 = mean(Err0.*conj(Err0));
if T1 >= T0
    Err1s = Err0;
else
    Err1s = Err1;
end
Z = Xr1s'*R;
Q = Z*Xr1s;
V = 1/(Rho + Q);
Zhat = V.*R*Xr1s;
R = (1/Rho).*((R - Zhat*Z);
W = W + Err1s*Zhat';
yar1s(bstart:bend) = ifft(Yr1s);
end

::: :::::::::
/home/ui/rholley/thesis_scripts/ffdrls_tdaf.m
::: :::::::::
% This script implements the Circular Frequency-Domain TDAF using the RLS
% algorithm to update the N^2 weights
% %
% N = input('Enter the number of frequency domain taps ...... ');
Rho = input('Enter the exponential forgetting factor, rho ... ');
% xr1s = input('Enter the input signal vector name .............. ');
dr1s = input('Enter the training signal vector name ............ ');
% echo on
% n = max(size(xr1s))
% yr1s = zeros(n,1);
ypr1s = zeros(n,1);
Xrls = zeros(N,1);
Drls = zeros(N,1);
Yrls = zeros(N,1);
Erls = zeros(N,1);
Zhat = zeros(N,1);
Z = zeros(1,N);

W = zeros(N);

R = 10000.*eye(N);
Q = zeros(1);
V = zeros(1);

if exist('Wtrain') == 1,
W = Wtrain;
end

for k = 1:(n/N),
bstart = (k-1)*N + 1;
bend = (k-0)*N;
Xrls = fft(xrls(bstart:bend));
Drls = fft(drls(bstart:bend));
Yrls = W*Xrls;
Erls = Drls - Yrls;
Z = Xrls'*R;
Q = Z*Xrls;
V = 1/(Rho + Q);
Zhat = V.*R*Xrls;
R = (1/Rho).*(R - Zhat*Z);
W = W + Erls*Zhat';

yarls(bstart:bend) = ifft(Yrls);
yprls(bstart:bend) = ifft(W*Xrls);
end

echo off

clear('bstart','bend','Xrls','Drls','Yrls','Erls','Z','Zhat','Q','V')
clear('Wnrls','Rnew','xrls','drls')

::: ::::: :::
/home/u1/rholley/thesis_scripts/getdata.m
::: ::::: :::::
% This scripts loads in all the data, computes the useful information, % and stores info. in several new vectors.
%
% ber?avg = average (mean) BER for each user group
% ber?max = maximum BER (worst performance) in each user group
% ber?min = minimum BER (best performance) in each user group
%
% catthresh = BER threshold above which the DFE is assumed to % to have experienced catastrophic failure
%
% catfail = # of users who experienced catastrophic failure
% catfailr = ratio of catastrophic failures to # users in group
%
% bernocat = average BER for all users who did not experience % a catastrophic failure
%
% 1

berthresh = 1e-04
cathresh = 0.4
%
numuser = [8 16 24 32 48 64 80 96 112 128]';
%
beroavg = zeros(10,1);
beromax = zeros(10,1);
beromin = zeros(10,1);
%
bernavg = zeros(10,1);
bernmax = zeros(10,1);
bernmin = zeros(10,1);
%
catfail = zeros(10,1);
catfailr = zeros(10,1);
bernocat = zeros(10,1);
%
mseoavg = zeros(10,1);
msenavg = zeros(10,1);
%
msenocat = zeros(10,1);
%
%
load new_data_SNR10_8
%
beroavg(1) = mean(BERo);
beromax(1) = max(BERo);
beromin(1) = min(BERo);
%
bernavg(1) = mean(BERN);
bernmax(1) = max(BERN);
bernmin(1) = min(BERN);
%
mseoavg(1) = mean(abs(MSEo));
msenavg(1) = mean(abs(MSEn));
%
catfail(1) = sum(sign(sign(BERn - cathresh)+1));
catfailr(1) = catfail(1)/max(size(BERN));
%
if catfail(1) == max(size(BERN)),
bernocat(1) = cathresh;
else
mask = sign(1 - sign(BERN - cathresh));
bernocat(1) = sum(BERN.*mask)/(max(size(BERN)) - catfail(1));
end
%
%
load new_data_SNR10_16
%
beroavg(2) = mean(BERo);
beromax(2) = max(BERo);
beromin(2) = min(BERo);
bernavg(2) = mean(BERn);
bernmax(2) = max(BERn);
bermmn(2) = min(BERn);

mseavg(2) = mean(abs(MSeo));
msenavg(2) = mean(abs(MSen));

catfail(2) = sum(sign(sign(BERn - cathresh)+1));
catfailr(2) = catfail(2)/max(size(BERn));

if catfail(2) == max(size(BERn)),
bernocat(2) = cathresh;
else
mask = sign(1 - sign(BERn - cathresh));
bernocat(2) = sum(BERn.*mask)/(max(size(BERn)) - catfail(2));
end

load new_data_SWR10_24

beroavg(3) = mean(BERo);
beromax(3) = max(BERo);
beromin(3) = min(BERo);

bernavg(3) = mean(BERn);
bernmax(3) = max(BERn);
bermmn(3) = min(BERn);

mseavg(3) = mean(abs(MSeo));
msenavg(3) = mean(abs(MSen));

catfail(3) = sum(sign(sign(BERn - cathresh)+1));
catfailr(3) = catfail(3)/max(size(BERn));

if catfail(3) == max(size(BERn)),
bernocat(3) = cathresh;
else
mask = sign(1 - sign(BERn - cathresh));
bernocat(3) = sum(BERn.*mask)/(max(size(BERn)) - catfail(3));
end
%
%
load new_data_SNR10_32
%
beroavg(4) = mean(BERo);
beromax(4) = max(BERo);
beromin(4) = min(BERo);
%
bernavg(4) = mean(BERn);
bernmax(4) = max(BERn);
bernmin(4) = min(BERn);
%
mseoavg(4) = mean(abs(MSeo));
mseenavg(4) = mean(abs(MSen));
%
catfail(4) = sum(sign(sign(BERn - cathresh)+1));
catfailr(4) = catfail(4)/max(size(BERn));
%
if catfail(4) == max(size(BERn)),
bernocat(4) = cathresh;
else
mask = sign(1 - sign(BERn - cathresh));
bernocat(4) = sum(BERn.*mask)/(max(size(BERn)) - catfail(4));
end
%
%
load new_data_SNR10_48
%
beroavg(5) = mean(BERo);
beromax(5) = max(BERo);
beromin(5) = min(BERo);
%
bernavg(5) = mean(BERn);
bernmax(5) = max(BERn);
bernmin(5) = min(BERn);
%
mseoavg(5) = mean(abs(MSeo));
mseenavg(5) = mean(abs(MSen));
catfail(5) = sum(sign(sign(BERn - cathresh)+1));
catfailr(5) = catfail(5)/max(size(BERn));

if catfail(5) == max(size(BERn)),
bernocat(5) = cathresh;
else
mask = sign(1 - sign(BERn - cathresh));
bernocat(5) = sum(BERn.*mask)/(max(size(BERn)) - catfail(5));
end

load new_data SNR10_64
beroavg(6) = mean(BERo);
beromax(6) = max(BERo);
beromin(6) = min(BERo);

beravg(6) = mean(BERn);
bermax(6) = max(BERn);
bermin(6) = min(BERn);

mseavg(6) = mean(abs(MSEo));
msenavg(6) = mean(abs(MSEn));

if catfail(6) == max(size(BERn)),
bernocat(6) = cathresh;
else
mask = sign(1 - sign(BERn - cathresh));
bernocat(6) = sum(BERn.*mask)/(max(size(BERn)) - catfail(6));
end

load new_data SNR10_90
beroavg(7) = mean(BERo);
beromax(7) = max(BERo);
beromin(7) = min(BERo);
```matlab
\%
bernavg(7) = mean(BERn);
bermax(7) = max(BERn);
bermin(7) = min(BERn);
\%
msenavg(7) = mean(abs(MSEn));
msenavg(7) = mean(abs(MSEn));
\%
catfail(7) = sum(sign(sign(BERn - cathresh)+1));
catfailr(7) = catfail(7)/max(size(BERn));
\%
if catfail(7) == max(size(BERn)),
bernocat(7) = cathresh;
else
mask = sign(1 - sign(BERn - cathresh));
bernocat(7) = sum(BERn.*mask)/(max(size(BERn)) - catfail(7));
end
\%
\%
load new_data_SNR10_96
\%
beroavg(8) = mean(BERO);
beromax(8) = max(BERO);
beromin(8) = min(BERO);
\%
bernavg(8) = mean(BERN);
bermax(8) = max(BERN);
bermin(8) = min(BERN);
\%
msenavg(8) = mean(abs(MSEn));
msenavg(8) = mean(abs(MSEn));
\%
catfail(8) = sum(sign(sign(BERN - cathresh)+1));
catfailr(8) = catfail(8)/max(size(BERN));
\%
if catfail(8) == max(size(BERN)),
bernocat(8) = cathresh;
else
mask = sign(1 - sign(BERn - cathresh));
bernocat(8) = sum(BERN.*mask)/(max(size(BERN)) - catfail(8));
```

end
%
%
load new_data_SNR10_112
%
beravg(9) = mean(BERo);
bermax(9) = max(BERo);
bermin(9) = min(BERo);
%
bernavg(9) = mean(BERn);
bernmax(9) = max(BERn);
bernmin(9) = min(BERn);
%
mseavg(9) = mean(abs(MSEo));
mseavg(9) = mean(abs(MSEn));
%
catfail(9) = sum(sign(sign(BERn - cathresh)+1));
catfailr(9) = catfail(9)/max(size(BERn));
%
if catfail(9) == max(size(BERn)),
bernocat(9) = cathresh;
else
mask = sign(1 - sign(BERn - cathresh));
bernocat(9) = sum(BERn.*mask)/(max(size(BERn)) - catfail(9));
end
%
%
load new_data_SNR10_128
%
beravg(10) = mean(BERo);
bermax(10) = max(BERo);
bermin(10) = min(BERo);
%
bernavg(10) = mean(BERn);
bernmax(10) = max(BERn);
bernmin(10) = min(BERn);
%
mseavg(10) = mean(abs(MSEo));
mseavg(10) = mean(abs(MSEn));
%

catfail(10) = sum(sign(sign(BERn - cathresh)+1));
catfailr(10) = catfail(10)/max(size(BERn));

if catfail(10) == max(size(BERn)),
    bernocat(10) = cathresh;
else
    mask = sign(1 - sign(BERn - cathresh));
    bernocat(10) = sum(BERn.*mask)/(max(size(BERn)) - catfail(10));
end

beroavg = beroavg + berthresh;
beravg = bernavg + berthresh;
beromax = beromax + berthresh;
bermax = bernmax + berthresh;
beromin = beromin + berthresh;
bermin = bernmin + berthresh;
bernocat = bernocat + berthresh;

% clear('dmatchn','dmatcho','mask','BERn','BERo','MSEn','MSEo');

save new_graphs_SVR10_data

:::-------------------:::
/home/ui/rholley/thesis_scripts/fmgen.m
:::-------------------:::

% This routine generates an FM signal with
% fc = 32768 kHz (4 x 8192Hz)
% fs = 2^-21 kHz (16 x 8192Hz)
% fc = 32768
% fs = 2^-21
% Quant = 0
%
% 1) **** Generate the integrating filter for the message. Since an
% ideal integrator response is unreasonable, design is for an
% integrator over the audio bandpass of 40Hz to 4000Hz. Also,
% the DC response of an ideal integrator would tend to produce
% carrier drift in the final FM transmission, which is avoided
% with a "leaky" integrator.
%     ++++ The approach is to use a single-pole lowpass filter with
%     appropriate scaling to achieve integration over the -20dB/dec
%     cutoff region.
% echo on
%
num = [1.1852e-04    0]
den = [1    -0.99]
%
hfil = dimpulse(num,den,8192);
ffil = fft(hfil);
mfil = abs(ffil);
dbfil = 20*log10(mfil);
anfil = angle(ffil);
%
% 2) **** Now load in the audio signal (modulation signal)
%
% ------ This is for audio
mt = auread('/usr/deno/SOUND/sounds/laughter.au');
%ascale = max(abs(mt));
%mt = (1/ascale).*mt;
%
% ------ This is for random noise
%mt = randn(1024,1);
%ascale = max(abs(mt));
%mt = (1/ascale).*mt;
%
% ------ This is for a test sinusoid of freq. fmod (0<fmod<4000)
%fmod = 1000
%k = linspace(0,1023,1024)';
%mt = cos(2*pi*(fmod/8192).*k);
sound(mt)
%
% 3) **** Now integrate the signal with the integrating filter response
% developed in part 1)
intmt = filter(num,den,mt);

sound(intmt)

\( \% \) 4) **** Now interpolate the audio samples to match the overall
\( \% \) sample rate, fs
\( \% \)
intfac = round(fs/8192)
theta = interp(intmt,intfac);
llength = max(size(theta));

\( \% \) 5) **** Now generate the frequency modulated carrier
\( \% \)
Ts = 1/fs
Bta = 2000
\( \% \)
k = linspace(1,llength,llength)';
arg = (2*pi*fc*Ts).*k + (2*pi*Bta).*theta;
\( \% \)
A = 1
fmsig = A.*cos(arg);

\( \% \) 6) **** Finally, add in the necessary noise for the desired CNR.
\( \% \) Ideally, only the noise over the Carson's Rule FM signal
\( \% \) bandwidth should be computed, since that's the bandwidth
\( \% \) used in the optimal IF filter. For comparison purposes,
\( \% \) however, the CNR is derived using noise with flat PSD over
\( \% \) the entire 73728Hz (fs/2) bandwidth.
\( \% \)
\( \% \)
CNR = 30
cnr = 10^-(CNR/10)
ND = ((A^2)/2)/cnr
scale = sqrt(ND)
\( \% \)
\( \% \) CBW = 2*(0.5 + 1)*(4096)
\( \% \) No = ND/ CBW
\( \% \) requar = (fs/2)*No
\( \% \) scale = sqrt(requar)
noise = scale.*randn(Ilength,1);
fhmsig = fhmsig + noise;

% 7) **** Almost finally -- need to have option for quantization of
% the signal to simulate A/D process at output of receiver IF
% NOTE: Should set CNR above equal to infinity to accurately
% measure quantization noise.
%
if Quant == 8,
%
qscale = max(abs(fhmsig));
temp = round((128/qscale).*fhmsig);
qhmsig = (temp/128);
%
elseif Quant == 16,
%
qscale = max(abs(fhmsig));
temp = round((32768/qscale).*fhmsig);
qhmsig = (temp/32768);
end
%
8) **** Oh yea, clean up the garbage
%
clear('count','W','ffil','mfil','llength','k','cnr','ND')
clear('CBW','No','reqvar','scale','temp')
%
echo off

::: :::::::::
/home/u1/rholley/thesis_scripts/intdump.m
::: :::::::::

function [dmatch] = intdump(tempsig,dfac);
%% This fcn is used to filter and critically decimate (integrate & dump) % the highly oversampled bauds in the despread cdma signal. It % is useful for determining the bit error rates of signals. % This version breaks the (Lx1) column vector into a (N,M) matrix % where the columns represent individual bauds 

%% tempsig = input('Enter the signal vector to be decimated ... '); 
%% dfac = input('Enter the decimation factor ............... '); 
%%
%% n = max(size(tempsig)); 
%% M = floor(n/dfac); 
%%
%% x = reshape(tempsig,dfac,M); 
%% dmatch = (1/dfac).*sum(x).'; 

::: ::: ::: 

/home/rl/rholley/thesis_scripts/quaddemod.m 

::: ::: ::: 

%% Implements a Quadrature FM Demodulator with the analytic signal. 
%% The theory behind this demodulator is given in the report, 
%% but it basically functions just like the hardware 
%% demodulator which uses a 90 degree phase shifter at the 
%% RF frequency to get I & Q channels. 
%% Nice feature is that the phase is never "unwrapped" so numerical 
%% errors do not accumulate. 
%%
%%
%% **** 1) First, need to delay the analytic FM signal by 1 sample 
%%
%% echo on 
%%
%% sstop = max(size(cfm)) 
%%
%% dcfm = zeros(sstop,1); 
%%
%% dcfm(1:(sstop-1),1) = cfm(2:sstop,1);
% 2) New conjugate the analytic signal and multiply by delayed version
% dmcfm = conj(cfm).*dcfm;
%
% 3) Finally, take the argument to get the demodulated signal
% and subtract off the scaled carrier
% sig = angle(dmcfm) - ((fc*2*pi)/fs);
%
% 4) Well, almost finally -- rescale the signal and decimate it for
% comparison with the original audio signal
% sig = (fs/(2*pi*Sta)).*sig;
% ccsig = decimate(sig,intfac);
% echo off

::: home/u/rholley/thesis_scripts/rfde2_tdaf_function.m
:::
function [varls] = rfde2_tdaf_function(N,Rho,smblng,xrls,drls,Wtrain);
% This function implements a Blind Circular RLS Frequency-Domain TDAF
% minimum MSE receiver and modified DFE structure for weight
% adaptation using the RLS algorithm to update the N^2 weights.
% The DFE modification involves modification of the RLS weight update
% equation by a neural net radial basis type weighting function.
% The result is a DFE structure which can operate at significantly
% higher BER and in transient environments without catastrophic
% failure.
% The function assumes #weights = #samples/baud so that a 2-symbol
% alphabet can be used (+1,-1). It can easily be modified for larger
# of weights, so long as #weights = integer*(#samples/baud).
% The function assumes that the signal has been synchronously sampled,
% so that the symbol timing is known.
% The routine assumes BPSK (DS-SS) modulation; thus, dmin = 2. The
% boundary update value is chosen to be 20% --> perc = 0.20.
%
dmin = 2;
perc = 0.20;
Rb2 = -(dmin^2)/(4*log(perc));
%
n = max(size(xrls));
%
yrls = zeros(n,1);
%
Xrls = zeros(N,1);
Drls = zeros(N,1);
Yrls = zeros(N,1);
Erls = zeros(N,1);
Zhat = zeros(N,1);
Z = zeros(1,N);
%
Err1 = zeros(N,1);
Err0 = zeros(N,1);
T1 = 0;
T0 = 0;
Psi = 0;
%
Sym1 = fft( drls(1:smblnrg));
Sym0 = fft(-drls(1:smblnrg));
%
W = zeros(N);
%
R = 10000.*eye(N);
%
Q = zeros(1);
V = zeros(1);
%
W = Wtrain;
%
for k = 1:(n/N),
bstart = (k-1)*N + 1;
bend = (k-0)*N;
Xrls = fft(xrls(bstart:bend));
Yrls = W*Xrls;
Err1 = Sym1 - Yrls;
Err0 = Sym0 - Yrls;
T1 = (1/N^2)*(Err1'*Err1);
TO = (1/N^2)*(Err0'*Err0);
if T1 >= T0
Erls = Err0;
Psi = exp(-TO/Rb2);
else
Erls = Err1;
Psi = exp(-T1/Rb2);
end
Z = Xrls'*R;
Q = Z*Xrls;
V = 1/(Rho + Q);
Zhat = V.*R*Xrls;
R = (1/Rho).*((R - Zhat*Z);
W = W + Psi.*Erls*Zhat';
yrls(bstart:bend) = ifft(Yrls);
end

:::........................................
/home/u/xholley/thesis_scripts/sdfe_tdaf_function.m
:::........................................
function [yrls] = ddfe_tdaf_function(N,Rho,smb1ng,xrls,drls,Wtrain);

% This function implements a Blind Circular RLS Frequency-Domain TDAF
% minimum MSE receiver and DFE structure for weight adaptation using
% the RLS algorithm to update the N^2 weights.
% The function assumes #weights = #samples/baud so that a 2-symbol
% alphabet can be used (+1,-1). It can easily be modified for larger
% # of weights, so long as #weights = integer*(#samples/baud).
% The function assumes that the signal has been synchronously sampled,
% so that the symbol timing is known.
%
n = max(size(xrls));
% yarls = zeros(n,1);
% Xrls = zeros(N,1);
Drls = zeros(N,1);
Yrls = zeros(N,1);
Erls = zeros(N,1);
Zhat = zeros(N,1);
Z = zeros(1,N);
%
Err1 = zeros(N,1);
Err0 = zeros(N,1);
T1 = 0;
T0 = 0;
Psi = 0;
%
Sym1 = fft( drls(1:smblng));
Sym0 = fft(-drls(1:smblng));
%
W = zeros(N);
%
R = 10000.*eye(N);
%
Q = zeros(1);
V = zeros(1);
%
W = Wtrain;
%
for k = 1:(n/N),
bstart = (k-1)*N + 1;
bend = (k-0)*N;
Xrls = fft(xrls(bstart:bend));
Yrls = W*Xrls;
Err1 = Sym1 - Yrls;
Err0 = Sym0 - Yrls;
T1 = (1/N^2)*(Err1'*Err1);
T0 = (1/N^2)*(Err0'*Err0);
if T1 >= T0
Erls = Err0;
Psi = (1-T0)*(0.5)*(I + sign(1 - T0));
else
Erls = Err1;
Psi = (1-T1)*(0.5)*(1 + sign(1 - T1));
end
Z = Xrls'*R;
Q = Z*Xrls;
V = 1/(Rho + Q);
Zhat = V.*H*Xrls;
R = (1/Rho).*((R - Zhat)*Z);
W = W + Psi.*Erls*Zhat';
yarls(bstart:bend) = ifft(Yrls);
end

/*****
/home/u1/rholley/thesis_scripts/smooth.m
/*****
% %
% This script is useful for smoothing data curves. It forms an "moving
% average filter.
% tempsig = input('Enter the data vector to be smoothed .... ');
swin = input('Enter the smoothing window in samples ... ');
%
um = (1/swin).*ones(swin,1);
den = zeros(swin,1);
den(1) = 1;
%
ssig = filter(num,den,tempsig);
%
clear('tempsig', 'num', 'den');
%
/*****
/home/u1/rholley/thesis_scripts/train_tdaf_function.m
/*****
function [W] = train_tdaf(N,Rho,numtrain,xrls,drls);
This function simply adapts the Frequency-Domain Weights using the Circular RLS Frequency-Domain TDAF algorithm in order to train the weight vector. A modified version of Haykin's RLS Version 2 algorithm is used.

tstop = round(numtrain/N)*N;

xrls = zeros(N,1); Drls = zeros(N,1); Yrls = zeros(N,1); Erls = zeros(N,1); Zhat = zeros(N,1);
Z = zeros(1,N); W = zeros(N); R = 10000.*eye(N); Q = 0; V = 0;

for k = 1:(tstop/N),
bstart = (k-1)*N + 1; bend = (k-0)*N;
xrls = fft(xrls(bstart:bend)); Drls = fft(drls(bstart:bend)); Yrls = W*xrls;
Erls = Drls - Yrls;
Z = xrls'*R;
Q = Z*xrls;
V = 1/(Rho + Q);
Zhat = V.*R*xrls;
R = (1/Rho).*(R - Zhat*Z);
W = W + Erls*Zhat';
end
Bibliography


1993 IEEE Global Communications Conference. IEEE Communications Society, December 1993. Accepted for Publication.


Vita

Richard D. Holley was born in Roanoke, Virginia on December 24, 1967. He received the B.S. and M.S. degrees (with honors) in electrical engineering from Virginia Polytechnic Institute and State University in May of 1991 and December of 1993, respectively.

Since 1991, he has worked with the Mobile and Portable Radio Research Group at Virginia Tech on the development of a high capacity wireless system. His research interests include wireless communications, spread spectrum technologies, and interference cancellation techniques.