Chapter 10

A Digital Signal Processing Architecture for Soft-Output MIMO Lattice Reduction Aided Detection

Alan T. Murray and Steven R. Weller

Additional information is available at the end of the chapter

http://dx.doi.org/10.5772/51649

1. Introduction

Many wireless communication standards now include the use of multiple transmit and receive antennas as a means of achieving increased throughput or spectral efficiency, including LTE, WiMAX and WiFi (IEEE 802.11n). The task of a detector for a multi-input multi-output (MIMO) communications channel is to separate the spatially mixed and noise-corrupted data streams, and to produce reliable estimates of the transmitted bits. The brute-force maximum-likelihood (ML) detector provides optimal error-rate performance, but is computationally infeasible when either dense symbol constellations or large numbers of antennas are used. Hardware implementation of ML receivers is therefore very challenging, leading to linear detectors based on well-known approaches such as zero forcing (ZF) or minimum mean-square error (MMSE) detection, or nonlinear methods such as successive interference cancellation (SIC), which offer manageable receiver complexity at the expense of highly suboptimal error-rate performance.

One powerful class of receivers which have been developed over the past decade is based on the highly developed mathematical theory of point lattices, which are periodic arrangements of discrete points. The basic idea is to consider the distortion introduced by the noise-free part of a MIMO channel as a representation of a lattice, then to perform suboptimal detection on an “improved" representation of the channel matrix based derived from a “reduced" lattice. The suitably reduced lattice facilitates the search for the lattice point closest to the received vector, shifting most of the computational complexity to a pre-processing step before linear detection. Such lattice reduction aided detection (LRAD) based approaches to MIMO receiver design have significantly closed the gap between feasible yet high-performance MIMO detection, and optimal (but impractical) ML detection.

INTECH
open science | open minds

© 2013 Murray and Weller; licensee InTech. This is an open access article distributed under the terms of the Creative Commons Attribution License (http://creativecommons.org/licenses/by/3.0), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

To date, most LRAD-based MIMO detectors produce hard outputs, in which an estimate of the most likely vector of transmitted symbols is generated. For high-performance wireless communication systems, however, it is commonplace that the information transmitted over the air is coded, thereby containing not only raw data, but also the redundant information needed to perform forward error correction (FEC) at the receiver. State-of-the-art FEC codes such as turbo codes and low-density parity-check (LDPC) codes [1], which require estimates of the *probability* that a given transmitted bit was a 1 or a 0, therefore call for *soft output* detectors. The extension of hard-output LRAD detectors to the soft-output case is therefore of high practical relevance, but also recognized as a difficult problem [2, p16]. In this chapter, we present what is believed to be the first digital signal processing (DSP) implementation of a soft-output lattice reduction aided MIMO detector, based on an approach to MIMO detection known as subspace LRAD (SLRAD) proposed by Windpassinger [3, 4].

The chapter is organized as follows. In Section 2 we present the wireless MIMO system model, with an emphasis on how transmitted symbols are drawn from point sets consistent with the lattice theoretic approach to follow. In Section 3 we formally define lattices, and present the most celebrated algorithm for lattice reduction, known as the Lenstra-Lenstra-Lovász (LLL) algorithm. We then show how hard-output lattice-based detection can be used in conjunction with commonly used linear MIMO detectors in Section 4. In Section 5 we outline Windpassinger's subspace-based approach to LRAD in which a list of candidate symbols is produced, thereby facilitating soft-output LRAD. Finally in Section 6 we present a detailed description of our hardware implementation of a soft-output lattice reduction aided MIMO detector.

2. System model

We consider a MIMO wireless communication system with n_{T} transmit and n_{R} receive antennas. The complex baseband model for this MIMO system is

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

where $\mathbf{y} \in \mathbb{C}^{n_{\mathrm{R}}}$ is the received vector, $\mathbf{H} \in \mathbb{C}^{n_{\mathrm{R}} \times n_{\mathrm{T}}}$ is the channel matrix, $\mathbf{n} \in \mathbb{C}^{n_{\mathrm{R}}}$ is the channel noise, and $\mathbf{x} \in \mathbb{C}^{n_{\mathrm{T}}}$ is the vector of transmitted symbols, as shown in Fig. 1.

We assume that the noise $\mathbf{n} \triangleq [n_1, n_2, \ldots, n_{n_{\mathrm{R}}}]^T$ contains independent and identically distributed (i.i.d.) elements $n_m \sim \mathcal{CN}(0, \sigma^2), m = 1, \ldots, n_{\mathrm{R}}$. The channel matrix $\mathbf{H}$ has i.i.d. entries $h_{m,n} \sim \mathcal{CN}(0, 1)$, for $m = 1, \ldots, n_{\mathrm{R}}$ and $n = 1, \ldots, n_{\mathrm{T}}$, where it is assumed that there are at least as many receive antennas as transmit antennas: $n_{\mathrm{R}} \geq n_{\mathrm{T}}$.

An uncorrelated Rayleigh fading propagation environment is therefore assumed in this chapter, though it should be noted that lattice reduction aided detection receivers similar to those presented later in this chapter have been proposed for environments in which there is either temporal [5] or frequency-selective [6] fading.

The task of the MIMO receiver is to recover $\mathbf{x}$ from $\mathbf{y}$, based on knowledge of both the channel realization $\mathbf{H}$ and the channel noise variance σ^2.

The vector of transmitted symbols is denoted $\mathbf{x} \triangleq [x_1, x_2, \ldots, x_{n_{\mathrm{T}}}]^T$. In this chapter we restrict attention to transmit symbols drawn from finite sets of points, known as *constellations*,

http://dx.doi.org/10.5772/51649

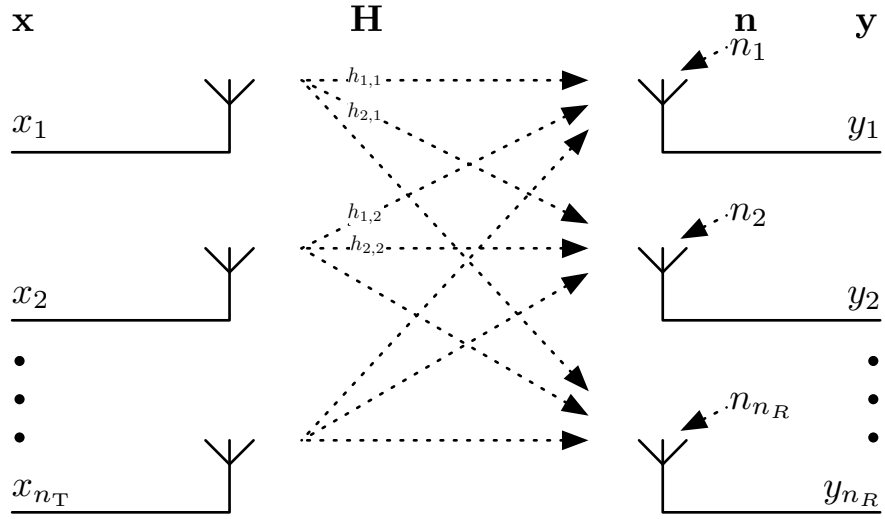

Figure 1. MIMO Wireless Channel

drawn from a square grid, and in particular the quadrature phase-shift keying (QPSK) and, 16-quadrature amplitude modulation (16-QAM) and 64-QAM constellations depicted in Fig. 2. We do not consider non-rectangular constellations, such as 8-PSK, due to an inherent incompatibility with the lattice-theoretic framework exploited by lattice reduction aided detection, and also the limited applicability of non-rectangular constellations in emerging wireless communication standards.

The symbol transmitted from the n^{th} antenna, denoted x_n, is drawn from a constellation $\mathcal{A}_n$:

$$x_n \in \sqrt{E_{sn}}\mathcal{A}_n, \tag{2}$$

where the scalar E_{sn} is the average transmitted symbol power. We define the vector $\mathbf{E_s} \triangleq [E_{s1}, E_{s2}, \ldots, E_{sn_T}]$ so that

$$\mathbb{E}\left[\mathbf{x}\mathbf{x}^H\right] = \text{diag}\left(\mathbf{E_s}\right). \tag{3}$$

The selection of $\mathbf{E_s}$ depends on the particular objective of transmit power scaling and indeed varies in practical implementations. In this chapter, to enable fair comparison between systems employing differing modulation formats, we constrain average unity power per information bit ($E_b = 1$).

The constellations considered in this chapter are formed from a subset of scaled and shifted Gaussian integers $\mathbb{Z}[i] \triangleq \{a + ib \mid a, b \in \mathbb{Z}\}$ [7, p. 230]:

$$\mathbb{X} \triangleq \left\{ a + ib + \frac{1+i}{2} \mid a, b \in \mathbb{Z} \right\}. \tag{4}$$

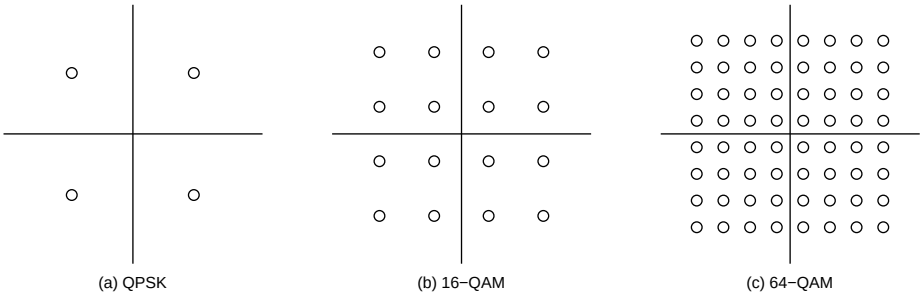

Figure 2. The three constellations used in this chapter

In this chapter, we restrict attention to the three subsets $\mathcal{X}_n \subset \mathbb{X}$ shown in Fig. 3, where the introduction of the offset term in (4) maintains symmetry of each constellation with respect to the axes. We refer to constellations formed in this manner as *Gaussian integer constellations*.

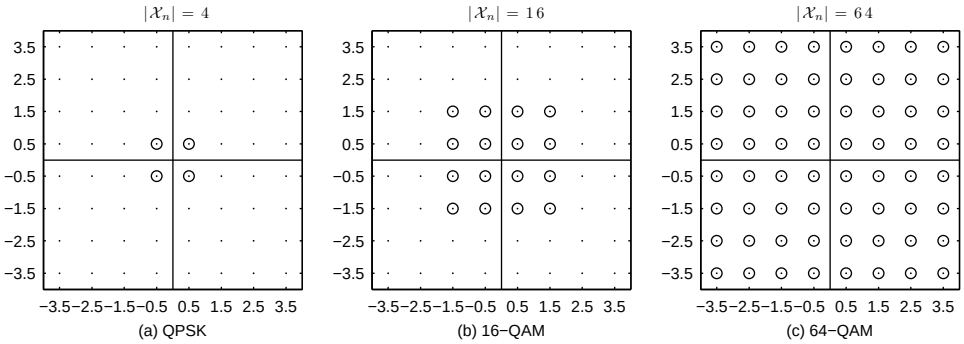

Figure 3. Three Gaussian integer constellations

The constellation $\mathcal{A}_n$ with elements $\alpha_n \in \mathcal{A}_n$ employed at the n^{th} transmit antenna is:

$$\mathcal{A}_n = \frac{\mathcal{X}_n}{\sqrt{c_n}}, \tag{5}$$

where c_n is the average energy of $\mathcal{X}_n$. Dividing each element of $\mathcal{X}_n$ by $\sqrt{c_n}$ ensures that $\mathcal{A}_n$ has unity average energy and is referred to as normalized constellations. For square QAM constellations such as those in Fig. 2, $c_n = (|\mathcal{X}_n| - 1)/6$.

It is important to note that we deliberately allow each transmit antenna to be independently mapped to a constellation set. In summary, the transmitted symbols x_n are formed by scaling

http://dx.doi.org/10.5772/51649

of the elements $\bar{x}_n \in \mathcal{X}_n$:

$$x_n = \sqrt{\frac{E_{sn}}{c_n}}\bar{x}_n. \tag{6}$$

The effect of a given channel realization $\mathbf{H}$ is to rotate and stretch (or contract) the axes of the otherwise square decision regions of the optimal, maximum-likelihood (ML) receiver. The error probability of a detector is determined by the distance of constellation points (mapped by $\mathbf{H}$) from the associated decision boundaries. The essential idea of LR-aided detectors is to obtain a "more orthogonal" representation for the channel realization $\mathbf{H}$, before detection using a low-complexity (sub-optimal) receiver. In the following section we make these ideas precise, drawing on the well-established mathematical literature on point lattices to formalize what is meant by the notion of a "more orthogonal" representation, and how it can be achieved and quantified.

3. Lattice reduction

3.1. Lattices

A *complex lattice* consists of all linear combinations of the set of linearly independent basis column vectors $\mathbf{b}_k, 1 \leq k \leq M$ of the basis matrix $\mathbf{B} \in \mathbb{C}^{N\times M}, M \leq N$. A complex lattice formed from basis matrix $\mathbf{B}$ is therefore the set of points

$$\mathcal{L}(\mathbf{B}) \triangleq \left\{ \sum_{k=1}^{M} s_k \mathbf{b}_k \mid s_k \in \mathbb{Z}[i] \right\},$$

where $\mathbb{Z}[i] \triangleq \{a + ib \mid a, b \in \mathbb{Z}\}$ is the ring of Gaussian integers [7].

The number of possible bases for a given lattice $\mathcal{L}$ is infinite, since any basis $\tilde{\mathbf{B}} = \mathbf{BT}$ forms the same lattice $\mathcal{L}(\tilde{\mathbf{B}}) = \mathcal{L}(\mathbf{B})$ when the transformation matrix $\mathbf{T}$ is unimodular, i.e. $\det(\mathbf{T}) = \pm 1$ and $\mathbf{T} \in \mathbb{Z}[i]^{M\times M}$. Finding a basis in which the basis vectors are (roughly speaking) reasonably short and almost orthogonal is known as lattice *basis reduction*, which we now describe formally.

3.2. Lenstra-Lenstra-Lovász (LLL) algorithm

The Lenstra-Lenstra-Lovász (LLL) algorithm was originally published as a lattice reduction algorithm operating on real-valued matrices [8]. Many works use the real decomposition of the complex-valued MIMO transmission model [3, 9]. Lattice reduction methods can operate on both real and complex integer lattices and in particular the LLL algorithm has been extended for complex lattice reduction [10]. The complex LLL (CLLL) algorithm can be summarized as follows. We make the following definitions:

- $\mathcal{H}_i$ is the squared Euclidean norm of the orthogonal vectors produced by the Gram-Schmidt orthogonalization (GSO) of $\mathbf{H}$

- μ_{ij} is the ratio of the length of the orthogonal projection of the i^{th} basis onto the j^{th} orthogonal vector and the length of the j^{th} orthogonal vector
- $\mathbf{H}^i{}_{\text{L}}$ and $\mathbf{T}^i$ represent the values of the reduced basis and transform after the i^{th} step of the LLL algorithm
- Initially, $\mathbf{H}_{\text{L}}{}^0 = \mathbf{H}$ and $\mathbf{T}^0 = \mathbf{I}_{n_{\text{T}}}$
- k is the index of the current column of $\mathbf{H}$ being processed such that $2 \leq k \leq n_{\text{T}}$

The LLL algorithm consists of three basic steps:

1. $\mathcal{H}$ and μ are computed using a modified GSO procedure [11]
2. Size reduction aims to make basis vectors shorter and more orthogonal by asserting the condition that $|\Re(\mu_{k,j})| \leq 0.5$ and $|\Im(\mu_{k,j})| \leq 0.5$ for all $j < k$
3. Basis vectors $\mathbf{h}_{k-1}$ and $\mathbf{h}_k$ are swapped if a so-called *swapping condition* is satisfied such that size reduction can be repeated to make basis vectors shorter

Size reduction and basis vector swapping iterates until the swapping condition is no longer satisfied by any pair of $\mathbf{h}_{k-1}$ and $\mathbf{h}_k$. The resultant basis is then said to be *reduced*. The swapping condition for LLL reduction, also called the *Lovász condition*, is:

$$\mathcal{H}_k < (\delta - |\mu_{k,k-1}|^2)\mathcal{H}_{k-1}, \tag{7}$$

where δ satisfying $\frac{1}{4} < \delta < 1$ is a factor selected to achieve an acceptable quality-complexity trade off [8].

After each swapping step, $\mathcal{H}_{k-1}$, $\mathcal{H}_k$ and some of the $\mu_{i,j}$ values needed to be updated. Techniques can be employed to minimize the number and frequency of recalculations of $\mathcal{H}$ and μ elements [11]. The LLL algorithm is detailed in Algorithm 1.

Example 3.1. *Suppose* $\delta = \frac{3}{4}$ *and*

$$\mathbf{H} = \begin{bmatrix} 0.75 & -0.5 \\ 0.5 & -0.5 \end{bmatrix}.$$

Then

$$\mathbf{H}^0{}_L = \begin{bmatrix} 0.75 & -0.5 \\ 0.5 & -0.5 \end{bmatrix} \text{ and } \mathbf{T}^0 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix},$$

and from the modified GSO

$$\mu = \begin{bmatrix} 1.0000 & 0.0000 \\ -0.7692 & 1.0000 \end{bmatrix} \text{ and } \mathcal{H} = \begin{bmatrix} 0.8125 \\ 0.0192 \end{bmatrix}.$$

Starting with columns 1 and 2, as $|\mu_{2,1}| > 0.5$*, size reduction is performed on these columns adding the first column to the second and yielding the following partially reduced matrix and corresponding transform:*

http://dx.doi.org/10.5772/51649

Algorithm 1 $[\mathbf{H}, \mathbf{T}] \Leftarrow \mathrm{LLL}(\mathbf{H}, \delta)$

Input: $\mathbf{H} \in \mathbb{C}^{n \times m}$ **and** $\delta \in \mathbb{R}^n$
$\mathbf{T} \Leftarrow \mathbf{I}_n$, $k \Leftarrow 2$
for $j = 1$ to n **do**
$\mathcal{H}_j = \left\langle \mathbf{H}_j, \mathbf{H}_j \right\rangle$
end for
for $j = 1$ to n **do**
for $i = j + 1$ to n **do**
$\mu_{i,j} \Leftarrow \frac{1}{\mathcal{H}_j} \left(\left\langle \mathbf{h}_i, \mathbf{h}_j \right\rangle - \sum_{k=1}^{j-1} \mu_{j,k}^H \mu_{i,k} \mathcal{H}_k \right)$
$\mathcal{H}_i \Leftarrow \mathcal{H}_i - |\mu_{i,j}|^2 \mathcal{H}_j$
end for
end for
while $k \leq n$ **do**
$[\mathbf{H}, \mathbf{T}, \mu] \Leftarrow \mathrm{Reduce}\,(\mathbf{H}, \mathbf{T}, \mu, k, k-1)$ // Size Reduction
if $\mathcal{H}_k < (\delta - |\mu_{k,k-1}|^2)\mathcal{H}_{k-1}$ **then** // Lovász condition check
Swap columns k and $k-1$ of $\mathbf{H}$ and $\mathbf{T}$
Update $\mathcal{H}$ and μ where $\dot{\mathcal{H}}$ and $\dot{\mu}$ denote the new values
$\dot{\mathcal{H}}_{k-1} = \mathcal{H}_k + |\mu_{k,k-1}|^2 \mathcal{H}_{k-1}$
$\dot{\mu}_{k,k-1} = \mu_{k,k-1}^H \left(\frac{\mathcal{H}_{k-1}}{\dot{\mathcal{H}}_{k-1}} \right)$
$\dot{\mathcal{H}}_k = \left(\frac{\mathcal{H}_{k-1}}{\dot{\mathcal{H}}_{k-1}} \right) \mathcal{H}_k$
$\dot{\mu}_{i,k-1} = \mu_{i,k-1} \dot{\mu}_{k,k-1} + \mu_{i,k} \left(\frac{\mathcal{H}_k}{\dot{\mathcal{H}}_{k-1}} \right)$ // $i = k+1$ to n
$\dot{\mu}_{i,k} = \mu_{i,k-1} - \mu_{i,k} \mu_{k,k-1}$ // $i = k+1$ to n
$\dot{\mu}_{k-1,j} = \mu_{k,j}$ // $j = 1$ to $k-2$
$\dot{\mu}_{k,j} = \mu_{k-1,j}$ // $j = 1$ to $k-2$
$k = \max(2, k-1)$
else
for $j = k-2$ **downto** 1 **do**
$[\mathbf{H}, \mathbf{T}, \mu] \Leftarrow \mathrm{Reduce}\,(\mathbf{H}, \mathbf{T}, \mu, k, j)$ // Size Reduction
end for
$k = k + 1$
end if
end while

$$\mathbf{H}^1{}_L = \begin{bmatrix} 0.75 & 0.25 \\ 0.5 & 0 \end{bmatrix} \text{ and } \mathbf{T}^1 = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix},$$

$$\mu = \begin{bmatrix} 1.0000 & 0.0000 \\ 0.2308 & 1.0000 \end{bmatrix} \text{ and } \mathcal{H} = \begin{bmatrix} 0.8125 \\ 0.0192 \end{bmatrix}.$$

Algorithm 2 $[\mathbf{H}, \mathbf{T}, \mu] \Leftarrow \text{Reduce}(\mathbf{H}, \mathbf{T}, \mu, k, j)$

if $|\Re\left(\mu_{k,j}\right)| > \frac{1}{2}$ **or** $|\Im\left(\mu_{k,j}\right)| > \frac{1}{2}$ **then**
  $c \Leftarrow \lfloor \mu_{k,j} \rceil$
  $\mathbf{H}_k \Leftarrow \mathbf{H}_k - c\mathbf{H}_j$
  $\mathbf{T}_k \Leftarrow \mathbf{T}_k - c\mathbf{T}_j$
  for $l = 1$ **to** j **do**
    $\mu_{k,l} \Leftarrow \mu_{k,l} - c\mu_{j,l}$
  end for
end if

Next the Lovász condition is checked and, since $\mathcal{H}_2 < (\delta - |\mu_{2,1}|^2)\mathcal{H}_1$, *the two columns are swapped, yielding:*

$$\mathbf{H}^2{}_L = \begin{bmatrix} 0.25 & 0.75 \\ 0 & 0.5 \end{bmatrix} \text{ and } \mathbf{T}^2 = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix},$$

$$\mu = \begin{bmatrix} 1.0000 & 0.0000 \\ 3.0000 & 1.0000 \end{bmatrix} \text{ and } \mathcal{H} = \begin{bmatrix} 0.0625 \\ 0.2500 \end{bmatrix}.$$

Size reduction is then performed on the columns once more; this time by subtracting three times the first column from the second we have:

$$\mathbf{H}_L = \begin{bmatrix} 0.25 & 0 \\ 0 & 0.5 \end{bmatrix} \text{ and } \mathbf{T} = \begin{bmatrix} 1 & -2 \\ 1 & -3 \end{bmatrix},$$

$$\mu = \begin{bmatrix} 1.0000 & 0.0000 \\ 0.0000 & 1.0000 \end{bmatrix} \text{ and } \mathcal{H} = \begin{bmatrix} 0.0625 \\ 0.2500 \end{bmatrix}.$$

The Lovász condition (7) *is now satisfied, and the algorithm terminates.* ■

3.3. Orthogonality defect

The orthogonality of a matrix $\mathbf{H}$ can be quantified using the *orthogonality defect*, defined as [4, §4.6.2]:

$$\delta(\mathbf{H}) = \frac{\prod_{k=1}^{n_\mathrm{T}} \|\mathbf{h}_k\|}{\left|\sqrt{\det(\mathbf{H}^H\mathbf{H})}\right|}, \tag{8}$$

where $\mathbf{h}_k$ is the k^{th} column of $\mathbf{H}$, $\delta(\mathbf{H}) \geq 1$ for all $\mathbf{H}$ and $\delta(\mathbf{H}) = 1$ if and only if the columns of $\mathbf{H}$ are orthogonal. When the number of columns and rows of $\mathbf{H}$ are equal, the denominator can be simplified to $|\det(\mathbf{H})|$. From (8), matrices with correlated columns or larger column norms will result in higher orthogonality defects. This also causes their inverse

http://dx.doi.org/10.5772/51649

or generalized inverse to have larger row norms, leading to noise enhancement. As will be shown in Section 4, matrices with a lower orthogonality defect therefore induce less noise enhancement in ZF- or MMSE-based detectors as the probability of error, for example as calculated in (15), can be reduced.

To illustrate the impact of lattice reduction on orthogonality defect, we generated 10^6 randomly chosen $\mathbf{H} \in \mathbb{C}^{4\times 4}$, and computed the lattice reduced equivalent $\mathbf{H}_{\mathrm{L}}$. The orthogonality defect was calculated using (8) both before and after lattice reduction. The results are presented in the form of cumulative distributions in Fig. 4, where the effect of lattice reduction on orthogonality defect is clearly apparent. Lattice basis reduction has also been shown to improve matrix conditioning [12]. It is this improvement that reduces noise enhancement in linear detection methods and reduces the error rate of LRAD-based systems.

Numerous researchers have investigated and compared the application of various lattice reduction algorithms for MIMO detection. In addition to the LLL algorithm, these include Korkine–Zolotarev (KZ) [13], and Seysen's [14] lattice reduction algorithms; see [2] and the references therein for applications to MIMO detection. In this chapter we restrict attention to the LLL algorithm, since numerous simulation studies suggest that lattice-reduction-aided detection is well suited to low-complexity MIMO receivers when large constellations are used [15, 16].

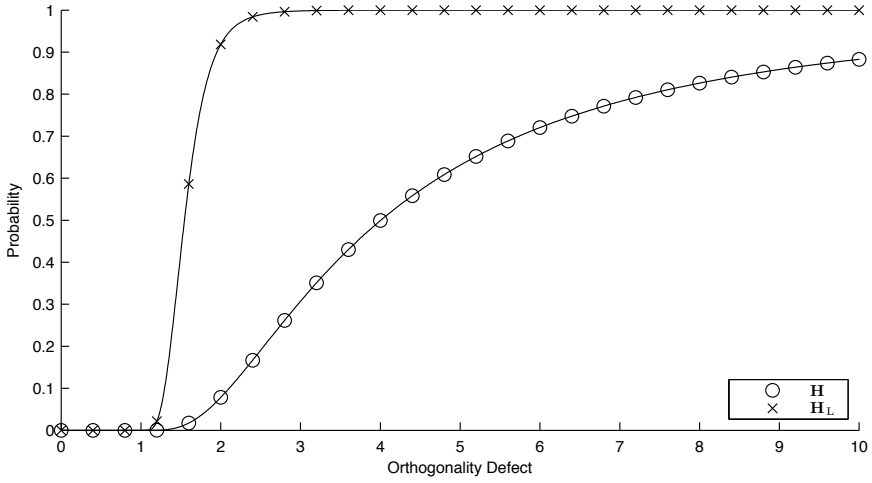

Figure 4. Cumulative distributions of the orthogonality defect for non-reduced and reduced basis channel matrices

4. Hard detection using lattice reduction

Detectors which output an estimate of the most likely vector of transmitted symbols are said to be *hard output* detectors. Hard estimates are denoted $\widehat{\mathbf{b}}$ for bit vector estimates and $\widehat{\mathbf{x}}$ for symbol vector estimates. Detectors which generate not just a vector of bit estimates but also an estimate of the *probability* that a given transmitted bit was a 1 or a 0 are said to be *soft output* detectors. Soft output detectors provide a significant benefit when combined with channel

coding schemes which make use of soft information, such as turbo codes or low-density parity-check (LDPC) codes, but typically increase receiver complexity by a significant degree.

4.1. Maximum-Likelihood detection

The maximum-likelihood (ML) detector selects from the set of possible transmitted symbol vectors $\mathbf{x} \in \mathcal{A}^{n_\mathrm{T}}$ the vector $\widehat{\mathbf{x}}_\mathrm{ML}$ which minimizes the Euclidean distance to the receive vector:

$$\widehat{\mathbf{x}}_\mathrm{ML} = \underset{\mathbf{x}\in\mathcal{A}^{n_\mathrm{T}}}{\arg\min} \|\mathbf{y} - \mathbf{H}\mathbf{x}\|^2. \tag{9}$$

This is achieved by exhaustively examining all possible transmit vectors; see Algorithm 3. Whilst the ML detection algorithm is conceptually simple, its complexity is exponential in the size of the constellation and number of transmit antennas, and is therefore practical for real-time hardware implementation only in the simplest of settings. As the optimal detector, the performance of the the ML detector serves as a benchmark for the detection schemes of the following sections.

Algorithm 3 ML Algorithm - $[\widehat{\mathbf{x}}, \mathbf{b}] \Leftarrow \mathrm{MLdetect}\left(\mathbf{H}, \sigma^2, \mathbf{y}\right)$

```
e_min ⇐ inf
for b = 0 to 2^n_BPT − 1 do                          // 2^n_BPT iterations
   x̂ ⇐ mod(b)
   e ⇐ ‖y − Hx̂‖^2                                    // 4n_R n_T + 2n_R M, 4n_R n_T + 4n_R A
   if e < e_min then
      e_min = e
      x̂_ml = x̂
      b_ml = b
   end if
end for
```

4.2. Zero Forcing estimation

The most straightforward linear detection scheme is *zero forcing* (ZF), also known as least squares estimation, which works to reverse the effect of the MIMO channel matrix on the transmitted symbols. By finding the least squares solution to (1), it is referred to as zero forcing as the interference caused by $\mathbf{H}$ is forced to zero by multiplication of the received vector $\mathbf{y}$ by $\mathbf{W}_\mathrm{ZF}$, the inverse (or generalized inverse) of the channel matrix:

$$\tilde{\mathbf{x}}_\mathrm{ZF} = \mathbf{W}_\mathrm{ZF}\mathbf{y}. \tag{10}$$

We use the notation $\tilde{\mathbf{x}}$ to represent an unconstrained estimate of the vector of transmitted symbols. The likelihood that $\tilde{\mathbf{x}}$ actually maps to a constellation point is negligibly small and so the nearest valid constellations point must be found. ZF finds the estimate of the vector of transmitted symbols $\widehat{\mathbf{x}}_\mathrm{ZF}$ as follows:

$$\widehat{\mathbf{x}}_\mathrm{ZF} = \underset{\mathbf{x}\in\mathcal{A}^{n_\mathrm{T}}}{\arg\min} \|\mathbf{W}_\mathrm{ZF}\mathbf{y} - \mathbf{x}\|^2, \tag{11}$$

http://dx.doi.org/10.5772/51649

where $\hat{\mathbf{x}}_{\mathrm{ZF}}$ is found by independently rounding each element of $\tilde{\mathbf{x}}$ to the nearest constellation point. The vector $\hat{\mathbf{x}}_{\mathrm{ZF}}$ can then be demodulated to find $\mathbf{b}_{\mathrm{ZF}}$, an estimate of the vector of transmitted bits, as shown in Algorithm 4.

There are numerous methods to find the least squares solution to (1), including those that directly calculate the matrix $\mathbf{W}_{\mathrm{ZF}}$. In this chapter, we utilize the well known Moore-Penrose pseudoinverse:

$$\mathbf{W}_{\mathrm{ZF}} = \left(\mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{H}^H. \tag{12}$$

Algorithm 4 ZF Algorithm - $[\hat{\mathbf{x}}, \mathbf{b}] \Leftarrow \mathrm{ZFdetect}\left(\mathbf{H}, \sigma^2, \mathbf{y}\right)$

$\mathbf{W}_{\mathrm{ZF}} = \left(\mathbf{H}^H\mathbf{H}\right)^{-1}\mathbf{H}^H$
$\tilde{\mathbf{x}} \Leftarrow \mathbf{W}_{\mathrm{ZF}}\mathbf{y}$ // $4n_{\mathrm{R}}n_{\mathrm{T}}$ M, $4n_{\mathrm{R}}n_{\mathrm{T}}$ A
$\delta \Leftarrow [1+i]/2$
$\hat{\mathbf{x}}_{\mathrm{ZF}} \Leftarrow \mathrm{round}\left(\tilde{\mathbf{x}}, \delta\right)$
$\mathbf{b}_{\mathrm{ZF}} \Leftarrow \mathrm{demod}\left(\hat{\mathbf{x}}_{\mathrm{ZF}}\right)$

4.3. Noise enhancement

Whilst ZF completely reverses the effects of the MIMO channel matrix, if the columns of $\mathbf{H}$ are correlated, ZF will amplify or enhance the noise. By identifying that $\mathbf{W}_{\mathrm{ZF}}\mathbf{H} = \mathbf{I}$ and then multiplying (1) by $\mathbf{W}_{\mathrm{ZF}}$ we can calculate the effective additive noise component of the estimated vector of transmitted symbols:

$$\tilde{\mathbf{x}}_{\mathrm{ZF}} = \mathbf{x} + \mathbf{W}_{\mathrm{ZF}}\mathbf{n}. \tag{13}$$

It is intuitive that the noise existing in the unconstrained transmit symbol estimate $\tilde{\mathbf{x}}_{\mathrm{ZF}}$ is $\mathbf{W}_{\mathrm{ZF}}\mathbf{n}$. When the rows of $\mathbf{W}_{\mathrm{ZF}}$ have a large Euclidean distance, multiplication of the received vector leads to the additive noise component in $\mathbf{y}$ being amplified. We can now show how a poorly conditioned or correlated channel matrix will result in significant noise enhancement in ZF by examining the probability of error:

$$\begin{aligned} \mathbf{e} &= \tilde{\mathbf{x}} - \mathbf{x} \\ &= \mathbf{W}_{\mathrm{ZF}}\mathbf{n} \end{aligned} \tag{14}$$

$$\begin{aligned} p_e &= \mathrm{diag}\left(\epsilon_n(\mathbf{e}\mathbf{e}^H)\right) \\ &= \sigma^2 \mathrm{diag}\left(\left(\mathbf{H}^H\mathbf{H}\right)^{-1}\right) \end{aligned} \tag{15}$$

Existing work [17] has looked at the statistical properties of the channel matrix, and in particular the effect of this noise enhancement, leading to a tight analytical bound of the performance of ZF detectors in Rayleigh fading channels.

4.4. Minimum Mean-Square Error (MMSE) estimation

MMSE estimation acts to balance the reduction of the interference caused by $\mathbf{H}$ and the noise enhancement due to correlation of the columns in $\mathbf{H}$. Rather than completely remove the effect of the MIMO channel, MMSE estimation works to find a coefficient which minimizes the criterion:

$$\mathbf{W}_{\text{MMSE}} = \arg\min_{\mathbf{W}} \|\mathbf{W}\mathbf{y} - \mathbf{x}\|^2. \tag{16}$$

The solution to (16) is the well-known MMSE estimator, also known as the Wiener filter:

$$\mathbf{W}_{\text{MMSE}} = \left(\mathbf{H}^H\mathbf{H} + \sigma^2\mathbf{I}_{n_{\text{T}}}\right)^{-1}\mathbf{H}^H \tag{17}$$

$$= \begin{bmatrix}\mathbf{H} \\ \sigma\mathbf{I}\end{bmatrix}^{\dagger} \tag{18}$$

The shorthand notation of (18) was first proposed in [18] and is referred to as the *extended channel matrix*, which in this chapter is denoted

$$\overline{\mathbf{H}} = \begin{bmatrix}\mathbf{H} \\ \sigma\mathbf{I}\end{bmatrix}. \tag{19}$$

Similarly to ZF detection, MMSE detection finds the estimate of the vector of transmitted symbols $\widehat{\mathbf{x}}_{\text{MMSE}}$ as follows:

$$\widehat{\mathbf{x}}_{\text{MMSE}} = \arg\min_{\mathbf{x}\in\mathcal{A}^{n_{\text{T}}}} \|\mathbf{W}_{\text{MMSE}}\mathbf{y} - \mathbf{x}\|^2, \tag{20}$$

where $\widehat{\mathbf{x}}_{\text{MMSE}}$ is found by independently rounding each element of $\tilde{\mathbf{x}}$ to the nearest constellation point. It is well-known that as the noise term approaches zero (at high signal-to-noise ratios), the MMSE estimator becomes equivalent to a ZF estimator.

Compared to ZF detection, MMSE results on average in less noise enhancement, as $\overline{\mathbf{H}}$ is better conditioned. This can be seen intuitively as a result of adding a diagonal matrix relating to the noise variance as in (17) or alternatively due to the stacked structure of (18) resulting in a decrease in correlation. Unlike ZF, however, MMSE does not perfectly reverse or remove the interference of $\mathbf{H}$, leading to interference between the otherwise independent transmit antennas. As with ZF, analytical performance bounds for MMSE detectors have been developed [17, 19] for various channel models.

Utilizing the shorthand notation of the extended channel matrix of (18), ZF detection can be readily extended to perform MMSE detection, as shown in Algorithm 5. Note that due to the extra rows of $\overline{\mathbf{H}}$ as compared to $\mathbf{H}$, the computational complexity of calculating $\mathbf{W}_{\text{MMSE}}$ is roughly double that of $\mathbf{W}_{\text{ZF}}$.

http://dx.doi.org/10.5772/51649

Algorithm 5 MMSE Algorithm - $[\widehat{\mathbf{x}}, \mathbf{b}] \Leftarrow \text{MMSEdetect}\left(\mathbf{H}, \sigma^2, \mathbf{y}\right)$

$$\overline{\mathbf{H}} \Leftarrow \begin{bmatrix} \mathbf{H} \\ \sigma \mathbf{I} \end{bmatrix}$$

$$[\widehat{\mathbf{x}}_{\text{MMSE}}, \mathbf{b}_{\text{MMSE}}] \Leftarrow \text{ZFdetect}\left(\overline{\mathbf{H}}, \sigma^2, \mathbf{y}\right)$$

4.5. Detection using Lattice Reduction

Lattice basis reduction [20, §2.6.1] reduces the orthogonality defect, thereby reducing noise enhancement. This is achieved by finding a closer to orthogonal set of basis vectors. This reduced lattice basis is found by optimizing the generating matrix, which in the present application is a MIMO channel matrix realization. This closer-to-orthogonal set is found using elementary operations on basis vectors. Complex integer linear combinations of the column vectors of $\mathbf{H}$ are taken to form the reduced matrix $\mathbf{H}_{\text{L}}$ which spans the same set of points $\mathbf{H}\mathbb{X}^{n_{\text{T}}} \equiv \mathbf{H}_{\text{L}}\mathbb{X}^{n_{\text{T}}}$ and so

$$\mathbf{H}_{\text{L}} = \mathbf{H}\mathbf{T} \text{ or } \mathbf{H} = \mathbf{H}_{\text{L}}\mathbf{T}^{-1}, \tag{21}$$

where $\mathbf{T}$ is a unimodular matrix with complex integer entries and $\det(\mathbf{T}) = \pm 1$, therefore $\mathbf{T}^{-1}$ also contains only complex integer entries.

As in [3], by finding an equivalent and closer to orthogonal set of the basis vectors, $\mathbf{H}_{\text{L}}$, noise enhancement is reduced when quantization is performed. Importantly, as $\mathbf{T}^{-1}$ and $\bar{\mathbf{x}}$ both contain only integer spaced entries, so does $\mathbf{T}^{-1}\bar{\mathbf{x}}$ and so symbol detection or quantization is merely rounding to the grid $\mathbb{X}$.

Once the lattice reduced channel matrix is found, we then calculate the pseudoinverse as would be done in ZF or MMSE detection. LRAD therefore operates using the following steps, which are adapted from [3] and detailed in [21]:

1. Find the reduced lattice basis
2. Use the pseudoinverse of the reduced basis to form estimates
3. Quantize estimates to $\mathbb{X}$
4. Transform and bound points to constellation points

As shown in Algorithm 6, received vectors $\mathbf{y}$ are multiplied with the pseudoinverse of the reduced basis $\mathbf{H}_{\text{L}}$ to find a soft estimate of the vector of transmitted symbols in the reduced domain. These symbols are then quantized to an integer grid. (Depending on the transform generated, this integer grid may be offset by a half in both real and imaginary dimensions.) These hard estimates are then transformed, using the transform matrix $\mathbf{T}$ generated by the LR algorithm, to find an estimate of the vector of transmitted symbols. However, as these symbols may fall outside the range of constellation points invalid constellation points are clipped back to the nearest constellation point.

Algorithm 6 LRAD Algorithm - $[\hat{\mathbf{x}}, \mathbf{b}] \Leftarrow \text{LRADdetect}\left(\mathbf{H}, \sigma^2, \mathbf{y}\right)$

$[\mathbf{H}_L, \mathbf{T}] \Leftarrow \text{LR}(\mathbf{H})$ // LR is a lattice reduction algorithm such as Algorithm 1
$\delta \Leftarrow \mathbf{T}[1+i]/2$
$\tilde{\mathbf{x}} \Leftarrow \bar{\mathbf{H}}_L^{\dagger}\mathbf{y}$
$\hat{\mathbf{x}}_{LRZF} \Leftarrow \mathbf{T}\left(\text{round}\left(\tilde{\mathbf{x}}, \delta\right)\right)$ // $4{n_\text{T}}^2$ M, $4{n_\text{T}}^2$ A
$\mathbf{b}_{ZF} \Leftarrow \text{demod}\left(\hat{\mathbf{x}}_{LRZF}\right)$

5. Subspace-based LRAD

5.1. Hard-output SLRAD

For hard estimation, quantization of the ZF or MMSE estimate in the transmit constellation domain is replaced by the same quantization in the lattice reduced domain. The equivalent for soft estimation calls for the calculation of the error induced by quantization in the lattice reduced domain. Unfortunately, just as it is hard to ensure quantization to valid symbols in the lattice reduced domain, it is equally hard to iterate over all possible valid symbols in the lattice reduced domain in order to estimate each bit probability.

Whilst Zhang et al. [22] present a detailed comparison of various soft output based detectors and proposes several powerful methods for generating soft output information, there are some key shortcomings, and the performance of the detectors in [22] are only evaluated using QPSK constellations. This is problematic in that a range of wireless communication standards are moving to denser constellations, such as 16-QAM and 64-QAM. This motivates the investigation of lattice reduction based detectors capable of producing *candidate lists*.

The subspace lattice reduction aided detection (SLRAD) approach of Windpassinger [3] forms a subspace of the channel matrix $\mathbf{H}$ by removing a single column from the channel matrix. This column removal allows the corresponding transmit antenna's symbol estimate to be constrained in order to calculate an estimate for what the other transmit antennae sent. For each transmit antenna a number of symbols is systematically proposed and for each proposal the set of most likely symbols transmitted on the other antennae is calculated, as shown in Algorithm 7.

The SLRAD algorithm therefore creates a list of candidate symbols, the Euclidean distance of each of these candidates from the origin being used to determine the most likely vector of transmitted symbols for a hard-output detector.

Whilst performance of SLRAD is close to that of ML (see Fig. 5), the complexity is proportional only to the sum of the size of the constellations employed on each transmit antenna. Therefore only a modest number of candidate symbols needs to be investigated, even for dense constellations. For example, a system with 4 transmit antennas each utilizing 64-QAM results in only $4 \times 64 = 256$ candidates.

5.2. Soft-output SLRAD

As a candidate-based detector, the hard-output SLRAD detector can be extended to generate soft output information. The probability of all the candidates where a bit is one is divided by the probability of all candidates where the bit is zero. An attractive property of subspace

http://dx.doi.org/10.5772/51649

Algorithm 7 SLRAD Algorithm - $[\widehat{\mathbf{x}}, \mathbf{b}] \Leftarrow \text{SLRADdetect}\left(\mathbf{H}, \sigma^2, \mathbf{y}\right)$

$e_{min} \Leftarrow \text{inf}$
for $k = 1$ to n_{T} **do**
  $\mathbf{H}_s \Leftarrow \mathbf{H}_{[1\ldots(k-1)(k+1)\ldots n_{\text{T}}]}$
  for all s in $\mathcal{A}_k$ **do**
    $\mathbf{y}_s \Leftarrow \mathbf{y} - \mathbf{h}_k s$ // $4n_{\text{R}}$ M, $4n_{\text{R}}$ A
    $\widehat{\mathbf{x}}_s \Leftarrow \text{LRADdetect}\left(\mathbf{H}_s, \mathbf{y}_s, \sigma^2\right)$
    $\widehat{\mathbf{x}}_{[1\ldots(k-1)(k+1)\ldots n_{\text{T}}]} = \widehat{\mathbf{x}}_s$
    $\widehat{\mathbf{s}}_{\mathbf{k}} = s$
    $e \Leftarrow \|\mathbf{y} - \mathbf{H}\widehat{\mathbf{x}}\|^2$
    if $e < e_{min}$ **then**
      $e_{min} = e$
      $\widehat{\mathbf{x}}_{SLR} = \widehat{\mathbf{x}}$
    end if
  end for
end for
$\mathbf{b}_{SLR} \Leftarrow \text{demod}\left(\widehat{\mathbf{x}}_{SLR}\right)$

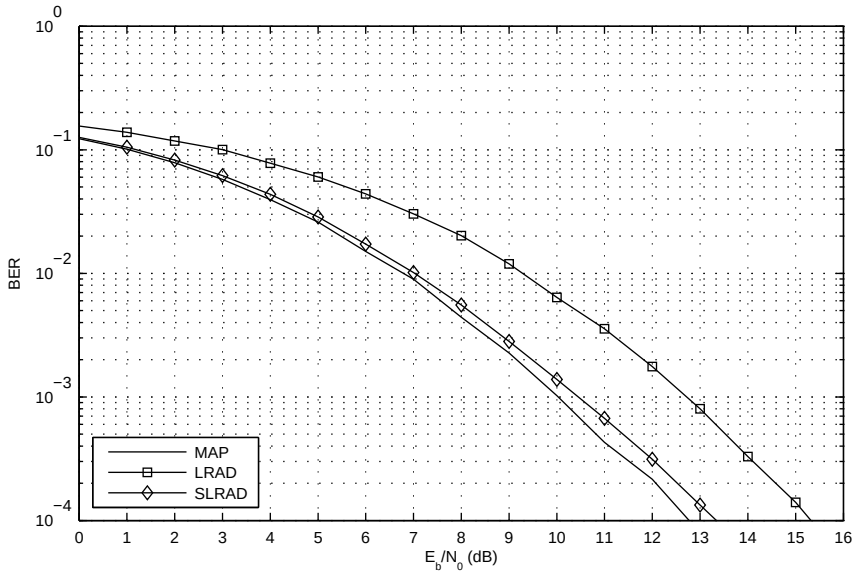

Figure 5. Bit error rate (BER) Performance of ML, LRAD and SLRAD for 4×4 MIMO with 16-QAM

detectors is that every bit is guaranteed to have at least one candidate where the bit is a one and likewise a candidate where it is zero. Without this property, it is not possible to accurately form an estimate for the ratio of the bit's value probabilities.

Algorithm 8 Soft Output SLRAD Algorithm - $[\mathbf{L}_e] \Leftarrow$ SLRADdetect-soft $\left(\mathbf{H}, \sigma^2, \mathbf{y}\right)$

$\mathbf{n}_{bit} \Leftarrow 0$
$\mathbf{d}_{bit} \Leftarrow 0$
for k = 1 to n_{T} **do**
  $\mathbf{H}_s \Leftarrow \mathbf{H}_{[1\ldots(k-1)(k+1)\ldots n_{\mathrm{T}}]}$
  for all s in $\mathcal{A}_k$ **do**
    $\mathbf{y}_s \Leftarrow \mathbf{y} - \mathbf{h}_k s$
    $\widehat{\mathbf{x}}_s \Leftarrow$ LRADdetect $\left(\mathbf{H}_s, \mathbf{y}_s, \sigma^2\right)$
    $\widehat{\mathbf{x}}_{[1\ldots(k-1)(k+1)\ldots n_{\mathrm{T}}]} = \widehat{\mathbf{x}}_s$
    $\widehat{\mathbf{s}}_{\mathbf{k}} = s$
    $\mathbf{b} \Leftarrow$ demod $(\widehat{\mathbf{x}})$
    $e \Leftarrow \exp\left(\frac{-\|\mathbf{y}-\mathbf{H}\widehat{\mathbf{x}}\|^2}{\sigma^2}\right)$
    for all bits in current bit vector **do**
      if the current bit is a '1' **then**
        $\mathbf{n}_{bit} = \mathbf{n}_{bit} + e$
      else
        $\mathbf{d}_{bit} = \mathbf{d}_{bit} + e$
      end if
    end for
  end for
end for
$\mathbf{L}_e \Leftarrow \log[\mathbf{n}] - \log[\mathbf{d}]$

The soft-output SLRAD algorithm is shown in Algorithm 8. This algorithm leads in a natural fashion to the top-level data flow diagram in Fig. 6. The candidate chain block in Fig. 7 performs the following key steps (once for each submatrix of $\mathbf{H}$ formed by deleting one column from $\mathbf{H}$):

1. subspace candidate estimate generation;
2. lattice reduced domain quantization;
3. reversal of the lattice basis transform;
4. bounding to ensure valid constellation symbols; and
5. demodulation and Euclidean distance calculation.

6. Hardware implementation

6.1. Existing work

The first published VLSI implementation of a lattice reduction aided detector [23] is based on Brun's algorithm for finding integer relations [24]. Brun's algorithm offers lower complexity at a performance cost when compared to the commonly utilized complex LLL algorithm.

http://dx.doi.org/10.5772/51649

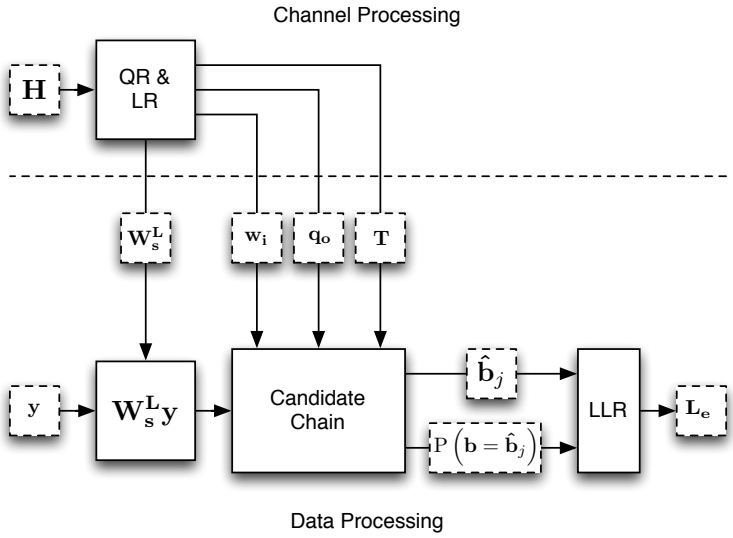

Figure 6. Top Level Data Flow Diagram

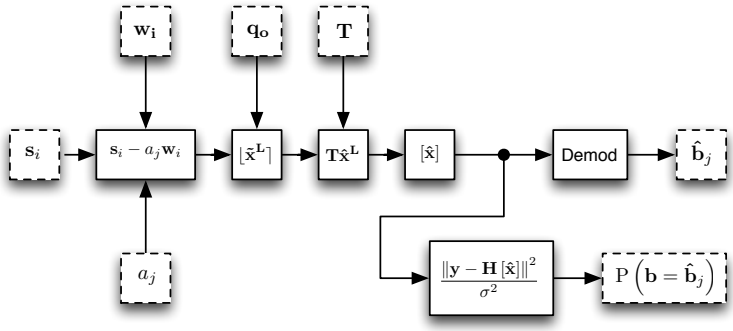

Figure 7. Candidate Chain Data Flow Diagram

Brun's algorithm is criticised in [25] as it achieves inferior performance and no analytical result has been reported to prove the level of diversity that can be achieved. This work applies a uniform scaling factor to the elements of the same matrix or vector to ensure that the magnitudes of the largest real and imaginary parts are as close as possible to, but smaller than one. This pragmatic approach offers a good compromise between true floating-point arithmetic, with its computational overhead, and a simple fixed point arithmetic with significantly reduced dynamic range. However, it appears that no active scaling is performed in the algorithm to prevent numeric overflow. Instead, it is claimed without substantiation that a bound exists which is used to calculate the required number of integer bits.

The work in [26] implements a sorted QR decomposition using Householder CORDIC units to reduce the number of LLL iterations needed. The complex LLL algorithm is used but, as with most LLL implementations, requires the use of divisions, using the Newton–Raphson algorithm, throughout the LLL iterations.

The work of [27] builds on [26] and discusses novel search based extensions to LRAD introduced in [28] which generate a candidate list and therefore soft outputs. However, the hardware implementation does not discuss this and therefore it is presumed that the hardware implementation is hard output. Due to the time-multiplexed complex multiplier pipeline, this approach is forced to rely on the use of priority inversion to prevent deadlocks due to data dependencies. Analysis is not performed on the precision required and in particular magnitude bounding is not performed which results in a large number of integer bits being required.

In [29], the authors build on their prior work [26, 27] by offering several improvements. This revision implements Sorted QRD to reduce the number of LLL swapping steps. Once again, the hardware implementation is presumed to only offer hard outputs as no mention is made of the candidate generation required to form soft outputs nor the hardware required to calculate LLRs. Unlike the prior works, an upper bound of 4 integer bits is identified for the elements of the R matrix which offers a significant reduction in the precision required.

Several works [30, 31] make use of systolic arrays in their implementation. This requires careful scheduling to maximize component utilization. The former work makes use of the Complex LLL algorithm whereas the later extends the LLL algorithm through the use of the Siegel condition to avoid the requirement for division operations.

The field-programmable gate array (FPGA) implementation of [32] implements the Clarkson's algorithm variant of LLL [33]. However this implementation only considers slower off-the-shelf FPGA components, including the use of square root and division operations that have not been optimized. The FPGA and application-specific integrated circuit (ASIC) implementation [34] claims to achieve a "fivefold improvement in terms of throughput at the cost of only slightly more FPGA resources" over [26] and [32]. This work uses CORDIC units along with a modification of the LLL algorithm by replacing the size-reduction criterion with the reverse Siegel condition. The hard output performance of this implementation is also enhanced by the use of soft interference cancellation (SIC), which requires the use of the sorted QR decomposition.

http://dx.doi.org/10.5772/51649

6.2. Architecture for Subspace Lattice Reduction Aided Detection

Our proposed architecture implements a soft-output lattice reduction-aided detector based on the subspace LRAD (SLRAD) approach of Windpassinger [3, 4]. The top-level schematic layout is shown in Fig. 8. A key feature of the detector is the separation of channel and data processing sections, shown above and below the dashed line in Fig. 8, respectively. Channel processing is computationally expensive, and includes the decomposition and lattice reduction of the MIMO channel matrix $\mathbf{H}$. The separation of channel and data processing therefore enables the receiver to exploit the typically slow variation in channel gains relative to the symbol rate, whereby the output of the computationally expensive channel processing step is used is used in processing the data spanning multiple data frames.

The channel processing section in Fig. 8 is fed with elements h_{in} of the estimated MIMO channel matrix $\mathbf{H}$ generated by an external MIMO channel estimator (not shown), while channel multiply and accumulator (CMAC) units perform rotations under control of the Givens control unit. Data processing involves the subspace-based detection of incoming received values, in addition to the calculation of soft outputs in the form of log-likelihood ratio (LLR) values. The data processing section is fed elements of the received vector $\mathbf{y}$, scaled by automatic gain control (AGC) to ensure that analog-to-digital converters (ADCs) are not saturated, and therefore that fixed-point inputs are within a defined range. The data multiply and accumulate (DMAC) and detection (DET) blocks in Fig. 8 are described in Section 6.4. The outputs of the data processing section are LLR values for the bits corresponding to each vector of transmitted symbols $\mathbf{x}$.

Unlike [26, 27], this work implements the Scaled and Decoupled QR (SDQR) Decomposition [35]. The use of the SDQR provides a definitive bound on the required integer precision and allows the number of fractional bits to be varied with a constant and small number of integer bits.

6.3. Channel processing

6.3.1. Givens Control Unit

The calculation of the SDQR rotation values is performed by a Givens Control unit. This unit is a single cycle processor which generates the Givens rotation $\mathbf{G}$ which zeros the element $\mathbf{P}_{j,i}$ by rotating the j^{th} row with the i^{th} row of $\mathbf{P}$ and $\mathbf{\Phi}$. The Givens Control unit is capable of a throughput of one rotation variable per cycle by calculating a Givens rotation every four cycles. Two rotation variables are emitted in the third cycle (values $\mathbf{G}_{1,1}$ and $\mathbf{G}_{1,2}$) and fourth cycle (values $\mathbf{G}_{2,1}$ and $\mathbf{G}_{2,2}$) of each Givens rotation calculation. The Givens Control Unit also maintains the decoupled k values and also dynamically scales $\mathbf{G}$ to maintain scaling of not only k but indirectly $\mathbf{P}$ and the rotated $\mathbf{y}$. This processor implements the reciprocal function required for division through the use of Newton–Raphson iterations.

6.3.2. Channel MAC (CMAC) Unit

The application of rotation operations are performed by processor units referred to as Channel Multiply and Accumulator (CMAC) units. Each CMAC unit includes sufficient register space to store a full column of the MIMO channel $\mathbf{H}$ as well as necessary intermediate values. All input, output and stored register values are complex numbers specified using

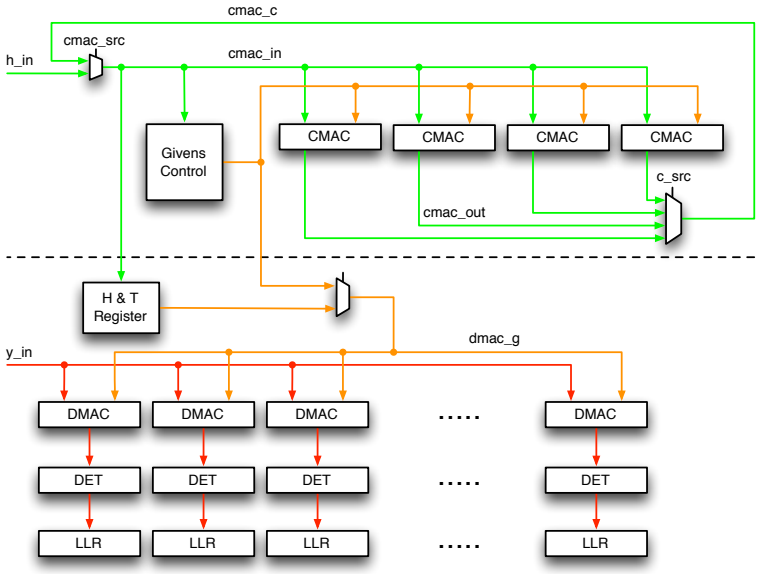

Figure 8. Top-level schematic layout. Channel and data processing sections are shown above and below the dashed line, respectively

custom extensions to the VHDL fixed point math package. Arithmetic implemented includes a complex multiplier and complex addition unit with the output of the multiplier being one of the operands of the adder, as shown in Fig. 9.

Whilst this architecture greatly simplifies the challenge of processor unit scheduling, the units are still unavoidably under-utilized. The CMAC units become unused once the their corresponding column of $\mathbf{H}$ is fully zeroed. As a result, the CMAC unit corresponding to the i^{th} column of $\mathbf{H}$ is in use for i/n_{T} of the SDQR execution period.

The CMAC units provide outputs which feed a multiplexer, as shown in Fig. 8. Required in order to perform back substitution, this allows the transfer of register values between CMAC units by feeding the output of a unit to the input of another.

6.4. Data processing

6.4.1. Data MAC (DMAC) Unit

Processor units referred to as Data Multiply and Accumulator (DMAC) units, are implemented to apply Givens rotation operations on the received vector $\mathbf{y}$. Each DMAC unit includes sufficient register space to store a full vector of received values $\mathbf{y}$ as well as necessary intermediate values.

Multiple DMAC units are implemented so that the necessary rotations required to apply a Givens rotation to a full row of $\mathbf{H}$ can be performed in parallel. This avoids the need to stall not only the Givens Control unit but also stalling of DMAC units that would otherwise need to occur whilst each row element of $\mathbf{H}$ is rotated.

http://dx.doi.org/10.5772/51649

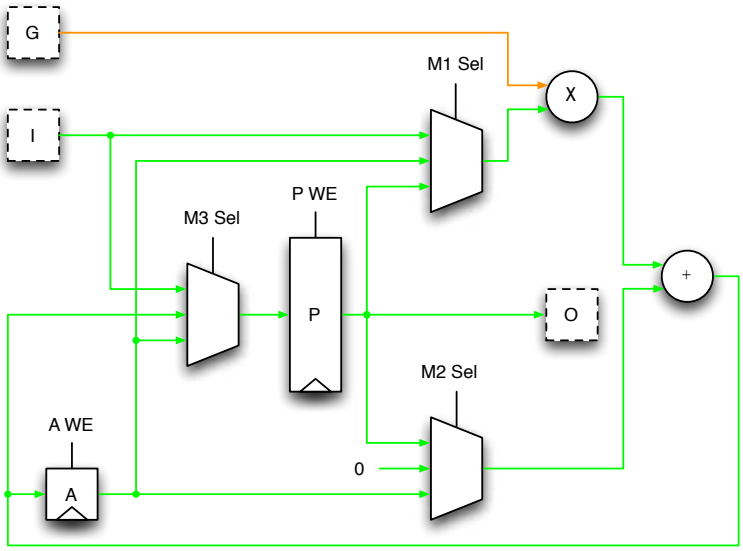

Figure 9. Custom Multiply and Accumulate Schematic Layout

Multiple DMAC units are implemented to achieve the necessary data throughput rate such that a single rotation operation can be applied to multiple received vectors in parallel. This builds on the presumption that the MIMO channel is approximately constant for multiple symbol periods. Given a sufficiently static MIMO channel, any number of DMAC units can be implemented. This allows a linear scaling of data throughput by simply adding more DMAC units, a key design feature of the the proposed architecture.

6.4.2. H&T Register File

As well as being loaded into CMAC units, when a new **H** is loaded into the processor, it is cached in the H&T register file. This is done to provide a copy of **H** for use when calculating the Euclidean distance of candidate estimates. The H&T register file is also used to store **T**, the lattice basis required to translate candidate estimates from the reduced basis prior to demodulation.

6.4.3. Candidate Detection (DET)

Each DMAC unit feeds a symbol detection chain which performs candidate generation and finally bitwise log-likelihood accumulation. This implements the data flow detailed in Fig. 7.

6.4.4. Log-likelihood ratio (LLR) Accumulator

Once a list of vectors of transmit symbol candidates has been generated, the probability of each of these vectors needs to be generated. Many approaches exist that avoid the need to implement the required log operations inherit in the calculation of log-likelihood ratio (LLR)

values. We implement the shifting method Log-MAP algorithm presented in [36], which utilizes the following piecewise linear approximation:

$$f(x) = \begin{cases} 0.70 - x/2 & 0.00 \leq x < 0.51 \\ 0.57 - x/4 & 0.51 \leq x < 1.44 \\ 0.39 - x/8 & 1.44 \leq x < 2.88 \\ 0.03 & 2.88 \leq x < 4.00 \\ 0.00 & 4.00 \leq x \end{cases} \tag{22}$$

The schematic for the LLR block is shown in Fig. 10.

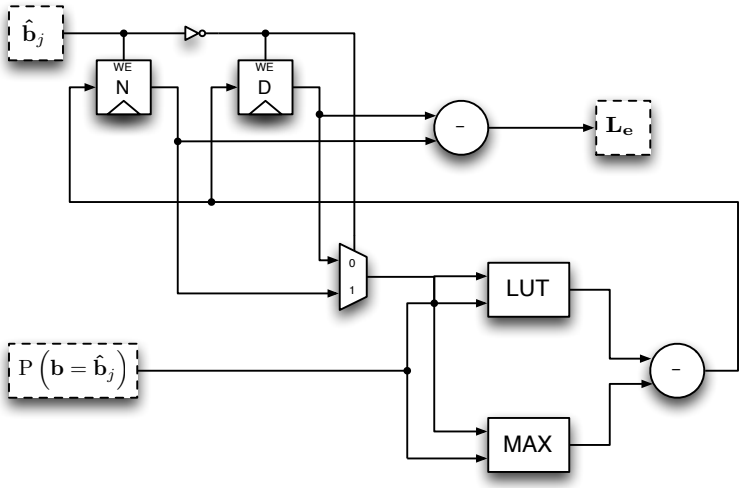

Figure 10. Log-likelihood ratio (LLR) Marginalization Schematic Layout

6.5. Processor instruction set

The overall architecture is a microcode-based system with detailed low level micro-operations that combine to implement higher level complex machine instructions. Each component including the Givens Control Unit, CMACs, DMACs and Detection Chains have their own micro-operations. The benefit is the provision of a flexible architecture capable of implementing the SLRAD algorithm, but which is also able to switch to simpler LRAD or even ZF algorithms based on the prevailing channel conditions.

6.5.1. Control Unit Micro-operations

The bulk of the channel processing involves the execution of the four operations that generate the Givens rotation $\mathbf{G}$. The first two, C1 and C2, calculate the new values for $\mathbf{P}_{j,i}$ and k_j;

http://dx.doi.org/10.5772/51649

the third C3 updates k_i and calculates $\mathbf{G}_{2,1}$ and $\mathbf{G}_{2,2}$; and the fourth C4 updates $\mathbf{P}_{i,i}$ and calculate $\mathbf{G}_{1,1}$ and $\mathbf{G}_{1,2}$. The control unit also includes an operation CR which performs the reciprocation the value k_j as needed. This supports back substitution as well as part of the LLL algorithm and is implemented using the Newton–Raphson algorithm. Finally, the control unit also performs other operations to marshal data between channel processing and data processing.

6.5.2. CMAC and DMAC Micro-operations

For the CMAC and DMAC units, the micro-operations, and their corresponding complex machine instructions are detailed in Table 1. In this table, the first column lists the micro-operation code, the third column lists the value provided on the input I of the CMAC and DMAC units and the final two columns detail the implemented function.

	Description	I	CMAC	DMAC
			Data Load	
LC	Load elements of $\mathbf{H}$	$\mathbf{H}_{m,n}$	$\mathbf{P}_{m,n} = I$	-
LD	Load elements of $\mathbf{y}$	$\mathbf{y}_x$	-	$\mathbf{\Phi}_x = I$
			Givens Rotation for SDQR and Basis Vector Swap Update	
G1	Multiply by $\mathbf{G}_{2,1}$	-	$A = G \times \mathbf{P}_i + 0$	$A = G \times \mathbf{\Phi}_i + 0$
G2	Multiply by $\mathbf{G}_{2,2}$ and add	-	$A = G \times \mathbf{P}_j + A$	$A = G \times \mathbf{\Phi}_j + A$
G3	Multiply by $\mathbf{G}_{1,2}$	-	$A = G \times \mathbf{P}_j + 0; \mathbf{P}_j = A$	$A = G \times \mathbf{\Phi}_j + 0; \mathbf{\Phi}_j = A$
G4	Multiply by $\mathbf{G}_{2,2}$ and add	-	$\mathbf{P}_i = G \times \mathbf{P}_i + A$	$\mathbf{\Phi}_i = G \times \mathbf{\Phi}_i + A$
			Back Substitution	
B1	Multiply $\mathbf{\Phi}$ row by $-\mathbf{P}_{j,i}$	$-\mathbf{P}_{j,i}$	-	$A = G \times \mathbf{\Phi}_i + 0$
B2	Accumulate with j^{th} row	-	-	$\mathbf{\Phi}_j = 1 \times \mathbf{\Phi}_j + A$
			Lattice Size Reduction	
R1	$G = \text{round}\left(\mathbf{P}_{l,j}\right)$	$\mathbf{P}_{l,j}$	$O = \mathbf{P}_{l,j}$	$A = G \times \mathbf{\Phi}_j + 0$
R2	Get reduced row of $\mathbf{\Phi}$	-	-	$\mathbf{\Phi}_l = 1 \times \mathbf{\Phi}_l + A$
R3	$G = -\text{round}\left(\mathbf{P}_{l,j}\right)$	$\mathbf{P}_{x,l}$	$\mathbf{P}_{x,j} = G \times I + \mathbf{P}_{x,j}$	-

Table 1. CMAC and DMAC Instruction Set

The CMAC and DMAC units implement the micro-operations LC and LD to perform data load operations that load the channel matrix and received vector; G1 to G4 which implement the Givens rotations for not only the SDQR but also the zeroing step of LLL algorithm; B1 and B2 that performs back substitution operations; and R1 to R3 which perform the LLL column swap step.

The DMAC units require more or less the same operations as their CMAC counterparts, however, for the case of back substitution and lattice size reduction the implementation differs. For back substitution, the CMAC units must pass off-diagonal elements $\mathbf{P}_{j,i}$ to the DMAC units. For lattice size reduction, the CMAC units add an integer multiple of one column of $\mathbf{P}$ to another by iteratively executing R3 which passes elements between CMAC units where the target unit performs the multiplication and addition. On the

other hand, DMAC units are able to perform the equivalent reduction operation in the two micro-operations R1 and R2 as lattice size reduction is performed in a row-wise fashion.

6.6. Comparisons with previously published work

The results in this section represent the first known digital signal processing architecture for a soft-output lattice reduction aided MIMO detector. For this reason we are unable to provide a direct comparison of our architecture with previously published work. Nevertheless, it is still possible to compare our implementation with three state-of-the-art VLSI implementations of hard-output LRAD-based MIMO detectors [32], [26], [34].

For $n_T = n_R = 4$, the combination of the CMAC micro-operations leads to the system latency outlined in Table 2. This table assumes a MIMO system represented by an extended channel matrix, requiring the zeroing of 16 elements of $\overline{\mathbf{H}}$. The majority of these elements require 4 cycles with the exception being the final element of each column requiring a 5^{th} cycle due the the extra cycle required to compute the Newton-Raphson based reciprocal. An overhead of 12 cycles exists to load data into the processor.

For the LLL algorithm, column swap operations require 5 cycles to perform the single Givens rotation. Size reduction requires at most 3 cycles per pass over the full matrix. As with prior works, a simple strategy is used to fix the number of iterations of the LLL algorithm which caps the number of swaps and size reduction passes to 3. This yields 24 cycles per subspace or 96 cycles for the four subspaces.

Component	Latency
QR decomposition	80 cycles
Subspace Generation	30 cycles
Subspace Back-substitution	16 cycles
Subspace Lattice Reduction	96 cycles
Total for SLRAD	222 cycles

Table 2. Latency of Channel Processor

To provide context for the results in Table 2, we compare in Table 3 the latency of the proposed architecture with the latencies of three hard-output LRAD-based MIMO detectors for a 4-input, 4-output MIMO system employing QPSK modulation.

	[32]	[26]	[34]	this work
average cycles per matrix	420	130	14	222
soft outputs?	No	No	No	Yes

Table 3. Latency comparison between the proposed architecture and three state-of-the-art implementations

While the latency of the proposed architecture compares favourably with Barbero et al.'s solution [32], the significant performance penalty for generating soft outputs is apparent in comparison with the results of Gestner et al. [26] and (esp.) Bruderer et al. [34]. We caution that the results in Table 3 need to be interpreted carefully, however, since it is well known that hard-output MIMO detectors such as [32], [26] and [34] do not facilitate high-performance

http://dx.doi.org/10.5772/51649

iterative receivers involving joint detection and decoding when error-control codes such as turbo codes and LDPC codes are employed [37], [22]. The proposed approach therefore trades off increased latency for improved BER performance and the ability to readily deal with dense constellations, e.g. 64-QAM.

7. Conclusion

In this chapter we have presented the first known digital signal processing implementation of a soft-output MIMO wireless communications receiver based on lattice reduction aided detection (LRAD). Further research is needed to provide the ASIC and FPGA synthesis results needed to facilitate a comprehensive comparison with prior works providing only hard outputs.

Author details

Alan T. Murray and Steven R. Weller

School of Electrical Engineering and Computer Science, University of Newcastle, Callaghan, NSW 2308, Australia

References

[1] S.J. Johnson. *Iterative Error Correction: Turbo, Low-Density Parity-Check and Repeat-Accumulate Codes*. Cambridge University Press, 2009.

[2] D. Wübben, D. Seethaler, J. Jaldén, and G. Matz. Lattice reduction. *IEEE Signal Process. Mag.*, 28(3):70–91, May 2011. DOI: 10.1109/MSP.2010.938758.

[3] C. Windpassinger, L.H.J. Lampe, and R. Fischer. From lattice-reduction-aided detection towards maximum-likelihood detection in MIMO systems. In *Int. Conf. on Wireless and Optical Commun. (WOC'03)*, July 2003.

[4] C. Windpassinger. *Detection and Precoding for Multiple Input Multiple Output Channels*. PhD thesis, Universität Erlangen-Nürnberg, 2004.

[5] A.T. Murray and S.R. Weller. Performance and complexity of adaptive lattice reduction in fading channels. In *Proc. Australian Comms. Workshop (AusCTW'09)*, pages 17–22, Sydney, Australia, February 2009. DOI: 10.1109/AUSCTW.2009.4805593.

[6] W. Liu, K.Choi, and H.Liu. Computationally efficient lattice reduction for MIMO-OFDM systems. In *Proc. 6th IEEE Int. Conf. on Wireless and Mobile Computing, Networking and Communications (WiMob'10)*, pages 264–267, October 2010. DOI: 10.1109/WIMOB.2010.5645056.

[7] G.H. Hardy, E.W. Wright, and J.H. Silverman. *An Introduction to the Theory of Numbers*. Oxford University Press, 6th edition, 2008.

[8] A.K. Lenstra, H.W. Lenstra, and L. Lovász. Factoring polynomials with rational coefficients. *Math. Ann.*, 261(4):515–534, 1982.

[9] P. Silvola, K. Hooli, and M. Juntti. Suboptimal soft-output MAP detector with lattice reduction. *IEEE Sig. Proc. Letters*, 13(6):321–324, June 2006. DOI: 10.1109/LSP.2006.871726.

[10] Y.H. Gan and W.H. Mow. Complex lattice reduction algorithms for low-complexity MIMO detection. In *Proc. IEEE Global Telecommunications Conf. (GLOBECOM '05)*, pages 2953–2957, St. Louis, MO, 28 November–2 December 2005. DOI: 10.1109/GLOCOM.2005.1578299.

[11] W.H. Mow. Universal lattice decoding: Principles and recent advances. *Wirel. Commun. Mob. Com.*, 3(5):553–569, August 2003. DOI: 10.1002/wcm.140.

[12] F.T. Luk and S. Qiao. Conditioning properties of the LLL algorithm. In M.S. Schmalz, G.X. Ritter, J. Barrera, J.T. Astola, and F.T. Luk, editors, *Mathematics for Signal and Information Processing*, August 2009.

[13] A. Korkine and G. Zolotarev. Sur les formes quadratiques. *Math. Ann.*, 6:366–389, 1873.

[14] M. Seysen. Simultaneous reduction of a lattice basis and its reciprocal basis. *Combinatorica*, 13(3):363–376, September 1993. DOI: 10.1007/BF01202355.

[15] C. Windpassinger, L.H.J. Lampe, R. Fischer, and T. Hehn. A performance study of MIMO detectors. *IEEE Trans. Wireless Commun.*, 5(8):2004–2008, August 2006. DOI: 10.1109/TWC.2006.1687712.

[16] X. Ma and W. Zhang. Performance analysis for MIMO systems with lattice-reduction aided linear equalization. *IEEE Trans. Commun.*, 56(2):309–318, February 2008. DOI: 10.1109/TCOMM.2008.060372.

[17] X. Li and Z. Nie. Performance losses in V-BLAST due to correlation. *IEEE Antennas Wireless Propag. Lett.*, 3(1):291–294, January 2004. DOI: 10.1109/LAWP.2004.838813.

[18] B. Hassibi. An efficient square-root algorithm for BLAST. In *Proc. IEEE Int. Conf. on Acoustics, Speech, and Signal Processing (ICASSP '00)*, volume 2, pages 737–740, 2000. 10.1109/ICASSP.2000.859065.

[19] M.R. McKay, I.B. Collings, and A.M. Tulino. Achievable sum rate of MIMO MMSE receivers: A general analytic framework. *IEEE Trans. Inform. Theory*, 56(1):396–410, January 2010. DOI: 10.1109/TIT.2009.2034893.

[20] H. Cohen. *A Course in Computational Algebraic Number Theory*. Springer, December 1993.

[21] E. Agrell, T. Eriksson, A. Vardy, and K. Zeger. Closest point search in lattices. *IEEE Trans. Inform. Theory*, 48(8):2201–2214, August 2002. DOI: 10.1109/TIT.2002.800499.

[22] W. Zhang X. Ma. Low-complexity soft-output decoding with lattice-reduction-aided detectors. *IEEE Trans. Commun.*, 58(9):2621–2629, September 2010. DOI: 10.1109/TCOMM.2010.080310.070641.

http://dx.doi.org/10.5772/51649

[23] A. Burg, D. Seethaler, and G. Matz. VLSI implementation of a lattice-reduction algorithm for multi-antenna broadcast precoding. In *Proc. IEEE Int. Symp. on Circuits and Systems (ISCAS'07)*, pages 673–676, New Orleans, LA, 27–30 May 2007. DOI: 10.1109/ISCAS.2007.377898.

[24] D. Seethaler and G. Matz. Efficient vector perturbation in multi-antenna multi-user systems based on approximate integer relations. In *Proc. European Signal Proc. Conf. (EUSIPCO'06)*, pages 4–8, Florence, Italy, 4–8 September 2006.

[25] W. Zhang, X. Ma, B. Gestner, and D.V. Anderson. Designing low-complexity equalizers for wireless systems. *IEEE Comms. Mag.*, 47(1):56–62, January 2009. DOI: 10.1109/MCOM.2009.4752677.

[26] B. Gestner, W. Zhang, X. Mai, and D.V. Anderson. VLSI implementation of a lattice reduction algorithm for low-complexity equalization. In *IEEE Int. Conf. on Circuits and Systems for Communications (ICCSC'08)*, pages 643–647, Shanghai, China, 26–28 May 2008. DOI: 10.1109/ICCSC.2008.142.

[27] W. Zhang. *Wireless Receiver Designs: From Information Theory to VLSI Implementation*. PhD thesis, Georgia Institute of Technology, December 2009.

[28] W. Zhang and X. Ma. Approaching optimal performance by lattice-reduction aided soft detectors. In *Proc. 41st Annual Conf. on Information Sciences and Systems (CISS '07)*, pages 818–822, 2007. DOI: 10.1109/CISS.2007.4298422.

[29] B. Gestner, W. Zhang, X. Ma, and D.V. Anderson. VLSI implementation of an effective lattice reduction algorithm with fixed-point considerations. In *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP 2009)*, pages 577–580, 2009. DOI: 10.1109/ICASSP.2009.4959649.

[30] J. Soler-Garrido, H. Vetter, M. Sandell, D. Milford, and A. Lillie. Implementation of a reduced-lattice MIMO detector for OFDM systems. In *Design, Automation & Test in Europe Conference & Exhibition (DATE '09)*, pages 1626–1631, 2009.

[31] N.-C. Wang, E. Biglieri, and K. Yao. Systolic arrays for lattice-reduction-aided MIMO detection. *J. Commun. Netw.*, 13(5):481–493, October 2011. DOI: 10.1109/JCN.2011.6112305.

[32] L.G. Barbero, D.L. Milliner, T. Ratnarajah, J.R. Barry, and C. Cowan. Rapid prototyping of Clarkson's lattice reduction for MIMO detection. In *Proc. IEEE Int. Conf. on Communications (ICC'09)*, pages 1–5, Dresden, Germany, 14–18 June 2009. DOI: 10.1109/ICC.2009.5199388.

[33] I.V.L. Clarkson. *Approximation of Linear Forms by Lattice Points with Applications to Signal Processing*. PhD thesis, Australian National University, January 1997.

[34] L. Bruderer, C. Studer, M. Wenk, D. Seethaler, and A. Burg. VLSI implementation of a low-complexity LLL lattice reduction algorithm for MIMO detection. In *Proc. IEEE Int. Symp. on Circuits and Systems (ISCAS'10)*, pages 3745–3748, Paris, France, 30 May–2 June 2010. DOI: 10.1109/ISCAS.2010.5537742.

[35] L.M. Davis. Scaled and decoupled Cholesky and QR decompositions with application to spherical MIMO detection. In *Proc. IEEE Wireless Communications and Networking (WCNC'2003)*, volume 1, pages 326–331, New Orleans, LA, 16–20 March 2003. DOI: 10.1109/WCNC.2003.1200369.

[36] L. Zhong, M. Gang, T. Yi-Zheng, and C. Yan-Min. A simplification of the log-MAP algorithm for turbo decoding. In *Proc. IEEE Asia-Pacific Conference on Circuits and Systems*, volume 2, pages 1057–1060, 2004. DOI: 10.1109/APCCAS.2004.1413065.

[37] D.L. Milliner and J.R. Barry. A lattice-reduction-aided soft detector for multiple-input multiple-output channels. In *Proc. IEEE Global Telecommunications Conference (GLOBECOM '06)*, 2006. DOI: 10.1109/GLOCOM.2006.84.