Fast Multiple Antenna Differential Decoding

KENNETH L. CLARKSON* WIM SWELDENS* ALICE ZHENG†

*Bell Laboratories, Lucent Technologies
Murray Hill, NJ
{clarkson,wim}@bell-labs.com
http://mars.bell-labs.com

†Department of Electrical Engineering and Computer Sciences
University of California, Berkeley
alicez@cs.berkeley.edu

October 1999, Revised February 2000

Abstract

We present an algorithm based on lattice reduction for the fast decoding of diagonal differential modulation across multiple antenna. While the complexity of the maximum likelihood algorithm is exponential both in the number of antenna and the rate, the complexity of our approximate lattice algorithm is polynomial in the number of antennas and the rate. We show that the error performance of our lattice algorithm is very close to the maximum likelihood algorithm.

Index Terms—Multi-element antenna arrays, wireless communications, fading channels, differential modulation, fast decoding, lattice reduction

1 Introduction

Multiple-antenna wireless communication links allow much higher data rates than traditional single antenna links. Under the assumption that the fading coefficients between the different antenna are independent and known to the receiver, it was proven in [1, 2] that the capacity of a multiple antenna links scales linearly with the lesser of the number of send and receive antennas. Several schemes for modulation and coding over such channels have been proposed [3, 4].

In a mobile wireless environment, the fading coefficients change rapidly and the receiver may not have enough time to learn the fading coefficients. Even if the receiver does not know the fading coefficients, a substantial increase in channel capacity is still possible [5]. For high SNR, it is argued heuristically in [6] that the capacity of the unknown channel is achieved with a new class of so called *unitary space-time* signals. In [7], a systematic construction of unitary space-time signals is presented.

A natural way of dealing with unknown channels is differential modulation. Recently, a multiple antenna differential scheme using unitary space-time signals was introduced [8]. The framework is formally similar to standard one-antenna differential modulation and connects the known channel case with the unknown channel case. In particular [8] introduces so-called *diagonal codes* which are simple to generate: every antenna transmits a PSK symbol in turn.

In [9] a different differential multiple antenna scheme that builds upon the orthogonal designs of [4] was introduced. The advantage of the orthogonal design differential codes is their good performance, and the fact that fast decoding algorithms exist. Their disadvantage is that they only exist for a restricted number of antennas, while diagonal codes exist for any number of antennas. Even though diagonal codes are simple to generate, so far only a slow decoding algorithm has been introduced; this algorithm is exponential in the number of antennas and the rate [8].

In this paper we present a fast approximate decoding algorithm for diagonal multiple antenna differential codes. Our approach relies on approximately recasting the decoding problem into that of finding the closest point in an integer lattice. To solve the latter, we use the *basis reduction* or LLL algorithm introduced by Lovasz, Lenstra and Lenstra in [10]. Our lattice algorithm is polynomial in the number of antennas, and independent of the rate. It approximates maximum likelihood decoding, and we show empirically that its performance is extremely close to it.

2 Single Antenna Differential Modulation

In this section we give a short review of single-antenna differential phase shift keying (DPSK); this review sets the stage for the multiple-antenna differential modulation technique we discuss in the next section. Differential modulation is traditionally used when the channel changes the phase of the symbol in an unknown, but consistent or continuously varying way. The data information is sent in the difference of the phases of two consecutive symbols. Under the assumption that the unknown fading coefficient changes little between two symbols, the difference in phase is preserved and can be used to carry data.

We assume a Rayleigh flat fading channel where the received signals are corrupted by additive noise and fading. We use complex baseband notation: at time t we transmit the signal s_t and we receive the noisy signal x_t at the receiver antenna. The action of the channel is given by:

$$x_t = \sqrt{\rho}\, h_t\, s_t + w_t, \qquad t = 0, 1, \ldots . \tag{1}$$

The additive noise w_t is independent, identically distributed $\mathcal{CN}(0,1)$ which is the complex Gaussian zero-mean unit-variance distribution. The complex-valued fading coefficient h_t is $\mathcal{CN}(0,1)$ distributed but not independent over time. We assume that the fading coefficients change continuously according to a model such as Jakes' [11]. The transmitted signals are normalized to have power one when averaged out over time: $\mathbf{E}|s_t|^2 = 1$. Then ρ represents the expected signal-to-noise ratio (SNR) at the receiver.

For a data rate of R bits per channel use, we need $L = 2^R$ symbols. A common technique is phase shift keying (PSK) which uses symbols that are Lth roots of unity

$$v_\ell = e^{2\pi i \ell / L} \qquad \ell = 0, \ldots, L-1. \tag{2}$$

Suppose we want to send a data sequence of integers $z_1, z_2, \ldots$ with $z_t \in \{0, \ldots, L-1\}$. The transmitter sends the symbol stream $s_1, s_2, \ldots$ where

$$s_t = v_{z_t}\, s_{t-1} \qquad t = 1, 2, \ldots \quad (s_0 = 1). \tag{3}$$

The initial symbol $s_0 = 1$ does not carry any information and can be thought of as a training symbol. If we assume that the fading coefficient changes slowly so that $h_t \approx h_{t-1}$, the maximum likelihood (ML) decoder

is given by

$$\hat{z}_t^{\mathrm{ML}} = \arg \min_l |x_t - v_l \, x_{t-1}|^2.$$

The above expression is equal to

$$|x_t|^2 + |x_{t-1}|^2 - 2\,|x_t \, x_{t-1}| \, \cos(\arg x_t - \arg x_{t-1} - 2\pi l/L).$$

This expression is minimized by minimizing the argument of the cosine. Thus the maximum likelihood decoder can be computed as

$$\hat{z}_t^{\mathrm{ML}} = \lfloor \arg \left(x_t / x_{t-1} \right) L/(2\pi) \rceil \, , \tag{4}$$

where $\lfloor x \rceil$ stands for the integer closest to x. An error is made when $\hat{z}_t^{ML} \neq z_t$. The decoding does not depend on earlier decoding decisions, but only on the received symbols x_{t-1} and x_t; decoding errors therefore do not propagate.

A word about models and approximations is in order here. Differential PSK is a decades old technique used, among many applications, in voice-band modems operating over fixed channels, but when carrier recovery was not desired. Of course the application to slowly varying channels follows naturally. Similarly in our extension to multiple antennas given in the next section, one can as a starting point consider a fixed channel. We mentiond the need for approximate constancy of fading but this is not essential to the definition of the method. However, in the simulations reported on in Section 5 actual Jake fading models are included to evaluate the error rate performance.

3 Multiple Antenna Differential Modulation

3.1 The Rayleigh Flat Fading Channel

Consider a communication link with M transmitter antennas and N receiver antennas operating in a Rayleigh flat-fading environment. We assume a rich scattering environment — either indoor or urban outdoor — where each receiver antenna responds to each transmitter antenna through a statistically independent fading coefficient. The received signals are corrupted by additive noise that is statistically independent among the N receivers and the symbol periods. We also assume a narrow band transmission so no ISI or delay spread occurs. This is equivalent to assuming that the fading process is non frequency selective. In case of wide band transmission delay spread becomes important and can be dealt by subdividing the band using OFDM

and then using the above model in each band.

For differential modulation, it is convenient to group the signals transmitted over the M antennas in time blocks of size M. We will use τ to index the time blocks. The transmitted signals are organized in an $M \times M$ matrix S_τ where the column indices represent the different antennas and the row indices represent the time samples $t = \tau M, \ldots, \tau M + M - 1$. The matrices are power normalized so that the expected Euclidean norm of each row is equal to one. This way the total transmitted power does not depend on the number of antennas. The received signals are organized in $M \times N$ matrices X_τ. Assuming that the fading coefficients are constant within each block of size M, the action of the channel is given by

$$X_\tau = \sqrt{\rho}\, S_\tau\, H_\tau + W_\tau \quad \text{for} \quad \tau = 0, 1, \ldots. \tag{5}$$

Here W_τ is an $M \times N$ matrix of receiver noise coefficients. The noise coefficients are independent across time and receive antenna, and are identically complex normal $\mathcal{CN}(0,1)$ distributed. The $M \times N$ matrix H_τ contains the fading coefficients $h_{\tau;\, m,n}$ which are $\mathcal{CN}(0,1)$ distributed. Within a block the fading coefficients between the different antennas are independent, i.e., $h_{\tau;\, m,n}$ and $h_{\tau;\, m',n'}$ are independent if $m \neq m'$ or $n \neq n'$. However, the fading coefficients are not necessarily independent across blocks, i.e, $h_{\tau;\, m,n}$ and $h_{\tau';\, m,n}$ are not independent. Because of the power normalization, ρ is the expected SNR at the receiver.

3.2 Differential Modulation

The generalization of differential modulation to multiple antennas was introduced in [8] and we here briefly review the main results. Given that a single block takes up M uses of the channel, a rate R requires $L = 2^{RM}$ different signals. Each signal is an $M \times M$ unitary matrix V_ℓ from a constellation $\mathcal{V}$ of L such matrices. The bits to be transmitted are packed into an integer data sequence $z_1, z_2, \ldots$ with $z_\tau \in \{0, \ldots, L-1\}$. The *fundamental differential encoding equation* [8] is given by

$$S_\tau = V_{z_\tau}\, S_{\tau-1} \quad \text{for} \quad \tau = 1, 2, \ldots \quad \text{with} \quad S_0 = I. \tag{6}$$

Clearly all S_τ matrices are unitary as well; hence the power normalization is satisfied. We next assume that the fading coefficients are approximately constant over $2M$ time samples ($H_{\tau-1} \approx H_\tau$). We can substitute

the above equation in (5) to obtain

$$X_\tau = V_{z_\tau}\, X_{\tau-1} + W_\tau - V_{z_\tau}\, W_{\tau-1}.$$

Because the noise matrices are independent and statistically invariant with respect to multiplication by unitary matrices, we may write this as

$$X_\tau = V_{z_\tau}\, X_{\tau-1} + \sqrt{2}\, W'_\tau, \tag{7}$$

where W'_τ is an $M \times N$ matrix of additive independent $\mathcal{CN}(0,1)$ noise. This is the *fundamental differential receiver equation*. The maximum likelihood (ML) decoder [8] is given by

$$\hat{z}_\tau^{\mathrm{ML}} = \arg \min_{\ell=0,\ldots,L-1} \|X_\tau - V_\ell X_{\tau-1}\|_F\,, \tag{8}$$

where the matrix norm is the Frobenius norm

$$\|A\|_F^2 = \mathrm{tr}\,(A^\dagger A) = \mathrm{tr}\,(AA^\dagger) = \sum_{i=1}^{I}\sum_{j=1}^{J} |a_{ij}|^2. \tag{9}$$

We will use this matrix norm throughout the paper.

The quality of a constellation $\mathcal{V}$ is determined by the probability of error of mistaking one symbol of $\mathcal{V}$ for another. In [8] it is shown that the probability of mistaking a signal V_ℓ for a signal $V_{\ell'}$ depends dominantly on the absolute value determinant of $V_\ell - V_{\ell'}$. In particular the quality of a constellation $\mathcal{V}$ is measured by the so-called *diversity product* [8]

$$\zeta = \frac{1}{2} \min_{0\leqslant\ell<\ell'<L} |\det\,(V_\ell - V_{\ell'})|^{1/M}. \tag{10}$$

The diversity product is always between 0 and 1; the closer to 1, the better the error performance. Any constellation with non-zero diversity product is said to have full diversity.

A challenging problem which only recently was posed is how to design constellations of unitary matrices with high diversity product. At this point no general construction exists. In [8] it is argued that for differential modulation it is natural to let the constellations form a group. This way all transmitted matrices S_τ come from the same constellation. Again no general group construction exists. Only for Abelian (commutative) groups is a general design available [8]. These groups are suboptimal, but perform well for rate $R = 1$ or 2. Assuming that the group is Abelian is equivalent to letting all $\mathcal{V}_\ell$ be diagonal, with L'th roots of unity on

M	R	L	$[u_1\, u_2 \cdots u_M]$
2	1	4	[1 1]
3	1	8	[1 1 3]
4	1	16	[1 3 5 7]
5	1	32	[1 5 7 9 11]
6	1	64	[1 7 15 23 25 31]
2	2	16	[1 7]
3	2	64	[1 11 27]
4	2	256	[1 25 97 107]
5	2	1024	[1 157 283 415 487]
6	2	4096	[1 439 789 1539 1911 2015]

Table 1: Diagonal antenna constellations for $M = 1, 2, 3, 4$ and 5 transmitter antennas and rate $R = 1, 2$ that maximize ζ in (10). The number of signals in the constellation is $L = 2^{RM}$.

the diagonal [8]. The decoding method presented in this paper applies to this special case and exploits the diagonal structure of the matrices. In particular the constellation is given by

$$V_\ell = V_1^\ell, \quad \text{where} \quad V_1 = \text{diag}\,[e^{i2\pi u_1/L} \ \cdots \ e^{i2\pi u_M/L}], \quad 0 \leqslant l < L,$$

and u_m are integers between 0 and $L-1$. Without loss of generality we can let $u_1 = 1$. The constellation is thus entirely defined by the integers $u_2 \ldots u_M$. For a given M, the u_m are chosen to maximize the diversity product ζ as defined in (10). Table 1 lists the diagonal constellations used in this paper.

Because $\mathcal{V}$ forms a group every transmitted matrix S_τ belongs to $\mathcal{V}$. This implies that at any given time only one antenna transmits a PSK symbol. Diagonal constellations are thus very simple to implement.

3.3 Maximum likelihood decoding

We focus on the case of $N = 1$ receiver which is most typical for a mobile setting. We later show how to generalize our decoding algorithm to $N > 1$. The received signals form a length M vector X_τ whose elements we denote as $x_{\tau;\,m}$. The maximum likelihood decoder for diagonal codes can be written as

$$\hat{z}_\tau^{\text{ML}} = \arg\, \min_\ell \|X_\tau - V_\ell\, X_{\tau-1}\|_F^2 = \arg\, \min_\ell \sum_{m=1}^{M} |x_{\tau;\,m} - e^{i2\pi u_m \ell/L} x_{\tau-1;\,m}|^2.$$

Using the cosine triangle rule the summands are equal to

$$|x_{\tau;\,m}|^2 + |x_{\tau-1;\,m}|^2 - 2\,|x_{\tau;\,m}\, x_{\tau-1;\,m}|\, \cos(\arg x_{\tau;\,m} - \arg x_{\tau-1;\,m} - 2\pi u_m \ell/L).$$

Given that only the cosine depends on ℓ the maximum likelihood decoder is equivalent to

$$\hat{z}_\tau^{\mathrm{ML}} = \arg\, \max_\ell \sum_{m=1}^{M} A_m^2 \cos((u_m \ell - \varphi_m)\, 2\pi/L), \tag{11}$$

where

$$A_m = |x_{\tau;\, m}\, x_{\tau-1;\, m}|^{1/2} \quad \text{and} \quad \varphi_m = \arg\,(x_{\tau;\, m}/x_{\tau-1;\, m})\, L/(2\pi).$$

Thus A_m is the geometric mean of the modulus of the signals received from antenna m in block $\tau - 1$ and τ while φ_m represents their phase difference in units of $2\pi/L$. We let the $\arg$ operator take values in $[-\pi, \pi)$ so that $\varphi_m \in [-L/2, L/2)$. Clearly A_m and φ_m also depend on τ, but we from now on drop the block index τ to simplify notation.

In the one antenna case ($M = 1$ with $u_1 = 1$) the maximum likelihood decoder can be found at a cost independent of L by rounding φ_1 to the closest integer, cf. (4). However, in case $M > 1$, finding the ℓ that minimizes the above sum is nontrivial. Therefore earlier approaches [8] resorted to a brute force search among the $L = 2^{RM}$ candidates. The cost of this approach is clearly exponential in both the rate and the number of antennas. In the next section we discuss a fast algorithm for finding an approximate solution.

4 Fast Decoding

Surprisingly, a maximization identical to (11) shows up in number theory. To disprove the Mertens Conjecture, which if true would have implied the Riemann hypothesis, Odlyzko and te Riele [12] use a maximization like (11). To quickly find approximate solutions, they use a lattice reduction method. Our approach is inspired by their work.

4.1 Constellations as lattices

We first define a lattice in $\mathbf{R}^M$ as the set of all points

$$\left\{ \sum_{m=1}^{M} a_m\, \mathbf{b}_m \mid a_m \in \mathbf{Z} \right\},$$

where $\{\mathbf{b}_1, \ldots, \mathbf{b}_M\}$ is a set of independent vectors in $\mathbf{R}^M$, called the *basis* of the lattice. We will use bold to denote vectors in $\mathbf{R}^M$.

An important insight is that diagonal constellations can be thought of as M-dimensional lattices. This

comes from the fact that the cosine functions in (11) are 2π periodic and the arguments thus can be reduced to the interval $[-\pi, \pi)$. To do so we use a symmetric modulo operation, denoted as mod*, where mod$^* x$ has range $[-x/2, x/2)$ instead of the usual $[0, x)$. The arguments of the cosine can now be written as

$$[(u_m \ell - \varphi_m) \bmod^* L]\, 2\pi/L.$$

Define the M-vector $\mathbf{u} = [u_1 \cdots u_M]^t$ and let $\mathbf{e}_m$ be the standard unit vectors in $\mathbf{R}^M$. The vectors $\ell\,\mathbf{u}$ mod$^* L$ for $0 \leqslant l < L$ form the part of a lattice which lies in $[-L/2, L/2)^M$. To bring the component ℓu_m in the range $[-L/2, L/2)$ we simply need to add an integer multiple of L. This is equivalent to adding an integer multiple of the vector $L\,\mathbf{e}_m$ to $\ell\,\mathbf{u}$. Thus the lattice is formed by taking integer linear combinations of the vector $\mathbf{u}$ and the vectors $L\mathbf{e}_m$. A basis for the lattice is formed by $\mathbf{u}$ and $L\mathbf{e}_m$ for $2 \leqslant m \leqslant M$.

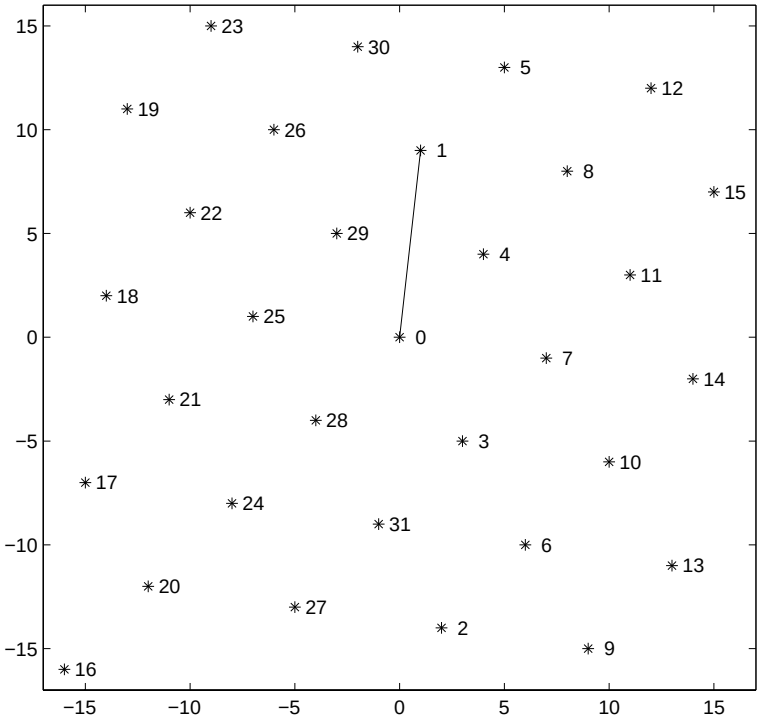

Figure 1: One can think of diagonal constellations as lattices. This is an example in $M = 2$ dimensions where $\mathbf{u} = [1\ \ 9]^t$ and $L = 32$. The element V_ℓ of the constellation is represented by the vector $\ell\,[1\ \ 9]^t \bmod^* 32$.

Example: Consider the case where $M = 2$, $\mathbf{u} = [1\ 9]^t$, and $L = 32$; we end up with 32 vectors in the

square $[-16, 15]^2$, see Figure 1. The lattice is generated by summing integer multiples of the vectors

$$\begin{bmatrix} 1 \\ u_2 \end{bmatrix} \quad \begin{bmatrix} 0 \\ L \end{bmatrix} \quad \begin{bmatrix} L \\ 0 \end{bmatrix}.$$

Clearly the first two form a basis as L times the first one minus u_2 times the second is the third.

4.2 Approximation of the cosine

As mentioned above, we can restrict the arguments of the cosines in (11) to a interval $[-\pi, \pi)$ around zero. Thus maximizing a cosine is equivalent to forcing its argument to be close to zero. For arguments α close to zero the cosine can be approximated as:

$$\cos(\alpha) \approx 1 - \frac{\alpha^2}{2}.$$

Hence we can approximate the maximization of (11) by a minimizing of the sum of the squares of the arguments of the cosines. Then the expression becomes the square of a Euclidean distance. We denote the index which minimizes this approximate likelihood as $\hat{z}^{\text{eucl}}$:

$$\hat{z}^{\text{eucl}} = \arg\min_{\ell} \sum_{m} A_m \left((u_m \ell - \varphi_m) \bmod^* L\right)^2 = \arg\min_{\ell} \sum_{m} \left((A_m u_m \ell - A_m \varphi_m) \bmod^* A_m L\right)^2. \quad (12)$$

The vectors with components $A_m u_m \ell \bmod^* A_m L$ form a lattice, which is the above lattice where each dimension m has been scaled by A_m. Define the diagonal matrix $\mathbf{A}$ as

$$\mathbf{A} = \operatorname{diag}[A_1 \cdots A_m]. \quad (13)$$

The lattice is generated by the basis

$$\mathbf{b}_1 = \mathbf{A}\mathbf{u}, \quad \text{and} \quad \mathbf{b}_m = A_m \mathbf{e}_m = \mathbf{A}\mathbf{e}_m \text{ for } 2 \leqslant m \leqslant M, \quad (14)$$

or equivalently by the columns of the matrix

$$\begin{bmatrix} A_1 & 0 & 0 & \cdots & 0 \\ A_2u_2 & A_2L & 0 & \cdots & 0 \\ A_3u_3 & 0 & A_3L & \cdots & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ A_Mu_M & 0 & 0 & \cdots & A_mL \end{bmatrix} \tag{15}$$

Consider the vector $\mathbf{y}$ with components $A_m\varphi_m$. We can think of $\hat{z}^{\text{eucl}}$ as the index of the lattice vector closest to $\mathbf{y}$ in $\mathbf{R}^M$. This implies that we have transformed our problem to the well known problem finding the closest vector in a lattice.

Example: Figure 2 shows an example of the cosine approximation for the lattice of Figure 1. Assuming that $A_1 = A_2 = 1$, the shaded region are the points $\mathbf{y}$ for which the maximum likelihood decoder returns the origin ($\hat{z}^{\text{ML}} = 0$). We call this the maximum likelihood Voronoi cell. It is found by drawing the curves of points $\mathbf{y}$ for which two lattice points have the same likelihood. For example for $\ell = 0$ and $\ell = 7$ this is the nearly vertical curved line. When using the Euclidean approximation these curved lines become straight lines and the Voronoi cell becomes an exact polygon. In this example the Euclidean and maximum likelihood Voronoi cells are virtually indistinguishable. However, in higher dimensions they may differ more.

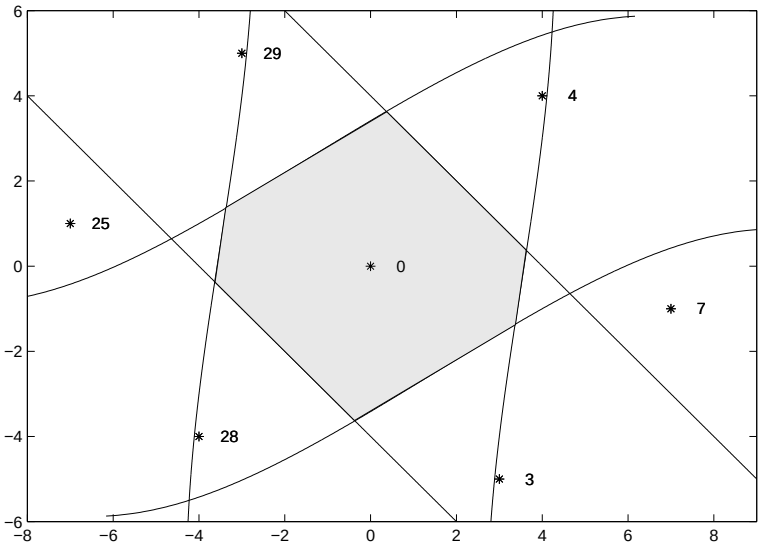

Figure 2: The maximum likelihood Voronoi cell is almost exactly a polygon.

4.3 Finding the closest point in a lattice

Approximating the maximum likelihood decoding with a closest point in a lattice does not immediately lead to a fast algorithm. Finding the closest point in a lattice is NP-hard [13]. Thus any known algorithm will be exponential in the number of dimensions M. However, there is a well-known polynomial-time approximation algorithm introduced by Lenstra, Lenstra, and Lovász in [10]; it uses a technique called *basis reduction* or "the LLL algorithm." The LLL algorithm relies on the observation that when a lattice has an orthogonal basis, the closest point can be found trivially by rounding each component to the closest lattice component. Thus for a given lattice the LLL algorithm attempts to find the "most orthogonal" basis, or more precisely the basis with the shortest vectors, and then use component wise rounding to find the closest lattice point. Finding the basis with the shortest vectors itself is a NP-hard problem; LLL tries to find a basis with reasonably short vectors.

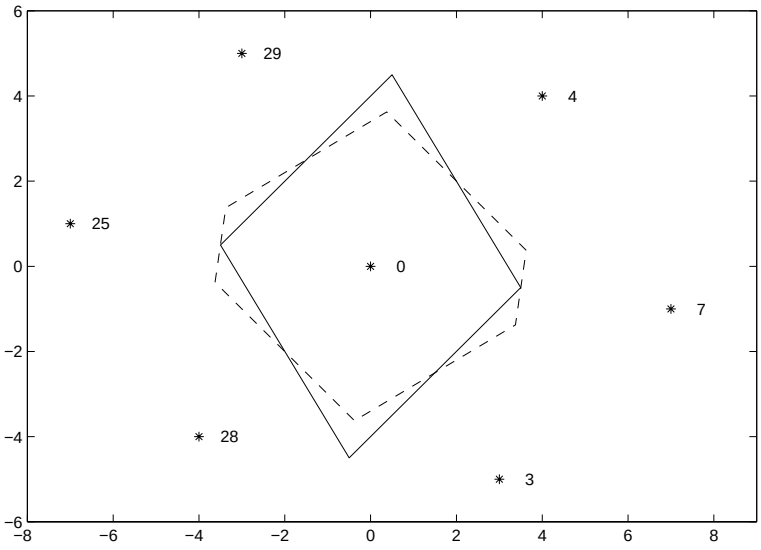

Figure 3: Close up of the lattice of Figure 1. The shortest vectors are (4,4) ($\ell = 4$) and (3,-5) ($\ell = 3$) The solid diamond is the region for which component wise rounding decides the origin is the closest point. The dashed polygon is the true Voronoi cell. Both regions have an area of 32 and tile the plane. The overlap is 90.2%.

Example: Figure 3 shows a close-up of the lattice in Figure 1. The Euclidean Voronoi cell of the origin, i.e., the region for which the origin is the closest lattice point, is the dashed polygon. The original basis is $[1\ 9]^t$ and $[0\ 32]^t$. The vectors $[4\ 4]^t$ ($\ell = 4$) and $[3\ -5]^t$ ($\ell = 3$) are the shortest vectors and form a basis. The solid diamond is the region for which component wise rounding decides the origin is the closest point. Clearly the two regions are quite close. The overlap is 90%; thus for a uniform distribution of query points

component wise rounding would fail in 10% of the queries. However, the distribution of the query points is typically highly peaked around the lattice points, and the performance of component wise rounding can be much better. In higher dimensions the overlap can be much less.

4.4 The LLL algorithm

For completeness, we sketch the basis reduction algorithm; everything in this subsection is explained in more detail in textbooks [14]. Let $\mathbf{b}_1 \ldots \mathbf{b}_M$ denote the original basis of the lattice, as given in (14) above. The basis reduction algorithm does a series of operations of the form

$$\mathbf{b}_i \leftarrow \mathbf{b}_i + a\mathbf{b}_j, \text{ integer } a; \tag{16}$$

such operations yield a different basis for the same lattice. The operations are intended to make the vectors "more orthogonal" to each other, in a way we make precise shortly. The algorithm also swaps (relabels) vectors, exchanging $\mathbf{b}_i$ and $\mathbf{b}_j$. Such an operation clearly does not change the basis set.

The basis reduction algorithm includes the use of Gram-Schmidt (GS) orthogonalization, and also uses some techniques similar to Gram-Schmidt. Recall that the GS orthogonalization is a set of orthogonal vectors $\{\widehat{\mathbf{b}}_1 \ldots \widehat{\mathbf{b}}_M\}$ with the same linear span as $\{\mathbf{b}_1 \ldots \mathbf{b}_M\}$, and can be obtained as follows: for i from 1 to M, obtain $\widehat{\mathbf{b}}_i$ by removing the $\widehat{\mathbf{b}}_j$-components from $\mathbf{b}_i$: that is, set $\widehat{\mathbf{b}}_i$ to $\mathbf{b}_i$, and then for j from $i-1$ down to 1, set

$$\widehat{\mathbf{b}}_i \leftarrow \widehat{\mathbf{b}}_i - \frac{\widehat{\mathbf{b}}_i \cdot \widehat{\mathbf{b}}_j}{\widehat{\mathbf{b}}_j^2}\widehat{\mathbf{b}}_j.$$

Here $\widehat{\mathbf{b}}_j^2$ denotes $\widehat{\mathbf{b}}_j \cdot \widehat{\mathbf{b}}_j = \|\widehat{\mathbf{b}}_j\|^2$. Each such assignment completely removes the $\widehat{\mathbf{b}}_j$-component from $\widehat{\mathbf{b}}_i$, so that $\widehat{\mathbf{b}}_i$ and $\widehat{\mathbf{b}}_j$ are orthogonal. The resulting vector $\widehat{\mathbf{b}}_i$ is also orthogonal to the vectors $\mathbf{b}_j$ for $1 \leqslant j < i$. Note that

$$\mathbf{b}_i = \widehat{\mathbf{b}}_i + \sum_{1 \leqslant j < i} \beta_{ij}\widehat{\mathbf{b}}_j,$$

for some coefficients β_{ij}. Consequently we can use the modified step

$$\widehat{\mathbf{b}}_i \leftarrow \widehat{\mathbf{b}}_i - \frac{\widehat{\mathbf{b}}_i \cdot \widehat{\mathbf{b}}_j}{\widehat{\mathbf{b}}_j^2}\mathbf{b}_j, \tag{17}$$

where the only change is the use of $\mathbf{b}_j$ instead of $\widehat{\mathbf{b}}_j$; because the steps are done in decreasing j order, any $\mathbf{b}_{j'}$-components of $\mathbf{b}_j$ that are added to $\widehat{\mathbf{b}}_i$ are removed later, since $j' < j$.

The basis reduction algorithm tries to make each vector $\mathbf{b}_i$ more orthogonal to the other vectors; an operation such as (17) would be ideal for this, but is not of the form (16). Instead, the algorithm can come close to this by employing *weak reduction*: for $j = i - 1$ down to 1, set

$$\mathbf{b}_i \leftarrow \mathbf{b}_i - \left\lfloor \frac{\widehat{\mathbf{b}}_i \cdot \widehat{\mathbf{b}}_j}{\widehat{\mathbf{b}}_j^2} \right\rceil \mathbf{b}_j, \tag{18}$$

where $\lfloor x \rceil$ denotes x rounded to the nearest integer. With the rounding, we are removing the $\widehat{\mathbf{b}}_j$ component of $\mathbf{b}_i$, up to some $\alpha_{ij}\widehat{\mathbf{b}}_j$ with $|\alpha_{ij}| \leqslant 1/2$.

The result of this operation is that $\mathbf{b}_i$ is *weakly reduced*, meaning that the following holds:

$$\mathbf{b}_i = \widehat{\mathbf{b}}_i + \sum_{1 \leqslant j < i} \alpha_{ij}\widehat{\mathbf{b}}_j, \tag{19}$$

where the coefficients α_{ij} have $|\alpha_{ij}| \leqslant 1/2$.

(We should remark that other sequences of steps of the form (16) can be done that also achieve the weakly reduced state (19); however weak reduction is the simplest method we've seen.)

One problem with the weak reduction steps (19) is that if $\mathbf{b}_1$ is very long compared to the other vectors, then little or no removal of $\mathbf{b}_1$ components from the other vectors may occur. Intuitively, such a vector $\mathbf{b}_1$ should be reduced by the other vectors, instead of the other way around. That is, it would be good to swap a long vector $\mathbf{b}_i$ with some shorter vector $\mathbf{b}_j$ with $j > i$. This is roughly what basis reduction does: having computed the GS orthogonalization and made all basis vectors weakly reduced, the algorithm tests if there is an i such that

$$\widehat{\mathbf{b}}_i^2 > \frac{4}{3}(\widehat{\mathbf{b}}_{i+1}^2 + \alpha_{i+1,i}^2 \widehat{\mathbf{b}}_i^2). \tag{20}$$

If there is such an i, the algorithm interchanges $\mathbf{b}_i$ and $\mathbf{b}_{i+1}$. The GS orthogonalization is done again, and again all basis vectors are weakly reduced. The algorithm repeats this test, and these steps, until no such i is found, and the algorithm stops.

This is a complete overall description of the basis reduction algorithm. There are some obvious inefficiencies in this high-level version; for example, after interchanging $\mathbf{b}_i$ and $\mathbf{b}_{i+1}$, there is no need to recompute $\widehat{\mathbf{b}}_j$ or $\mathbf{b}_j$ for $j < i$. Moreover, for given k, there's no need to compute $\widehat{\mathbf{b}}_j$ or $\mathbf{b}_j$ for $j > k$ until all needed interchanges are done for $i \leqslant k$. A recursive version of the basis reduction procedure is shown in Figure 4. This version removes some of these inefficiencies.

```
proc reduce(i)
  if (i ⩽ 1) return; fi
  while (TRUE) do
      reduce(i − 1);
      for j ← i − 1 downto 1 do b_i ← b_i − ⌊ (b_i · b̂_j) / b̂_j^2 ⌉ b_j od; //weakly reduce
      b̂_i ← b_i;
      for j ← i − 1 downto 1 do b̂_i ← b̂_i − ((b̂_i · b̂_j) / b̂_j^2) b_j od; //update GS vector
      if (b̂_{i−1}^2 ⩽ 4/3(b̂_i^2 + α_{i,i−1}^2 b̂_{i−1}^2)) then exit; fi
      swap b_i and b_{i−1};
  od
```

Figure 4: The basis reduction algorithm.

Plainly the output basis is weakly reduced. Another important property of the output, a consequence of the interchanges based on (20), is that

$$\widehat{\mathbf{b}}_i^2 \geqslant \widehat{\mathbf{b}}_{i-1}^2/2, \tag{21}$$

for $i = 2 \ldots M$, where again $\widehat{\mathbf{b}}_i^2$ denotes $\widehat{\mathbf{b}}_i \cdot \widehat{\mathbf{b}}_i = \|\widehat{\mathbf{b}}_i\|^2$. That is, the orthogonal vectors $\widehat{\mathbf{b}}_i$ are not too far apart in size. Note that since the basis reduction algorithm finished, we have $\widehat{\mathbf{b}}_{i-1}^2 \leqslant \frac{4}{3}(\widehat{\mathbf{b}}_i^2 + \alpha_{i,i-1}^2 \widehat{\mathbf{b}}_{i-1}^2)$ for $i = 2 \ldots M$, or

$$\widehat{\mathbf{b}}_{i-1}^2 \leqslant \frac{1}{3/4 - \alpha_{i,i-1}^2} \widehat{\mathbf{b}}_i^2 \leqslant 2\widehat{\mathbf{b}}_i^2,$$

since $\alpha_{i,i-1} \leqslant 1/2$. This gives the claimed property.

Now suppose we are given a vector $\mathbf{y}$, and we want to find the integer combination of the basis vectors $\mathbf{b}_1 \ldots \mathbf{b}_M$ that is closest to $\mathbf{y}$. We simply perform a weak reduction step for $\mathbf{y}$ using $\mathbf{b}_1 \ldots \mathbf{b}_M$; the total of the subtracted vectors is a lattice element $\mathbf{y}'$ that is near to $\mathbf{y}$: we have

$$\mathbf{x} \equiv \mathbf{y} - \mathbf{y}' = \sum_i \lambda_i \widehat{\mathbf{b}}_i, \tag{22}$$

where $|\lambda_i| \leqslant 1/2$. Then $\mathbf{y}'$ will be our guess for the closest lattice point.

For our decoding problem with original basis vectors given by (14), $\mathbf{y}'$ can be written as

$$\mathbf{y}' = \sum_{m=1}^{M} y'_m \mathbf{b}_m. \tag{23}$$

Thus $\mathbf{y}'$ is equal to $y'_1\mathbf{Au}$ plus multiples of vectors $A_m e_m$ for $2 \leqslant m \leqslant M$. Because $u_1 = 1$, we can find the approximate decoding answer as

$$\hat{z}^{\text{latt}} = y'_1 \bmod L.$$

The decoder answer $\hat{z}^{\text{latt}}$ is only approximate due to the following simplifications:

1. The cosine approximation, i.e, $\hat{z}^{\text{eucl}}$ need not be $\hat{z}^{\text{ML}}$. In other words, the M-dimensional polyhedral Euclidean Voronoi cell does not coincide exactly with the curved maximum likelihood cell.

2. In the basis with the shortest possible vectors the M-dimensional polyhedral Euclidean Voronoi cell is approximated by a M-dimensional parallelepiped.

3. The LLL algorithm does not necessarily find the basis with the shortest possible vectors.

4.5 An Exact Algorithm

This subsection shows that it is possible to apply basis reduction techniques to solve the closest-point problem exactly, in polynomial time for fixed dimension. The main idea is that while the above component wise rounding is not enough to get the exact answer, we can show that not too many different coordinate values need be considered to obtain the closest lattice point. While the algorithm we obtain for proving this complexity bound is probably not of practical use, such a proof shows that the difficulty of the problem is related to the dimension, not the size of L. A similar result was already known for the similar problem of finding the shortest vector in a lattice [14]; it is possible that the our algorithm is already "folklore."

Suppose we are given a vector $\mathbf{y}$, and we want to find the nearest vector in the lattice. As above, we perform a weak reduction step for $\mathbf{y}$ using $\mathbf{b}_1 \ldots \mathbf{b}_M$, to obtain lattice element $\mathbf{y}'$. We are left with the problem of finding the lattice point nearest to the vector $\mathbf{x} = \mathbf{y} - \mathbf{y}'$.

We claim the following: a lattice point $\mathbf{z}$ closest to $\mathbf{x}$ can be expressed as $\mathbf{z} = \sum_i \alpha_i \mathbf{b}_i$, where $|\alpha_M| \leqslant 2^{M/2}$. This bound, and some brute-force search, yield the algorithm: for $k = -2^{(M-1)/2} \ldots 2^{(M-1)/2}$, we recursively search for the closest lattice point to $\mathbf{x}$, subject to the condition that the point has form $k\mathbf{b}_M + \sum_{i<M} \alpha_i \mathbf{b}_i$. Equivalently, let $\mathbf{x}' = \mathbf{x} - k\mathbf{b}_M$, and look for the closest point to $\mathbf{x}'$ in the lattice generated by $\{\mathbf{b}_1 \ldots \mathbf{b}_{M-1}\}$.

To prove the bound on the size of α_M, note that (21) implies that $\widehat{\mathbf{b}}_M^2 \geqslant \widehat{\mathbf{b}}_i / 2^{M-i}$. We have also

$\mathbf{x}^2 \leqslant \sum_i \widehat{\mathbf{b}}_i^2/4$, from (22), and $\mathbf{z}^2 \leqslant 4\mathbf{x}^2$, since $\mathbf{z}$ must be at least as close to $\mathbf{x}$ as the origin. We have

$$2^M \widehat{\mathbf{b}}_M^2 \geqslant \sum_i \mathbf{b}_i^2 \geqslant 4\mathbf{x}^2 \geqslant \mathbf{z}^2 \geqslant \alpha_M^2 \widehat{\mathbf{b}}_M^2,$$

and so $\alpha_M^2 \leqslant 2^M$, yielding the claim.

In the recursive application of the algorithm, note that by the construction of the orthogonalization, we can use $\mathbf{x}' - (\mathbf{x}' \cdot \widehat{\mathbf{b}}_M / \widehat{\mathbf{b}}_M^2)\widehat{\mathbf{b}}_M$ and get the same answer. That is, we can ignore λ_M, and the subproblem is $(M-1)$-dimensional; thus a recursive solution will satisfy the relevant conditions and it is enough to consider coefficients α_i with $\alpha_i^2 \leqslant 2^i$.

The resulting algorithm has a running time proportional to $2^{(M+1)M/4+M}$, but polynomial in the other parameters (rate and length of the operands).

4.6 Multiple antenna receiver

We show here how lattice decoding can be used in the case of a multiple antenna receiver. For N receivers, the received symbol X_τ is an $M \times N$ matrix with elements $x_{\tau;\, m,n}$. The maximum likelihood decoder is now given by

$$\hat{z}_\tau^{\mathrm{ML}} = \arg\, \min_\ell \|X_\tau - V_\ell\, X_{\tau-1}\|_F^2 = \arg\, \min_\ell \sum_{n=1}^{N} \sum_{m=1}^{M} |x_{\tau;\, m,n} - e^{i2\pi u_m \ell/L} x_{\tau-1;\, m,n}|^2.$$

This in turn can be written as

$$\hat{z}_\tau^{\mathrm{ML}} = \arg\, \max_l \sum_{n=1}^{N} \sum_{m=1}^{M} A_{m,n}^2 \cos((u_m \ell - \varphi_{m,n})\, 2\pi/L), \tag{24}$$

where

$$A_{m,n} = |x_{\tau;\, m,n}\, x_{\tau-1;\, m,n}|^{1/2} \quad \text{and} \quad \varphi_{m,n} = \arg\,(x_{\tau;\, m,n}/x_{\tau-1;\, m,n})\, L/(2\pi).$$

The maximization (24) is formally similar to the one in (11). Using the same cosine approximation as before, we see that the decoding problem can be recast as a closest point in a lattice of MN dimensions and we can use the approximate lattice decoding.

Using N receive antennas multiplies the dimension of the lattice by N. For increasing dimensions, brute force search algorithms quickly become unusable, and fast approximate algorithms like LLL are the only option.

5 Results

M	R	$L = 2^{RM}$	T^{ML}	T^{latt}
2	1	4	.04	.08
3	1	8	.08	.11
4	1	16	.13	.17
5	1	32	.26	.26
6	1	64	.54	.41
2	2	16	.11	.09
3	2	64	.47	.15
4	2	256	1.77	.24
5	2	1024	7.49	.39
6	2	4048	33.8	.56
2	3	64	.42	.09
3	3	512	3.61	.16
4	3	4096	27.6	.23
5	3	32768	240	.44
6	3	262144	2170	.72

Table 2: Times in milliseconds for an implementation of the brute force maximum likelihood algorithm (T^{ML}) and the lattice algorithm (T^{latt}). For rate 1 the lattice algorithm is faster for more than 32 symbols. For rate 2 and higher the lattice algorithm always wins. Even for constellations with over 200K symbols, the lattice algorithm finishes in less than a millisecond. Times are measured on a SGI R10000 at 195MHz.

We implemented the maximum likelihood and lattice decoding algorithms for the diagonal differential codes assuming one receiver. Table 2 gives the decoding times in milliseconds for the maximum likelihood (T^{ML}) and lattice algorithm (T^{latt}) on a SGI R10000 processor at 195MHz. For rate 1 the lattice algorithm is faster for more than 32 symbols. For rate 2 and more the lattice algorithm always wins. For a constellation with 260K symbols, the maximum likelihood algorithm needs 2 seconds while the lattice algorithm finishes in less than a millisecond. We see that the running time of the lattice algorithm depends much less on the rate than it does on the number of antennas. This is partly because the cost of the operation in software is independent off the lengths of the operands. Our implementation uses *C++* with the STL template library. Because of the overloading of the common arithmetic vector operators, there is a fair amount of time spent allocating memory for temporary vectors in the lattice algorithm. For a fixed number of antennas one could build an optimized fixed length vector allocation routine, but we did not do this. For real time decoding at a sample rate of 30kHz, one has to be able to run the decoding algorithm in $M \times .03$ msec. Our software prototype already almost obtains this; it is clear that an optimized implementation or a hardware implementation can easily run in real time.

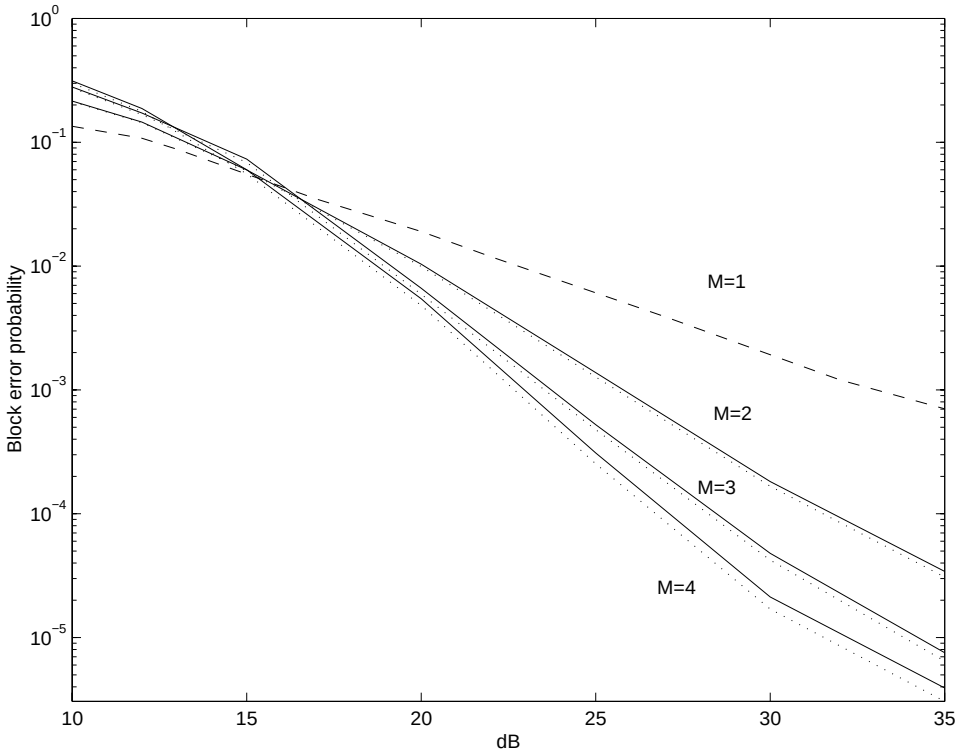

Figure 5: Block error rate for constellations with $R = 2$ and $M = 2, 3, 4$. The error rate of the lattice algorithm is shown in full while the maximum likelihood is dotted. Clearly they are very close. For comparison we also included a single antenna DPSK scheme (dashed).

We also computed the error performance of the lattice algorithm versus maximum likelihood. We do so by computing the block error rate for both algorithms, i.e, the relative number of times that $\hat{z}_\tau^{\mathrm{ML}}$ or $\hat{z}_\tau^{\mathrm{latt}}$ is not equal to z_τ. The fading per antenna is correlated in time according to Jakes' model [11]. We assume that the carrier is 900MHz, the sampling rate is 30kHz, and the mobile receiver moves at 55 mph. The Jakes correlation after t time samples then is $J_0(2\pi 0.0025t)$ where J_0 is the zero-order Bessel function of the first kind. Figure 5 shows the error performance for rate $R = 2$ codes on 2,3, and 4 antennas. The codes used are given in Table 1. The solid lines are the block error rates of the lattice algorithm and the dotted lines are the rates of the maximum likelihood algorithm. Clearly the performance is very close. We found the lattice algorithm to be between 1% worse for low SNR to 10% worse for high SNR. For comparison we also put in the performance of a single antenna DPSK scheme (dashed). As already observed in [7, 8] the error rate drops significantly when going to multiple antennas.

6 Conclusion and Future Work

In this paper we showed how diagonal differential codes for M transmit antenna can be thought off as lattices in M dimensions. This insight allows the use of the well known LLL algorithm as a fast approximate decoding algorithm. We show that the performance of the lattice algorithm is close to the maximum likelihood algorithm.

There are several directions for future work. It may be possible to get a better basis than the LLL algorithm returns: in the test (20) used to decide if swapping is needed, the value $4/3$ can be replaced by a value closer to 1. This makes the output basis more orthogonal, at the cost of more iterations of the reduction algorithm; perhaps the number of iterations could be bounded at some fixed limit to assure that decoding is not too slow. It may also be possible to incorporate the cosine measure into the closest vector calculation, using a small amount of local search.

At this point very little is known about how to design good diagonal codes and one typically resorts to exhaustive or random searches. The connection with lattices perhaps can add insight in how to design diagonal codes. For example, one can approximately relate the diversity product of a diagonal code to the minimum distance between two lattice points.

So far we have only studied the block error rate. To find the bit error rate one needs to assign bit sequences to each of the L symbols. Again the interpretation as lattices may help in find good bit allocation schemes.

Acknowledgements

Special thanks to James Mazo for help with the exposition and to Amin Shokrollahi for many stimulating discussions and help on the LLL algorithm. Alice Zheng was partially supported by a Lucent GRPW Fellowship.

References

[1] I. E. Telatar, "Capacity of multi-antenna Gaussian channels," tech. rep., Bell Laboratories, Lucent Technologies, 1995. Also published in *European Telecommunications Transactions,* November 1999.

[2] G. J. Foschini, "Layered space-time architecture for wireless communication in a fading environment when using multi-element antennas," *Bell Labs. Tech. J.*, vol. 1, no. 2, pp. 41–59, 1996.

[3] V. Tarokh, N. Seshadri, and A. R. Calderbank, "Space-time codes for high data rate wireless communication: Performance criterion and code construction," *IEEE Trans. Inf. Thy.*, vol. 44, pp. 744–765, 1998.

[4] V. Tarokh, H. Jafarkhani, and A. R. Calderbank, "Space-time block codes from orthogonal designs," *IEEE Trans. Inf. Thy.*, vol. 45, pp. 1456–1467, July 1999.

[5] T. L. Marzetta and B. M. Hochwald, "Capacity of a mobile multiple-antenna communication link in Rayleigh flat fading," *IEEE Trans. Inf. Thy.*, vol. 45, pp. 139–157, 1999.

[6] B. M. Hochwald and T. L. Marzetta, "Unitary space-time modulation for multiple-antenna communication in Rayleigh flat-fading," tech. rep., Bell Laboratories, Lucent Technologies, 1998. Download available at `http://mars.bell-labs.com`.

[7] B. Hochwald, T. Marzetta, T. Richardson, W. Sweldens, and R. Urbanke, "Systematic design of unitary space-time constellations," tech. rep., Bell Laboratories, Lucent Technologies, 1998. To appear in IEEE Trans. Inform. Th. Download available at `http://mars.bell-labs.com`.

[8] B. Hochwald and W. Sweldens, "Differential unitary space time modulation," tech. rep., Bell Laboratories, Lucent Technologies, March, 1999. To appear in IEEE J. Comm. Th. Download available at `http://mars.bell-labs.com`.

[9] V. Tarokh and H. Jafarkhani, "A differential detection scheme for transmit diversity," tech. rep., AT&T Laboratories, 1999.

[10] A. K. Lenstra, H. W. Lenstra, and L. Lovász, "Factoring polynomials with rational coefficients," *Math. Ann.*, pp. 515–534, 1982.

[11] W. C. Jakes, *Microwave Mobile Communications*. Piscataway, NJ: IEEE Press, 1993.

[12] A. M. Odlyzko and A. J. J. te Riele, "Disproof of the Mertens conjecture," *J. Reine Angew. Math.*, no. 357, pp. 138–160, 1985.

[13] P. van Emde Boas, "Another NP-complete problem and the complexity of computing short vectors in a lattice," Tech. Rep. 81-04, Math. Inst. Univ. Amsterdam, 1981.

[14] L. Lovász, *An Algorithmic Theory of Numbers, Graphs, and Convexity*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 1986.