# Evaluation of the Complexity, Performance and Implementability of Geometrical MIMO Detectors: the Example of the Exploration and Exploitation List Detector

Bastien Trotobas*, Youness Akourim†, Amor Nafkha*, Yves Louët* and Jacques Weiss‡
*SCEE/IETR UMR CNRS 6164, CentraleSupélec
Avenue de la boulaie, 35576 Cesson Sévigné, France
Email: bastien.trotobas@centralesupelec.fr
amor.nafkha@centralesupelec.fr
yves.louet@centralesupelec.fr
†École normale supérieure de Rennes
Campus de Ker Lann, Avenue Robert Schuman, 35170 Bruz, France
Email: youness.akourim@ens-rennes.fr
‡CentraleSupélec, Avenue de la boulaie, 35576 Cesson Sévigné, France
Email: jacques.weiss@centralesupelec.fr

***Abstract*—This paper develops a new paradigm for the multi-input multi-output detection problem with bit interleaved coded modulation (MIMO-BICM). This new paradigm is based on a geometric method rather than the traditional interference cancellation or tree search. It describes in greater detail the soft-output detector called list exploration and exploitation (L2E), which builds a list of candidates from a geometrical interpretation of a given objective function. It then computes the log-likelihood ratios (LLRs) using the max-log approximation. A comparative study between L2E and a classical tree-based algorithm is carried out in both computational complexity and detection performance. This study highlights that although the framework is entirely different, the complexity and performance are comparable with the state of the art tree-based paradigm. Finally, the proposed algorithm is implemented on a field-programmable gate array (FPGA) device. Simulations carried out with Xilinx Vivado tools and measurements are provided to analyze the resource utilization, power consumption, and timing metrics. We further estimate these metrics for an application-specific integrated circuit (ASIC) implementation based on multiplicative factors from literature. This projection demonstrates that our implementation yields results of the same magnitude as the well-known detectors.**

***Keywords–Geometrical algorithm; MIMO detection; Complexity analysis; BER performance; FPGA; Max-log approximation***

## I. INTRODUCTION

This paper carries on the work initiated in [1], where we described the interest of geometric detectors. It takes place in the context of MIMO systems that have been integrated into many standards, such as IEEE 802.11n [2] or long-term evolution (LTE). It is well known that increasing the number of antennas improves the data rate and the link reliability [3], but at the same time increases the complexity of detection at the receiver side. Detection problems can be classified into three types: over-determined MIMO, iso-determined MIMO, and under-determined MIMO.

Over-determined problems occur in particular in the massive MIMO uplink case, where a base station is equipped with a very large number of receiving antennas to detect messages from a small number of users. Under the hypothesis of a perfectly-known channel, the problem is very easily solved with linear detectors such as zero-forcing (ZF) or minimum mean square error (MMSE). Linear detectors require the inversion of the channel matrix, which is the most complex step. Current research offers binary error rate (BER) efficient detectors with even less algorithmic complexity. These detectors approximate the MMSE criteria or the channel matrix inversion by conjugate gradient least square [4], symmetric successive over relaxation [5], Neumann series expansion [6] or using the Jacobi method [7].

For iso-determined systems, linear detectors cannot provide acceptable BER performance because there is just enough information available to retrieve the transmitted signal. In this situation, the maximum likelihood (ML) detector offers an optimal result at the cost of very high complexity. This complexity prevents its use for real-time hardware implementations, which is why simpler sub-optimal detectors are required. The earliest proposals were based on iterative detection using successive suppression of interference between antennas [8]. They led to the successive interference cancellation (SIC) and ordered successive interference cancellation (OSIC) detectors [9], [10] that are used nowadays. Parallel interference cancellation (PIC) is another variant of this paradigm [11], [12]. Subsequently, detectors based on tree-search became the reference with three variations: the breadth-first tree-search [13]–[16], the depth-first tree-search as the sphere decoding (SD) [17], [18] and the best-first or metric-first [19]. More exotic approaches have also been proposed, such as geometric detection [20], deep neural network detection [21] or bio-inspired algorithms [22].

Soft-output detectors are known to be more efficient than hard-output ones [23] because they exploit the uncertainty that a '0' or a '1' is being received to provide an LLR on the received bits. This probability is then used in conjunction with an error-correcting code to improve BER performance. Finding exactly the a-posterior probabilities is known to be exponentially complex with respect to the number of transmitting antennas [24]. That is why the reference algorithms approximate the LLRs using the max-log approximation [23], [25], [26]. Thus,the large majority of soft-output MIMO detectors construct a list of possible candidates and then use this approximation to produce approached values of LLRs. Examples include the list sphere decoding [23] or the soft version of the breadth-first M-algorithm [13], [15].

In the last years, efforts have been made to propose very large scale integration (VLSI) implementations aiming for high throughput, low energy consumption, and high flexibility. Different architectures have been explored: arrays of reconfigurable units [5], [27]; pipelines with parallel units [16], [28]; systolic arrays [6] and multiple cores managed by a task scheduler [18].

In this paper, we detail more precisely the L2E algorithm [1], which builds a list from geometric considerations rather than from tree paths. In particular, we present some algorithmic improvements and compare the complexity to the one of tree-based detectors (i.e., detectors with quasi-optimal performance). We also present initial investigations on the hardware implementability of geometric algorithms. To do so, we provide the architecture of the corresponding hard-output detector and highlight the intrinsic parallel structure of this algorithm. Finally, we present Monte-Carlo simulations to compare L2E performance with the best-known detectors.

In Section II, we present the mathematical system model. Then we describe the L2E and K-best Schnorr-Euchner (KSE) algorithms in Section III. Subsequently, Section IV compares the two algorithms and Section V presents an FPGA-based hardware implementation. Finally, Section VI concludes this paper.

## II. Mathematical model

In this paper, bold lower cases (*resp.* capital letters) denote vectors (*resp.* matrices) and other characters refer to scalars. We call $\mathbf{v}(i)$ the $i^{\text{th}}$ coefficient of the vector $\mathbf{v}$ and $\mathbf{A}(i,j)$ the coefficient on the $i^{\text{th}}$ row and the $j^{\text{th}}$ column of $\mathbf{A}$. In this section, we present the modeling of the MIMO-BICM system, detail the LLRs calculations and discuss on the advantages of the QR decomposition.

### A. MIMO-BICM system model

We consider a quadrature phase-shift keying (QPSK), and we note $\phi$ the set of possible symbols. Let $\mathbf{y}_c \in \mathbb{C}^M$ be the vector containing the signal received on the $M$ reception antennas. These signals are the result of the $N$ transmitted symbols $\mathbf{x}_c \in \phi^N$ after passing through the channel described by the matrix $\mathbf{H}_c \in \mathbb{C}^{M\times N}$ and after adding a Gaussian noise $\mathbf{w}_c \in \mathbb{C}^M$. With these notations, the transmission system is modeled by

$$\mathbf{y}_c = \mathbf{H}_c\mathbf{x}_c + \mathbf{w}_c. \tag{1}$$

For the sake of implementation, we propose a real-valued version of the previous model using $\Re$ (*resp.* $\Im$) the real (*resp.* imaginary) part operator. We introduce the real-valued equivalent of the previous vectors, matrix and set:

$$\mathbf{y} \triangleq \begin{bmatrix}\Re(\mathbf{y}_c)\\ \Im(\mathbf{y}_c)\end{bmatrix} \tag{2}$$

$$\mathbf{H} \triangleq \begin{bmatrix}\Re(\mathbf{H}_c) & -\Im(\mathbf{H}_c)\\ \Im(\mathbf{H}_c) & \Re(\mathbf{H}_c)\end{bmatrix} \tag{3}$$

$$\mathbf{x} \triangleq \begin{bmatrix}\Re(\mathbf{x}_c)\\ \Im(\mathbf{x}_c)\end{bmatrix} \tag{4}$$

$$\mathbf{w} \triangleq \begin{bmatrix}\Re(\mathbf{w}_c)\\ \Im(\mathbf{w}_c)\end{bmatrix} \tag{5}$$

$$\xi \triangleq \Re(\phi) \tag{6}$$

The MIMO-BICM model can now be expressed as

$$\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{w}. \tag{7}$$

This real-model representation of (1) has exactly the same amount of information as the complex-valued version and we define $n \triangleq 2N$, $m \triangleq 2M$. Indeed, it is like treating the real and imaginary axis independently. In the remainder, we assume $n \leq m$: we have at least as many receiving antennas as transmitting one.

This model can be used to describe any iso-determined or over-determined MIMO-BICM system. On transmission, the message is coded by an encoder, interleaved with an assumed perfect interleaver, and then mapped to symbols. On reception, the data flow is processed in the opposite way: it goes through a detector, a de-interleaver, and then a decoder. A perfect channel state information is assumed only at the reception, and no information at all is known at the transmitter.

### B. LLRs generation

Let us consider $b_{ij}$ the $i^{\text{th}}$ bit of information coded in the $j^{\text{th}}$ component of $\mathbf{x}$. To minimize the BER, a soft detector must maximize the a posteriori probability that is calculated from the probability mass function $P(b_{ij}|(\mathbf{H},\mathbf{y}))$ of the bit $b_{ij}$ given the state of the channel $\mathbf{H}$ and the received vector $\mathbf{y}$. The LLR is defined as

$$L_{ij} = \ln\frac{P(b_{ij}=1|(\mathbf{H},\mathbf{y}))}{P(b_{ij}=0|(\mathbf{H},\mathbf{y}))}. \tag{8}$$

The closed-form evaluation is known to be exponentially complex with respect to the dimensions of the MIMO system [24]. This is why we use the classic max-log approximation [23], [25], [26]

$$L_{ij} \approx \frac{1}{2\sigma^2}\left(\min_{x\in\chi_{ij}^0}\|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2 - \min_{x\in\chi_{ij}^1}\|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2\right) \tag{9}$$

where $\chi_{ij}^k = \{x \in \xi^n : b_{ij} = k\}$ is the set of all symbols with $b_{ij}$ equals to $k$, and $\sigma^2$ is the noise variance. Although simpler, this new expression remains exponentially complex since we have $\mathrm{card}(\chi_{ij}^k) = 2^{n-1}$ where $\mathrm{card}()$ denotes the cardinality of a set. therefore,we introduce a new subset $\Gamma \subset \xi^n$ representing a list of candidates and we compute LLRs only from this subset:

$$L_{ij} \approx \frac{1}{2\sigma^2}\left(\min_{x\in\Gamma\cap\chi_{ij}^0}\|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2 - \min_{x\in\Gamma\cap\chi_{ij}^1}\|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2\right). \tag{10}$$

If $\Gamma \cap \chi_{ij}^k = \emptyset$, we consider that $b_{ij}$ is different from $k$ for sure and set $L_{ij}$ to its maximum or minimum value in order to reflect this certainty.

The algorithmic cost to approximate LLRs is now fully controlled by $\mathrm{card}(\Gamma)$: the more points in $\Gamma$, the better the approximation, but at the same time the more difficult the computation becomes. The L2E algorithm constructs a set $\Gamma$ whose cardinal grows linearly with $n$ and is independent with respect to the signal-to-noise ratio (SNR) values. In the remainder of this paper, we refer to $\|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2$ as the objective

function and the best points of a set refer to the points that minimize this $\ell_2$-norm.

### C. QR decomposition

QR decomposition is a well-known technique to simplify the norm evaluation. Let be $\mathbf{H} = \mathbf{QR}$ the QR decomposition of the channel matrix with $\mathbf{Q}$ an orthogonal matrix and $\mathbf{R}$ an upper triangular matrix. Then we have:

$$\|\mathbf{y} - \mathbf{Hx}\|_2 = \|\mathbf{y} - \mathbf{QRx}\|_2 = \|\mathbf{Q}^T\mathbf{y} - \mathbf{Q}^T\mathbf{QRx}\|_2 \quad (11)$$

as orthogonal matrices act as isometries. Thus, by exploiting the property of orthogonal matrices $\mathbf{Q}^T = \mathbf{Q}^{-1}$ this norm can be rewritten as

$$\|\mathbf{y} - \mathbf{Hx}\|_2 = \|\tilde{\mathbf{y}} - \mathbf{Rx}\|_2 \quad (12)$$

with $\tilde{\mathbf{y}} \triangleq \mathbf{Q}^T\mathbf{y}$.

We recognize the same pattern so that an algorithm able to solve the full matrix version can solve the QR-based version in a similar way. Because of this mathematical similarity, we will continue in the following to use the full-matrix notation. All the results obtained for the full-matrix version are directly applicable in the triangular case. Thus, unless otherwise stated, the following developments are valid in both cases, even if only the full-matrix case is described.

## III. Algorithmic interpretation

In this section, we present an algorithm to obtain a set with a reduced cardinality without compromising decoding properties. The L2E detector is based on two steps: exploration and exploitation, which will be detailed in the first two subsections. Next, we will present a classical tree-based algorithm that will be used as a reference in the following.

### A. Exploration step

The exploration phase has two main objectives: to begin to create the $\Gamma$ set for the LLRs' evaluations and to build a set of promising points $\Gamma_b$ that will be processed by the exploitation step.

To achieve these objectives, we first rewrite the objective function by introducing the singular value decomposition (SVD) of $\mathbf{H} = \mathbf{UDV}^T$ with $\mathbf{U}$ and $\mathbf{V}$ two orthogonal matrices and $\mathbf{D}$ the diagonal matrix containing the singular values $\{\lambda_i : 1 \leqslant i \leqslant n\}$. The order of these values being arbitrary, we choose to number them in ascending order.

We introduce the starting point define by $\mathbf{x}_0 = \mathbf{H}^{-L}\,\mathbf{y}$ where $\mathbf{H}^{-L}$ denotes the left Moore-Penrose inverse of $\mathbf{H}$. Consequently, the objective function can be rewritten as

$$\|\mathbf{y} - \mathbf{Hx}\|_2^2 = \left(\mathbf{V}^T(\mathbf{x} - \mathbf{x}_0)\right)^T \mathbf{DU}^T\mathbf{UD}\left(\mathbf{V}^T(\mathbf{x} - \mathbf{x}_0)\right). \quad (13)$$

As the vectors of $\mathbf{V}$, named $\{\mathbf{v}_i : 1 \leqslant i \leqslant n\}$, constitute a basis, we can define $\{\alpha_i : 1 \leqslant i \leqslant n\}$ the coordinates of $\mathbf{x} - \mathbf{x}_0$ on this basis. Using the orthogonality of $\mathbf{U}$ and $\mathbf{V}$ and the diagonality of $\mathbf{D}$, equation (13) leads to

$$\|\mathbf{y} - \mathbf{Hx}\|_2^2 = \sum_{i=1}^{n} \alpha_i^2\lambda_i^2. \quad (14)$$

Let $\Delta_i$ be the straight line passing through $\mathbf{x}_0$ and directed by $\mathbf{v}_i$. One can note that (14) highlights that the objective function grows more slowly along the first $\Delta_i$ rather than along the last ones. Thus, promising points are more likely to be found around these first straight lines. Figure 1 illustrates the previous reasoning in the simplified case of a space of dimension $n = m = 2$. The black ellipses represent the evolution of the objective function along the two positions of $\mathbf{x}$. The objective function minimum is reached in $\mathbf{x}_0$. The two singular vectors $\mathbf{v}_i$ and the corresponding lines $\Delta_i$ are also plotted. We can see that the objective function grows more slowly along $\Delta_1$ than along $\Delta_2$. In this example, the solution that minimize the objective function is the point $\mathbf{x} = (-1, 1)$. It is confirmed that this solution is located close to the line $\Delta_1$.

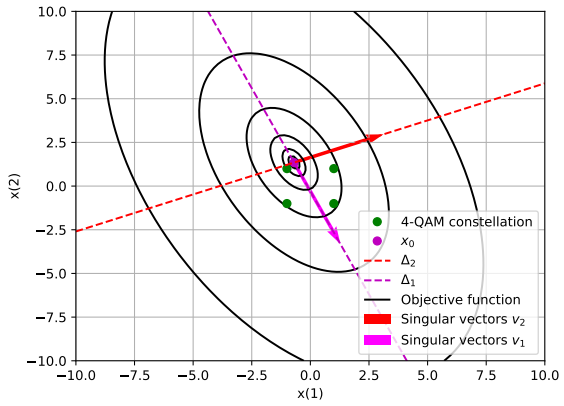



Figure 1. Link between the objective function and the SVD of $\mathbf{H}$

There are several methods for choosing points near a line. To reduce the complexity, we choose to compute the intersection of the first straight lines with each hyperplane defined by $\mathcal{H}_i = \{\mathbf{x} \in \mathbb{R}^n : \mathbf{x}(i) = 0\}$ and then to search in $\xi^n$ for the nearest point to each intersections. Figure 2 illustrates, in the same situation as for Figure 1, the process of obtaining points from intersections. The points are obtained by a two-step process. First, the algorithm computes all the intersections between the line $\Delta_1$ with the brown hyperplanes $\mathcal{H}_i$. Then, it searches for the nearest point in the constellation. When two points are equidistant, we take the closest to $x_0$.

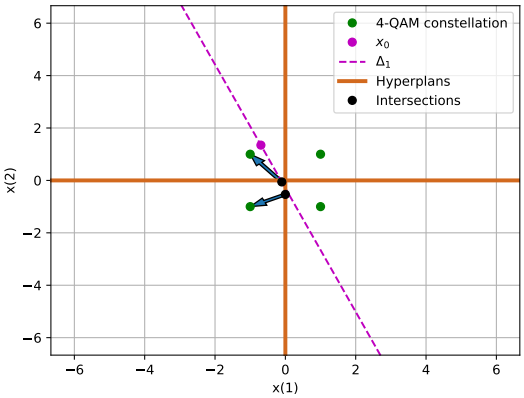



Figure 2. Getting promising points from a line $\Delta_i$

The set $\Gamma_b$ is obtained by picking the best $C$ points obtained for each line $\Delta$ considered. Thus,we have by construction $\mathrm{card}(\Gamma_b) = CD$. Besides, all the points that have been computed are included to start building the set $\Gamma$. This operation does not cost anything since the points, and their objective function have already been calculated to obtain $\Gamma_b$. As better results are observed with $x_0$ being the MMSE estimator, we use it in the following.

### B. Exploitation step

Exploitation step aims to enrich the already computed $\Gamma$ set without increasing too much its size (i.e., the upper-bound of the number of added points is $nCD$ per exploitation iteration). Let $\mathcal{B}_{\mathbf{x},r}$ be the closed Hamming ball of radius $r$ and centered at a point $\mathbf{x}$. To increase $\mathrm{card}(\Gamma)$, we add each points in $\mathcal{B}_{\mathbf{x},r}$ for all $\mathbf{x} \in \Gamma_b$. This process can be seen as flipping at most $r$ bits of each procpoints in $\Gamma_b$. As there are $\binom{n}{i}$ ways to change $i$ bits among $n$, the exploitation of a point requires $\sum_{i=0}^{r}\binom{n}{i}$ evaluations of the objective function. essIn order to reduce the computational complexity of the L2E detector, we fixed the value of $r$ to 1. After one iteration, L2E makes a new one if it finds a better point in $\mathcal{B}_{\mathbf{x},r}$ than the center of the ball (i.e., L2E iterates until it reaches a stable point).

After this second step, we obtain a set $\Gamma$ such that

$$\mathrm{card}(\Gamma) \leq (n_i + n)CD \tag{15}$$

where $n_i$ is the number of exploitation iterations. This result is consistent with both the linear growth in the number of antennas and the independence from the SNR. Figure 3 gives an overview of the whole L2E algorithm where we recognize the pre-processing from instruction 1 to instruction 5, the exploration phase in the first for-loop, the exploitation phase in the last for-loop and eventually the soft-output computations to obtain LLRs at instruction 18. If the QR decomposition representation is adopted, its computation should be added in the pre-processing, and $\tilde{\mathbf{y}}$ should be processed before the first for-loop.

### C. A tree-based reference: K-best Schnorr-Euchner (KSE)

In this section, we will briefly describe a tree-based algorithm that will serve as a reference later on. We select the breadth-first algorithm KSE mode 1 without maximal radius as described in [14] since it is a well-known algorithm with a very acceptable complexity and performance. This algorithm also uses the max-log approximation and QR decomposition of the channel matrix and additionally requires that diagonal coefficients of $\mathbf{R}$ to be in ascending order.

KSE first completes the QR decomposition transformation as described in Section II-C and then uses the triangularity of $\mathbf{R}$ to compute the objective function iteratively. Thus, we can introduce $\mathbf{d}(k) = \tilde{\mathbf{y}}(k) - (\mathbf{R}\mathbf{x})(k)$ such that the objective function is written as

$$\|\tilde{\mathbf{y}} - \mathbf{R}\mathbf{x}\|_2^2 = \sum_{k=1}^{n} \mathbf{d}(k)^2. \tag{16}$$

Moreover, triangularity gives for all $k \in \{1, \ldots, n\}$

$$(\mathbf{R}\mathbf{x})(k) = \sum_{j=1}^{n} \mathbf{R}(k,j)\mathbf{x}(j) = \sum_{j=k}^{n} \mathbf{R}(k,j)\mathbf{x}(j) \tag{17}$$

```
1  Extract the first D right singular vectors v_i from H
2  Compute left pseudo-inverse H^{-L}
3  while H does not change do
4      Compute starting point x_0
5      Initialize Γ with the projection of x_0 on ξ^n
6      foreach Δ in {Δ_1, ..., Δ_D} do
7          foreach H in {H_1, ..., H_n} do
8              Find the intersection of Δ with H
9              Project the intersection on ξ^n
10             Evaluate objective function
11             Add the point to Γ and save its cost
12         Add C best points to Γ_b
13     foreach x in Γ_b do
14         Add the points in B_{x,1} to Γ
15         if there is a better point in B_{x,1} than x then
16             Center a new ball at this best point
17             Make a new iteration starting at line 13
18     Compute LLRs using (10)
19     return LLRs
```

Figure 3. L2E algorithm

that leads to

$$\mathbf{d}(k) = \tilde{\mathbf{y}}(k) - \sum_{j=k}^{n} \mathbf{R}(k,j)\mathbf{x}(j). \tag{18}$$

The basic idea underlying the KSE algorithm is to compute partial estimations of the objective function and $\mathbf{d}(k)$ coefficients as the solution vector is built. Indeed, starting from the last component, it is possible to add a new operand in (16) as soon as a hypothesis is made on $\mathbf{x}(j)$ since (18) is fully evaluated for this position. Figure 4 gives a brief overview of the KSE decoding process with $\mathcal{X}$ being the set of partially constructed solutions. For the sake of readability, the indexes of the $\mathbf{d}(k)$ and the objective function relating to each partially constructed vector are omitted, but there is one vector $\mathbf{d}$ and one partial norm per solution in $\mathcal{X}$.

## IV. Comparing the two strategies

In this section, we provide a full analysis of the above algorithms in terms of hardware complexity (number of products and number of additions) and BER performance.

### A. Computational complexity

The complexity analysis should be done distinguishing between the pre-processing, which is performed once per coherence block and the decoding itself. Indeed, during a coherence block, several hundred symbol vectors can be sent [29]. That is why pre-processing does not profoundly impact the total complexity.

Concerning pre-processing, KSE requires a QR decomposition when L2E requires an SVD and the computation of a Moore-Penrose inverse. These two decompositions are known to have a cubic complexity in the largest size of the channel matrix [30]. It is noteworthy that the Moore-Penrose inverse can be obtained far more quickly based on an SVD of the

Compute QR-decomposition $\mathbf{QR} = \mathbf{H}$
**while** $\mathbf{H}$ does not change **do**
  Compute $\tilde{\mathbf{y}} = \mathbf{Q}^T\mathbf{y}$
  Initialize the partial objective function at 0
  Initialize $\mathbf{d} = \tilde{\mathbf{y}}$
  **for** $k = n$ **to** 1 **do**
    **foreach** $\boldsymbol{x}$ **in** $\mathcal{X}$ **do**
      **foreach** $s$ **in** $\xi$ **do**
        Evaluate $s\mathbf{R}(k,k)$
        Update the partial $\mathbf{d}(k)$
        Update the partial objective function
    Update $\mathcal{X}$ by keeping the best $K$ partial vectors
    Update $\mathbf{d}(j)$ for $j \in \{1,\ldots,k-1\}$
  Compute LLRs using (18)
  **return** LLRs

Figure 4. KSE algorithm

matrix. Thus, the pre-processing of L2E is faster than if the decomposition and inversion were performed independently. Finally, it should also be noted that a QR decomposition may be added to this pre-processing if this approach is adopted. To sum up, KSE has a slightly lighter pre-processing than L2E, but both have complexity in $\mathcal{O}(m^3)$ products.

Table I details the complexity of each step of the L2E decoder for a QPSK depending on the parameters $C$ and $D$ and using the tricks presented in Section V-B. The addition of the calculation of $\tilde{\mathbf{y}}$ in the QR-based setup would, in practice, reduce the number of operations required due to the triangularity of the channel matrix that speeds up the objective function evaluation. The experimental results presented in Section V-D2 support this point. As a comparison, we also evaluate KSE using a QPSK denoting by $K$ the maximum number of paths kept at each step. KSE required $4K$ additions and $4K$ products per position to compute $\mathbf{d}(k)$ and to update the partial objective function. Moreover, it also required some operations to update $\mathbf{d}(k)$, the amount depending on the processed coordinate. Indeed, (18) highlights that the fewer coordinates are left to process, and the fewer calculations are required.

TABLE I. Complexity of L2E assuming one exploitation iteration and KSE

| Step | Additions | Multiplications |
|---|---|---|
| Compute $\mathbf{x}_0$ | $nm$ | $nm$ |
| Intersections & Projections | $Dn^2$ | $Dn^2$ |
| Evaluate objective function | $Dn^2(m+1)$ | $Dn^2m$ |
| Exploitation step | $CDn(m+n+1)$ | $CDn(m+n)$ |
| Compute LLRs | 1 | 0 |
| **Asymptotic complexity L2E** | $\mathcal{O}(Dn^2m)$ | $\mathcal{O}(Dn^2m)$ |

In total, the KSE algorithm has a lighter asymptotic complexity in $\mathcal{O}(n(m+nK/2))$. However, the number of antennas in a MIMO system is not always large enough to match the asymptotic behavior. For this reason, Figure 5 shows the exact number of operations depending on the number of antennas of a square MIMO ($n = m$). We compare the two detectors on both the number of additions and products. These two numbers are very close, given that the most significant operation is the norm computations. One can note that both algorithms have an effective operating range with a cross-point around 16 antennas.

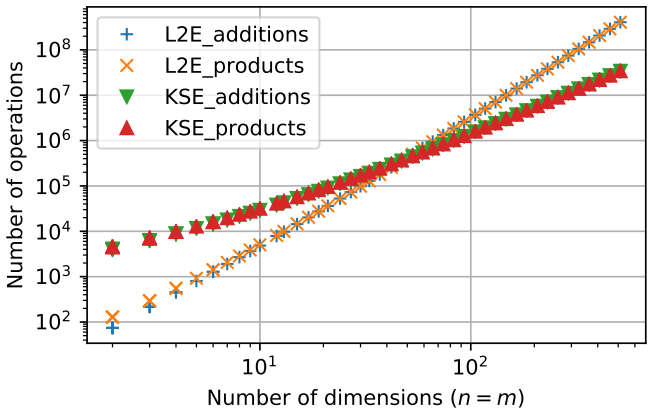



Figure 5. Number of operation of L2E ($C = 4$, $D = 3$) with soft KSE mode 1 ($K = 252$) from [14] using a QPSK.

### B. BER performance

In this section, we propose Monte-Carlo simulations to evaluate the BER for different SNR $= n\mathrm{E_s}/\mathrm{N_0}$ with $\mathrm{E_s}$ the average energy symbol and $\mathrm{N_0}$ the noise variance per entry [4]. We consider a MIMO-BICM system as modeled in Section II-A transmitting over an uncorrelated Rayleigh fading channel. The associate matrix $\mathbf{H}$ is constructed as independent and identically distributed (i.i.d.) variables following a circularly-symmetric normal distribution and then transformed to its real-valued equivalent. Similarly, $\mathbf{w}$ is elaborated with i.i.d. zero mean Gaussian noise with a variable variance to match the required SNR.

We assume a perfect interleaver so that all transmitted symbols are independent of one another. The binary data are mapped to a QPSK and encoded using a simple convolutional code of rate $1/2$ generated by the polynomials $(5, 7)$. The decoder uses the Viterbi algorithm with a traceback depth of 10. Figure 6 compares the performance of L2E using 3 singular directions and 4 candidates per directions ($C = 4$ and $D = 3$) with KSE using 252 paths in the case of a 4x4 MIMO ($n = m = 8$). One can note that the performance is strictly equivalent up to 13 dB and that L2E loses only 2 dB for a BER of $10^{-5}$ while requiring roughly six times fewer operations.

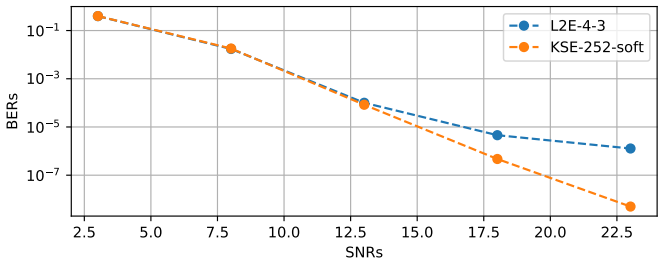



Figure 6. Performance comparison of L2E ($C = 4$, $D = 3$) with soft KSE mode 1 ($K = 252$) from [14] on a 4x4 MIMO channel using a QPSK.

## V. FPGA IMPLEMENTATION

In this section, we present an exploratory work on the implementation of geometrical algorithms on FPGA hardware platforms. Given the exploratory nature of this work, we focus our description and implementation efforts on the most specific part of the L2E detector. The hardware platform selected is the Xilinx Zynq-7000 SoC that contains a 28nm programmable logic (PL) and an ARM processing system (PS). We choose to study only the hard algorithm [20] since it involves all steps but the LLRs computation. Moving from the hard version to the soft version only requires having a minimum search and a subtraction since all the norms required to calculate (10) are already provided by the hard part of the algorithm. This part is implemented using the PL.

The calculations that are well documented in the literature are performed upstream by the PS: SVD and QR decompositions and the $\mathbf{x}_0$ calculation. The hardware architecture in PL described includes all the other stages, namely exploration and exploitation. Since this is an exploratory work, we have chosen to place parameterization and reusability before the performance, which is why we present an implementation using a data-driven pipeline where each module is independent of the others.

In the sequel, we set the $n$-dimension and the $m$-dimension to 10. We denote a fixed point quantization using the following notation: total word length is $i + f$, where $i$ is the number of integer bits, $f$ is the number of fractional bits. We summarize the fixed point design parameters for the reference solution $\mathbf{x}_0$, the directing vectors $\{\mathbf{v}_i\}_{i=1}^{D}$, the received vector $\mathbf{y}$ and the entries of channel matrix $\mathbf{R}$ in Table II.

TABLE II. Quantization of the fixed point parameters.

| | Integer length | Fractionnal length | Total length |
|---|---|---|---|
| $\mathbf{x}_0$ | 3 bits | 3 bits | 6 bits |
| $\{\mathbf{v}_i\}_{i=1}^{D}$ | 1 bit | 5 bits | 6 bits |
| $\mathbf{y}$ | 6 bits | 2 bits | 8 bits |
| $\mathbf{R}$ | 6 bits | 2 bits | 8 bits |

In this section, we will present the chosen FPGA architecture and detail the content of each module as well as the global structure based on a data-driven pipeline. Then, we give the results and compare them with the simulation and the current state-of-the-art literature. It is not straightforward to compare these results from a small FPGA with those from optimized ASICs. Some studies evaluated the performance gap between FPGAs and ASICs on general-purpose projects [31] and for convolutional neural networks (CNN) [32]. It is noteworthy that even if MIMO detection and CNN are two different problems, they are both mainly based on sum-product operations. These two studies draw similar factors on the performance gap on three quantities: clock frequency (which is comparable to throughput, resource use, and dynamic power consumption. Moreover, it is a common practice to scale up ASICs performance to 65nm CMOS. Table III details the mapping factors to be used to convert the results from a 28nm FPGA to a 28nm ASIC based on the performance gap highlighted earlier and then to a 65nm ASIC based on the same method as [16].

TABLE III. Multiplicative factor to approximate ASICs results

| | 28nm FPGA | 28nm ASIC | 65nm ASIC |
|---|---|---|---|
| Throughput ($\sim f_{clk}$) | 1 | 3 to 6 | 1.3 to 2.6 |
| Resource use / Area | 1 | 1/30 to 1/13 | 1/5.6 to 1/2.4 |
| Dynamic power consumption | 1 | 1/14 to 1/12 | 1/6 to 1/5 |

### A. FPGA architecture

Figure 7 shows the architecture considered as a simplified data flow and as a layout on the PL after the place and root optimizations. On the layout, we notice that the exploration modules noted D-1 and D-2 (for *diversification*) represent the most significant part of the area used. The four exploitation modules marked I (for *intensification*) are the second most important module in the resource use. The other slices relate to the interfaces and the final sorting.

The data flow diagram (Figure 7a) highlights the previously underlined parallelization, which is carried out at three levels. The $D$ explorations are performed at the same time as the $CD$ exploitations. Within these modules, the sub-modules are also executed in parallel: see, for example, the intersections and the computation of their norm. Finally, the internal operations within each sub-module can be parallelized using classical methods. It is evident, for example, that the contribution of each component of a vector can be treated independently when processing its norm. This data flow omits the data-driven synchronization modules for the sake of simplicity.

Two versions have been developed to check the decoding correctness and to measure the throughput. For the correctness verification, the PL is interfaced with the PS using AXI interfaces. The PS supplies the PL on request with the channel data ($\mathbf{y}, \mathbf{H}$), the starting point $\mathbf{x}_0$, and the first singular vectors. The result is then returned to the PS that compares the solution to the one provided by the simulation. The throughput measurement is performed on PL without interaction with the PS to reduce its impact. The FPGA decodes 6365 datasets while counting the number of clock periods and then transmits this counter to the PL. The results of these tests are described in Section V-D.

### B. Module description

Exploration structure is straightforward as it is the direct transcription of the algorithm in Figure 3: $\mathbf{x}_{\text{ZF}}$ and all $\Delta \cap \mathcal{H}$ are processed and projected on $\xi^n$, the objective function (see (16)) is computed and then the $C$ best points per direction are kept for the second step.

The exploitation module introduces the intermediate variable

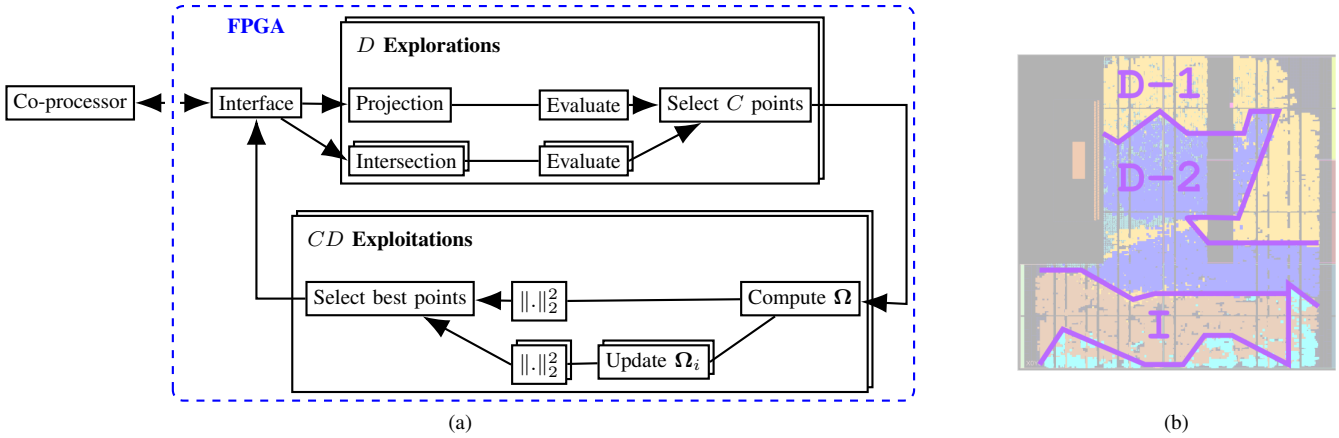
$$\mathbf{\Omega} = \mathbf{y} - \mathbf{H}\mathbf{x} \tag{19}$$

to reduce the complexity of this step. Thus, once $\mathbf{\Omega}$ is calculated, flipping the $i^{\text{th}}$ components of $\mathbf{x}$ only requires updating the preliminary result with

$$\mathbf{\Omega}_i = \mathbf{\Omega} - \mathbf{H}\mathbf{z} \tag{20}$$

where all the components of $\mathbf{z}$ but the $i^{\text{th}}$ are null. In addition, as we use a QPSK, the non-zero component is $\mathbf{z}_i = -2\mathbf{x}_i$ and the update of the result can be reduced to $n$ additions.

The two selection modules cannot use the same principle since one seeks a minimum norm while the other searches for several ones. The selection of the best point is made using a reduction tree in which each level contains a flip-flop. In contrast, the selection of the best $C$ points is based on $C$ registers containing the best norms encountered so far. These




(a) (b)

Figure 7. FPGA architecture implementing exploration and exploitation steps: (a) simplified data flow, (b) layout of main modules for $C = D = 2$ where D-i refer to the $i^{\text{th}}$ exploration module and I to the exploitation modules

registers are initialized with the highest possible norm. Then each candidate is inserted to maintain an increasing order among the registers. Thus, for each norm, $C$ comparisons and at most $C$ shifts are required. Table IV gives an example of the selection process.

TABLE IV. Example of selecting $C = 3$ best points from 5 candidates

| Input norm | Inserted in | Reg 0 | Reg 1 | Reg 2 |
|---|---|---|---|---|
| 26 | Reg 0 | max | max | max |
| 15 | Reg 0 | 26 | max | max |
| 25 | Reg 1 | 15 | 26 | max |
| 19 | Reg 1 | 15 | 25 | 26 |
| 34 | Discarded | 15 | 19 | 26 |
| **Result** | | **15** | **19** | **26** |

### C. Data-driven pipeline

Data-driven pipeline is an approach adapted to prospective work since it divides the global system into small independent elements. Then, each sub-module can be developed and studied independently. This type of pipeline is close to the globally asynchronous locally synchronous (GALS) paradigm. Indeed, it requires an interface with a handshake protocol to control the data flow. Thus, each sub-module represented in Figure 7a is implemented as a GALS element, but the entire FPGA is clocked by the same signal, avoiding the potential metastabilities at the interfaces.

Regarding the interface, we use a single-rail 2-phase protocol as described in [33]: a request and acknowledgment signals are added to the data bus to control the propagation between the different stages. Each GALS interface is then equipped with a request port $R$ driven by the sending module, an acknowledgment port $A$ driven by the receiving module, and a data bus $D$ driven by the sending one. Communications follow the given protocol with $\bar{X}$ being the negation of $X$ and $\oplus$ the exclusive or:

1) Initially, $R = A$ and the data bus state is unspecified.
2) When the sending module has some data to transmit, it sets $D$ to the according value and sets $R = \bar{A}$. From this point, the data bus must keep its state.
3) The receiving module notices the event by testing $R \oplus A$, saves the data from $D$, and sets $A = R$. This module can start its computations.
4) The sending module takes note of the acknowledgment by processing $R \oplus A$ and starts to process the next data, potentially changing the data bus state.

However, additional caution must be taken when there is more than one sender or receiver since there is no guarantee that the processing time is the same for all modules. Muller C-elements are therefore inserted on the $A$ or $R$ ports to transmit requests and acknowledgments only when all modules are ready. A Muller C-element is a gate that replicates the state of its inputs at its output if they are all identical and keeps its previous state otherwise [34].

### D. Result evaluation

As mentioned earlier, the evaluation of the results is done using two variants: one using the PL to check the decoding and one without to evaluate the throughput and resource use.

*1) Correctness test:* The correctness setup involves the PL and the PS and a computer that runs the equivalent decoding. The PS and the computer use a serial port to communicate. The computer starts by generating 500 datasets in the worst condition (SNR = 0dB) that contains a quantized version of a channel matrix, a received vector, the corresponding starting point, and two singular vectors. These values are then export as C files and fed to the PL by the PS. After decoding, the PS read the value from the PL and send it back to the computer through the serial port to be compared with the simulation results.

The PL decoding is compared with a double-valued simulation (i.e., the simulation with the best precision available used in Section IV-B) and a simulation with the same quantization as the hardware to check that it behaves as expected. Figure 8 represents the relative error on the final objective function as a histogram after 500 tests.

The right figure corresponding to the quantized simulation shows that the FPGA produces a solution with the same objective function as the simulation in the vast majority of tests and that the error is always lower than 15%. A closer look during the execution reveals that the errors are due to tricky situations where two choices are equivalent because of the quantization at one stage but do not produce an equivalent final solution. For example, two intersections with identical objective functions

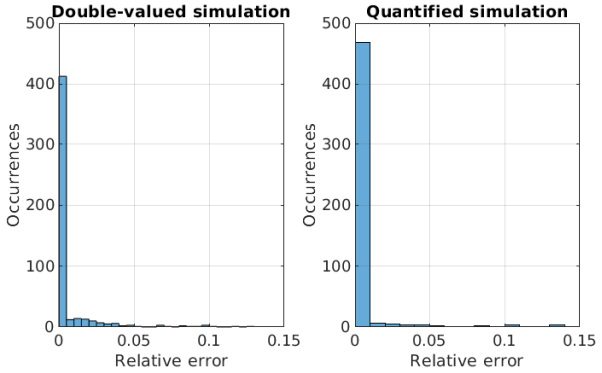



Figure 8. Histogram of the relative error between the simulation and the FPGA implementation over 500 tests

can be selected but do not give the same final point after the exploitation process. The left figure compares the hardware implementation with the perfect unquantized simulation. It indicates that quantization is not too damaging and validates the proof of concept, but a real system would require a better accuracy to obtain a valid BER.

*2) Throughput and resource use test:* The second test is performed without interacting with the PL to investigate the decoder without interference. Measurements are carried out after synthesis, placement, and routing in Vivado using the default settings. Results for the full channel matrix and the QR decomposition setup are compared on Table V when using the same frequency.

TABLE V. Implementation metrics (full matrix & QR-based matrix)

| Channel matrix | Full | QR-based | Change |
|---|---|---|---|
| **Clock frequency (MHz)** | 140 | 140 | 0 |
| **Look Up Tables (LUT) used** | 38 758 | 29 811 | - 23 % |
| **Flip-Flops (FF) used** | 40 708 | 37 630 | - 7.56 % |
| **Digital Signal Processors (DSP) used** | 110 | 70 | - 36.4 % |
| **Throughput (Mb/s)** | 7,88 | 26,9 | + 241 % |
| **Dynamic power consumption (mW)** | 955 | 642 | - 32.8 % |

The first observation to be drawn from these results is that the QR decomposition greatly reduces the use of resources while increasing the throughput recorded. This is due to the reduction by a factor of almost two in the number of channel matrix coefficients that represent the largest part of the data. For example, in the case of $n = m = 10$, storing the full matrix requires $n^2 = 100$ coefficients, however, the QR-based requires to store only $\frac{n(n+1)}{2} = 55$ coefficients. For the rest of the processing, memory usage is shallow. Hence, the channel matrix entries storage has the most significant impact on block memory usage. Thus, by reducing the number of coefficients, we can significantly decrease the number of slices used to store the matrix in the pipeline and reduce the complexity during the placement and routing phase. Besides, the triangularity exploitation reduces the number of computations required for the norms evaluation, which represents the most complex step, as pointed out in Table I.

As stated earlier, it is unclear how these results can be compared with state of the art ASICs optimized implementations with whom the throughput spans from a dozen Mb/s to a few Gb/s [16]. However, we can get an insight into the ASICs results based on the prediction factor described in Table III. Thus, we measured the throughput, and the dynamic power consumption of our 28nm FPGA implementation then multiplied with the projection factor to get the estimation of Table VI. One can notes that the proposed implementation would be in the lower range but would not be far behind. Moreover, we believe that switching from a data-driven pipeline to a fully synchronous implementation would significantly increase the performance leading to an even better result.

TABLE VI. Estimated performance on 65nm ASIC

| | Actual 28nm FPGA | Prediction for 65nm ASIC |
|---|---|---|
| **Throughput (Mb/s)** | 26.9 | 35 to 70 |
| **Dynamic power (mW)** | 642 | 107 to 129 |

## VI. Conclusion

In contrast to the main-stream detection paradigm, which is based on tree-search algorithms, this paper study a new paradigm, namely the geometrical algorithm. This paper highlights that geometrical detection can produce soft outputs using a list and the well-known max-log approximation. After describing the algorithm, we compared the complexity as well as the BER performance of these two approaches. We show that L2E is equivalent or only slightly inferior to the common KSE. Moreover, an implementability study is carried out on a Xilinx Zynq-7000 SoC using a data-driven pipeline. The results are validated by comparison to the simulation, and the resource use and the throughput are evaluated. This throughput is in the lower bound of the state-of-the-art, but it is important to note that references work on ASICs while we use an FPGA.

In light of these results, we think that even if the geometrical paradigm currently shows slightly inferior results that the tree-based one, it could lead to very efficient detectors in the future. Indeed, the tree-based paradigm is matured and has been developed for years, while geometrical detectors are instead on their beginning, and many improvements are still possible. For instance, we highlight that the hardware implementation throughput can be significantly increased by upgrading from FPGA to ASIC and from a data-driven pipeline to synchronous design. In the same way, BER performance could be easily improved by using an iterative detector-decoder framework as already done by the tree-based algorithms.

## References

[1] B. Trotobas and A. Nafkha, "Fixed Complexity Soft-Output Detection Algorithm Through Exploration and Exploitation Processes," in AICT 2018, Barcelona, Spain, Jul. 2018, pp. 89–93.

[2] T. Paul and T. Ogunfunmi, "Evolution, insights and challenges of the PHY layer for the emerging ieee 802.11n amendment," IEEE Communications Surveys & Tutorials, vol. 11, no. 4, 2009, pp. 131–150.

[3] F. Rusek, D. Persson, B. K. Lau, E. G. Larsson, T. L. Marzetta, O. Edfors, and F. Tufvesson, "Scaling Up MIMO: Opportunities and Challenges with Very Large Arrays," IEEE Signal Processing Magazine, vol. 30, no. 1, Jan. 2013, pp. 40–60.

[4] B. Yin, M. Wu, J. R. Cavallaro, and C. Studer, "Conjugate Gradient-based Soft-Output Detection and Precoding in Massive MIMO Systems," 2014 IEEE Global Communications Conference, Dec. 2014, pp. 3696–3701, arXiv: 1404.0424.

[5] Q. Wei, L. Liu, G. Peng, S. Yin, and S. Wei, "Efficient and Flexible VLSI Architecture for Soft-Output Massive MIMO Detector," in Proceedings of Information Science and Cloud Computing. Guangzhou, China: Sissa Medialab, Feb. 2018, p. 055.

[6] M. Wu, B. Yin, G. Wang, C. Dick, J. R. Cavallaro, and C. Studer, "Large-Scale MIMO Detection for 3gpp LTE: Algorithms and FPGA Implementations," IEEE Journal of Selected Topics in Signal Processing, vol. 8, no. 5, Oct. 2014, pp. 916–929, arXiv: 1403.5711.

[7] X. Qin, Z. Yan, and G. He, "A Near-Optimal Detection Scheme Based on Joint Steepest Descent and Jacobi Method for Uplink Massive MIMO Systems," IEEE Communications Letters, vol. 20, no. 2, Feb. 2016, pp. 276–279.

[8] P. W. Wolniansky, G. J. Foschini, G. D. Golden, and R. A. Valenzuela, "V-BLAST: an architecture for realizing very high data rates over the rich-scattering wireless channel," in 1998 URSI International Symposium on Signals, Systems, and Electronics. Conference Proceedings (Cat. No.98EX167), Oct. 1998, pp. 295–300.

[9] A. Riadi, M. Boulouird, and M. M. Hassani, "ZF/MMSE and OSIC Detectors for UpLink OFDM Massive MIMO systems," in 2019 IEEE Jordan International Joint Conference on Electrical Engineering and Information Technology (JEEIT), Apr. 2019, pp. 767–772.

[10] Hongyuan Zhang, Huaiyu Dai, and B. Hughes, "Analysis on the diversity-multiplexing tradeoff for ordered MIMO SIC receivers," IEEE Transactions on Communications, vol. 57, no. 1, Jan. 2009, pp. 125–133.

[11] W. Chin, A. Constantinides, and D. Ward, "Parallel multistage detection for multiple antenna wireless systems," Electronics Letters, vol. 38, no. 12, 2002, p. 597.

[12] C. Studer, S. Fateh, and D. Seethaler, "ASIC Implementation of Soft-Input Soft-Output MIMO Detection Using MMSE Parallel Interference Cancellation," IEEE Journal of Solid-State Circuits, vol. 46, no. 7, Jul. 2011, pp. 1754–1765.

[13] K. K. Y. Wong and P. J. McLane, "A Low-Complexity Iterative MIMO Detection Scheme Using the Soft-Output M-Algorithm," IST Mobile Summit, Jun. 2005, p. 5.

[14] Z. Guo and P. Nilsson, "Algorithm and implementation of the K-best sphere decoding for MIMO detection," IEEE Journal on Selected Areas in Communications, vol. 24, no. 3, Mar. 2006, pp. 491–503.

[15] W. Shin, H. Kim, M. Son, and H. Park, "An Improved LLR Computation for QRM-MLD in Coded MIMO Systems," in 2007 IEEE 66th Vehicular Technology Conference, Sep. 2007, pp. 447–451.

[16] Z. Yan, G. He, Y. Ren, W. He, J. Jiang, and Z. Mao, "Design and Implementation of Flexible Dual-Mode Soft-Output MIMO Detector With Channel Preprocessing," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 62, no. 11, Nov. 2015, pp. 2706–2717.

[17] U. Fincke and M. Pohst, "Improved methods for calculating vectors of short length in a lattice, including a complexity analysis," Mathematics of Computation, vol. 44, no. 170, 1985, pp. 463–471.

[18] P. I.-J. Chuang, M. Sachdev, and V. C. Gaudet, "VLSI implementation of high-throughput, low-energy, configurable MIMO detector," in 2015 33rd IEEE International Conference on Computer Design (ICCD). New York City, NY, USA: IEEE, Oct. 2015, pp. 335–342.

[19] G. He, X. Zhang, and Z. Liang, "Algorithm and Architecture of an Efficient MIMO Detector With Cross-Level Parallel Tree-Search," IEEE Transactions on Very Large Scale Integration (VLSI) Systems, 2019, pp. 1–13.

[20] A. Nafkha, E. Boutillon, and C. Roland, "Quasi-maximum-likelihood detector based on geometrical diversification greedy intensification," IEEE Transactions on Communications, vol. 57, no. 4, Apr. 2009, pp. 926–929.

[21] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO detection," in 2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC), Jul. 2017, pp. 1–5, iSSN: 1948-3252.

[22] A. Datta and V. Bhatia, "A near maximum likelihood performance modified firefly algorithm for large MIMO detection," Swarm and Evolutionary Computation, vol. 44, Feb. 2019, pp. 828–839.

[23] B. Hochwald and S. ten Brink, "Achieving near-capacity on a multiple-antenna channel," IEEE Transactions on Communications, vol. 51, no. 3, Mar. 2003, pp. 389–399.

[24] C. Xu, D. Liang, S. Sugiura, S. X. Ng, and L. Hanzo, "Reduced-Complexity Approx-Log-MAP and Max-Log-MAP Soft PSK/QAM Detection Algorithms," IEEE Transactions on Communications, vol. 61, no. 4, Apr. 2013, pp. 1415–1425.

[25] W. Koch and A. Baier, "Optimum and sub-optimum detection of coded data disturbed by time-varying intersymbol interference (applicable to digital mobile radio receivers)," in Proceedings GLOBECOM '90: IEEE Global Telecommunications Conference and Exhibition, Dec. 1990, pp. 1679–1684.

[26] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in Proceedings IEEE International Conference on Communications, vol. 2, Jun. 1995, pp. 1009–1013.

[27] B. Yin, M. Wu, J. R. Cavallaro, and C. Studer, "VLSI design of large-scale soft-output MIMO detection using conjugate gradients," in 2015 IEEE International Symposium on Circuits and Systems (ISCAS). Lisbon, Portugal: IEEE, May 2015, pp. 1498–1501.

[28] M. S. Khairy, C.-A. Shen, A. M. Eltawil, and F. J. Kurdahi, "Algorithms and Architectures of Energy-Efficient Error-Resilient MIMO Detectors for Memory-Dominated Wireless Communication Systems," IEEE Transactions on Circuits and Systems I: Regular Papers, vol. 61, no. 7, Jul. 2014, pp. 2159–2171.

[29] E. Bjrnson, E. G. Larsson, and T. L. Marzetta, "Massive MIMO: Ten Myths and One Critical Question," IEEE Communications Magazine, vol. 54, no. 2, Feb. 2016, pp. 114–123, arXiv: 1503.06854.

[30] G. H. Golub and C. F. Van Loan, Matrix computations, 4th ed., ser. Johns Hopkins studies in mathematical sciences. Baltimore, Md: Johns Hopkins Univ. Press, 2013, oCLC: 824733531.

[31] I. Kuon and J. Rose, "Measuring the Gap Between FPGAs and ASICs," IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems, vol. 26, no. 2, Feb. 2007, pp. 203–215.

[32] A. Boutros, S. Yazdanshenas, and V. Betz, "You Cannot Improve What You Do not Measure: FPGA vs. ASIC Efficiency Gaps for Convolutional Neural Network Inference," ACM Transactions on Reconfigurable Technology and Systems, vol. 11, no. 3, Dec. 2018, pp. 1–23.

[33] M. Krstic, E. Grass, and X. Fan, "Asynchronous and GALS Design -Overview and Perspectives," in 2017 New Generation of CAS (NGCAS), Sep. 2017, pp. 85–88.

[34] D. E. Muller, University of Illinois at Urbana-Champaign. Graduate College. Digital Computer Laboratory, and University of Illinois at Urbana-Champaign. Department of Computer Science, Theory of asynchronous circuits. Urbana, Illinois : University of Illinois, Graduate College, Digital Computer Laboratory, 1955.