ADMM-based Infinity-Norm Detection for Massive MIMO: Algorithm and VLSI Architecture

Shahriar Shahabuddin, *Member, IEEE,* Ilkka Hautala *Member, IEEE,*,
Markku Juntti, *Fellow, IEEE,* and Christoph Studer, *Senior Member, IEEE*

***Abstract*—We propose a novel data detection algorithm and a corresponding very large scale integration (VLSI) design for massive multi-user (MU) multiple-input multiple-output (MIMO) wireless systems. Our algorithm uses alternating direction method of multipliers (ADMM)-based infinity norm constrained equalization and is called ADMIN. ADMIN is an iterative algorithm that outperforms linear detectors by a large margin when the ratio between the numbers of base-station (BS) and user antennas is small. In the first iteration, ADMIN computes the linear minimum mean-square error (MMSE) solution, which is sufficient when the ratio between the numbers of BS and user antennas is large. We develop time-shared and iterative VLSI architectures for LDL-decomposition based soft-output ADMIN supporting 16- and 32-user systems. We present application-specific integrated circuit (ASIC) designs for 16 to 64 antenna base stations in 28 nm complementary metaloxidesemiconductor (CMOS) that supports up to 64 quadrature amplitude modulation (QAM). The 16-user ADMIN ASIC achieves 303 Mbps while dissipating 85 mW. The 32-user ADMIN ASIC achieves 287 Mbps and 241 Mbps while dissipating 121 mW and 135 mW for 32 and 64 BS antennas, respectively. ADMIN has also been implemented on a Xilinx Virtex-7 field-programmable gate array (FPGA) and is compared with state-of-the-art massive MIMO data detectors.**

***Index Terms*—ADMM, MIMO Detection, massive MIMO, FPGA, ASIC, equalization, soft-output data detection, convex optimization.**

I. Introduction

Massive multi-user (MU) multiple-input multiple-output (MIMO) is a key technology for fifth-generation (5G) wireless communication systems. Traditional small-scale single-user MIMO systems support typically from two to eight antennas at both ends of the communication link. In contrast, massive MU-MIMO equips the base station (BS) with a large number of antenna elements that simultaneously serve a large number of user terminals in the same frequency band [2], [3]. As the number of the BS antennas grows large, random matrix theory demonstrates that the effects of uncorrelated noise and small-scale fading are diminished and the number of supported users per cell becomes independent of the size of the cell [4]. The improvements in spectral efficiency of massive MU-MIMO (over small-scale MIMO) come at the cost of higher computational complexity at the BS. In fact, data detection in the uplink (user transmit to BS) is among the most computationally intensive tasks at the BS side [5]. The receiver at the BS observes a linear superposition of the separately transmitted information symbols and the task of the symbol detector is to separate those transmitted symbols. The complexity of the detection process grows exponentially with the number of antennas at the BS. Thus, the detection problem becomes more challenging in the massive MU-MIMO context.

A. Relevant Prior Art

One of the earliest near-optimal data detectors for massive MIMO was the likelihood ascent search (LAS) algorithm proposed by Vardhan *et al.* in [6]. It searches a sequence of bit vectors with monotonic likelihood ascent. Another local neighborhood search based near maximum likelihood (ML) algorithm called reactive tabu search was presented in [7]. Belief propagation based near-ML detection for massive MIMO can be found in [8]. However, these detectors have not been the most popular choice for circuits and systems community over the past decade. Approximate inversion-based linear data detectors, due to their low complexity, have been the popular choice for hardware implementation of massive MIMO detection. One of the earliest approximate matrix inversion-based detection is Neumann series approximation (NSA) and several implementations of NSA can be found in the literature [5]. Gauss-Seidel (GS) approximate inversion based detection algorithm is also a popular choice for implementation and can be found in [9]. Several other iterative variants have been proposed, such as conjugate gradient (CG) [10], coordinate descent (CD) [11], mismatched approximate message passing [12], and stair matrix based massive MIMO detection [13]. We refer interested readers to [14] for more details on massive MIMO detection techniques.

Such approximate inversion-based linear methods work well under the assumption that the ratio between the number of antennas in the BS and the number of users is large. These algorithms provide high throughput, but entail a significant performance loss compared to exact inversion-based MMSE data detectors, especially in systems with a comparable number of BS antennas and users. The performance of approximate inversion based detectors are also very unstable and their error-rate performance fluctuates greatly depending on the system and

S. Shahabuddin and I. Hautala are with Mobile Networks, Nokia, Oulu, Finland; e-mail: firstname.lastname@nokia.com

M. Juntti is with Centre for Wireless Communications, University of Oulu, Finland, e-mail: markku.juntti@oulu.fi

C. Studer is with the Department of Information Technology and Electrical Engineering at ETH Zurich, Zurich, Switzerland; e-mail: studer@ethz.ch

The research was supported financially by Academy of Finland 6Genesis Flagship (grant 318927). The work of C. Studer was supported in part by ComSenTer, one of six centers in JUMP, an SRC program sponsored by DARPA, by an ETH Research Grant, and by the US National Science Foundation (NSF) under grants CNS-1717559 and ECCS-1824379.

The detector implemented in this paper builds upon the VLSI architecture presented at the IEEE International Symposium on Circuits and Systems [1].

channel models, the number of users or BS antennas, coding scheme, algorithm structure, and the number of iterations. The performance can get worse to the point that such algorithms fail to successfully detect the transmitted data. The instability of approximate inversion based algorithms can jeopardize the BS product development for network vendors who must support a variety of scenarios and configurations as customer requirement [15]. In a practical MU detection scenarios below 6 GHz, the deployment of a large number of radio-frequency (RF) chains may not be feasible due to the size, weight, and cost of the BS. The convergence rates of approximate-inversion based data detectors will significantly deteriorate in such scenarios. A study in [16] suggested that exact matrix inversion based detectors are viable alternatives from both complexity and latency perspective. A few implementations of exact-inversion based data detection have been proposed in [17] and [18].

According to Björnson *et al.*, there is a common misconception that massive MIMO refers to systems with at least an order of magnitude more base station antennas than terminals [19]. In fact, there is no strict requirement on the relation between the numbers of users and antennas in massive MIMO, since the ratio depends on the system performance metric, propagation environment, and coherence time block length. In addition, network providers would be able to serve as many users as possible for the given number of BS antennas. Popular massive MIMO products, such as Nokia AirScale, Ericsson Air 6468, and Huawei AAU are equipped with 64 antennas and can support up to 16 layers [20]. Therefore, massive MIMO systems with a small ratio between the number of BS antennas and users are of practical importance—however, only a few papers have studied data detection in such scenarios.

B. Contributions

We propose a novel data detection algorithm based on alternating direction method of multipliers (ADMM). Our algorithm is referred to as ADMM-based infinity-norm (ADMIN) and performs box-relaxation-based equalization, which outperforms linear detectors by a large margin in systems with small ratios between the number of BS antennas and users (two or less). ADMIN is iterative by nature and performs linear MMSE equalization in the first iteration. Therefore, for systems where the numbers of BS antennas are an order of magnitude larger than that of the users, it is sufficient to perform a single ADMIN iteration. We present an LDL-decomposition based soft-output version of the algorithm and extensive simulation results to compare the error-rate performance of ADMIN with existing state-of-the-art data detectors. In addition, we design two iterative and time-shared very large scale integration (VLSI) architectures for ADMIN. Our architectures support 16 and 32 users for 16 and 32 BS antennas, respectively. We propose implementation results in a 28 nm complementary metaloxidesemiconductor (CMOS) technology and on a Xilinx Virtex-7 field-programmable gate array (FPGA), and we compare our data detectors to existing ones in the literature.

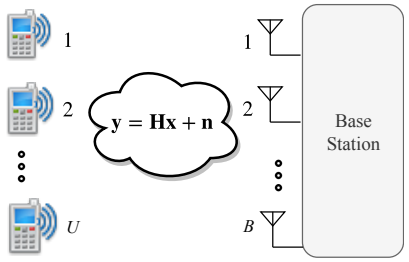


Fig. 1: A massive MU-MIMO system in which a large number of BS antennas are serving a large number of users. The channel between the BS and users is modeled as $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$.

C. Notation

Boldface lowercase and boldface uppercase letters stand for column vectors and matrices, respectively. For a matrix, $\mathbf{A}$, we denote its Hermitian transpose by $\mathbf{A}^H$. We use $A_{k,l}$ for the entry in the kth row and lth column of the matrix $\mathbf{A}$. The real and imaginary part of a complex-valued matrix $\mathbf{A}$ are denoted by $\Re(\mathbf{A})$ and $\Im(\mathbf{A})$, respectively. The identity matrix is $\mathbf{I}$ and ℓ_2-norm of the vector $\mathbf{a}$ is $\|\mathbf{a}\|_2 = \sqrt{\sum_K |a_k|^2}$.

D. Outline of the Paper

The rest of the paper is organized as follows. Section II introduces the system model and discusses the data detection problem. Section III details the ADMIN algorithm. Section IV presents an LDL-based soft-output detector variant and provides a complexity and error-rate performance comparison. Section V presents the VLSI architecture of ADMIN along with a fixed-point analysis. Section VI presents application-specific integrated circuit (ASIC) and FPGA implementation results. We conclude in Section VII.

II. System Model and Data Detection

We consider a massive MU-MIMO wireless uplink system that employs orthogonal frequency division multiplexing (OFDM). We assume that U single-antenna user terminals send data simultaneously to a BS with $B \geq U$ antennas over W subcarriers. The U users first encode their own bit stream with a forward error correcting code (e.g., a convolutional code) and map the coded bit stream to constellation points in the finite alphabet set $\mathcal{O}$ (e.g., 16 quadrature amplitude modulation (QAM) with Gray mapping) with an average transmit power E_s per symbol. We assume perfect channel state information (CSI) and synchronization at the receiver, as well as a sufficiently long cyclic prefix such that we can consider each subcarrier to be frequency flat. By omitting the subcarrier index, the per-subcarrier input-output can be written as $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$, where $\mathbf{y} \in \mathbb{C}^B$ is the received signal vector, $\mathbf{x} \in \mathbb{C}^U$ is the transmit symbol vector, $\mathbf{H} \in \mathbb{C}^{B \times U}$ is the channel matrix, and $\mathbf{n} \in \mathbb{C}^B$ is the circularly symmetric complex white Gaussian noise vector with zero mean and variance N_0 per complex entry. An illustration of the considered system model is shown in Fig. 1.

A. Maximum-Likelihood Data Detection

Maximum likelihood (ML) data detection is optimal in terms of minimizing the vector error rate when all data vectors are equally likely. The ML detector finds the transmit vector for which the Euclidean distance after passing it through the MIMO channel is minimized, i.e.,

$$\hat{\mathbf{x}}_{\text{ML}} = \arg\min_{\mathbf{x}\in\mathcal{O}^U} \|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2. \tag{1}$$

Unfortunately, this problem is of combinatorial nature and an exhaustive search requires exponential complexity in the number of users [21]. Hence, alternative low-complexity methods are often used in practical systems.

B. Equalization-based Data Detection

Equalization-based data detection first forms an estimate of $\mathbf{x}$ as $\hat{\mathbf{x}}_{\text{G}} = \mathbf{G}\mathbf{y}$ by using linear equalization, i.e., multiplication by matrix $\mathbf{G}$. For zero-forcing (ZF) equalization, the matrix $\mathbf{G}$ is the pseudo-inverse of $\mathbf{H}$. ZF equalization can be viewed as the solution of a relaxed ML problem in which the discrete constellation set $\mathcal{O}$ is relaxed to the complex numbers $\mathbb{C}$ [22]. Concretely, ZF-equalization can be expressed as

$$\hat{\mathbf{x}}_{\text{ZF}} = \arg\min_{\mathbf{x}\in\mathbb{C}^U} \|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2. \tag{2}$$

Linear minimum mean-square error (MMSE) equalization also relaxes the discrete constellation $\mathcal{O}$ to the complex numbers $\mathbb{C}$ with an additional regularization term. Concretely, linear MMSE equalization can be expressed as

$$\hat{\mathbf{x}}_{\text{MMSE}} = \arg\min_{\mathbf{x}\in\mathbb{C}^U} \|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2 + N_0E_s^{-1}\|\mathbf{x}\|_2^2. \tag{3}$$

The regularizer $N_0E_s^{-1}\|\mathbf{x}\|_2^2$ prevents $\mathbf{x}$ from growing too large, since the transmit signals are taken from a finite-energy discrete constellation $\mathcal{O}$ centered around zero. Because the objective function of (3) is quadratic in $\mathbf{x}$, MMSE equalization has a well-known closed form solution [22].

III. ADMIN: ADMM-based Infinity Norm Detection

We now propose our ADMIN data detector for massive MU-MIMO systems.

A. Infinity-Norm Constrained Equalization

We define the component-wise ℓ_∞-norm of a complex-valued vector $\mathbf{x}$ can be expressed as follows:

$$\|\mathbf{x}\|_{\tilde{\infty}} = \max_i\{\max\{\Re(x_i),\Im(x_i)\}\}. \tag{4}$$

Constraining this norm as $\|\mathbf{x}\|_{\tilde{\infty}} \le \rho$ can be seen as putting a box around each component with side length 2ρ. The idea of box-constrained equalization [23], [24] is to relax the finite-alphabet constraint $\mathbf{x}\in\mathcal{O}^U$ of the ML problem to the convex polytope $\mathcal{C}_\mathcal{O}$ around the constellation set $\mathcal{O}$ and to solve the following convex optimization problem:

$$\hat{\mathbf{x}}_{\text{BOX}} = \arg\min_{\mathbf{x}\in\mathcal{C}_\mathcal{O}^U} \|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2. \tag{5}$$

The convex polytope for QAM alphabets can be expressed as $\mathcal{C}_\mathcal{O} = \{x_R + jx_I : x_R, x_I \in [-\alpha,+\alpha]\}$ where $\alpha = \max_{u\in\mathcal{O}}\Re\{u\}$ is the tightest radius of the box around the square constellation. For example, the convex polytope for quadrature phase-shift keying (QPSK) is given by a square box with radius $\alpha = 1$ around the square constellation of QPSK. With the ℓ_∞-norm, we have $\mathcal{C}_\mathcal{O}^U = \{\mathbf{x}\in\mathbb{C}^U : \|\mathbf{x}\|_{\tilde{\infty}} \le \alpha\}$. Since the problem in (5) is convex, a solution could be found using off-the-shelf interior-point methods. Such methods, however, are not particularly hardware friendly and special-purpose solvers are required to efficiently find a solution to (5). We next propose an ADMM-based solution that is hardware friendly and is able to solves the box-relaxation problem in (5) efficiently.

B. ADMM-based Algorithm

The alternating direction method of multipliers (ADMM) is a well known numerical method to solve convex optimization problems in which objective function and constraints are convex [25]. ADMM solves the original convex problem by breaking it into smaller sub-problems that can be solved efficiently. ADMM blends the decomposability of the dual ascent method with the superior convergence properties of the method of multipliers. A generic constrained convex optimization problem

$$\text{minimize } f(\mathbf{x}) \quad \text{subject to } \mathbf{x}\in\mathcal{C}$$

with a variable $\mathbf{x}\in\mathbb{R}^n$ can be re-written in ADMM form as

$$\text{minimize } f(\mathbf{x}) + g(\mathbf{z}), \quad \text{subject to } \mathbf{x} = \mathbf{z}$$

where g is an indicator function of the convex set $\mathcal{C}$. The scaled ADMM form for this problem is

$$\begin{aligned}
\mathbf{x}^{k+1} &= \arg\min_{\mathbf{x}}\left\{f(\mathbf{x}) + \frac{\rho}{2}\|\mathbf{x}-\mathbf{z}^k+\mathbf{u}^k\|_2^2\right\}\\
\mathbf{z}^{k+1} &= \arg\min_{\mathbf{z}}\left\{g(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{x}^{k+1}-\mathbf{z}+\mathbf{u}^k\|_2^2\right\}\\
\mathbf{u}^{k+1} &= \mathbf{u}^k + \mathbf{x}^{k+1} - \mathbf{z}^{k+1},
\end{aligned}$$

where $\mathbf{u}$ is the scaled dual variable [25]. Here, the $\mathbf{x}$-update involves minimizing f and $\mathbf{z}$-update involves minimizing g. As g is an indicator function of a closed nonempty convex set $\mathcal{C}$, the $\mathbf{z}$ update can be written as

$$\mathbf{z}^{k+1} = \arg\min_{\mathbf{z}}\left\{g(\mathbf{z}) + \frac{\rho}{2}\|\mathbf{z}-\mathbf{v}\|_2^2\right\} = \Pi_\mathcal{C}(\mathbf{v}), \tag{6}$$

where $\mathbf{v} = \mathbf{x}^{k+1} + \mathbf{u}^k$ is a constant vector for the purpose of $\mathbf{z}$-update and $\Pi_\mathcal{C}(\mathbf{v})$ denotes the Euclidean projection onto $\mathcal{C}$ [25]. Thus, the scaled ADMM problem can be written as

$$\begin{aligned}
\mathbf{x}^{k+1} &= \arg\min_{\mathbf{x}}\left\{f(\mathbf{x}) + \frac{\rho}{2}\|\mathbf{x}-\mathbf{z}^k+\mathbf{u}^k\|_2^2\right\}\\
\mathbf{z}^{k+1} &:= \Pi_\mathcal{C}(\mathbf{x}^{k+1}+\mathbf{u}^k)\\
\mathbf{u}^{k+1} &= \mathbf{u}^k + \mathbf{x}^{k+1} - \mathbf{z}^{k+1}.
\end{aligned}$$

C. ADMM-Based Infinity-Norm Detection

We rewrite (5) into the following equivalent form

$$\underset{\mathbf{x},\mathbf{z}\in\mathbb{C}^U}{\text{minimize}} \ \frac{1}{2}\|\mathbf{y}-\mathbf{H}\mathbf{x}\|_2^2 + g(\mathbf{z}) \quad \text{subject to } \mathbf{z} = \mathbf{x} \tag{7}$$

where $g(\mathbf{z})$ is the indicator function for the convex set $\mathcal{C}_{\mathcal{O}}$ defined by

$$g(\mathbf{z}) = \begin{cases} 0, & \text{if } \mathbf{z} \in \mathcal{C}_{\mathcal{O}}^{U} \\ \infty, & \text{otherwise.} \end{cases}$$

The augmented Lagrangian for the problem in (7) is

$$L_{\beta}(\mathbf{x}, \mathbf{z}, \boldsymbol{\lambda}) = \frac{1}{2}\|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2^2 + g(\mathbf{z}) + \frac{\beta}{2}\|\mathbf{z} - \mathbf{x} - \boldsymbol{\lambda}\|_2^2, \quad (8)$$

where $\boldsymbol{\lambda}$ is the scaled dual variable associated with the constraint $\mathbf{z} = \mathbf{x}$ and $\beta > 0$ is a suitably chosen regularization parameter.

We can now use ADMM to solve the augmented Lagrangian over $\mathbf{x}$ and $\mathbf{z}$. Concretely, the procedure is as follows:

$$\begin{aligned} \hat{\mathbf{x}}^{k+1} &= \arg\min_{\mathbf{x}} L_{\beta}(\mathbf{x}, \mathbf{z}^k, \boldsymbol{\lambda}^k) \\ \hat{\mathbf{z}}^{k+1} &= \arg\min_{\mathbf{z}} L_{\beta}(\hat{\mathbf{x}}^{k+1}, \mathbf{z}, \boldsymbol{\lambda}^k) \\ \boldsymbol{\lambda}^{k+1} &= \boldsymbol{\lambda}^k + \hat{\mathbf{z}}^{k+1} - \hat{\mathbf{x}}^{k+1}. \end{aligned}$$

In the first step of the ADMM iteration, we minimize $\mathbf{x}$ while $\mathbf{z}$ is fixed. We take the derivative of (8) with respect to $\mathbf{x}$ and set it to zero to compute the first step of ADMIN as

$$\begin{aligned} &\mathbf{H}^H(\mathbf{y} - \mathbf{H}\mathbf{x}) - \beta(\mathbf{z} - \mathbf{x} - \boldsymbol{\lambda}) = 0 \\ &\quad \Longrightarrow \hat{\mathbf{x}} = (\mathbf{H}^H\mathbf{H} + \beta\mathbf{I})^{-1}(\mathbf{H}^H\mathbf{y} + \beta(\mathbf{z} - \boldsymbol{\lambda})). \quad (9) \end{aligned}$$

Here, $\mathbf{G} = \mathbf{H}^H\mathbf{H} + \beta\mathbf{I}$ is a regularized Gramian matrix and $\mathbf{y}_{\text{MF}} = \mathbf{H}^H\mathbf{y}$ is a matched filter. In other words, the $\mathbf{x}$-update of ADMIN solves a regularized least-squares problem. Thus, ADMIN can be viewed as a method for solving the box-constrained problem of (5) by iteratively carrying out regularized least-square computations. Note that, initializing $\mathbf{z}$ and $\boldsymbol{\lambda}$ with zero at (9) provides us with the linear MMSE equalization solution in the first iteration. The $\mathbf{z}$ update step can be expressed as

$$\hat{\mathbf{z}} = \arg\min_{\mathbf{z} \in \mathcal{C}_{\mathcal{O}}^{U}} (\beta/2)\|\mathbf{z} - (\hat{\mathbf{x}} + \boldsymbol{\lambda})\|_2^2. \quad (10)$$

Equation (10) is an orthogonal projection of $\hat{\mathbf{x}} + \boldsymbol{\lambda}$ onto the convex polytype $\mathcal{C}_{\mathcal{O}}^{U}$. This projection is given by

$$\text{proj}_{\mathcal{C}_{\mathcal{O}}}(w) = \begin{cases} w, & \text{if } w \in \mathcal{C}_{\mathcal{O}} \\ \arg\min_{q \in \mathcal{C}_{\mathcal{O}}} |w - q|, & \text{otherwise.} \end{cases}$$

In words, if w is outside the set $\mathcal{C}_{\mathcal{O}}$, the projection outputs the value closest to w within the set $\mathcal{C}_{\mathcal{O}}$ in terms of the Euclidean distance. For example, if w is outside of a box of $\alpha = 1$ that encloses the square constellation of QPSK, the projection outputs a value q that is closest to w within the box. The update for the Lagrange vector is given by

$$\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} - \gamma(\hat{\mathbf{z}} - \hat{\mathbf{x}}), \quad (11)$$

where we introduce a suitably chosen penalty parameter (or the augmented Lagrangian parameter) $0 < \gamma$ for the ADMIN algorithm. Note that $0 < \gamma < 1$ ensures the convergence of the ADMM, but larger choices may lead to improved results for a very small number of iterations.

Algorithm 1 ADMIN

inputs: $\mathbf{y}$, $\mathbf{H}$, N_0 and E_s
preprocessing
  $\beta = N_0 E_s^{-1}\epsilon$
  $\mathbf{G} = \mathbf{H}^H\mathbf{H} + \beta\mathbf{I}_U$
  $\mathbf{G} = \mathbf{L}\mathbf{D}\mathbf{L}^H$
  $\tilde{\mathbf{L}} = \mathbf{L}^{-1}$, $\tilde{\mathbf{D}} = \mathbf{D}^{-1}$
initialization
  $\mathbf{z} = \mathbf{0}$
  $\boldsymbol{\lambda} = \mathbf{0}$
detection
  $\mathbf{y}_{\text{MF}} = \mathbf{H}^H\mathbf{y}$
  for $i = 1, \ldots, K$
    $\hat{\mathbf{x}} \leftarrow \tilde{\mathbf{L}}^H\tilde{\mathbf{D}}\tilde{\mathbf{L}}(\mathbf{y}_{\text{MF}} + \beta(\mathbf{z} - \boldsymbol{\lambda}))$
    $\hat{\mathbf{z}} \leftarrow \text{proj}_{\mathcal{C}_{\mathcal{O}}}(\hat{\mathbf{x}} + \boldsymbol{\lambda}, \alpha)$
    $\boldsymbol{\lambda} \leftarrow \boldsymbol{\lambda} - \gamma(\hat{\mathbf{z}} - \hat{\mathbf{x}})$
    $\mathbf{z} \leftarrow \hat{\mathbf{z}}$
  end
output: $\hat{\mathbf{x}}$

IV. LDL-Decomposition based Soft-output ADMIN

Inversion of the regularized Gram matrix $\mathbf{G} = \mathbf{H}^H\mathbf{H} + \beta\mathbf{I}_U$ is required to implement the $\mathbf{x}$-update of ADMIN. Since $\mathbf{G}$ is Hermitian and positive-definite, we can compute the exact inverse using an LDL-decomposition [9]. Fortunately, the computations required for $\mathbf{G}$, LDL, and inversion of $\mathbf{L}$ and $\mathbf{D}$ can be done during the preprocessing, and thereby the detection stage can be simplified.The Hermitian transpose of the inversion of $\mathbf{L}$ yields the inversion of $\mathbf{L}^H$. A similar approach of using LDL-decomposition for soft-output detection can be found in [18]. It should be noted that similar functionality can be accomplished using Cholesky decomposition [17], however, we choose LDL-decomposition because of its less complexity.

ADMIN computes the matched filter at the beginning of the detection process. The matched filter should be calculated at symbol rate as $\mathbf{y}$ is required for $\mathbf{y}_{\text{MF}} = \mathbf{H}^H\mathbf{y}$ calculation. During the detection stage, ADMIN computes the matched filter and afterwards, iteratively updates $\hat{\mathbf{x}}$, $\hat{\mathbf{z}}$, and $\boldsymbol{\lambda}$. The complete ADMIN algorithm is presented in Algorithm 1. An intuitive functional diagram of the preprocessing and detection stages is presented in Fig. 2. The post-equalization signal-to-noise-plus-interference ratio (SINR) vector $\boldsymbol{\rho}$, which is required to compute the log-likelihood ratio (LLR) values, can be computed as $\rho_i = 1/N_0 E_s^{-1} g_i$ where g_i is the i-th entry of the main diagonal of $\mathbf{G}^{-1}$ [5]. This can be done efficiently with the help of $\tilde{\mathbf{L}} = \mathbf{L}^{-1}$ and $\tilde{\mathbf{D}} = \mathbf{D}^{-1}$ as $g_i = (\tilde{\mathbf{l}}_i)^H \text{diag}(\tilde{\mathbf{D}})(\tilde{\mathbf{l}}_i)$ where $\tilde{\mathbf{l}}_i$ is the i-th column of $\tilde{\mathbf{L}}$. If the ratio between the numbers of BS antennas and users is large[1], we can approximate the preprocessing stage by only taking the inverse of the diagonal elements of the Gram matrix. Therefore, the calculation of g_i can be expressed as

$$g_i = \begin{cases} 1/\mathbf{G}_{ii}, & \text{if } B > U \\ (\text{diag}(\mathbf{G}^{-1}))_i & \text{otherwise} \end{cases}$$

[1]Typically, a ratio of 10 or above is considered as large. For example, a 128×8 system.

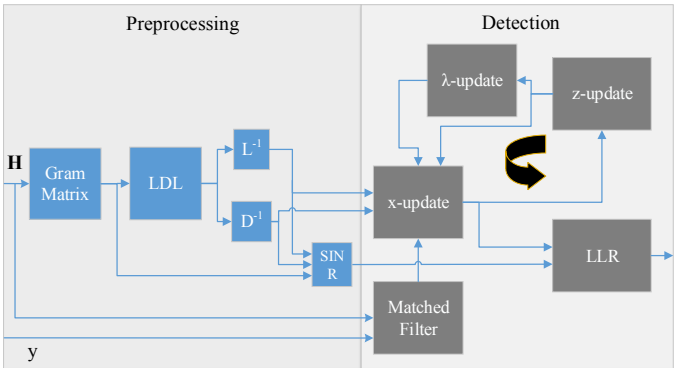


Fig. 2: Functional diagram of ADMIN preprocessing and detection. The curved and bold arrow represents the iterative process of ADMIN.

Note that, β uses a scaled version of $N_0 E_s^{-1}$, while the SINR calculation uses an unscaled version of $N_0 E_s^{-1}$. The max-log approximated LLR values can be computed from the SINR and $\hat{\mathbf{x}}$ following [26]. We model the transmit symbol of user i as $\hat{x}_i = \mu_i x_i + z_i$, where μ_i is the effective channel gain and z_i is the post-equalization noise-plus-interference with variance ${v_i}^2$ [11]. By defining $\rho_i^2 = (\mu_i^2)^2/v_i^2$ as the post-equalization SINR, the max-log approximated LLR of bit b for user i can be computed as

$$L_{i,b} = \rho_i \left(\min_{a \in \mathcal{X}_b^0} |\frac{\hat{x}_i}{\mu_i} - a|^2 - \min_{a' \in \mathcal{X}_b^1} |\frac{\hat{x}_i}{\mu_i} - a'|^2 \right) \qquad (12)$$

where the sets $\mathcal{X}_b^0$ and $\mathcal{X}_b^1$ contain the constellation symbols where bit b of the symbol in $\mathcal{O}$ equals to 0 and 1, respectively [27]. According to [26], the calculation inside the parentheses of (12) can be greatly simplified by exploiting modulation of Gray mapping.

A. Complexity Analysis

We now analyze the computational complexity of ADMIN and compare it with state-of-the-art detection algorithms. Most of the massive MIMO detection algorithms are iterative and therefore, we compare the number of real-valued multiplications of a single iteration of the detection algorithms. The number of complex-valued multiplications needed for line 12 of Algorithm 1 is $2U^2 + 2U$ where U is the number of users. Here, U^2 multiplications are needed for every matrix-vector multiplication. Thus, the matrix-vector multiplication of $\tilde{\mathbf{L}}$ and $\mathbf{y}_{\text{MF}} + \beta(\mathbf{z} - \boldsymbol{\lambda})$ requires U^2 complex operations and results in a vector with U elements. This vector is multiplied element-wise with the diagonal vector, $\tilde{\mathbf{D}}$. The $\tilde{\mathbf{L}}^H$ matrix is finally multiplied with the resulting vector in earlier sentence, which requires another U^2 complex multiplications. The cost of projection is negligible and another scalar-vector multiplication is required for line 14 that requires U multiplications. Therefore, the total complexity of ADMIN per iteration is $2U^2 + 3U$ which scales with KU^2 where K is the number of ADMIN iterations. An additional BU complex multiplication is required for the MF calculation at line 10 of Algorithm 1. In Table I, the number of real-valued multiplications for K iterations are shown for different iterative data detection algorithms. It can be seen that

TABLE I: Complexity analysis of massive MU-MIMO detection algorithms

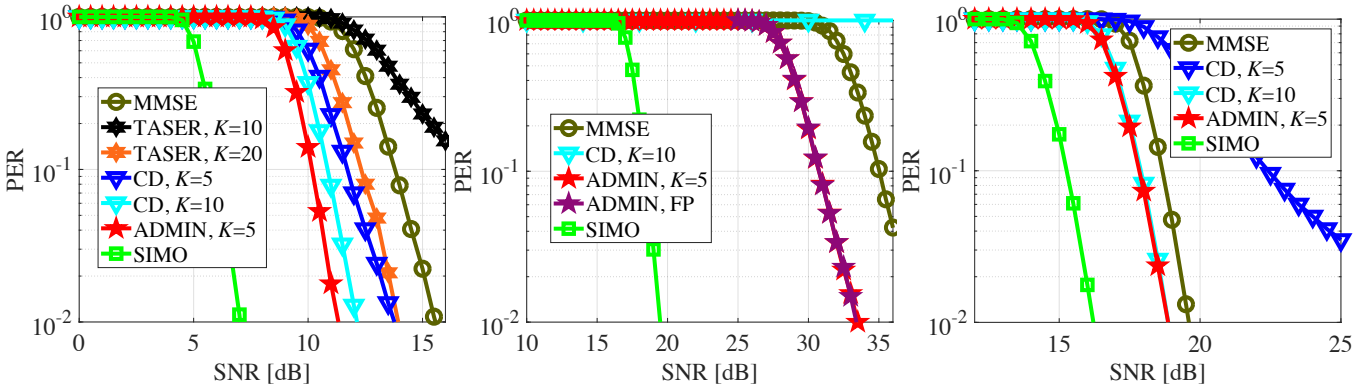
Algorithm	Computational complexity
BPSK TASER [28]	$K(\frac{1}{3}U^3 + \frac{5}{2}U^2 + \frac{37}{6}U + 4)$
QPSK TASER [28]	$K(\frac{8}{3}U^3 + 10U^2 + \frac{37}{3}U + 4)$
CG [10]	$(K+1)(4U^2 + 20U)$
NSA [5]	$(K-1)(2U^3 + 2U^2 - 2U)$
CD [11]	$K(8BU + 4U)$
GS [9]	$6KU^2$
ADMIN	$K(8U^2 + 12U) + 4BU$
MMSE	$4U(U^2 + B^2 + B)$

TASER [28] and the NSA [5] algorithms scale with KU^3 and exhibit more complexity compared to ADMIN. CG [10] and GS [9] scale with KU^2 and are slightly less complex compared to ADMIN. CD [11] scales with KBU. It should be noted that we listed the complexity of a single iteration while the number of iterations requires to achieve a satisfactory performance differs for each algorithm. The error-rate comparison is used to determine the required number of iterations and discussed in the next subsection.

We also like to point out that CG, NSA, CD, and GS algorithms include pre-processing. Thus, the result in Table I is not entirely fair. However, BPSK TASER and QPSK TASER complexity also do not include any pre-processing in Table I. ADMIN follows a standard pre-processing process based on Gramian matrix calculation, LDL-decomposition and inverse of $\mathbf{L}$ computation. A BS with B antennas supporting U users need to multiply a $B \times U$ and $U \times B$ matrices for the Gramian matrix. This calculation requires a total B^2U multiplications. The size of the Gramian matrix is $U \times U$ and the LDL-decomposition of this matrix requires $\frac{1}{6}U^3$ multiplications. The inversion of the triangular matrix, $\mathbf{L}$ can be accomplished by back-substitution. The back-substitution of the $U \times U$ matrix takes $\frac{1}{2}(U^2 - U)$ multiplications.

B. Error-Rate Performance

We simulate a typical 40 MHz IEEE 802.11n OFDM uplink scenario with a rate-3/4 convolutional code where the channel matrices are generated using WINNER-phase-2 model. A max-log BCJR algorithm is used for soft-input channel decoding. We use 6000 packets per simulation where each packet contains one OFDM symbol. Each OFDM symbol uses 128 subcarriers in our simulations. We invite interested readers to explore [29] more detail on WINNER-phase-2 channel model. We show the (coded) packet error-rate (PER) for ADMIN as well as linear MMSE, single-input multiple-output (SIMO) lower bound, TASER and box-constrained coordinate descent (CD) detector [30] in Fig. 3. CD outperforms other MMSE approximations like the Neumann series approach [31] or the conjugate gradient (CG) [11] based detectors in terms of PER. For a small (4 or less) base-station-to-user-antenna ratio, the performance gain of CD over the state-of-the-art detectors is more pronounced. Therefore, outperforming CD for any antenna configuration means ADMIN can outperform the state-of-the-art approximate inversion-based massive MIMO detectors. Therefore, we compare ADMIN with five iterations

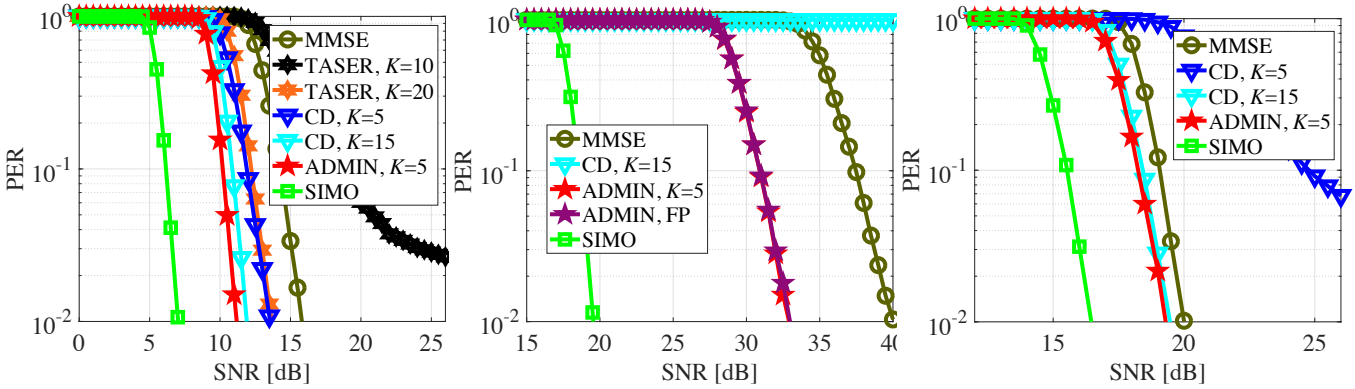


(a) 16 BS antennas, 16 users and QPSK.

(b) 16 BS antennas, 16 users and 64-QAM.

(c) 32 BS antennas, 16 users and 64-QAM.

Fig. 3: Packet error rate (PER) for a massive MU-MIMO-OFDM system with rate-3/4 code and WINNER channels for 16 users [1].



(a) 32 BS antennas, 32 users and QPSK.

(b) 32 BS antennas, 32 users and 64-QAM.

(c) 64 BS antennas, 32 users and 64-QAM.

Fig. 4: Packet error rate (PER) for a massive MU-MIMO-OFDM system with rate-3/4 code and WINNER channels for 32 users.

($K = 5$) to the CD with five and ten ($K = 10$) iterations respectively. Figs. 3a and 3b illustrates the PER performance for a 16-user and 16 BS antennas system. For lower order modulation (e.g., QPSK), ADMIN achieves a substantial performance gain over MMSE. ADMIN also outperforms TASER and CD with a smaller number of iterations. For higher order modulation (e.g. 64-QAM), CD fails to detect the symbols for square systems. ADMIN provides approximately 5 dB gain compared to linear MMSE in this scenario. Fig. 3c illustrates the PER performance for 16 users and 32 BS antennas. ADMIN with five and CD with ten iterations performs similarly in this scenario; however, CD performs worse than MMSE with five iterations.

In Fig. 4, the detectors are simulated for 32-user systems. Figs. 4a and 4b illustrate the ADMIN provide substantial gain over linear MMSE for lower and higher order modulations in a square system. Similar to 16-user square MIMO configurations, CD is ineffective for a square MIMO system and higher order modulations. In lower order modulations, TASER and CD requires significantly higher number of iterations to reach even close to ADMIN's performance. Fig. 4c illustrates the performance of 64-antenna BS and 32-user systems. It can be seen that the ADMIN with $K = 5$ iterations provides similar performance to CD with $K = 15$ iterations. For this configuration, ADMIN provides better performance than exact inversion based MMSE. The simulation results show that ADMIN algorithm provides significant performance improvements when the ratio of numbers of BS antennas and users are small. Most of the state-of-the-art massive MIMO detection algorithms fail to successfully detect the transmitted data in such scenarios. The fixed-point performance of ADMIN algorithm is shown in Figs. 3b and 4b and denoted as ADMIN, FP. A detail on the chosen word-lengths can be found in the beginning of Section V.

We simulate the performance of ADMIN for 256-QAM and for different BS antennas and users ratios in Fig 5. First, we simulate different detectors for 128 BS antennas and 16 users in Fig 5a. Here, the ratio between BS antennas and users is 8 and it can be seen that the MMSE performs well in this scenario. ADMIN provides approximately 0.5 dB to 1 dB gain compared to linear MMSE with its 2nd iteration. CD fails to detect symbols with $K = 2$ iterations. This simulation shows the efficacy of ADMIN with low number of iterations even when the BS antennas and users ratio is high. In Fig 5b, the

detectors are simulated when the ratio between BS antennas and users is 4. Again, ADMIN provides 0.5 dB to 1 dB gain compared to MMSE in this scenario. In this scenario, CD, with 5 iterations, can reach close to ADMIN's 2 iterations and CD with $K = 2$ and $K = 3$ iterations fail to detect received symbol vectors. We also simulate 256-QAM when BS antennas and users ratio is 1 in Fig 5c. We can see that ADMIN provides significant gain over MMSE with $K = 5$ and $K = 7$ iterations.

C. Parameter Selection

The performance of ADMIN depends on the parameters β and γ discussed in Section III. It should be noted that the number of iterations to reach a desired performance is dependent on the ADMM parameters [32]. The mathematical analysis to determine the optimal values of β and γ is beyond the scope of this paper. We use simulations to determine the values of β and γ that provides us satisfactory error-rate performance. In Fig. 6, ADMIN detector is simulated for 32 users and 32 BS antennas. It can be seen that the performance varies notably between different values of β and γ. We selected $\beta = 3$ and $\gamma = 2$ for all of our simulations of Figs. 3 and 4.

V. VLSI Architecture

A. Architecture Overview and Fix-Point Optimization

We propose VLSI architectures which takes $\mathbf{H}$, $\mathbf{y}$, $\tilde{\mathbf{L}}$, $\tilde{\mathbf{d}} = \text{diag}(\tilde{\mathbf{D}})$ as inputs for the ADMIN detector. We use fixed-point arithmetic to optimize the efficiency of our designs. We quantize various parts in such a way that the components such as complex multipliers and adder tree can be reused. Complex-valued multipliers with 18-bit real and imaginary numbers are sufficient for all computations. Therefore, all the inputs are quantized to 18-bits. It should be noted that in a systolic array architecture, the inputs can be quantized to smaller values. However, due to the iterative nature of ADMIN, the output of the adder tree, that is used as inputs of the multipliers in subsequent iterations, requires a higher number of bits. The VLSI architectures support ADMIN detection (lines $6-16$) of Algorithm 1. The architectures are divided in two parts and they are explained next.

B. Vector Multiplication Unit

The vector multiplication unit (VM) computes the $\mathbf{x}$-minimization step of ADMIN (line 12) of Algorithm 1. VM is designed with time-shared processing elements to compute vector-vector multiplication. A block diagram of the VM unit is shown in Fig. 7. It consists of complex multipliers followed by an adder tree. The number of complex multipliers is 16 and 32 for the 16-user and 32-user ADMIN, respectively. We insert pipeline registers between each complex multiplier and adder tree to reduce the critical path. The adder trees are used to sum 16 or 32 complex values for 16- or 32-user architectures respectively. $\mathbf{H}$ is stored in a standard cell-based memory [33][2] in such a way that each address can read a column of $\mathbf{H}$ in a single cycle. The VM unit first computes the matched filter $\mathbf{y}^{\text{MF}} = \mathbf{H}^H\mathbf{y}$. The same values of $\mathbf{y}^{\text{MF}}$ are needed for all the five ADMIN iterations and that is why they are stored in a register array.

The lower triangular matrix $\tilde{\mathbf{L}}$ is stored in another standard cell-based memory. The triangular memory is designed in such a way that it is possible to read an entire column or row of $\tilde{\mathbf{L}}$ in a single cycle. $\tilde{\mathbf{L}}$ is read row-wise to compute the $\tilde{\mathbf{L}}\mathbf{y}^{\text{MF}}$ and the output is stored in a temporary register array $\mathbf{t}$. The next step is to compute the element-wise multiplication between $\tilde{\mathbf{d}}$ and $\mathbf{t}$. The output from the multiplier array is written back to $\mathbf{t}$ unlike the previous computations. The triangular memory is read column-wise in next 16 or 32 cycles to compute $\tilde{\mathbf{L}}^H\mathbf{t}$ that results in $\hat{\mathbf{x}}$ for 16 users or 32 users respectively which is the output of VM unit.

C. Matched Filter Update Unit

The matched filter update (MFU) unit computes the $\mathbf{z}$ minimization and $\boldsymbol{\lambda}$-update steps of ADMM. The output values from the VM unit, $\hat{x}_i$, where $i = 1, 2, \ldots, M$, are obtained sequentially. Therefore, we choose a pipelined architecture for MFU. The architecture for the MFU unit is shown in Fig. 8.

The $\boldsymbol{\lambda}$ array is stored in a register array and initialized as zero. The projection unit compares the real and imaginary parts of the addition of $\hat{x}_i$ and λ_i and outputs $\hat{z}$. The difference between $\hat{z}$ and $\hat{x}_i$ is multiplied with the scaling parameter γ.

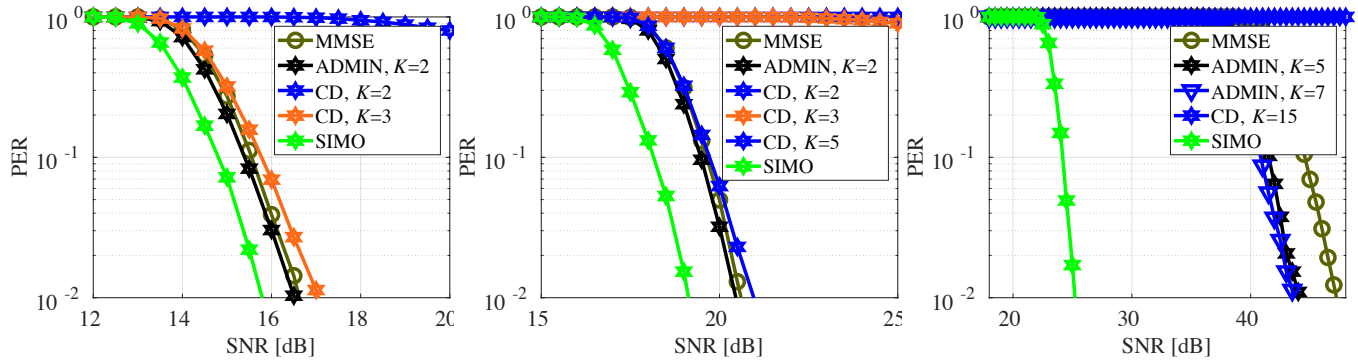
A shimming register is used after $\hat{x}_i$ to synchronize with $\hat{z}$. Similarly another shimming register is used to synchronize λ_i to add with $\gamma(\hat{x}_i - \hat{z})$ that results in an updated λ_{i+1}. The updated λ_{i+1} is stored back in the same register array designated for λ_i and they are used for the next iteration. The subtraction of $\hat{z}$ and λ_{i+1} is multiplied with the penalty parameter β. The output is added with the corresponding matched filter value y_i^{MF} for an updated y_i^u. The updated matched filter values $\mathbf{y}^u$ are stored in a register array and sent back to the VM unit to compute for the next ADMIN iteration.

D. Control Signals

A large number of control signals is required to reuse the multiplier array and the adder tree inside the VM unit. Firstly, four control signals are used as read enable, write enable, read address and write address for $\mathbf{H}$ memory. Secondly, there are several control signals associated with $\tilde{\mathbf{L}}$ memory. As both row and column-wise read is needed from $\tilde{\mathbf{L}}$, a control signal is used to select the reading mode. A two dimensional array of load signals are used to control read access of every individual value of $\tilde{\mathbf{L}}$. The writing of $\tilde{\mathbf{L}}$ memory block is done only row-wise and there is no need of a control signal to select the read mode. Another signal is used to select a row of $\tilde{\mathbf{L}}$ for write access. The control signals associated with $\mathbf{H}$ and $\mathbf{L}$ memory can be seen in Fig. 9.

In addition to write access control signals of $\mathbf{H}$ and $\tilde{\mathbf{L}}$ memory, there are single-bit enable signals, which control the write enable process of register arrays dedicated for $\mathbf{y}$ and $\tilde{\mathbf{d}}$. The matched filter register array, $\mathbf{y}^{\text{MF}}$ includes an array of write enable signals to write output from adder array to appropriate y_i^{MF} register. Similarly, an address signal is used to read the

[2]The authors does not have access to a 28 nm SRAM memory compiler. Therefore, the authors had to use the standard-cell based memory. A SRAM memory could potentially save a lot of logic gates for the $\mathbf{H}$ memory



(a) 128 BS antennas, 16 users and 256-QAM.

(b) 128 BS antennas, 32 users and 64-QAM.

(c) 32 BS antennas, 32 users and 256-QAM.

Fig. 5: Packet error rate (PER) for a massive MU-MIMO-OFDM system with rate-3/4 code and WINNER channels for 256-QAM.

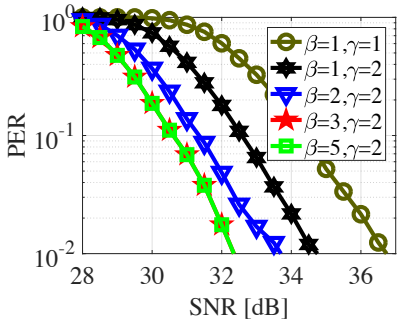


Fig. 6: Packet error rate (PER) for a MU-MIMO-OFDM system with 32 users, 32 antennas BS, rate-3/4 convolutional code and WINNER channel model and 64-QAM modulation scheme. The simulations used different values of β and γ.

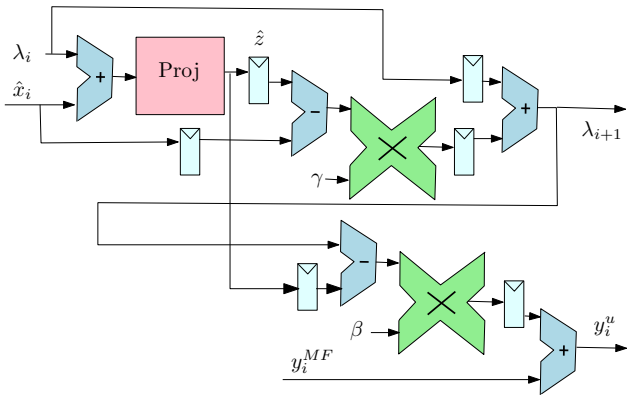


Fig. 8: MFU unit: Computes $\mathbf{z}$ minimization and $\boldsymbol{\lambda}$-update in pipelined fashion [1].

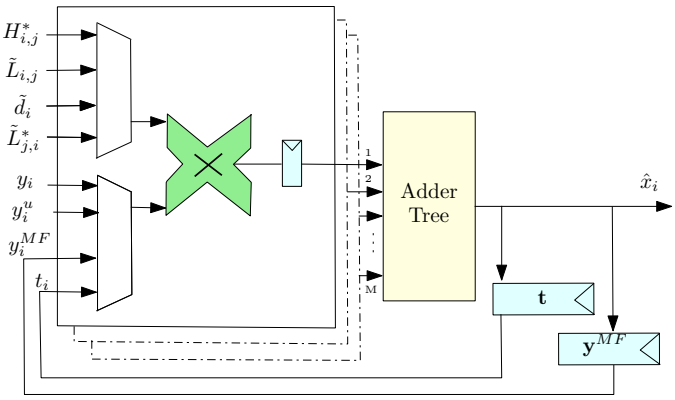


Fig. 7: VM unit: Computes vector-vector multiplication. It has $i = 1, 2, \ldots, M$ multiplier units in parallel. The adder tree sum the output of the multipliers [1].

correct value of y_i^{MF}. An array of load registers is also used for writing to different registers of $\mathbf{t}$. This register array is used to write either the output of the multiplier array or the output of the adder tree. Therefore, another control signal is used to select between the outputs of multiplier array and adder tree. The adder tree output is connected to all the registers of $\mathbf{t}$ and the array of load signals is used to activate the relevant write enable.

A couple of two bit signals are used to control the inputs of the multiplier array. These signals are used in multiplexers of VM unit, which can be seen on the left side Fig. 7. For readibility, these two bit signals are not shown in Fig. 9. The control mechanism of $\boldsymbol{\lambda}$ register array is very similar to that of $\mathbf{y}^{\text{MF}}$. It has an array of write enable signals and an address signal to write or read the suitable λ_i values. There are also several interrupt signals for debugging purpose, which are also not shown in Fig. 9. A list of the control signals associated with ADMIN and their functionalities are presented in Table II.

E. Scalable Architecture for More Antennas

As we use a time-shared and iterative architecture instead of a systolic array, the ADMIN implementation can be modified for higher numbers of BS antennas with relatively modest effort. We extend the ADMIN architecture for 32×32 to a BS with 64 antennas. To our advantage, the size of Gramian matrix $\mathbf{G}_{32\times32}$ and output of matched filter $\mathbf{y}^{\text{MF}}_{32\times1}$ remains the same as of 32 antenna BS. Thus, the operation of lines $6-16$ of Algorithm 1 is identical to 32×32 case and we can keep the datapath very similar to 32×32. The major difference is line 10 of Algorithm 1 due to the size difference of $\mathbf{H}_{64\times32}$ and $\mathbf{y}_{64\times1}$ compared to the square configurations. We use 32 complex multipliers for this architecture. Using 64 complex multipliers could enable us to compute the matched

TABLE II: List of Control Signals

Signal	Functionality
load_DinxI	Controls access of $\tilde{\mathbf{d}}$ inputs
load_yxI	Controls access of $\mathbf{y}$ inputs
load_yMFxI	Array of write enable signals for $\mathbf{y}_{MF}$ registers
load_tempxI	Array of write enable signals for $\mathbf{t}$ registers
selector1xI	Selects between $\mathbf{H}$, $\tilde{\mathbf{L}}$ and their conjugates as one input of multiplier array
selector2xI	Selects between $\mathbf{y}$, $\mathbf{y}^{\text{MF}}$, $\mathbf{t}$ as the other input of multiplier array
yMF_addrxI	Address to read appropriate $\mathbf{y}^{\text{MF}}$ value
tempinModexI	Selects between multiplier array and adder tree outputs
RAdrxI	Read address of $\mathbf{H}$ memory
RExI	Read enable of $\mathbf{H}$ memory
WExI	Write enable of $\mathbf{H}$ memory
WAdrxI	Write address of $\mathbf{H}$ memory
load_ResultxI	Array of write enable signals of $\tilde{\mathbf{L}}$
ReadModexI	Selects column or row wise read access
ControlxI	Activates column line of $\tilde{\mathbf{L}}$
load_lambdaxI	Array of write enable signals for $\boldsymbol{\lambda}$ registers
lambda_addrxI	Address to read appropriate $\boldsymbol{\lambda}$ value
yMFaddmodexI	Controls $\mathbf{y}^{\text{MF}}$ accumulation in 64×32 architecture
firstinvDonexI	Interrupt when MMSE / 1st iteration is done
secondinvDonexI	Interrupt when 2nd ADMIN iteration is done
resultDonexI	Interrupt when 5th ADMIN iteration is done
MFDonexI	Interrupt when matched filtering is done

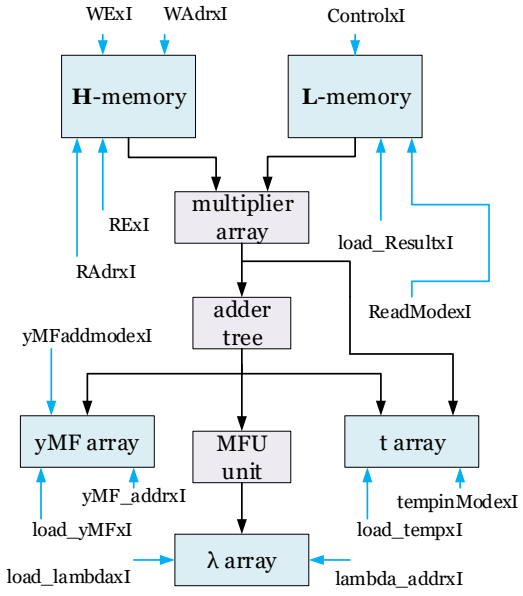


Fig. 9: A block diagram of control signal arrangements in ADMIN architecture. The blue arrows show the control signals.

filtering faster. However, only 32 is required for the rest of the process (lines $6-16$ of Algorithm 1) and the other 32 complex multipliers are not be utilized. The channel matrix $\mathbf{H}_{64\times32}$ is stored in two separate memory blocks $\mathbf{Ha}_{32\times32}$ and $\mathbf{Hb}_{32\times32}$ as $\mathbf{H} = \begin{bmatrix}\mathbf{Ha}\\ \mathbf{Hb}\end{bmatrix}$.

It is not essential to divide the channel matrix into different parts. However, dividing $\mathbf{H}$ memory in such a way leads to a scalable architecture. We read the first input of the multiplier array from $\mathbf{Ha}$. During these 32 cycles, we read half of $\mathbf{y}$, i.e., $\mathbf{y}_{32\times1}$ as the other input of the multiplier array. The adder tree outputs are sequentially stored in $\mathbf{y}^{\text{MF}}$ register array. In the next 32 cycles, we read $\mathbf{Hb}$ and other half of $\mathbf{y}$, multiply them and create 32 new outputs sequentially through adder array. These new 32 values are added with the earlier values of $\mathbf{y}^{\text{MF}}$. Due to the similarity of the datapaths, the rest of the control logic also can be the same as the 32×32 implementation. As extra 32 cycles are required to compute $\mathbf{y}^{\text{MF}}$, the rest of the control signals are delayed by 32 cycles too. The architecture can be easily scaled for a BS with more antennas. For example, in case of a 128 BS antennas, the channel matrix $\mathbf{H}_{128\times32}$ can be divided in four 32×32 matrix. Thus, we can add two more $\mathbf{H}_{32\times32}$ memories in the 64 BS antennas architecture and modify the control for matched filtering process. In a similar way, the ADMIN architecture can be extended to 128 or higher number of BS antennas. Note that, the design can also be scaled for binary phase-shift keying (BPSK) to 64-QAM modulation schemes. The LLRs are computed in a simplified form according to [26] for different modulations schemes. The LLR computation logic is quite small and does not have any noticable impact on the overall area or complexity of the design.

F. Dual Mode Architecture

The simplest way to design a dual mode ADMIN architecture, which supports both 32 and 64 BS antennas, is to reuse the 64×32 architecture and modify the write access of $\mathbf{y}^{\text{MF}}$ to support 32 BS antennas. In this way, we can avoid creating multiplexers with every control signals to select between 32 and 64 BS antenna mode. We use the same control signals of 64×32 architecture and only disable the accumulation in $\mathbf{y}^{\text{MF}}$ for 32 BS antennas mode. In this way, the multiplication of $\mathbf{Hb}$ and second half of $\mathbf{y}_{64\times1}$ are calculated, but the results are not used. This is similar to the NOP instructions used in a off-the-shelf processor architectures. Using this method, both 32×32 and 64×32 take equal number of cycles to compute the output. In other words, the 32 BS antennas will require 32 more cycles more in the dual mode architecture than a dedicated architecture solely for 32×32.

VI. Implementation Results and Comparison

A. FPGA Implementation

The ADMIN architecture is developed and optimized in VHDL on register-transfer level (RTL). We design two separate implementations for 16- and 32-user terminals and present the post place-and-route implementation results on a Xilinx Virtex-7 XC7VX690T FPGA in this subsection. Vivado default settings is used as the synthesis and implementation strategy. The default mode is selected for the `-flatten_hierarchy` option in Vivado design tool to keep the same top level hierarchy after synthesis. The 16-user design can reach 263.16 MHz. For 32-user designs, the maximum clock frequency is 232.55 MHz and 222.22 MHz for 32 and 64 BS antennas respectively.

The throughput and latency of different iterations for the FPGAs are provided in Table III. The throughput decreases linearly with respect to the number of iterations K, because the number of operations remains the same in each iteration of ADMIN. ADMIN requires $K = 5$ iterations to achieve the desired PER shown in Figs. 3 and 4. In the first 70 cycles, the

TABLE III: Throughput and latency of ADMIN FPGA implementation for K iterations

		K=1	K=2	K=3	K=4	K=5
16×16	Cycles	70	109	148	187	226
	Throughput (Mbps)	360	231	170	135	111
	Latency (μs)	0.26	0.41	0.56	0.71	0.85
32×32	Cycles	134	205	276	347	418
	Throughput (Mbps)	332	217	161	128	106
	Latency (μs)	0.57	0.88	1.18	1.49	1.79
64×32	Cycles	198	269	340	411	482
	Throughput (Mbps)	215	159	125	103	88
	Latency (μs)	0.89	1.21	1.53	1.85	2.17

TABLE IV: Component wise breakdown of ADMIN in FPGA

		LUT slices	FF slices	DSP
16×16	$\mathbf{H}$ memory	2577	9300	0
	$\mathbf{L}$ memory	4608	4320	0
	VM unit	1620	3168	64
	MFU unit	324	1440	4
	Others	4623	13	8
	Total	13392	18241	76
32×32	$\mathbf{H}$ memory	8641	30782	0
	$\mathbf{L}$ memory	10000	14880	0
	VM unit	1740	5280	128
	MFU unit	448	2160	4
	Others	11220	44	8
	Total	32049	53146	140
64×32	$\mathbf{H1}$ memory	10368	36962	0
	$\mathbf{H2}$ memory	10945	36962	0
	$\mathbf{L}$ memory	10780	17856	0
	VM unit	2592	7488	128
	MFU unit	540	2592	4
	Others	13006	55	8
	Total	48231	101915	140

16-user architecture computes the first ADMIN iteration that provides the MMSE estimates. Here, 16 cycles are used for storing the inputs to $\mathbf{H}$ and $\tilde{\mathbf{L}}$ memory. The architecture needs 226 cycles to compute $K = 5$ ADMIN iterations that results in a throughput of 111.71 Mbps. The first 134 cycles computes the first ADMIN iteration and provides the MMSE estimates for the 32-user configuration. Here, 32 cycles are used for storing the $\mathbf{H}$ and $\tilde{\mathbf{L}}$. The architecture computes five ADMIN iterations that results in a throughput of 106.56 Mbps. The 32-user and 64-antenna architecture compute five iterations with a throughput of 88 Mbps. An FPGA device can be reconfigured on-field by programming with different bitstreams. Therefore, an FPGA device configured for 32 antennas can be updated on-field with a new bitstream for 64 antennas. Thus, there is no crucial need for a dual mode architecture in FPGA that supports both 32 and 64 antennas.

In Table IV, the resource utilization of the ADMIN variants on a Virtex-7 FPGA is presented. Even though the number of elements in $\mathbf{L}$ memory is roughly half of $\mathbf{H}$ memory, it can be seen from Table IV that $\mathbf{L}$ memory uses nearly same or more LUTs. This is due to the control logic used in for accessing both row and columns of the triangular register bank dedicated for $\mathbf{L}$. The 16-user configuration uses 16 complex multipliers which are mapped to the 64 real multipliers available in the DSP elements. Therefore, the LUT slice usage of the VM unit is relatively lower than other units. Similarly, the 32-user configurations uses 128 DSP elements because it uses 32 complex multipliers. The $Others$ section contains control logics, counters etc.

A comparison with different state-of-the-art data detectors is presented in Table V. The most popular configuration for the FPGA implementations use 8 users and 128 BS antennas and thus, our design is not really comparable. Our goal is to compare ADMIN with other designs where the ratio between BS antennas and number of users is equal or less than 4. For this reason, we compare our design with two variants of TASER [28] and two variants of CD detectors [11], [30]. All implementations in Table V used Xilinx Virtex-7 XC7VX690T FPGA, which gives us a fair comparison between the implementations. In [28], the TASER FPGA implementation results are provided for different BPSK and QPSK users per time slot. We take 64-users BPSK or 32-user QPSK TASERs to compare with our detectors. The resource utilization in TASER increases quadratically with the size of the systolic array. Therefore, while the number of LUTs in 8-user BPSK TASER is only 4790, the number of LUTs of 64-user BPSK TASER is 149942. Thus our design, provides a significantly higher throughput/slices for all the configurations. The modulation scheme plays a crucial role in the lower throughput of TASER as it can support only BPSK and QPSK.

A MMSE based optimized CD was implemented in [11] which was extended using box-relaxation in the journal version [30]. Both implementations provide very high throughput due to the 24 pipelined stages and the authors claim that the throughput of 15.81 Mbps can be multiplied 24 times [11]. Thus, the CD implementations provide about $3.5\times$ higher throughput than our implementation. As the CD results of Table V is for 8 users and 32 antennas, the number of slices used in the implementation is significantly lower than our implementation. Thus, the scaled throughput per slices is also significantly higher in CD. In addition, the results provided in Table V for CD only use 3 iterations, where it needs around $K = 15$ iterations to reach ADMIN performance (Figure 4c). Therefore, the throughput of CD will decrease approximatey $5\times$ to get a similar error rate performance as of ADMIN. Similarly, the PER results in Figs. 3 and 4 demonstrate that $K = 5$ iterations of ADMIN performs better than $K = 10$ to $K = 15$ iterations of TASER. Consequently, the scaled throughput of these algorithms should be based on higher number of iterations to fairly compare to ADMIN. In the last row of Table V, we normalize the results for number of users. The difference between ADMIN and TASER; and the difference between CD and ADMIN becomes more pronounced after this scaling.

B. ASIC Implementation

The ADMIN architecture is developed and optimized in VHDL on RTL for the ASIC design. The architecture is synthesized using Synopsys DC with a 28 nm CMOS standard cell library. For a 16-user and 16-antenna BS, ADMIN achieves a maximum clock frequency of 714 MHz and takes an area of 0.225 mm^2 which equals to 460.6 k gate equivalents. For 32-user architectures, ADMIN achieves 625 MHz and 606 MHz

TABLE V: Comparison of FPGA Implementations for MU-MIMO symbol detection

	Proposed			[28]		[11]	[30]
No. of users	16	32	32	64	32	8	8
BS antenna & users ratio	1	1	2	2	4	4	4
Algorithm	ADMIN	ADMIN	ADMIN	TASER	TASER	CD	CD
Iteration	5	5	3	1	3	3	5
Modulation scheme	64-QAM	64-QAM	64-QAM	BPSK	QPSK	64-QAM	64-QAM
Preprocessing included	No	No	No	No	No	Yes	Yes
Clock freq. [MHz]	263	232	222	111		262	261
LUT slices	13392	32049	48231	149942		5909	6059
FF slices	18241	53146	101915	91829		15148	10704
DSP	76	140	140	2208		195	198
Throughput [Mbps]	111.71	106.56	125	98		379	378
Throughput/slices^a Mbps/K slices	8342	3324	2591	653		64139	62386
Throughput/(slices^a $\times$ users) Mbps/(K slices$\times$ users)	521	104	81	10	20	8017	7798

^{a}Summation of LUT and FF slices.

TABLE VI: Throughput and latency of ADMIN ASIC implementation for K iterations

		K=1	K=2	K=3	K=4	K=5
16×16	Cycles	70	109	148	187	226
	Throughput (Mbps)	979	629	463	366	303
	Latency (μs)	0.09	0.15	0.2	0.26	0.31
32×32	Cycles	134	205	276	347	418
	Throughput (Mbps)	895	585	434	345	287
	Latency (μs)	0.21	0.32	0.44	0.55	0.66
64×32	Cycles	198	269	340	411	482
	Throughput (Mbps)	588	432	342	283	241
	Latency (μs)	0.32	0.44	0.56	.67	.79

and takes an area of 0.554 mm^2 and 0.688 mm^2 for 32 and 64 BS antennas respectively. The worst critical path goes through the multiplier array to the temporary register, $\mathbf{t}$ for all implementations. The throughput and latency of different ADMIN iterations for the ASICs are provided in Table VI. The 16-user architecture needs 226 cycles to compute $K = 5$ ADMIN iterations that results in a throughput of 303 Mbps. The 32-user architectures requires 418 and 482 cycles to compute $K = 5$ ADMIN iterations that results in a throughput of 287 and 241 Mbps for 32 and 64 BS antennas respectively. As the dual mode architecture's control and data path are very similar to that of 64×32, a separate implementation results for the dual mode is not necessary.

The synthesized RTL for all the configurations are placed and routed with Cadence Encounter tool. In Fig. 10, we present only the layout diagram of a 32-user and 64-antenna ADMIN. The figure shows the module placement of the main components of ADMIN implementations. It can be seen from Fig. 10 that the standard cell placements of ASIC are centered around the VM unit. The VM unit is communicating with other major parts such as **Ha** memory, **Hb** memory, **L** memory and MFU unit. Most of the major parts of the ASIC have some connection to the I/O ring because they each have connection to the top level inputs or outputs of the design.

A breakdown for different components for the ASICs are shown in Table VII. The **H** and **L** memory took a significant portion of the whole ASIC. Synopsys DC is able to optimize and save resources for the multiplier array as the number did not increase linearly. The other part of the VM unit consists of several intermediate register banks.

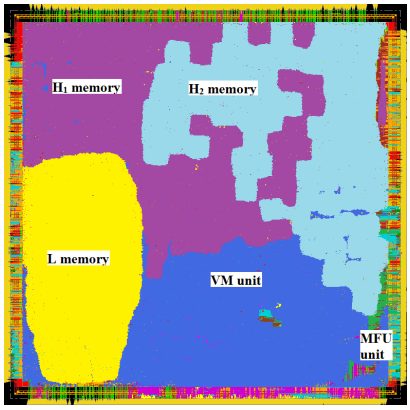


Fig. 10: Layout of the ADMIN ASIC for 64 BS antenna and 32 users. The blue, violet, yellow and green palettes represent the VM unit, H memory, L memory and the MFU units respectively.

TABLE VII: Component wise breakdown of ASICs (k gates)

	16×16	32×32	64×32
H memory	153.389	401.149	-
Ha memory	-	-	369.808
Hb memory	-	-	369.928
L memory	88.792	366.331	232.780
VM unit: mul. array	128.724	196.77	187.238
VM unit: adder tree	2.741	5.614	5.733
VM unit: others	59.405	117.782	167.112
MFU unit	22.618	44.58	28.096
Total	460.599	1132.226	1405.640

Table VIII presents the average power consumption distribution of the ASICs. The power consumption is measured by capturing signal activity traces of the post place-and-route design during the detection. The multiplier array consumes the majority of the VM unit. The rest of the VM slice consists of the consumption by adder tree, intermediate registers etc. The *Other* row in the table represents the consumption of the clock tree buffer cells.

In Table IX, our ADMIN ASICs are compared to the state-

TABLE VIII: Distribution of power of the ASICs

	16×16	32×32	64×32
H memory	19%	37%	-
Ha memory	-	-	18%
Hb memory	-	-	18%
L memory	10%	19%	15%
VM unit: multiplier array	42%	31%	34%
Rest of the VM unit	12%	7%	9%
MFU unit	4%	4%	2%
Others	13%	2%	4%
Total	100%	100%	100%

of-the-art massive MIMO detection. The throughput and area of the implementations are normalized to 28nm technology as

$$t \sim 1/s, \quad A =\sim 1/s^2, \quad P_{\text{dyn}} \sim (1/s)(V_{\text{dd}}/V'_{\text{dd}})^2$$

s, t, A and P_{dyn} are scaling factor, throughput, area and power respectively. This is a fairly standard practice to calculate the area and power efficiency [35]. The 16-user architecture provides an area efficiency of 1.39 Mbps per kGE and energy efficiency of 3.56 Gbps per W. The 32-user architectures provide an area efficiency of 0.7810 and 0.7896 Mbps per kGE and energy efficiency of 2.37 and 2.53 Gbps per W for 32 and 64 BS antennas respectively. We compare ADMIN ASICs with implementations where the ratio between BS antennas and users is equal or less than 4.

In [28], the authors provided three separate TASER ASICs for 8, 16 and 32 BPSK users. We compare the 32-user TASER implementation with our designs in Table IX. The area and energy efficiency of TASER implementations are lower compared to all our architectures. The area and energy efficiency of 32-user ADMIN implementations are approximately $6\times$ and $1.5\times$ higher than 32-user TASER implementation. The throughput of TASER will also be significantly lower for higher numbers of iterations which is required for a comparable throughput of ADMIN. For example, TASER with $K = 20$ iterations can reach close to ADMIN's performance as shown in Fig. 3a and in Fig. 4a, which will further decrease TASER area and energy efficiency. A reason for low throughput lies in the fact that TASER only supports BPSK and QPSK and not suitable for higher order modulation.

In [12], a VLSI architecture for NOnParametric Equalizer (NOPE) massive MIMO detector is presented. The architecture is presented for 16 users and 64 BS antennas. NOPE architecture is synthesized in 28 nm CMOS and achieves a throughput of 920 Mbps. We compare NOPE with ADMIN in Table IX. The scaled throughput of NOPE is slightly better than our 16-user ASIC. However, NOPE architecture results are optimistic as they are based on synthesis while our results are based on post-layout routing and placement. In addition, no power results of the NOPE architecture have been presented. We would like to note that NOPE uses $K = 7$ iterations and with $K = 5$ iterations like 16-user ADMIN, the area efficiency of NOPE will be higher. On the other hand, NOPE results are based on 256-QAM while ADMIN results are based on 64-QAM. Hence, the area efficiency of NOPE will be lower with similar modulation order as ADMIN.

In [34], an ASIC for channel hardening-exploiting message passing (CHEMP) detector is presented for 32 users and 128 BS antennas. We compare the CHEMP results with 32-user ADMIN implementations in Table IX. CHEMP ASIC outperforms ADMIN architectures significantly in terms of area and energy efficiency. The design provides approximately $11\times$ and $8\times$ better area and energy efficiency respectively than ADMIN. However, CHEMP ASIC has been highly optimized to fulfill the BER requirements for an i.i.d. Rayleigh fading channel. The reason is the original CHEMP algorithm was also developed based on the assumption that the channel matrix is i.i.d. [36]. A highly optimized detector realization for such a simplistic channel model might be impractical for real life and thus, there is a question mark on MPD detector's performance in a realistic channel model.

VII. Conclusions

We have proposed ADMIN, a novel data-detection algorithm and a corresponding VLSI architecture. The algorithm outperforms linear MMSE equalization in terms of PER by a large margin when the ratio of numbers between BS antennas and users is rather small (two or less). Our ADMIN architecture also provides promising results for 16-user and 32-user MIMO detectors in terms of area and energy efficiency. Thus, ADMIN enables a realistic large-scale MU-MIMO detector implementation for the fifth generation (5G) communication systems.

References

[1] S. Shahabuddin, M. Juntti, and C. Studer, "ADMM-based infinity norm detection for large MU-MIMO: Algorithm and VLSI architecture," in *Proc. IEEE Int'l Conf. Circuits Syst. (ISCAS)*, May 2017, pp. 1–4.

[2] T. L. Marzetta, "Noncooperative cellular wireless with unlimited numbers of base station antennas," *IEEE Trans. Wireless Commun.*, vol. 9, no. 11, pp. 3590–3600, Nov. 2010.

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

[4] L. Lu, G. Y. Li, A. L. Swindlehurst, A. Ashikhmin, and R. Zhang, "An Overview of Massive MIMO: Benefits and Challenges," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 742–758, Oct 2014.

[5] M. Wu, B. Yin, G. Wang, C. Dick, J. Cavallaro, and C. Studer, "Large-Scale MIMO Detection for 3GPP LTE: Algorithm and FPGA Implementation," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 916–929, Oct. 2014.

[6] K. V. Vardhan, S. K. Mohammed, A. Chockalingam, and B. S. Rajan, "A Low-Complexity Detector for Large MIMO Systems and Multicarrier CDMA Systems," *IEEE J. Sel. Topics Signal Process.*, vol. 26, no. 3, pp. 473–485, April 2008.

[7] N. Srinidhi, S. K. Mohammed, A. Chockalingam, and B. S. Rajan, "Low-complexity near-ML decoding of large non-orthogonal STBCs using reactive tabu search," in *Proc. IEEE Int'l Symp. Inf. Theory (ISIT)*, June 2009, pp. 1993–1997.

[8] M. Suneel, P. Som, A. Chockalingam, and B. S. Rajan, "Belief propagation based decoding of large non-orthogonal STBCs," in *Proc. IEEE Int'l Symp. Inf. Theory (ISIT)*, June 2009, pp. 2003–2007.

[9] Y. Hu, Z. Wang, X. Gaol, and J. Ning, "Low-complexity signal detection using CG method for uplink large-scale MIMO systems," in *Proc. IEEE Conf. Commun. Sys.*, Nov. 2014, pp. 477–481.

[10] B. Yin, M. Wu, J. R. Cavallaro, and C. Studer, "Conjugate gradient-based soft-output detection and precoding in massive MIMO systems," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2014, pp. 3696–3701.

[11] M. Wu, C. Dick, J. R. Cavallaro, and C. Studer, "FPGA design of a coordinate descent data detector for large-scale MU-MIMO," in *Proc. IEEE Int'l Conf. Circuits Syst. (ISCAS)*, May 2016, pp. 1894–1897.

TABLE IX: Comparison of data detector ASIC for massive MU-MIMO systems

	Proposed			[28]	[12]	[34]
Technology [nm]	28	28	28	40	28	40
No. of users	16	32	32	32	16	32
BS antenna & users ratio	1	1	2	4	4	4
Modulation Scheme	64-QAM	64-QAM	64-QAM	BPSK	256-QAM	256-QAM
Supply Voltage [V]	1.0	1.0	1.0	1.1	-	0.9
Clock freq. [MHz]	714	625	606	454	800	425
Core area[a] [mm^2]	0.225	0.554	0.688	1.382 (0.677[a])	0.28	0.58 (0.284[a])
Cell area[b] [kGE]	218.41	364.7	433.124	1382.76	571.9	459.558
Preprocessing Included	No	No	No	No	No	No
Results	Post-layout	Post-layout	Post-layout	Post-layout	Synthesis	ASIC
Algorithm	ADMIN	ADMIN	ADMIN	TASER	NOPE	CHEMP
Iteration	5	5	3	3	7	-
Throughput [Mb/s]	303	287	342	121	920	2760
Power [W]	0.085	0.121	0.135	0.216	-	0.220
Scaled throughput[a] [Mb/s]	-	-	-	173	-	3942
Normalized area efficiency[a][Mb/(s×kGE)]	1.39	0.7810	0.7896	0.1251	1.6	8.58
Normalized energy efficiency[a][Gb/(s×W)]	3.56	2.37	2.53	1.55	-	20.85

[a]Technology scaling to 28 nm assuming $A \sim 1/s^2$, $t \sim 1/s$ and $P_{\text{dyn}} \sim (1/s)(V_{\text{dd}}/V'_{\text{dd}})^2$.
[b]Excluding the gate count of memories.

[12] C. Jeon, G. Mirza, R. Ghods, A. Maleki, and C. Studer, "VLSI design of a nonparametric equalizer for massive MU-MIMO," in *Proc. IEEE Conf. Rec. Asilomar Conf. Signals, Sys., and Comp.*, Oct 2017, pp. 1504–1508.

[13] S. Shahabuddin, M. A. Albreem, M. S. Shahabuddin, Z. Khan, and M. Juntti, "FPGA Implementation of Stair Matrix based Massive MIMO Detection," in *Proc. IEEE Latin America Symp. Circuits Syst. (LASCAS)*, 2021.

[14] M. A. Albreem, M. Juntti, and S. Shahabuddin, "Massive MIMO detection techniques: A survey," *IEEE Commun. Surveys Tuts.*, vol. 21, no. 4, pp. 3109–3132, 2019.

[15] S. Shahabuddin, M. H. Islam, M. S. Shahabuddin, M. A. Albreem, and M. Juntti, "Matrix Decomposition for Massive MIMO Detection," in *Proc. IEEE Nordic Circuits Syst. Conf. (NORCAS)*, 2020, pp. 1–6.

[16] O. Gustafsson, E. Bertilsson, J. Klasson, and C. Ingemarsson, "Approximate neumann series or exact matrix inversion for massive MIMO?" in *2017 IEEE 24th Symposium on Computer Arithmetic (ARITH)*, July 2017, pp. 62–63.

[17] H. Prabhu, J. N. Rodrigues, L. Liu, and O. Edfors, "3.6 A 60 pJ/b 300 Mb/s 128x8 Massive MIMO precoder-detector in 28 nm FD-SOI," in *Proc. IEEE Int'l Conf. Solid-State Circuits Conf. (ISSCC)*, Feb 2017, pp. 60–61.

[18] W. Tang, H. Prabhu, L. Liu, V. wall, and Z. Zhang, "A 1.8 Gb/s 70.6 pJ/b 128×16 link-adaptive near-optimal massive MIMO detector in 28 nm UTBB-FDSOI," in *Proc. IEEE Int'l Conf. Solid-State Circuits Conf. (ISSCC)*, Feb 2018, pp. 224–226.

[19] E. Bjrnson, E. G. Larsson, and T. L. Marzetta, "Massive MIMO: Ten myths and one critical question," *IEEE Commun. Mag.*, vol. 54, no. 2, pp. 114–123, 2016.

[20] E. Björnson, L. Sanguinetti, H. Wymeersch, J. Hoydis, and T. L. Marzetta, "Massive MIMO is a reality What is next?: Five promising research directions for antenna arrays," *Digital Signal Processing*, vol. 94, pp. 3–20, 2019.

[21] A. Paulraj, R. Nabar, and D. Gore, *Introduction to Space-Time Wireless Communications*. Cambridge Univ. Press, 2003.

[22] W.-K. Ma, T. N. Davidson, K. M. Wong, Z.-Q. Luo, and P.-C. Ching, "Quasi-maximum-likelihood multiuser detection using semi-definite relaxation with application to synchronous CDMA," *IEEE Trans. Signal Process.*, vol. 50, no. 4, pp. 912–922, Apr. 2002.

[23] P. H. Tan, L. K. Rasmussen, and T. J. Lim, "Constrained maximum-likelihood detection in CDMA," *IEEE Trans. Commun.*, vol. 49, no. 1, pp. 142–153, Jan. 2001.

[24] C. Jeon, A. Maleki, and C. Studer, "On the performance of mismatched data detection in large MIMO systems," in *Proc. IEEE Int'l Symp. Inf. Theory (ISIT)*, Jul. 2016, pp. 180–184.

[25] S. Boyd, N. Parikh, E. Chu, B. Peleato, and J. Eckstein, "Distributed Optimization and Statistical Learning via the Alternating Direction Method of Multipliers," *Foundations and Trends in Machine Learning*, 2010.

[26] I. B. Collings, M. R. G. Butler, and M. McKay, "Low complexity receiver design for MIMO bit-interleaved coded modulation," in *Proc. IEEE Int'l Symp. Sprd Spec. Tech. and App.*, Aug. 2004, pp. 12–16.

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

[28] O. Castañeda, T. Goldstein, and C. Studer, "Data Detection in Large Multi-Antenna Wireless Systems via Approximate Semidefinite Relaxation," *IEEE Trans. Circuits Syst. I*, pp. 2659–2662, Dec. 2016.

[29] L. Hentilä, P. Kyösti, M. Käske, M. Narandzic, and M. Alatossava, "MATLAB Implementation of the WINNER Phase II channel model ver. 1.1, Dec. 2007."

[30] M. Wu, C. Dick, J. R. Cavallaro, and C. Studer, "High-throughput data detection for Massive MU-MIMO-OFDM using Coordinate Descent," *IEEE Trans. Circuits Syst. I*, Dec. 2016.

[31] B. Yin, M. Wu, G. Wang, C. Dick, J. R. Cavallaro, and C. Studer, "A 3.8 Gb/s large-scale MIMO detector for 3GPP LTE-Advanced," in *Proc. IEEE Int'l Conf. Acoust., Speech and Signal Process. (ICASSP)*, May 2014, pp. 3907–3911.

[32] E. Ghadimi, A. Teixeira, I. Shames, and M. Johansson, "Optimal Parameter Selection for the Alternating Direction Method of Multipliers (ADMM): Quadratic Problems," *IEEE Trans. Autom. Control*, vol. 60, no. 3, pp. 644–658, March 2015.

[33] P. A. Meinerzhagen, J. N. Rodrigues, and A. P. Burg, "Standard-Cell Based Memories (SCMs): from Sub-VT to Error-Resilient Systems," in *Proc. IEEE Int'l Conf. Solid-State Circuits Conf. (ISSCC)*, no. EPFL-CONF-178203, 2012.

[34] W. Tang, C. H. Chen, and Z. Zhang, "A 0.58mm2 2.76gb/s 79.8pj/b 256-QAM massive MIMO message-passing detector," in *Proc. IEEE Int'l Conf. Circuits Syst. (ISCAS)*, June 2016, pp. 1–2.

[35] B. Razavi, "Design of Analog CMOS Integrated Circuits, McGraw-Hill Higher Education," 2001.

[36] T. Narasimhan and A. Chockalingam, "Channel hardening-exploiting message passing (CHEMP) receiver in large-scale MIMO systems," *IEEE J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 847–860, Oct. 2014.