A Space Division Multiple Access System

An Analysis And Prototype Proposal

Ryan Kelly

ENEL 698 Final Project Report

Department of Electrical and Computer Engineering

The University of Calgary

December 17, 1997

Abstract

In the ongoing quest for high wireless communications bandwidth efficiency, adaptive arrays intended for the base station are being investigated as a means of allowing multiple users simultaneous access to a common frequency and time slot. The background theory for the corresponding spatial division multiple access signal processing problem is considered. A description of simulations undertaken for the receive array situation is given, and the results are presented and discussed. The results show that when comparing the LMS algorithm training time required for different numbers of users, the effect of the received SINR seems to dominate, in apparent contradiction to what the eigenvalue spread predicts. When examining the training time performance for a given number of users, the received SINR and eigenvalue spread are the dominant factors. A proposal for a prototype hardware implementation of a receive array system is given, and numerous aspects of its design are considered, including a discussion of the relevant characteristics of a physical antenna array design proposed for this system. The material in this project is considered new since there is not a significant body of work involving experiments with array processing hardware intended for spatial multiplexing, nor with the analysis and simulation of adaptive arrays in a fading environment.

Acknowledgements

I gratefully acknowledge assistance and support from Dr. R. Johnston and Dr. B. Petersen. Facilities and financial support were provided by Telecommunications Research Laboratories, Calgary, and by The Department of Electrical and Computer Engineering, The University of Calgary. I would also like to thank Grant McGibney and John McRory for the discussion and help they provided.

Contents

List of Figures

Chapter 1

Introduction

Wireless communications is becoming increasingly widespread. In particular, personal communications is presently in a growth explosion with the goal of a global personal communication network where a multitude of types of wireless devices will have uniform, global access to digital communications data. In high density use areas, there will be the need to extract as much utilization as possible from a given bandwidth by using multiplexing. The multiplexing techniques used or being introduced at present, consisting of time division multiple access (TDMA), frequency division multiple access (FDMA), and code division multiple access (CDMA), can each be augmented by space division multiple access (SDMA). This multiplexing technique employs an adaptive antenna array either at the portable unit, or more likely, at the base station. The introduction of this technique has recently become more attractive due to the decreasing cost of digital signal processing hardware, advances in adaptive signal processing theory, and the previously mentioned increasing capacity demand. This report assumes that the array is at the centrally located base station. An extensive list of references regarding array theory is found in [1].

A significant problem in existing multiuser communication systems is inter-user interference, which can severely degrade the performance and capacity. Even in practical CDMA systems, which are designed to allow simultaneous cochannel user trans-

missions, the varying delays of different users can induce non-orthogonal codes. Thus adaptive arrays would be useful to combat these side-effects in such systems by helping to minimize the effects of interferers [1].

An array-equipped base station can provide multiple access, where a number of portable users simultaneously access the base station and are considered mutual interferers. The number of users is limited by the number of elements in the array, and in practice by the numerical precision of the digital signal processors used with the array. In addition, the array can be used to suppress interference from other sources such as users in adjacent cells, users in other radio systems, or even other types of radiating devices, thereby allowing operation in a high interference environment. Adaptive arrays can also be used to reduce the effects of multipath fading (by providing diversity) and multipath delay spread (by nulling whichever signals are causing intersymbol interference). Whatever algorithm is implemented to perform the adaptation, it must be able to initially acquire the desired and interference signals and then track them in real time, a task which requires significant processing power since the environment is continuously changing, to various degrees of severity depending on the situation.

A receive array, intended for the reverse link, can observe its own output and adapt its spatial filtering to the propagation environment in order to suppress interference and enhance the desired signal(s). This situation has been extensively studied (see [2], [3], and [4] for references). Multiplexing for the forward link can be handled by the base station array as well; this situation is complicated by the fact that in general, in order for the array to adapt itself for transmission, the portables must provide feedback as to their reception environment. The latter situation has been studied [2].

Adaptive array algorithms can be classified into two main categories: spectral-based and parametric-based [1]. In the former, some spatial spectrum-like function is formed that characterizes the parameter of interest. This category can be further divided into beamforming techniques and subspace methods. Beamforming attempts to "steer" the array gain to the desired signal; unless the signals are plane or spherical

waves, the concept of a direction of arrival does not hold, but the essential method remains the same. Examples of this class of adaptive array are the Bartlett beamformer, which attempts to maximize the signal power from a given "direction", and the Capon algorithm, which minimizes the array output power subject to a linear constraint of some parameter of interest (such as direction of arrival). The LMS algorithm, which is discussed later in this report, is also a member of this class. The other division within spectral-based algorithms is subspace methods, in which spectral decomposition of the array autocorrelation matrix is used for the analysis. An example is the MUSIC algorithm, used for direction finding.

The above spectral-based methods are attractive due to their computational simplicity, but don't always result in sufficient accuracy. Parametric methods more fully exploit the underlying data model. These algorithms are more efficient and robust; however, they usually require a multidimensional search to adapt the array [1].

A different method of distinguishing the algorithm employed in an adaptive array is whether or not the algorithm operates in array space or beam space, the latter being the case when the array data are linearly transformed, perhaps to a reduced dimension space to ease processing speed requirements [5]. The transformation can also act as a spatial prefilter, which emphasizes a favorable angular location or some other spatial characteristic which can enhance the array's ability to reject interference [1]. Another example is where the transformed space consists of a set of orthogonal beams [6]; an advantage of this transformation is that the adaptive algorithm can work with real as opposed to complex values.

The particular array discussed later in this report is intended as an in-building personal communication system base station array. Thus it operates with a high user density in an indoor severe multipath fading environment. Only the reverse link, in which case the array receives, will be considered since as discussed later on, the forward link is usually a reciprocal problem in the indoor environment.

The remainder of this report will be divided as follows. Chapter 2 discusses the background theory for spatial array processing and spatial division multiple access, as

well as some related topics. Following that, Chapter 3 presents the simulation work performed and an analysis of the results obtained. Chapter 4 describes the design of the proposed hardware prototype of a receive array system, and Chapter 5 discusses potential future work pertaining to the simulations and hardware discussions. Chapter 6 summarizes the report, and finally Appendix A discusses the cross-correlation among the training sequences used in the simulations.

Chapter 2

Background

As was discussed in Chapter 1, an adaptive array can be used to select one received signal out of many, and the terms *desired signal* and *interfering signals* are used to distinguish them, respectively. A condition for this spatial selection is that the desired signal must have some spatial characteristic that distinguishes it from the interferers. In the case of plane wave reception, it could be direction of arrival. The array can also be used to simultaneously distinguish a number of signals, if more than one signal are considered desired signals. In this case the desired signals are considered mutual interferers, which can exist in addition to undesirable interferers. This chapter will discuss these topics, beginning with the theory of an adaptive array used to receive a single desired signal.

2.1 General Theory of Adaptive Spatial Arrays

Figure 2.1 shows a generalized receive adaptive array. The antenna elements are arranged in some spatial configuration, and the elements shown are assumed to include demodulators and digital samplers, so that the received signal is available in digital, complex baseband form. The distance between elements in the array is assumed to be

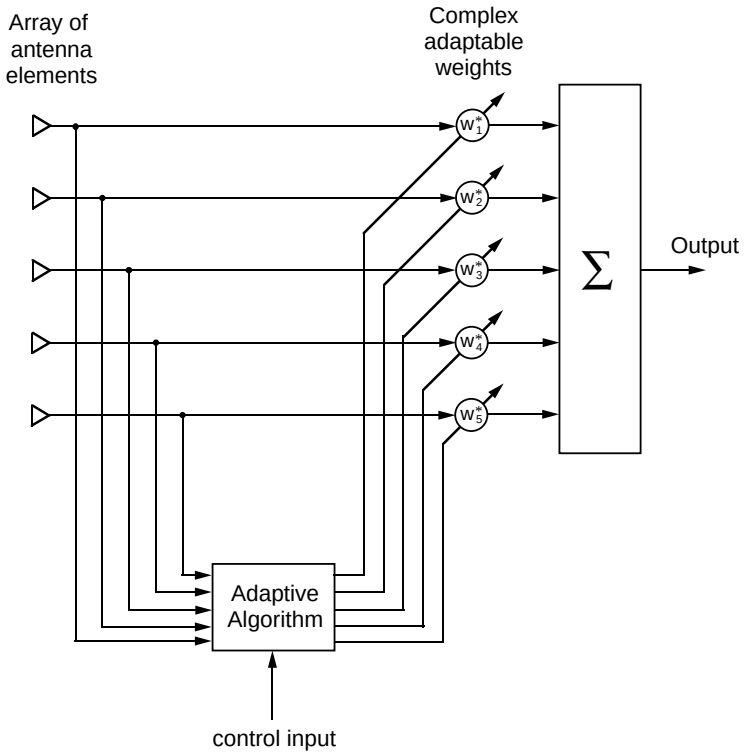


Figure 2.1: A Generalized Adaptive Antenna Array

about half of one wavelength, since this is true for the proposed array briefly described in Chapter 4.

A transmitter will emit a desired data signal that impinges on the array. A carrier frequency-dependent *propagation vector* $\mathbf{c}(\omega)$ is formed whose elements $c(\omega)_i$ are the complex channel gains from the transmitter to the ith element. In order to adequately describe the channel with a single vector, the assumption

$$\delta_{mp} \ll BW^{-1} \tag{2.1}$$

must hold, where δ_{mp} is the maximum differential delay due to multipath and BW is the signal bandwidth [2]. This condition holds in the indoor environment for bit rates less than about 1 Mbps [7]. A sample of the array at time n is called a snapshot of the array, and is defined as

$$\mathbf{u}_n = \mathbf{c}(\omega)d_n + \boldsymbol{v}_n \tag{2.2}$$

where d_n is the data symbol transmitted at time n and $\boldsymbol{v}_n$ is the vector of additive

noise at the array output. As is shown in Figure 2.1 the array snapshot is linearly combined by a complex weight vector $\mathbf{w}$, resulting in the complex estimated data symbol at time n

$$\hat{d}_n = \mathbf{w}^H \mathbf{u}_n \tag{2.3}$$

where H means Hermitian transpose.

In an interference environment the interfering signals, each with its own propagation vector, add with the desired signal at the array. The weight vector is determined by an algorithm that selects the desired signal from the interference. If plane wave reception is assumed, this may be done by choosing a $\mathbf{w}$ that steers the array to the desired signal, or if the propagation environment is unknown, an adaptive algorithm can be used in conjunction with a training sequence to determine a $\mathbf{w}$ that selects the desired signal. This report will emphasize an iterative adaptive algorithm, namely the least mean square (LMS) algorithm.

The LMS algorithm will choose $\mathbf{w}$ in order to minimize the mean square error, defined as

$$\epsilon(\mathbf{w}) = E\left[(d_n - \hat{d}_n)^2\right] \tag{2.4}$$

which means spatially selecting the desired signal and rejecting interference signals, optimized in a least squares sense. Figure 2.2 shows a typical array polar gain plot which results after $\mathbf{w}$ has been found. A plot such as this is generated by assuming some form of propagation such as plane or spherical wave fronts, and then examining the gain at all angles that results from the set of coefficients in $\mathbf{w}$. As can be seen from the figure, a large *lobe* of high gain has been steered in the direction of the desired signal, whereas *nulls* of zero gain (in practice the gain in a null will be non-zero due to finite precision arithmetic in the signal processor) have been placed in the directions of interfering signals. Since with an M–length vector $\mathbf{w}$ there are M degrees of freedom, a theoretical maximum of $M-1$ independent nulls can be placed to eliminate interferers.

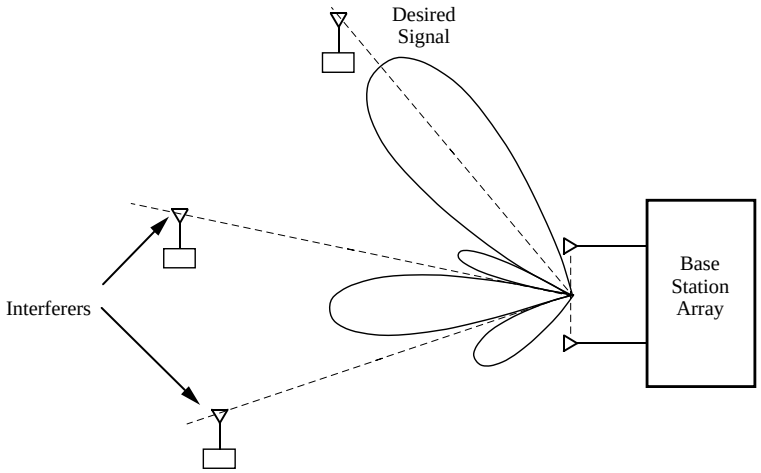


Figure 2.2: Adaptive Array Gain Plot Showing Nulled Interferers

A plane wavefront model can be used in situations where there is a large, relatively unobstructed path from a transmitter to the receive array. However, in the indoor environment considered in this report, the distance between transmitter and receiver is short, and cluttered with many objects which induce multipath, diffraction, shadowing, etc. The signals from a transmitter are not received as plane or spherical waves, but as a mixture of signals from many directions which add at the receive elements. Thus the concept of a directional gain plot is not valid [8], but this doesn't deter the adaptive algorithm from arriving at a solution, *as long as the propagation vector of the desired signal is sufficiently different from those of the interfering signals.* In an indoor system using many antennas in the array, the probability of not being able to suppress an interfering signal is very small [9].

2.2 Theory Behind Spatial Multiplexing

Figure 2.3 shows a block diagram of a multi-user communication system where spatial multiplexing (or SDMA) is employed to provide an orthogonal channel for each user. There are N users and M array elements. The frequency dependent channel

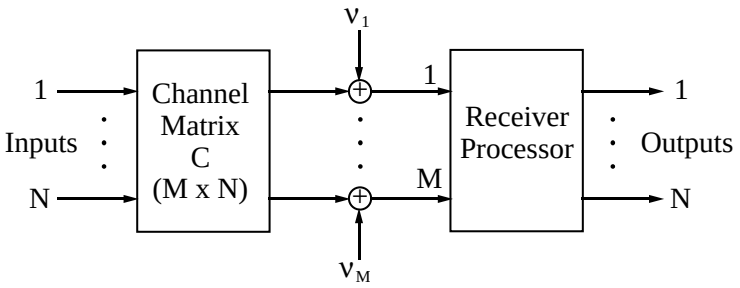


Figure 2.3: Spatial Division Multiple Access Block Diagram

propagation vectors $\mathbf{c}_j(\omega)$, for each user j, can be placed into a matrix

$$C(\omega) = \left(\mathbf{c}_1(\omega), \quad \mathbf{c}_2(\omega) \quad , \ldots , \quad \mathbf{c}_N(\omega) \right) \tag{2.5}$$

where $C(\omega)$ completely describes all the desired signals that are received by the array, which mutually interfere with each other. The array received signal vector then becomes

$$\mathbf{u}_n = C(\omega)\mathbf{d}_n + \boldsymbol{\upsilon}_n \tag{2.6}$$

where $\mathbf{d}_n$ is the vector of transmitted signals from the users at time n.

A weight vector is associated with each user, and each one can be adapted to select a particular user and null the other users' signals. Since each weight vector can null $M-1$ user signals besides the particular desired signal, the theoretical maximum capacity is M users, and $N \leq M$. For flat Rayleigh fading, the elements of $C(\omega)$ are not frequency dependent and the matrix will be denoted as C.

Figure 2.3 shows that in modelling the system, the N data signals are transformed by the channel matrix C to form the M received signals at the array corrupted by additive noise at each element. The receiver processor uses the N weight vectors to recover estimates of the transmitted signals.

Note that if the channel coherence time is low enough, as in the indoor environment, the same array can be used with the complex conjugates of the receive weights to transmit with the antenna pattern that was adapted for receiving. Just as the received signals are spatially isolated, this allows the array to transmit independent

signals to the portables. The spatial processing is then concentrated at the base station, allowing the portables to be simpler. For this time division retransmission to be able to work, the transmit and receive circuitry must be reciprocal, the transmit and receive frequencies must be within a coherence bandwidth of each other, and synchronous time division must be used with all signals, even those in other systems.

When a user is added to the system, each existing user's weight vector must be immediately retrained in order to spatially accommodate the new user. On the other hand, when a user is removed, no retraining is necessary at that time. These factors must be considered when designing a SDMA system, in particular the algorithm for the addition and deletion of users accessing the array.

2.3 Some Methods of Adaptation

Numerous methods exist to arrive at the optimal weight vector associated with a given user. Three of them that will be described here are direct measurement and subsequent inversion of the channel propagation matrix C, recursive techniques, and so-called direct matrix inversion. All discussion will assume a receive array.

2.3.1 Channel Matrix Inversion

In order to reverse the effects of the vector channel, a matrix is required that is effectively the inverse[1] of the channel matrix C. The elements of C must be measured using orthogonal channels, i.e., the users must transmit a known sequence using time division, frequency division or code division in order to measure each user's channel propagation vector. Due to noise, multiple measurements have to be made and then the actual channel is estimated by least squares techniques, such as pseudoinverse methods [2]. The transmitted data vector can then be estimated as

$$\hat{\mathbf{d}}_n = C^{+}\mathbf{u}_n \tag{2.7}$$

[1] Without noise the exact inverse, or pseudoinverse if C is non-square, is used.

where + indicates pseudoinverse. Because of the additive noise present at the array, this method does not result in the optimal inverse of C in the least squares sense.

2.3.2 Recursive Techniques

First the LMS algorithm is considered. The LMS algorithm for the array consists of the weight vector update equation

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \mu \mathbf{u}_n e_n^* \tag{2.8}$$

where the error $\mathbf{e}_n$ is given by

$$e_n = d_n - \mathbf{w}_n^H \mathbf{u}_n \tag{2.9}$$

and μ is a constant adjustment factor. The LMS algorithm converges near the least squares solution, with a larger misadjustment for larger μ. However, a larger μ results in faster convergence. This algorithm has low computational complexity, (order M) but the convergence time depends on the eigenvalues of the array spatial autocorrelation matrix R, i.e., on the relative powers of the desired and interfering signals. This means a weaker signal is acquired and tracked at a slower rate than more powerful interference. In addition, when the desired signal enters a fade, it will be tracked slower at a time when accurate tracking becomes more important [4].

For faster convergence rates, the RLS algorithm can be used. It has a higher computational complexity (order M^2, although faster versions exist with order M) but doesn't depend on the eigenvalues of R. It won't be detailed here. It requires a matrix inversion, leading to instability issues; however, the recursive structure of the algorithm allows this to be done implicitly, thus avoiding the order M^3 explicit inversion.

Some discussion concerning the startup of the RLS algorithm is needed. When applied to the spatial array problem here, startup would occur for instance at the acquisition phase, or perhaps during tracking if the channel has changed significantly between training sequences available in the time slots and the autocorrelation matrix

from the previous training sequence doesn't reflect the channel statistics. When applied to spatial arrays, the RLS algorithm has the same startup problem as in a transverse (time domain) situation, namely M snapshots are required in order to make the estimate of R of full rank and thus invertible and hence start the algorithm. This problem can be overcome using the same techniques available in a transverse filter, which are using M snapshots of the array to estimate R and then performing an explicit inverse, or setting R to δI where δ is a small positive constant and using this R to start the recursion. Winters [4] indicates that if the autocorrelation matrix in the RLS algorithm becomes singular then pseudoinverse techniques can be employed, and reference is made to Dembo and Salz [10] where a recursive pseudoinverse algorithm is presented to deal with singular matrices for the RLS algorithm startup (although this pseudoinverse recursion is intended for a transverse filter startup).

2.3.3 Direct Matrix Inversion

The direct matrix inversion (DMI) method solves the least squares problem, i.e., the Wiener solution, using estimates of the autocorrelation matrix

$$\hat{R} = \frac{1}{L} \sum_{k=1}^{L} \mathbf{u}(k)\mathbf{u}^H(k) \tag{2.10}$$

and cross-correlation vector for user j

$$\hat{\mathbf{p}}_j = \frac{1}{L} \sum_{k=1}^{L} \mathbf{u}(k)d_j^*(k) \tag{2.11}$$

where L is the number of samples used. The weight vector for user j is then given by

$$\mathbf{w}_j = \hat{R}^{-1}\hat{\mathbf{p}}_j \ . \tag{2.12}$$

Note that R is assumed to be nonsingular; if not, pseudoinverse methods can be used [4]. About $L = 2M$ samples are usually adequate to achieve good results [11, p. 123].

The DMI algorithm is the most computationally intensive algorithm; since it needs a matrix inversion its complexity is order M^3. However, it has the fastest convergence [4], and its convergence rate is independent of the eigenvalues of R. The

DMI algorithm is inherently a block algorithm, and a number of methods exist to modify it for tracking a time-varying channel. One way is to use a sliding window for the data with a fixed L in (2.10) and (2.11). Another method is to use an exponential forgetting factor for $\hat{R}$ and $\hat{\mathbf{p}}_j$, i.e.,

$$\hat{R}(n+1) = \beta \hat{R}(n) + \mathbf{u}(n)\mathbf{u}^H(n) \tag{2.13}$$

$$\hat{\mathbf{p}}_j(n+1) = \beta \hat{\mathbf{p}}_j(n) + \mathbf{u}(n)d_j^*(n) \tag{2.14}$$

where β is the forgetting factor. This method appears to be a border-line RLS algorithm, with the major difference being the use of an explicit inverse and not the matrix inversion lemma as in the latter.

2.3.4 Summary of Adaptation Methods

Since the LMS algorithm converges near the optimal least squares solution and is very computationally simple, it is a logical choice for adapting the array coefficients to the channel conditions. If a faster convergence and tracking rate is necessary, the RLS algorithm can be employed with a computational penalty. However, the indoor channel may allow the LMS algorithm to be successfully employed since the coherence time is large enough. In addition, once the propagation environment has been acquired, the detected symbols may be used for tracking, perhaps making further training sequences unnecessary. For $M = 2$ the DMI algorithm has about the same computational complexity as the LMS algorithm [4], but the M^3 complexity growth of DMI makes it exceedingly complex for larger numbers of array elements.

2.4 The Data-To-Fading-Rate Ratio

The ratio of data rate to fading rate is an important quantity. Since the algorithm needs the received data symbols to update the array coefficients, a higher data rate implies a better ability to keep up with a changing channel. Thus, for a given channel

coherence time, a relatively high data rate lets the algorithm adequately adapt and keep the array output performance within tolerance, assuming the signal processing hardware can operate at the desired rate. For example, in IS-54, at a mobile speed of 60 mph the data-to-fading ratio is 300, and in GSM it is about 2000. Experiments have shown that with eight elements and three mutual interferers, the LMS algorithm had an adequate tracking performance with data-to-fading-rate ratios down to 25 [4]. In an indoor system such as that emphasized in this report, the data-to-fading-rate is much higher (possibly as high as 10^5), indicating that the LMS algorithm would be suitable, although the antenna redundancy may be lower than in the example stated above.

2.5 Spatial Diversity Versus Multiplexing

An interesting result concerning the relationship between diversity and multiplexing in a flat Rayleigh fading environment with optimum combining at the receiver is given in Winters et al. [3]; namely an upper bound for the probability of bit error for the jth user is given as

$$P_{e,j} \leq \left(1 + \frac{\rho_j}{\sigma_{d,j}^2}\right)^{-(M-N+1)} \tag{2.15}$$

where as before M is the number of antenna elements and N is the number of mutual interferers, $\sigma_{d,j}^2$ is the power of the complex data symbols for the user, and ρ_j is the SNR for the user,

$$\rho_j = \frac{\sigma_{d,j}^2 \sigma_j^2}{N_0}$$

where σ_j^2 is the variance of the complex Gaussian channel gains in the propagation vector for the user, and N_0 is the noise power at each of the array elements.

This indicates that the probability of error with optimum combining is the same as maximal ratio combining with $M-N+1$ antennas and no interferers. Thus the error rate for a particular user depends only on its own SNR and is unaffected by the other users. If there are as many users N as there are array elements M, the

performance behaves as if each user had only one antenna and were isolated from all other users. Each additional array element adds another diversity antenna for *each* user. Thus diversity for each user can be traded for increased system capacity.

Chapter 3

Simulations

Monte Carlo simulations were undertaken to investigate a multiuser adaptive receive array, and were intended primarily to investigate the factors influencing the convergence speed of the LMS algorithm in adapting the coefficients of the array. The simulation work is designed to complement future experimental results from the hardware prototype whose design is described in Chapter 4. The experimental methods will be described first, including a detailed description of the simulations, a description of the parameters whose effect on the training time will be investigated as well as some other parameters of interest, and a discussion of the expected results.

3.1 Experimental Methods

3.1.1 Description of Simulations

An adaptive array used for multiple simultaneous access was simulated. The array has twelve elements, and the number of users accessing it is variable — four separate numbers of users were used, namely two, four, eight, and twelve. The channels from the portables to the array are assumed to be independent, flat Rayleigh fading channels. This is valid since the array is intended to be used in the indoor environ-

ment and is assumed to have independent fading among the elements by virtue of its construction (this is further explained in Chapter 4). Thus the elements of the channel matrix C are independent, zero-mean, complex Gaussian random variables with variance σ_j^2 for the jth user. For the simulations, the elements of C have unity variance for all users. Complex data symbols are used, there are no ISI effects, and the bare received symbols are processed directly, i.e., there is no temporal filtering of the received data symbols.

The transmitters send QPSK symbols of unity variance and zero mean which are generated as

$$d_n = (a_n + jb_n)/\sqrt{2}$$

where the inphase component $a_n \in \{-1, +1\}$ and quadrature component $b_n \in \{-1, +1\}$ are taken from the computer's pseudorandom number generator; the generator is seeded to a different value each time the simulation is run. The correlation between sequences, which is important in this application because all users are simultaneously trained, was found to be insignificant (see Appendix A). The sequences are intended to be mutually independent. The symbols received at the array are found using (2.6), reproduced here for convenience with the frequency dependence of C removed:

$$\mathbf{u}_n = C\mathbf{d}_n + \boldsymbol{v}_n \ . \tag{3.1}$$

Each element of the noise vector $\boldsymbol{v}_n$ is an independent sample from a zero-mean, complex Gaussian distribution with variance determined as follows. First the matrix C is chosen, and then the noise variance is computed using a system-wide received SNR of 30 dB given by

$$SNR = \frac{\sum_{i=1}^{M}\sum_{j=1}^{N} P_{r\,i,j}}{M\sigma_v^2} \tag{3.2}$$

where $P_{r\,i,j}$ is the received power at element i from user j and σ_v^2 is the noise variance. Thus the SNR used to find the noise power is the total received power across the entire array divided by the total noise power in the array.

All users are simultaneously trained using the LMS algorithm, which can be put into a matrix version for simplicity by forming a matrix of complex weight coefficients

$$W = \left(\begin{array}{cccc} \mathbf{w}_1, & \mathbf{w}_2 & , \ldots , & \mathbf{w}_N \end{array} \right) \tag{3.3}$$

where as before N is the number of users. The weight matrix update equation then becomes

$$W_{n+1} = W_n + \mu \mathbf{u}_n \mathbf{e}_n^* \tag{3.4}$$

where for the simulations $\mu = 0.001$ and where the error vector $\mathbf{e}_n$ is given by

$$\mathbf{e}_n = \mathbf{d}_n - W_n^H \mathbf{u}_n \ . \tag{3.5}$$

The initial value of the weight matrix is $W_0 = \emptyset_{MN}$, i.e., the $M \times N$ null matrix. After W has converged, the output estimated data symbol vector is given by

$$\hat{\mathbf{d}}_n = W^H \mathbf{u}_n \ . \tag{3.6}$$

In order to investigate the training time of each user for a given C, a threshold square error is used with the training curve, below which the weight vector is said to have converged. In order to relate this to a bit error rate, an exponentially tight upper bound from [3] is used, which with high SNR is

$$(MSE)_{0j} \leq \frac{-1}{\ln P_{e,j}(C)} \tag{3.7}$$

where $(MSE)_{0j}$ is the minimum mean square error (MMSE) for user j and $P_{e,j}(C)$ is the probability of error for user j based on C. In the simulation, a probability of error of 10^{-3} is used to compute the corresponding upper bound on the MMSE. This MMSE is used as the convergence criterion. A fixed training length of 1000 symbols is used, and if the square error fails to pass below this value within those 1000 symbols, the training curve for that user based on the particular C in use is recorded as not having converged. In order to remove the effects of noise, the mean of the square error for every ten consecutive iterations is examined to see if it has passed below the criterion. When a curve fails to converge, its training time is recorded as 1000. Note

that since an upper bound is used to compute the square error criterion, it can be said that the curve must pass *at least* below the criterion to result in a BER of 10^{-3} but exactly how much is not known. However since the bound is exponentially tight and the SNR is high, this is a minor concern.

The value of $\mu = 0.001$ was chosen such that when used with the BER threshold of 10^{-3} and training length of 1000 symbols, a reasonable reliability (to be defined below) would result.

To elaborate on the sliding window mentioned above which is used to find the point on the noisy training curve when the curve has passed a threshold, the length of the window is a trade-off between high resolution and a long enough window to remove the effects of noise. A window length of ten seemed to provide high enough accuracy and robustness to noise. In addition, the increment for the window was chosen to be ten to provide noise immunity and improve the speed of the search along the curve for the threshold crossing point.

For each of the four numbers of users, a total of 500 channel matrices C are generated and each is used to train the users' weight vectors. The output for each of the four runs saved for later analysis is a record of the 500 matrices C, a record of the training time for each user represented in each matrix C, and the noise variance used for each C. In addition, the reliability, the calculation of which is described below, is output for each run.

3.1.2 Parameters of Interest

Reliability

The reliability is defined as the percent of instances that the training curves do converge (according to the threshold criterion given above) for each number of users.

The reliability is calculated as

$$reliability = \left(1 - \frac{outages}{N \times 500}\right) \times 100\%$$

where $outages$ is the total number of outages over all 500 trials and N users.

Eigenvalue Spread

It is well known that the eigenvalues of the spatial autocorrelation matrix affect the training time of the LMS algorithm [4]. Specifically, a larger eigenvalue spread $\chi(R)$, defined to be the ratio of the maximum eigenvalue to the minimum eigenvalue for a given propagation and noise environment, results in larger training times, since unequal eigenvalues cause a distortion of the error surface whose minimum the LMS algorithm searches for [6].

An autocorrelation matrix with a large $\chi(R)$ is termed ill-conditioned, and in practice this can correspond to a large difference in received powers of the users' signals at the array, i.e., a large received power spread. However, a large eigenvalue spread can also be caused by small angles between the propagation vectors, by correlation between users' signals, and when the number of users is less than the number of array elements [12], [13].

It can be shown that the autocorrelation matrix R is

$$R = \sigma_v^2 I_M + \sigma_d^2 C C^H \tag{3.8}$$

where I_M is the identity matrix of size M and σ_d^2 is the variance of the transmitted complex data symbols for all users.

Received SINR

The received SINR for user k is defined to be

$$SINR_k = \frac{\sum_{i=1}^{M} P_{r\,i,k}}{M\sigma_v^2 + \sum_{i=1}^{M}\sum_{\substack{j=1 \\ j\neq k}}^{N} P_{r\,i,j}} \tag{3.9}$$

where $P_{r\,i,j}$ is the power received at element i from user j. In other words, the SINR for user k is the total power received at the array from user k divided by the sum of the total noise power at the array and the power received at the array from all other users.

Power Spread

Power spread is defined to be the variance of the total received powers from each user at the array for a given channel matrix C, and is found as

$$\chi(P_r) = \sigma_{P_r}^2 = \frac{\sum_{j=1}^{N} \left(P_{r\,j} - \overline{P_r}\right)^2}{N-1} \tag{3.10}$$

where $P_{r\,j}$ is the total received power for user j, $\overline{P_r}$ is the mean received power over all users, and N is the number of users. If there is a large difference in received user powers at the array, the variance of the powers will be large compared with when the received powers are all similar. Note that eigenvalue spread, received SINR, and power spread are similar metrics in that they attempt to quantify the relative power of users' received signals. However, each represents different quantities, and the effect that each has on the LMS algorithm training time will be investigated. The eigenvalue spread does depend on the users' received powers, but it also depends on other variables. A user's SINR directly measures that user's power in relation to the other users' received powers, and the power spread measures the amount of deviation of all users' received powers from the mean.

Propagation Vector Orthogonality

An important concept in a spatial multiplexing system is that the propagation vectors for the users must be sufficiently different from each other and from any interference present in order for the adaptive algorithm to distinguish and thus spatially isolate the signals from each other. A good measure of the similarity between two vectors in M-space (where M as before is the number of array elements) is the angle between them. For users i and j this is given as

$$\phi_{i\,j} = \cos^{-1}\left\{\frac{Re[\mathbf{c}_i^H \mathbf{c}_j]}{\sqrt{\mathbf{c}_i^H \mathbf{c}_i \; \mathbf{c}_j^H \mathbf{c}_j}}\right\} \; . \tag{3.11}$$

Ideally each propagation vector will be orthogonal (90 deg) to all other vectors, since this makes it easiest for the adaptive algorithm to spatially isolate the users' signals, and the training time should be fastest in this case. The worst case would be parallel vectors, in which case there is nothing the processor can do to separate the two signals, at least in a spatial sense.

Correlation Coefficient

When investigating the relationship between two variables, the correlation coefficient is a useful metric of how well one variable is correlated with the other. It will be used as a quantitative measure of the effect that some of the above parameters have on the training time. The correlation coefficient is related to linear regression (least squares linear curve fitting) and is a measure of how well a straight line can be fit to a set of points.

The correlation coefficient between variables X and Y is defined as

$$r = \frac{\sigma_{XY}^2}{\sigma_X \sigma_Y} \tag{3.12}$$

where c_{XY} is the covariance of X and Y and s_X and s_Y are the standard deviations of X and Y respectively [14]. The correlation coefficient of a sample population may

be written as

$$r = \frac{\sum(X - \overline{X})(Y - \overline{Y})/(n-1)}{\sqrt{\frac{\sum(X-\overline{X})^2}{n-1}}\sqrt{\frac{\sum(Y-\overline{Y})^2}{n-1}}} \tag{3.13}$$

where $\overline{X}$ and $\overline{Y}$ are the sample means of X and Y respectively and n is the number of elements in the sample.

If the sample points (X, Y) are perfectly positively correlated (i.e. they lie on a straight line with positive non-zero slope) then $r = 1.0$, and conversely when they are perfectly negatively correlated $r = -1.0$. As the points deviate from a straight line, r will approach zero, and when there is no correlation present at all, for instance when the points form a circle, $r = 0$. Note that the correlation coefficient is different from the regression coefficient, where the latter is the slope of the line found from a linear regression. Thus the correlation coefficient is not equal to the slope of the regressed line, but it does indicate the sign of the slope of the regressed line. In addition, it is not necessary to perform a regression to calculate r, since all that is needed is the covariance and variance information of the variables.

The relationships to be investigated in this chapter are not assumed to be linear, and visual inspection will indicate that they are not linear. But with this in mind, the correlation coefficient can be used in an effort to quantify the degree of relationship present.

3.1.3 Expected Results

The simulations are designed to investigate how the number of users, the eigenvalue spread of the autocorrelation matrix, the received SINR, the received power spread, and the propagation vector orthogonality affect the training time of the users in the system. The results are divided into two groups: inter-user results consider the effect that changing the number of users has on training time, and the smaller scale intra-user results involve what affects the training time for a fixed number of users.

Inter-User Effects

It is expected that increasing the number of users will increase the training time needed for all users, since each user will experience more interference power and this will slow the convergence of any adaptive algorithm since the interference power will appear as additive noise to a user being trained. As a direct result of this, the received SINR for a user is expected to drop as more users are added. Since the training time will increase with more users, which implies a more ill-conditioned autocorrelation matrix, the eigenvalue spread $\chi(R)$ is expected to increase. The power spread isn't expected to change with increasing users since each user's channel gain statistics remain constant regardless of the number of users. The propagation vector orthogonality isn't expected to change either since the vectors are fixed at size twelve and the channel gain statistics are independent of the number of users.

Intra-User Effects

For a given number of users, the effect that the four parameters (eigenvalue spread, received SINR, power spread, and propagation vector orthogonality) have on training time will be investigated. It is expected that a larger eigenvalue spread will result in a larger training time, and a smaller SINR will result in a larger training time. A larger power spread should result in a larger training time, and propagation vector angles closer to 0 deg or 180 deg (closer to parallel) are expected to result in a larger training time. The SINR is expected to have the largest effect since the training time of the LMS algorithm depends directly on the relative received power of a given user. The effect of the eigenvalue spread may not be as pronounced since it doesn't depend on the particular user being considered but is based on all users' received powers.

The eigenvalue spread, power spread and the received SINR results are intended to investigate the *near-far* problem, which is the detrimental effect that a variation of received user powers has on a multiple access wireless system. In the present context, it is the reduced convergence rate for the LMS algorithm that results when a user's

signal is weak compared with the other signals present. Note that since one power spread metric is obtained for the entire C matrix, it is used to examine whether the power spread has a general effect on the training time for all users for a given C. In addition, since the eigenvalue spread depends on more than just the received powers, it may not be the best metric to use when examining the near-far problem, but examining the eigenvalues will be of interest nonetheless.

3.2 Simulation Results

The simulation results primarily show how a number of factors influence the LMS algorithm training time. In the following sections the results are described and discussed.

3.2.1 Effect of Number of Users

In order to examine the effect of the number of users on the training time, the average training time over all users and all 500 channel matrices was computed for each number of users and plotted in Figure 3.1. As expected, increasing the number of users increases the training time due to the fact that there is more interfering power (a lower SINR for each user) which slows the LMS algorithm down. The graph seems to follow somewhat of an exponential curve, with large increases in training time as the number of users approaches the theoretical maximum of twelve for this array. The number of iterations needed for convergence for the LMS algorithm should be about five to ten times the number of coefficients to be trained [15]. The absolute training time for the lower number of users appears to agree with this, but the training time for the larger number of users exceeds this approximation considerably, perhaps due to the higher interference levels there.

Table 3.1 shows the reliabilities for various numbers of users. Only the twelve users situation results in a reliability less than 100%, namely 94.4%. These results

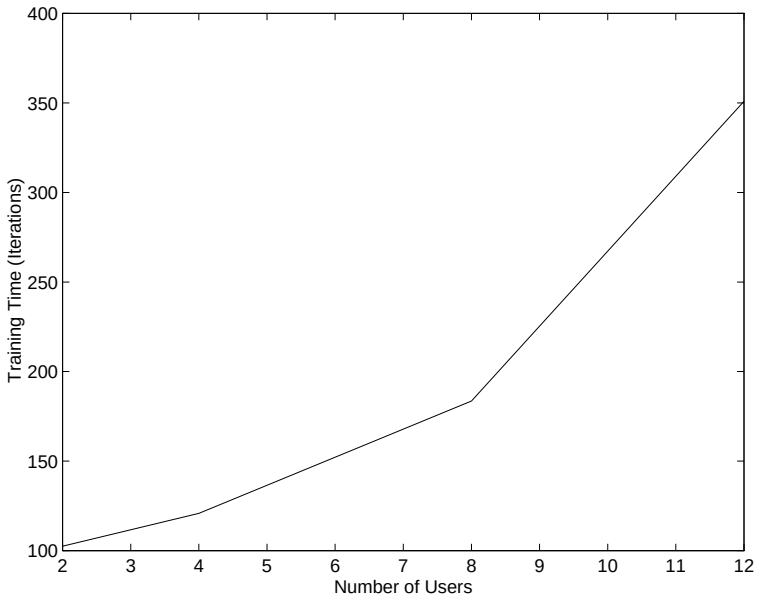
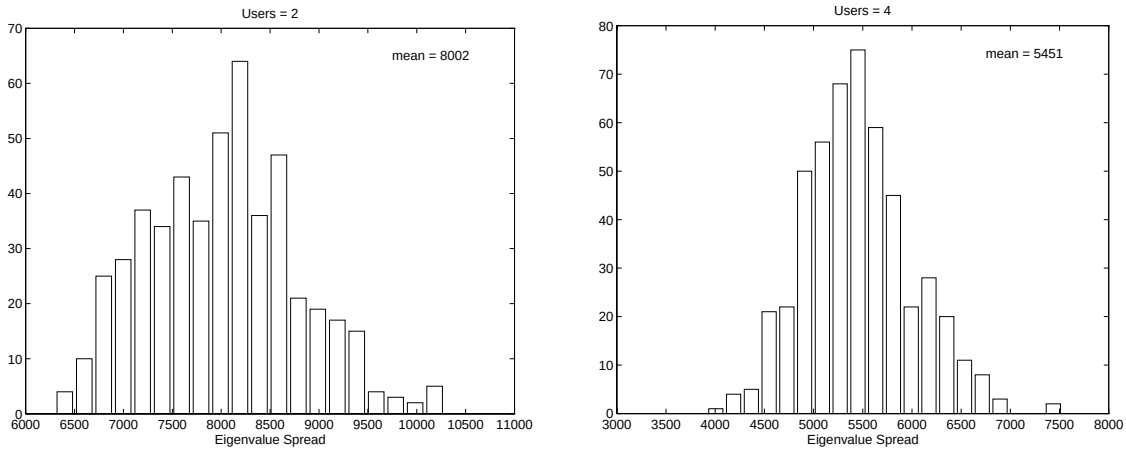


Figure 3.1: Effect of Number of Users on Training Time

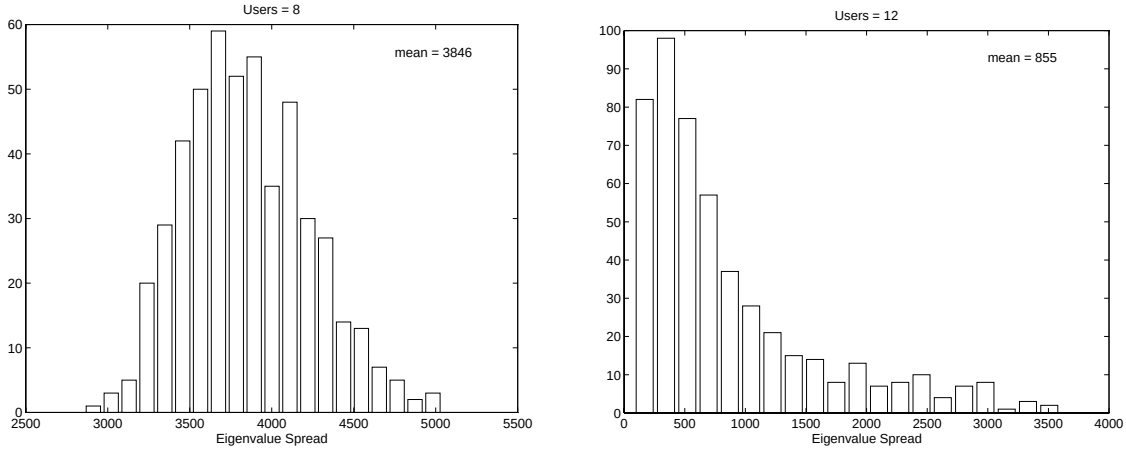
should only be considered relative to each other since the simulation parameters don't necessarily reflect a practical system. However, the parameters were chosen to keep the reliability results greater than about 95%, which is a typical value of a practical system. As expected, the reliability degrades with increasing numbers of users.

Number of Users	**Reliability (%)**
2	100.0
4	100.0
8	100.0
12	94.4

Table 3.1: Reliability for Various Numbers of Users



(a) (b)



(c) (d)

Figure 3.2: Histograms of Autocorrelation Matrix Eigenvalue Spread

3.2.2 Effect of Eigenvalue Spread

Figure 3.2 shows histograms of the eigenvalue spread of R for two, four, eight, and twelve users along with the mean spread for each number of users. Interestingly, the mean value of the spread becomes less as the number of users increases.

Figure 3.3 is a plot of training time versus $\chi(R)$ for the various numbers of users. The ordinate value is the mean of the training times for all the users found for a given C. The correlation coefficient r is shown on the graphs. For each number of users, a moderate correlation exists indicating that the average training time increases

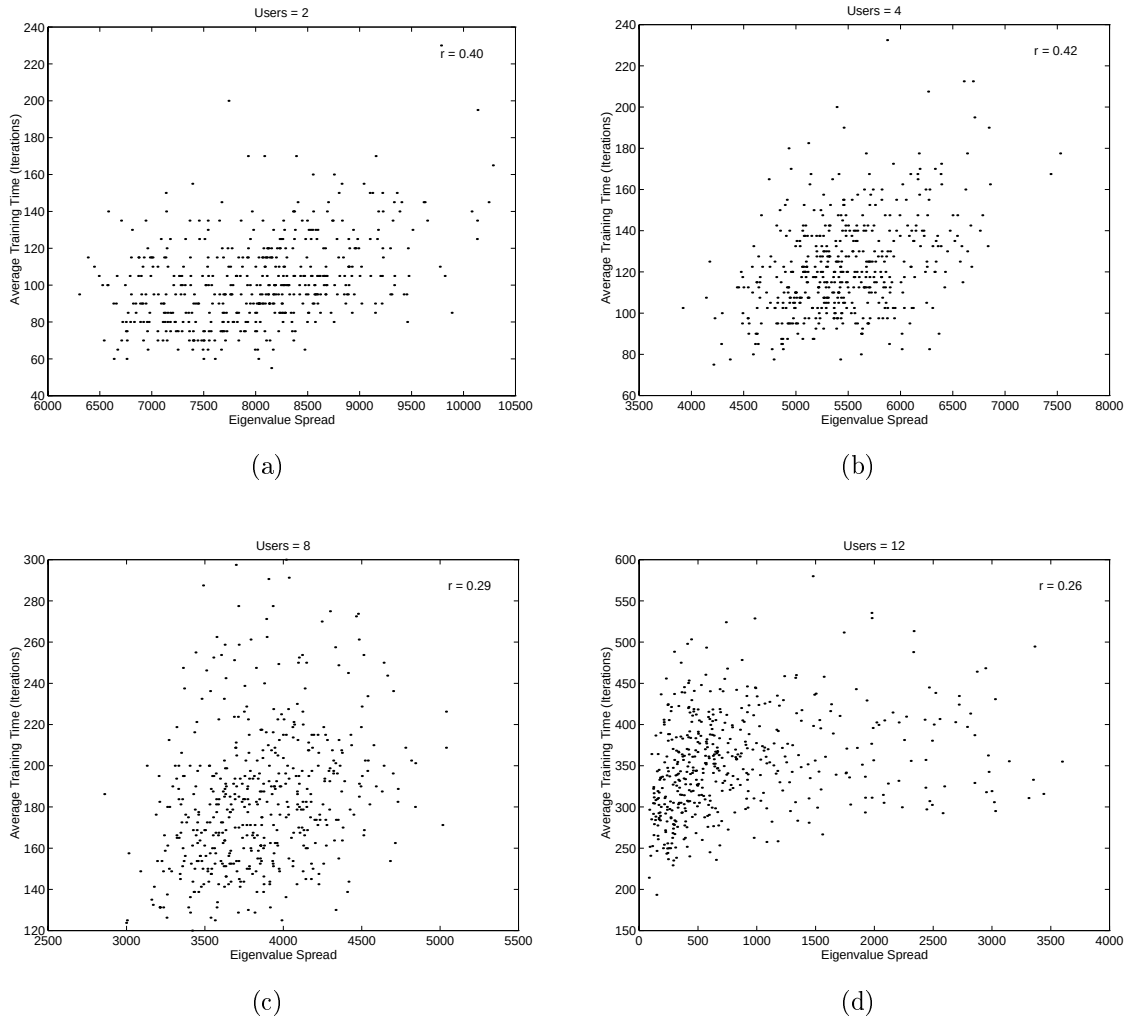


Figure 3.3: Effect of Eigenvalue Spread on Average Training Time

with $\chi(R)$ as expected. The curves are too noisy to identify any particular type of curve, although in part (d) of the figure, the curve is perhaps flattening out as $\chi(R)$ increases, indicating that for twelve users as $\chi(R)$ increases to larger values (past about 2000) the training time may not significantly increase.

Figure 3.4 is also a plot of training time versus $\chi(R)$ except that the ordinate now consists of the individual training time for each user as opposed to the average training time found for each matrix C. The correlation coefficient r is shown on each graph, and the values of r are much smaller than for Figure 3.3.

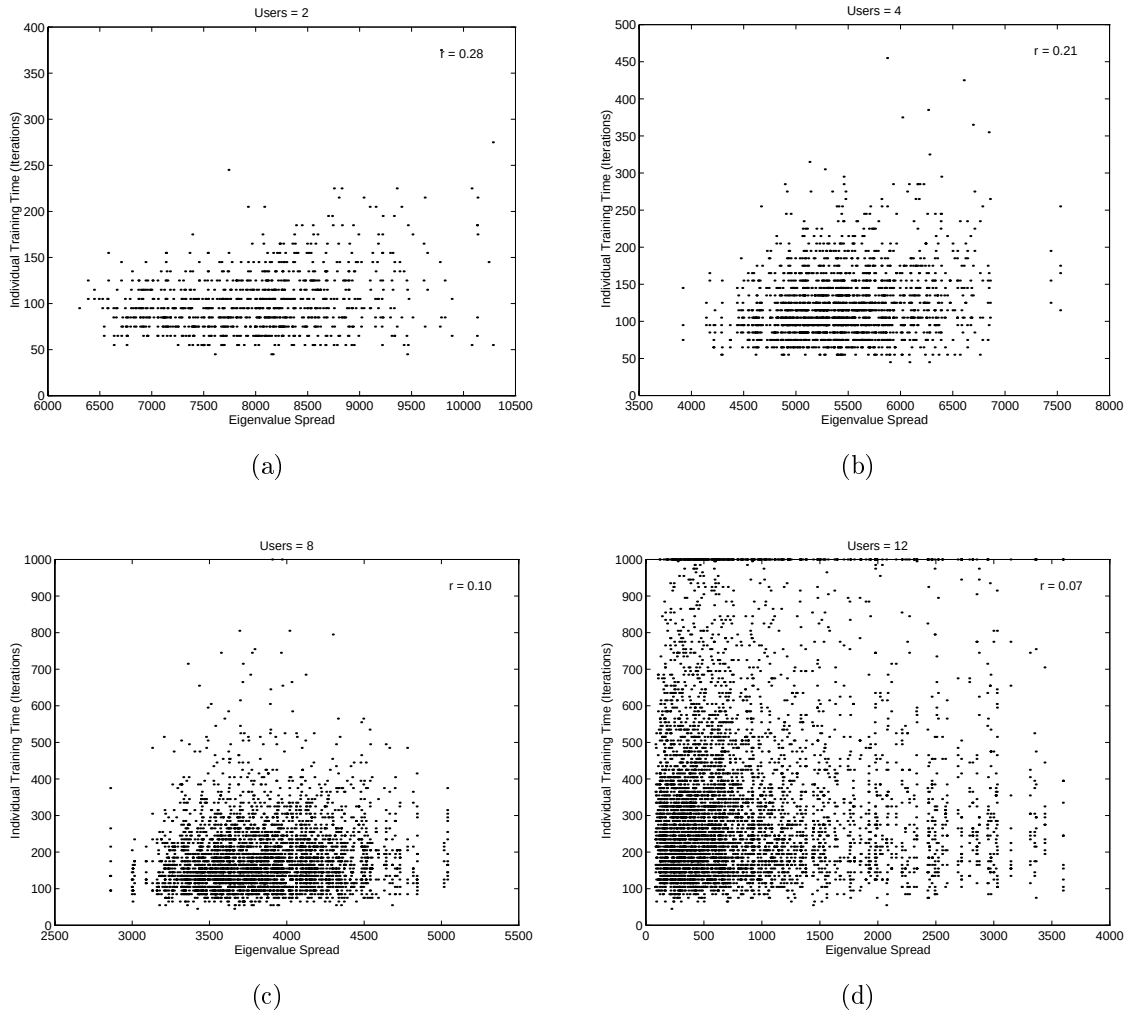


Figure 3.4: Effect of Eigenvalue Spread on Individual Training Time

3.2.3 Effect of Received SINR

Figure 3.5 shows plots of training time versus SINR for each number of users. The abscissa in this figure is the SINR for each user, and the ordinate is the training time for each user, hence the labels "individual" on the graphs. A clear relationship is seen for all numbers of users in which decreasing SINR results in increasing training time. The correlation coefficient r is shown on the graphs; for all four plots they show that there is significant correlation, and more correlation than was observed for the eigenvalue spread results. In Figure 3.5(d) many curves do not converge before the 1000 iterations limit below an SINR of approximately –10 dB.

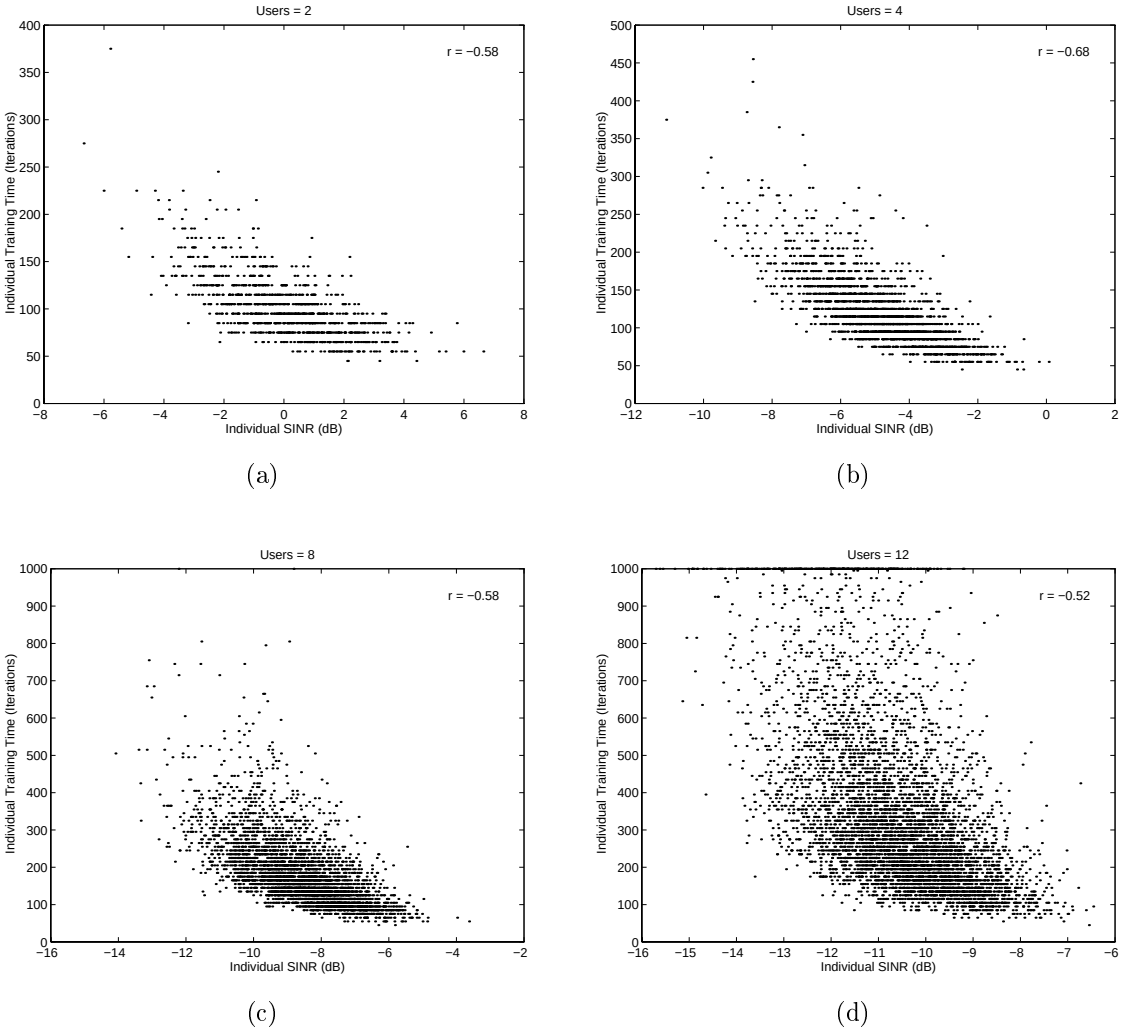


Figure 3.5: Effect of Individual SINR on Individual Training Time

Figure 3.6 also shows plots of training time versus SINR; however here the abscissa is the average SINR over the users for any given C and the ordinate is the average training time over all the users for a given C. The correlation coefficients for these graphs are considerably smaller than in the previous figure.

It should also be noted that the "hard" lower limit of the SINR, most pronounced in Figure 3.6(a), can be explained by examining, without any loss of generality, the average SINR function for two users, which neglecting noise is

$$SINR_{ave}(P_1, P_2) = \frac{1}{2}\left(\frac{P_1}{P_2} + \frac{P_2}{P_1}\right) \quad P_1, P_2 \geq 0 \tag{3.14}$$

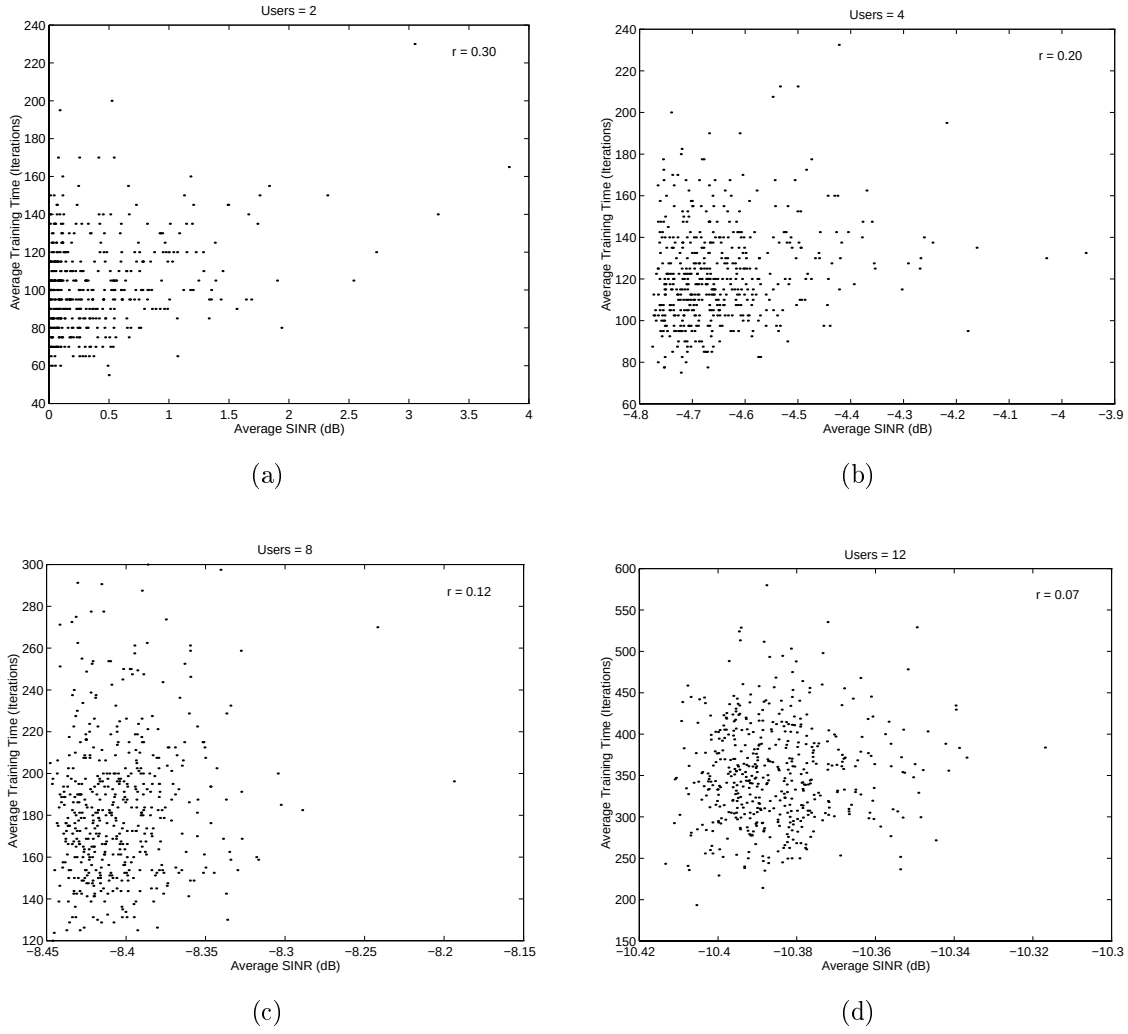


Figure 3.6: Effect of Average SINR on Average Training Time

where P_1 and P_2 are the received powers for user 1 and 2 respectively. This function is greater or equal to one, and thus the average SINR's that are plotted in Figure 3.6(a) are limited to greater or equal to 0 dB.

3.2.4 Effect of Received Power Spread

Figure 3.7 shows the individual training time versus the power spread for the four different numbers of users; the ordinate is the training time of each user. No discernible relationship can be seen. Figure 3.8 shows similar results, but the ordinate is the

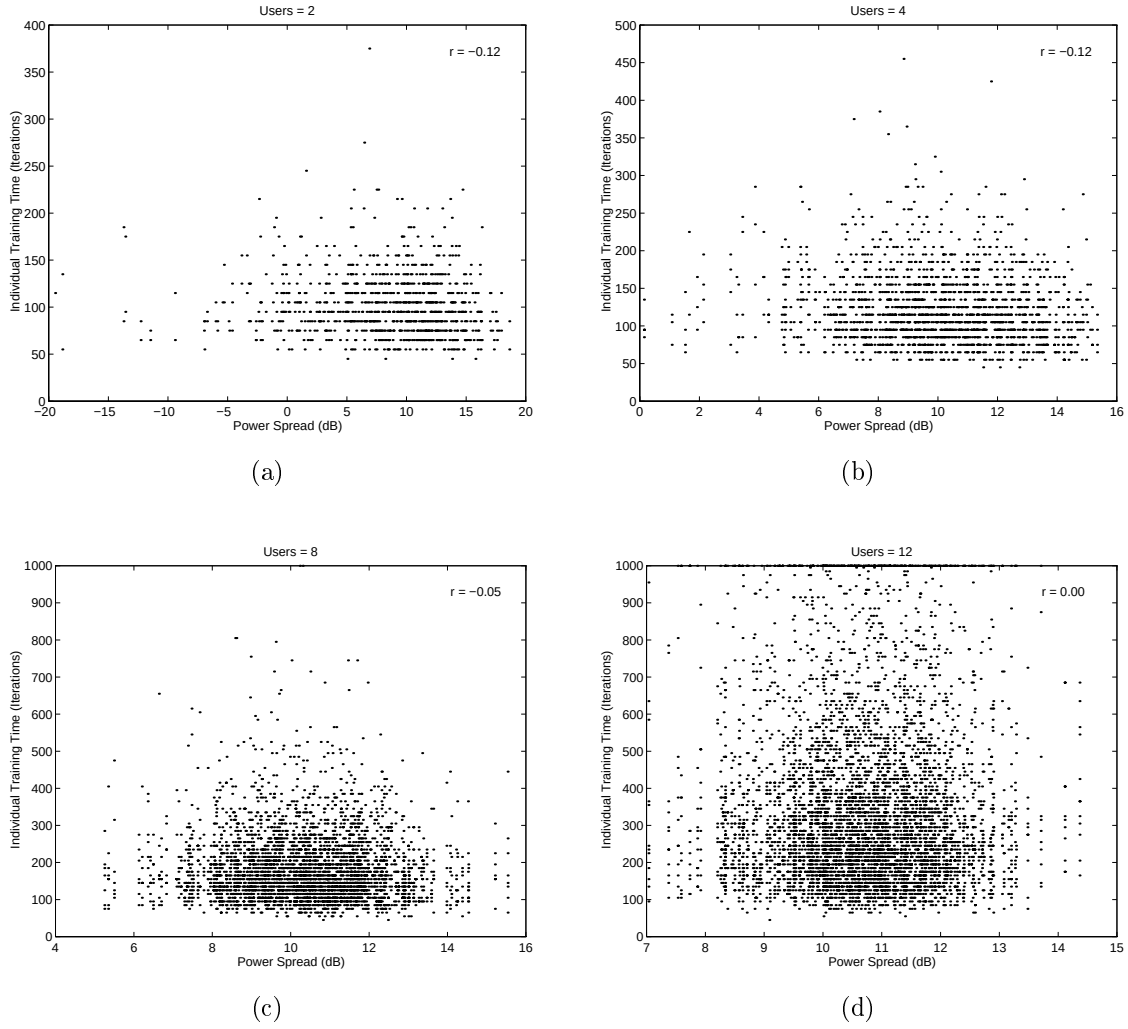


Figure 3.7: Effect of Received Power Spread on Individual Training Time

average training time among all users for a given C. Here as well no relationship is seen. Thus power spread as measured by variance appears to have little or no effect on the rate at which the LMS algorithm converges for the spatial array.

3.2.5 Effect of Propagation Vector Orthogonality

Figure 3.9 shows plots for each number of users consisting of training time versus the angles between propagation vectors. The abscissa is the average angle between propagation vector j and all the other propagation vectors. The ordinate is the

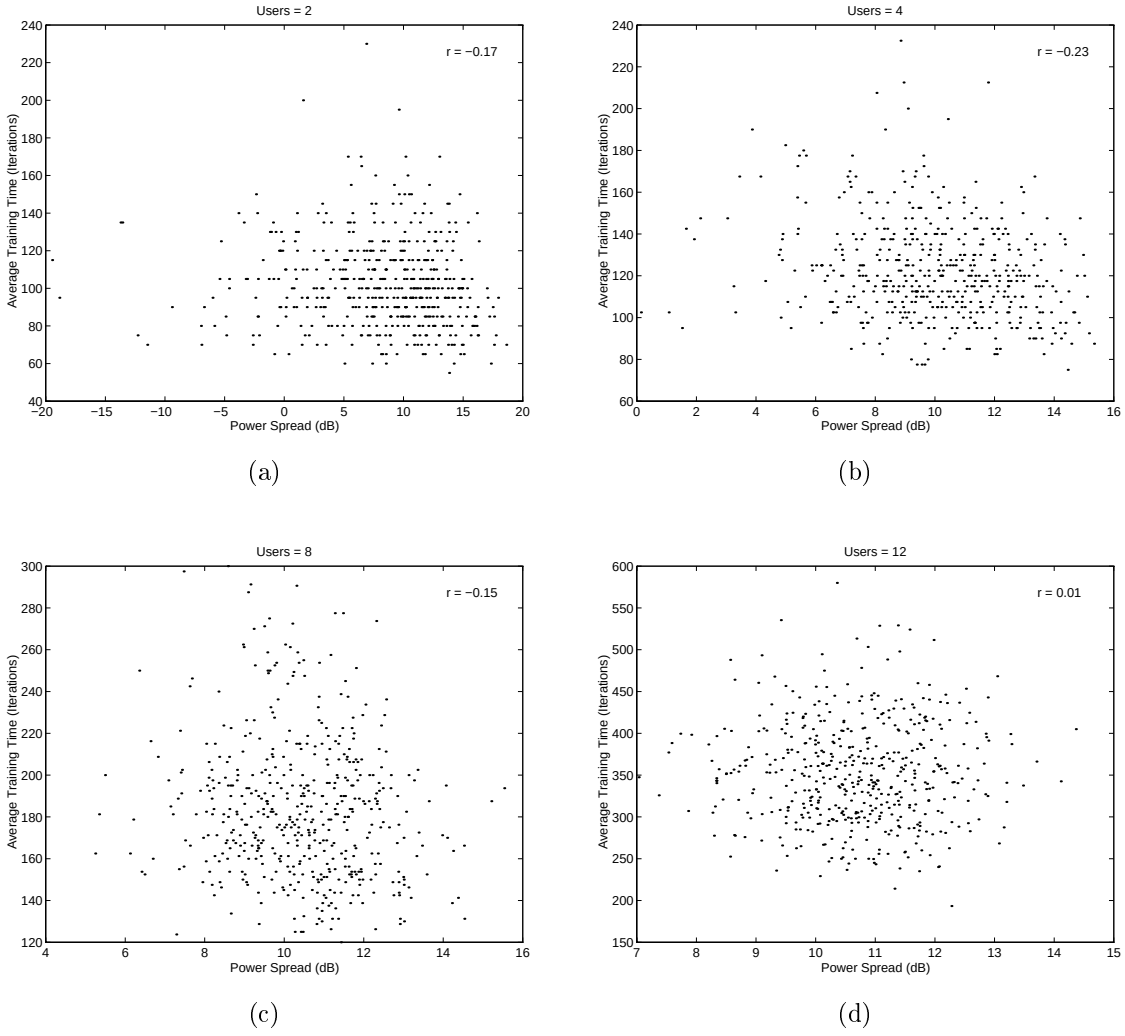


Figure 3.8: Effect of Received Power Spread on Average Training Time

training time for user j. No relationship can be seen through the noise, which would indicate that the vector angles don't have any significant influence on the training time. The angle mean and standard deviations are shown for each number of users on the graphs. The mean angle is 90.0 deg for all four plots, and the standard deviation drops from 12.5 deg for two users to 3.6 deg for twelve users. This indicates that the spread of angles around the mean decreases with more users.

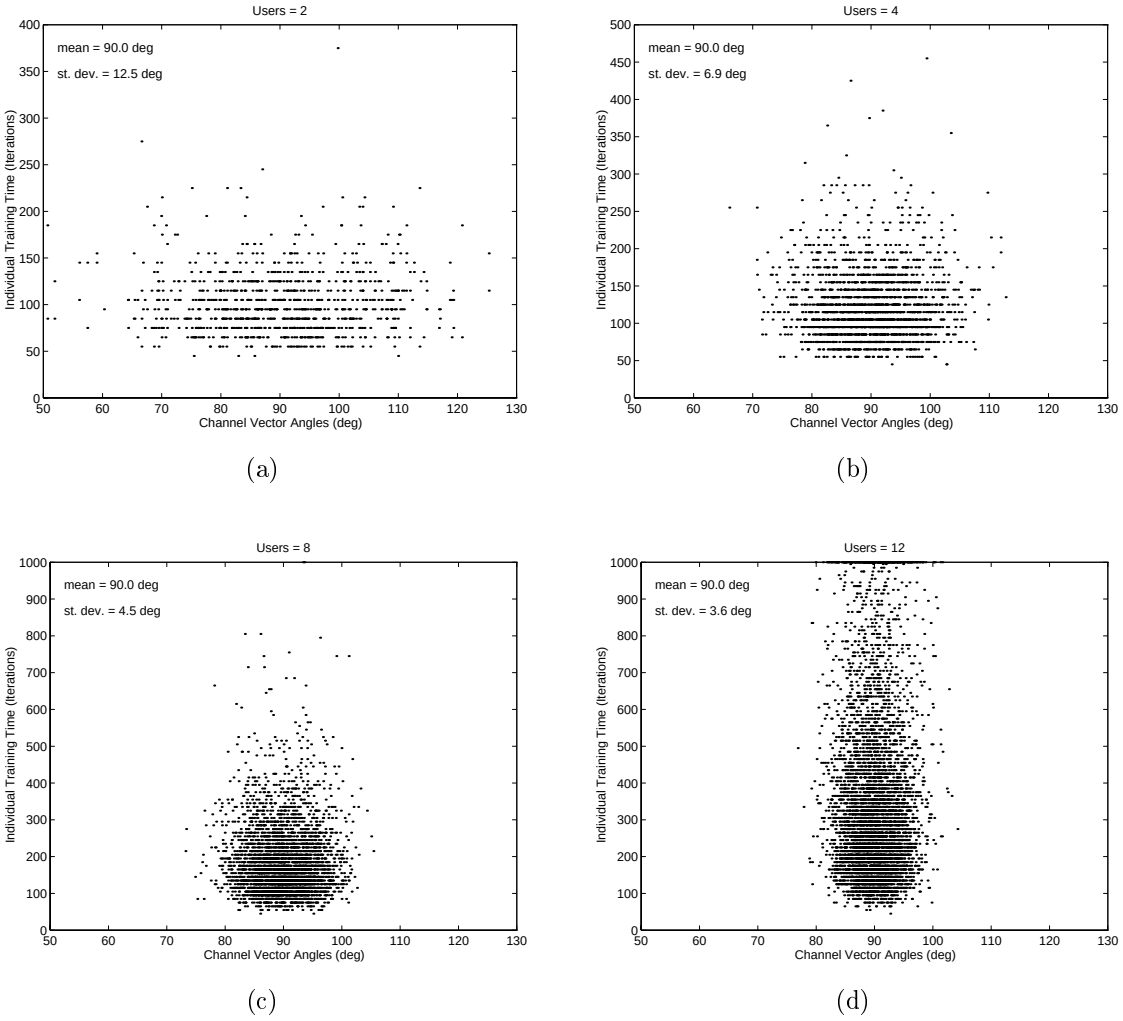


Figure 3.9: Effect of Propagation Vector Orthogonality on Training Time

3.3 Discussion of Results

Discussion of the dominant effects on training time will be divided into inter-user effects (when the number of users is varied) and intra-user effects (when the number of users is fixed).

3.3.1 Inter-User Effects

Since it is known that lower values of eigenvalue spread $\chi(R)$ result in higher rates of the LMS algorithm convergence, the lower values of $\chi(R)$ that result from increasing

the number of users don't directly affect the training time since the training time increases with increasing numbers of users. Even though more users means more interference power for a given desired signal, $\chi(R)$ becomes lower. The fact that the mean $\chi(R)$ becomes lower with increasing numbers of users can be explained by examining in more detail what the eigenvalues of R represent.

The number of eigenvalues is equal to the number of elements in the array, and the eigenvalues and the corresponding eigenvectors can be divided into two groups: signal eigenvalues with associated eigenvectors, and noise eigenvalues with associated eigenvectors. (A detailed discussion of the eigenstructure of the spatial autocorrelation matrix won't be given here, but suffice it to say that the signal eigenvectors form an orthonormal basis for the signals received at the array [16].)

The signal eigenvalues can be arranged as

$$\lambda_1 \geq \ldots \geq \lambda_N > \sigma_v^2$$

and the noise eigenvalues are represented as

$$\lambda_{N+1} = \ldots = \lambda_M = \sigma_v^2$$

[1]. The sum of the signal eigenvalues is equal to the total power received at the array from the users' signals (neglecting the noise components), and each signal eigenvalue is equal to some value representing a portion of the total signal power (but not necessarily any one user's received power) plus the noise power σ_v^2 present at an array element, provided all elements have the same noise power.

The noise power σ_v^2 is found after each channel matrix C is generated by requiring the SNR to be 30 dB, as previously explained. As more users are added, the value of σ_v^2 must be decreased, on average, in order to account for the increasing total signal power being received at the array. Since for two, four, and eight users the minimum eigenvalue is equal to σ_v^2, this accounts for the decreasing mean $\chi(R)$ as the number of users is increased from two to four, and then from four to eight. When the number of users is increased from eight to twelve, the value of $\chi(R)$ decreases much more than

the decrease from two to four and four to eight users. This large decrease is due to the fact that with twelve users the minimum eigenvalue is considerably larger than σ_v^2, since it has an added signal component.

By examining the abscissa scales of the plots in Figure 3.5 it can be seen that the SINR drops with increasing number of users, whereas examining the abscissa scales of Figures 3.7 and 3.9 shows that the power spread and propagation vector angles, respectively, don't change significantly for different numbers of users. Thus the SINR must have the dominant effect on training time for different numbers of users.

3.3.2 Intra-User Effects

The received SINR results are used to examine the effects that the received power of a single user relative to the other users' received signals has on training time, and the results show a significant correlation. However, there was no relationship seen for power spread (based on variance of received powers). Thus for this system the near-far problem makes its presence known by slowing the convergence rate for a user when that user has a low relative power, but has no apparent general effect on training time when there is a large variance in received users' powers.

As shown in Figure 3.5, for a given number of users the SINR appears to have the greatest effect on the LMS algorithm training time. A lesser effect results from the eigenvalue spread $\chi(R)$ which from Figure 3.3 is correlated with training time, although less significantly than for the SINR results. The other two effects examined, namely power spread and propagation vector angles, don't have any effect on the training time for a given number of users.

Looking at the propagation vector orthogonality results, the deviation of the angles about the mean of 90 deg for all four numbers of users isn't large enough for any vectors to become anywhere near parallel. The averaging effect (each point represents the average of the angles between one user's propagation vector and each of the other users' propagation vectors) may hide some angles that represent near-parallel

vectors, but most angles are near 90 deg. The fact that the vectors have length twelve appears to be beneficial in keeping the vectors near-orthogonal since there are twelve dimensions in which the vectors can differ. In other words having twelve elements results in excellent orthogonality among propagation vectors, assuming the vector elements are complex Gaussian distributed. As the number of users increases, the spread of the angles around 90 deg decreases because there are more angles being averaged (there are $N-1$ angles in the average, where N is the number of users).

Examining the eigenvalue spread and SINR results, and comparing whether or not averaging was used in the results, it can be seen that for the eigenvalue spread, averaged training times in Figure 3.3 were much more correlated than the individual data in Figure 3.4; in contrast for the SINR results, in Figure 3.5 the individual data showed a strong correlation whereas in Figure 3.6 where averaged data is used the correlation is small.

This set of observations can be explained by noting that the eigenvalue spread is the ratio of the largest eigenvalue to the smallest eigenvalue, and since the eigenvalues represent a power with contributions from most if not all users' signal powers, the spread has little or no individual information contained in it. Thus a plot using average training time is better suited than one using individual training time, since the average accounts for all the users. Conversely, the SINR is defined for one user only, and so a plot of individual training time versus SINR is better suited than one of average training time.

Chapter 4

Hardware Design and System Proposal

A prototype hardware system is proposed that will test the feasibility of a receive array used for spatial multiplexing. Although simulations can provide valuable information about a system, eventually the system will have to be built in order to verify the simulation results and work out practical problems not experienced in the simulations. No amount of simulating can replace implementation, and the simulation results described in Chapter 3 will complement the experience and results gained from implementing the prototype. This chapter describes the components of the system in detail and discusses relevant topics.

4.1 Overall System Design

The hardware system proposed is intended as a proof of concept for a multiuser system using an antenna array to implement spatial multiplexing in the indoor environment. The prototype implementation will have a data rate that is kept very low (in the single kbps range) in order to simplify the hardware design and avoid any significant multipath delay spread. In other words the important concept to be investigated is

the spatial processing and multiplexing. A number of aspects of the prototype will be unrealistic: for a system to be useful it must have a larger data rate which may induce intersymbol interference caused by delay spread, especially if the data rate is greater than 1 Mbps; and the data-to-fading-rate ratio of the prototype will be very low and this may preclude any realistic experiments on the tracking performance of the array. However, having acknowledged these simplifications, the experience gained from just the static multiplexing will be valuable.

The prototype will operate at a carrier frequency of 1.7 GHz. This frequency is chosen in order to keep the array small and because future personal communication systems (PCS's) will operate in this range. Figure 4.1 shows a block diagram of the proposed system. It consists of a DSP in a host computer which will send data to a number of portable transmitters through a cable. The transmitters will simultaneously operate and send the data over the wireless channel. The signals are received by the array, and fed back into the DSP. Thus the system is somewhat academic since the DSP knows what it will be receiving, but this simplifies the training process

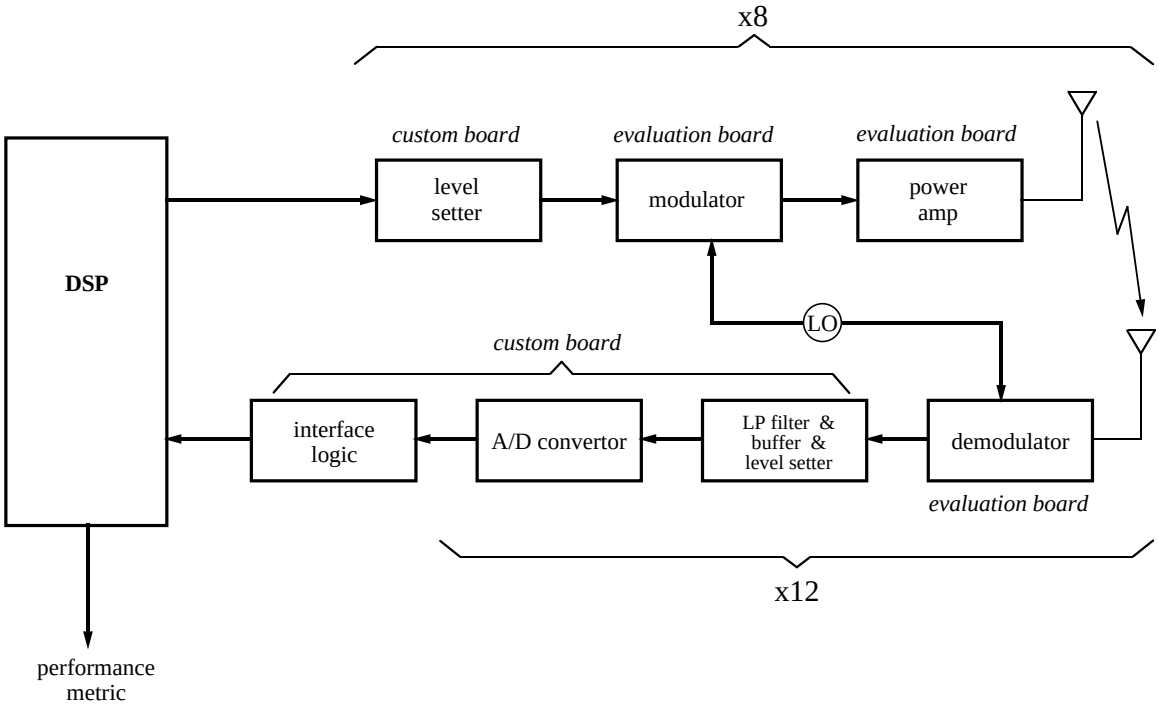


Figure 4.1: Prototype System Block Diagram

and eliminates the need for symbol synchronization problems to be solved. The DSP will perform the adaptation to the current environment and then as the figure shows produce a performance metric such as the bit error rate using the converged weight vectors. The modulation will be QPSK and demodulation will be coherent. This form of demodulation is readily implemented due to the availability of a synchronized carrier as described below. Each part of the system shown in the figure will now be discussed.

4.1.1 Digital Signal Processor and Host Interface

The DSP is a TMS320C30 which is a RISC processor with 32 bit floating point arithmetic. This should be adequate for the task. The DSP is housed in a Macintosh host computer, and the interface to the outside world is provided by the National Instruments NB-DIO-32F host interface board which is also in the host computer. The interface board has four data ports, each containing eight bits capable of input or output. Two of these ports will be dedicated to sending out data. Since the data symbols are intended for QPSK modulation they each contain two bits, and thus a total of eight transmitters can operate simultaneously by having the interface board send parallel data from the two interface ports.

4.1.2 Transmitter

The transmitter section takes the TTL level data from the host interface and transmits it using a modulator and power amplifier. The modulators are intended to be localized, and the RF signal will be sent from each modulator to its antenna via coaxial cable. The components of the transmit section are described below.

Level Setter

The output from the host interface is TTL and the corresponding voltage levels must be set to those required by the modulator. A simple op-amp circuit is suggested that allows the offset and amplitude of the data signal to be controlled. The modulator described in the next section requires a single-ended signal with an offset of 3 Volts and maximum peak-to-peak level of 2 Volts. If the maximum eight modulators are used, since each transmitter is I/Q and thus requires two parallel bits, a total of sixteen op-amp based level setters would be required.

Modulator

Two modulators were investigated for this system, the RF Micro Devices RF2422 2.5 GHz Direct Quadrature Modulator and the NEC UPC8104GR Upconverter And Quadrature Modulator. Both are available in IC form. Since the RF2422 is single stage whereas the UPC8104GR is two stage, the RF2422 was chosen for simplicity's sake. It requires a single 5 Volt supply and has an on-chip RF quadrature network (phase splitter) which can provide the correct quadrature signals over the frequency range of the device. It can be purchased on a preassembled evaluation board, which makes it easy to include in the system, i.e., it is intended that for this prototype, the evaluation boards will be used directly, thus avoiding any RF design work (the same applies for the power amplifiers and demodulators). It has a 50 Ω output.

The local oscillator (LO) input power is rated as low as –5 dBm (experiments showed it could probably go even lower without problems). The LO for both the modulators and demodulators will come from the same signal generator, using a large signal splitter. With twelve demodulators and the maximum eight modulators this requires a twenty way splitter, which will drop the LO by at least 13 dB. The modulators will be located at a central location along with the signal generator, so there will not be much loss getting the signal from the splitter to the modulators.

Since a typical signal generator can produce +17 dBm, this leaves adequate signal power at the modulator LO input.

By feeding all modulators and demodulators from the same LO generator, and if they are both single stage, this will keep the system phase synchronized over extended periods of time. If more than one generator were used, they could be phase synchronized using the synchronization circuitry provided with the generators, but this method has a tendency to drift and this setup would then be unacceptable.

Power Amplifier

The power amplifier chosen is the NEC UPC1678P Medium Power Broadband Amplifier. It has a rated saturated output power of 18 dBm at low frequency, and the worst case output power at 1.7 GHz is taken to be 8 dBm. It uses a single 5 Volt supply and has a 50 Ω input and output. The evaluation boards are not available for purchase, but the evaluation board mask is available and can be used to easily construct the boards. The IC's are available in quantity and are free as samples.

4.1.3 Receiver

The receiver section takes the signals from the antenna elements and by way of twelve I/Q demodulators converts the signals into complex baseband. The resulting 24 signals are filtered and sampled before being sent to the DSP via the host interface board.

Demodulator

Two demodulators were investigated, namely the NEC UPC2766GR Wideband IQ Demodulator and the RF Micro Devices RF2903 Integrated Spread Spectrum Receiver. The RF2903 is two stage and the UPC2766GR is single stage; both are rated to only 1 GHz. The UPC2766GR was chosen since it only requires one LO which

simplifies the system and eliminates the LO synchronization problem previously discussed. The device was tested at 1.7 GHz and found to operate nearly within specification. The fact that the device operated well out of the rated frequency range may be a chance occurrence, and it may be required to test many devices in order to get a batch of twelve that function well at 1.7 GHz. As with the power amplifier the assembled evaluation board for this IC can't be purchased, but the board mask is available so the boards can be constructed. The IC's are available free as samples. Similar to the modulator, the LO power for this device can go down to at least –5 dBm without any problem.

An issue with the UPC2766GR is that the circuit design provided for the evaluation board has a phase splitter used to generate the quadrature LO's necessary for I/Q demodulation that is designed for a frequency of 440 MHz (the UPC2766GR does not have an on-chip splitter like the RF2422). Since the circuit is to be operated at 1.7 GHz in this system, the resulting LO signals would not be equal amplitude or in quadrature, which would degrade the system performance since the signal processor requires true quadrature signals to perform optimal array processing. The phase splitter network can be easily redesigned to operate correctly at 1.7 GHz.

Low Pass Filter, Buffer, and Level Setter

The 24 output channels of the demodulators have to be conditioned before entering the analog to digital converter (ADC). Most importantly, the signal from the demodulator must be filtered to reduce noise and to prevent aliasing. A simple first order low pass RC filter should be satisfactory, and the cutoff frequency is proposed to be around 5 kHz. The signal must also be buffered and have its offset and amplitude adjusted to match the ADC input range, which is zero and 20 Volts respectively. Everything can be built around a single op-amp; thus there would be 24 op-amps in total required for this section. If better filtering is required a second order active RC filter could be used, but this would require at least two op-amps in order to accommodate the level setting function as well.

Analog to Digital Converter

An extensive search was conducted for a suitable ADC. The Analog Devices AD7874 was chosen. It has the ability to simultaneously sample and hold four analog inputs and convert all four samples to 12 bit digital numbers at about 29 kHz. This maximum sample frequency is more than adequate for the proposed system. It has a ± 10 Volt input range and ± 5 Volt power supplies. A total of six of these ADC's would be needed to simultaneously convert all 24 signals. The digital data is read out on a bus one sample at a time after conversion is completed. This device has the need for an analog reference input but the analog reference output of one of them can be used to drive the remaining five inputs.

Interface Logic

The interface logic is designed to transfer the sampled signal data from the ADC's to the DSP memory through the host interface board. Figure 4.2 shows the circuit designed for this task. A brief explanation of its operation will be given here. There are four eight bit wide ports on the NB-DIO-32F, and each port can be set to either input or output. For this system two ports are set to input and two to output. Twelve of the bits from the two input ports are used to read the sampled array data. The sixteen output bits are reserved for the transmit data output. One more output bit is necessary for the convert start signal to the ADC's, and so one of the "extra output signal" bits on the NB-DIO-32F is used for this.

The interface logic is designed to transfer the data to the host interface board using handshaking, and the NB-DIO-32F host interface is set to `level mode handshaking`. This allows an unattended transfer to the DSP memory through the host interface; the DSP can request a direct memory access (DMA) transfer to a designated memory location and all 24 numbers will automatically appear there.

The ACK and $\overline{REQ}$ lines shown in the figure are connected to the corresponding handshaking bits on the NB-DIO-32F, and the $\overline{CNVT}$ line is connected to the above

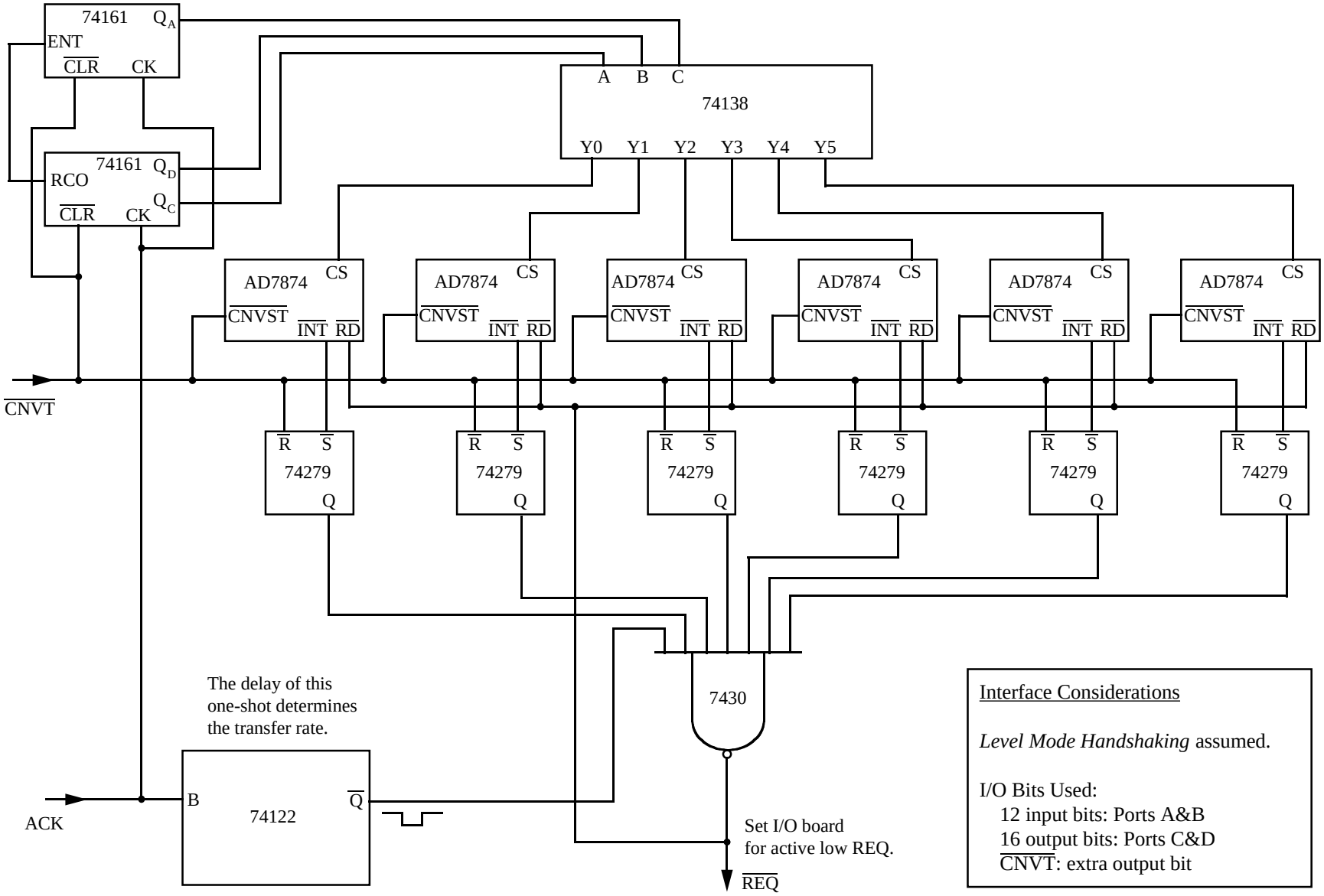


Figure 4.2: Interface Logic For Transferring Sampled Array Data to DSP

mentioned extra output bit on the NB-DIO-32F, to be controlled by the DSP in order to start the ADC conversion.

The two 74161 counters and the 74138 demultiplexer act as a chip select decoder to select which AD7874 is on the bus (the data bus is not shown). In the quiescent state, the one-shot output $\overline{Q}$ is high. When the $\overline{CNVT}$ signal comes in from the DSP, the 74161 counters are reset, the 74279 latches are reset and the AD7874's start their conversion. As each AD7874 finishes, it sets its corresponding latch, and once all the AD7874's have finished the conversion, the $\overline{REQ}$ goes low which triggers a handshake read. The host interface acknowledges the bus read by setting ACK high which increments the counters and triggers the one-shot. Note that the counters are set up so that the divide-by-four output is used as the chip select since there are four reads per AD7874. When the one-shot output $\overline{Q}$ goes high, $\overline{REQ}$ goes low and the cycle is repeated until the host interface has transferred all 24 numbers.

4.2 Power Budget Analysis

A power budget analysis was completed to ensure the system will have an adequate SNR at the input to the ADC [17]. Figure 4.3 shows a block diagram with the parameters relevant to a power budget analysis shown. For the analysis only one transmitter/receive element pair is considered, and the antenna gains are assumed to be unity. As indicated in the modulator section above, the worst case output power from the power amplifier is taken to be 8 dBm. Assuming there are 20 m of coaxial cable from the output to the transmit antenna and that the cable has a loss of 1 dB/m, the radiated power is –12 dBm.

The desired SNR at the ADC is 20 dB which was chosen to ensure a low bit error rate. As shown in the figure, 1 m of coaxial cable will connect an array element to the corresponding demodulator, with a loss of about 1 dB; the noise figure of this cable is then 1 dB as well. The demodulator has a rated noise figure of 24 dB. The

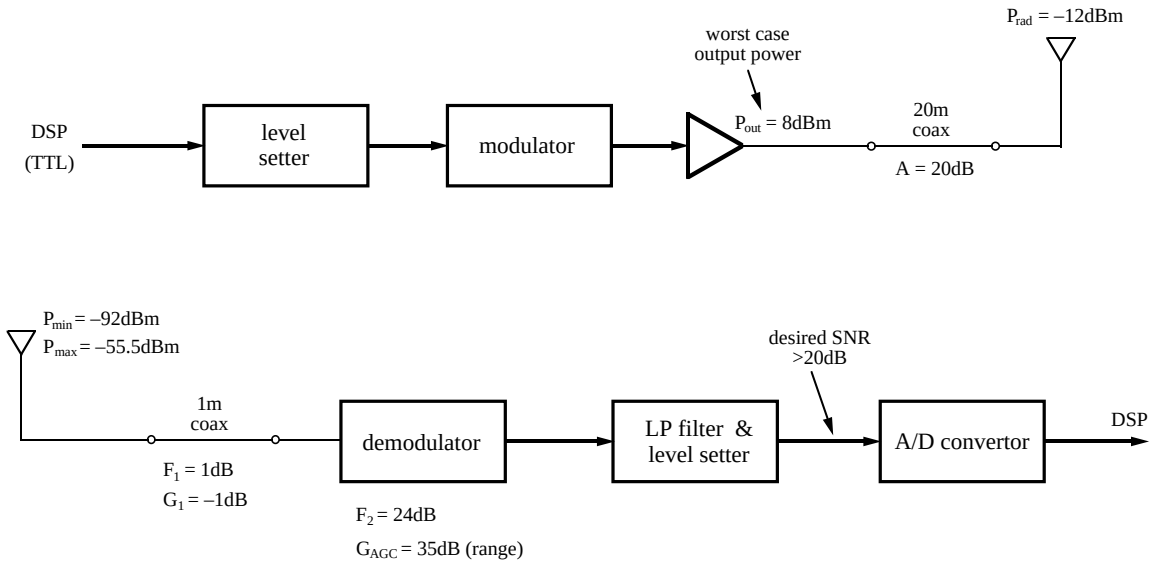


Figure 4.3: Power Budget Analysis of Proposed Prototype System

combined noise figure of the cable and demodulator is found as

$$F = F_1 + \frac{F_2 - 1}{G_1} \tag{4.1}$$

where $F_1 = 1.26$ is the noise figure of the cable, $G_1 = 1/F_1$ is the gain (attenuation) of the cable, and $F_2 = 251$ is the demodulator noise figure. Thus the combined noise figure is

$$F = 1.26 + \frac{251 - 1}{1/1.26} \tag{4.2}$$

$$= 316 \equiv 25\,dB\ . \tag{4.3}$$

The noise floor for the antenna is the standard resistive thermal noise power spectral density of –174 dBm/Hz (at Room Temperature). The bandwidth of the receiver section is limited to 5 kHz by the lowpass filter, and so the noise power of the receive antenna becomes –137 dBm.

Since the input to the ADC must have an SNR of 20 dB, by the definition of noise figure the required SNR at the antenna must be 20+25=45 dB. The minimum received signal power at the antenna then becomes

$$P_{min} = 45\,dB + (-137\,dBm) \tag{4.4}$$

$$= -92\,dBm\ . \tag{4.5}$$

By using the free space attenuation for 20 m of $A_{FS} = -63.5\,dB$ [18], the minimum transmit power is

$$\begin{aligned}(P_{tx})_{min} &= A_{FS} + P_{min} && (4.6)\\ &= 63.5\,dB + (-92\,dBm) && (4.7)\\ &= -27\,dBm\ . && (4.8)\end{aligned}$$

From before, the worst case transmit power is –12 dBm which leaves about 16 dB of margin in the transmit power. This margin could easily be used up by a fade or shadowing effect, for instance, but this is inevitable and the array processing algorithm helps to overcome this by optimally combining the array signals.

The above 20 m used to find the free space loss is used because this is likely the typical maximum distance from a transmit antenna to the receive array; this is also why the length of the cable from modulator to transmit antenna is set to 20 m. Figure 4.3 also shows a $P_{max} = -55.5$ dBm at the receive antenna, which corresponds to the worst case transmit power of –12 dBm attenuated over 2 m of free space, the attenuation of which is –43.5 dB. This would correspond to an approximate maximum power received assuming worst case transmit power, if the transmit antenna were placed close to the receive antenna.

The figure also shows a $G_{AGC} = 35$ dB as a gain range of the demodulator. This gain is set using an analog gain control input pin on the demodulator, and this is set for maximum gain on the evaluation board provided with the demodulator. It is suggested that the gain be left at maximum for all the demodulators in the system. Measurements of the device indicated that output saturation began to occur at an input RF power of about –20 dBm; thus demodulator saturation doesn't seem to be a problem since with the rated saturation power of the power amp of 18 dBm the received power would only be about –45 dBm over 2 m of free space. Even if saturation occured, this probably wouldn't be much of a problem since the phase in the QPSK modulated signal would still be available to the adaptive algorithm as a signal characteristic to be used to provide spatial separation.

4.3 Proposed Antenna Array

A physical design for the array has been proposed with a total of twelve elements. The elements are designed to be uncoupled by about 20 dB which is intended to reduce the fading correlation of the elements. Thus the probability that a given element is in a fade does not depend on whether other elements are in fades. When elements are combined for diversity, independent fading is optimal since any fading correlation present among array elements increases the probability of having the combined signal fade as a result of the elemental fading. Since the proposed array has independent fading, the use of the independent fading model in the simulations described in Chapter 3 is justified. The elements in the proposed array are spaced about a half a wavelength apart.

Chapter 5

Future Work

This report provided a proposal for a prototype hardware system to investigate spatial multiplexing using an antenna array, and results were provided from simulations of the proposed array operation. The following are some suggestions for the direction of future work in these areas.

5.1 Simulations

The simulations completed for this report assumed a flat Rayleigh fading channel. Recall that the simulator generates a matrix C whose elements are independent, zero mean, complex Gaussian variable samples; this simulates the Rayleigh fading model, in which the gain envelope is Rayleigh distributed and the phase is uniformly distributed. The Rayleigh model is typically used for outdoor propagation, where no line of sight between the transmitter and receiver exists. This is the motivation for using zero mean random variables. The indoor channel is usually modelled with the Rician fading model, where the addition of a direct path often found in the indoor environment from transmitter to receiver adds a mean to the random channel gain; the channel gain envelope becomes Rician distributed and the phase is no longer uniformly distributed.

The Rician model could be applied to an ideal uniform plane array[1] by choosing a complex value $c_{mean,j}$ for user j and adding $c_{mean,j}$ to each element of $\mathbf{c}_j$, which from before is the propagation vector for user j. If this were done for each user j, this would simulate a direct path from each user to the array. For the proposed array described in Chapter 4, the mean channel gain from a given user probably isn't the same for each array element as it is in the uniform plane array, and a more complex model may be required to account for this.

The simulations described in this report were not designed to investigate the tracking performance of the LMS algorithm for the array system, since the adaptable weight vector $\mathbf{w}_j$ for each user j was initialized to zero before adapting to each channel matrix C. Thus the results reflected acquisition performance. It would be valuable to simulate the tracking abilities of the system; this would mean choosing a data-to-fading-rate ratio to define the fading rate of the system. The value of the ratio would then be the number of symbols available to train with before a new matrix C is generated, i.e., before one coherence time has elapsed and the channel has thus completely changed. In a practical system, only a fraction of those symbols would be available for a training sequence, since most symbols in a time slot are dedicated to data. The value of $\mathbf{w}_j$ for each user j would not be reinitialized when training begins for a newly generated matrix C, but would be kept the same as when training was stopped for the previous channel matrix.

5.2 Hardware

The hardware prototype will be built, although the design presented in this report is only a suggestion for its construction. The main purpose of the prototype is to examine the potential of static spatial multiplexing by itself, without complicating factors such as dynamic environments and ISI being present. The low data-to-fading-

[1] A uniform plane array is defined to be an array of omnidirectional monopole elements equally spaced in a planar matrix above a ground plane.

rate ratio may make studying the effects of a dynamic channel on the acquisition phase impossible, and realistic tracking experiments may be difficult. Because of the low symbol rate, accurately measuring the BER may require a length of time on the order of hours (for the part of the BER curve where the BER is low), and the propagation environment would have to be kept extremely stable since at 1.7 GHz it doesn't take much movement, either by one of the antennas or by an object in the channel environment, to radically change the propagation vector. Perhaps a better measure of the performance after adaptation would be the mean square error since this could be acquired in a shorter period of time. Ultimately, a full duplex system should be constructed in order to test both array transmission as well as reception.

Chapter 6

Summary

The background theory for interference cancelling using an adaptive array was presented. The logical progression of this method to spatial multiplexing where a number of mutual interferers are spatially separated and thus isolated using array processing was then introduced. A number of array adaptation methods were described; the LMS algorithm was chosen for the simulations described in this report due to its computational simplicity. The results of a proof described in the literature were presented, namely that for each antenna added to an array being used for spatial division multiple access, one more degree of diversity is available to each user.

The simulations completed for this report were described. Their purpose was primarily to investigate what affects the LMS algorithm training time for a static environment. Specifically, the number of users was varied between two, four, eight and twelve in order to examine the effect on training time that varying the numbers of users has (the inter-user effects) and also what affects the training time for a fixed number of users (the intra-user effects).

The results show that when varying the number of users, the received SINR dominates, i.e., increasing the number of users typically decreases the SINR for a given user, which slows down the convergence rate for the LMS algorithm. This result was expected since adding more users increases the noise as seen by any one user.

As for the intra-user effects, the received SINR dominates the training time. The eigenvalue spread shows some correlation with training time, but the relationship is not nearly as pronounced as for the SINR. Two other effects were examined, namely the power spread (the variance of received powers for the various users) and the effect of the orthogonality of the propagation vectors; neither of these factors showed any effect on convergence speed.

A hardware prototype system design was given, with a detailed description of each component in the system. The prototype is intended as a proof of concept of a spatial multiplexing system using state of the art, readily available components. Only the receive array half will be constructed, with the transmit side left for future work. A twelve element array will be used, and a number of portable transmitters will be available to provide the signals. A DSP will drive the system. It will send out random data to be modulated and transmitted by QPSK. The signals received by the array and subsequently demodulated will be sampled and fed back into the DSP to be used in an adaptive algorithm with the data that was transmitted used as the training sequence. After training, the converged weight coefficients will be used to obtain a performance metric, perhaps the mean square error.

Appendix A

Correlation Between Training Sequences

Because simultaneous training is used, the mutual correlation between the training sequences becomes important since any correlation present will slow down the convergence of the algorithm and may increase the misadjustment after convergence has occured. In an experiment to investigate the correlation between any two of the 1000 symbol training sequences that were used for training, no significant correlation was found. These results will be described shortly, but first the random number generator used for the simulations will be discussed.

The function *rand()* in Matlab uses the pseudorandom number generator described by Park and Miller [19]. Specifically, the prime modulus multiplicative linear congruential generator with multiplier 16807 and prime modulus $2^{32}-1$ is used directly to produce a pseudorandom sequence of uniform distribution and period $2^{32}-2$. This period (approximately 4.29×10^9) is significantly larger than the periods of codes used in present communications systems, and for practical purposes can be considered infinite. For the simulation described in this report which is implemented in Matlab, the amplitude of the output from *rand()* is shifted so that it's output range is centred at zero; positive and negative samples are then mapped to -1 and $+1$ respectively

to form bipolar bits for the training sequences. For multiple users being trained, two samples from *rand()* are taken to form one QPSK symbol for the first user, then two more samples for the next user, and so on, until all users have their first training symbol. Then the process is repeated for all remaining training symbols.

From before, the inphase and quadrature bipolar bit components are, respectively, a_n and b_n. The training sequence for a given user can then be represented as

$$\{a_1\, b_1\, a_2\, b_2\, \ldots\, a_n\, b_n\, \ldots\, a_{1000}\, b_{1000}\}\ ,$$

which represents 1000 QPSK symbols since there are two bits per symbol. If $s_j(i)$ is used to represent element i from the above training sequence for user j, the cross-correlation between the sequences for user 1 and 2 is defined as

$$\varrho_{1,2} = \frac{\sum_{i=1}^{2000} s_1(i)s_2(i)}{2000}\ .$$

Note that the correlation is normalized by 2000 so that if a sequence is correlated with itself, the autocorrelation is unity, and cross-correlation between two sequences will lie between positive and negative unity.

To investigate the correlation between training sequences in the simulation, 10^4 pairs of sequences were generated and their cross-correlation measured. The sequences were generated exactly as in the simulation when there are two users present. The results, namely the mean, maximum, minimum, and standard deviation of the resulting 10^4 values, are shown in Table A.1.

Statistic	**Value**
mean	-2.50×10^{-4}
max	8.40×10^{-2}
min	-8.40×10^{-2}
std. dev.	2.22×10^{-2}

Table A.1: Correlation Between Two Training Sequences (Statistics for 10^4 Trials)

The mean correlation is extremely low, and will not affect the convergence of an adaptive algorithm using these sequences. The maximum, which has the same magnitude as the minimum, is about 300 times the mean value, but is still insignificant. The standard deviation shows that the spread of the values around the mean is large compared with the mean, but when considered in an absolute manner is insignificant.

Although the correlation is very low, some aspects of the sequences should be considered. The pseudorandom number generator is seeded to a different value each time, which may mean the sequence taken for various simulation trials will occur in vastly different places in the entire sequence available. This correlation experiment only considered a very small part of the entire sequence. There may be particular sequences that could be chosen which would have have large cross-correlations, but on average the correlation is probably small, as shown by this experiment.

Another noteworthy aspect is that this experiment involved the cross-correlation between two sequences, whereas in the simulation up to twelve simultaneous sequences are present, and the correlation among them all must be considered. Since the sequences are generated in parallel, i.e., the first element of *all* the sequences is found before the second, etc., the correlation characteristics of this particular pseudorandom generator may be different depending on how many simultaneous sequences are being used.

If simultaneous training of this type is used in a new commercial system design, orthogonal training sequences could be used, but if applied to an existing system, the sequences of symbols available for training may not necessarily be orthogonal. The reduction in performance because of this must be accounted for when implementing a SDMA system using simultaneous training.

Bibliography

[1] Hamid Krim and Mats Viberg. Two decades of array signal processing research. *IEEE Signal Processing Magazine*, 13(4), July 1996.

[2] Derek Gerlach and Arogyaswami Paulraj. Adaptive transmitting antenna arrays with feedback. *IEEE Signal Processing Letters*, 1(10), October 1994.

[3] Jack H. Winters, Jack Salz, and Richard D. Gitlin. The impact of antenna diversity on the capacity of wireless communication systems. *IEEE Transactions on Communications*, 42(2/3/4), February/March/April 1994.

[4] Jack H. Winters. Signal acquisition and tracking with adaptive arrays in the digital mobile radio system IS-54 with flat fading. *IEEE Transactions on Vehicular Technology*, 42(4), November 1993.

[5] John Litva and Titus Kwok-Yeung Lo. *Digital Beamforming In Wireless Communications.* Artech House, Boston, 1996.

[6] Simon Haykin. *Adaptive Filter Theory.* Prentice Hall, Englewood Cliffs, NJ 07632, 2nd edition, 1991.

[7] Charles L. B. Despins, David D. Falconer, and Samy A. Mahmoud. Compound strategies of coding, equalization, and space diversity for wideband TDMA indoor wireless channels. *IEEE Transactions on Vehicular Technology*, 41(4), November 1992.

[8] Jack H. Winters. Optimum combining in digital mobile radio with cochannel interference. *IEEE Journal On Selected Areas In Communications*, SAC-2(4), July 1984.

[9] Jack H. Winters. Optimum combining for indoor radio systems with multiple users. *IEEE Transactions on Communications*, COM-35(11), November 1987.

[10] Amir Dembo and Jack Salz. On the least squares tap adjustment algorithm in adaptive digital echo cancellers. *IEEE Transactions on Communications*, 38(5), May 1990.

[11] J.E. Hudson. *Adaptive Array Principles*. Peter Peregrinus Ltd., Stevenage, U.K., 1981.

[12] William F. Gabriel. Adaptive arrays — an introduction. *Proceedings of the IEEE*, 64(2), February 1976.

[13] Marek Klemes. A practical method of obtaining constant convergence rates in lms adaptive arrays. *IEEE Transactions on Antennas and Propagation*, AP-34(3), March 1986.

[14] Allen L. Edwards. *An Introduction to Linear Regression and Correlation*, chapter 4. W. H. Freeman and Company, San Francisco, USA, 1976.

[15] Brent Petersen. Personal correspondence, May 1997.

[16] R.N. McDonough. A note on eigenvector decomposition of correlation matrices. *Journal of the Acoustical Society of America*, 68(1), July 1980.

[17] R.E. Ziemer and W.H. Tranter. *Principles of Communications: Systems, Modulation, and Noise*. Houghton Mifflin Company, Boston, 2nd edition, 1985. Appendix A.

[18] Ron Johnston. ENEL 577 Course Notes, 1995.

[19] Stephen K. Park and Keith W. Miller. Random number generators: Good ones are hard to find. *Communications of the ACM*, 31(10), October 1988.