MSE-OPTIMAL 1-BIT PRECODING FOR MULTIUSER MIMO VIA BRANCH AND BOUND

Sven Jacobsson, Weiyu Xu, Giuseppe Durisi, and Christoph Studer

ABSTRACT

In this paper, we solve the sum mean-squared error (MSE)-optimal 1-bit quantized precoding problem exactly for small-to-moderate sized multiuser multiple-input multiple-output (MU-MIMO) systems via branch and bound. To this end, we reformulate the original NP-hard precoding problem as a tree search and deploy a number of strategies that improve the pruning efficiency without sacrificing optimality. We evaluate the error-rate performance and the complexity of the resulting 1-bit branch-and-bound (BB-1) precoder, and compare its efficacy to that of existing, suboptimal algorithms for 1-bit precoding in MU-MIMO systems.

Index Terms—multiuser multiple-input multiple-output, 1-bit quantization, precoding, branch and bound

1. INTRODUCTION

Massive multiuser multiple-input multiple-output (MU-MIMO) technology, a scaled-up version of what is used in today’s cellular communication systems, is expected to play a critical role in next-generation wireless systems [1]. In the downlink, the basestation (BS) transmits data to multiple users in the same time-frequency resource by mapping the information symbols to the antenna array via a precoder [2,3]. For massive MU-MIMO systems, the increase in the number of BS antenna elements entails significant growths in circuit power consumption and interconnect bandwidth over the link connecting the baseband processing unit to the radio unit. These challenges are further aggravated when operating over large bandwidths at millimeter-wave frequencies [4].

The use of low-resolution digital-to-analog converters (DACs) has recently been proposed to reduce power consumption and mitigate the interconnect-bandwidth bottleneck at the BS. When low-resolution DACs are used, each entry of the precoded vector must be quantized to the low-cardinality alphabet that is supported by the transcoder in the DAC. For the special case of 1-bit DACs and frequency-flat channels, 1-bit precoding has been studied in, e.g., [5-12]; the frequency-selective scenario has been studied recently in [13-14].

Unfortunately, the mean-squared error (MSE)-optimal precoding problem for the case where the precoded vector is quantized to a finite alphabet is, in general, NP-hard. By relaxing the finite-alphabet constraint to a convex set, suboptimal precoders (with near-optimal performance) have been developed [9-10]. In this paper, we solve the sum MSE-optimal 1-bit precoding exactly for small-to-moderate sized MU-MIMO systems (e.g., 12 BS antennas) that operate over frequency-flat channels without resorting to an exhaustive search. To this end, we reformulate the NP-hard problem as a tree search, which we then solve by the proposed 1-bit branch-and-bound (BB-1) precoding algorithm. We deploy a number of strategies that improve the pruning efficiency of BB-1 without sacrificing optimality, and we compare the error-rate performance and complexity of BB-1 to that of existing, suboptimal precoders.

2. MSE-OPTIMAL QUANTIZED PRECODING

2.1. The Quantized Precoding (QP) Problem

We consider quantized (or finite-alphabet) precoding for the MU-MIMO downlink. The BS is equipped with $B$ antennas and serves $U$ single-antenna users in the same time-frequency resource. The goal of the sum MSE-optimal quantized precoder is to compute a precoded vector $x \in \mathcal{X}^B$, with $\mathcal{X}$ being the finite-cardinality transmit alphabet, by solving the following quantized precoding (QP) problem [9]:

$$(QP) \quad \begin{cases} \minimize_{x \in \mathcal{X}^B, \beta \in \mathbb{R}} \|s - \beta Hx\|_2^2 + \beta^2 UN_0 \\ \text{subject to } \|x\|_2^2 \leq 1 \text{ and } \beta > 0. \end{cases}$$

Here, $s \in \mathbb{C}^U$ is the (known) data vector to be transmitted to the $U$ users, $\mathcal{X}$ is the constellation alphabet (e.g., 16-QAM), $H \in \mathbb{C}^{U \times B}$ is the (known) downlink channel matrix, and $N_0$ is the noise variance at each user (assumed to be equal for all users and known at the BS). We define the signal-to-noise ratio (SNR) as $SNR = 1/N_0$. The precoding factor $\beta$ takes into account the gain of the channel [9]. We note that for a given value of $\beta$, the (QP) problem is a closest
vector problem (CVP) that is NP hard [15]. As a consequence, solving (QP) via an exhaustive search requires evaluating the objective function in (QP) for $|\mathcal{X}|^B$ candidate vectors, which is infeasible for moderate-to-large $B$. Hence, more efficient precoding algorithms are required in practice.

### 2.2. Rewriting the (QP) Problem

We start by using the fact that the precoding factor $\beta > 0$ in (QP) is a continuous parameter. Hence, given a precoded vector $x$ for which $\Re\{x^H Hs\} > 0$, the optimal associated precoding factor can be readily computed as

$$\hat{\beta}(x) = \frac{\Re\{x^H Hs\}}{\|Hx\|_2^2 + N_0 U}.$$

By inserting this optimal precoding factor $\hat{\beta}(x)$ into the objective function of (QP), we obtain

$$\|s - \hat{\beta}(x) Hx\|_2^2 + \hat{\beta}(x) U N_0 = \|s\|_2^2 - \frac{\Re\{x^H Hs\}^2}{\|Hx\|_2^2 + N_0 U}.$$

Consequently, solving the problem (QP) is equivalent to solving the following optimization problem:

$$(QP^*) \begin{cases} \text{minimize} & \frac{\|Hx\|_2^2 + N_0 U}{\Re\{x^H Hs\}^2} \\ \text{subject to} & \|x\|_2 \leq 1. \end{cases}$$

Let $\hat{x}$ denote the optimal solution to the problem (QP*). Note that the corresponding precoding factor $\hat{\beta}(\hat{x})$ can be negative. In this case, we use that $\hat{\beta}(-\hat{x}) = -\hat{\beta}(\hat{x})$ to simply flip the sign of the solution $\hat{x}$. For this not to affect the solution, we require that $\mathcal{X}$ is symmetric, i.e., that $x \in \mathcal{X}$ implies $-x \in \mathcal{X}$.

### 3. BB-1: 1-BIT BRANCH-AND-BOUND PRECODER

#### 3.1. Simplifying (QP*) for Constant-Modulus Alphabets

To arrive at a formulation of (QP*) that is amenable to branch and bound, we triangulate the problem. For this to work, we require constant-modulus (CM) transmit alphabets $\mathcal{X}$, i.e., that $|x|^2 = 1/B$ for all $x \in \mathcal{X}$, which implies that $\|x\|_2^2 = 1$. We use this property to rewrite the objective of (QP*) as follows:

$$\frac{\|Hx\|_2^2 + N_0 U}{\Re\{x^H Hs\}^2} = \frac{\|Hx\|_2^2 + N_0 U}{\Re\{x^H Hs\}^2} = \frac{\|Hx\|_2^2}{\Re\{x^H z_{\text{MRT}}\}^2}.$$

Here, $\tilde{H} = \begin{bmatrix} H^T & \sqrt{N_0 U} I_B \end{bmatrix}^T$ is the $(U + B) \times B$ augmented channel matrix and $z_{\text{MRT}} = Hb$ is the $B$-dimensional maximal-ratio transmission (MRT) vector. By applying the QR factorization $H = QR$, where $Q \in \mathbb{C}^{(U + B) \times B}$ has unitary rows and $R \in \mathbb{C}^{B \times B}$ is an upper-triangular matrix with nonnegative values on the main diagonal, we can formulate the CM quantized precoding problem as

$$(CMQP) \ \text{minimize} \ x \in \mathcal{X}^B \ \frac{\|Rx\|_2^2}{\Re\{x^H z_{\text{MRT}}\}^2},$$

which is sum MSE-optimal for CM transmit alphabets $\mathcal{X}$.

#### 3.2. Branch-and-Bound Procedure

The branch-and-bound procedure proposed in this paper finds the optimal solution $\hat{x} \in \mathcal{X}^B$ to the problem (CMQP). For the sake of brevity, we will focus exclusively on the 1-bit quantized case, where $\mathcal{X} = \{x_1, x_2, x_3, x_4\}$ and $x_m = \frac{1}{\sqrt{B}} e^{j\pi (\frac{m}{2} - \frac{1}{4})}$.

The goal of using branch and bound to solve the problem (CMQP) is to reformulate it as a tree-search problem for which we can prune large parts of the tree in order to reduce the computational complexity. It is key to realize that the problem (CMQP) can be associated with a $B$-level $|\mathcal{X}|$-tree; see Fig. 1 for an illustration. Each node in the tree at level $L$ can be uniquely described by the partial symbol vector (PSV) $x^{(L)} = [x_L, x_{L+1}, \ldots, x_B]^T \in \mathcal{X}^{B-L-1}$. Consider branching out from a node in the tree at level $L + 1$. The goals are to decide (i) which child node should be visited next and (ii) which child nodes can be pruned. To this end, we need a cost that represents the objective function in (CMQP) given a previously chosen PSV $x^{(L+1)}$ and the potential child nodes $\tilde{x} \in \mathcal{X}$ so that we can prune whenever the cost associated with a node at level $L$ exceeds some bound. With this in mind, we lower-bound the numerator of the objective function in (CMQP) as

$$\|Rx\|_2^2 \geq \sum_{b=1}^B \sum_{k=1}^B R_{b,k} x^{(L+1)}_k \geq n_L(\tilde{x}; x^{(L+1)}),$$

where $n_L(\tilde{x}; x^{(L+1)}) = 0$ depends only on the potential child node $\tilde{x} \in \mathcal{X}$ and the PSV $x^{(L+1)}$. Similarly, we upper-bound the denominator of the objective function in (CMQP) as

$$\Re\{x^H z_{\text{MRT}}\}^2 \geq \sum_{b=1}^B \Re\{x^H z_{\text{MRT}}^b\}^2 \leq d_L(\tilde{x}; x^{(L+1)}),$$

where $z_{\text{MRT}}^b$ is the $b$th entry of $z_{\text{MRT}}$ and $d_L(\tilde{x}; x^{(L+1)}) \geq 0$ depends only on the potential child node $\tilde{x} \in \mathcal{X}$ and the PSV $x^{(L+1)}$. Given these two quantities we can define the cost $c_L(\tilde{x}; x^{(L+1)})$ as follows:

$$c_L(\tilde{x}; x^{(L+1)}) = \frac{n_L(\tilde{x}; x^{(L+1)})}{d_L(\tilde{x}; x^{(L+1)})} \leq \frac{\|Rx\|_2^2}{\Re\{x^H z_{\text{MRT}}\}^2}.$$
3.3. Bounding The Cost Function

Given a PSV $x^{(L+1)} \in \mathcal{X}^{B-L}$ and a candidate child $\tilde{x}$, we write the numerator of the cost function $c_{L}(\cdot, \cdot)$ as a sum of three parts—past, present, and future—as follows:

$$n_{L}(\tilde{x}; x^{(L+1)}) = n_{L}^{\text{past}}(\tilde{x}; x^{(L+1)}) + n_{L}^{\text{present}}(\tilde{x}; x^{(L+1)}) + n_{L}^{\text{future}}(\tilde{x}; x^{(L+1)}).$$

The past is determined by the previously chosen PSV $x^{(L+1)}$ and is given by $n_{L}^{\text{past}}(x^{(L+1)}) = \sum_{b=L+1}^{B} \sum_{\ell=b}^{L} R_{b,\ell} x_{\ell} \tilde{x}_{\ell}^{2}$. The present depends on the choice of the child node $\tilde{x} \in \mathcal{X}$ and on the PSV $x^{(L+1)}$ and is given by $n_{L}^{\text{present}}(\tilde{x}; x^{(L+1)}) = |R_{L,L} \tilde{x} + \sum_{\ell=L+1}^{B} R_{L,\ell} x_{\ell}|^{2}$. The future depends on the cost of all possible leaf nodes and is given by

$$n_{L}^{\text{future}}(\tilde{x}; x^{(L+1)}) = \min_{x \in \mathcal{X}^{L-1}} \sum_{\ell=1}^{L-1} \sum_{b=1}^{B} R_{b,\ell} x_{\ell} + \sum_{\ell=L+1}^{B} R_{b,\ell} x_{\ell}^{2}.$$

Unfortunately, computing the future cost exactly is as hard as solving the original precoding problem. A trivial lower bound is obtained by setting $n_{L}^{\text{future}}(\tilde{x}; x^{(L+1)}) = 0$, which results, however, in a poor pruning behavior. In Section 4.4, we provide a more sophisticated approach that improves the pruning behavior.

Using a similar approach, we decompose the denominator of the cost associated with branching out from a node at level $L+1$ into three parts: past, present, and future. To arrive at an upper bound on the denominator of the cost function, we use the triangle inequality to bound $(\sum_{b=1}^{B} \mathbb{R}\{x_{b}^{*} z_{b}^{\text{MRT}}\})^{2}$ by

$$d_{L}(\tilde{x}; x^{(L+1)}) = \left( |d_{L}^{\text{past}}(x^{(L+1)})| + |d_{L}^{\text{present}}(\tilde{x})| + |d_{L}^{\text{future}}(\tilde{x})| \right)^{2},$$

where the past and present are given by $d_{L}^{\text{past}}(x^{(L+1)}) = \sum_{b=L+1}^{B} \mathbb{R}\{x_{b}^{*} z_{b}^{\text{MRT}}\}$ and $d_{L}^{\text{present}}(\tilde{x}) = \mathbb{R}\{\tilde{x}^{*} z_{L}^{\text{MRT}}\}$, respectively. Finally, the future cost is

$$d_{L}^{\text{future}}(\tilde{x}; x^{(L+1)}) = \max_{x \in \mathcal{X}^{L-1}} \sum_{\ell=1}^{L-1} \mathbb{R}\{\tilde{x}_{\ell}^{*} z_{\ell}^{\text{MRT}}\}.$$

It can be shown that the maximum is achieved by $\tilde{x}_{\ell} = x_{\ell}^{\text{MRT}}$, where $x_{\ell}^{\text{MRT}} = \arg\min_{x \in \mathcal{X}} |x - z_{\ell}^{\text{MRT}}|^{2}$.

4. FIVE TRICKS THAT MAKE BB-1 FASTER

We now propose five tricks that improve the pruning behavior of the proposed algorithm without sacrificing optimality.

4.1. Trick 1: Depth-First Best-First Tree Traversal with Radius Reduction

We traverse the search tree in the following manner: at level $L+1$, we pick the $\tilde{x}$ that minimizes the current cost $c_{L}(\tilde{x}; x^{(L+1)})$; we then proceed in a depth-first manner. Whenever a valid leaf node $x^{(1)}$ is found, we update the radius (bound) to

$$\rho \leftarrow \frac{||R_{x}^{(1)}||^{2}}{2},$$

and we perform backtracking by proceeding upwards and selecting the next-best symbol, excluding branches that have been explored or with a cost that exceeds the new radius.

Remark 1 Any other tree-traversal strategy could be used, such as breadth-first used in the (suboptimal) K-best algorithm that can be implemented efficiently in hardware [16][17].

4.2. Trick 2: Radius Initialization

The pruning efficiency can further be improved by initializing the tree search with some radius $\rho < \infty$, which is sufficiently large not to exclude the optimal solution [18]. We initialize the radius using the Wiener-filter (WF) solution, which can be computed at low complexity and can be shown to be optimal in the low-SNR regime. Specifically, we initialize

$$\rho = \frac{||R_{x}^{\text{WF}}||^{2}}{\mathbb{R}\{||x^{\text{WF}}||^{2}\}}.$$

Here, the $b$th entry of $x^{\text{WF}}$ is $x_{b}^{\text{WF}} = \arg\min_{x_{b} \in \mathcal{X}} |x_{b} - z_{b}^{\text{WF}}|^{2}$, where $z_{b}^{\text{WF}}$ is the $b$th entry of $z^{\text{WF}} = \mathbb{H}^{H}(\mathbb{H}\mathbb{H}^{H} + UN_{0}I_{L})^{-1}$.

4.3. Trick 3: Sorted QR Decomposition

We permute the columns of $\mathbb{H}$ (and the corresponding entries in $x$) using the sorted-QR-decomposition algorithm put forward in [19], so that the diagonal elements of $\mathbb{R}$ are sorted in ascending order. This approach improves substantially the pruning behavior for nodes close to the root because larger part of the search tree can be pruned early on.

4.4. Trick 4: Predicting the Future

The pruning efficiency can further be improved by finding a nontrivial lower bound on $n_{L}^{\text{future}}(x_{L}; x^{(L+1)})$. We denote by $R_{L-1} \in \mathbb{C}^{L-1 \times L-1}$ the submatrix of $R$ whose entry on the $b$th row ($b = 1, 2, \ldots, L-1$) and on the $\ell$th column ($\ell = 1, 2, \ldots, L-1$) is $R_{b,\ell}$. Furthermore, we denote by $D_{L}(\tilde{x}; x^{(L+1)}) \in \mathbb{C}^{L-1}$ the vector whose entry on the $b$th row ($b = 1, 2, \ldots, L-1$) is $R_{b,\ell} \tilde{x} + \sum_{\ell=L+1}^{B} R_{b,\ell} x_{\ell}$. With these definitions, we find a lower bound on $n_{L}^{\text{future}}(x_{L}; x^{(L+1)})$ using the eigenbound technique in [20] Sec. VII. Specifically, $n_{L}^{\text{future}}(\tilde{x}; x^{(L+1)}) \geq \lambda_{L-1}^{\min}(R_{L-1}^{H}R_{L-1}) \times \min_{x \in \mathcal{X}^{L-1}} ||x - R_{L-1}^{-1}D_{L}(\tilde{x}; x^{(L+1)})||^{2}_{2}$.

Here, $\lambda_{L-1}^{\min}(R_{L-1}^{H}R_{L-1})$ is the smallest eigenvalue of the Gram matrix $R_{L-1}^{H}R_{L-1}$. The vector $\tilde{x} \in \mathcal{X}^{L-1}$ that achieves the minimum is readily obtained by quantizing $R_{L-1}^{-1}D_{L}(\tilde{x}; x^{(L+1)})$ to the nearest vector in the set $\mathcal{X}^{L-1}$.
We now investigate the bit error rate (BER) and the complexity we use for the branch and bound procedure (SQUID) [9, Sec. IV-B], and sphere precoding (SP) [9, Sec. IV-C]. We note that by traversing the tree as in Section 4.1, BB-1 has to visit orders-of-magnitude fewer nodes compared to an exhaustive search, especially at moderate to low SNR values. Indeed, if \( N_0 \) is small, then the augmented channel matrix \( \mathbf{H} \) is ill-conditioned and many eigenvalues of the Gram matrix are small, resulting in poor pruning behavior.

By using the tricks proposed in Section 4.2–Section 4.5, the complexity of BB-1 is further reduced drastically. Note that the tricks in Sections 4.1 to 4.3 can be used also for SP. Further note that the complexity of SP, which delivers near-optimal performance (cf. Fig. 4), is noticeably lower than that of BB-1.

5. SIMULATION RESULTS

We now investigate the bit error rate (BER) and the complexity impact of the five tricks proposed in Section 4.1. QPSK, \( B = 12 \), and \( U = 3 \).

5.2. Complexity Impact of the Five Tricks

In Fig. 3 we show the complexity (measured in terms of the number of nodes visited during a tree search) as a function of the SNR, with and without the tricks presented in Section 4. We also show the complexity for exhaustive search, for which \( \frac{4^B \cdot 4^B - 1}{3} \) nodes are visited during a tree search, and for SP. We note that by traversing the tree as in Section 4.1, BB-1 has to visit orders-of-magnitude fewer nodes compared to an exhaustive search, especially at moderate to low SNR values. Indeed, if \( N_0 \) is small, then the augmented channel matrix \( \mathbf{H} \) is ill-conditioned and many eigenvalues of the Gram matrix are small, resulting in poor pruning behavior.

By using the tricks proposed in Section 4.2–Section 4.5, the complexity of BB-1 is further reduced drastically. Note that the tricks in Sections 4.1 to 4.3 can be used also for SP. Further note that the complexity of SP, which delivers near-optimal performance (cf. Fig. 4), is noticeably lower than that of BB-1.
7. REFERENCES


