VLSI Designs for Joint Channel Estimation and Data Detection in Large SIMO Wireless Systems

Oscar Castañeda, Tom Goldstein, and Christoph Studer

***Abstract*—Channel estimation errors have a critical impact on the reliability of wireless communication systems. While virtually all existing wireless receivers separate channel estimation from data detection, it is well known that joint channel estimation and data detection (JED) significantly outperforms conventional methods at the cost of high computational complexity. In this paper, we propose a novel JED algorithm and corresponding VLSI designs for large single-input multiple-output (SIMO) wireless systems that use constant-modulus constellations. The proposed algorithm is referred to as PRojection Onto conveX hull (PrOX) and relies on biconvex relaxation (BCR), which enables us to efficiently compute an approximate solution of the maximum-likelihood JED problem. Since BCR solves a biconvex problem via alternating optimization, we provide a theoretical convergence analysis for PrOX. We design a scalable, high-throughput VLSI architecture that uses a linear array of processing elements to minimize hardware complexity. We develop corresponding field-programmable gate array (FPGA) and application-specific integrated circuit (ASIC) designs, and we demonstrate that PrOX significantly outperforms the only other existing JED design in terms of throughput, hardware-efficiency, and energy-efficiency.**

***Index Terms*—FPGA and ASIC designs, joint channel estimation and data detection (JED), large single-input multiple-output (SIMO) wireless systems, biconvex relaxation (BCR).**

I. Introduction

WIRELESS communication with a large number of antennas at the base-station (BS) will play a major role in fifth-generation (5G) systems. By equipping the BS with hundreds or thousands of antenna elements, large wireless systems enable fine-grained beamforming and, hence, improved spectral-efficiency within each cell compared to traditional communication systems that use a small number of BS antennas [2]–[7]. All these advantages come at significantly increased signal-processing complexity at the BS [8], which necessitates the development of high-performance transceiver algorithms that scale well to large BS array sizes and can be implemented in very-large scale integration (VLSI) circuits at low costs and in an energy-efficient manner.

A. The Importance of Accurate Channel State Information

Due to the fine-grained nature of beamforming in such large wireless systems, the acquisition of accurate channel state information (CSI) at the BS is critical for reliable, high-throughput data transmission. In particular, accurate channel state information is not only required in the uplink (to coherently detect data transmitted from the users to the BS), but also required for beamforming or precoding in the downlink (to precisely focus the transmit energy towards the users and to mitigate multi-user interference). However, the fading nature of wireless channels as well as pilot contamination, i.e., channel training that may be contaminated by pilots or data transmission of users communicating in adjacent cells [2], render the acquisition of accurate CSI without a significant channel-training overhead a difficult task.

Most proposed large wireless systems rely on time-division duplexing (TDD) and perform pilot-based channel training during the uplink phase [3], [5]. It is, however, well known that the quality of CSI can be improved significantly by *joint channel estimation and data detection* (JED), which is capable of approaching the performance of idealistic systems with perfect CSI [9], [10]. While a few JED algorithms have been proposed for traditional, small-scale wireless systems [9]–[16], their computational complexity is typically high and not much is known about their efficacy for systems with hundreds or thousands of antennas. Moreover, with the exception of the recently proposed VLSI designs in [17], no hardware implementations for JED have been described in the literature.

B. Contributions

In this paper, we propose a novel, computationally efficient and near-optimal JED algorithm for large SIMO wireless systems, and we develop corresponding very-large scale integrated (VLSI) designs. Our contributions are summarized as follows:

- We use biconvex relaxation (BCR) [18] to develop PRojection Onto conveX hull (PrOX), a novel algorithm that achieves near-optimal JED performance at low complexity.
- We theoretically analyze the convergence of PrOX, which solves a biconvex problem via alternating optimization.
- We introduce an approximation that significantly reduces the preprocessing complexity of PrOX at virtually no loss in terms of error-rate performance.
- We provide simulation results that showcase the robustness of PrOX to a broad range of system parameters.
- We develop a scalable VLSI architecture for PrOX that uses a linear array of processing elements (PEs) to achieve high-throughput at low hardware complexity.
- We show reference FPGA and ASIC implementation results, and compare our designs to that of the recently-reported JED implementations in [17].

O. Castañeda and C. Studer are with the School of Electrical and Computer Engineering, Cornell University, Ithaca, NY; e-mail: oc66@cornell.edu, studer@cornell.edu; web: http://vip.ece.cornell.edu

T. Goldstein is with the Department of Computer Science, University of Maryland, College Park, MD; e-mail: tomg@cs.umd.edu

A short version of this paper summarizing the PrOX FPGA design has been presented at the IEEE Int. Symp. on Circuits and Systems (ISCAS) 2017 [1].

The MATLAB simulator for PrOX used in this paper is available on GitHub: https://github.com/VIP-Group/PrOX.

Our results demonstrate that PrOX provides near-optimal JED at significantly lower complexity than existing reference designs.

C. Related Relevant Results

While a considerable number of algorithms and VLSI designs have been proposed for small- and large-scale multi-antenna wireless systems that separate channel estimation and data detection (see, e.g., [8], [17], [19] and the references therein), only a few of results have been proposed for JED. Sphere-decoding (SD) algorithms have been proposed to perform *exact* maximum-likelihood (ML) JED in SIMO and MIMO systems that use a small number of time slots [9]–[12], [20], [21]. Unfortunately, the complexity of SD methods quickly becomes prohibitive for larger dimensional problems [22], [23] and approximate linear methods, which are widely used for coherent data detection in massive MIMO systems [8], cannot be used for JED (see Section II-B for the reasons). Very recently, a handful of approximate JED algorithms have been proposed for large wireless systems [17], [24]–[26]. To the best of our knowledge, reference [17] describes the only VLSI design of a JED algorithm reported in the open literature. While this design achieves near-ML-JED performance, the algorithm relies on semidefinite relaxation (SDR), which lifts the dimensionality of the problem to the square of the number of time slots causing high hardware complexity. In contrast to all these results, PrOX avoids lifting while achieving near-ML-JED performance and can be implemented efficiently in VLSI, even for scenarios that operate with a large number of time slots.

Another line of related work investigates data detection algorithms that are robust to imperfect CSI [27], [28]. The idea is to statistically model channel estimation errors and to solve suitably adapted data detection problems. The resulting algorithms, however, (i) do not achieve the error-rate performance of JED as they are not taking into account all the received information (i.e., from training and data symbols received over multiple time slots) and (ii) do not improve the channel estimate itself—the latter is critical in TDD systems that perform beamforming in the downlink using acquired CSI in the uplink [2]–[5]. In contrast, we propose JED algorithms that (i) approach optimal error-rate performance as they jointly process all received information and (ii) generate improved channel estimates that can be used for beamforming.

D. Notation

Lowercase and uppercase boldface letters stand for column vectors and matrices, respectively. For the matrix $\mathbf{A}$, the Hermitian is $\mathbf{A}^H$ and the kth row and ℓth column entry is $A_{k,\ell}$. For the vector $\mathbf{a}$, the kth entry is a_k. The Euclidean norm of $\mathbf{a}$, the Frobenius norm, and the spectral norm of $\mathbf{A}$ are denoted by $\|\mathbf{a}\|_2$, $\|\mathbf{A}\|_F$, and $\|\mathbf{A}\|$, respectively. The real and imaginary parts of the vector $\mathbf{a}$ are $\Re(\mathbf{a})$ and $\Im(\mathbf{a})$, respectively.

E. Paper Outline

The rest of the paper is organized as follows. Section II describes the system model as well as ML-optimal JED. Section III introduces the PrOX algorithm and provides a theoretical convergence analysis. Section IV details the VLSI architecture for PrOX. Section V shows error-rate performance and VLSI implementation results, and compares PrOX to existing FPGA and ASIC designs. We conclude in Section VI.

II. System Model and ML-Optimal JED

We now introduce the considered SIMO system model and develop the associated ML-JED problem.

A. System Model

We study a (potentially large) SIMO wireless uplink system in which a single-antenna user transmits data over $K+1$ time slots to a BS with B antennas. We consider the standard block-fading, narrow-band[1] channel model with the following input-output relation [9]–[11], [20]:

$$\mathbf{Y} = \mathbf{h}\mathbf{s}^H + \mathbf{N}. \tag{1}$$

Here, the matrix $\mathbf{Y} \in \mathbb{C}^{B\times(K+1)}$ contains the B-dimensional receive vectors for the $K+1$ time slots, i.e., $\mathbf{Y} = [\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{K+1}]$ with $\mathbf{y}_k \in \mathbb{C}^B$ and $k = 1, 2, \ldots, K+1$, $\mathbf{h} \in \mathbb{C}^B$ is the unknown SIMO channel vector (assumed to remain constant over the $K+1$ time slots), $\mathbf{s} \in \mathcal{O}^{K+1}$ contains the transmitted data symbols for all $K+1$ time slots, and $\mathbf{N} \in \mathbb{C}^{B\times(K+1)}$ models i.i.d. complex circularly-symmetric Gaussian noise with variance N_0 per entry. In the remainder of the paper, we assume constant-modulus constellations, i.e., $|s| = \sigma$ for all $s \in \mathcal{O}$ and a fixed $\sigma > 0$.

Remark 1. *The SIMO case is relevant in many rural deployment scenarios in which only a few users are active at a time [30]. The assumption of having constant-modulus constellations limits our results to low-rate modulation schemes, such as BPSK and PSK constellations. While the (multi-user) MIMO case and higher-order QAM modulation schemes would be of interest in more general deployment scenarios, the associated ML-JED problem is significantly more challenging to solve, and requires different, more complex, and complicated algorithms; see, e.g., [12], [21] for more details—we are planning to address this scenario in the future.*

B. Joint Channel Estimation and Data Detection (JED)

Let $\mathbf{h}$ be a deterministic—but unknown—channel vector with unknown prior statistics. Then, one can formulate the following ML-JED problem [11]:

$$\{\hat{\mathbf{s}}^{\text{JED}}, \hat{\mathbf{h}}\} = \underset{\mathbf{s}\in\mathcal{O}^{K+1},\, \mathbf{h}\in\mathbb{C}^B}{\arg\min} \left\|\mathbf{Y} - \mathbf{h}\mathbf{s}^H\right\|_F. \tag{2}$$

This problem aims at simultaneously finding an estimate for the transmitted data vector $\mathbf{s}$ and the channel vector $\mathbf{h}$. We emphasize that there is a phase ambiguity between both output vectors of ML-JED in (2). Specifically, if $\hat{\mathbf{s}}^{\text{JED}} e^{j\phi} \in \mathcal{O}^{K+1}$ for $\phi \in [0, 2\pi)$, then $\hat{\mathbf{h}} e^{j\phi}$ is also a valid solution to (2). In order to

[1] An extension to wideband systems that use OFDM [29] is straightforward as the flat-fading model in (1) remains to be valid per active sub-carrier. An extension of our methods to channels with multi-user interference or wideband channels with inter-symbol interference is, however, not straightforward; see also Remark 1 for more details.

avoid this phase ambiguity, one can either transmit information as phase differences among the entries in the vector $\mathbf{s}$, which is known as differential signaling [24], or fix the phase of at least one entry in the transmit vector $\mathbf{s}$. As in [17], we set the first entry of the transmit vector to a fixed constellation point $\check{s} \in \mathcal{O}$ and exploit this knowledge at the receiver.

Remark 2. *While this approach resembles that of conventional, pilot-based data transmission, we emphasize that ML-JED is fundamentally different. In traditional, pilot-based data transmission, a small number of known training symbols are used to generate channel estimates, which are then used during the detection of the data symbols. In contrast, ML-JED uses* all *received symbols, i.e., pilots* and *data symbols, to improve the quality of CSI. We also note that ML-JED as in* (2) jointly *solves for the transmitted data vector* $\mathbf{s}$ *and the channel vector* $\mathbf{h}$*; this is in contrast to JED methods that alternate between channel estimation and data detection (see, e.g., [13]–[16]) and for which ML-JED optimality can, in general, not be guaranteed.*

Since we assumed the entries in $\mathbf{s}$ to be constant-modulus, the ML-JED problem in (2) can be rewritten in the following, more compact form [11]:

$$\hat{\mathbf{s}}^{\text{JED}} = \underset{\mathbf{s}\in\mathcal{O}^{K+1}}{\arg\max} \ \|\mathbf{Y}\mathbf{s}\|_2, \tag{3}$$

and the associated channel estimate is given by $\hat{\mathbf{h}} = \mathbf{Y}\hat{\mathbf{s}}^{\text{JED}}/\|\hat{\mathbf{s}}^{\text{JED}}\|_2^2$. We note that the optimization problem in (3) resembles the famous MaxCut problem that is known to be NP-hard [31]. Indeed, general problems of the form (3) are known to be NP-hard with respect to randomized reductions, even when only approximate solutions are sought [32]. Nevertheless, for a small number of time slots $K+1$, the problem in (3) can be solved exactly and at low average complexity using sphere decoding (SD) methods [11]. For a large number of time slots, however, SD is known to entail prohibitively high complexity [33]. Furthermore, linear approximations are useless for JED, since the entries of $\mathbf{s}$ would grow without bound when relaxing the finite-alphabet constraint $\mathbf{s} \in \mathcal{O}^{K+1}$ to the complex numbers $\mathbf{s} \in \mathbb{C}^{K+1}$. As a consequence, the design of non-linear and low-complexity algorithms is necessary to enable JED for practical SIMO systems that use a large number of time slots.

III. PROX: PROJECTION ONTO CONVEX HULL

We now develop a novel algorithm to approximately solve the ML-JED problem in (3) using biconvex relaxation (BCR) [18], a recent framework to solve large semidefinite programs in computer vision. The resulting algorithm is referred to as PRojection Onto conveX hull (PrOX), requires low computational complexity, and achieves near-ML-JED performance.

A. Biconvex Relaxation (BCR) of the ML-JED Problem

We start from (3) and include a regularization term that forces the transmit vector $\mathbf{s} \in \mathcal{O}^{K+1}$ to be close to a copy $\mathbf{q} \in \mathbb{C}^{K+1}$ that is relaxed to the complex numbers as follows:

$$\underset{\mathbf{s}\in\mathcal{O}^{K+1},\mathbf{q}\in\mathbb{C}^{K+1}}{\text{minimize}} \ -\|\mathbf{Y}\mathbf{q}\|_2^2 + \alpha\|\mathbf{q}-\mathbf{s}\|_2^2, \tag{4}$$

where $\alpha > 0$ is a suitably chosen regularization parameter. To ensure that the problem in (4) is convex in the vector $\mathbf{q}$, the parameter α must be larger than the maximum eigenvalue of the Gram matrix $\mathbf{G} = \mathbf{Y}^H\mathbf{Y}$, i.e., $\alpha > \|\mathbf{Y}^H\mathbf{Y}\|$.

We next relax the finite-alphabet constraint to the convex hull $\mathcal{C}_{\mathcal{O}}$ of the set $\mathcal{O}$ given by [34]

$$\mathcal{C}_{\mathcal{O}} = \left\{ \sum_{i=1}^{|\mathcal{O}|} \alpha_i s_i \mid (\alpha_i \in \mathbb{R}^+, \forall i) \wedge \sum_{i=1}^{|\mathcal{O}|} \alpha_i = 1 \right\},$$

with s_i, $i = 1, \ldots, |\mathcal{O}|$. For example, the convex hull for BPSK constellations $\mathcal{C}_{\text{BPSK}}$ is the line along the real axis in $[-1, +1]$; the convex hull $\mathcal{C}_{\text{QPSK}}$ for QPSK is the square with the four constellation points as corners. By relaxing $\mathcal{O}$ to the convex hull $\mathcal{C}_{\mathcal{O}}$, we arrive at the following relaxed ML-JED problem:

$$\underset{\mathbf{s}\in\mathcal{C}_{\mathcal{O}}^{K+1},\mathbf{q}\in\mathbb{C}^{K+1}}{\text{minimize}} \ -\|\mathbf{Y}\mathbf{q}\|_2^2 + \alpha\|\mathbf{q}-\mathbf{s}\|_2^2.$$

For the above problem, it is likely to obtain a solution that is inside the convex hull. We are, however, interested in finding solutions that lie at the boundary of the convex hull $\mathcal{C}_{\mathcal{O}}$ as the original constellation points in $\mathcal{O}$ lie at the boundary. To this end, we include a norm constraint that promotes large values in $\mathbf{s}$, with the goal of pushing the entries of $\mathbf{s}$ towards the boundary of the convex hull. This leads to the final BCR formulation of the ML-JED problem:

$$\hat{\mathbf{s}}^{\text{BCR}} = \underset{\mathbf{s}\in\mathcal{C}_{\mathcal{O}}^{K+1},\mathbf{q}\in\mathbb{C}^{K+1}}{\arg\min} \ -\|\mathbf{Y}\mathbf{q}\|_2^2 + \alpha\|\mathbf{q}-\mathbf{s}\|_2^2 - \beta\|\mathbf{s}\|_2^2. \tag{5}$$

Here, β is a suitably chosen algorithm parameter that satisfies $\beta < \alpha$, which ensures that the optimization problem is *biconvex* in the vectors $\mathbf{s}$ and $\mathbf{q}$. In words, for a fixed vector $\mathbf{q}$, the problem in (5) is convex in $\mathbf{s}$, and vice versa.

B. Alternating Optimization

We solve the BCR problem in (5) using alternating minimization, i.e., we keep one of the vectors fixed while solving for the other. The resulting iterative procedure is given by

$$\mathbf{q}^{(t)} = \underset{\mathbf{q}\in\mathbb{C}^{K+1}}{\arg\min} \ -\|\mathbf{Y}\mathbf{q}\|_2^2 + \alpha\|\mathbf{q}-\mathbf{s}^{(t-1)}\|_2^2 \tag{6}$$

$$\mathbf{s}^{(t)} = \underset{\mathbf{s}\in\mathcal{C}_{\mathcal{O}}^{K+1}}{\arg\min} \ \alpha\|\mathbf{q}^{(t)}-\mathbf{s}\|_2^2 - \beta\|\mathbf{s}\|_2^2, \tag{7}$$

where $t = 1, 2, \ldots, t_{\text{max}}$ is the iteration counter.

We initialize the above algorithm with $\mathbf{s}^{(0)} = \check{s}(G_{1,1})^{-1}\mathbf{g}_1^c$, where $\mathbf{g}_1^c$ is the first column of $\mathbf{G}$, and $\check{s} \in \mathcal{O}$ is the fixed symbol transmitted in the first time slot and known at the receiver.

Since the problem in (5) is biconvex in $\mathbf{s}$ and $\mathbf{q}$, both of the above steps are convex and can be solved optimally. Furthermore, each of these optimization problems turns out to have a closed form solution. Concretely, we obtain the following two-step PrOX algorithm:

$$\mathbf{q}^{(t)} = (\mathbf{I} - \alpha^{-1}\mathbf{G})^{-1}\mathbf{s}^{(t-1)} \tag{8}$$

$$\mathbf{s}^{(t)} = \text{prox}_{\mathcal{C}_{\mathcal{O}}^{K+1}}(\theta\mathbf{q}^{(t)}), \tag{9}$$

where $\theta = \frac{\alpha}{\alpha-\beta} > 0$, $\mathbf{I}$ is the identity matrix, and the proximal operator [35] is defined as

$$\text{prox}_{\mathcal{C}_{\mathcal{O}}^{K+1}}(\mathbf{v}) = \underset{\mathbf{s}\in\mathcal{C}_{\mathcal{O}}^{K+1}}{\arg\min} \|\mathbf{v}-\mathbf{s}\|_2^2. \tag{10}$$

This proximal operator is the element-wise orthogonal projection of the vector $\mathbf{v} = \theta\mathbf{q}^{(t)}$ onto the convex hull $\mathcal{C}_{\mathcal{O}}$ around the constellation set $\mathcal{O}$. For BPSK, the proximal operator in (10) is given by $\text{prox}_{\mathcal{C}_{\text{BPSK}}}(v_b) = \max\{\min\{\Re(v_b), +1\}, -1\}$, i.e., the projection onto the real line in $[-1, +1]$; for QPSK, (10) corresponds to applying the same operation, independently, to both $\Re(v_b)$ and $\Im(v_b)$. In the case of (9), we have $\mathbf{v} = \theta\mathbf{q}^{(t)}$ with $\theta > 0$ and hence, the proximal operator in (10) projects a scaled version of $\mathbf{q}^{(t)}$ onto the convex hull of the constellation set $\mathcal{C}_{\mathcal{O}}$—this is why we dub our algorithm *PRojection Onto conveX hull* (PrOX).

C. Convergence Theory of PrOX

We now show that the PrOX algorithm is well behaved, even though it solves a biconvex problem using alternating optimization. More specifically, we begin by establishing that the iterates $\mathbf{q}^{(t)}$ and $\mathbf{s}^{(t)}$ generated by PrOX converge to stationary points of the objective function in (5), provided that the algorithm parameters α and β are chosen appropriately. We then show that these stationary points lie on the boundary of the convex hull $\mathcal{C}_{\mathcal{O}}^{K+1}$, which implies that our biconvex relaxation is reasonably tight. In what follows, we denote the objective function of PrOX in (5) as

$$f(\mathbf{q},\mathbf{s}) = \begin{cases} -\frac{1}{2}\|\mathbf{Y}\mathbf{q}\|_2^2 + \frac{\alpha}{2}\|\mathbf{q}-\mathbf{s}\|_2^2 - \frac{\beta}{2}\|\mathbf{s}\|_2^2, \text{ if } \mathbf{s}\in\mathcal{C}_{\mathcal{O}}^{K+1} \\ \infty, \text{ otherwise.} \end{cases}$$

Note that the factors of $\frac{1}{2}$ do not affect the solution of PrOX.

At an optimal solution, the sub-gradients of f with respect to both $\mathbf{s}$ and $\mathbf{q}$ should be zero. Because $\mathbf{s}^{(t)}$ is computed by minimizing f with respect to $\mathbf{s}^{(t)}$, we already know that the sub-gradient of f with respect to $\mathbf{s}^{(t)}$ contains zero. We can thus measure the optimality of the iterates by examining only $\nabla_{\mathbf{q}} f$, the gradient of f with respect to $\mathbf{q}$. We now show that this gradient vanishes for a large number of iterations t; the proof is given in Appendix A.

Theorem 1. *Suppose that the parameters α and β satisfy $\alpha > \|\mathbf{G}\|$ and $\alpha > \beta$. Then, the PrOX algorithm in (8) and (9) converges in the gradient sense, i.e.,*

$$\lim_{t\to\infty} \nabla_{\mathbf{q}} f(\mathbf{q}^{(t)}, \mathbf{s}^{(t)}) = 0.$$

Furthermore, any limit point of the sequence of iterates is a stationary point.

The PrOX algorithm is founded on the idea of replacing the discrete constellation $\mathcal{O}$ with its convex hull $\mathcal{C}_{\mathcal{O}}$. This results in a biconvex problem that is easily solved without resorting to expensive discrete programming or lifting-based methods. However, in using a biconvex relaxation, there is a risk of finding a minimizer that lies somewhere in the interior of the convex hull $\mathcal{C}_{\mathcal{O}}$, far away from any of the constellation points. We now provide conditions for which this situation does not happen. The following theorem shows that minimizers of (5) always lie on the boundary of the convex hull $\mathcal{C}_{\mathcal{O}}$; a proof is given in Appendix B.

Theorem 2. *Let the conditions of Theorem 1 hold and further assume $\alpha > \beta > 0$. Then, any non-zero stationary point of* (5) *lies along the boundary of the set $\mathcal{C}_{\mathcal{O}}$.*

Remark 3. *Theorem 1 and Theorem 2 do not guarantee PrOX to find a* global *minimizer to the ML-JED problem (3), but rather stationary (or saddle) points that lie on the boundary of the convex hull—although we often observe global minimizers in practice (cf. Section V-A). The development of stronger results that provide conditions for which PrOX finds the optimal solution is challenging and left for future work.*

D. Simplifying the Preprocessing Stage

For a large number of time slots $K+1$, computing the matrix inverse $\widehat{\mathbf{G}} = (\mathbf{I} - \alpha^{-1}\mathbf{G})^{-1}$ in (8) results in high preprocessing complexity. We now propose an approximate method that substantially reduces the preprocessing complexity for PrOX at a negligible loss in terms of error-rate performance.

Let $\|\mathbf{G}\| < \alpha$ for any consistent matrix norm $\|\cdot\|$. Then, we have the following Neumann series expansion [36]:

$$(\mathbf{I} - \alpha^{-1}\mathbf{G})^{-1} = \sum_{k=0}^{\infty} (\alpha^{-1}\mathbf{G})^k. \tag{11}$$

As put forward in [8] for data detection in massive MIMO systems, one can approximate a matrix inverse by truncating the series expansion to the first two terms, which is

$$(\mathbf{I} - \alpha^{-1}\mathbf{G})^{-1} \approx \mathbf{I} + \alpha^{-1}\mathbf{G}. \tag{12}$$

Evidently, the expression on the right-hand side does not require the computation of a matrix inverse, which significantly reduces the preprocessing complexity. More specifically, the matrix to be inverted in the left-hand side of (12) is of dimension $(K+1)\times(K+1)$. The complexity (in terms of the number of real-valued multiplications) of an explicit matrix inversion using, e.g., the Cholesky factorization scales with $(K+1)^3$ and exhibits stringent data dependencies when implemented in VLSI; see [8] for more details on the computational complexity of Cholesky-based matrix inversion and a VLSI design. Hence, for a large number of time slots K, the approximate preprocessing method in (12) yields significant savings in terms of hardware complexity.

We have the following result that bounds the error of the approximation in (12); the proof is given in Appendix C.

Theorem 3. *Let $\alpha > \|\mathbf{G}\|$. Then, the error of the approximation in (12) is bounded by*

$$\|(\mathbf{I} - \alpha^{-1}\mathbf{G})^{-1} - (\mathbf{I} + \alpha^{-1}\mathbf{G})\| \leq \frac{\|\alpha^{-1}\mathbf{G}\|^2}{1 - \|\alpha^{-1}\mathbf{G}\|}.$$

This result implies that if $\|\mathbf{G}\|$ is smaller than α, then the approximation error will be small. In other words, the approximation error will depend on the parameter α, which we can tune in practice for optimal performance. In Section V-A, we will show that PrOX with this approximation results in excellent error-rate performance while significantly reducing the complexity compared to using the original matrix $\widehat{\mathbf{G}}$.

E. Hardware-Friendly Variant of PrOX

As a last step, we now modify the PrOX algorithm in (8) and (9) to make it more hardware friendly. Since the BS is assumed to know the first entry of $\mathbf{s}$, there is no need to apply the algorithm to this particular entry. Consequently, we force the entry $s_1^{(t)}$ to be $\check{s}$ at the end of each iteration.

The matrix $\widehat{\mathbf{G}}$ exhibits, in general, a large dynamic range for different system configurations and channel realizations. To facilitate fixed-point design, we divide all of its elements by a constant $\gamma > 0$, so that the entries of the resulting matrix are close to one in absolute value. This scaling procedure requires us to introduce a new vector $\tilde{\mathbf{q}} = \gamma^{-1}\mathbf{q}$, resulting in the following modified PrOX algorithm:

$$\tilde{\mathbf{q}}^{(t)} = \gamma^{-1}\mathbf{q}^{(t)} = \gamma^{-1}\widehat{\mathbf{G}}\mathbf{s}^{(t-1)} \tag{13}$$

$$\mathbf{s}^{(t)} = \mathrm{prox}_{\mathcal{C}_{\mathcal{O}}^{K+1}}(\theta\gamma\tilde{\mathbf{q}}^{(t)}) = \mathrm{prox}_{\mathcal{C}_{\mathcal{O}}^{K+1}}(\varrho\tilde{\mathbf{q}}^{(t)}), \tag{14}$$

where $\varrho = \theta\gamma$. With these two modifications, we arrive at the hardware-friendly version of PrOX summarized as follows:

Algorithm 1 (Hardware-Friendly PrOX)**.** *Fix the algorithm parameters $\varrho > 0$ and $\alpha > \|\mathbf{G}\|$. Precompute the matrix $\widehat{\mathbf{G}}$ using one of the following expressions:*

$$\widehat{\mathbf{G}} = \gamma^{-1}(\mathbf{I} - \alpha^{-1}\mathbf{G})^{-1} \tag{15}$$

$$\widehat{\mathbf{G}} = \gamma^{-1}(\mathbf{I} + \alpha^{-1}\mathbf{G}), \tag{16}$$

and initialize $\mathbf{s}^{(0)} = \check{s}(G_{1,1})^{-1}\mathbf{g}_1^c$. Then, for every iteration $t = 1, 2, \ldots, t_{max}$, compute the following steps:

$$\tilde{\mathbf{q}}^{(t)} = \widehat{\mathbf{G}}\mathbf{s}^{(t-1)} \tag{17}$$

$$\mathbf{s}^{(t)} = \mathrm{prox}_{\mathcal{C}_{\mathcal{O}}^{K+1}}(\varrho\tilde{\mathbf{q}}^{(t)}) \tag{18}$$

$$s_1^{(t)} = \check{s} \tag{19}$$

At the end of iteration t_{max}, compute hard-output estimates of the transmitted signals as follows:

$$\hat{s}_k = \arg\min_{s\in\mathcal{O}} |s_k^{(t_{max})} - s|, \quad k = 1, 2, \ldots, K+1. \tag{20}$$

In what follows, we will use PrOX to refer to Algorithm 1 with exact preprocessing as in (15) and approximate PrOX (APrOX) to refer to Algorithm 1 with approximate preprocessing as in (16). We also note that $\alpha > 0$ and $\varrho > 0$ are algorithm parameters that can be tuned via numerical simulations to improve the error-rate performance.

IV. VLSI Architecture

We now propose a VLSI architecture for PrOX, which exhibits high regularity and enables us to achieve high throughput at low hardware complexity.

A. Architecture Overview

Figure 1 shows the VLSI architecture for PrOX in a system that uses QPSK[2] modulation. At a high level, our architecture consists of a linear array of $N = K+1$ processing elements (PEs), each one dedicated to computing an entry of the vectors $\mathbf{s}$ and $\tilde{\mathbf{q}}$. To execute Algorithm 1, each PE has three key components: (i) a $\widehat{\mathbf{G}}$-matrix memory, (ii) a complex-valued multiply-accumulate (MAC) unit, and (iii) a projection unit (see the right side of Figure 1). The $\widehat{\mathbf{G}}$-matrix memory comprises two memories to store the real and imaginary parts of $\hat{\mathbf{g}}_k^r$, i.e., the kth row of the matrix $\widehat{\mathbf{G}}$. We assume that $\widehat{\mathbf{G}}$, which is the result of either (15) (for PrOX) or (16) (for APrOX), was computed and loaded in the memories during a preprocessing step. The MAC unit of the kth PE is used to sequentially compute the kth row of the matrix-vector product (MVP) in Step (17) of Algorithm 1, resulting in $\tilde{q}_k^{(t)}$ (see Section IV-B for the details). Finally, the projection unit of the kth PE computes the kth entry of Step (18) of Algorithm 1, i.e., it uses $\tilde{q}_k^{(t)}$ to acquire $s_k^{(t)}$. Since $s_1^{(t)} = \check{s}$, the first PE does not need the previous three key components. Instead, PE 1 only contains two multiplexers and flip-flops that store the predefined constellation point $\check{s}$ in order to implement Step (19) of Algorithm 1. The implementation of all other PEs is identical and uses the aforementioned three key components. For these PEs, the hard-output estimates in Step (20) are extracted directly from the sign bits at the outputs of the projection units.

B. Matrix-Vector Product (MVP)

Before explaining the operation details of the proposed VLSI architecture, we focus on the main computation of PrOX: the MVP operation in Step (17) of Algorithm 1. A straightforward way for computing $\tilde{\mathbf{q}}^{(t)} = \widehat{\mathbf{G}}\mathbf{s}^{(t-1)}$ using a linear array of PEs is depicted in Figure 2(a) for $N = K+1 = 3$. This architecture computes $\tilde{\mathbf{q}}^{(t)}$ sequentially and on a column-by-column basis, i.e., it evaluates $\tilde{\mathbf{q}}^{(t)} = \sum_{k=1}^{N} \hat{\mathbf{g}}_k^c s_k^{(t-1)}$ in a sequential manner over $k = 1, \ldots, N$ clock cycles. Here, $\hat{\mathbf{g}}_k^c$ represents the kth column of $\widehat{\mathbf{G}}$. While such an approach is conceptually simple, it requires a centralized vector memory which suffers from a high fan-out at its output (highlighted with red color in Figure 2(a)). More concretely, the vector memory output must be connected to $N = K+1$ MAC units, eventually becoming the critical path for systems with a large number of time slots $K+1$. While wire pipelining at the output of the vector memory could be used to mitigate the fan-out, we propose an alternative solution that avoids this issue altogether.

Consider the MVP architecture depicted in Figure 2(b). In this architecture, we replace the centralized vector memory by a set of registers: each register is associated with a PE and contains an entry of the vector $\mathbf{s}^{(t-1)}$. Furthermore, the registers are chained together in a cyclic manner, forming a shift register that enables all entries of $\mathbf{s}^{(t-1)}$ to reach all the PEs in a round-robin fashion. Then, each PE only needs to access the correct entry of $\hat{\mathbf{g}}_k^r$ to ensure that the accumulator of their MAC unit contains the correct MVP result after N clock cycles. By following this approach, each register only drives the next register in the cycle and one of the MAC unit inputs, regardless of N, hereby completely avoiding the fanout issue. We henceforth call this architecture *input-cyclic MVP*, which builds the core component of the PrOX architecture shown in Figure 1 and detailed in the next subsection.

[2]For BPSK modulation, the VLSI architecture shown in Figure 1 can be simplified as data-path elements used to compute $\Im(\mathbf{s})$ can be removed.

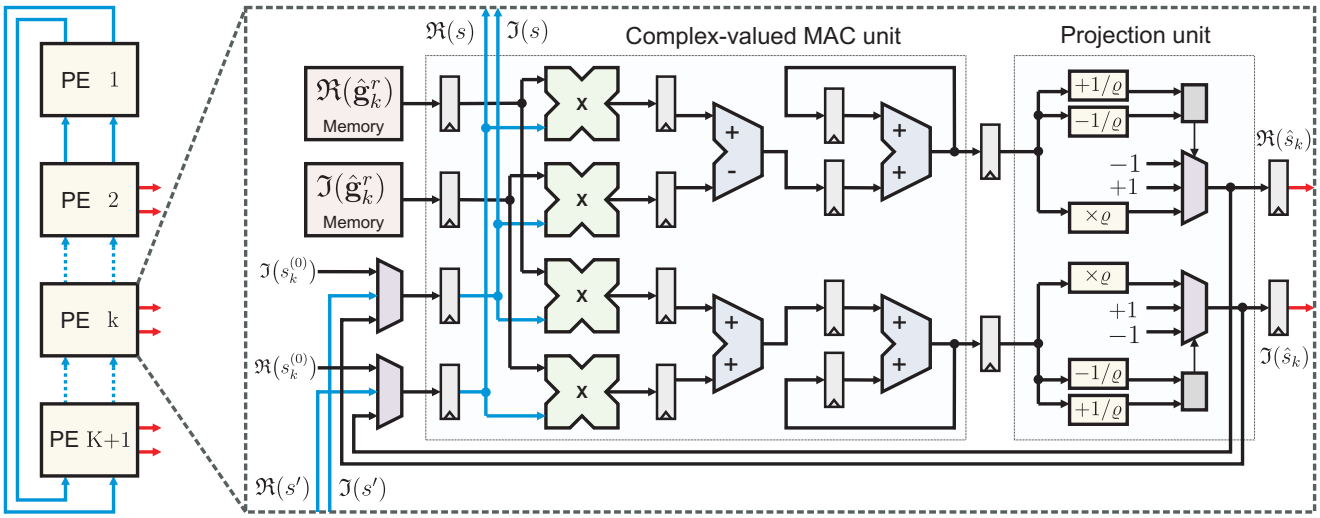


Fig. 1. VLSI architecture of PrOX. Left: linear array of $K+1$ processing elements (PEs), which enables us to achieve high throughput at low complexity. Right: architecture details of the kth PE, which mainly consists of a complex-valued multiply-accumulate (MAC) unit and a projection unit.

We note that the reading pattern of the $\widehat{\mathbf{G}}$-matrix memory for each PE corresponds to simply accessing the entries of $\hat{\mathbf{g}}_k^r$ in order. However, in contrast to the conventional architecture in Figure 2(a), reading can start from any element and wraps around after reaching the end of $\hat{\mathbf{g}}_k^r$. Instead of using address generation logic required to implement this behavior, we store a cyclically-permuted version of $\hat{\mathbf{g}}_k^r$ in the $\widehat{\mathbf{G}}$-matrix memory of each PE. By doing so, each PE simply needs to access its $\widehat{\mathbf{G}}$-matrix memory in regular order starting from the first address, as it would be done in the conventional architecture depicted in Figure 2(a).

C. Operation Details of the PrOX Architecture

To implement the input-cyclic MVP architecture, the entries of the $\widehat{\mathbf{G}}$ matrix for the kth PE are stored in the following way: The first address of the PE memory contains $\widehat{G}_{k,k}$; the second address contains $\widehat{G}_{k,k+1}$, and so on. For example, in the case shown in Figure 2(b), the second PE would have its $\widehat{\mathbf{G}}$-matrix entries organized as follows: $\{\widehat{G}_{2,2}, \widehat{G}_{2,3}, \widehat{G}_{2,1}\}$.

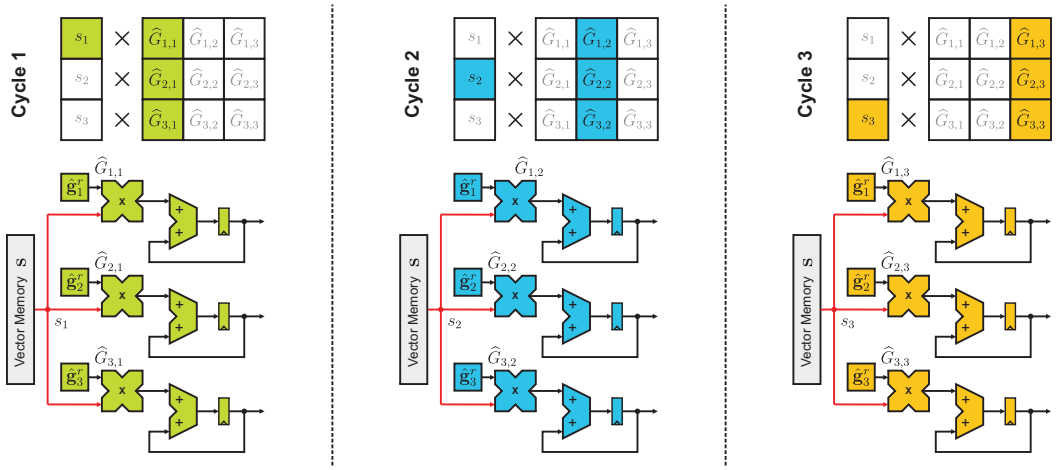
In the first clock cycle, the kth PE has access to both $\widehat{G}_{k,k}$ and $s_k^{(t-1)}$, so it can compute the $\widehat{G}_{k,k}s_k^{(t-1)}$ product, and store the result in its accumulator. At the same time, this PE passes its current $\mathbf{s}^{(t-1)}$ entry to the subsequent PE in the chain, i.e., the $(k-1)$th PE. Passing the current $\mathbf{s}^{(t-1)}$ entry to the next PE will be performed in all PEs during all the clock cycles of the MVP computation. An example of this first clock cycle is shown in Figure 2(b), where the second PE computes $\widehat{G}_{2,2}s_2$ and passes the s_2 value to the first PE, so that the first PE has access to s_2 in the second clock cycle.

In the second clock cycle, the kth PE receives $s_{k+1}^{(t-1)}$ from the $(k+1)$th PE, the previous PE in the array. As a result, the kth PE can compute $\widehat{G}_{k,k+1}s_{k+1}^{(t-1)}$ and accumulate it with the previous product. For example, in the second clock cycle of Figure 2(b), the second PE receives s_3 from the third PE and is hence able to compute $\widehat{G}_{2,3}s_3$ and add it with $\widehat{G}_{2,2}s_2$.

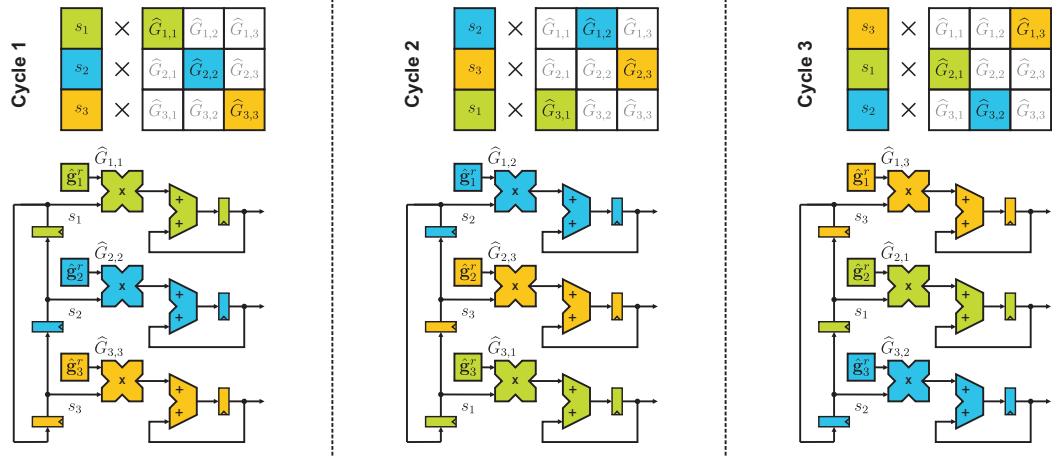
In the third clock cycle, the kth PE receives $s_{k+2}^{(t-1)}$ from the $(k+1)$th PE. Note that, while the $s_{k+2}^{(t-1)}$ value was initially in the $(k+2)$th PE, the $(k+2)$th PE previously passed this value to the $(k+1)$th PE. Then, the kth PE has the necessary operands to continue executing its complex-valued MAC operation. To clarify this aspect, let us examine the third clock cycle in Figure 2(b): the second PE has access to $\widehat{G}_{2,1}$ and s_1 to finish its computation. While the second PE received s_1 from the third PE, the third PE received s_1 from the first PE in the previous clock cycle.

This example illustrates how the exchange of values between PEs enables each entry of $\mathbf{s}^{(t-1)}$ to circulate through the linear array of PEs. In the case of the example in Figure 2(b), a complete circulation of all values in the shift registers takes three clock cycles. In general, after $K+1$ clock cycles, all PEs have had access to all the entries of $\mathbf{s}^{(t-1)}$ and used them to compute their respective entry of $\tilde{\mathbf{q}}^{(t)}$. During this procedure, the first PE (which is implemented differently than the other PEs) executes the same procedure: it passes $\check{s}$ to the $(K+1)$th PE during the first clock cycle, and in the subsequent clock cycles sends the $\mathbf{s}^{(t-1)}$ entry received from the second PE to the $(K+1)$th PE. As the MAC units contain three pipeline stages, two clock cycles are required to flush the pipeline. Hence, the MVP operation in Step (17) of Algorithm 1 is computed in only $K+3$ clock cycles.

We now describe the operation of the projection unit shown on the right side of Figure 1, which implements $\mathbf{s}^{(t)} = \mathrm{prox}_{\mathcal{C}^{K+1}_{\mathcal{O}}}(\varrho\tilde{\mathbf{q}}^{(t)})$. For QPSK, the projection unit of the kth PE consists of two identical modules: one module takes $\bar{q} = \Re(\tilde{q}_k^{(t)})$ as its input, while the other module takes $\bar{q} = \Im(\tilde{q}_k^{(t)})$ as its input. For BPSK, only the module with an input of $\bar{q} = \Re(\tilde{q}_k^{(t)})$ is required. The function of each module is to compute $\varrho\bar{q}$ and to clip its value in case it exceeds an absolute value of 1. In other words, each module determines if $\varrho\bar{q}$ is smaller than -1, larger than $+1$, or in-between these numbers, and outputs -1, $+1$, or $\varrho\bar{q}$, respectively. As $\varrho > 0$, to determine if $\varrho\bar{q} \geq +1$, one can also check $\bar{q} \geq +1/\varrho$ or, equivalently, if $\bar{q} - 1/\varrho \geq 0$, which can be extracted from the sign bit of $\bar{q} - 1/\varrho$. Analogously, to determine if $\varrho\bar{q} < -1$, one



(a) Conventional column-by-column matrix-vector product



(b) Proposed input-cyclic matrix-vector product for PrOX

Fig. 2. Processing details of the matrix-vector product (MVP) operation: (a) conventional approach that proceeds on a column-by-column basis and leads to high memory fan-out; (b) proposed input-cyclic method that reduces fan-out.

can also check $\bar{q} + 1/\varrho < 0$, which can be extracted from the sign bit of $\bar{q} + 1/\varrho$. As a result, each module takes its input $\bar{q}$ and computes $\bar{q} - 1/\varrho$ and $\bar{q} + 1/\varrho$. Using the sign bits of these two results, the module can select if the output will be -1, $+1$, or $\varrho\bar{q}$. The quantity $\varrho\bar{q}$ is computed in parallel with the calculation of $\bar{q} + 1/\varrho$ and $\bar{q} - 1/\varrho$, which reduces the critical path. By restricting the parameter ϱ to be a power of two, the multiplication can be carried out with inexpensive arithmetic shifts. The projection unit uses one clock cycle to complete its operation, but it can start only after the $K + 3$ clock cycles used by the complex MAC unit. After the projection unit has finished, the new $s_k^{(t)}$ will be available at the inputs of the complex-valued MAC unit of the kth PE, ready to start a new iteration. Consequently, each PrOX iteration requires a total number of $K + 4$ clock cycles.

D. Implementation Details

We now provide the remaining implementation details, including the fixed-point parameters as well as memory considerations for our FPGA and ASIC designs.

1) Fixed-Point Parameters: To minimize area and power consumption, and to maximize the throughput, we deploy fixed-point arithmetic in our design. Specifically, our architecture uses 6 bit signed fixed-point values for representing the $\mathbf{s}^{(t)}$ entries, with 3 fraction bits. For the elements of $\widehat{\mathbf{G}}$, 12 bit signed fixed-point values are used, with 11 fraction bits. While 18 bit values are generated at the outputs of the multipliers in the complex-valued MAC unit, only 15 bits are used in the subsequent adders and accumulators (of which 11 are fraction bits). The adder and subtractor that receive the outputs from the multipliers do not saturate, but rather wrap-around which reduces area and delay. In contrast, the accumulator adders saturate. The outputs of the complex-valued MAC unit are also 15 bit signed fixed-point values with 11 fraction bits. In the projection unit, the $-1/\varrho$ and $+1/\varrho$ quantities are represented using 12 bit signed fixed-point numbers with 11 fraction bits, and are added to the outputs of the complex MAC unit using 15 bit saturating adders. The outputs of these adders are used to determine the output of the projection unit. Furthermore, for all the considered system configurations, $\varrho > 1$. As ϱ is restricted to be a power of two, a multiplication with ϱ is

implemented by an arithmetic left shift. The number of shifts is encoded with a 4 bit number. Only the 6 most significant bits of the shifted value are taken into the multiplexer of the projection unit, as the output of this multiplexer will become the next iterate $\mathbf{s}^{(t+1)}$. We note that these fixed-point parameters are sufficient to achieve near-floating-point performance; see Figures 3 and 4 in Section V-A for corresponding symbol error-rate performance results.

2) Memory Considerations: For the FPGA designs, the $\widehat{\mathbf{G}}$-matrix memory is implemented using look-up tables (LUTs) as distributed RAM, i.e., no block RAMs have been used. For the ASIC designs, the $\widehat{\mathbf{G}}$-matrix memory is implemented using latch arrays as described in [37]; these memories are built from standard cells and simplify automated design synthesis, without a significant area penalty compared to SRAM macrocells.

V. Results

We now show error-rate performance results as well as FPGA and ASIC implementation results. We also compare our design to the only other existing JED design reported recently in [17].

A. Error-Rate Performance

1) Uplink Data Detection: Figure 3 shows *uplink* symbol error-rate (SER) simulation results for PrOX and APrOX, both running $t_{\text{max}} = 5$ iterations. The simulation results are obtained from $50,000$ Monte-Carlo trials in a $B = 16$ BS antenna SIMO system with $K = 16$ time slots. We consider both an i.i.d. flat Rayleigh block-fading channel model as well as a line-of-sight (LoS) channel with a spherical wave model and a linear BS antenna array with $\lambda/2$-wavelength spacing as described in [38]. As reference points, we also include the performance of maximum ratio combining (MRC)-based data detection with both perfect receive-side CSI (denoted by "CSIR") and with conventional channel estimation, optimal ML-JED detection using the sphere-decoding algorithm put forward in [12], and the recently proposed TASER (short for Triangular Approximate SEmidefinite Relaxation) algorithm [17] running for 10 iterations. Note that MRC with perfect CSIR is optimal for these scenarios. However, the assumption of perfect CSIR cannot be realized in practice and one must resort to channel estimation (CHEST), which entails a performance loss of more than 4 dB at 1% SER. PrOX and AProX achieve similar error rate performance. Furthermore, both of our algorithms significantly outperform MRC with CHEST and approach near-ML-JED performance but, in stark contrast to ML-JED data detection, at very low computational complexity. Our algorithms also outperform TASER, especially for QPSK modulation; see Section V-B for a hardware comparison. We also show the fixed-point performance of the VLSI designs of PrOX. The markers of PrOX and APrOX correspond to the fixed-point performance of golden models that exactly match the outputs of our VLSI designs, whereas the curves correspond to MATLAB floating-point performance—evidently, our fixed-point VLSI designs exhibit virtually no implementation loss.

2) Downlink Beamforming: Figure 4 shows *downlink* SER simulation results for PrOX and APrOX with the same system and algorithm parameters. We assume channel reciprocity [5], i.e., the downlink channel is the transpose of the uplink channel. Hence, we can use the estimated channel acquired in the uplink for MRC-based beamforming in the downlink. We observe similar trends as for the uplink shown in Figure 3. In particular, PrOX, APrOX, and TASER all achieve near-ML-JED performance. MRC and MRC that uses the estimated data vector $\mathbf{s}^{\text{MRC}}$ to compute a re-trained channel estimate according to $\hat{\mathbf{h}} = \mathbf{Y}\hat{\mathbf{s}}^{\text{MRC}}/\|\hat{\mathbf{s}}^{\text{MRC}}\|_2^2$ (referred to as "RT") is able to approach our algorithms by 2 dB and 4 dB, respectively. Hence, we conclude that JED also significantly improves the reliability of data transmission in the downlink.

3) Performance vs. Complexity Trade-offs: Figure 5 shows the trade-offs between the ASIC throughput of PrOX and AProX, and the minimum signal-to-noise ratio (SNR) required to achieve 1% SER in the uplink for SIMO systems with $K = 16$ time slots and a varying number of BS antennas. As a reference, we include the performance of MRC with perfect CSIR and that of the optimal ML-JED detector. We observed that by increasing the number of PrOX and APrOX iterations t_{max}, the mean squared error (MSE) of the channel estimate is reduced monotonically and the desired SER is achieved at a lower SNR at the expense of reduced throughput. Clearly, PrOX outperforms APrOX for a small number of iterations; this implies that the preprocessing approximation proposed in Section III-D entails a small performance loss when a small number of iterations is used. Note that for BPSK modulation, $t_{\text{max}} = 2$ PrOX iterations are typically sufficient to reach near-ML-JED performance at an ASIC throughput of 338 Mb/s per PrOX instance. For QPSK, 3 PrOX iterations are sufficient at an ASIC throughput of 451 Mb/s.

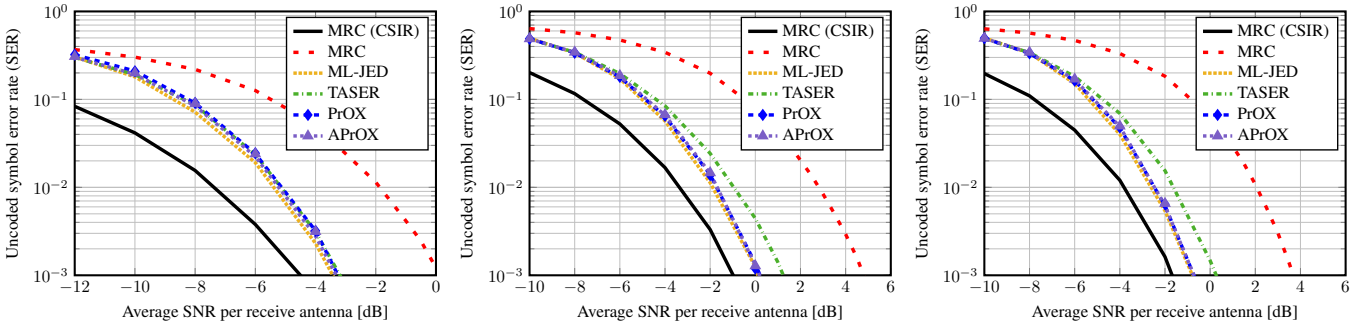
Remark 4. *For all provided simulation results, we have assumed perfect synchronization and a transmission free of impairments or pilot contamination*[3]*. Hence, the provided simulation results may not be representative for other system configurations or more realistic communication scenarios. The MATLAB simulator for PrOX used in this paper is available on GitHub: https://github.com/VIP-Group/PrOX; this enables the interested readers to investigate such aspects in more detail.*

B. FPGA and ASIC Implementation Results

To demonstrate the efficacy of PrOX, we now provide FPGA implementation results on a Xilinx Virtex-7 XC7VX690T and ASIC implementation results in a 40 nm CMOS technology. We implemented for different array sizes $N \in \{5, 9, 17, 33\}$ in Verilog on register transfer level (RTL) and hence, our FPGA and ASIC designs support near-ML-JED for $K \in \{4, 8, 16, 32\}$ time slots, respectively. We also provide a comparison to the only other existing FPGA and ASIC designs for JED proposed in the literature [17].

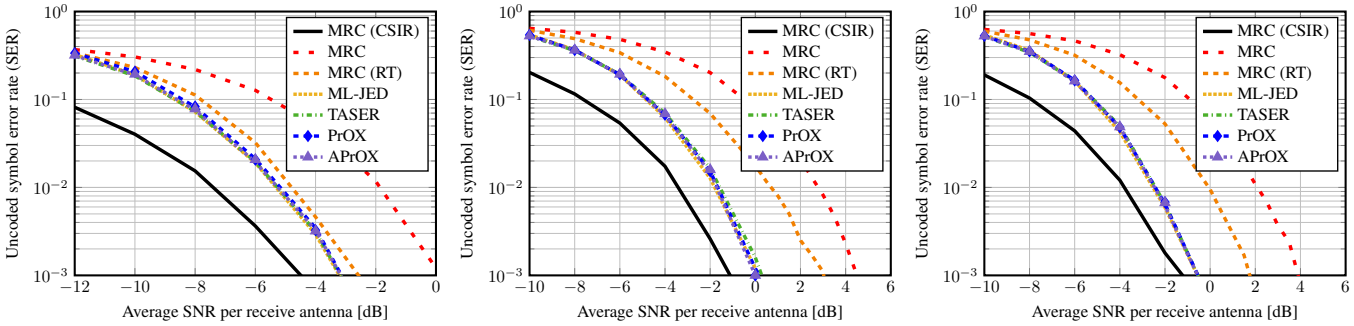
1) FPGA Implementation Results: Our FPGA implementation results were obtained using the Xilinx Vivado Design Suite, and are summarized in Table I. As expected, the resource utilization scales nearly linearly with the array size N. For the

[3]A straightforward model for pilot contamination would be to add independent Gaussian noise to the received signals—such a model would simply reduce the operating SNR. Without more accurate models, however, it is not clear what the SNR reduction would be in a practical cellular system.



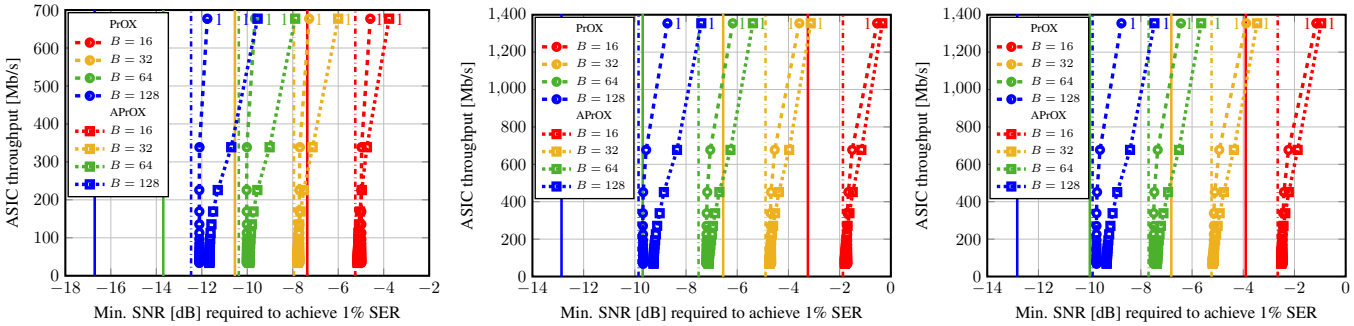
(a) BPSK, Rayleigh fading (b) QPSK, Rayleigh fading (c) QPSK, LoS

Fig. 3. Uncoded *uplink* symbol-error rate (SER) for a SIMO uplink with $B = 16$ BS antennas and transmission over $K = 16$ time slots. PrOX and APrOX achieve near-optimal SER performance (close to that of MRC detection with perfect CSIR) and exhibit a performance similar to ML-JED. At a SER of 0.1%, TASER [17] entails a 1 dB SNR loss for QPSK, while MRC with conventional CHEST suffers more than 3 dB SNR loss for BPSK and QPSK. For PrOX and APrOX, the curves show the floating-point performance whereas the markers show the fixed-point performance of our golden models.



(a) BPSK, Rayleigh fading (b) QPSK, Rayleigh fading (c) QPSK, LoS

Fig. 4. Uncoded *downlink* symbol-error rate (SER) for a SIMO uplink with $B = 16$ and $K = 16$. Beamforming with the channel estimate from PrOX, APrOX and TASER [17] achieve near-optimal SER performance (close to that of perfect CSIR) and a performance similar to that of ML-JED. At a SER of 0.1%, MRC with conventional CHEST suffers a 4 dB SNR loss for QPSK; retraining the channel with the estimated uplink data vector can reduce this loss, as in MRC (RT). For PrOX and APrOX, the curves show the floating-point performance whereas the markers show the fixed-point performance of our golden models.



(a) BPSK, Rayleigh fading (b) QPSK, Rayleigh fading (c) QPSK, LoS

Fig. 5. ASIC throughput vs. performance trade-off of PrOX and APrOX for $K = 16$ time slots. Vertical solid lines represent MRC with CSIR; vertical dashed lines represent ML-JED performance. The numbers next to the markers correspond to the number of iterations $t_{\max}$. The throughput was obtained from post place-and-route ASIC implementation results. We can see that PrOX requires a smaller number of iterations than APrOX to achieve ML-JED performance.

TABLE I
FPGA IMPLEMENTATION RESULTS OF PROX ON A XILINX VIRTEX-7 XC7VX690T FOR DIFFERENT PROX ARRAY SIZES

Array size ($N=K+1$)	$N=5$	$N=9$	$N=17$	$N=33$
Time slots ($K=N-1$)	$K=4$	$K=8$	$K=16$	$K=32$
Slices	327	658	1 491	3 018
LUTs	990	1 991	4 818	9 861
FFs	515	989	1 953	3 857
DSP48s	16	32	64	128
Max. clock freq. [MHz]	358	341	297	240
Min. latency [cycles]	8	12	20	36
Max. throughput[a] [Mb/s]	358	454	475	426
Power estimate[b] [W]	0.45	0.47	0.79	1.14

[a]Assuming QPSK modulation; for BPSK, the throughput is halved.
[b]Statistical power estimation at max. clock freq. and 1.0 V supply voltage.

TABLE II
COMPARISON OF JED FPGA DESIGNS FOR A QPSK, $B = 128$, $K = 8$ LARGE-SIMO SYSTEM ON A XILINX VIRTEX-7 XC7VX690T

Detection algorithm	PrOX	TASER [17]
Error-rate performance	Near ML-JED	Near ML-JED
Iterations $t_{\max}$	3	3
Slices	658 (0.61 %)	4 350 (4.02 %)
LUTs	1 991 (0.46 %)	13 779 (3.18 %)
FFs	989 (0.11 %)	6 857 (0.79 %)
DSP48s	32 (0.89 %)	168 (4.67 %)
Clock frequency [MHz]	341	225
Latency [clock cycles]	12	72
Throughput [Mb/s]	151	50
Power estimate[a] [W]	0.47	1.30
Throughput/LUTs	75 841	3 629
Energy/bit [nJ/b]	3.09	26.0

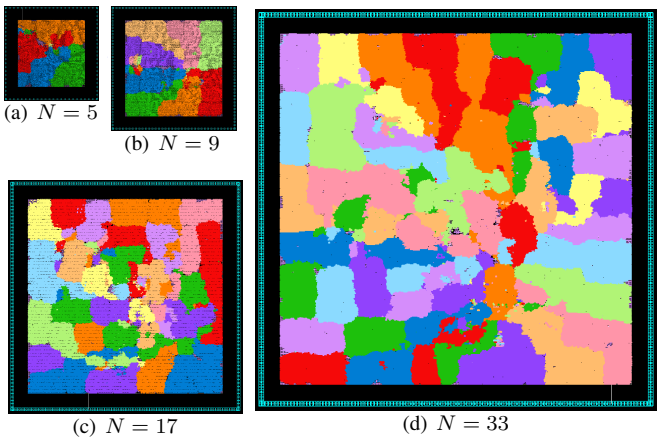
[a]Statistical power estimation at max. clock freq. and 1.0 V supply voltage.

$N \in \{5, 9, 17\}$ arrays, the critical path of PrOX is in the PEs' projection unit; for the largest $N = 33$ array, the critical path is in the distribution of the control signals.

Table II compares PrOX with TASER [17], which have been implemented on the same FPGA for a SIMO system with $B = 128$ BS antennas and communication through $K = 8$ time slots. PrOX requires significantly fewer resources and lower power than TASER, while achieving a substantially higher throughput. This makes our design superior to TASER in terms of both hardware-efficiency (measured in throughput per FPGA LUTs) and energy per bit. Concretely, PrOX is 20× more hardware-efficient and 8× more energy-efficient than TASER for the considered scenario.

2) ASIC Implementation Results: Our ASIC post place-and-route implementation results summarized in Table III were obtained using Synopsys DC and IC compilers with a 40 nm CMOS standard-cell technology. Figure 6 shows the corresponding ASIC layouts. For the $N \in \{5, 9\}$ arrays, the critical path of PrOX is in the PEs' projection unit; for $N = 17$, the critical path is in the multipliers of the MAC unit and, for the largest $N = 33$ array, the critical path is in the address generation logic and readout circuitry of the $\hat{\mathbf{g}}_k^r$ memories.

Table IV compares PrOX with TASER [17], which have



(a) $N = 5$ (b) $N = 9$ (c) $N = 17$ (d) $N = 33$

Fig. 6. Layouts of the PrOX ASIC designs for array sizes $N \in \{5, 9, 17, 33\}$. The PEs of each design were colored differently. One can observe a nearly linear increase in silicon area depending on the array size $N = K + 1$.

TABLE III
ASIC IMPLEMENTATION RESULTS OF PROX IN 40 NM CMOS FOR DIFFERENT ARRAY SIZES

Array size ($N=K+1$)	$N=5$	$N=9$	$N=17$	$N=33$
Time slots ($K=N-1$)	$K=4$	$K=8$	$K=16$	$K=32$
Core area [μm^2]	26 419	53 361	132 903	303 722
Core density [%]	77.24	77.39	72.96	76.38
Cell area [GE]	28 922	58 522	137 428	328 753
Max. clock freq. [MHz]	976	942	846	695
Min. latency [cycles]	8	12	20	36
Max. throughput [Mb/s]	976	1 256	1 354	1 235
Power estimate[a] [mW]	10	18	38	58

[a]Post place-and-route power estimation at max. clock freq. and 1.1 V.

TABLE IV
COMPARISON OF JED ASICS FOR A QPSK, $B = 128$, $K = 8$ LARGE-SIMO SYSTEM IN 40 NM CMOS

Detection algorithm	PrOX	TASER [17]
Error-rate performance	Near ML-JED	Near ML-JED
Iterations $t_{\max}$	3	3
Core area [μm^2]	53 361	482 677
Core density [%]	77.39	68.89
Cell area [GE]	58 522	471 238
Max. clock freq. [MHz]	942	560
Latency [clock cycles]	12	24
Throughput [Mb/s]	418	125
Power estimate[a] [mW]	18	87
Throughput/cell area [b/(s×GE)]	7 153	279
Energy/bit [pJ/b]	43	697

[a]Post place-and-route power estimation at max. clock freq. and 1.1 V.

been implemented on the same 40 nm CMOS technology for a SIMO system with $B = 128$ and $K = 8$. Similar to the FPGA case, the PrOX ASIC designs require significantly fewer resources and lower power than the TASER ASIC designs, while achieving a substantially higher throughput. Concretely, PrOX is $25\times$ more hardware-efficient (in terms of the throughput per cell area) and $16\times$ more energy-efficient than TASER for the considered scenario.

We also note that PrOX exhibits a similar (for BPSK) or better (for QPSK) SER performance than TASER, while using fewer algorithm iterations; see Figures 3 and 4 for the associated SER results. However, while PrOX is only suitable for JED in large SIMO systems, TASER is also able to perform near-ML data detection in coherent massive multi-user MIMO systems that use conventional pilot-based channel training [17]. Hence, the error-rate performance and hardware complexity advantages of PrOX over TASER come at the cost of reduced flexibility.

Remark 5. *While a plethora of VLSI designs exist for data detection in* coherent *small-scale and massive (multi-user) MIMO systems that separate channel estimation from data detection (see [8], [19], [22], [23], [39]–[46] and the references therein), TASER is the only other SIMO-JED VLSI design that has been described in the literature [17]. Furthermore, low-complexity linear methods that are commonly used for conventional (multi-user) MIMO data detection cannot be used in the JED scenario (as briefly discussed in Section II-B). Hence, we limited our comparisons in Table II and Table IV to PrOX and TASER.*

VI. Conclusions

We have proposed a novel joint channel estimation and data detection (JED) algorithm, referred to as PRojection Onto conveX hull (PrOX). Our algorithm builds upon biconvex relaxation (BCR) [18], which enables near-maximum-likelihood (ML) JED performance at low computational complexity for large SIMO systems with constant-modulus constellations. We have provided theoretical convergence guarantees for PrOX and introduced an approximation that significantly reduces the preprocessing complexity. To demonstrate the effectiveness of our algorithm, we have proposed a VLSI architecture that consists of a linear array of processing elements (PEs). Our architecture enables high-throughput near-ML-JED at low silicon area and power consumption. We have provided FPGA and ASIC implementation results, which demonstrate that PrOX significantly outperforms the only other existing JED design [17] in terms of hardware- and energy-efficiency as well as error-rate performance. More generally, PrOX constitutes a first step towards hardware accelerators that are able to find approximate solutions to problems that resemble the famous MaxCut problem in a hardware-efficient manner.

There are many avenues for future work. An extension of PrOX to compute soft-outputs in the form of log-likelihood ratios (LLRs) is part of ongoing work. Furthermore, a theoretical error-rate performance analysis of PrOX is a challenging open research problem. Finally, while some algorithms exist for JED in the MIMO case and for general (non-constant modulus) constellation sets [12], [21], they are either computationally complex or exhibit a considerable loss with respect to the optimal ML-JED performance. The development of efficient JED algorithms that can be implemented as high-throughput VLSI designs for large (multi-user) MIMO systems with arbitrary constellations is part of ongoing work.

Appendix A
Proof of Theorem 1

We need a closed form expression for the gradient so that we can say something about its convergence. We have

$$\partial_{\mathbf{q}} f(\mathbf{q}^{(t)}, \mathbf{s}^{(t)}) = -\mathbf{Y}^T\mathbf{Y}\mathbf{q}^{(t)} + \alpha(\mathbf{q}^{(t)} - \mathbf{s}^{(t)}).$$

From the optimality condition for the minimization step (6), we have

$$\begin{aligned} 0 &= -\mathbf{Y}^T\mathbf{Y}\mathbf{q}^{(t)} + \alpha(\mathbf{q}^{(t)} - \mathbf{s}^{(t-1)}) \\ &= -\mathbf{Y}^T\mathbf{Y}\mathbf{q}^{(t)} + \alpha(\mathbf{q}^{(t)} - \mathbf{s}^{(t)}) + \alpha(\mathbf{s}^{(t)} - \mathbf{s}^{(t-1)}) \\ &= \partial_{\mathbf{q}} f(\mathbf{q}^{(t)}, \mathbf{s}^{(t)}) + \alpha(\mathbf{s}^{(t)} - \mathbf{s}^{(t-1)}), \end{aligned}$$

and thus

$$\partial_{\mathbf{q}} f(\mathbf{q}^{(t)}, \mathbf{s}^{(t)}) = \alpha(\mathbf{s}^{(t-1)} - \mathbf{s}^{(t)}). \tag{21}$$

Now that we have a simple representation for the gradient, we can bound its norm. Note that f is strongly convex in $\mathbf{q}$ with parameter $\alpha - \|\mathbf{Y}^T\mathbf{Y}\|$, and so

$$f(\mathbf{q}, \mathbf{s}) - f(\mathbf{q}_{\mathbf{s}}, \mathbf{s}) \geq \frac{\alpha - \|\mathbf{Y}^T\mathbf{Y}\|}{2}\|\mathbf{q} - \mathbf{q}_s\|_2^2 \tag{22}$$

where $\mathbf{q}_{\mathbf{s}} = \arg\min_{\mathbf{q}} f(\mathbf{q}, \mathbf{s})$. Likewise, f is strongly convex in $\mathbf{s}$ with parameter $\alpha - \beta$, and so

$$f(\mathbf{q}, \mathbf{s}) - f(\mathbf{q}, \mathbf{s}_{\mathbf{q}}) \geq \frac{\alpha - \beta}{2}\|\mathbf{s} - \mathbf{s}_{\mathbf{q}}\|_2^2 \tag{23}$$

where $\mathbf{s}_{\mathbf{q}} = \arg\min_{\mathbf{s}} f(\mathbf{q}, \mathbf{s})$.

If we choose $\mathbf{q} = \mathbf{q}^{(t-1)}$ and $\mathbf{s} = \mathbf{s}^{(t-1)}$ in (22), then $\mathbf{q}_s = \mathbf{q}^{(t)}$ and we have

$$\begin{aligned} &f(\mathbf{q}^{(t-1)}, \mathbf{s}^{(t-1)}) - f(\mathbf{q}^{(t)}, \mathbf{s}^{(t-1)}) \\ &\qquad \geq \frac{\alpha - \|\mathbf{Y}^T\mathbf{Y}\|}{2}\|\mathbf{q}^{(t-1)} - \mathbf{q}^{(t)}\|_2^2. \end{aligned} \tag{24}$$

Similarly, from (23) we get

$$f(\mathbf{q}^{(t)}, \mathbf{s}^{(t-1)}) - f(\mathbf{q}^{(t)}, \mathbf{s}^{(t)}) \geq \frac{\alpha - \beta}{2}\|\mathbf{s}^{(t-1)} - \mathbf{s}^{(t)}\|_2^2.$$

Adding these two inequalities yields

$$\begin{aligned} &f(\mathbf{q}^{(t-1)}, \mathbf{s}^{(t-1)}) - f(\mathbf{q}^{(t)}, \mathbf{s}^{(t)}) \\ &\geq \frac{\alpha - \|\mathbf{Y}^T\mathbf{Y}\|}{2}\|\mathbf{q}^{(t-1)} - \mathbf{q}^{(t)}\|_2^2 + \frac{\alpha - \beta}{2}\|\mathbf{s}^{(t-1)} - \mathbf{s}^{(t)}\|_2^2. \end{aligned}$$

Summing from $t = 1$ to $t = T$, and observing that the summation on the left telescopes, we get

$$f(\mathbf{q}^{(0)}, \mathbf{s}^{(0)}) - f(\mathbf{q}^{(T)}, \mathbf{s}^{(T)}) \tag{25}$$

$$\geq \sum_{t=1}^{T} \frac{\alpha - \|\mathbf{Y}^T\mathbf{Y}\|}{2}\|\mathbf{q}^{(t-1)} - \mathbf{q}^{(t)}\|_2^2 \tag{26}$$

$$+ \frac{\alpha - \beta}{2}\|\mathbf{s}^{(t-1)} - \mathbf{s}^{(t)}\|_2^2.$$

Because the iterates remain bounded and f is continuous, the left-hand side of (25) is bounded from above. We can conclude that $\|\mathbf{s}^{(t-1)} - \mathbf{s}^{(t)}\| \to 0$, and from (21), we see that the residuals converge as well.

Finally, because the sub-gradients of f depend continuously on $\mathbf{s}$ and $\mathbf{q}$, any limit point of the sequence of iterates must have zero sub-gradients, and is thus a stationary point.

APPENDIX B
PROOF OF THEOREM 2

Let $(\mathbf{q}^*, \mathbf{s}^*)$ denote a non-zero stationary point of f that lies in the interior of $\mathcal{C}_{\mathcal{O}}$. Such a point satisfies the first-order conditions

$$-\mathbf{Y}^T\mathbf{Y}\mathbf{q}^* + \alpha(\mathbf{q}^* - \mathbf{s}^*) = 0 \tag{27}$$

$$\alpha(\mathbf{s}^* - \mathbf{q}^*) - \beta\mathbf{s}^* = 0. \tag{28}$$

From (28), we see that $\mathbf{q}^* = \frac{\alpha-\beta}{\alpha}\mathbf{s}^*$. Plugging this result into (27) yields

$$-\mathbf{Y}^T\mathbf{Y}\frac{\alpha-\beta}{\alpha}\mathbf{s}^* + (\alpha-\beta)\mathbf{s}^* - \alpha\mathbf{s}^* = 0,$$

which re-arranges to

$$\mathbf{Y}^T\mathbf{Y}\mathbf{s}^* = \frac{-\alpha\beta}{\alpha-\beta}\mathbf{s}^*.$$

Since $\mathbf{s}^*$ is non-zero, this shows that $\mathbf{s}^*$ is an eigenvector of $\mathbf{Y}^T\mathbf{Y}$ with negative eigenvalue. This is impossible because the Gram matrix $\mathbf{Y}^T\mathbf{Y}$ is positive semi-definite. It follows that $(\mathbf{q}^*, \mathbf{s}^*)$ cannot satisfy the first-order conditions (27) and (28), and thus, cannot lie in the interior of $\mathcal{C}_{\mathcal{O}}$.

APPENDIX C
PROOF OF THEOREM 3

Since $\|\mathbf{G}\| < \alpha$, we have the Neumann series expansion [36] in (11). Hence, we can bound the approximation error from above as follows:

$$\begin{aligned}
\|(\mathbf{I} - \alpha^{-1}\mathbf{G})^{-1} - (\mathbf{I} + \alpha^{-1}\mathbf{G})\| &= \left\|\sum_{k=2}^{\infty}(\alpha^{-1}\mathbf{G})^k\right\| \\
&\overset{(a)}{\leq} \sum_{k=2}^{\infty}\left\|(\alpha^{-1}\mathbf{G})^k\right\| \overset{(b)}{\leq} \sum_{k=2}^{\infty}\left\|\alpha^{-1}\mathbf{G}\right\|^k \\
&\overset{(c)}{=} \sum_{k=0}^{\infty}\left\|\alpha^{-1}\mathbf{G}\right\|^k - 1 - \|\alpha^{-1}\mathbf{G}\| \\
&\overset{(d)}{=} \frac{1}{1-\|\alpha^{-1}\mathbf{G}\|} - 1 - \|\alpha^{-1}\mathbf{G}\| \overset{(e)}{=} \frac{\|\alpha^{-1}\mathbf{G}\|^2}{1-\|\alpha^{-1}\mathbf{G}\|}.
\end{aligned}$$

Here, step (a) follows from the triangle inequality, step (b) from the fact that the spectral norm is a consistent matrix norm, step (c) is a result of basic arithmetic manipulations, step (d) follows from geometric series expansions, and step (e) is the same expression in simplified form. We note that the proof continues to hold for any consistent matrix norm.

ACKNOWLEDGMENTS

The authors would like to thank S. Jacobsson and G. Durisi for discussions on line-of-sight channels. We also thank C. Jeon and G. Mirza for discussions on the input-cyclic MVP engine. The work of O. Castañeda and C. Studer was supported by Xilinx, Inc. and by the US National Science Foundation (NSF) under grants ECCS-1408006, CCF-1535897, CAREER CCF-1652065, and CNS-1717559. The work of T. Goldstein was supported by the US NSF under grant CCF-1535902 and by the US Office of Naval Research under grant N00014-17-1-2078.

REFERENCES

[1] O. Castañeda, T. Goldstein, and C. Studer, "FPGA design of low-complexity joint channel estimation and data detection for large SIMO wireless systems," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2017, pp. 128–131.

[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. G. Larsson, T. L. 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] J. Hoydis, S. Ten Brink, and M. Debbah, "Massive MIMO in the UL/DL of cellular networks: How many antennas do we need?" *IEEE J. Sel. Areas Commun.*, vol. 31, no. 2, pp. 160–171, Feb. 2013.

[5] E. Larsson, O. Edfors, F. Tufvesson, and T. Marzetta, "Massive MIMO for next generation wireless systems," *IEEE Commun. Mag.*, vol. 52, no. 2, pp. 186–195, Feb. 2014.

[6] J. G. Andrews, S. Buzzi, W. Choi, S. V. Hanly, A. Lozano, A. C. Soong, and J. C. Zhang, "What will 5G be?" *IEEE J. Sel. Areas Commun.*, vol. 32, no. 6, pp. 1065–1082, Jun. 2014.

[7] L. Lu, G. Li, A. 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.

[8] 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 J. Sel. Topics Signal Process.*, vol. 8, no. 5, pp. 916–929, Oct. 2014.

[9] P. Stoica and G. Ganesan, "Space–time block codes: Trained, blind, and semi-blind detection," *Elsevier Dig. Signal Process.*, vol. 13, no. 1, pp. 93–105, Jan. 2003.

[10] H. Vikalo, B. Hassibi, and P. Stoica, "Efficient joint maximum-likelihood channel estimation and signal detection," *IEEE Trans. Wireless Commun.*, vol. 5, no. 7, pp. 1838–1845, Jul. 2006.

[11] H. A. J. Alshamary, M. F. Anjum, T. Al-Naffouri, A. Zaib, and W. Xu, "Optimal non-coherent data detection for massive SIMO wireless systems with general constellations: A polynomial complexity solution," *arXiv preprint: 1507.02319*, Jul. 2015.

[12] W. Xu, M. Stojnic, and B. Hassibi, "On exact maximum-likelihood detection for non-coherent MIMO wireless systems: A branch-estimate-bound optimization framework," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2008, pp. 2017–2021.

[13] T.-H. Pham, Y.-C. Liang, and A. Nallanathan, "A joint channel estimation and data detection receiver for multiuser MIMO IFDMA systems," *IEEE Trans. Commun.*, vol. 57, no. 6, pp. 1857–1865, Jun. 2010.

[14] R. Prasad, C. R. Murthy, and B. D. Rao, "Joint channel estimation and data detection in MIMO-OFDM systems: A sparse Bayesian learning approach," *IEEE Trans. Sig. Proc.*, vol. 63, no. 20, pp. 5369–5382, Oct. 2015.

[15] E. Kofidis, C. Chatzichristos, and A. L. F. de Almeida, "Joint channel estimation / data detection in MIMO-FBMC/OQAM systems - a tensor-based approach," *arXiv preprint: 1609.09661*, Sep. 2016.

[16] C.-K. Wen, C.-J. Wang, S. Jin, K.-K. Wong, and P. Ting, "Bayes-optimal joint channel-and-data estimation for massive MIMO with low-precision ADCs," *IEEE Trans. Sig. Proc.*, vol. 64, no. 10, pp. 2541–2556, Dec. 2016.

[17] O. Castañeda, T. Goldstein, and C. Studer, "Data detection in large multi-antenna wireless systems via approximate semidefinite relaxation," *IEEE Trans. Circuits Syst. I*, no. 99, pp. 2334–2346, Nov. 2016.

[18] S. Shah, A. Kumar, C. Castillo, D. Jacobs, C. Studer, and T. Goldstein, "Biconvex relaxation for semidefinite programming in computer vision," *European Conf. Comput. Vision (ECCV)*, pp. 717–735, Sep. 2016.

[19] S. Yang and L. Hanzo, "Fifty years of MIMO detection: The road to large-scale MIMOs," *IEEE Commun. Surveys Tuts.*, vol. 17, no. 4, pp. 1941–1988, Sep. 2015.

[20] P. Stoica, H. Vikalo, and B. Hassibi, "Joint maximum-likelihood channel estimation and signal detection for SIMO channels," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 4, May 2003, pp. 13–16.

[21] H. A. J. Alshamary and W. Xu, "Efficient optimal joint channel estimation and data detection for massive MIMO systems," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT)*, Jul. 2016, pp. 875–879.

[22] A. Burg, M. Borgmann, M. Wenk, M. Zellweger, W. Fichtner, and H. Bölcskei, "VLSI implementation of MIMO detection using the sphere decoding algorithm," *IEEE J. Solid-State Circuits*, vol. 40, no. 7, pp. 1566–1577, Jul. 2005.

[23] C. Studer, M. Wenk, and A. Burg, "VLSI implementation of hard-and soft-output sphere decoding for wide-band MIMO systems," in *VLSI-SoC: Forward-Looking Trends in IC and Systems Design*. Springer, 2010, pp. 128–154.

[24] A. Schenk and R. F. H. Fischer, "Noncoherent detection in massive MIMO systems," in *Int. ITG Workshop on Smart Antennas (WSA)*, Apr. 2013, pp. 1–8.

[25] G. Yammine and R. F. Fischer, "Soft-decision decoding in noncoherent massive MIMO systems," in *Int. ITG Workshop on Smart Antennas (WSA)*, Mar. 2016, pp. 1–7.

[26] J. Feng, H. Gao, T. Wang, T. Lv, and W. Guo, "A noncoherent differential transmission scheme for multiuser massive MIMO systems," in *Proc. IEEE Wireless Commun. Networking Conf. (WCNC)*, Mar. 2017, pp. 1–6.

[27] B. S. Thian and A. Goldsmith, "Decoding for MIMO systems with imperfect channel state information," in *Proc. IEEE Global Telecommun. Conf. (GLOBECOM)*, Dec. 2010, pp. 1–6.

[28] R. Ghods, C. Jeon, G. Mirza, A. Maleki, and C. Studer, "Optimally-tuned nonparametric linear equalization for massive MU-MIMO systems," in *Proc. IEEE Int. Symp. Inf. Theory (ISIT); arXiv preprint: 1705.02985*, Jun. 2017.

[29] R. Prasad, *OFDM for Wireless Communications Systems*. Norwood, MA, USA: Artech House, Inc., 2004.

[30] "3GPP TS 36.211 Evolved Universal Terrestrial Radio Access (E-UTRA) physical channels and modulation (release 8)," 3rd Generation Partnership Project.

[31] D. Steinberg, "Computation of matrix norms with applications to robust optimization," Master's thesis, Technion, Israel, 2005.

[32] M. Ajtai, "The shortest vector problem in L2 is NP-hard for randomized reductions," in *Proc. ACM Symp. Theory Comput.* ACM, 1998, pp. 10–19.

[33] D. Seethaler, J. Jaldén, C. Studer, and H. Bölcskei, "On the complexity distribution of sphere decoding," *IEEE Trans. Inf. Theory*, vol. 57, no. 9, pp. 5754–5768, Sep. 2011.

[34] 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*, no. 99, pp. 2357–2367, Nov. 2016.

[35] N. Parikh and S. Boyd, "Proximal algorithms," *Foundations and Trends Optimization*, vol. 1, no. 3, pp. 123–231, Jan. 2013.

[36] G. Stewart, *Matrix Algorithms: Basic decompositions*, 1998.

[37] P. Meinerzhagen, C. Roth, and A. Burg, "Towards generic low-power area-efficient standard cell based memory architectures," in *Proc. IEEE Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Aug. 2010, pp. 129–132.

[38] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge University Press, 2005.

[39] K. Wong, C. Tsui, R. Cheng, and W. Mow, "A VLSI architecture of a K-best lattice decoding algorithm for MIMO channels," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, vol. 3, May 2002, pp. 273–276.

[40] C. Studer, A. Burg, and H. Bölcskei, "Soft-output sphere decoding: Algorithms and VLSI implementation," *IEEE J. Sel. Areas Commun.*, vol. 26, no. 2, pp. 290–300, Feb. 2008.

[41] C. H. Liao, T. P. Wang, and T. D. Chiueh, "A 74.8 mW soft-output detector IC for 8×8 spatial-multiplexing MIMO communications," *IEEE J. Solid-State Circuits*, vol. 45, no. 2, pp. 411–421, Feb. 2010.

[42] C. H. Yang, T. H. Yu, and D. Marković, "A 5.8 mW 3GPP-LTE compliant 8×8 MIMO sphere decoder chip with soft-outputs," in *Symp. VLSI Circuits*, Jun. 2010, pp. 209–210.

[43] E. M. Witte, F. Borlenghi, G. Ascheid, R. Leupers, and H. Meyr, "A scalable VLSI architecture for soft-input soft-output single tree-search sphere decoding," *IEEE Trans. Circuits and Syst. II*, vol. 57, no. 9, pp. 706–710, Sep. 2010.

[44] C.-F. Liao, J.-Y. Wang, and Y.-H. Huang, "A 3.1 Gb/s 8×8 sorting reduced K-Best detector with lattice reduction and QR decomposition," *IEEE Trans. VLSI Syst.*, vol. 22, no. 12, pp. 2675–2688, Feb. 2014.

[45] C. Senning, L. Bruderer, J. Hunziker, and A. Burg, "A lattice reduction-aided MIMO channel equalizer in 90 nm CMOS achieving 720 Mb/s," *IEEE Trans. Circuits Syst. I*, vol. 61, no. 6, pp. 1860–1871, Jun. 2014.

[46] B. Yin, M. Wu, J. Cavallaro, and C. Studer, "VLSI Design of Large-Scale Soft-Output MIMO Detection Using Conjugate Gradients," in *Proc. IEEE Int. Conf. Circuits Syst. (ISCAS)*, May 2015, pp. 1498–1501.