Achieving Near-Capacity on a Multiple-Antenna Channel

Bertrand M. Hochwald and Stephan ten Brink

***Abstract*—Recent advancements in iterative processing of channel codes and the development of turbo codes have allowed the communications industry to achieve near-capacity on a single-antenna Gaussian or fading channel with low complexity. We show how these iterative techniques can also be used to achieve near-capacity on a multiple-antenna system where the receiver knows the channel. Combining iterative processing with multiple-antenna channels is particularly challenging because the channel capacities can be a factor of ten or more higher than their single-antenna counterparts. Using a "list" version of the sphere decoder, we provide a simple method to iteratively detect and decode any linear space–time mapping combined with any channel code that can be decoded using so-called "soft" inputs and outputs. We exemplify our technique by directly transmitting symbols that are coded with a channel code; we show that iterative processing with even this simple scheme can achieve near-capacity. We consider both simple convolutional and powerful turbo channel codes and show that excellent performance at very high data rates can be attained with either. We compare our simulation results with Shannon capacity limits for ergodic multiple-antenna channel.**

***Index Terms*—Bell Labs Layered Space–Time (BLAST), concatenated codes, fading channels, receive diversity, soft-in/soft-out, sphere decoding, transmit diversity, turbo codes, wireless communications.**

I. Introduction and Model

ONE WAY to get high rates on a scattering-rich wireless channel is to use multiple transmit and/or receive antennas [1], [2]. Many of the practical space–time schemes that achieve these high rates, such as Bell Labs layered space–time (BLAST) [1], orthogonal designs [3], [4], and linear dispersion codes [5] are designed to have simple symbol detection at the receiver because they map the symbols linearly to the transmit antennas. The codes of [3] and [4] have very simple detectors and are generally designed to optimize a raw block or bit pairwise error performance criteria, while the codes in [5] are designed to optimize an information-theoretic criterion.

However, any effort to achieve capacity on a channel usually requires some form of "outer" channel code that provides redundancy and/or interleavers to guard against bursty fading, interference, and additive receiver noise. In this case, the space–time encoder or mapper acts like an "inner" code that transmits symbols that have redundancy introduced by the outer code. At the receiver, the space–time detector is, therefore, confronted by symbols that are correlated through the channel code, thus significantly complicating the detection process.

The space–time transmission scheme and channel code can be combined, as in the trellis codes of [6], but these combined codes are generally designed by hand and have exponential state complexity in the number of antennas. We seek simple schemes that work for any combination of transmit and receive antennas, and at the high capacities that these antennas promise on a flat-fading channel.

We propose a method to iteratively detect and decode any linear space–time mapper that is combined with an outer channel code. By a linear space–time mapper, we mean that the symbols to be transmitted on each antenna should be linear functions of the encoded data stream; the method, therefore, applies to many existing space–time mapping schemes. The channel code can be any code that may be decoded using "soft" inputs and outputs. Convolutional and turbo channel codes [7] are natural candidates. The method works in an iterative fashion to approximate the optimal joint detector/decoder. It is simple to implement, computationally tractable, and, as we show, can be used to achieve near-capacity on a multiple-antenna channel.

At the heart of our method lies a modified version of the so-called "sphere decoder" [8]. The sphere decoder is introduced for space–time processing in [9], where it is used to compute the maximum-likelihood (ML) symbol estimate with complexity comparable, at a high signal-to-noise ratio (SNR), to the vertical (V)-BLAST nulling/cancelling algorithm [10]. Our modification provides a list of candidates at the detector that allows us to compute, with low complexity, the bit posterior probabilities needed for our iterative decoder.

Some examples of iterative methods that combine channel codes and space–time processing include [11]–[14]. These studies are limited to small signal constellations or few antennas because the underlying optimal detection algorithm is often exponentially complex in one or both. Other examples include [15] and [16], where the suboptimal V-BLAST nulling/cancelling detection is combined with iterative processing. Another iterative method [17] uses a suboptimal group-nulling/cancelling approach to detection. Our method approaches the performance of optimal joint detection and decoding, while avoiding the exponential complexity in the number of antennas and data rate. We are, therefore, able to handle huge rates (tens of bits/channel use) at very low error probabilities.

Our method is fastest when the linear space–time mapper gives us at least as many equations as unknown symbols at the

Paper approved by H. Leib, the Editor for Communication and Information Theory of the IEEE Communications Society. Manuscript received October 16, 2001; revised June 27, 2002. This paper was presented in part at the Allerton Conference on Communication, Control, and Computing, Monticello, IL, October 2001.

B. M. Hochwald is with Bell Laboratories, Lucent Technologies, Murray Hill, NJ 07974 USA (e-mail: hochwald@lucent.com).

S. ten Brink is with Bell Laboratories, Lucent Technologies, Holmdel, NJ 07733 USA (e-mail: stenbrink@lucent.com).

Digital Object Identifier 10.1109/TCOMM.2003.809789

receiver. When there are at least as many receive antennas as transmit antennas, this condition is generally satisfied. When there are more transmit than receive antennas, space–time mappers [3]–[5] can be used to achieve this condition by adding spatial redundancy. Nevertheless, our method can be used without any space–time mapper at all.

For simplicity, our simulations consider only examples where the number of receive antennas equals the number of transmit antennas. No special space–time mapping is used and, as we show, we are able to approach capacity with a simple interleaver, off-the-shelf turbo code, and our iterative detector. No special multi-antenna code-design notions such as "diversity" [6], [18]–[20] are needed.

Our capacity comparisons assume that the receiver knows the fading channel characteristics, which are changing sufficiently rapidly that the channel can be viewed as "ergodic." We do not consider a static model, where a comparison with an "outage" capacity would be more appropriate.

A. Linear Model for Multiple-Input/Multiple-Output (MIMO) Channel

Let $\mathbf{s}$ be an $M \times 1$ vector of symbols (also referred to as "vector constellation symbol") whose entries are chosen from some complex constellation $\mathcal{C}$ [e.g,. quaternary phase-shift keying (QPSK), 16-quadrature amplitude modulation (QAM)] with 2^{M_c}, $M_c \geq 1$ possible signal points, and let $\mathbf{y}$ be an $N \times 1$ vector of received signals (also referred to as "vector channel symbol") related by

$$\mathbf{y} = \mathbf{H}\mathbf{s} + \mathbf{n} \tag{1}$$

where $\mathbf{H}$ is a complex matrix, known perfectly to the receiver, and $\mathbf{n}$ is a vector of independent zero-mean complex Gaussian noise entries with variance σ^2 per real component. We assume that the vector $\mathbf{s} = [s_1, \ldots, s_M]^T$ obeys the component-wise energy constraint $\mathrm{E}||s_m||^2 = E_s/M$: this normalization makes the total transmitted power E_s. Many narrowband flat-fading space–time transmission schemes can be written in this form. For example, BLAST [1] uses the transmit antennas to send a layered structure of signals, and therefore, M represents the number of transmit antennas, N represents the number of receive antennas, and $\mathbf{H}$ is the true MIMO matrix channel. Other examples include orthogonal designs [3], [4], and linear dispersion (LD) codes [5] where $\mathbf{H}$ is an *effective* channel derived from one or more uses of the true channel. In this case, M and N are generally only proportional to the number of transmit and receive antennas. We refer to any use of the transmit antennas such that $\mathbf{H}$ represents the true channel as *direct transmission*. BLAST is an example of direct transmission.

We assume that the vector model (1) is used repeatedly to transmit a continuous stream of data bits, separated into blocks representing uses of the channel. For any given block, let the components of the symbol vector $\mathbf{s}$ be obtained using the mapping function $s_m = \mathrm{map}(\mathbf{x}^{<m>})$, $m = 1, \ldots, M$ (i.e., gray mapping [21]) where $\mathbf{x}^{<m>}$ is an $M_c \times 1$ vector (block) of data bits, and M_c is the number of bits per constellation symbol. The vector of bits transmitted during one application of the model (1) is written $\mathbf{x}$; it is obtained by concatenating $\mathbf{x}^{<1>}, \ldots, \mathbf{x}^{<M>}$, such that the transmitted vector constellation symbol is $\mathbf{s} = \mathrm{map}(\mathbf{x})$. The uncoded transmitted information rate is then $M \cdot M_c$ bits per use of the channel (1). We designate a sequence of blocks $\mathbf{x}$ by $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots$.

If the information bits in the blocks $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots$ are uncoded, then decisions on the bits can be made either by nulling/cancelling [10] or ML on a block-by-block basis. We, however, consider $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots$ to themselves be the output of a channel code of rate $R \leq 1$ that introduces redundancy and correlation between its entries. The transmitted information rate is then RMM_c bits per effective channel use, and it is suboptimal for the signal detector and channel decoder to operate separately and only on individual blocks. The detector should make decisions jointly on all the blocks using knowledge of the correlations across blocks introduced by the channel code, and the channel code should decode using likelihood information on all the blocks obtained from the signal detector. An iterative method to accomplish joint detection and decoding is presented in the next section.

II. Iterative Detection and Decoding

We regard the channel code and the MIMO channel as a serially concatenated scheme [22], with an outer channel encoder (typically a convolutional or turbo code), bit interleaver, and inner space–time constellation mapping with block encoding matrix $\mathbf{H}$. To decode $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots$ optimally, the joint detector/decoder should compute the likelihood of each bit given all the blocks of received complex data $\mathbf{y}^{(1)}, \mathbf{y}^{(2)}, \ldots$ and the constraints imposed by the channel code. Generally, with codes of even reasonable block lengths, this is computationally infeasible. Therefore, we are often content to solve the simpler problems of having the MIMO detector incorporate soft reliability information provided by the channel decoder, and the channel decoder incorporate soft information provided by the MIMO detector. Information between the detector and decoder is then exchanged in an iterative fashion until desired performance is achieved. While this iterative process is not strictly optimal, it has been shown that the "turbo principle" is very effective and computationally efficient in other joint detection/decoding problems [23]–[26]. In this section, we describe the basic principles of iterative detection and decoding while emphasizing the portions that are important to the model (1) and leaving the well-known channel coding details to references.

Fig. 1 gives a flowchart of the iterative algorithm that we use. The detector takes channel observations $\mathbf{y}$ and *a priori* knowledge L_{A_1} on the inner coded bits and computes new (also referred to as "extrinsic") information L_{E_1} for each of the $M \cdot M_c$ coded bits per vector channel symbol $\mathbf{y}$. Then L_{E_1} is deinterleaved to become the *a priori* input L_{A_2} to the outer soft-in/soft-out decoder (maximum *a posteriori* (MAP), *a posteriori* probability (APP), Bahl–Cocke–Jelinek–Raviv (BCJR) algorithm [27], [28]) which calculates extrinsic information L_{E_2} on the outer coded bits. Then L_{E_2} is reinterleaved and fed back as *a priori* knowledge L_{A_1} to the inner detector, thus completing a cycle or "iteration." Each iteration reduces the bit-error rate (BER) by this exchange of information. In

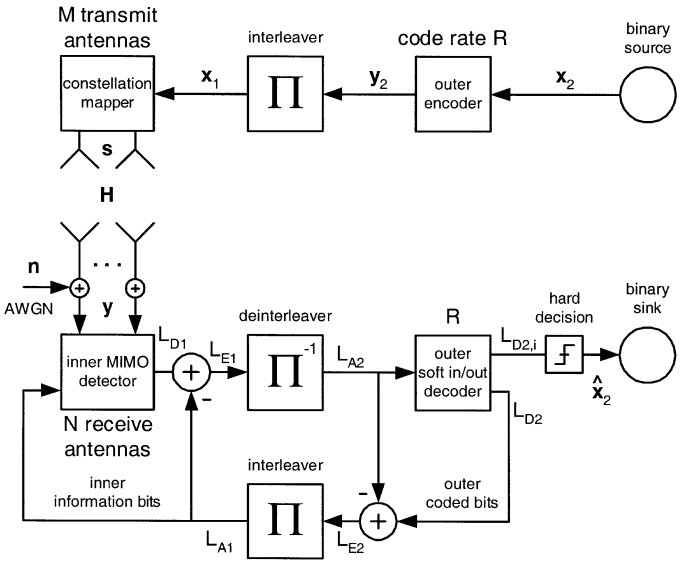


Fig. 1. Transmitter, MIMO channel, and receiver with iterative detection and decoding. The subscript "1" denotes variables associated with the inner code (consisting of space–time mapping and channel), and the subscript "2" denotes variables associated with the outer channel code. For example, $\mathbf{x}_1$ is the same as the variable $\mathbf{x}$ in (1).

Fig. 1, the subscript "1" denotes processing blocks that are connected with the inner mapping/detection operation, whereas the subscript "2" denotes processing blocks connected to the outer encoding/decoding operations. One complete cycle of information exchange between the sections labeled "1" and "2" is an iteration.

Although the overall flow of the algorithm described in Fig. 1 is generally accepted and standard, the actions within the subblocks largely determine the overall complexity and feasibility of the algorithm. Since we are using standard convolutional or turbo channel codes, the outer encoder and decoder are also relatively standard. However, because we are using the multiantenna model (1), the inner detector must be carefully designed to be computationally efficient at the high data rates we are considering. We focus in detail on the detector in Section III, and now describe how to compute the various *a priori* and extrinsic quantities used in Fig. 1.

A. MAP Bit Detection

Maximizing the MAP or APP for a given bit minimizes the probability of making an error on that bit. The APP is usually expressed as a log-likelihood ratio value (*L*-value [23]). *L*-values provide a convenient notation for describing the operation of iterative decoding algorithms; simple add/subtract operations are sufficient to separate *a priori* or old information from new ("extrinsic") information obtained during an APP detection/decoding cycle. As shown in Fig. 1, usually only extrinsic information is exchanged in processing cycles. A decision is made from an *L*-value by using its sign to tell whether the bit is a one or zero. The magnitude of the *L*-value indicates the reliability of the decision; *L*-values near zero correspond to unreliable bits. In this paper, the logical zero for a bit is represented by amplitude level $x_k = -1$, and logical one by $x_k = +1$.

We assume, for the moment, that we are working on a block of bits $\mathbf{x}$ corresponding to one use of the linear model (1). (For the moment, we omit the subscripts used in Fig. 1.) The *a posteriori* *L*-value of the bit x_k, $k = 0, \ldots, M \cdot M_c - 1$, conditioned on the received vector channel symbol $\mathbf{y}$, is

$$L_D\left(x_k|\mathbf{y}\right) = \ln \frac{P\left[x_k = +1|\mathbf{y}\right]}{P\left[x_k = -1|\mathbf{y}\right]}. \tag{2}$$

We assume that the bits in $\mathbf{x}$ have been encoded with a channel code, but that an interleaver at the encoder is used to "scramble" the bits from other blocks into our block, so that the bits within $\mathbf{x}$ are approximately statistically independent of one another. Using Bayes' theorem, and exploiting the independence of $x_0, \ldots, x_{M \cdot M_c - 1}$ by splitting up joint probabilities into products, we can write the soft output value as

$$L_D(x_k|\mathbf{y}) = L_A(x_k) + \underbrace{\ln \frac{\sum_{\mathbf{x} \in \mathbb{X}_{k,+1}} p\left(\mathbf{y}|\mathbf{x}\right) \cdot \exp \sum_{j \in \mathbb{J}_{k,\mathbf{x}}} L_A\left(x_j\right)}{\sum_{\mathbf{x} \in \mathbb{X}_{k,-1}} p(\mathbf{y}|\mathbf{x}) \cdot \exp \sum_{j \in \mathbb{J}_{k,\mathbf{x}}} L_A(x_j)}}_{L_E(x_k|\mathbf{y})} \tag{3}$$

where $\mathbb{X}_{k,+1}$ is the set of $2^{M \cdot M_c - 1}$ bit vectors $\mathbf{x}$ having $x_k = +1$; that is

$$\mathbb{X}_{k,+1} = \{\mathbf{x}|x_k = +1\}, \quad \mathbb{X}_{k,-1} = \{\mathbf{x}|x_k = -1\}. \tag{4}$$

$\mathbb{J}_{k,\mathbf{x}}$ is the set of indices j with

$$\mathbb{J}_{k,\mathbf{x}} = \{j | j = 0, \ldots, M \cdot M_c - 1, j \neq k,\ x_j = 1\} \tag{5}$$

$$L_A(x_j) = \ln \frac{P[x_j = 1]}{P[x_j = -1]}. \tag{6}$$

By multiplying the numerator and denominator with $\exp[-1/2 \cdot \sum_{k=0}^{M \cdot M_c - 1} L_A(x_k)]$, we may write (3) as

$$L_D\left(x_k|\mathbf{y}\right) = L_A(x_k) + \underbrace{\ln \frac{\sum_{\mathbf{x} \in \mathbb{X}_{k,+1}} p\left(\mathbf{y}|\mathbf{x}\right) \cdot \exp\left(\frac{1}{2}\mathbf{x}_{[k]}^T \cdot \mathbf{L}_{A,[k]}\right)}{\sum_{\mathbf{x} \in \mathbb{X}_{k,-1}} p\left(\mathbf{y}|\mathbf{x}\right) \cdot \exp\left(\frac{1}{2}\mathbf{x}_{[k]}^T \cdot \mathbf{L}_{A,[k]}\right)}}_{L_E(x_k|\mathbf{y})} \tag{7}$$

where $\mathbf{x}_{[k]}$ denotes the subvector of $\mathbf{x}$ obtained by omitting its kth element x_k, and $\mathbf{L}_{A,[k]}$ denotes the vector of all L_A values, also omitting x_k. Thus, L_D can be written as a sum of *a priori* *L*-value L_A and extrinsic *L*-value L_E. These manipulations are standard and more details may be found in [23].

We may rewrite (7) using the subscripts used in Fig. 1 as

$$L_{D_1}\left(x_1|\mathbf{y}\right) = L_{A_1}(x_1, k) + \underbrace{\ln \frac{\sum_{\mathbf{x}_1 \in \mathbb{X}_{k,+1}} p\left(\mathbf{y}|\mathbf{x}_1\right) \cdot \exp\left(\frac{1}{2}\mathbf{x}_{1,[k]}^T \cdot \mathbf{L}_{A_1,[k]}\right)}{\sum_{\mathbf{x}_1 \in \mathbb{X}_{k,-1}} p\left(\mathbf{y}|\mathbf{x}_1\right) \cdot \exp\left(\frac{1}{2}\mathbf{x}_{1,[k]}^T \cdot \mathbf{L}_{A_1,[k]}\right)}}_{L_{E_1}(x_{1,k}|\mathbf{y})}. \tag{8}$$

Equation (8) applies to the channel model (1), where $\mathbf{x}_1$ represents the coded bits to be transmitted and $\mathbf{y}$ is the vector measurement obtained at the receiver, but we may also apply (7) to the channel (error-correcting) code. Equation (7) then becomes

the *a posteriori* L-value obtained from APP decoding of the outer channel code. Thus, the channel decoder processing can also be decomposed into *a priori* and extrinsic components. We omit a detailed description of this standard interpretation and simply give the resulting equation for the channel code

$$L_{D_2}(x_{2,k}|\mathbf{L}_{A_2}) = L_{A_2}(x_{2,k}) + \underbrace{\ln \frac{\sum_{\mathbf{x}_2 \in \mathbb{X}_{k,+1}} \exp\left(\frac{1}{2}\mathbf{x}_{2,[k]}^T \cdot \mathbf{L}_{A_2,[k]}\right)}{\sum_{\mathbf{x}_2 \in \mathbb{X}_{k,-1}} \exp\left(\frac{1}{2}\mathbf{x}_{2,[k]}^T \cdot \mathbf{L}_{A_2,[k]}\right)}}_{L_{E_2}(x_{2,k}|\mathbf{L}_{A_2,[k]})}. \quad (9)$$

In this equation, the raw data bits are denoted $\mathbf{x}_2$ (see Fig. 1 variables with subscript "2"), and $\mathbb{X}_{k,+1}$ is now the set of vectors whose length is the same as the interleaver, with $x_k = +1$.

B. Likelihood Function for APP Detection

An essential part of computing the L-value (7) for the detector is computing the likelihood function $p(\mathbf{y}|\mathbf{x})$. This is easily found from (1)

$$p(\mathbf{y}|\mathbf{s} = \text{map}(\mathbf{x})) = \frac{\exp\left[-\frac{1}{2\sigma^2} \cdot \|\mathbf{y} - \mathbf{H} \cdot \mathbf{s}\|^2\right]}{(2\pi\sigma^2)^N}. \quad (10)$$

For the L-value calculation, only the term in the exponent is relevant, and the constant factor outside the exponent can be omitted.

C. Some Standard Simplifications for Digital Signal Processor (DSP) Implementation

To evaluate the numerator and denominator in the log-likelihood ratio computation in (7), it is sometimes advantageous to use the "Jacobian logarithm."

$$\begin{aligned}\text{jacln}(\mathrm{a}_1, \mathrm{a}_2) &:= \ln(e^{a_1} + e^{a_2}) \\ &= \max(a_1, a_2) + \underbrace{\ln\left(1 + e^{-|a_1 - a_2|}\right)}_{r(|a_1 - a_2|)} \quad (11)\end{aligned}$$

where $r(\cdot)$ can be viewed as a "refinement" of the coarse approximation $\max(a_1, a_2)$. On a DSP with no exponential or logarithm function, a Jac-log approximation can be obtained by storing $r(\cdot)$ in a lookup table [28]. To compute the Jac-log approximation to $\ln \sum_{i=1}^{N_i} e^{a_i}$ for $N_i > 2$, we can use the recursive calculation:

1) initialize: $v = -\infty$
2) compute: for $i = 1$ to N_i do $v := \text{jacln}(\mathrm{v}, \mathrm{a_i})$.

Further simplications are possible by using the Max-log approximation, which omits $r(\cdot)$ altogether. Simulations in [28] and [29] show that the performance degradation over the Jac-log approximation is often very small. With the Max-log approximation, the extrinsic L-value of (7) becomes

$$\begin{aligned}L_E(x_k|\mathbf{y}) \approx \frac{1}{2} &\max_{\mathbf{x} \in \mathbb{X}_{k,+1}} \left\{-\frac{1}{\sigma^2}\|\mathbf{y} - \mathbf{H} \cdot \mathbf{s}\|^2 + \mathbf{x}_{[k]}^T \cdot \mathbf{L}_{A,[k]}\right\} \\ -\frac{1}{2} &\max_{\mathbf{x} \in \mathbb{X}_{k,-1}} \left\{\frac{1}{\sigma^2}\|\mathbf{y} - \mathbf{H} \cdot \mathbf{s}\|^2 + \mathbf{x}_{[k]}^T \cdot \mathbf{L}_{A,[k]}\right\} \quad (12)\end{aligned}$$

where $\mathbf{s} = \text{map}(\mathbf{x})$.

Unfortunately, even with these simplifications, computing $L_E(x_k|\mathbf{y})$ is exponential in the length of the bit vector $\mathbf{x}$ or the number of symbols in the constellation $\mathcal{C}$. To find the maximizing hypotheses in (12) for each x_k, there are $2^{M \cdot M_c - 1}$ hypotheses to search over in each of the two terms. For even a moderate block size M, or bits per symbol M_c, this complexity may be overwhelming. For example, if the model (1) is used with eight transmit and eight receive antennas and direct transmission of a 16-QAM constellation, then $M \cdot M_c - 1 = 31$ and $2^{M \cdot M_c - 1} \approx 2 \times 10^9$. In the next section, we therefore concentrate on finding a method to approximate (12) that avoids this exhaustive search.

III. MIMO Detection Using the Sphere Decoder

A simple way to approximate (12) is to exclude from our search $\mathbf{s}$ for which

$$\|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 \quad (13)$$

is large and include only the hypotheses for which (13) is small; it turns out that, in practice, there are generally only a handful of hypotheses for which (13) is small. In this handful, which we call our *candidate list*, we can search for the hypotheses that maximize the two terms in (12). Searching the candidate list generally provides a good approximation of (12). In this section, we describe the application of a list sphere decoding (LSD) algorithm to rapidly find the candidate list. In the process, we show how the sphere decoder, originally designed for real constellations, may be modified to handle complex constellations.

We first give an overview of the sphere decoder for real constellations and channels. The sphere decoder (or sphere detector, as we may also call it in the context of MIMO detection) solves

$$\min_{\mathbf{s} \in \Lambda} (\mathbf{s} - \hat{\mathbf{s}})^T \mathbf{H}^T \mathbf{H} (\mathbf{s} - \hat{\mathbf{s}}) \quad (14)$$

where $\hat{\mathbf{s}}$ is the center of our search sphere, and Λ is the lattice defined by having each entry of the M-dimensional vector $\mathbf{s}$ be taken from a constellation of 2^{M_c} consecutive integers. We observe that

$$\begin{aligned}\|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 &= (\mathbf{s} - \hat{\mathbf{s}})^T \mathbf{H}^T \mathbf{H} (\mathbf{s} - \hat{\mathbf{s}}) \\ &\quad + \mathbf{y}^T \left(\mathbf{I} - \mathbf{H}(\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\right)\mathbf{y} \quad (15)\end{aligned}$$

where $\hat{\mathbf{s}} = (\mathbf{H}^T\mathbf{H})^{-1}\mathbf{H}^T\mathbf{y}$ is the unconstrained ML estimate of $\mathbf{s}$. The true (constrained) ML estimate is, therefore

$$\hat{\mathbf{s}}_{\text{ml}} = \arg\min_{\mathbf{s} \in \Lambda} \|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 = \arg\min_{\mathbf{s} \in \Lambda} (\mathbf{s} - \hat{\mathbf{s}})^T \mathbf{H}^T \mathbf{H} (\mathbf{s} - \hat{\mathbf{s}}). \quad (16)$$

The sphere decoder may thus be used to find $\hat{\mathbf{s}}_{\text{ml}}$.

Solving (14) is generally difficult unless $\mathbf{H}$ has orthogonal columns, in which case, the M-dimensional search becomes M simple one-dimensional searches. Otherwise, an exhaustive search needs to examine $2^{M \cdot M_c}$ different hypotheses. The sphere decoder avoids an exhaustive search by examining only those points that lie inside a sphere

$$(\mathbf{s} - \hat{\mathbf{s}})^T \mathbf{H}^T \mathbf{H} (\mathbf{s} - \hat{\mathbf{s}}) \le r^2 \quad (17)$$

with the given radius r large enough to contain the solution. The algorithm is described originally in [30] and refined in [8], and

has its origins in finding the shortest vector in a lattice. Its application as a decoder for fading channels is described in [31], and as an ML decoder for multiple antenna channels in [5], [9], [32], and [33]. As we show below, sphere decoding uses the same Cholesky factorization of the channel matrix as in V-BLAST nulling/cancelling algorithm described in [10], but it makes a joint decision on the symbols.

We assume, for the moment, that $r \geq 0$ has been chosen so that the sphere (17) contains the solution to (14) and possibly some additional points of the lattice. Let $\mathbf{U}$ be an upper triangular $M \times M$ matrix chosen such that $\mathbf{U}^T\mathbf{U} = \mathbf{H}^T\mathbf{H}$ (using, for example, Cholesky factorization). Let the entries of $\mathbf{U}$ be denoted u_{ij}, $i \leq j = 1, \ldots, M$, and assume, without loss of generality, that $u_{ii} > 0$. Then (17) may be written

$$(\mathbf{s}-\hat{\mathbf{s}})^T\mathbf{U}^T\mathbf{U}(\mathbf{s}-\hat{\mathbf{s}}) = \sum_{i=1}^{M} u_{ii}^2 \left[s_i - \hat{s}_i + \sum_{j=i+1}^{M} \frac{u_{ij}}{u_{ii}}(s_j - \hat{s}_j) \right]^2 \leq r^2. \quad (18)$$

Each term in the sum over i is nonnegative. The sphere decoder establishes bounds on $s_1, \ldots, s_M$ by examining these terms in subsets.

Starting with $i = M$, and throwing out the terms $i = 1, \ldots, M-1$, we obtain from (18)

$$u_{MM}^2(s_M - \hat{s}_M)^2 \leq r^2$$

or

$$\left\lceil \hat{s}_M - \frac{r}{u_{MM}} \right\rceil \leq s_M \leq \left\lfloor \hat{s}_M + \frac{r}{u_{MM}} \right\rfloor. \quad (19)$$

(The function $\lceil \cdot \rceil$ finds the smallest integer greater than or equal to its argument, and $\lfloor \cdot \rfloor$ finds the largest integer less than or equal to its argument; these functions are used because the constellation is assumed to be set of consecutive integers.) After computing the lower and upper bounds in (19), the sphere decoder chooses a candidate value for s_M and computes the implications of this choice on s_{M-1}. (Contrast this with the V-BLAST nulling/cancelling algorithm, which makes a decision on s_M at this point.) To find the influence of the choice of s_M on s_{M-1}, the sphere decoder looks at the two terms $i = M-1$ in (18), throws out the remaining terms, and obtains the inequality

$$u_{M-1,M-1}^2 \left[s_{M-1} - \hat{s}_{M-1} + \frac{u_{M-1,M}}{u_{MM}}(s_M - \hat{s}_M) \right]^2 + u_{MM}^2(s_M - \hat{s}_M)^2 \leq r^2$$

which yields the upper bound

$$s_{M-1} \leq \left\lfloor \hat{s}_{M-1} + \frac{\sqrt{r^2 - u_{MM}^2(s_M - \hat{s}_M)^2}}{u_{M-1,M-1}} - \frac{u_{M-1,M}}{u_{MM}}(s_M - \hat{s}_M) \right\rfloor \quad (20)$$

and a corresponding lower bound. The sphere decoder now chooses a candidate for s_{M-1} within the range given by the upper and lower bounds, and proceeds to s_{M-2}, and so on.

Eventually, one of two things happens: 1) the decoder reaches s_1 and chooses a value within the computed range; or 2) the decoder finds that no point in the constellation falls within the upper and lower bounds obtained for some s_m. In the first case, the sphere decoder has a candidate solution for the entire vector $\mathbf{s}$, computes its radius (which cannot exceed r), and starts the search process over, using this new smaller radius to find any better candidates. In the second case, the decoder must have made at least one bad candidate choice for $s_{m+1}, \ldots, s_M$. The decoder revises the choice for s_{m+1} (which immediately preceded the attempt for s_m) by finding another candidate value within its range, and proceeds again to try s_m. If no more candidates are available at s_{m+1}, the decoder backtracks to s_{m+2}, and so on.

The performance of the algorithm is closely tied to the choice of the initial radius r. The radius should be chosen large enough so that the sphere contains the solution to (14). However, the larger r is chosen, the longer the search takes. If r is chosen too small, the algorithm could fail to find any point inside the sphere, requiring that r be increased. For good choices of r (we have more to say about how to choose r later), the algorithm appears to be roughly cubic in M for the values of M that we consider [33], [34]. This is a vast improvement over an exhaustive search, which is exponential in M.

A. Complex Sphere Decoder

The sphere-decoding algorithm described above applies to a real system of equations when $\mathbf{s}$ is chosen from a real lattice. Therefore, we may apply the algorithm to the complex system (1) only when the real and imaginary components of $\mathbf{y}$, $\mathbf{H}$, and $\mathbf{s}$ can be decoupled to create a system of real equations with twice the dimension of the original system. This decoupling is possible, for example, when the entries of $\mathbf{s}$ are chosen from a QAM constellation. It is not generally possible, however, for PSK or other complex constellations. Fortunately, as we show, the sphere decoder may be modified to handle complex constellations as well.

We wish to solve

$$\min_{\mathbf{s}\in\Lambda} (\mathbf{s}-\hat{\mathbf{s}})^*\mathbf{H}^*\mathbf{H}(\mathbf{s}-\hat{\mathbf{s}}) \quad (21)$$

where $\hat{\mathbf{s}}$ and $\mathbf{H}$ are complex, $(\cdot)^*$ denotes conjugate-transpose, and Λ is a complex lattice in the sense that each coordinate of $\mathbf{s}$ is chosen from a complex constellation. The complex sphere search is then

$$(\mathbf{s}-\hat{\mathbf{s}})^*\mathbf{H}^*\mathbf{H}(\mathbf{s}-\hat{\mathbf{s}}) \leq r^2. \quad (22)$$

We use the Cholesky factorization to find an upper triangular $\mathbf{U}$ with u_{ii} real and positive such that $\mathbf{U}^*\mathbf{U} = \mathbf{H}^*\mathbf{H}$. Then (22) may be written

$$(\mathbf{s}-\hat{\mathbf{s}})^*\mathbf{U}^*\mathbf{U}(\mathbf{s}-\hat{\mathbf{s}}) \times \sum_{i=1}^{M} u_{ii}^2 \sum_{i=1}^{M} u_{ii}^2 \left| s_i - \hat{s}_i + \sum_{j=i+1}^{M} \frac{u_{ij}}{u_{ii}}(s_j - \hat{s}_j) \right|^2 \leq r^2. \quad (23)$$

As in the real case, these terms are nonnegative and are examined in subsets to find bounds on $s_1, \ldots, s_M$.

The term $i = M$ yields

$$|s_M - \hat{s}_M| \leq \frac{r}{u_{MM}}. \quad (24)$$

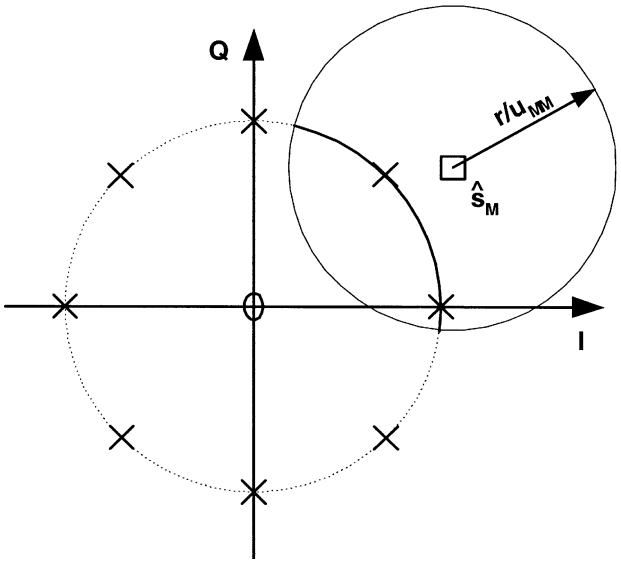


Fig. 2. Intersection of search disk and 8-PSK constellation yields a subset of the PSK constellation contained within the arc obtained by the intersection of the two circle boundaries.

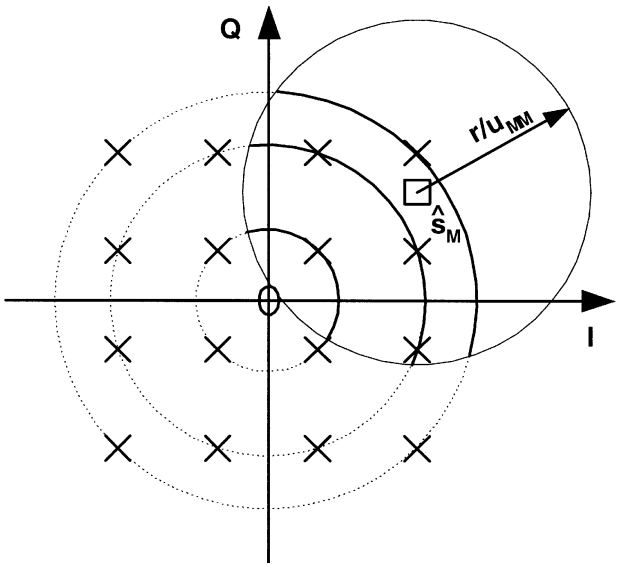


Fig. 3. Intersection of search disk and a 16-QAM constellation can be obtained by considering the QAM constellation as three concentric rings of points.

This inequality limits the search to points of the constellation contained in a complex disk of radius r/u_{MM} centered at $\hat{s}_M$. These points are easily found when the constellation forms a complex circle (as in PSK). Fig. 2 shows graphically that the intersection of a disk and a circle is generally an arc. The angular sweep of this arc can be obtained analytically by solving for the overlap of the search disk and the constellation circle.

Let $s_M = r_c e^{i\theta_M}$, where $\theta_M \in \{0, 2\pi/2^{M_c}, \ldots, 2\pi(2^{M_c} - 1)/2^{M_c}\}$ are the 2^{M_c} angles of the 2^{M_c}-PSK constellation, and $r_c > 0$ is the radius of the circle formed by the PSK constellation. Denote $\hat{s}_M = \hat{r}_c e^{i\hat{\theta}_M}$ where $\hat{r}_c > 0$. Then (24) becomes

$$|s_M - \hat{s}_M|^2 = r_c^2 + \hat{r}_c^2 - 2r_c\hat{r}_c\cos(\theta_M - \hat{\theta}_M) \leq \frac{r^2}{u_{MM}^2}$$

which yields

$$\cos(\theta_M - \hat{\theta}_M) \geq \frac{1}{2r_c\hat{r}_c}\left[r_c^2 + \hat{r}_c^2 - \frac{r^2}{u_{MM}^2}\right] =: \eta. \quad (25)$$

If $\eta > 1$, then the search disk does not contain any point of the PSK constellation. If $\eta < -1$, then the search disk includes the entire constellation. For $-1 \leq \eta \leq 1$, the arc is described by

$$|\theta_M - \hat{\theta}_M| \leq \cos^{-1}\eta.$$

(We assume that $0 \leq \cos^{-1}(\cdot) \leq \pi$.) Alternatively, the range of allowable constellation points is given by

$$\left\lceil \frac{2^{M_c}}{2\pi}(\hat{\theta}_M - \cos^{-1}\eta) \right\rceil \leq \frac{2^{M_c}}{2\pi}\theta_M \leq \left\lfloor \frac{2^{M_c}}{2\pi}(\hat{\theta}_M + \cos^{-1}\eta) \right\rfloor. \quad (26)$$

We may now choose a candidate s_M by letting θ_M be a point within the range (26). The remainder of the algorithm proceeds as in the real case. The sphere decoder establishes bounds on θ_{M-1} by finding its allowable arc using the two terms $i = M - 1, M$ in (23), chooses a candidate for θ_{M-1}, and so on.

We note that complex constellations other than PSK may also be efficiently decoded using the complex sphere decoder. For efficient decoding, the decoder must be able to quickly recognize which constellation points are contained within the search disk for every s_m. Because identifying constellation points within the search disk is simple when the points are arranged in a circle, it follows that constellation points that are arranged in concentric circles can also easily be identified. For example, Fig. 3 shows how the 16-QAM constellation can be expressed as an arrangement of points in three concentric circles. Solving for the points within the search disk simply requires solving the inequality (25) for three different values of r_c. While 16-QAM can also be handled by the real sphere decoder by decoupling the real and imaginary equations to form a system of real equations that is twice as large (see, for example, [9]), the complex sphere decoder has a speed advantage because it does not double the effective dimension of the search lattice.

B. LSD

The previous section shows that the sphere decoder solves (14), (16), or (21). However, we are interested in computing (12). Finding the ML estimate $\hat{\mathbf{s}}_{\mathrm{ml}}$ does not necessarily help, because, although it is the estimate that makes (13) smallest, it is not necessarily the estimate that maximizes the two terms in (12).

However, a simple modification to the sphere decoder helps us to compute (12). The sphere decoder is modified to generate a list $\mathcal{L}$ of the N_{cand} points $\mathbf{s}$ that make (13) smallest. This list, by definition, must include $\hat{\mathbf{s}}_{\mathrm{ml}}$, but its size N_{cand} obeys $2^{M\cdot M_c} > N_{\mathrm{cand}} \geq 1$, and is predetermined sufficiently large so that $\mathcal{L}$ also contains the maximizer of (12) with high probability. To create $\mathcal{L}$, the sphere decoder needs to be modified in two ways. Every time it finds a point inside the initial radius r it: 1) does not decrease r to correspond to the radius of this new point; 2) adds this point to $\mathcal{L}$ if the list is not already full; or if $\mathcal{L}$ is full, it compares this point with the point in $\mathcal{L}$ with the largest radius and replaces this point if the new point has smaller radius.

Hence, $\mathcal{L}$ contains the ML estimate and $N_{\text{cand}} - 1$ neighbors for which (13) is smallest. The "soft" information about any given bit x_k is essentially contained in $\mathcal{L}$ because if there are many entries in $\mathcal{L}$ with $x_k = 1$, then it can be concluded that the likely value for x_k is indeed one, whereas, if there are few entries in $\mathcal{L}$ with $x_k = 1$, then the likely value is minus one. If there are no entries in $\mathcal{L}$ with a prescribed bit value, then we can set its corresponding L-value $L_E(x_k|\mathbf{y})$ to an extreme value whose size can be made an increasing function of the radius r. A larger r generally allows for larger N_{cand}, which makes the list more reliable. In practice, a simple clipping of L-values (in our case to $\pm$ 8) also yields good results.

Equation (12) is approximated using $\mathcal{L}$ as

$$L_E(x_k|\mathbf{y}) \approx \frac{1}{2} \max_{\mathbf{x} \in \mathcal{L} \cap \mathbb{X}_{k,+1}} \left\{ \frac{1}{\sigma^2} \|\mathbf{y} - \mathbf{H} \cdot \mathbf{s}\|^2 + \mathbf{x}_{[k]}^T \cdot \mathbf{L}_{A,[k]} \right\} - \frac{1}{2} \max_{\mathbf{x} \in \mathcal{L} \cap \mathbf{X}_{k,-1}} \left\{ -\frac{1}{\sigma^2} \|\mathbf{y} - \mathbf{H} \cdot \mathbf{s}\|^2 + \mathbf{x}_{[k]}^T \cdot \mathbf{L}_{A,[k]} \right\} \tag{27}$$

where $\mathbf{s} = \text{map}(\mathbf{x})$. The approximation (27) becomes an equality only when $N_{\text{cand}} = 2^{M \cdot M_c}$, but the size of N_{cand} needed for good performance is usually far less. Therefore, in practice, computing (27) is much faster than computing (12).

There is also a tradeoff between the accuracy of (27) and the speed of the LSD. Finding N_{cand} points is generally slower than just finding $\hat{\mathbf{s}}_{\text{ml}}$ (which corresponds to $N_{\text{cand}} = 1$), because the search radius always stays at r and does not decrease with every point that is found. But as noted in [34], the added complexity of holding the radius fixed is small. Generally, we would like to make N_{cand} as large as possible, while still having acceptable complexity. We have more to say about how to choose r and N_{cand} in Sections III-C and IV-B.

We observe that for the LSD soft value calculation (LSD/APP), the candidate list $\mathcal{L}$ per block channel symbol $\mathbf{y}$ can be computed just once and stored in memory, no matter how many iterations are used between the detector and decoder. With every iteration, the updated *a priori* knowledge $\mathbf{L}_{A_1}$ from the outer decoder is used for the metric calculation of (27), searching the same $\mathcal{L}$ to find the maximizing hypotheses. If buffer sizes are severely limited, the sphere detector can be rerun at every iteration.

C. Note on Choosing the Sphere Radius

The list size N_{cand} measures how well (27) approximates (12). Suppose that the desired degree of approximation is obtained for some N_{cand}, and we need to choose r to obtain $\mathcal{L}$ with N_{cand} candidates, on average. Clearly, if r is chosen too small, only a few points will be found inside the sphere, no matter how large N_{cand} is. On the other hand, choosing r too large slows the LSD down because it searches through many candidates before it finds the best N_{cand} of them.

To obtain a rough idea of a typical value of r, we note that for the true $\mathbf{s}$

$$\|\mathbf{y} - \mathbf{H}\mathbf{s}\|^2 = \|\mathbf{n}\|^2 \sim \sigma^2 \cdot \chi_{2N}^2$$

where χ_{2N}^2 is a chi-square random variable with $2N$ degrees of freedom. The expected value of this random variable is $\sigma^2 \text{E}\chi_{2N}^2 = 2\sigma^2 N$. Therefore, from (15), one possible choice of radius is

$$r^2 = 2\sigma^2 KN - \mathbf{y}^* \left(\mathbf{I} - \mathbf{H}(\mathbf{H}^*\mathbf{H})^{-1}\mathbf{H}^* \right) \mathbf{y} \tag{28}$$

where $K \geq 1$ is chosen so that we are reasonably sure, as measured by a confidence interval for the χ_{2N}^2 random variable, that we will capture the true $\mathbf{s}$. Depending on the size of N_{cand}, we may increase this radius by some multiple of the covering radius (or its approximation) of the lattice [35]. We have found that simple trial and error provides a satisfactory value for r without difficulty.

IV. Performance Examples of Using Multiple Antennas at High Data Rates

In this section, we demonstrate the near-capacity performance of the iterative LSD/APP detector/decoder. We focus on direct transmission, with an equal number of transmit and receive antennas ($M \times M$ system). We first compute some channel capacities and mutual information of constrained constellations.

A. Capacity of the Ergodic MIMO Channel

With direct transmission, and assuming that the entries of the complex matrix $\mathbf{H}$ are independent complex Gaussian random variables (Rayleigh amplitude, uniform phase) with unit variance, the channel capacity of the model (1) is [2]

$$C = \text{E} \log \det \left(\mathbf{I}_N + \frac{\rho}{M} \mathbf{H}\mathbf{H}^* \right) \tag{29}$$

where $\rho = E_s/2\sigma^2$ is the SNR as physically measured at each receive antenna, and the expectation is over the entries of $\mathbf{H}$. We use the convention that $N_0/2 = \sigma^2$ (double-sided noise power spectral density) to define the SNR measure $\rho = E_s/N_0$. For (29) to be meaningful, the channel should be ergodic in the sense that the statistical nature of $\mathbf{H}$ is observed as the channel is used. We assume that the channel is perfectly tracked by the receiver and interleaved so that successive channel uses see independent samples of $\mathbf{H}$.

To achieve any point on the capacity curve, a symbol constellation with a Gaussian distribution is generally needed. However, to be practical, we restrict our attention to PSK or QAM constellations. To see the effect of a PSK or QAM constellation on the maximum achievable rate in the model (1), we compute the mutual information between the output $\mathbf{y}$ and input $\mathbf{s}$, assuming that $s_1, \ldots, s_M$ are chosen independently and equally likely from the constellation. The mutual information is computed using the formula

$$I(\mathbf{s}; \mathbf{y}) = H(\mathbf{y}) - H(\mathbf{y}|\mathbf{s}) \tag{30}$$

where $H(\cdot) = -\text{E} \log p(\cdot)$ is the entropy function. Standard arguments show that $H(\mathbf{y}|\mathbf{s}) = N \log 2\pi\sigma^2 e$ for the Gaussian channel (1) for any symbol constellation. The term $H(\mathbf{y})$ in (30) is more difficult to compute and generally has no closed-form expression. For our purposes, it suffices to note that the expectation in $H(\mathbf{y}) = -\text{E} \log p(\mathbf{y})$ is over the three sources of randomness in the choices of $\mathbf{s}$, $\mathbf{H}$, and $\mathbf{n}$. This expectation is easily approximated numerically using sampling (Monte-Carlo)

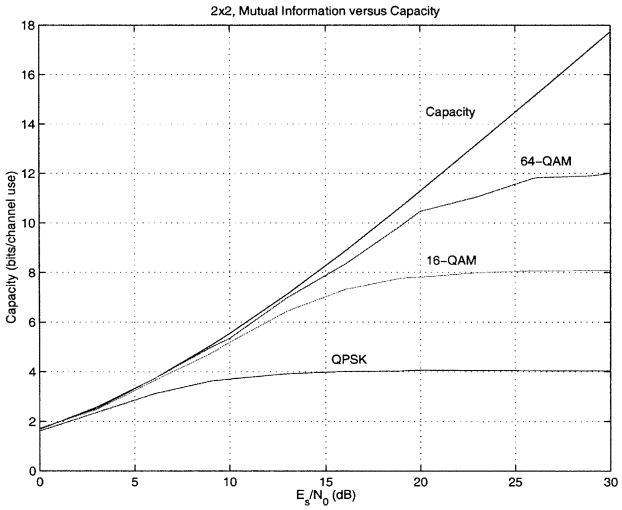


Fig. 4. Capacity (29) and mutual information versus E_s/N_0 for two-transmit/two-receive system in Rayleigh ergodic flat fading. The uppermost curve is the capacity, and the remaining curves represent the maximum data rates achievable by various symbol constellations.

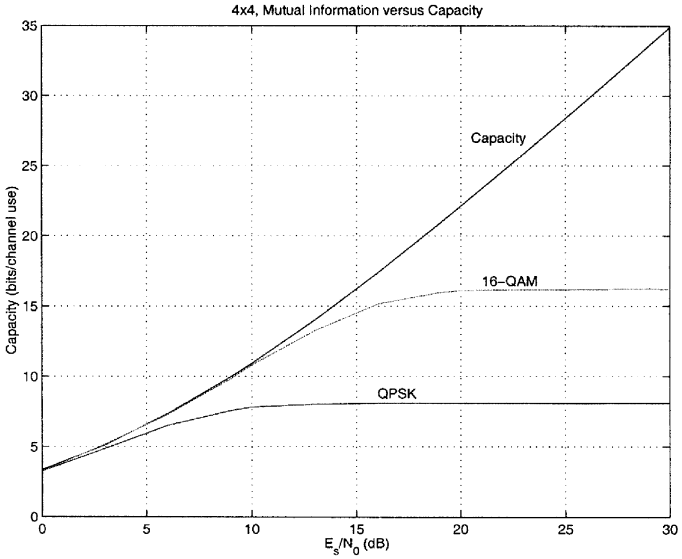


Fig. 5. Capacity and mutual information versus E_s/N_0 for four transmit/four receive system in Rayleigh ergodic flat fading. The uppermost curve is the capacity, and the remaining curves represent the maximum data rates achievable by various symbol constellations.

methods. When MM_c is not too large, we may compute the expectation over $\mathbf{s}$ without approximation using a sum

$$H(\mathbf{y}) = -\mathrm{E}\log\left(\frac{1}{2^{MM_c}(2\pi\sigma^2)^N}\sum_{\mathbf{s}}\exp\left[-\frac{1}{2\sigma^2}\|\mathbf{y}-\mathbf{Hs}\|^2\right]\right)$$

where the sum allows the entries of the vector $\mathbf{s}$ to run over all 2^{MM_c} possible combinations of the constellation values.

The result of computing (30) for various constellations for two transmit and two receive antennas appears in Fig. 4. The capacity is represented by the uppermost curve [Gaussian input, (29)]. The remaining curves can be thought of as generalizations of constellation-constrained rate curves commonly available for single-antenna systems [36].

We would like to achieve a point on the capacity curve (29) at some rate C. To make our transmitted data rate C, we must choose the vector constellation size 2^{MM_c} and channel code rate R such that $RMM_c = C$ (channel coding theorem [37]: error-free transmission possible for $RMM_c \leq C$). We also must consult Fig. 4 to ensure that the mutual information attained by the constellation is close to the capacity curve at C. For example, suppose it is desired to achieve rate $C = 6$ (at $E_s/N_0 \approx 11$dB). One possibility is to choose a 64-QAM symbol constellation, which has an uncoded maximum data rate of 12 bits/channel use, and a channel code rate $R = 1/2$. Fig. 4 confirms that the mutual information of a 64-QAM constellation at six bits/channel use is very close to capacity.

B. Discussion of Simulation Results

We first provide a definition of E_b/N_0 that is used in some of our performance curves. By our definition of E_s in Section I-A, the (average) signal energy per transmitted PSK or QAM constellation symbol s_m is E_s/M. Because the fading coefficients are independent with unit variance, the (average) signal energy per receive antenna is E_s. Hence, the N receive antennas collect total power NE_s, carrying MM_c coded bits, or RMM_c information bits. We therefore define the signal energy per transmitted information bit at the receiver to be $E_b = (N/RMM_c)\cdot E_s$, or, expressed in terms of logarithmic SNR measures

$$\frac{E_b}{N_0}\Big|_{\mathrm{dB}} = \frac{E_s}{N_0}\Big|_{\mathrm{dB}} + 10\log_{10}\frac{N}{RMM_c}. \tag{31}$$

Since system capacity grows linearly with the number of antennas when $M = N$, capacity for a given PSK or QAM constellation is attained at (approximately) the same E_b/N_0 as defined in (31), independently of the number of antennas.

In our examples, we use the same number of transmit and receive antennas, so $M = N$. We use direct transmission, with no special space–time mapping. The sphere detector operates very rapidly because it has as many equations as unknowns [5], [33].

For the simulations, a rate $R = 1/2$ parallel concatenated (turbo) code [7] of memory 2 with (recursive) feedback polynomial $G_r(D) = 1 + D + D^2$ and feedforward polynomial $G(D) = 1 + D^2$ is used. The interleaver size of the turbo code is 9216 information bits. As can be seen from Figs. 4 and 5, for a code rate of $R = 1/2$ the contrained input capacity (QPSK/16-QAM/64-QAM) is generally very close to the continuous (Gaussian) input capacity.

Fig. 6 shows the performance of iterative detection and decoding for different modulation schemes (gray mapping) up to $M = N = 8$ transmit/receive antennas. For $MM_c \leq 8$ bits per vector channel symbol, full APP detection was applied, which searches over 2^{MM_c} hypotheses per detected bit. For $MM_c > 8$, sphere detection with candidate lists of maximal lengths $N_{\mathrm{cand}} = 512 (8 < MM_c \leq 32)$ and $N_{\mathrm{cand}} = 1024$ $(32 < MM_c \leq 48)$ were used. The respective capacity limits (dashed lines) indicate how closely the MIMO capacity is approached. The transmission is organized in blocks of length 9216 information bits. For each block, we performed four iterations over the MIMO detection loop, and eight iterations within the turbo decoder. These choices for the number of iterations

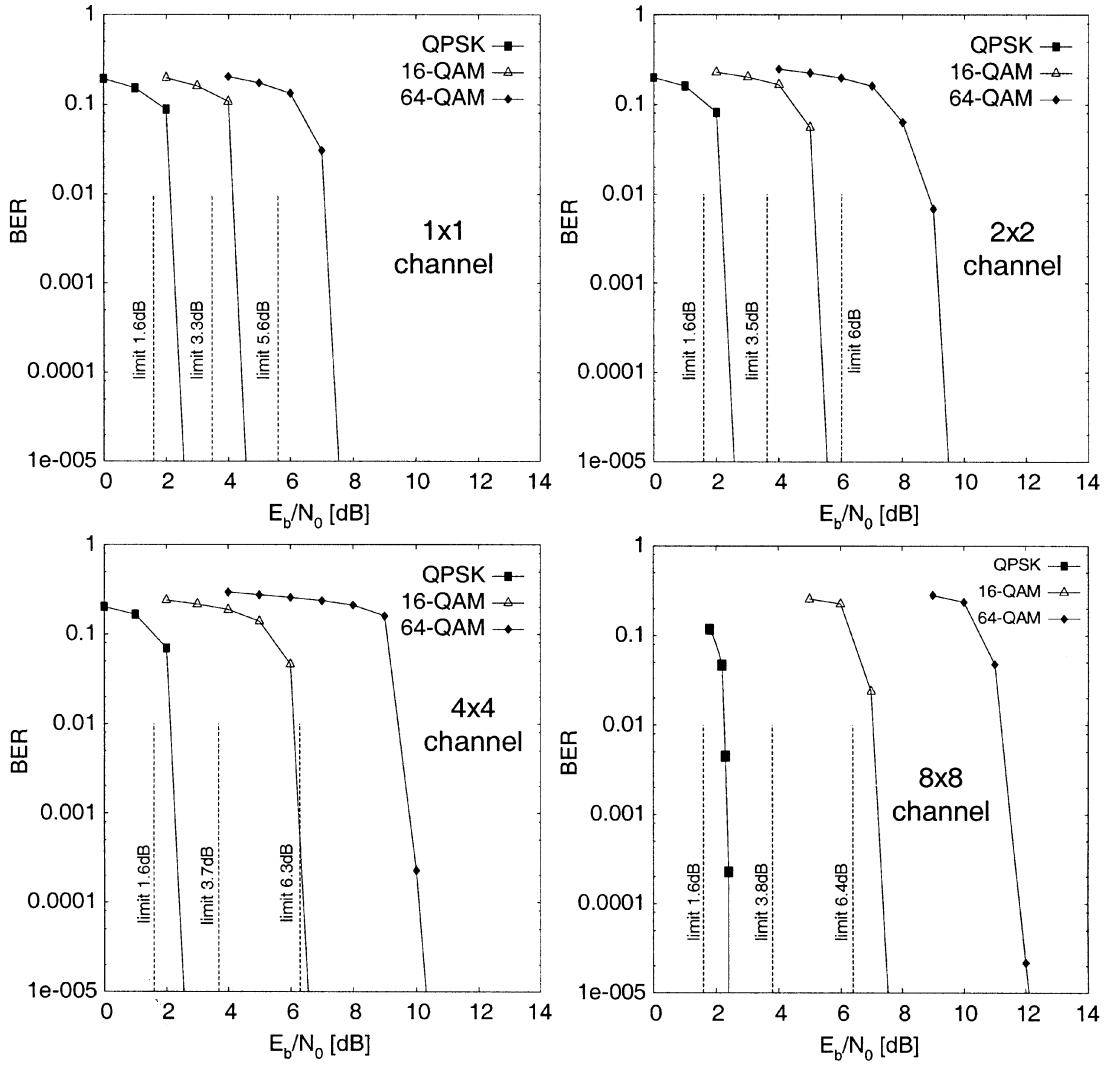


Fig. 6. BER curves of QPSK, 16-QAM and 64-QAM over 1×1, 2×2, 4×4, and 8×8 MIMO channels; block size 9216 information bits, code rate $R = 1/2$, memory 2 turbo code; APP and LSD/APP detection. The transmitted information rate (bits/channel use) is RMM_c; for example, the 4×4 case with 64-QAM transmits 12 bits/channel use and is roughly 4 dB from capacity.

were found to yield good overall BER performance. Increasing either number of iterations past these suggested numbers increases the decoding complexity without materially improving performance.

The BER curves in Fig. 6 for the 1×1 case (one transmit, one receive) are given as references for turbo code performance on a Rayleigh channel. As a general rule, the more bits that are involved in the detection process $(M \cdot M_c)$, the more candidates should be kept for computing the soft output values. Hence, the process of limiting the candidate list to a reasonable number is especially restrictive for the 64-QAM, 8×8 case (lower right BER chart of Fig. 6), where $MM_c = 48$. Most of the gap of approximately 6 dB from the capacity limit is due to our setting $N_{\text{cand}} = 1024$, which is a tiny fraction of $2^{MM_c} \approx 2.81 \times 10^{14}$, the total number of hypotheses required for full APP detection. To show that even this small list of $N_{\text{cand}} = 1024$ candidates is very helpful, we note that reducing this list to $N_{\text{cand}} = 1$ (list contains ML estimate only) results in a BER curve with "turbo cliff" at about 17 dB (not shown), representing a loss of 5 dB over $N_{\text{cand}} = 1024$.

Fig. 7 compares the performance of iterative detection and decoding using a very simple outer convolutional code ($R = 1/2$, memory 2) with the turbo code. Although the final (after iterating) performance of the turbo code is better, the advantage is only approximately 4 dB. Interestingly, we see that the gains from the detector/decoder iterations are more pronounced with the convolutional code.

In Fig. 8, we can see how BER performance improves as N_{cand} is increased from 1 to 512. Since the size of N_{cand} is a measure of time spent in the list sphere decoder and time spent computing (27), this figure gives a measure of the performance/complexity tradeoff when using our proposed iterative LSD/APP decoder.

In Fig. 9, we apply a rate $R = 3/4$ memory 2 turbo code (punctured version of $R = 1/2$ code used in previous examples) to yield the huge spectral efficiency of $RMM_c = 36$ bits per channel use (64-QAM on 8×8 channel). In this case, the ML estimate by itself already performs quite well, but iterating with a candidate list of length $N_{\text{cand}} = 512$ gains another 3 dB and puts us less than 5 dB from capacity.

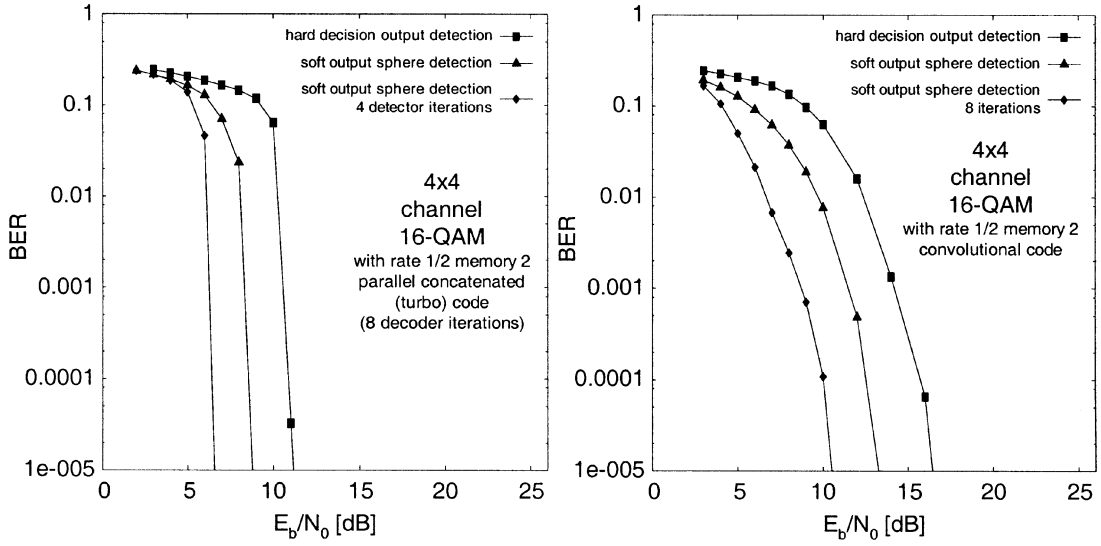


Fig. 7. BER curves for ML detection (hard decision output), LSD with no iterations (soft output detection, where the LSD output is used just once), and LSD with multiple iterations of 16-QAM over a 4×4 channel; comparison of outer memory 2 turbo code (left) and outer memory 2 convolutional code (right), both of code rate $R = 1/2$. The information rate is 8 bits/channel use, and capacity is at 3.7 dB.

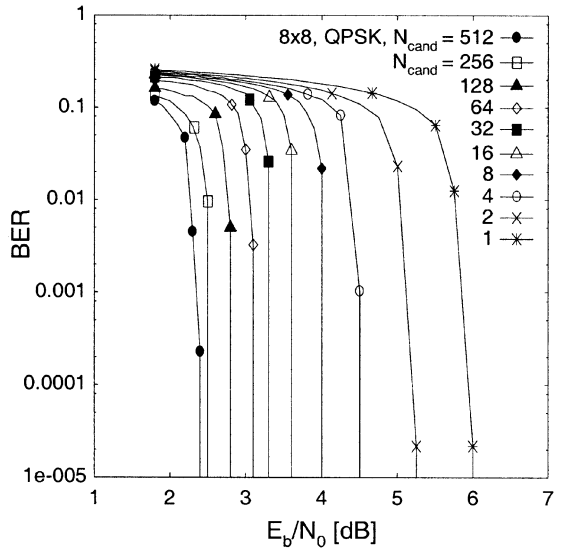


Fig. 8. BER curves as a function of SNR for various values of N_{cand} ranging from 1 to 512, for an 8×8 channel transmitting QPSK with $R = 1/2$ memory 2 turbo code. The information rate is 8 bits/channel use, and capacity is at 1.6 dB (see Fig. 6). As N_{cand} increases, performance improves, but the time needed to compute (27) also rises.

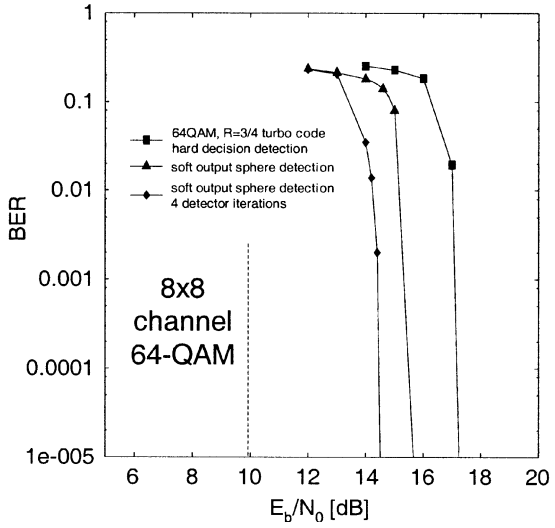


Fig. 9. BER curves for ML detection (hard decision output), LSD with no iterations (soft output detection, where the LSD output is used just once), and LSD with multiple iterations of 64-QAM over an 8×8 channel; memory 2 turbo code of rate $R = 3/4$. The information rate is 36 bits/channel use, and we are a little over 4 dB from capacity.

V. Conclusion

We have presented a computationally efficient method of achieving near-capacity on a multiantenna channel. The method iterates the channel decoder and an LSD that finds a set of candidates from which the posterior bit probabilities can be accurately computed.

Our approach scales easily with the symbol constellation size and number of antennas, but we have focused primarily on cases with equal number of transmit and receive antennas. We note that space–time code design notions such as "diversity" that are derived from pairwise probability of error criteria are not needed for our iterative method to achieve near-capacity. All that is needed, in principle, for any combination of transmit and receive antennas is a single channel code followed by a linear map of the coded data symbols to the transmit antennas. These coded data symbols are then interleaved and sent over the transmit antennas.

When there are more transmit than receive antennas, our experiments show (not reported here) that direct transmission still successfully achieves capacity, but sphere detection becomes more computationally burdensome. Alternatively, a mapping such as used in [5] can be used to ensure that the number of equations at the receiver is at least as large as the number of unknowns, without sacrificing channel capacity.

The size of the list N_{cand} in the modified sphere decoder closely determines the running time and closeness to capacity.

We have provided some guidelines for choosing N_{cand} as a function of SNR and number of antennas. Generally, the larger the information rate, the larger the list should be. In all cases, the complexity is reasonable and is not exponential in the rate or number of antennas, as optimal processing would be.

We have not yet examined the performance of our method on a static channel, where a comparison with outage capacity might be more appropriate than ergodic capacity.

Some possible ways that we have not considered to close the remaining gap to capacity include improving the turbo code and constellation shaping, especially at high rates.

REFERENCES

[1] 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.

[2] I. E. Telatar, "Capacity of multi-antenna Gaussian channels," *Eur. Trans. Telecommun.*, vol. 10, pp. 585–595, Nov. 1999.

[3] S. M. Alamouti, "A simple transmitter diversity scheme for wireless communications," *IEEE J. Select. Areas Commun.*, vol. 16, pp. 1451–1458, Oct. 1998.

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

[5] B. Hassibi and B. Hochwald, "High-rate codes that are linear in space and time," *IEEE Trans. Inform. Theory*, vol. 48, pp. 1804–1824, July 2002.

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

[7] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding: turbo-codes," in *Proc. Int. Conf. Communications*, May 1993, pp. 1064–1070.

[8] 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.

[9] M. O. Damen, A. Chkeif, and J.-C. Belfiore, "Lattice code decoder for space–time codes," *IEEE Commun. Lett.*, pp. 161–163, May 2000.

[10] G. D. Golden, G. J. Foschini, R. A. Valenzuela, and P. W. Wolniansky, "Detection algorithm and initial laboratory results using V-BLAST space–time communication architecture," *Electron. Lett.*, vol. 35, pp. 14–16, Jan. 1999.

[11] A. M. Tonello, "Space–time bit-interleaved coded modulation with an iterative decoding strategy," in *Proc. Vehicle Technology Conf.*, Sept. 2000, pp. 473–478.

[12] A. van Zelst, R. van Nee, and G. Awater, "Turbo-BLAST and its performance," in *Proc. Vehicle Technology Conf.*, May 2001, pp. 1282–1386.

[13] H.-J. Su and E. Geraniotis, "Space–time turbo codes with full antenna diversity," *IEEE Trans. Commun.*, vol. 49, pp. 47–57, Jan. 2001.

[14] C. Schlegel and A. Grant, "Concatenated space–time coding," in *Proc. PIMRC*, San Diego, CA, Sept. 2001, pp. 139–143.

[15] M. Sellathurai and S. Haykin, "Turbo-BLAST for high-speed wireless communications," in *Proc. WCNC*, Sept. 2000, pp. 315–320.

[16] S. L. Ariyavisitakul, "Turbo space–time processing to improve wireless channel capacity," *IEEE Trans. Commun.*, vol. 48, pp. 1347–1359, Aug. 2000.

[17] A. Stefanov and T. Duman, "Turbo-coded modulation for systems with transmit and receive antenna diversity over block fading channels: System model, decoding approaches, and practical considerations," *IEEE J. Select. Areas Commun.*, vol. 19, pp. 958–968, May 2001.

[18] J.-C. Guey, M. P. Fitz, M. R. Bell, and W.-Y. Kuo, "Signal design for transmitter diversity wireless communication systems over Rayleigh fading channels," in *Proc. IEEE VTC*, Atlanta, GA, 1996, pp. 136–140.

[19] A. R. Hammons, Jr. and H. El Gamal, "On the theory of space–time codes for PSK modulation," *IEEE Trans. Inform. Theory*, vol. 46, pp. 524–542, Mar. 2000.

[20] B. Hochwald and T. L. Marzetta, "Unitary space–time modulation for multiple-antenna communication in Rayleigh flat fading," *IEEE Trans. Inform. Theory*, vol. 46, pp. 543–564, Mar. 2000.

[21] E. Biglieri, G. Taricco, and E. Viterbo, "Bit-interleaved space–time codes for fading channels," in *Proc. Conf. Information Science and Systems (CISS)*, Princeton, NJ, Mar. 15–17, 2000, pp. WA4/1–WA4/6.

[22] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: performance analysis, design and iterative decoding," *IEEE Trans. Inform. Theory*, vol. 44, pp. 909–926, May 1998.

[23] J. Hagenauer, E. Offer, and L. Papke, "Iterative decoding of binary and block convolutional codes," *IEEE Trans. Inform. Theory*, vol. 42, pp. 429–445, Mar. 1996.

[24] J. Hagenauer, "The turbo principle: tutorial introduction and state of the art," in *Proc. 1st Int. Symp. Turbo Codes*, Sept. 1997, pp. 1–12.

[25] X. Li and J. A. Ritcey, "Bit-interleaved coded modulation with iterative decoding," in *Proc. Int. Conf. Communications*, June 1999, pp. 858–862.

[26] S. ten Brink, J. Speidel, and R. Yan, "Iterative demapping and decoding for multilevel modulation," in *Proc. GLOBECOM*, Nov. 1998, pp. 579–584.

[27] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284–287, Mar. 1974.

[28] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and suboptimal MAP decoding algorithms operating in the log domain," in *Proc. Int. Conf. Communications*, June 1995, pp. 1009–1013.

[29] J. Hagenauer, P. Robertson, and L. Papke, "Iterative ('turbo') decoding of systematic convolutional codes with the MAP and SOVA algorithms," in *Proc. ITG Symp. Source and Channel Coding*, 1994, pp. 21–29.

[30] M. Pohst, "On the computation of lattice vectors of minimal length, successive minima and reduced bases with applications," in *Proc. ACM SIGSAM*, 1981, pp. 37–44.

[31] E. Viterbo and J. Boutros, "A universal lattice code decoder for fading channels," *IEEE Trans. Inform. Theory*, vol. 45, pp. 1639–1642, July 1999.

[32] B. Hochwald and B. Hassibi, "Cayley differential unitary space–time codes," *IEEE Trans. Inform. Theory*, vol. 48, pp. 1485–1503, June 2002.

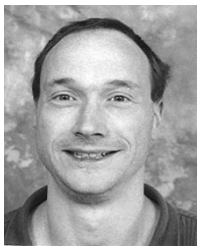
[33] M. O. Damen, K. Abed-Meraim, and M. S. Lemdani, "Further results on the sphere decoder," in *Proc. IEEE Int. Symp. Information Theory*, June 2001, p. 333.

[34] B. Hassibi, Private Communication, 2001.

[35] J. H. Conway and N. J. A. Sloane, *Sphere Packings, Lattices and Groups*. New York: Springer-Verlag, 1988.

[36] J. Wozencraft and I. Jacobs, *Principles of Communication Engineering*. New York: Wiley, 1965.

[37] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, pp. 379–423, Oct. 1948.



Bertrand Hochwald was born in New York, NY. He received the undergraduate degree from Swarthmore College, Swarthmore, PA, and the M.S. degree in electrical engineering from Duke University, Durham, NC. In 1989, he enrolled at Yale University, New Haven, CT, where he received the M.A. degree in statistics and the Ph.D. degree in electrical engineering.

From 1986 to 1989, he worked for the United States Department of Defense, Fort Meade, MD. In 1995–1996 he was a Research Associate and Visiting Assistant Professor at the Coordinated Science Laboratory, University of Illinois, Urbana-Champaign. He joined the Mathematics of Communications Research Department at Lucent Technologies Bell Laboratories, Murray Hill, NJ, in September, 1996, where he is now a Distinguished Member of the Technical Staff. He holds several patents in the field of multiantenna wireless communication.

Dr. Hochwald is the recipient of several achievement awards while with the Department of Defense and the Prize Teaching Fellowship at Yale.

Stephan ten Brink received the Dipl.-Ing. degree in electrical engineering and information technology from the University of Stuttgart, Stuttgart, Germany, in 1997.

From 1997 to 2000 he was a Research Assistant at the Institute of Telecommunications, Stuttgart, Germany, where he was working toward the doctoral degree. Since November 2000, he has been with the Wireless Research Lab, Bell Laboratories, Lucent Technologies, Holmdel, NJ. His research interests include error-correcting coding, channel estimation, and iterative detection and decoding for digital communication systems.