Beamspace Channel Estimation for Massive MIMO mmWave Systems: Algorithm and VLSI Design
Seyed Hadi Mirfarshbafan, Alexandra Gallyas-Sanhueza, Ramina Ghods, and Christoph Studer

Abstract—Millimeter-wave (mmWave) communication in combination with massive multiuser multiple-input multiple-output (MU-MIMO) enables high-bandwidth data transmission to multiple users in the same time-frequency resource. The strong path loss of wave propagation at such high frequencies necessitates accurate channel state information to ensure reliable data transmission. We propose a novel channel-estimation algorithm called BEAmspace CHannel ESItemation (BEACHES), which leverages the fact that wave propagation at mmWave frequencies is predominantly directional. BEACHES adaptively denoises the channel vectors in the beamspace domain using a nonparametric shrinkage procedure that relies on Stein’s unbiased risk estimator (SURE). Simulation results for line-of-sight (LoS) and non-LoS mmWave channels reveal that BEACHES performs on par with state-of-the-art channel estimation methods while requiring orders-of-magnitude lower complexity. To demonstrate the effectiveness of BEACHES in practice, we develop a very large-scale integration (VLSI) architecture and provide field-programmable gate array (FPGA) implementation results. Our results show that nonparametric channel estimation can be performed at high throughput and in a hardware efficient manner for massive MU-MIMO mmWave systems with hundreds of antennas.

Index Terms—Millimeter wave (mmWave), massive multiuser MIMO, channel estimation, nonparametric denoising, beamspace, Stein’s unbiased risk estimator (SURE), very large-scale integration (VLSI), field-programmable gate array (FPGA).

I. INTRODUCTION

Millimeter-wave (mmWave) communication [2], [3] and massive multiuser (MU) multiple-input multiple-output (MIMO) [4], [5] are expected to be core technologies of next-generation wireless communication systems. By combining both of these technologies, one can achieve unprecedentedly high-bandwidth data transmission to multiple user equipments (UEs) in the same time-frequency resource via fine-grained beamforming. The strong path loss of wave propagation at mmWave frequencies, however, necessitates the infrastructure basestations (BSs) to acquire accurate channel state information (CSI) in order to perform data detection in the uplink (UEs transmit to BS) and MU precoding in the downlink (BS transmits to UEs) [6], [7]. To optimally tune the beamforming weights, accurate CSI is not only of paramount importance for hybrid analog-digital BS architectures [8]–[10] but also for emerging all-digital BS architectures [11]. In addition, the trend towards BS architectures with low-precision data converters to reduce power consumption, interconnect bandwidth, and system costs [12], [13] requires novel algorithms and hardware designs that are able to denoise the estimated channel vectors.

A. Sparsity-Based Channel Estimation

Fortunately, wave propagation at mmWave frequencies is predominantly directional and real-world channels typically comprise only a small number of strong propagation paths, such as a line-of-sight (LoS) component and a few first-order reflections [14]. These properties enable the design of sparsity-exploiting CSI estimation algorithms that effectively suppress channel estimation errors [15]–[19]. Compressive sensing (CS)-based methods have been proposed for mmWave channel estimation in [21], [22], including methods that rely upon orthogonal matching pursuit (OMP) [22]–[24]. The majority of such methods uses a discretization procedure of the number of propagation paths that can be resolved in the beamspace (or angular) domain [25], which results in a problem widely known as basis mismatch [26]. To avoid the basis mismatch problem, sparse channel estimation for mmWave channels can, for example, be accomplished with atomic norm minimization (ANM) [27], [28] or Newtonized OMP [29]. ANM estimates a discrete set of propagation paths off-the-grid by solving a semidefinite program (SDP). Newtonized OMP (NOMP) is a more efficient alternative to ANM and iteratively refines the incident angles of the dominant propagation paths off-the-grid with a complexity only slightly higher than that of conventional OMP. Although both of these methods do not suffer from the basis mismatch problem and exhibit excellent denoising performance, they entail high computational complexity; this makes them practically less attractive, especially in massive MU-MIMO systems where the complexity is dominated by the large number of BS antennas. In addition, the performance of these methods strongly depends on algorithm parameters that need to be tuned for the given propagation conditions.

B. Contributions

In order to successfully perform denoising-based channel estimation in real-world systems, we propose a low-complexity and nonparametric channel estimation algorithm for massive MU-MIMO mmWave systems, and we present corresponding
Our main contributions are summarized as follows:

- We propose a novel channel estimation algorithm that relies on Stein’s unbiased risk estimator (SURE), which we call BEAmspace CHannel ESTimation (BEACHES).
- BEACHES exploits sparsity of mmWave channels in the beamspace domain and adaptively denoises the channel vectors at a fixed computational complexity that scales with \( O(B \log(B)) \), where \( B \) is the number of BS antennas.
- We prove that BEACHES minimizes the mean-square error (MSE) between the noiseless and denoised channel vector in the large-antenna limit, i.e., when \( B \to \infty \), without requiring tedious parameter tuning.
- We evaluate the efficacy of BEACHES for LoS and non-LoS mmWave channel models and show that it performs on par with state-of-the-art channel estimation algorithms in terms of uncoded bit error-rate, but at orders-of-magnitude lower computational complexity.
- We develop a very large-scale integration (VLSI) architecture and present corresponding reference FPGA implementation results, which demonstrate that BEACHES enables high-throughput channel estimation in a hardware-efficient manner.

### C. Notation

Lowercase and uppercase boldface letters designate column vectors and matrices, respectively. For a vector \( \mathbf{a} \), the \( k \)-th entry is denoted by \( [\mathbf{a}]_k = a_k \); the real and imaginary parts are indicated with \( [\mathbf{a}]_R = \Re{\mathbf{a}} \) and \( [\mathbf{a}]_I = \Im{\mathbf{a}} \), respectively. The \( \ell_1 \)-norm and \( \ell_2 \)-norm of a vector \( \mathbf{a} \) is \( \|\mathbf{a}\|_1 \) and \( \|\mathbf{a}\|_2 \), respectively. For a matrix \( \mathbf{A} \), we define its transpose and conjugate transpose as \( \mathbf{A}^T \) and \( \mathbf{A}^H \), respectively. The \( N \times M \) all-zeros, \( N \times N \) identity, and \( N \times N \) discrete Fourier transform (DFT) matrices are \( \mathbf{0}_{N \times M} \), \( \mathbf{I}_N \), and \( \mathbf{F} \), respectively; the DFT matrix is normalized so that \( \mathbf{F}^H = \mathbf{I}_N \). Vectors in the DFT domain are denoted with a hat as in \( \mathbf{\hat{a}} = \mathbf{F} \mathbf{a} \). A proper complex-valued Gaussian vector \( \mathbf{a} \) with mean vector \( \mathbf{m} \) and covariance matrix \( \mathbf{K} \) is written as \( \mathbf{a} \sim \mathcal{CN}(\mathbf{m}, \mathbf{K}) \) and its probability density function (PDF) as \( f^{\mathcal{CN}}(\mathbf{a}; \mathbf{m}, \mathbf{K}) \). A real-valued Gaussian vector \( \mathbf{a} \) with mean vector \( \mathbf{m} \) and covariance matrix \( \mathbf{K} \) is written as \( \mathbf{a} \sim \mathcal{N}(\mathbf{m}, \mathbf{K}) \) and its PDF as \( f^{\mathcal{N}}(\mathbf{a}; \mathbf{m}, \mathbf{K}) \). The expectation operator is \( \mathbb{E}[\cdot] \). Optimal values are designated with the superscript \( \star \).

### D. Paper Outline

The rest of the paper is organized as follows. Section II introduces the system model and outlines the concept of denoising-based beamspace channel estimation. Section III details the BEACHES algorithm and presents the simulation results. Section IV proposes a VLSI architecture and provides FPGA implementation results. We conclude in Section V. All proofs are relegated to the appendices.

## II. SYSTEM MODEL

We now introduce the system model and summarize existing methods that perform beamspace channel estimation.

### A. System Model

We consider a massive MU-MIMO mmWave uplink system as illustrated in Figure 1. The BS is equipped with \( B \) antennas arranged as a uniform linear array (ULA) and communicates with \( U \) single-antenna UEs in the same time frequency resource.\(^1\) We focus on pilot-based channel estimation, i.e., where the UEs transmit orthogonal pilots in a dedicated training phase and the BS estimates the propagation paths between the UEs and the BS antenna array. Assuming flat-fading channel conditions, the BS estimates the \( B \)-dimensional complex channel vector \( \mathbf{h} \in \mathbb{C}^B \) for each UE. Furthermore, by assuming that (i) wave propagation is predominantly directional, which is valid if the wavelength is much smaller than the objects interacting with the waves \([6], [30]\), and (ii) the distance between UE (as well as the scatterers) and BS is sufficiently large, we can use the following well-known plane-wave approximation to model wave propagation at mmWave frequencies from a given UE to the BS \([31]\):

\[
\mathbf{h} = \sum_{\ell=1}^{L} \alpha_\ell \mathbf{a}(\Omega_\ell), \quad \mathbf{a}(\Omega) = [e^{j\Omega}, e^{j2\Omega}, \ldots, e^{j(B-1)\Omega}]^T.
\]

Here, \( L \) refers to the total number of paths arriving at the antenna array (including a potential line-of-sight path), \( \alpha_\ell \in \mathbb{C} \) is the complex-valued channel gain of the \( \ell \)-th path, and \( \mathbf{a}(\Omega_\ell) \) represents a complex-valued sinusoid containing the relative phases between BS antennas, where \( \Omega_\ell \in [0, 2\pi) \) is determined by the incident angle of the \( \ell \)-th path to the antenna array.

With pilot-based channel estimation methods, we only have access to noisy measurements of the channel vector \( \mathbf{h} \). We model such noisy measurements in the antenna domain as

\[
y = \mathbf{h} + \mathbf{e},
\]

where \( \mathbf{e} \sim \mathcal{CN}(\mathbf{0}_{B \times 1}, \mathbf{E}_0 \mathbf{I}_B) \) represents channel estimation error with variance \( \mathbf{E}_0 \) per complex entry. Note that for pilot-based channel estimation methods, the channel estimation errors are Gaussian and there is a linear relationship between \( \mathbf{E}_0 \) and the thermal noise variance \( N_0 \); see Section III-E for the details.

\(^1\)An extension of our algorithm and hardware designs to two-dimensional BS antenna arrays is part of ongoing work.
Remark 1. The channel model in (1) is appropriate for flat-fading channels assuming UEs with a single transmit antenna. For UEs that are equipped with an antenna array but transmit a single stream (layer) via beamforming, the channel vectors can still be modeled as in (1). For channels that exhibit frequency selectivity, we can consider orthogonal frequency-division multiplexing (OFDM), where each subcarrier is associated with a channel vector as in (1). For single-carrier (SC) transmission in frequency-selective channels or UEs that transmit multiple streams concurrently, multiple channel vectors would need to be estimated (one for each tap in the impulse response and for each layer). An analysis of this scenario is ongoing work. Finally, we emphasize that BEACHES continues to work if the channel vectors follow a more realistic propagation model than the one in (1). The simulation results provided in Section III-E with mmWave channel models confirm this claim.

B. Beamspace Representation

The model in (1) describes the channel vector in the antenna domain, i.e., each entry of the channel vector $h$ is associated with an antenna element in the BS array. Since the channel vectors $h$ are modeled as a superposition of $L$ complex-valued sinusoids, it is advantageous to transform the observed vector $y$ into the discrete Fourier transform domain according to $\hat{y} = F y$, where $F$ is the $B \times B$ DFT matrix. This transformation is known to convert the noisy channel vector $y = h + e$ into the so-called discrete beamspace domain (also known as angular domain) $\hat{y}$, in which each entry is associated with a specific incident angle (with respect to the BS antenna array) [25]. More importantly, if the number of paths $L$ is significantly smaller than the number of BS antennas $B$, then the beamspace representation $\hat{h}$ of the noiseless channel vector $h$ will be (approximately) sparse [19]. In other words, most of the channel vector’s energy is concentrated on a few entries, which are associated with the indices corresponding to the angles of the arriving waves. This key property of the beamspace representation is illustrated in Figure 2 which shows the magnitude of $\hat{h}$ for noiseless LoS and non-LoS channel vectors generated with the QuaDRiGa mmMAGIC urban micro (UMi) model at a carrier frequency of 60 GHz [32]. For the LoS case in Figure 2(a), we see that the channel vector consists of a strong LoS component and two weak first-order reflections arriving at two distinct angles. For the non-LoS case in Figure 2(b), we see that the arriving waves are (i) weaker than for the LoS case and (ii) spread across a wider range of angles. Nevertheless, the channel vector remains to be approximately sparse in the non-LoS case.

C. Channel Vector Denoising in the Beamspace Domain

The sparse nature of mmWave channel vectors in the beamspace domain enables the use of algorithms that denoise the channel vectors at the BS. The main idea behind such channel estimation methods is to first transform the observed noisy channel vector $y$ in the antenna domain (2) to the beamspace domain

$$\hat{y} = F y = \hat{h} + \hat{e},$$

where $\hat{e} = F e$ has the same statistics as the antenna domain channel estimation error vector $e$. It is then possible to exploit the fact that most of the arriving signal energy is concentrated on a few incident angles and to suppress noise associated with angles that do not pertain to the incoming signals. To perform denoising, a variety of algorithms have been proposed in the literature (see also the discussion in Section I-A). While most existing methods, such as OMP, suffer from the off-the-grid problem [26], more sophisticated methods such as ANM [27], [28] and NOMP [29], avoid this problem by identifying the dominant paths in the continuous beamspace domain. Unfortunately, such methods exhibit high computational complexity, especially for a large number of BS antennas $B$, which hinders their use in real-time applications. We next introduce a nonparametric beamspace denoising algorithm that is computationally efficient, can be implemented in hardware, and performs on par with sophisticated off-the-grid beamspace channel estimation algorithms.

III. BEACHES: BEAMSPACE CHANNEL ESTIMATION

We now introduce BEACHES, an efficient algorithm for channel vector denoising in the beamspace domain.

A. Channel Vector Denoising via Soft-Thresholding

The denoising and sparse signal recovery literature [16]–[20] describes a number of algorithms that are suitable for channel-vector denoising in the beamspace domain. The least absolute shrinkage and selection operator (LASSO) [33]–[35] is among the most popular methods, which, in our application, corresponds to the following optimization problem:

$$\eta(\hat{y}, \tau) = \arg\min_{\hat{h}' \in C^B} \frac{1}{2} ||\hat{y} - \hat{h}'||^2_2 + \tau ||\hat{h}'||_1.$$  

Here, we apply LASSO directly to the beamspace representation of the observed channel vector (3) and $\tau \in \mathbb{R}_+$ is a carefully-chosen denoising parameter. A closed-form expression for the solution to (4) in the complex case has

![Fig. 2. Examples of a line-of-sight (LoS) channel vector (a) and a non-LoS channel vector (b) in the discrete beamspace domain. The channel vectors are generated with the mmMAGIC UMi model at 60 GHz for a 128 antenna BS with a uniform-linear array (ULA) using $\lambda/2$ antenna spacing. One can clearly see the sparse nature of channel vectors in the beamspace domain.](image-url)
where we define
\[ y/\tau \]
and is given by the well-known
\[ \text{soft-thresholding operator } \eta(\hat{y}, \tau) \]
defined entry-wise as
\[ [\eta(\hat{y}, \tau)]_b = \frac{\hat{y}_b}{|\hat{y}_b|} \max\{|\hat{y}_b| - \tau, 0\}, \quad b = 1, \ldots, B, \]
where \( y/\tau \) is an unknown vector and \( \tau \) is generally known as it is determined by the noiseless beamspace channel vector \( h \). Since the propagation conditions, such as the number of arriving paths (the sparsity), the incident angles (the support), and the received signal strength (the magnitudes), can vary widely in wireless communication systems, the design of robust methods to adaptively select the optimal denoising parameter is critical. We now show a nonparametric approach that optimally tunes the denoising parameter \( \tau \) in a computationally-efficient manner.

**Remark 2.** BEACHES only requires knowledge of the noise variance \( N_0 \), which is generally known as it is determined by thermal noise originating in the receiver’s RF circuitry.

**B. Computing the Optimal Denoising Parameter**

We are interested in computing the optimal denoising parameter \( \tau^* \) that minimizes the mean square error (MSE) between the denoised beamspace channel vector and the true beamspace channel vector \( h \), defined as follows:

\[ \text{MSE} = \frac{1}{B} \mathbb{E} \left[ \| \eta(\hat{y}, \tau) - \hat{h} \|^2 \right]. \]

In (6), expectation is with respect to \( \hat{y} \). In what follows, we denote the optimal denoised channel vector by \( h^* = \eta(\hat{y}, \tau^*) \).

Unfortunately, determining the optimal denoising parameter \( \tau^* \) that minimizes the MSE in (6) requires knowledge of the noiseless beamspace channel vector \( h \), which is unknown in practice. To resolve this issue, we propose to minimize Stein’s unbiased risk estimate (SURE) as a surrogate for the MSE. The following result provides an expression for SURE in the complex domain and shows that SURE is an unbiased estimator of the MSE that is independent of \( h \). The proof of the following result is given in Appendix [A](#)

**Theorem 1.** Let \( \hat{h} \in \mathbb{C}^B \) be an unknown vector and \( \hat{y} \in \mathbb{C}^B \) a noisy observation vector distributed as \( \hat{y} \sim \mathcal{CN}(h, E_0 I_B) \). Let \( \mu(\hat{y}) \) be an estimator of \( h \) from \( \hat{y} \) that is weakly differentiable and operates element-wise on vectors. Then, Stein’s unbiased risk estimate given by

\[ \text{SURE} = \frac{1}{B} \| \mu(\hat{y}) - \hat{y} \|^2 + E_0 \]

\[ + E_0 \sum_{b=1}^B \left( \frac{\partial [\mu(\hat{y})]_b}{\partial [\hat{y}]_{b}} + \frac{\partial [\hat{y}]_{b}}{\partial [\hat{y}]_{b}} - 2 \right), \]

is an unbiased estimate of the MSE, i.e., satisfies

\[ \mathbb{E}[\text{SURE}] = \text{MSE}. \]

Unfortunately, no closed-form solution to this optimization problem is known. Reference [37] uses a bisection procedure to approximate the optimal value of a similar SURE expression in a sparse signal recovery application. In stark contrast to such approximate methods, we next propose BEACHES, a hardware-friendly algorithm that computes the optimal denoising parameter \( \tau^* \) in (11) using a deterministic procedure whose complexity scales only with \( O(B \log(B)) \).

**Remark 3.** SURE-based denoising was put forward in [33] for wavelet denoising of real-valued signals. Complex-valued

3As discussed in Appendix [B](#) the value of \( \text{SURE}_0 \) is undefined for \( \tau = |\hat{y}_b| \), \( b = 1, \ldots, B \), due to the non-differentiability of the function \( \eta(\hat{y}, \tau) \).
denoising via SURE has been used in \[38\] to denote CSI in OFDM-based single-input single-output communication systems. In this application, sparsity of the impulse response in the delay domain was exploited. In contrast to these results, BEACHES exploits sparsity in the beamspace domain and determines the optimal denoising parameter \( \tau^* \) in \( O(B \log(B)) \) time. We note that BEACHES could be combined with the method in \[38\] in order to improve channel estimation in OFDM-based massive MU-MIMO mmWave systems.

C. The BEACHES Algorithm

Reference \[33\] outlines an efficient procedure to minimize SURE for wavelet-denoising of real-valued signals. In what follows, we propose a similar strategy to minimize (9) for the complex-valued case. Instead of continuously searching the denoising parameter \( \tau \) through the interval \([0, \infty)\), we first sort the absolute values of the vector \( \hat{y} \) in ascending order and call the resulting sorted vector \( \hat{y}^s \). We then search for the optimal denoising parameter \( \tau \) only between each pair of consecutive elements of the sorted vector, i.e., \( \tau \in (\hat{y}^s_{k-1}, \hat{y}^s_k) \) for \( k = 1, \ldots, B + 1 \), where we define \( \hat{y}^s_0 = 0 \) and \( \hat{y}^s_{B+1} = \infty \) to account for the first interval \((0, \hat{y}^s_1)\), and last the interval \((\hat{y}^s_B, \infty)\). In each such interval, SURE in (9) is a quadratic function of \( \tau \) given by

\[
SURE_{\tau,k} = \frac{B}{B} \sum_{k=1}^{B} (\hat{y}^s_k)\tau^2 + E_0
- \frac{E_0}{B} \left( \frac{B}{k} \right) \tau^2 + E_0 (k-1).
\tag{12}
\]

For each index \( k \in \{1, \ldots, B + 1\} \), we compute the value of \( \tau = \tau^*_k \) that locally minimizes \( SURE_{\tau,k} \) in the interval \( \tau \in (\hat{y}^s_{k-1}, \hat{y}^s_k) \). Since SURE in (12) is a quadratic function of \( \tau \), the minimal value in each interval is either at the minimum of the quadratic function (12) or at one of the two interval boundaries, i.e., \( \hat{y}^s_{k-1} \) or \( \hat{y}^s_k \). The minimum value of the expression in (12) is attained by \( \tau^*_k = \frac{B}{B-0} \sum_{k=0}^{B} (\hat{y}^s_k)^{-1} \). Since the function \( SURE_{\tau,k} \) is convex within each interval \((\hat{y}^s_{k-1}, \hat{y}^s_k)\), the optimal parameter \( \tau^*_k \) in each interval \( k = 1, \ldots, B + 1 \) is given by

\[
\tau^*_k = \begin{cases} 
\tau^Q_k, & \hat{y}^s_{k-1} < \tau^Q_k < \hat{y}^s_k, \\
\hat{y}^s_{k-1}, & \tau^Q_k < \hat{y}^s_{k-1}, \\
\hat{y}^s_k, & \tau^Q_k > \hat{y}^s_k, 
\end{cases}
\tag{13}
\]

or simply \( \tau^*_k = \max\{\hat{y}^s_{k-1}, \min\{\hat{y}^s_k, \tau^Q_k\}\} \). After identifying the optimal value \( \tau^*_k \) in each interval, the parameter \( \tau^* \) that achieves the global minimum can be found by comparing all the local minima, i.e., by solving

\[
\tau^* = \text{arg min}_{k=1, \ldots, B+1} SURE_{\tau^*_k,k}.
\tag{14}
\]

\(^{3}\text{Note that SURE}_k \text{ and SURE}_{\tau,k} \text{ are not defined for } \tau = \hat{y}^s_{k-1} \text{ and } \tau = \hat{y}^s_k. \text{ We evaluate SURE}_{\tau,k} \text{ for two values arbitrarily close to these boundaries, i.e., } \tau = \hat{y}^s_{k-1} + \epsilon \text{ and } \tau = \hat{y}^s_k - \epsilon \text{ where } \epsilon > 0 \text{ is small compared to } \tau.\]

It is now key to realize that we do not need to recompute SURE in (12) from scratch while searching through \( k = 1, \ldots, B + 1 \). Instead, for each value of \( k \), we can sequentially update the two quantities \( S = \sum_{k=1}^{B} (\hat{y}^s_k)^2 \) and \( V = \sum_{k=1}^{B} (\hat{y}^s_k)^{-1} \), noting that the magnitudes of the vector \( \hat{y}^s \) are sorted. Algorithm 1, which we call BEACHES, exploits exactly this observation. Lines 3 to 14 detail the search procedure described in (14); this part of the algorithm only involves scalar operations (additions, multiplications, divisions, and comparisons) all of which scale with \( O(1) \). As a consequence, this iterative search has a complexity of only \( O(B) \). If we assume that the DFT and inverse DFT in line 2 and line 16 are carried out with a fast Fourier transform (FFT) and inverse FFT (IFFT), respectively, and the sorting procedure in line 3 uses a fast sorting algorithm (e.g., merge sort) with complexity \( O(B \log(B)) \), then the computational complexity of BEACHES scales only with \( O(B \log(B)) \). Furthermore, we emphasize that sorting, FFT, iterative scan, and IFFT are all hardware friendly operations; see Section IV for a corresponding VLSI design. A detailed complexity comparison of BEACHES to NOMP and ANM can be found in Section III-E.

D. Algorithm Simplification for Hardware Implementation

To enable a simpler hardware implementation of BEACHES, which is described in detail in Section IV, we can approximate \( \tau^*_k \) by \( \hat{y}^s_{k+1} \) instead of computing the optimal value \( \tau^*_k \) on line 6 of Algorithm 1. While this approximation helps to reduce the complexity of our hardware implementation, the simulation results shown next reveal that the resulting performance is virtually indistinguishable from the original BEACHES algorithm. In addition, we avoid the reciprocal

\[^3\text{An alternative approach would be to replace } \tau^*_k \text{ by } \hat{y}^s_{k+1} + \hat{y}^s_k, \text{ which results in slightly higher hardware complexity but avoids evaluating SURE at the boundaries. The error-rate and MSE performance of both of these approximations is practically the same as the optimal method.} \]
computations $1/B$ on line 7 in Algorithm 1 by scaling the SURE expression by $B$; we also omit the constant term $E_0$. Both of these simplifications do not affect the value of $\tau^*$ that minimizes this expression.

**E. Simulation Results**

To demonstrate the effectiveness of BEACHES, we now present simulation results and a comparison with existing channel vector denoising methods.

1) **Simulated Scenario:** We consider a massive MU-MIMO scenario in which $U$ UEs communicate with a $B$-antenna BS over $t = 1, \ldots, T$ time slots. The input-output relation of the flat-fading system in time slot $t$ is modeled by

$$r_t = Hs_t + n_t.$$  \hspace{1cm} (15)

Here, $r_t \in \mathbb{C}^B$ is the received vector at the BS, $H \in \mathbb{C}^{B \times U}$ represents the (unknown) MIMO channel, $s_t = [s_{1,t}, \ldots, s_{U,t}]^T$ is the transmit vector with entries chosen from a discrete constellation $O$ and normalized as $E[\|s_t\|^2] = \rho^2$, and $n_t \sim \mathcal{CN}(0, N_0I_B)$ models thermal noise.

During the channel estimation phase, we sequentially train each column of $H$ over $U$ time slots. Concretely, in each time slot $t = 1, \ldots, U$, one UE is active and transmits $s_{u,t} = \rho$, whereas all others remain inactive. With this training scheme, the estimate of the $u$th column of the MIMO channel matrix $H$ can be modeled as $y_u = h_u + e_u$ as done in (2), where the channel estimation error corresponds to $e \sim \mathcal{CN}(0, E_0I_B)$ with variance $E_0 = N_0/\rho^2$ per complex entry. We then perform channel vector denoising independently for each column of the noisy observation of $H$ in order to obtain an improved MIMO channel matrix $H^*$. 

During the data transmission phase, all UEs $u = 1, \ldots, U$ transmit a constellation point from the set $O$ to the BS concurrently and in the same frequency band; with the same power normalization $E[\|s_t\|^2] = \rho^2$, as in the training phase. Data detection is carried out using linear minimum-mean-square-error (L-MMSE) equalization [39] with the estimated channel matrix $H^*$.

To characterize the performance of BEACHES and other denoising algorithms, we simulate (i) the uncoded bit error rate for $16$-QAM and (ii) the channel estimation MSE as in (6). The channel matrices are generated for both a LoS and a non-LoS conditions using the QuaDRiGa mmMAGIC UMi model [32], at a carrier frequency of $60$ GHz with a ULA using $\lambda/2$ antenna spacing. The UEs are placed randomly within a $120^\circ$ circular sector with minimum and maximum distance of 10 and 110 meters from the BS antenna array, respectively. In addition, we enforce a UE separation of at least $1^\circ$ (with respect to the BS antenna array) and assume optical UE power control.

**Remark 4.** To enable the interested readers to perform numerical simulations with other system parameters or channel estimation algorithms, we will release our MATLAB simulator on GitHub after (possible) acceptance of the paper.

2) **BER Performance:** Figure 4 shows uncoded bit error rate (BER) simulation results for $B = 128$ BS antennas with $U = 8$ UEs, and $B = 256$ BS antennas with $U = 16$ UEs, for LoS and non-LoS channel conditions. In addition to BEACHES as detailed in Algorithm 1, we show the BER of the hardware-friendly version described in Section III-D, called “BEACHES (hw)”, and that of our fixed-point hardware design called “BEACHES (fp)”. We also compare our methods to the following channel estimation methods: (i) Maximum likelihood (ML) channel estimation, (ii) NOMP with software package provided by [29], where we manually tuned the false alarm rate $P_{fa}$ for each scenario to optimize performance, (iii) ANM-based denoising, where we use the atomic line spectral estimation toolbox provided by [27] (we use the exact noise variance and the debiased output). As a reference, the results for “exact MSE” use the same soft-thresholding function as in BEACHES, but the optimal denoising parameter $\tau^*$ is determined by minimizing the MSE (6), using the noiseless (ground truth) channel vector. Furthermore, “perfect CSI” directly uses the noiseless channel vectors.

From Figure 4 we see that channel vector denoising in the beamspace domain provides $2$ dB to $3$ dB SNR performance improvements at $BER = 10^{-3}$ compared to conventional ML channel estimation for the considered scenarios. The performance gains are more pronounced under LoS conditions. More importantly, we observe that BEACHES performs on par with all other denoising-based channel estimation methods in terms of uncoded BER for the considered scenarios. This observation indicates that off-the-grid denoising methods, such as NOMP and ANM, do not provide a critical performance advantage over BEACHES. Furthermore, our hardware friendly algorithm “BEACHES (hw)” and the fixed-point version “BEACHES (fp)” deliver the same performance as BEACHES.

3) **MSE Performance:** Figure 5 shows the MSE of the channel estimation for the same scenarios and algorithms considered in Figure 4. In terms of MSE, the performance of ANM and NOMP is superior to that of BEACHES for LoS channels. We address this to the fact that the channel realizations are extremely sparse under such conditions (cf. Figure 2(a)). For non-LoS channels, all methods perform equally well. We address this to the fact that the beamspace representation for these non-LoS channels is not sufficiently sparse (cf. Figure 2(b)). Note that these observations also indicate that the MSE is not a particularly reliable metric to predict the BER performance of channel estimation methods in massive MU-MIMO mmWave systems.

**F. Complexity Scaling and Runtime Comparison**

We now compare the complexity scaling of BEACHES to that of NOMP and ANM. We furthermore provide a MATLAB runtime comparison for LoS and non-LoS channels. In what follows, we assume that the complexity of a $B \times B$ matrix inversion and eigenvalue decomposition scales with $O(B^3)$.

1) **Complexity Scaling:** As detailed in Section III-D, the complexity of BEACHES scales with $O(B \log(B))$, dominated by the FFT, IFFT, and sorting operations.

The complexity of NOMP scales with $[29]$,

$$O(K\gamma B \log(\gamma B) + K^2 B + BK^3 + K^4),$$ \hspace{1cm} (16)

6The BER at high SNR for the LoS scenario differs slightly to that of our conference paper [1], due to fewer Monte-Carlo trials in that paper.
(b) LoS, $B = 128, U = 8$

(c) Non-LoS, $B = 128, U = 8$

(d) Non-LoS, $B = 256, U = 16$

Fig. 4. Uncoded bit error-rate (BER) performance of channel denoising methods for LoS and non-LoS channels. We see that BEACHES performs on par with atomic norm minimization (ANM) and Newtonized OMP (NOMP), and provides 2 dB to 3 dB SNR improvements over ML channel estimation at $BER = 10^{-3}$.

where $\gamma$ is the frequency oversampling factor (typically set to 4) and $K$ is the number of OMP iterations, which specifies the number of detected complex sinusoids. The exact value of $K$ is determined internally by NOMP and depends on a number of factors, including the false alarm rate $P_{fa}$, the SNR, and the channel scenario, all of which affect the sparsity level of the observation vector. We observed average values for $K$ ranging from 2 to 45 for the simulated scenarios in Section III-E.

For large $B$, the complexity of NOMP is dominated by the term $K \gamma B \log(\gamma B)$ in [16]. Hence, by ignoring the term $BK^3$ in [16], the complexity of NOMP is at least $K$ times higher than that of BEACHES—we confirm this observation in the runtime comparison of Section III-F.

The complexity of ANM scales with $O(K' B^3)$, where $K'$ is the number of iterations of the fast alternating direction method of multipliers (ADMM) implementation provided by [27]. Each algorithm iteration requires a projection onto the semidefinite cone, which is implemented via an eigenvalue decomposition whose complexity scales with $O(B^3)$ [40]. We observed average values of $K'$ ranging from 130 to 360 for the simulated scenarios in Section III-E. Consequently, ANM has orders-of-magnitude higher complexity than BEACHES, especially for a large number of BS antennas $B$—we confirm this observation by the runtime comparison detailed next.

2) Runtime Comparison: While the performance in terms of uncoded BER is comparable for all considered channel estimation methods, BEACHES exhibits (often significantly) lower complexity than NOMP and ANM. To reinforce this claim, we measured their MATLAB runtimes in milliseconds on an Intel core i5-7400 CPU with 16 GB RAM at a signal-to-noise ratio (SNR) of 5 dB; at higher SNRs, the runtimes of

| Scenario | BEACHES (fp) | NOMP | ANM |
|----------|--------------|------|-----|
| $B = 128$, LoS | 0.57 (1x) | 28.36 (50x) | 5.221 (9100x) |
| $B = 128$, non-LoS | 0.40 (1x) | 260.4 (650x) | 7.725 (19000x) |
| $B = 256$, LoS | 1.64 (1x) | 199.9 (120x) | 47.968 (29000x) |
| $B = 256$, non-LoS | 1.45 (1x) | 2204 (1500x) | 837.50 (58000x) |

MATLAB RUNTIMES IN MILLISECONDS (AND NORMALIZED RUNTIMES) ON AN INTEL CORE i5-7400 CPU WITH 16 GB RAM.
Fig. 5. Mean-square error (MSE) performance of channel denoising methods for LoS and non-LoS channels. We see that BEACHES provides 2.5× to 6× MSE improvement over ML channel estimation at SNR = 0 dB. NOMP and ANM increase by up to 2× whereas the runtime of BEACHES remains unaffected. Table II demonstrates that the runtime of BEACHES is orders-of-magnitude lower than that of NOMP (up to 1500×) and ANM (up to 58,000×), while the speedup is more pronounced for $B = 256$ BS antennas than for $B = 128$ BS antennas.

Remark 5. The complexity scaling analysis and runtime comparison in Table II hide an important aspect: NOMP and ANM can be used for compressive channel estimation whereas BEACHES can only be used for beamspace channel vector denoising. The development of efficient off-the-grid channel estimation methods specialized to beamspace channel vector denoising is an interesting open research problem.

IV. VLSI DESIGN AND FPGA IMPLEMENTATION

We now describe a VLSI architecture of the simplified version of BEACHES described in Section II-D and present reference FPGA implementation results.

A. Architecture Overview

Figure 6 provides a high-level overview of the proposed VLSI architecture that implements the hardware (hw) version of BEACHES presented in Section II-D. The architecture consists of three main modules: (i) an antenna-to-beamspace (A2B) conversion module, (ii) a SURE-based denoiser (SBD) module, and (iii) a beamspace-to-antenna (B2A) conversion module. The A2B module converts the received antenna-domain channel vector $\mathbf{y}$ into the beamspace domain vector $\mathbf{\hat{y}}$ as in (3). The same module also transforms the entries of the vector $\mathbf{\hat{y}}$ from Cartesian coordinates to polar coordinates, which simplifies the denoising procedure. The SBD module implements SURE-based denoising, i.e., identifies the optimal denoising parameter $\tau^*$ and applies the shrinkage to the magnitudes of the beamspace vector entries $\hat{y}_k$, $k = 1, \ldots, B$. The B2A module transforms the entries of the denoised beamspace vector $\mathbf{\hat{h}}^*$, $k = 1, \ldots, B$, from polar into Cartesian coordinates. The same module also converts the denoised beamspace vector $\mathbf{\hat{h}}^*$ back into the antenna domain $\mathbf{h}^*$. To maximize throughput, the proposed architecture relies on input/output streaming. Concretely, the architecture reads a new channel vector entry...
which transforms the Cartesian number representation into polar
which allows for more compact fixed-point data representation

we use a Xilinx LogiCORE FFT IP with radix-2 pipelined
v vector into the beamspace domain. In our implementation,

\[ \tau \]
the inputs of the SBD module so that they are ready as soon as

subtractor and a multiplexer) to apply soft-thresholding to the

to determine the optimal denoising threshold, and logic (a
buffer, a module to perform sort-and-scan (SAS) in order

on the fixed-point parameters of our design.

the CORDIC core is determined by the IP so that the required

using a Xilinx LogiCORE IP. The number of microrotations in
each entry in the SBD module. The CORDIC is implemented

on the intermediate values by a factor of two in each of the
B stages. This configuration reduces the dynamic

range in each FFT stage and also reduces resource utilization.

Additionally, this scaling approach yields FFT outputs that
have smaller dynamic range compared to the unscaled case,
which allows for more compact fixed-point data representation
and reduces storage requirements in the subsequent modules.

After FFT processing, each complex-valued beamspace
domain sample \( \hat{y}_k \) is passed through a vectoring CORDIC,
which transforms the Cartesian number representation into polar
coordinates. This transform simplifies the soft-thresholding
operation, as it only needs to be applied to the magnitude of
each entry in the SBD module. The CORDIC is implemented
using a Xilinx LogiCORE IP. The number of microrotations in
the CORDIC core is determined by the IP so that the required
accuracy of 10 bit is achieved; see Section IV-C for more details
on the fixed-point parameters of our design.

2) SURE-based Denoiser (SBD) Module: As shown in
Figure 6, this module consists of a first-in first-out (FIFO)
buffer, a module to perform sort-and-scan (SAS) in order
to determine the optimal denoising threshold, and logic (a
subtractor and a multiplexer) to apply soft-thresholding to the
values in the FIFO buffer. The role of the FIFO buffer is to delay
the inputs of the SBD module so that they are ready as soon as
the optimal threshold \( \tau^* \) has been computed. The FIFO buffer
has a depth of \( 2B + 5 \) entries, corresponding to the latency of
the SAS submodule as detailed in the next paragraphs. The
details of the SAS architecture are shown in Figure 7. The
architecture consists of a sort unit and a subsequent scan unit,
corresponding to lines 3 to 4 of Algorithm 1. The following
paragraphs summarize the most important architecture details.

As depicted in Figure 7, the sort unit consists of an array
of \( B \) identical processing elements (PEs). The details of the
PEs are shown for the second PE (PE-2), which consists of (i)
a register to keep one of the sorted elements, (ii) a multiplexer
that selects whether the new input data or the value stored in
the previous PE should enter the register, (iii) a comparator
(denoted by “cmp”) that compares the new input value with the
value stored in the PE’s register, and (iv) a control unit (denoted
by “ctrl”) that determines the multiplexer output and whether
the register must be updated. As for the FFT core, the sort unit
is using I/O streaming, i.e., the architecture continuously reads
and generates data. This approach also allows for a seamless
integration with the scan unit (discussed below), and eliminates
the need to buffer the sorted data separately in a memory, for
the scan unit to work on. The sort unit sorts the data as they
enter, by finding the appropriate position within the array for
each new input data. Assume that \( k \) entries of a \( B \times 1 \) vector
have already been sorted and reside in the PEs 1 to \( k \). In the
next clock cycle, the \((k + 1)\)th (unsorted) element enters the
sort unit and is broadcast to all PEs. Each PE compares the
new element with the value stored in its own register, and
additionally, receives the result of the same comparison from
its preceding PE. For the case of sorting in descending order
(i.e., PE-\( B \) stores the smallest element), the new input data will
be placed in PE-\( m \), if the new element is larger than the data
stored in PEs 1, \ldots, \( m \) and smaller than or equal to the
value stored in PE-\( m \). At the same time, the PEs \( m, \ldots, k \) will
pass their previously stored values to their adjacent PE (e.g.,
PE-\( m \) to PE-\((m + 1)\)), so that no data is lost. This approach
is repeated until all \( B \) elements are sorted in the PEs at the
clock cycle after receiving the last element. When loading the
first element of the next denoising problem, PE-\( B \) will pass its
value (which is the smallest element of the last channel vector)
to the scan unit and will receive the data of PE-\((B - 1)\), and
therefore the sorted data will be flushed at the same time the
next problem is being loaded and sorted.

Fig. 6. High-level VLSI architecture of BEACHES (hw). The architecture operates in input/output streaming mode and consists of three modules: an antenna-to-beamspace (A2B) conversion module, a SURE-based denoiser (SBD) module, and a beamspace-to-antenna (B2A) conversion module. The only required parameter is the variance \( E_0 \) of the channel estimation noise, which is known in practical systems.
Algorithm 1, the scan unit receives the unsorted data at the
Remark 6. Although the algorithm complexity of BEACHES is $O(B \log(B))$, the implemented sorting architecture has a hardware complexity of $O(B^2)$ in terms of the area-delay product. The reason for this architecture choice is the fact that this sorting method supports I/O streaming without a significant overhead in terms of latency and buffering. Furthermore, our implementation results in Section IV-D demonstrate that this architecture is efficient for the targeted BS antenna numbers.

The scan unit is depicted in Figure 7. In order to compute the cumulative sum of reciprocals denoted by $V$ on line 4 of Algorithm 1, the scan unit receives the unsorted data at the same time they enter the sort unit. The reciprocal values of the unsorted data are computed sequentially using a look-up-table (LUT) with 512 entries and are accumulated in a register. Therefore, the cumulative sum of reciprocals is ready as soon as the scan unit receives the first sorted element of the frame from the sort unit, which happens one clock cycle after receiving the last unsorted element. As the scan unit receives the sorted elements $\hat{y}_k$, it updates the value of the quantity $V$ according to the line 15 of Algorithm 1. The rest of the scan unit contains arithmetic logic to compute SURE corresponding to line 7 of the Algorithm 1 (with modifications detailed in Section III-D). Finally, the registers and the comparator at the right end of the scan unit in Figure 7 implement the conditional assignments corresponding to the lines 8 to 11 of the algorithm.

3) Beamspace-to-Antenna (B2A) Conversion Module: As shown in Figure 6, the B2A conversion module resembles that of the A2B module. This module contains a rotation CORDIC, implemented by Xilinx LogiCORE IP, to transform the denoised entries from polar into Cartesian coordinates, and a Xilinx LogiCORE FFT IP to convert the denoised beamspace entries into the antenna domain. The FFT core is configured to perform an IFFT without scaling in any of its stages. The unscaled configuration results in a word-length growth in every stage. However, since the beamspace domain signals are already scaled by the FFT core in the A2B module, the same word-length as the input channel entries is sufficient to accommodate the dynamic range of the outputs from the unscaled IFFT.

C. Fixed-Point Parameters

To maximize hardware-efficiency, we use two’s complement fixed-point arithmetic. The number of bits used for signals in our implementation has been determined based on extensive bit error-rate (BER) simulations, with the goal of achieving near-floating-point performance while minimizing area. For the antenna domain channel entries, we use 16 bit of which are 8 fractional bits. Due to the FFT scaling described in Section IV-B1, 10 bits are sufficient for the beamspace vector entries. Therefore, the entries of the vectors $\hat{y}$ and $\hat{y}^*$ are represented with 10 bit of which are 8 fractional bits, both in the Cartesian and polar coordinates. Since the SBD module (cf. Figure 5) operates only on the beamspace domain data, most of its signals are represented by 10 bit, with the exception of some intermediate signals in the scan unit, which have customized word-lengths to accommodate the dynamic range growth caused by multiplications and additions. For the entries of the LUT used to compute reciprocals in the scan unit, we used 12 bit of which 2 are fractional bits. The BER and MSE of our fixed-point architecture are shown in Figure 4 and Figure 5 respectively, where “fp” stands for fixed-point performance. Clearly, the loss due to finite-precision arithmetic is negligible compared to the reference floating-point MATLAB model.

D. FPGA Implementation Results

To demonstrate the efficacy of BEACHES in practice, we have implemented our architecture on a Xilinx Virtex-7 XC7VX690T FPGA (speed grade $-3$) for various BS antenna configurations ($B = 64, 128, 256, 512$). The implementation results are summarized in Table II and confirm the low complexity of BEACHES when implemented in hardware. In fact, the resource utilization (in terms of slices, LUTs, flip-flops, DSP48 units, and block RAMs) is within a few percent of the total FPGA resources. Furthermore, we observe that the resource utilization (measured in terms of LUTs and flip-flops) increase roughly linearly with the number of BS antennas, which is mainly due to the fact that the number of comparison PEs in the SAS module grows linearly in $B$. Similarly, we
TABLE II
IMPLEMENTATION RESULTS FOR DIFFERENT NUMBERS OF BS ANTENNAS ON A XILINX VIRTEX-7 XC7VX690T FPGA.

| BS antennas B | 64      | 128     | 256     | 512     |
|---------------|---------|---------|---------|---------|
| Slices        | 1532 (1.41%) | 2099 (1.94%) | 3089 (2.85%) | 4886 (4.51%) |
| LUTs          | 4564 (1.05%) | 6391 (1.48%) | 9394 (2.17%) | 14449 (3.34%) |
| – logic LUTs  | 3970 (0.92%) | 5566 (1.28%) | 8336 (1.92%) | 13523 (3.12%) |
| – memory LUTs | 594 (0.34%)  | 825 (0.47%)  | 1058 (0.61%) | 926 (0.53%)   |
| Flipflops     | 24 (0.67%)  | 32 (0.89%)  | 32 (0.89%)  | 40 (1.11%)    |
| DSP48 units   | 6 (0.67%)   | 10 (1.11%)  | 13 (1.11%)  | 17 (1.11%)    |
| Block RAMs    | 1 (0.07%)   | 1 (0.07%)   | 2 (0.14%)   | 5.5 (0.37%)   |
| Max. clock frequency [MHz] | 303 | 303 | 303 | 294 |
| Latency [clock cycles] | 575 | 972 | 1752 | 3301 |
| Throughput [Mvectors/s] | 4.73 | 2.36 | 1.18 | 0.57 |
| Power consumption [W] | 0.76 | 0.87 | 0.98 | 1.29 |
| Efficiency [Mentries/s/LUT] | 66 389 | 47 410 | 32 255 | 29 070 |

aThe throughput is given in million vectors denoised per second and calculated as \( f/B \), where \( f \) is the maximum clock frequency.
bStatistical power estimation at maximum clock frequency and for 1.0 V supply voltage.

TABLE III
FPGA RESOURCE BREAKDOWN FOR DIFFERENT NUMBERS OF BS ANTENNAS ON A XILINX VIRTEX-7 XC7VX690T FPGA.

| BS antennas B | 64      | 128     | 256     | 512     |
|---------------|---------|---------|---------|---------|
| Module        | A2B     | SBD     | B2A     | A2B     | SBD     | B2A     | A2B     | SBD     | B2A     | A2B     | SBD     | B2A     |
| LUTs          | 1650    | 1517    | 1408    | 1947    | 2798    | 1658    | 2176    | 5331    | 1899    | 2381    | 9985    | 2092    |
| – logic LUTs  | 1354    | 1450    | 1177    | 1542    | 2691    | 1345    | 1701    | 5144    | 1503    | 1885    | 9978    | 1669    |
| – memory LUTs | 296     | 67      | 231     | 405     | 107     | 313     | 475     | 187     | 396     | 496     | 7       | 423     |
| Flipflops     | 2470    | 994     | 2097    | 2834    | 1769    | 2412    | 3190    | 3312    | 2780    | 3646    | 6374    | 3113    |
| DSP48 units   | 9       | 5       | 10      | 13      | 5       | 14      | 13      | 5       | 14      | 17      | 5       | 18      |

We conclude by noting that there exist, to our knowledge, no channel vector denoising designs in the literature that would allow us to perform a fair comparison. However, a few results in the literature are concerned with hardware designs for sparsity-based channel estimation algorithms, such as [23], [41], [42]. The hardware designs reported in [23] are for wideband single-input single-output (SISO) channels in 3GPP-LTE systems. These implementations exploit sparsity in the delay domain and are based on three serial greedy pursuit algorithms, namely matching pursuit, gradient pursuit, and OMP. The FPGA design reported in [41], focuses on channel estimation of indoor SISO systems—again, this result exploits sparsity in the time domain. The results in [42] focus on short-range, point-to-point, indoor communication with hybrid precoding—a direct comparison of these methods to our work is difficult. We reiterate that all these results do not focus on massive MU-MIMO mmWave denoising in the beamspace domain and require the user to set certain algorithm parameters. In contrast, BEACHES is used to perform denoising in the beamspace domain while only requiring an estimate of the noise variance.

V. CONCLUSIONS
We have proposed a nonparametric channel estimation algorithm for massive MU-MIMO mmWave systems, which we call BEAmspace CHannel Estimation (BEACHES). BEACHES exploits channel sparsity of mmWave channels in the beamspace domain in order to perform adaptive denoising via Stein’s unbiased risk estimate (SURE). We have established that BEACHES achieves MSE-optimal performance in the large-antenna limit. For realistic LoS and non-LoS mmWave channel models, we have shown that BEACHES performs on par with sophisticated channel estimation algorithms in terms of uncoded bit-rate performance but at orders-of-magnitude lower complexity. As a direct consequence of the nonparametric nature of our algorithm, BEACHES continues to minimize the channel estimation MSE even in scenarios where no sparsity can be exploited (e.g., for Rayleigh fading channels).

In order to demonstrate the practical efficacy of BEACHES, we have developed reference FPGA implementations for massive MU-MIMO mmWave systems with hundreds of BS antennas. Our results are a proof-of-concept that high-
quality mmWave channel estimation can be performed at high throughput and in a hardware-efficient manner.

There are many avenues for future work. An extension of BEACHES to single-carrier (SC) transmission in mmWave channels is a challenging open research problem. A theoretical performance analysis of BS architectures that rely on low-precision analog-to-digital converters (ADCs) in combination with the considered channel estimation methods is left for future work. Finally, alternative sorting architectures should be investigated when targeting systems with thousands of antenna elements.

ACKNOWLEDGMENTS

The authors thank O. Castañeda for his help during the FPGA design, C. Jeon for discussions on SURE-based denoising, and D. Stiverson for his preliminary exploration of low-complexity beamspace channel estimation algorithms for mmWave systems.

APPENDIX A

PROOF OF THEOREM II

We first derive the general form for SURE with complex-valued signals. The MSE for a weakly-differentiable estimator function $\hat{\mu}(\hat{y})$ is defined as

$$MSE = \frac{1}{T} \mathbb{E} \left[ |\hat{\mu}(\hat{y}) - \hat{h}|^2 \right].$$  

(17)

Note that expectation is with respect to the noisy observation $\hat{y}$. We decompose the complex-valued vector $\hat{y}$ into the real part $\hat{y}_R \sim N(\hat{y}_R; \hat{h}_R, \frac{\hat{E}_B}{2} I_B)$ and imaginary part $\hat{y}_I \sim N(\hat{y}_I; \hat{h}_I, \frac{\hat{E}_B}{2} I_B)$ and define $g(\hat{y}) = \hat{\mu}(\hat{y}) - \hat{y}$. Hence,

$$MSE = \frac{1}{T} \mathbb{E} \left[ \left| g(\hat{y}) \right|^2 \right] + \frac{1}{T} \mathbb{E} \left[ \left| \hat{y} - \hat{h} \right|^2 \right] + \frac{1}{T} \mathbb{E} \left[ 2 \left| g(\hat{y})^R (\hat{y} - \hat{h}) \right|_R \right].$$  

(19)

The last term can be expanded as follows:

$$\frac{2}{T} \mathbb{E} \left[ g(\hat{y})^R (\hat{y} - \hat{h}) \right]_R = \frac{2}{T} \mathbb{E} \left[ g(\hat{y})^T (\hat{y} - \hat{h}) \right] + \frac{2}{T} \mathbb{E} \left[ g(\hat{y})^T (\hat{y}_I - \hat{h}_I) \right].$$

(20)

We can now expand $\frac{2}{T} \mathbb{E} \left[ g(\hat{y})^T (\hat{y} - \hat{h}) \right]$, which yields

$$\frac{2}{T} \mathbb{E} \left[ g(\hat{y})^T (\hat{y} - \hat{h}) \right] = \frac{2}{T} \int_{\mathbb{C}^N} f^C(\hat{y}; \hat{h}, E_0 I_B) \sum_{b=1}^B \left| g(\hat{y}) \right|_b \times \left( \hat{y}_R - \hat{y}_R \right) d\hat{y}$$

(21)

$$= \frac{2}{T} \int_{\mathbb{C}^N} f^C(\hat{y}; \hat{h}, E_0 I_B) \sum_{b=1}^B \int_{y_R} \frac{1}{(2\pi)^{N/2}} e^{-\frac{(y_R - \hat{y}_R)^2}{2\hat{E}_B}} \times \exp \left( -\frac{1}{2\hat{E}_B} \left[ g(\hat{y}) \right]_b \times \left( \hat{y}_R - \hat{y}_R \right) \right) d\hat{y} d\hat{y}_I$$

(22)

By remembering that $g(\hat{y}) = \hat{\mu}(\hat{y}) - \hat{y}$, applying the relationship (7) reduces to

$$SURE = \frac{1}{B} \sum_{b=1}^B \min \left\{ |\hat{y}_b|, \tau \right\}^2 + E_0$$

For $|\hat{y}_b| > \tau$, we have

$$\frac{\partial g(\hat{y}_b)}{\partial \hat{y}_b} = \frac{\partial \hat{\mu}(\hat{y}_b)}{\partial \hat{y}_b} = -\frac{\tau \hat{y}_b}{\sqrt{\left\| \hat{y}_b \right\|_2^2 + \tau^2}}$$

(31)

and

$$\frac{\partial g(\hat{y}_b)}{\partial \hat{y}_b} = \frac{\partial \hat{\mu}(\hat{y}_b)}{\partial \hat{y}_b} = -\frac{\tau \hat{y}_b}{\sqrt{\left\| \hat{y}_b \right\|_2^2 + \tau^2}}$$

(32)

Note that the derivative of $|\hat{y}_b|$, $|\hat{y}_b|$, $|\hat{y}_b|$ has a discontinuity at $\tau = |\hat{y}_b|$, $\hat{y}_b$, and thus, SURE is not defined for this value. Using (30), (31), and (32), the complex-valued SURE expression (7) reduces to

$$SURE = \frac{1}{B} \sum_{b=1}^B \min \left\{ |\hat{y}_b|, \tau \right\}^2 + E_0$$

+ $E_0 \sum_{b=1}^B \mathbb{E} \left[ |\hat{y}_b| - \tau \right] \sqrt{|\hat{y}_b|_2^2 + \tau^2} - 2$ (33)

+ $E_0 \sum_{b=1}^B \mathbb{E} \left[ 0 - \tau \right] \sqrt{|\hat{y}_b|_2^2 + \tau^2} - 2$ (34)
where,
\[
\lim_{B \to \infty} \frac{1}{B} \sum_{b=1}^{B} \gamma(\hat{y}_b, \tau, \hat{h}_b) = E [\gamma(H + \sqrt{\mathbb{E} G}, \tau), H].
\]"
evaluation,” *IEEE Journal on Selected Areas in Communications*, vol. 32, no. 6, pp. 1164–1179, Jun. 2014.

[31] D. Tse and P. Viswanath, *Fundamentals of Wireless Communication*. New York, NY, USA: Cambridge University Press, 2005.

[32] S. Jäckel, L. Raschkowski, K. Börner, L. Thiele, F. Burkhardt, and E. Eberlein, “QuaDRiGa - Quasi Deterministic Radio Channel Generator User Manual and Documentation,” Fraunhofer Heinrich Hertz Institute, Tech. Rep. v2.0.0, Aug. 2017.

[33] D. L. Donoho and I. M. Johnstone, “Adapting to unknown smoothness via wavelet shrinkage,” *Journal of the American Statistical Association*, vol. 90, no. 432, pp. 1200–1224, Dec. 1995.

[34] R. Tibshirani, “Regression shrinkage and selection via the LASSO,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, Jan. 1996.

[35] J. A. Tropp and S. J. Wright, “Computational methods for sparse solution of linear inverse problems,” *Proceedings of the IEEE*, vol. 98, no. 6, pp. 948–958, Jun. 2010.

[36] A. Maleki, L. Anitori, Z. Yang, and R. G. Baraniuk, “Asymptotic analysis of complex LASSO via complex approximate message passing (CAMP),” *IEEE Transactions on Information Theory*, vol. 59, no. 7, pp. 4290–4308, Jul. 2013.

[37] A. Mousavi, A. Maleki, and R. G. Baraniuk, “Parameterless optimal approximate message passing,” Oct. 2013. [Online]. Available: http://arxiv.org/abs/1311.0035

[38] K. Upadhya, C. S. Seelamantula, and K. Hari, “A risk minimization framework for channel estimation in OFDM systems,” *Signal Processing*, vol. 128, pp. 78–87, Nov. 2016.

[39] C. Studer, S. Fateh, and D. Seethaler, “ASIC implementation of soft-input soft-output MIMO detection using MMSE parallel interference cancellation,” *IEEE Journal of Solid-State Circuits*, vol. 46, no. 7, pp. 1754–1765, Jul. 2011.

[40] G. H. Golub and C. F. van Loan, *Matrix Computations*, 3rd ed. The Johns Hopkins Univ. Press, 1996.

[41] P. K. Korrai, K. Deergha Rao, and C. Gangadhar, “FPGA implementation of OFDM-based mmWave indoor sparse channel estimation using OMP,” *Springer Circuits, Systems, and Signal Processing*, vol. 37, no. 5, pp. 2194–2205, May 2018.

[42] S. Birke, W. Chen, G. Wang, D. Auras, C. Shen, R. Leupers, and G. Ascheid, “VLSI implementation of channel estimation for millimeter wave beamforming training,” in *IEEE Latin American Symposium on Circuits Systems (LASCAS)*, Feb. 2018, pp. 1–4.

[43] A. Mousavi, A. Maleki, and R. G. Baraniuk, “Consistent parameter estimation for LASSO and approximate message passing,” *arXiv preprint: 1511.01017 [math.ST]*, Nov. 2015.