Single-Block Coded Modulation for MIMO Systems

Mohammad Janani and Aria Nosratinia, *Senior Member, IEEE*

***Abstract*—This paper introduces a new class of space-time codes that achieve coding gain without a trellis or any form of inter-block dependency. The construction of the new codes starts from an existing (parent) space-time block code (STBC). Then by increasing the constellation size followed by expurgation of the expanded codebook, a better code is obtained at the original transmission rate. This method can be applied to a wide variety of space-time block codes, including orthogonal codes and quasi-orthogonal codes. A multi-stage design algorithm is presented, and for orthogonal parent codes, an efficient decoding algorithm is developed, and its decoding complexity is analyzed. Despite altering the regular structure of the orthogonal code, the decoding complexity is only affected by a constant factor.**

***Index Terms*—Space-time code, MIMO, coded modulation.**

I. Introduction

We propose a new class of space-time *block* codes that provide coding gain without the benefit of an outer code or a trellis.

Space-time block codes (STBC) such as [1], [2], [3], [4] are motivated by simple decoding schemes and space-time diversity. However, they do not provide much (if any) coding gain. Space-time *trellis* codes (STTC) [5] provide diversity as well as coding gain, but have higher decoding complexity and also higher decoding delay compared to the block codes. Trellis codes were also built over orthogonal STBC components by Jafarkhani and Seshadri [6] (*super-orthogonal codes*) and also independently by Siwamogsatham and Fitz [7]. Super-orthogonal codes have coding gain, but due to the trellis they also have higher delay and complexity compared with block codes. In contrast, our goal is to maintain relatively small delay and complexity, so we wish to introduce coding gain within the structure of the block code itself.

We use a methodology similar to Ungerböck, namely expanding the constellation size, followed by a careful choice of codewords from the enlarged constellation. The scalar signal constellations transmitted along each MIMO dimension is expanded, and the expurgated code is chosen from among all expanded code vectors.

Although the basic idea seems easy enough, the execution is nontrivial due to structure as well as dimensionality of MIMO signaling. The signal expansion happens in a higher dimensional space, and in general it is not clear how to systematically pick approximate centers of packed spheres from the elements of a given higher-rate lattice in high-dimensional spaces. Furthermore, the search space for expurgation can be very large, even with a modest number of antennas and constellation dimension. For example, for a STBC with L_t transmit antennas antennas transmitting QAM, a constellation expansion by a factor of two results in a factor of 2^{2L_t} expansion in total number of codewords.

In this paper, we start with parent codes that are either orthogonal (OSTBC) or quasi-orthogonal (QOSTBC) space-time block codes. We then expand the codebooks by using higher-order transmit modulations, then expurgate the resulting codebook down to desired rate using an incremental, semi-greedy algorithm. Because the resulting codes are no longer OSTBC/QOSTBC, we construct efficient decoding algorithms for them, and analyze their complexity. Although this paper concentrates on OSTBC and QOSTBC parent codes, the proposed method is general and can be applied to other classes of block codes.

II. System Model

The system model consists of a MIMO system with L_t transmit and L_r receive antennas. A frequency-nonselective (flat) block-fading channel is assumed, where the channel gains are constant during each fade interval and independent in successive intervals. The received signal, denoted by a $T \times L_r$ matrix $\mathbf{R}$, after matched filtering has the following form:

$$\mathbf{R} = \sqrt{\frac{\rho}{L_t}}\,\mathbf{SH} + \mathbf{N} \quad . \tag{1}$$

T represents the number of time slots for transmitting one block of symbols. ρ is the average received signal-to-noise ratio per antenna. The matrix $\mathbf{S}$ is a block space-time codeword of size $T \times L_t$. The channel matrix $\mathbf{H} = \{h_{ij}\}$ has the size of $L_t \times L_r$ where h_{ij} is the fading channel coefficient between the receive antenna j and transmit antenna i. The AWGN is shown by the matrix $\mathbf{N}$. The receiver employs maximum likelihood (ML) decoder with perfect knowledge of channel state information.

We now discuss the distance metric used in this work. The commonly used minimum-PEP analysis, which leads to the asymptotic coding gain, is not altogether suitable for our purposes, because it over-emphasizes the minimum distance and under-emphasizes the multiplicity. This metric gives a low score to an otherwise good codebook that has a single bad codeword pair, while giving a more favorable rank to another codebook whose minimum distance is only slightly better, but has many codeword pairs close to the minimum distance. In our work this is undesirable for two reasons. First, the worst-case PEP is essentially an asymptotic metric that is suitable for high-SNR, but may not be appropriate for realistic SNR.

Paper approved by I. Lee, the Editor for Wireless Communication Theory of the IEEE Communications Society. Manuscript received January 25, 2007; revised July 26, 2007.

The authors are with the Department of Electrical Engineering, The University of Texas at Dallas, Richardson, TX 75080, USA (e-mail: aria@utdallas.edu, mh_janani@yahoo.com).

This work appeared in part in the IEEE International Symposium on Information Theory, July 2006, Seattle, WA.

Digital Object Identifier 10.1109/TCOMM.2009.02.070032

Second, experience has shown that the min-PEP metric falters when used with a greedy or semi-greedy multi-stage design algorithm, because a metric that depends, in each design stage, only on two codewords, carries insufficient information about the overall structure of the code. As a result, a greedy optimization with min-PEP is likely to fall into shallow local optima.

Motivated by these considerations, we use a metric that carries information from multiple codewords, known as the average union bound [8]. We introduce the AUB below in a manner similar to [9].

$$P_U = \frac{1}{n_c} \sum_{i,j,i\neq j} P_e(\mathbf{c}_i, \mathbf{c}_j) \tag{2}$$

where $\mathbf{c}_i$ and $\mathbf{c}_j$ are codewords from code S and n_c is the total number of codewords and $P_e(\mathbf{c}_i, \mathbf{c}_j)$ is the pairwise error probability (PEP) expression,

$$P_e(\mathbf{c}_i, \mathbf{c}_j)_{i\neq j} = \frac{1}{\pi} \int_0^{\pi/2} \prod_{i=1}^{r} (1 + \frac{\lambda_i \rho}{4sin^2\theta})^{-L_r} d\theta \tag{3}$$

where $r = \min(L_t, L_r)$ and λ_i are the singular values of

$$\mathbf{A} = (\mathbf{c}_i - \mathbf{c}_j)(\mathbf{c}_i - \mathbf{c}_j)^H.$$

In the remainder of this paper, all codeword optimizations are done via the AUB criterion.

III. Code Design

We start with an existing *parent code*, which in this paper is either an orthogonal or quasi-orthogonal block code transmitting a complex-valued modulation from each antenna. Signal expansion is done via increasing the size of the modulation constellation, e.g., QPSK to 8-PSK. The resulting expanded codebook (OSTBC or QOSTBC) obviously has more codewords than needed for the desired rate. Therefore we expurgate the expanded code down to the original rate, in a manner so that the minimum distance of the codebook is increased compared to the parent code, and thus the performance of the code is improved.

We start with an example of Alamouti code [10] using 8PSK constellation. In this case, we have $L_t = 2$, and the code is

$$X_{2\times 2} = \begin{pmatrix} x_1 & x_2 \\ -x_2^* & x_1^* \end{pmatrix} \tag{4}$$

where x_1 and x_2 are taken from 8PSK constellation. Thus the parent code consists of 64 codewords. We can expand the parent code by substituting 16QAM for 8PSK, which increases the number of codewords to 256. Then the expanded code can be pruned down by a factor of four to arrive at a new codebook which has the original rate. The new code has shown about 0.9dB gain when compared with the original Alamouti code using 8PSK (please see the simulations for details).

The pruning (expurgation) of the extended codebook plays an important role in achieving this gain. We will use the following notations in the sequel: The basic (original) codebook will be denoted $\mathcal{C}$, the expanded codebook $\mathcal{C}'$, and the pruned codebook $\mathcal{C}^*$.

It is clearly not possible to examine all possible pruned codebooks, thus a systematic algorithm is necessary. One may either start from an empty codebook and add codewords successively (incremental approach) or start from $\mathcal{C}'$ and remove codewords from it successively (decremental approach). We choose a greedy incremental approach. The basic algorithm starts by picking a codeword at random from $\mathcal{C}'$. Then, it chooses codewords one after another from $\mathcal{C}'$ according to maximum distance, in other words, at each step, it picks the next codeword to be as far as possible from all previously picked codewords.[1] It continues until the required number of codewords have been selected from the expanded codebook.

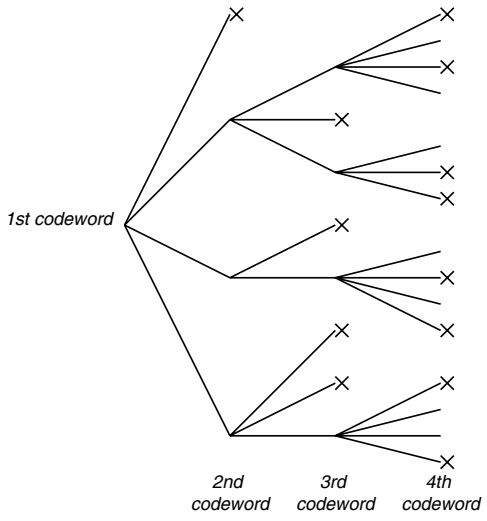


Fig. 1. Incremental codeword selection algorithm. Paths with bad distance properties are terminated with a cross.

This basic algorithm, while displaying the essential ideas, needs to be refined for the following reason. Each greedy choice affects the future actions, therefore picking the very best codeword at each stage may be shortsighted because it may block desirable future options. In other words, a greedy algorithm is susceptible to shallow local optima. To address this issue, we modify the algorithm so that at each step several codewords candidates are retained, instead of a single greedy choice. This leads to a tree-based algorithm. To limit the search complexity we must limit the number of active paths, therefore at each step a few of the least-attractive paths in the tree are terminated. Tree search methods have a long and important history in discrete optimization [11] but further discussion of them will take us away from our main task, and is therefore omitted in the interest of brevity.

Figure 1 shows how the code selection process works for the first four codewords in a hypothetical example. The first codeword is selected arbitrarily. For the second codeword, we consider the best, second best, ... (according to the AUB of pairwise error probability) and include several of the top choices as potential contenders. For each of the possibilities that are alive, a third codeword is sought, and so on. With each new codeword that is added, the pairwise AUB of the new codeword is calculated with respect to the so-far selected codewords. If the new codeword is leading to undesirably high AUB's of error, we terminate the corresponding path in

[1]In practice, we choose the codewords so that the AUB of the error probabilities is minimized.

the selection tree at this codeword (shown with a "x" on the graph). In our fictitious example of Figure 1, after choosing 4 codewords we have 7 surviving candidate codebooks. Once the codebooks have grown to requisite size, we choose the best from among the surviving codebooks.

The initial codeword(s) in the tree have a strong impact on the quality of the eventual codebook. One may start from a few different "seeds" for the first codeword, design several codebooks, and pick the best one. We also found that a good solution is obtained if the optimization is started with a set of well-spaced codewords that are derived by a systematic set-partitioning similar to [6]. This is the method used in all our simulations.

The algorithm consists of simple steps that can be expressed as follows.

1) Select a codeword or a set of codewords as initial set S_1,
2) Augment the sets S_i by adding only one codeword from the expanded code $\mathbf{C}'$,
3) Normalize the power of augmented codeword sets
4) Check the distance of each new normalized sets and only keep the sets that give minimum PEP (using AUB),
5) If number of codewords in S_i is not sufficient then go to Step 2,
6) All remaining S_i are potential new codes.

The normalization in Step 3 can be explained as follows. If a constellation with non-uniform power is used, e.g. 16QAM, then the codewords may not all have the same power. To compare the error performance of the different sets, codebook powers should be the same, otherwise comparisons are not meaningful. Therefore, whenever each set of codewords is augmented, all members of that set are multiplied by an appropriate constant (normalization) so that the average power remains the same. This allows us to compare the *geometry* of the sets independent of the question of transmit power.

IV. Decoding

In the following discussion, we concentrate on OSTBC parent codes; extension to QOSTBC parents is straight forward. The orthogonal codes allow the distance metric to be written as a sum of element-wise metrics, thus simplifying the ML detection. We refer to this construction of the metric, and the corresponding detection algorithm, as *orthogonal decoding* [1].

At first glance, since the codewords of our new codes do not (fully) populate a lattice, the new codebook $\mathcal{C}^*$ does not have enough structure to admit a simple ML decoding similar to OSTBC. However, our knowledge that the codewords of $\mathcal{C}^*$ are a subset of the expanded orthogonal codebook $\mathcal{C}'$, can be used to construct an efficient decoding algorithm.

Our decoding algorithm is motivated by the conditioning ideas that have been used, e.g., in sphere decoding [12]. Consider that each codeword $\mathcal{C}^*$ also belongs to the expanded code $\mathcal{C}'$. Our decoder performs orthogonal decoding according to the expanded codebook $\mathcal{C}'$, arriving at a candidate solution. We then check to see if this candidate is actually a member of the expurgated code $\mathcal{C}^*$. If it is, the decoding is deemed successful. If it is not a valid codeword, we perform a wider search. The flow of decoding is summarized below.

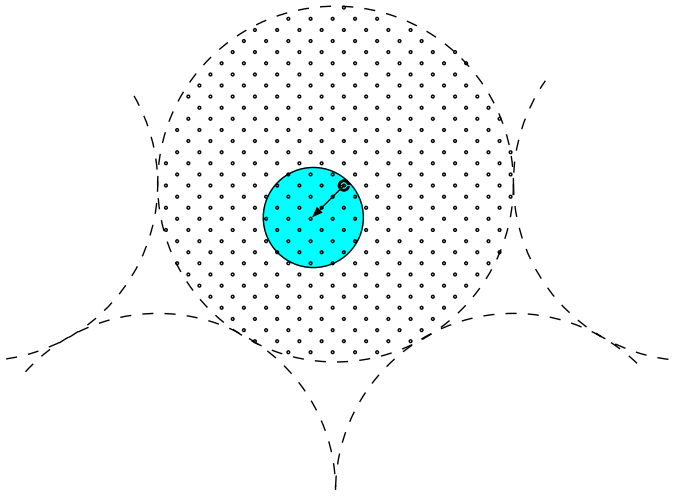


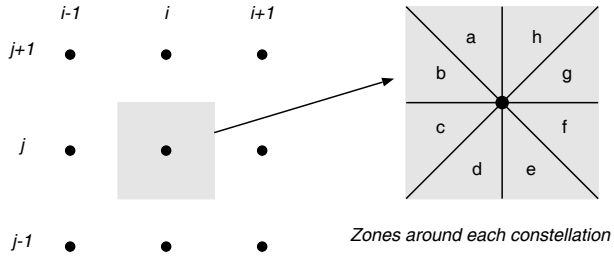
Fig. 2. Demonstrating decoder search. The points are expanded codewords, arrow is noise vector pointing from transmitted codeword to received value. The expanded codewords are searched outwards from received value (shaded area) until a valid codeword is found.

1) Do orthogonal (quasi-orthogonal) decoding in the expanded codebook $\mathcal{C}'$, call the result $\hat{\mathbf{c}}$.
2) If $\hat{\mathbf{c}} \in \mathcal{C}^*$, it is our decoded codeword
3) If $\hat{\mathbf{c}} \notin \mathcal{C}^*$, look for the next closest $\hat{\mathbf{c}} \in \mathcal{C}'$. Continue until a valid codeword of $\mathcal{C}^*$ is found.

When the first decoding choice is not a member of $\mathcal{C}^*$, we use a list decoding method to construct a sequence of candidates until an acceptable one is found. The list decoder successively tests for the second, or third, ... closest codeword in $\mathcal{C}'$ to the received vector until a valid codeword in the pruned codebook $\mathcal{C}^*$ is found, as illustrated in Figure 2. In this figure, each dot is a codeword in $\mathcal{C}'$, the larger dot is the transmitted codeword (from $\mathcal{C}^*$), the arrow is the noise vector, and the shaded area is the search region for the list decoder until the a codeword in $\mathcal{C}^*$ is found. Due to the orthogonal structure of the expanded codebook, each step of the list decoding is easy.

We now describe the specifics of a list decoding algorithm for codes derived from OSTBC. As an example, consider the general M-QAM constellation. The relative distance to nearby constellation points naturally depends on the received signal. To arrive at an ordered list of distances, we start from the nearest constellation point, which was obtained in the first step of decoding. Then, depending on the location of received vector with respect to that nearest point, we can systematically list other nearby constellation points in increasing order of distance. Inspection shows that 8 such lists exist. We divide the area around an MQAM point into 8 regions, and depending where the receive vector falls with respect to its closest constellation point, one of these zones is chosen, and the respective list is used. The idea is that, because these lists have to be calculated and stored only once, the list itself does not impact the decoding complexity, since its execution is equivalent to a table-lookup.

We now briefly outline the process of building lists for the list decoder. This process can be shown via Figure 3. Assume that we have found the closest codeword to the received vector.



Zone	n_1	n_2	n_3	n_4	n_5	n_6 , n_7	n_8 , n_9
a	s_{ij}	$s_{(i+1)j}$	$s_{i(j-1)}$	$s_{(i+1)(j-1)}$	$s_{i(j+1)}$	$s_{(i-1)j}$, $s_{(i+1)(j+1)}$	$s_{(i-1)(j-1)}$, $s_{(i+2)j}$
b	s_{ij}	$s_{i(j-1)}$	$s_{(i+1)j}$	$s_{(i+1)(j-1)}$	$s_{i(j-1)}$	$s_{i(j+1)}$, $s_{(i-1)(j-1)}$	$s_{(i+1)(j+1)}$, $s_{i(j-2)}$
c	s_{ij}	$s_{i(j-1)}$	$s_{(i-1)j}$	$s_{(i-1)(j-1)}$	$s_{(i+1)j}$	$s_{i(j+1)}$, $s_{(i+1)(j-1)}$	$s_{(i-1)(j+1)}$, $s_{i(j-2)}$
d	s_{ij}	$s_{(i-1)j}$	$s_{i(j-1)}$	$s_{(i-1)(j-1)}$	$s_{i(j+1)}$	$s_{i(j+1)}$, $s_{(i-1)(j+1)}$	$s_{(i+1)(j-1)}$, $s_{(i-2)j}$
e	s_{ij}	$s_{(i-1)j}$	$s_{i(j+1)}$	$s_{(i-1)(j+1)}$	$s_{i(j-1)}$	$s_{(i+1)j}$, $s_{(i-1)(j-1)}$	$s_{(i+1)(j+1)}$, $s_{(i-2)j}$
f	s_{ij}	$s_{i(j+1)}$	$s_{(i-1)j}$	$s_{(i-1)(j+1)}$	$s_{(i+1)j}$	$s_{i(j-1)}$, $s_{(i+1)(j+1)}$	$s_{(i-1)(j-1)}$, $s_{i(j+2)}$
g	s_{ij}	$s_{i(j+1)}$	$s_{(i+1)j}$	$s_{(i+1)(j+1)}$	$s_{(i-1)j}$	$s_{i(j-1)}$, $s_{(i-1)(j+1)}$	$s_{(i+1)(j-1)}$, $s_{i(j+2)}$
h	s_{ij}	$s_{(i+1)j}$	$s_{i(j+1)}$	$s_{(i+1)(j+1)}$	$s_{i(j-1)}$	$s_{(i-1)j}$, $s_{(i+1)(j-1)}$	$s_{(i-1)(j+1)}$, $s_{(i+2)j}$

Fig. 3. Zone assignment for s_{ij}, i and j respectively represent the columns and rows of M-QAM constellation and the table shows the neighbors list ordered distance-wise for each zone

Then, the list decoding order will depend on the relative position of the receive vector and its closest codeword, i.e., on the distance between the scalar components of the two vectors. To characterize these lists, we divide the area around a typical two-dimensional constellation point into 8 zones. Once the receive value is in one of these zones, it is easy to unambiguously list close-by constellation points in increasing order of distance.

The table in Figure 3 gives the first nine closest neighbors for each zone. The neighbors are represented as s_{ij} where i represents the column and j represents the row of the given codeword in the QAM constellation structure. The situation at the borders of the constellation is somewhat different, but straight forward: the same ordering is still valid but any points beyond the edge of the constellation will be missing. We omit the details in the interest of brevity.

A. Decoding Complexity

We study the decoding complexity along two directions: the average complexity and the instantaneous complexity. Average complexity, as the name implies, represents the complexity of decoding averaged over all possible codewords and received vectors. As we show below, our codes can be decoded in an average complexity that is polynomial in constellation size. Instantaneous complexity represents the complexity for a given codeword and receive noise. This is an important factor because any implementation of the decoding has finite processing power, and real-time processing requires that if the calculation is not finished within the allowed time, a decoding error must be issued. The instantaneous complexity is of course a random quantity, and we wish to understand its distribution.

1) Average Complexity: In this section we assume that the codewords have a uniform distribution (all codewords are equally likely). Define the expansion factor a as follows

$$a \triangleq \frac{p_e}{p_o} \tag{5}$$

where $p_e = |\mathcal{C}'|$ and $p_o = |\mathcal{C}^*|$ are the cardinalities of the respective codebooks.

We assume that our code is designed well, so the distance between the codewords of $\mathcal{C}^*$ are maximized, as shown in Figure 2. To decode, we can start from the received value, find the nearest codeword in $\mathcal{C}'$ (which is easy), and check if it is also in $\mathcal{C}^*$. If not, we will do a list decoding, namely, go to successively farther members of $\mathcal{C}'$ until we find one that is indeed in $\mathcal{C}^*$ (a valid codeword). We would visit at most a codewords of $\mathcal{C}'$ in this manner. Each of these visits requires an orthogonal decoding. Therefore, the overall complexity is bounded by a multiplied by a constant, which is independent of the constellation size.

2) Instantaneous Complexity: We now look at the list decoding mentioned above. We are now interested to see what is the probability of visiting a given number of expanded codewords before arriving at a valid codeword.

We note that in many practical scenarios, the closest codeword is a valid codeword with high probability, therefore the search terminates after the first try. For example, simulations show that for a 4×1 system employing a BPSK constellation, at codeword error rates of 10^{-2} and 10^{-3}, the probability of being forced into a wider search (after the first try) is less than 0.01 and 0.001 respectively.

Now we calculate an approximation of the probability distribution of list decoding length. This calculation is needed because we need a decoder that, with high probability, can accommodate the depth of list decoding. The following result demonstrates that the list decoding depth has at worst a geometric probability distribution.

Theorem 1: Consider a codebook $\mathcal{C}'$ with p_e codevectors, out of which p_o are labeled "valid." Then if we randomly pick n codewords from $\mathcal{C}'$, the probability of having at least one valid codeword among them is lower bounded by

$$1 - (1 - a^{-1})^n.$$

Proof: The probability of having no valid codeword in n

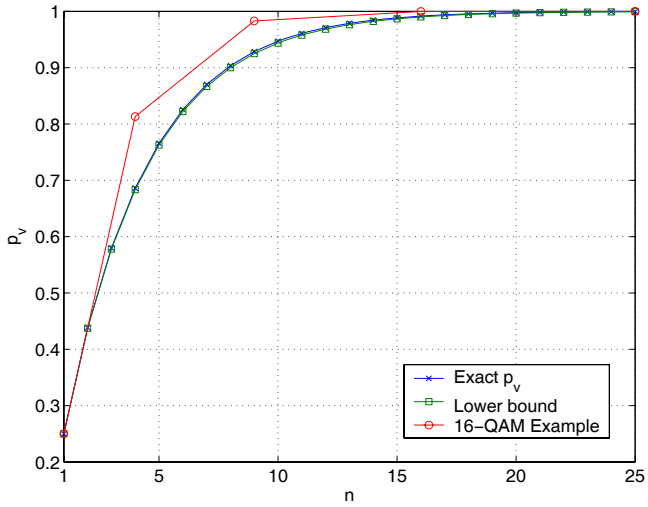


Fig. 4. Probability of having at least one valid codeword for n closest neighbors where $a = 1/4$, $p_o = 64$ and $p_e = 256$.

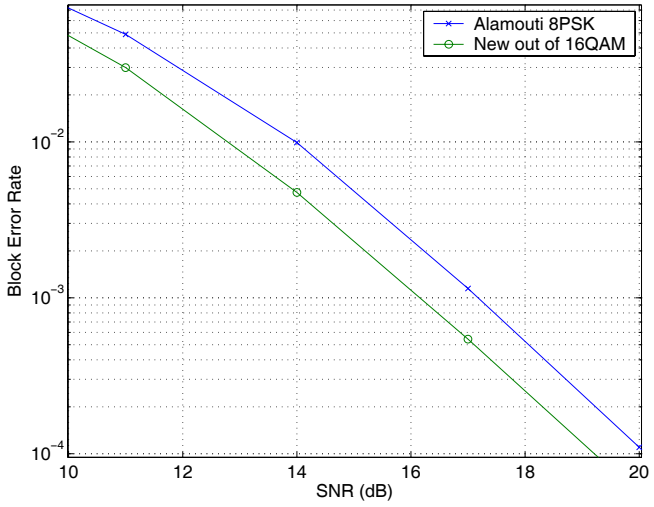


Fig. 5. Orthogonal space time codes and expurgated orthogonal code for $L_t = 2$, $T = 2$ and rate 3 bits per channel use in slow fading.

randomly selected codewords is

$$p_{nv}(n) = \frac{\binom{p_e - p_o}{n}}{\binom{p_e}{n}},$$

where $\binom{p_e}{n}$ gives the number of all possible combinations of choosing n from p_e. Now the probability of having at least one valid codeword is

$$\begin{aligned} p_v(n) &= 1 - p_{nv}(n) \\ &= 1 - \frac{(p_e - p_o)(p_e - p_o - 1)\cdots(p_e - p_o - n + 1)}{p_e(p_e - 1)\cdots(p_e - n + 1)} \\ &> 1 - (\frac{p_e - p_o}{p_e})^n \\ &= 1 - (1 - a^{-1})^n. \end{aligned} \quad (6)$$

■

Since n is in the exponent in (6), p_v converges to one very fast, however, for the code selected via our selection algorithm the convergence is even faster over n. The code selection algorithm tries to separate the codewords as much as possible in the space in a way that having codewords clustered together is highly unlikely. In other words, from every a neighbor codewords of $\mathcal{C}^{'}$, on average one codeword is valid. Therefore, many possible outcomes of random selection are ruled out. Figure 4 illustrate the lower bound and the exact p_v for $p_o = 64$ codewords taken from $p_e = 256$ codewords of the expanded code. For instance $p_v(16) = 0.9915$ and $p_v(25) = 0.9995$. For the example code stated in Table I which is derived from an Alamouti code using 16-QAM, p_v equals to $0.9828, 0.9998$ and 1 respectively for n equals to $9, 16$ and 25. This means the search for the first 25 neighbors, or in other words the first 5 neighbors for each symbol, gives the exact ML result.

V. Codes with Quasi-Orthogonal Parents

We may also design new codes using quasi-orthogonal parent codes. Each quasi-orthogonal codeword consists of two types of sub-matrices, each of them possibly an orthogonal or quasi-orthogonal codeword of lower dimension. The standard decoding process for quasi-orthogonal codes allows the distance metric to be written as a sum of sub-metrics for these two sections, thus simplifying the ML detection. We refer to this construction of the metric, and the corresponding detection algorithm, as *quasi-orthogonal decoding* [6]. The decoding algorithm is similar to the orthogonal codes except the type of decoding.

For codes with quasi-orthogonal parents, we can employ the sphere decoding method suggested by [13] incorporated in the same decoding algorithm seen earlier. When the initial decoded codeword is invalid (not a member of $\mathcal{C}^*$), the list decoding method is applied. This requires us to know the n nearest quasi-orthogonal neighbors (in $\mathcal{C}'$) of the received codeword. One can determine a radius for each of the quasi-orthogonal sub-metrics (as a function of noise power) to ensure that the required number of codewords fall within the neighborhood. The details are straight forward and are omitted for the sake of brevity.

VI. Simulations

The simulation results are provided for two, three, and four transmit antennas in block fading. We focus on examples of $L_t = 2$ and $L_t = 3$ for orthogonal codes. However the generalization for higher number of transmit antennas is straight forward. For each data point, Monte Carlo iterations are performed until 1000 codeword errors are observed.

Figure 5 shows the codeword error rate for the code rate $R = 3$ bits/Hz/Sec and two receive antenna system. The original orthogonal code is an Alamouti code using 8PSK modulation. The code consists of 64 codewords. Our code is a selection of 64 codewords taken from Alamouti code using 16QAM constellation. A gain of 0.9dB has been obtained over the original Alamouti code.

For the case of $L_t = 3$ the code is

$$X_{4\times 3} = \begin{pmatrix} x_1 & x_2 & x_3 \\ -x_2^* & x_1^* & 0 \\ -x_3^* & 0 & x_1^* \\ 0 & -x_3^* & x_2^* \end{pmatrix} \quad (7)$$

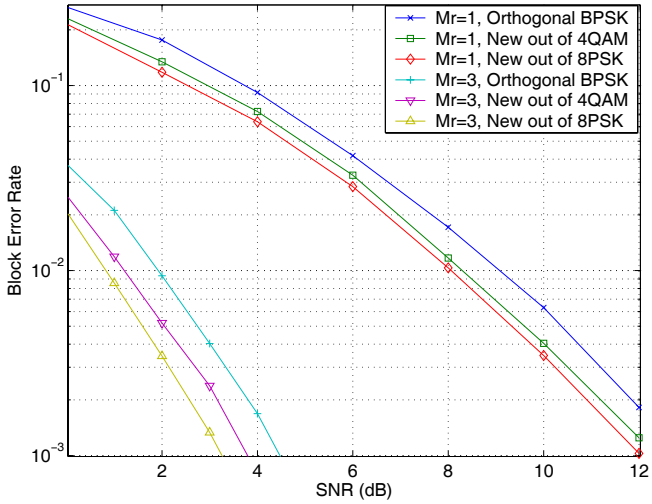


Fig. 6. Orthogonal space time codes and expurgated orthogonal code for $L_t = 3$, $T = 4$ and rate $3/4$ bits per channel use in slow fading.

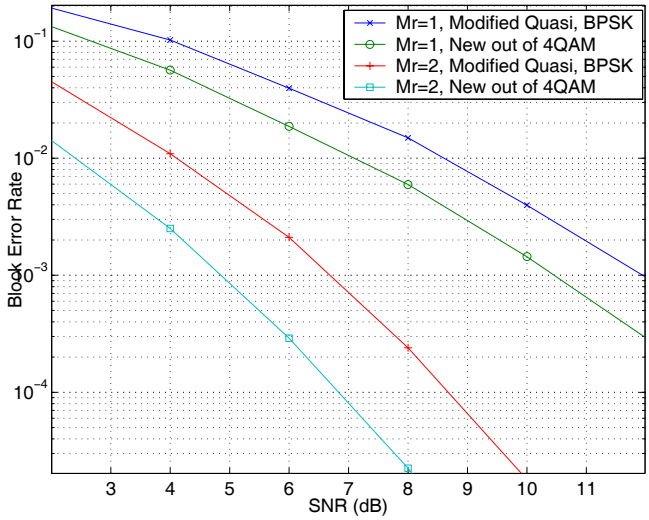


Fig. 7. Modified quasi-orthogonal space time codes and expurgated codes for $L_t = 4$, $T = 4$, and rate 1 bits per channel use in slow fading.

where x_1, x_2, and x_3 are symbols from the chosen constellation and "$*$" denotes the complex conjugate operator.

Figure 6 shows the codeword error rate for $R = 3/4$ bits/Hz/Sec. The original orthogonal code is a 4×3 code of (7) which has 8 codewords and uses BPSK. We expand this code to QPSK and 8PSK, and then choose 8 codewords optimally. Our codes derived from 8PSK show a gain of 1dB and 1.3dB respectively for one and two receive antennas.

When $L_t = 4$, each codeword transmits four symbols at four timing slots. To design our codes, we start with the modified QOSTBC [3], which achieves full diversity. The modified code applies a constellation rotation either on the first pair or the second pair of symbols. The code structure is

$$X_{4\times4} = \begin{pmatrix} e^{j\phi}x_1 & e^{j\phi}x_2 & x_3 & x_4 \\ e^{-j\phi}x_2^* & e^{-j\phi}x_1^* & x_4^* & x_3^* \\ x_3 & x_4 & e^{j\phi}x_1 & e^{j\phi}x_2 \\ x_4^* & x_3^* & e^{-j\phi}x_2^* & e^{-j\phi}x_1^* \end{pmatrix} \quad (8)$$

where $x_1, \ldots, x_4$ are constellation symbols and ϕ is the rotation angle applied on x_1, x_2. For each ϕ, matrices $X_{4\times4}$ constitute a complete quasi-orthogonal codebook. In the original work of [3], the angle ϕ was found so as to provide full diversity and acceptable coding gain.

Figure 7 shows the performance of a different code at rate $R = 1$ bits/Hz/Sec. For this figure, the starting code is the 4×4 modified quasi-orthogonal code of (8) with $\phi = \pi/2$ that uses BPSK and has 16 codewords. We expand this code to QPSK and then choose 16 codewords. A gain of 1.5dB and 1.8dB is achieved respectively for one and two received antennas at 10^{-3} block error rate.

Table I shows the designed codebook, via indices in the expanded codebook. We use a simple labeling according to the following convention. Each M-ary constellation is labeled naturally (not Gray). The codeword symbols are $x_1, \ldots, x_N$ taking values on this constellation. The function label$(\cdot)$ returns the label of a constellation symbol. The indices of the expanded codebook are calculated via:

$$I = \sum_{i=1}^{N} \text{label}(x_i) \cdot M^{N-i} + 1 \quad (9)$$

TABLE I
NEW CODE EXPRESSED WITH THE INDICES OF THE CODEWORDS TAKEN FROM EXPANDED CODE

Code	Constellation	L_r	Indices
$X_{2\times2}$	16-QAM	2	1, 4, 7, 13, 16, 18, 26, 31, 37, 40, 43, 46, 49, 52, 55, 61, 64, 66, 72, 74, 85, 87, 92, 93, 97, 99, 105, 111, 120, 122, 133, 135, 140, 141, 143, 145, 148, 155, 166, 173, 176, 178, 181, 188, 190, 193, 196, 202, 208, 210, 215, 217, 220, 222, 227, 229, 232, 239, 241, 244, 247, 250, 253, 256
$X_{4\times3}$	4-QAM	1	1, 7, 9, 15, 35, 37, 43, 45
$X_{4\times3}$	8-PSK	1	1, 21, 107, 160, 293, 305, 335, 444
$X_{4\times3}$	4-QAM	2	1, 7, 9, 15, 35, 37, 43, 45
$X_{4\times3}$	8-PSK	2	1, 21, 107, 160, 293, 305, 335, 444
$X_{4\times4}$	4-QAM	1	1, 11, 35, 41, 88, 94, 118, 128, 131, 137, 161, 171, 214, 224, 248, 254
$X_{4\times4}$	4-QAM	2	1, 11, 35, 41, 88, 94, 118, 128, 131, 137, 161, 171, 214, 224, 248, 254

Since it has been our goal to concentrate on the fundamentals of set expansion, and we did not wish to tamper with the original codes that we started with, we have not optimized ϕ. In general, one would expect that, since the gains of our technique come from better sphere packing, a better starting codebook will also give a better expanded codebook, *asymptotically* in rate and/or number of antennas. However, it is not inconceivable that with a very small number of codewords and in small dimensions, some gains may be obtained by a joint optimization of ϕ and pruning.

VII. Discussion

The coding gain of the proposed method depends on rate as well as the starting constellation. For example, the gains are often higher when starting with a BPSK constellation in the parent code, in part because of the additional dimension provided by the next higher-order constellation (QPSK). However, we also note that the gains can be nontrivial even when the starting constellation is *not* BPSK, e.g., the example shown in Figure 5.

The diversity of our codes can be easily established: constellation expansion does not diminish diversity, and expurgation also does not diminish diversity, therefore the resulting codebooks automatically inherit the full diversity of the parent codes we have used.

There is not much that can be said in closed form about coding gain. The basic coding gain expressions are available in, e.g., [5], but expressions *specific to our codes* are not easily calculated, because our codes are pruned with a nonparametric discrete optimization, which does not easily lend itself to closed-form expressions.

It is noteworthy that the signal expansion and set partitioning ideas of TCM, which is also repeated in [6], [7] are in principle not too far from the expansion and expurgation processes used in our work. But on the other hand, TCM as well as [6], [7] possess trellis structures and therefore a much higher delay and computational complexity. Therefore despite the similarities, the codes in [6], [7] are in a separate class from our codes.

VIII. Conclusion

We propose new space-time codes that are derived by pruning codewords from an expanded set of orthogonal space-time codewords. We select the codewords from the expanded set through an expurgation algorithm. The resulting codes exhibit attractive coding gains, while maintaining very reasonable decoding complexity.

As a final summary, it is instructive to revisit the types of space-time codes that have significant coding gain. The trellis space-time codes [5] allow coding gain but their complexity is generally higher than block space-time codes. The second type is the super-orthogonal codes of [6], [7] which are constructed by expanding block codebooks (e.g. Alamouti) and building trellises on them. The third type are codes presented in this paper, which incorporate coding gain into the block code itself[2]. Because our codes do not involve a trellis, their decoding complexity is smaller than the space-time trellis codes [5] and super-orthogonal codes of [6], [7].

References

[1] V. Tarokh, H. Jafarkhani, and A. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Trans. Inform. Theory*, vol. 45, no. 5, pp. 1456–1467, July 1999.

[2] H. Jafarkhani, "A quasi-orthogonal space-time code," *IEEE Trans. Commun.*, vol. 49, no. 1, pp. 1–4, Jan. 2001.

[3] N. Sharma and C. B. Papadias, "Improved quasi-orthogonal codes through constellation rotation," *IEEE Trans. Commun.*, vol. 51, no. 3, pp. 563–571, Mar. 2003.

[4] W. Su and X. Xia, "Signal constellations for quasi-orthogonal space-time block codes with full diversity," *IEEE Trans. Inform. Theory*, vol. 50, no. 10, pp. 2331–2347, Oct. 2003.

[5] V. Tarokh, N. Seshardi, and A. Calderbank, "Space-time codes for high data rate wireless communications: Performance criteria and code construction," *IEEE Trans. Inform. Theory*, vol. 44, no. 2, pp. 744–765, Mar. 1998.

[6] H. Jafarkhani and N. Seshadri, "Super-orthogonal space-time trellis codes," *IEEE Trans. Inform. Theory*, vol. 49, no. 4, pp. 937–950, Apr. 2003.

[7] S. Siwamogsatham and M. P. Fitz, "Improved high-rate space-time codes via orthogonality and set partitioning," in *Proc. IEEE WCNC*, vol. 1, pp. 264–270, Mar. 2002.

[8] M. Janani and A. Nosratinia, "Relaxed threaded space-time codes," in *Proc. IEEE GLOBECOM*, St. Louis, MO, Nov. 2005.

[9] M. K. Byun and B. G. Lee, "New bounds of pairwise error probability for space-time codes in Rayleigh fading channels," in *Proc. IEEE WCNC*, vol. 1, pp. 89–93, Mar. 2002.

[10] S. M. Alamouti, "A simple transmit diversity technique for wireless communications," *IEEE J. Select. Areas Commun.*, vol. 16, no. 8, pp. 1451–1458, Oct. 1998.

[11] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification and Regression Trees*. Chapman and Hall/CRC, 1984.

[12] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," *Math Comput.*, vol. 44, pp. 463–471, Apr. 1985.

[13] A. Y. Peng, I.-M. Kim, and S. Yousefi, "Low-complexity sphere decoding algorithm for quasi-orthogonal space-time block codes," *IEEE Trans. Commun.*, vol. 54, no. 3, pp. 377–382, Mar. 2006.

[2]Some block space-time codes operating below one symbol per transmission also have had nominal coding gain, but one can argue that it has been due to signal repetitions within the codeword matrices