Algorithms for Interpolation-Based QR Decomposition in MIMO-OFDM Systems

Davide Cescato, *Student Member, IEEE,* and Helmut Bölcskei, *Fellow, IEEE*

***Abstract*—Detection algorithms for multiple-input multiple-output (MIMO) wireless systems based on orthogonal frequency-division multiplexing (OFDM) typically require the computation of a QR decomposition for each of the data-carrying OFDM tones. The resulting computational complexity will, in general, be significant. Motivated by the fact that the channel matrices arising in MIMO-OFDM systems result from oversampling of a polynomial matrix, we formulate interpolation-based QR decomposition algorithms. An in-depth complexity analysis, based on a metric relevant for very large scale integration (VLSI) implementations, shows that the proposed algorithms, for a sufficiently large number of data-carrying tones and sufficiently small channel order, provably exhibit significantly smaller complexity than brute-force per-tone QR decomposition.**

***Index Terms*—Interpolation, polynomial matrices, multiple-input multiple-output (MIMO) systems, orthogonal frequency-division multiplexing (OFDM), QR decomposition, successive cancelation, sphere decoding, very large scale integration (VLSI).**

I. Introduction

The use of orthogonal frequency-division multiplexing (OFDM) drastically reduces data detection complexity in wideband multiple-input multiple-output (MIMO) wireless systems by decoupling a frequency-selective fading MIMO channel into a set of flat-fading MIMO channels. Nevertheless, MIMO-OFDM detectors still pose significant challenges in terms of computational complexity, as processing has to be performed on a per-tone basis.

Specifically, in the setting of coherent MIMO-OFDM detection, where the receiver is assumed to have perfect channel knowledge, linear detectors [1] require matrix inversion, whereas successive cancelation receivers [2] and sphere decoders [3], [4] require QR decomposition, in all cases on each of the data-carrying OFDM tones. The corresponding computations, termed as *preprocessing* in the following, have to be performed at the rate of change of the channel which, depending on the propagation environment, is typically much lower than the rate at which the transmission of actual data symbols takes place. Nevertheless, as payload data received during the preprocessing phase must be stored in a dedicated buffer, preprocessing represents a major bottleneck in terms of the size of this buffer and the resulting detection latency [5].

This work was supported in part by the Swiss National Science Foundation under grant No. 200021-100025/1. Parts of this paper were presented at the Sixth IEEE Workshop on Signal Processing Advances in Wireless Communications (SPAWC), New York, NY, June 2005. The authors are with the Communication Technology Laboratory, ETH Zurich, 8092 Zurich, Switzerland. E-mail: {dcescato, boelcskei}@nari.ee.ethz.ch.

In a very large scale integration (VLSI) implementation, the straightforward approach to reducing the preprocessing latency is to employ parallel processing over multiple matrix inversion or QR decomposition units, which, however, comes at the cost of increased silicon area. In [6], the problem of reducing preprocessing complexity in linear MIMO-OFDM receivers is addressed on an algorithmic level by formulating efficient interpolation-based algorithms for matrix inversion that take the polynomial nature of the MIMO channel transfer function explicitly into account. Specifically, the algorithms proposed in [6] exploit the fact that the channel matrices arising in MIMO-OFDM systems result from oversampling of the polynomial MIMO channel transfer function on the unit circle. The goal of the present paper is to devise computationally efficient interpolation-based algorithms for QR decomposition in MIMO-OFDM systems, using results on QR decomposition of Laurent polynomial (LP) matrices recently presented in [7]. Differently from the approach reported in [8], which approximates QR factors of a polynomial matrix (that, as shown in [7], in general are neither LP nor rational matrices) by LP matrices, the algorithms presented in this paper yield *exact* QR factors. Although, throughout the paper, we focus on QR decomposition in the context of coherent MIMO-OFDM detectors, our results also apply to transmit precoding schemes for MIMO-OFDM (under the assumption of perfect channel knowledge at the transmitter) requiring per-tone QR decomposition, both for point-to-point MIMO channels [9] and for multiantenna broadcast channels [10]. In [11], it is shown that unitary precoding matrices for transmit beamforming (based on singular value decomposition rather than QR decomposition) in MIMO-OFDM systems with limited feedback can be computed efficiently through inexact interpolation under unitarity constraints.

The redundancy in the channel transfer function samples arising from the use of OFDM (both in the single-antenna case and in the MIMO case) can be exploited in the context of channel estimation, as done, e.g., in [12], [13], [14] for single-antenna systems. In this paper, we show how this property can be used to reduce the complexity of computing QR factors of MIMO-OFDM channel matrices. Finally, we mention that follow-up work based on the conference version of this paper [15] was reported in [16].

The contributions of this paper can be summarized as follows. Based on the results in [7], we formulate interpolation-based algorithms for QR decomposition in MIMO-OFDM systems. Using a computational complexity metric relevant for VLSI implementations, we demonstrate that the proposed interpolation-based algorithms can, depending on the system parameters, exhibit significantly smaller complexity

than brute-force per-tone QR decomposition. Furthermore, we present strategies for efficient interpolation of LPs that take the specific structure of the problem at hand into account. Finally, we provide a numerical analysis of the trade-off between the computational complexity of the proposed interpolation-based QR decomposition algorithms and the performance of corresponding MIMO-OFDM detectors.

II. Mathematical Preliminaries

A. Notation

$\mathbb{C}^{P\times M}$ denotes the set of complex-valued $P\times M$ matrices. $\mathcal{U} \triangleq \{s\in\mathbb{C} : |s|=1\}$ indicates the unit circle. $\emptyset$ is the empty set. $|\mathcal{A}|$ stands for the cardinality of the set $\mathcal{A}$. mod is the modulo operator. All logarithms are to the base 2. $\mathbb{E}[\cdot]$ denotes the expectation operator. $\mathcal{CN}(0,\sigma^2)$ stands for the circularly-symmetric complex Gaussian distribution with variance σ^2. Throughout the paper, we use the following conventions. First, if $k_2<k_1$, $\sum_{k=k_1}^{k_2}\alpha_k=0$, regardless of α_k. Second, sequences of integers of the form $k_1, k_1+\Delta,\ldots,k_2$, with $\Delta>0$, simplify to the sequence k_1,k_2 if $k_2=k_1+\Delta$, to the single value k_1 if $k_2=k_1$, and to the empty sequence if $k_2<k_1$.

$\mathbf{A}^*$, $\mathbf{A}^T$, $\mathbf{A}^H$, $\mathbf{A}^\dagger$, and $\mathrm{rank}(\mathbf{A})$ denote the entrywise conjugate, the transpose, the conjugate transpose, the pseudoinverse, and the rank, respectively, of the matrix $\mathbf{A}$. $[\mathbf{A}]_{p,m}$ indicates the entry in the pth row and mth column of $\mathbf{A}$. $\mathbf{A}^{p_1,p_2}$ and $\mathbf{A}_{m_1,m_2}$ stand for the submatrix given by the rows $p_1,p_1+1,\ldots,p_2$ of $\mathbf{A}$ and the submatrix given by the columns $m_1,m_1+1,\ldots,m_2$ of $\mathbf{A}$, respectively. Furthermore, we set $\mathbf{A}^{p_1,p_2}_{m_1,m_2}\triangleq(\mathbf{A}_{m_1,m_2})^{p_1,p_2}$ and $\mathbf{A}^H_{m_1,m_2}\triangleq(\mathbf{A}_{m_1,m_2})^H$. $\mathrm{diag}(a_1,a_2,\ldots,a_M)$ indicates the $M\times M$ diagonal matrix with the scalar a_m as its mth main diagonal element. $\mathbf{I}_M$ is the $M\times M$ identity matrix. $\mathbf{0}$ denotes the all-zeros matrix of appropriate size. Column vectors and row vectors are represented by lower-case bold symbols and by lower-case bold underlined symbols, respectively. Finally, orthogonality and norm of complex-valued column vectors $\mathbf{a}_1,\mathbf{a}_2$ are induced by the inner product $\mathbf{a}_1^H\mathbf{a}_2$.

B. QR Decomposition

We consider a matrix $\mathbf{A}\in\mathbb{C}^{P\times M}$ with $P\geq M$. In this section, mostly taken from [7], we briefly review some basics on QR decomposition along with related results that will be needed later in the paper.

Definition 1. We call any factorization $\mathbf{A}=\mathbf{QR}$, for which the matrices $\mathbf{Q}\in\mathbb{C}^{P\times M}$ and $\mathbf{R}\in\mathbb{C}^{M\times M}$ satisfy the following conditions, a *QR decomposition* of $\mathbf{A}$ with *QR factors* $\mathbf{Q}$ and $\mathbf{R}$:

1) the nonzero columns of $\mathbf{Q}$ are orthonormal
2) $\mathbf{R}$ is upper triangular with real-valued nonnegative entries on its main diagonal
3) $\mathbf{R}=\mathbf{Q}^H\mathbf{A}$

Practical algorithms for QR decomposition are either based on Gram-Schmidt (GS) orthonormalization or on unitary transformations (UT) [17].

Definition 2. The *ordered column rank* of $\mathbf{A}$ is the number

$$K\triangleq\begin{cases}0, & \mathrm{rank}(\mathbf{A}_{1,1})=0\\ \max\{k:\mathrm{rank}(\mathbf{A}_{1,k})=k\}, & \text{else.}\end{cases}$$

Proposition 3. *If $\mathbf{A}$ has ordered column rank $K>0$, then $\mathbf{Q}_{1,K}$ and $\mathbf{R}^{1,K}$ are unique and satisfy*

1) $\mathbf{Q}_{1,K}^H\mathbf{Q}_{1,K}=\mathbf{I}_K$
2) $[\mathbf{R}^{1,K}]_{k,k}=[\mathbf{R}]_{k,k}>0,\quad k=1,2,\ldots,K.$

We note that for full-rank $\mathbf{A}$, we have $K=M$. In this case, the uniqueness of $\mathbf{Q}$ and $\mathbf{R}$ implies that $\mathbf{A}=\mathbf{QR}$ can be called *the* QR decomposition of $\mathbf{A}$ with *the* QR factors $\mathbf{Q}$ and $\mathbf{R}$. For $\mathrm{rank}(\mathbf{A})<M$, different QR decomposition algorithms will in general produce different QR factors. Throughout the paper, whenever $\mathbf{A}$ is not guaranteed to have full rank, we simply speak of *a* QR decomposition of $\mathbf{A}$ with QR factors $\mathbf{Q}$ and $\mathbf{R}$.

Proposition 4. *Let $\mathbf{A}=\mathbf{QR}$ be a QR decomposition of $\mathbf{A}$. Then, for $M>1$ and for a given $k\in\{2,3,\ldots,M\}$, $\mathbf{A}_{k,M}-\mathbf{Q}_{1,k-1}\mathbf{R}_{k,M}^{1,k-1}=\mathbf{Q}_{k,M}\mathbf{R}_{k,M}^{k,M}$ is a QR decomposition of $\mathbf{A}_{k,M}-\mathbf{Q}_{1,k-1}\mathbf{R}_{k,M}^{1,k-1}$.*

Definition 5. The *regularized QR decomposition* of $\mathbf{A}$ with the real-valued *regularization parameter* $\alpha>0$, is the unique factorization $\mathbf{A}=\mathbf{QR}$, where the *regularized QR factors* $\mathbf{Q}\in\mathbb{C}^{P\times M}$ and $\mathbf{R}\in\mathbb{C}^{M\times M}$ are obtained as follows: $\bar{\mathbf{A}}=\bar{\mathbf{Q}}\mathbf{R}$ is the unique QR decomposition of the full-rank $(P+M)\times M$ augmented matrix $\bar{\mathbf{A}}\triangleq[\,\mathbf{A}^T\quad\alpha\mathbf{I}_M\,]^T$, and $\mathbf{Q}\triangleq\bar{\mathbf{Q}}^{1,P}$.

C. Laurent Polynomials and Interpolation

In the following, we review basic results on the interpolation of LPs and establish the corresponding notation. Various strategies for computationally efficient interpolation of LPs making use of the specific structural properties of the problem at hand are presented in Section VIII.

Definition 6. Given a matrix-valued function $\mathbf{A}:\mathcal{U}\to\mathbb{C}^{P\times M}$ and integers $V_1,V_2\geq 0$, the notation $\mathbf{A}(s)\sim(V_1,V_2)$ indicates that there exist coefficient matrices $\mathbf{A}_v\in\mathbb{C}^{P\times M}$, $v=-V_1,-V_1+1,\ldots,V_2$, such that

$$\mathbf{A}(s)=\sum_{v=-V_1}^{V_2}\mathbf{A}_v s^{-v},\qquad s\in\mathcal{U}. \tag{1}$$

If $\mathbf{A}(s)\sim(V_1,V_2)$, then $\mathbf{A}(s)$ is a *Laurent polynomial* (LP) matrix with *maximum degree* V_1+V_2.

In the remainder of this section, we consider the LP $a(s)\sim(V_1,V_2)$ with maximum degree $V\triangleq V_1+V_2$. The following results can be directly extended to the interpolation of LP matrices through entrywise application. Borrowing terminology from signal analysis, we call the value of $a(s)$ at a given point $s_0\in\mathcal{U}$ the *sample* $a(s_0)$.

Definition 7. *Interpolation* of the LP $a(s)\sim(V_1,V_2)$ from the set $\mathcal{B}=\{b_0,b_1,\ldots,b_{B-1}\}\subset\mathcal{U}$, containing B distinct *base points*, to the set $\mathcal{T}=\{t_0,t_1,\ldots,t_{T-1}\}\subset\mathcal{U}$, containing T distinct *target points,* is the process of obtaining the samples $a(t_0),a(t_1),\ldots,a(t_{T-1})$ from the

samples $a(b_0), a(b_1), \ldots, a(b_{B-1})$, with knowledge of V_1 and V_2, but without explicit knowledge of the coefficients $a_{-V_1}, a_{-V_1+1}, \ldots, a_{V_2}$ that determine $a(s)$ according to (1).

In the following, we assume that $B \geq V+1$. By defining the vectors $\mathbf{a} \triangleq [a_{-V_1} \;\; a_{-V_1+1} \;\; \cdots \;\; a_{V_2}]^T$, $\mathbf{a}_{\mathcal{B}} \triangleq [a(b_0) \;\; a(b_1) \;\; \cdots \;\; a(b_{B-1})]^T$, and $\mathbf{a}_{\mathcal{T}} \triangleq [a(t_0) \;\; a(t_1) \;\; \cdots \;\; a(t_{T-1})]^T$, we note that $\mathbf{a}_{\mathcal{B}} = \mathbf{B}\mathbf{a}$, with the $B \times (V+1)$ *base point matrix* $[\mathbf{B}]_{i,v} = b_{i-1}^{V_1-v+1}$ $(i = 1,2,\ldots,B,\; v = 1,2,\ldots,V+1)$ and $\mathbf{a}_{\mathcal{T}} = \mathbf{T}\mathbf{a}$, with the $T \times (V+1)$ *target point matrix* $[\mathbf{T}]_{i,v} = t_{i-1}^{V_1-v+1}$ $(i = 1,2,\ldots,T,\; v = 1,2,\ldots,V+1)$. Now, $\mathbf{B}$ can be written as $\mathbf{B} = \mathbf{D}_{\mathcal{B}}\mathbf{V}_{\mathcal{B}}$, where $\mathbf{D}_{\mathcal{B}} \triangleq \operatorname{diag}(b_0^{V_1}, b_1^{V_1}, \ldots, b_{B-1}^{V_1})$ and $\mathbf{V}_{\mathcal{B}}$ is the $B \times (V+1)$ Vandermonde matrix $[\mathbf{V}_{\mathcal{B}}]_{i,v} = b_{i-1}^{-v+1}$ $(i = 1,2,\ldots,B,\; v = 1,2,\ldots,V+1)$. Since the base points $b_0, b_1, \ldots, b_{B-1}$ are distinct, $\mathbf{V}_{\mathcal{B}}$ has full rank [18]. Hence, $\operatorname{rank}(\mathbf{V}_{\mathcal{B}}) = V+1$, which, together with the fact that $\mathbf{D}_{\mathcal{B}}$ is nonsingular, implies that $\operatorname{rank}(\mathbf{B}) = V+1$. Therefore, the coefficient vector $\mathbf{a}$ is uniquely determined by the B samples of $a(s)$ at the base points $b_0, b_1, \ldots, b_{B-1}$ according to $\mathbf{a} = \mathbf{B}^{\dagger}\mathbf{a}_{\mathcal{B}}$, and interpolation of $a(s)$ from $\mathcal{B}$ to $\mathcal{T}$ can be performed by computing

$$\mathbf{a}_{\mathcal{T}} = \mathbf{T}\mathbf{B}^{\dagger}\mathbf{a}_{\mathcal{B}}. \tag{2}$$

In the remainder of the paper, we call the $T \times B$ matrix $\mathbf{T}\mathbf{B}^{\dagger}$ the *interpolation matrix*.

For later use, we briefly comment on interpolation from noisy samples at the base points. Specifically, for $\mathbf{a}_{\mathcal{B}}$ subject to additive noise $\mathbf{w}_{\mathcal{B}}$, there exists, in general, no LP $\tilde{a}(s) \sim (V_1, V_2)$ such that the entries of the vector $\mathbf{a}_{\mathcal{B}} + \mathbf{w}_{\mathcal{B}}$ can be seen as the samples of $\tilde{a}(s)$ at the base points $\mathcal{B}$. However, as is easily verified, interpolation according to (2), with $\mathbf{a}_{\mathcal{B}}$ replaced by $\mathbf{a}_{\mathcal{B}} + \mathbf{w}_{\mathcal{B}}$, forces the resulting vector $\mathbf{T}\mathbf{B}^{\dagger}(\mathbf{a}_{\mathcal{B}} + \mathbf{w}_{\mathcal{B}})$ to consist of samples of an LP $\hat{a}(s) \sim (V_1, V_2)$ at the target points $\mathcal{T}$.

We conclude this section by noting that in the special case $V_1 = V_2$, we have $\mathbf{B} = \mathbf{B}^*\mathbf{E}$ and $\mathbf{T} = \mathbf{T}^*\mathbf{E}$, where the $(V+1) \times (V+1)$ matrix $\mathbf{E}$ is obtained by flipping $\mathbf{I}_{V+1}$ upside down. Since the operation of taking the pseudoinverse commutes with entrywise conjugation, it follows that $\mathbf{B}^{\dagger} = \mathbf{E}(\mathbf{B}^{\dagger})^*$ and, as a consequence of $\mathbf{E}^2 = \mathbf{I}_{V+1}$, we obtain $\mathbf{T}\mathbf{B}^{\dagger} = (\mathbf{T}\mathbf{B}^{\dagger})^*$, i.e., the interpolation matrix is real-valued.

III. MIMO-OFDM and Problem Statement

A. System Model

We consider a MIMO system [1] with M_T transmit and M_R receive antennas. We make the conceptual assumption $M_R \geq M_T$ throughout the paper, as an M_T-dimensional signal can, in general, not be recovered from an observation of lower dimensionality. The matrix-valued impulse response of the frequency-selective MIMO channel is given by the taps $\mathbf{H}_l \in \mathbb{C}^{M_R \times M_T}$ $(l = 0,1,\ldots,L)$ with the corresponding matrix-valued transfer function

$$\mathbf{H}\big(e^{j2\pi\theta}\big) = \sum_{l=0}^{L} \mathbf{H}_l e^{-j2\pi l\theta}, \qquad 0 \leq \theta < 1$$

which satisfies $\mathbf{H}(s) \sim (0, L)$. Under OFDM signaling, the use of a cyclic prefix of length $L_{\mathrm{CP}} \geq L$ essentially turns the action of the channel (in the single-input single-output case) into the multiplication by a circulant matrix, which is diagonalized by the discrete Fourier transform. The effective input-output relation in a MIMO-OFDM system with $L_{\mathrm{CP}} \geq L$ and N OFDM tones is therefore given by [19]

$$\mathbf{d}_n = \mathbf{H}\big(s_n\big)\mathbf{c}_n + \mathbf{w}_n, \qquad n = 0,1,\ldots,N-1$$

with the tone index n, the transmit signal vector $\mathbf{c}_n \triangleq [c_{n,1} \;\; c_{n,2} \;\; \cdots \;\; c_{n,M_T}]^T$, the receive signal vector $\mathbf{d}_n \triangleq [d_{n,1} \;\; d_{n,2} \;\; \cdots \;\; d_{n,M_R}]^T$, the additive noise vector $\mathbf{w}_n$, and $s_n \triangleq e^{j2\pi n/N}$. Here, $c_{n,m}$ stands for the complex-valued data symbol, taken from a finite constellation $\mathcal{O}$, transmitted by the mth antenna on the nth tone and $d_{n,m}$ is the signal observed at the mth receive antenna on the nth tone. For $n = 0,1,\ldots,N-1$, we assume that $\mathbf{c}_n$ contains statistically independent entries and satisfies $\mathbb{E}[\mathbf{c}_n] = \mathbf{0}$ and $\mathbb{E}[\mathbf{c}_n^H\mathbf{c}_n] = 1$. Again for $n = 0,1,\ldots,N-1$, we assume that $\mathbf{w}_n$ is statistically independent of $\mathbf{c}_n$ and contains entries that are independent and identically distributed (i.i.d.) as $\mathcal{CN}(0, \sigma_w^2)$, where σ_w^2 denotes the noise variance and is assumed to be known at the receiver.

In practice, N is typically chosen to be a power of two in order to allow for efficient OFDM processing based on the Fast Fourier Transform (FFT). Moreover, a small subset of the N tones is typically set aside for pilot symbols and virtual tones at the frequency band edges, which help to reduce out-of-band interference and relax the pulse-shaping filter requirements. We collect the indices corresponding to the D tones carrying payload data into the set $\mathcal{D} \subseteq \{0,1,\ldots,N-1\}$. Typical OFDM systems have $D \geq 3L_{\mathrm{CP}}$.

B. QR Decomposition in MIMO-OFDM Detectors

Widely used algorithms for coherent detection in MIMO-OFDM systems include successive cancelation (SC) detectors [1], both zero-forcing (ZF) and minimum mean-square error (MMSE) [2], [20], as well as sphere decoders, both in the original formulation [3], [4] requiring ZF-based preprocessing, and in the MMSE-based form proposed in [21]. These detection algorithms require QR decomposition in the preprocessing step, or, more specifically, computation of matrices $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$, $n \in \mathcal{D}$, defined as follows. In the ZF case, $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$ are QR factors of $\mathbf{H}(s_n)$, whereas in the MMSE case, $\mathbf{Q}(s_n)\mathbf{R}(s_n)$ is the *MMSE-QR decomposition* of $\mathbf{H}(s_n)$, defined as the special case of regularized QR decomposition of $\mathbf{H}(s_n)$ obtained by choosing the regularization parameter as $\alpha = \sqrt{M_T}\sigma_w$. Both for ZF-based and MMSE-based preprocessing, SC detectors essentially solve the linear system of equations $\mathbf{Q}^H(s_n)\mathbf{d}_n = \mathbf{R}(s_n)\hat{\mathbf{c}}_n$ through back-substitution (with rounding of the intermediate results to elements of $\mathcal{O}$ [1]) to obtain $\hat{\mathbf{c}}_n \in \mathcal{O}^{M_T}$, whereas sphere decoders exploit the upper triangularity of $\mathbf{R}(s_n)$ to find the vector $\hat{\mathbf{c}}_n \in \mathcal{O}^{M_T}$ that minimizes $\|\mathbf{Q}^H(s_n)\mathbf{d}_n - \mathbf{R}(s_n)\hat{\mathbf{c}}_n\|^2$ through an efficient tree search [4].

C. Problem Statement

We assume that the MIMO-OFDM receiver has perfect knowledge (obtained, e.g., through channel estimation) of

the samples $\mathbf{H}(s_n)$ for $n \in \mathcal{E} \subseteq \{0, 1, \ldots, N-1\}$, with $|\mathcal{E}| \geq L+1$, from which $\mathbf{H}(s_n)$ can be obtained at any data-carrying tone $n \in \mathcal{D}$ by interpolating $\mathbf{H}(s) \sim (0, L)$. In the special case $\mathcal{D} \subseteq \mathcal{E}$, interpolation of $\mathbf{H}(s)$ is not necessary. We next formulate the problem statement by focusing on ZF-based detectors, requiring QR decomposition of the MIMO-OFDM channel matrices $\mathbf{H}(s_n)$. The problem statement for the MMSE case is analogous with QR decomposition replaced by MMSE-QR decomposition.

The MIMO-OFDM receiver needs to compute QR factors $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$ of $\mathbf{H}(s_n)$ for all data-carrying tones $n \in \mathcal{D}$. A straightforward approach to solving this problem consists of first interpolating $\mathbf{H}(s)$ to obtain $\mathbf{H}(s_n)$ at the tones $n \in \mathcal{D}$ and then performing QR decomposition on a per-tone basis. This method will henceforth be called *brute-force per-tone QR decomposition*. The interpolation-based QR decomposition algorithms presented in this paper are motivated by the following observations. First, performing QR decomposition on an $M \times M$ matrix requires $O(M^3)$ arithmetic operations [17], whereas the number of arithmetic operations involved in computing one sample of an $M \times M$ LP matrix by interpolation is proportional to the number of matrix entries M^2, as interpolation of an LP matrix is performed entrywise. This comparison suggests that we may obtain fundamental savings in computational complexity by replacing QR decomposition by interpolation. Second, consider a flat-fading channel, so that $L = 0$ and hence $\mathbf{H}(s_n) = \mathbf{H}_0$ for all $n = 0, 1, \ldots, N-1$. In this case, a single QR decomposition $\mathbf{H}_0 = \mathbf{QR}$ yields QR factors of $\mathbf{H}(s_n)$ for all data-carrying tones $n \in \mathcal{D}$. A question that now arises naturally is whether for $L > 0$ QR factors $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$, $n \in \mathcal{D}$, can be obtained from a smaller set of QR factors through interpolation. We will see that the answer is in the affirmative and will, moreover, demonstrate that interpolation-based QR decomposition algorithms can yield significant computational complexity savings over brute-force per-tone QR decomposition for a wide range of values of the parameters M_T, M_R, L, N, and D, which will be referred to as *the system parameters* throughout the paper. The key to formulating interpolation-based QR decomposition algorithms that realize these complexity savings are results on QR decomposition of LP matrices recently reported in [7] and briefly reviewed in the next section.

IV. QR Decomposition of LP Matrices

We consider a $P \times M$ LP matrix $\mathbf{A}(s) \sim (V_1, V_2)$, $s \in \mathcal{U}$, with $P \geq M$, and QR factors $\mathbf{Q}(s)$ and $\mathbf{R}(s)$ of $\mathbf{A}(s)$. In [7], it is shown that although $\mathbf{Q}(s)$ and $\mathbf{R}(s)$ are, in general, not LP matrices, there exists a mapping $\mathcal{M}$ that transforms $\mathbf{Q}(s)$ and $\mathbf{R}(s)$ into corresponding LP matrices $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$. This mapping constitutes the basis for the formulation of interpolation-based QR decomposition algorithms for MIMO-OFDM systems.

We review the mapping $\mathcal{M}$ by considering QR factors of $\mathbf{A}(s_0)$ for a given $s_0 \in \mathcal{U}$. In order to keep the notation compact, we omit the dependence of all involved quantities on s_0. In the following, $\mathbf{q}_k$ and $\underline{\mathbf{r}}_k$ denote the kth column of $\mathbf{Q}$ and the kth row of $\mathbf{R}$, respectively ($k = 1, 2, \ldots, M$). We start by defining the auxiliary variables Δ_k as

$$\Delta_k \triangleq \Delta_{k-1} [\mathbf{R}]_{k,k}^2, \qquad k = 1, 2, \ldots, M \tag{3}$$

with $\Delta_0 \triangleq 1$. Next, we introduce the vectors

$$\tilde{\mathbf{q}}_k \triangleq \Delta_{k-1} [\mathbf{R}]_{k,k} \mathbf{q}_k, \qquad k = 1, 2, \ldots, M \tag{4}$$

$$\tilde{\underline{\mathbf{r}}}_k \triangleq \Delta_{k-1} [\mathbf{R}]_{k,k} \underline{\mathbf{r}}_k, \qquad k = 1, 2, \ldots, M \tag{5}$$

and define the mapping $\mathcal{M} : (\mathbf{Q}, \mathbf{R}) \mapsto (\tilde{\mathbf{Q}}, \tilde{\mathbf{R}})$ through $\tilde{\mathbf{Q}} \triangleq [\tilde{\mathbf{q}}_1 \ \tilde{\mathbf{q}}_2 \ \cdots \ \tilde{\mathbf{q}}_M]$ and $\tilde{\mathbf{R}} \triangleq [\tilde{\underline{\mathbf{r}}}_1^T \ \tilde{\underline{\mathbf{r}}}_2^T \ \cdots \ \tilde{\underline{\mathbf{r}}}_M^T]^T$.

In the following, we denote the ordered column rank of $\mathbf{A}$ by K. For $K > 0$ and $k = 1, 2, \ldots, K$, we can compute $\mathbf{q}_k$ and $\underline{\mathbf{r}}_k$ from $\tilde{\mathbf{q}}_k$ and $\tilde{\underline{\mathbf{r}}}_k$, respectively, according to

$$\mathbf{q}_k = (\Delta_{k-1} [\mathbf{R}]_{k,k})^{-1} \tilde{\mathbf{q}}_k \tag{6}$$

$$\underline{\mathbf{r}}_k = (\Delta_{k-1} [\mathbf{R}]_{k,k})^{-1} \tilde{\underline{\mathbf{r}}}_k \tag{7}$$

where $\Delta_{k-1} [\mathbf{R}]_{k,k}$ is obtained from the entries on the main diagonal of $\tilde{\mathbf{R}}$ as

$$\Delta_{k-1} [\mathbf{R}]_{k,k} = \begin{cases} \sqrt{[\tilde{\mathbf{R}}]_{k,k}}, & k = 1 \\ \sqrt{[\tilde{\mathbf{R}}]_{k-1,k-1} [\tilde{\mathbf{R}}]_{k,k}}, & k = 2, 3, \ldots, K. \end{cases} \tag{8}$$

The *inverse mapping* $\mathcal{M}^{-1} : (\tilde{\mathbf{Q}}, \tilde{\mathbf{R}}) \mapsto (\mathbf{Q}, \mathbf{R})$ corresponds to the following procedure:[1]

1) If $K > 0$, for $k = 1, 2, \ldots, K$, compute the scaling factor $(\Delta_{k-1} [\mathbf{R}]_{k,k})^{-1}$ using (8) and scale $\tilde{\mathbf{q}}_k$ and $\tilde{\mathbf{r}}_k^T$ according to (6) and (7), respectively.
2) If $0 < K < M$, compute $\mathbf{Q}_{K+1,M}$ and $\mathbf{R}_{K+1,M}^{K+1,M}$ by performing QR decomposition on $\mathbf{A}_{K+1,M} - \mathbf{Q}_{1,K} \mathbf{R}_{K+1,M}^{1,K}$, and construct $\mathbf{R}^{K+1,M} = [\mathbf{0} \quad \mathbf{R}_{K+1,M}^{K+1,M}]$.
3) If $K = 0$, compute $\mathbf{Q}$ and $\mathbf{R}$ by performing QR decomposition on $\mathbf{A}$.

We note that the nonuniqueness of QR decomposition in the case $K < M$ has the following consequence. Given QR factors $\mathbf{Q}_1$ and $\mathbf{R}_1$ of $\mathbf{A}$, the application of the mapping $\mathcal{M}$ to $(\mathbf{Q}_1, \mathbf{R}_1)$ followed by application of the inverse mapping $\mathcal{M}^{-1}$ yields matrices $\mathbf{Q}_2$ and $\mathbf{R}_2$ that may not be equal to $\mathbf{Q}_1$ and $\mathbf{R}_1$, respectively. However, $\mathbf{Q}_2$ and $\mathbf{R}_2$ will be QR factors of $\mathbf{A}$ in the sense of Definition 1.

The following theorem, proven in [7], paves the way for the formulation of interpolation-based QR decomposition algorithms.

Theorem 8. *Given $\mathbf{A} : \mathcal{U} \to \mathbb{C}^{P \times M}$ with $P \geq M$, such that $\mathbf{A}(s) \sim (V_1, V_2)$ with maximum degree $V = V_1 + V_2$. The functions $\Delta_k(s)$, $\tilde{\mathbf{q}}_k(s)$, and $\tilde{\underline{\mathbf{r}}}_k(s)$, obtained by applying the mapping $\mathcal{M}$ as in (3)–(5) to QR factors $\mathbf{Q}(s)$ and $\mathbf{R}(s)$ of $\mathbf{A}(s)$ for all $s \in \mathcal{U}$, satisfy the following properties:*

1) $\Delta_k(s) \sim (kV, kV)$
2) $\tilde{\mathbf{q}}_k(s) \sim ((k-1)V + V_1, (k-1)V + V_2)$
3) $\tilde{\underline{\mathbf{r}}}_k(s) \sim (kV, kV)$.

We emphasize that Theorem 8 applies to any QR factors according to Definition 1 and is therefore not affected by

[1]Note that for $K < M$, the inverse mapping $\mathcal{M}^{-1}$ requires explicit knowledge of $\mathbf{A}_{K+1,M}$.

the nonuniqueness of QR decomposition arising in the rank-deficient case. Finally, we mention that Theorem 8, with the definitions of $\mathcal{M}$ and $\mathcal{M}^{-1}$ given above, carries over, in a straightforward fashion, to the case of regularized QR decomposition.

V. Application to MIMO-OFDM

We are now ready to show how the results on QR decomposition of LP matrices reviewed in the previous section lead to algorithms that exploit the polynomial nature of the MIMO channel transfer function $\mathbf{H}(s) \sim (0, L)$ to perform efficient interpolation-based computation of QR factors of $\mathbf{H}(s_n)$, for all data-carrying tones $n \in \mathcal{D}$, given knowledge of $\mathbf{H}(s_n)$ at the tones $n \in \mathcal{E}$. We note that the algorithms presented in this section are based on generic interpolation according to Definition 7. Specific interpolation methods will be discussed in Section VIII.

In the algorithms presented below, interpolation involves base points and target points on $\mathcal{U}$ that correspond to OFDM tones indexed by integers taken from the set $\{0, 1, \ldots, N-1\}$. For a given set $\mathcal{X} \subseteq \{0, 1, \ldots, N-1\}$ of OFDM tones, we define $\mathcal{S}(\mathcal{X}) \triangleq \{s_n : n \in \mathcal{X}\}$ to denote the set of corresponding points on $\mathcal{U}$. With this definition in place, we start by summarizing the brute-force approach described in Section III-C.

Algorithm I: Brute-force per-tone QR decomposition

1) Interpolate $\mathbf{H}(s)$ from $\mathcal{S}(\mathcal{E})$ to $\mathcal{S}(\mathcal{D})$.
2) For each $n \in \mathcal{D}$, perform QR decomposition on $\mathbf{H}(s_n)$ to obtain $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$.

It is obvious that for large $D = |\mathcal{D}|$, performing QR decomposition on a per-tone basis will result in high computational complexity. However, in the practically relevant case $L \ll D$ the OFDM system effectively highly oversamples the MIMO channel transfer function, so that $\mathbf{H}(s_n)$ changes slowly across n. This observation, combined with the results of [7] summarized in Section IV, constitutes the basis for a new class of algorithms that perform QR decomposition at a small number of tones and obtain the remaining QR factors through interpolation. More specifically, the basic idea of interpolation-based QR decomposition is as follows. By applying Theorem 8 to the $M_R \times M_T$ LP matrix $\mathbf{H}(s) \sim (0, L)$, we obtain $\tilde{\mathbf{q}}_k(s) \sim ((k-1)L, kL)$ and $\underline{\tilde{\mathbf{r}}}_k(s) \sim (kL, kL)$ for $k = 1, 2, \ldots, M_T$. In order to simplify the exposition, in the remainder of the paper we consider $\tilde{\mathbf{q}}_k(s)$ as satisfying $\tilde{\mathbf{q}}_k(s) \sim (kL, kL)$. The resulting statements

$$\tilde{\mathbf{q}}_k(s), \underline{\tilde{\mathbf{r}}}_k(s) \sim (kL, kL), \qquad k = 1, 2, \ldots, M_T \tag{9}$$

imply that both $\tilde{\mathbf{q}}_k(s)$ and $\underline{\tilde{\mathbf{r}}}_k(s)$ can be interpolated from at least $2kL + 1$ base points, and that, as a consequence of $V_1 = V_2 = kL$, the corresponding interpolation matrices are real-valued. For $k = 1, 2, \ldots, M_T$, the interpolation-based algorithms to be presented compute $\tilde{\mathbf{q}}_k(s_n)$ and $\underline{\tilde{\mathbf{r}}}_k(s_n)$, through QR decomposition followed by application of the mapping $\mathcal{M}$, at a subset of OFDM tones of cardinality at least $2kL + 1$, then interpolate $\tilde{\mathbf{q}}_k(s)$ and $\underline{\tilde{\mathbf{r}}}_k(s)$ to obtain $\tilde{\mathbf{q}}_k(s_n)$ and $\underline{\tilde{\mathbf{r}}}_k(s_n)$ at the remaining tones, and finally apply the inverse mapping $\mathcal{M}^{-1}$ at these tones. In the following, the sets $\mathcal{I}_k \subseteq \{0, 1, \ldots, N-1\}$, with $\mathcal{I}_{k-1} \subseteq \mathcal{I}_k$ and $B_k \triangleq |\mathcal{I}_k| \geq 2kL + 1$ ($k = 1, 2, \ldots, M_T$), contain the indices corresponding to the OFDM tones chosen as base points. For completeness, we define $\mathcal{I}_0 \triangleq \emptyset$. Specific choices of the sets $\mathcal{I}_k$ will be discussed in detail in Section VIII.

We start with a conceptually simple algorithm for interpolation-based QR decomposition, derived from the observation that the M_T statements in (9) can be unified into the single statement $\tilde{\mathbf{Q}}(s), \tilde{\mathbf{R}}(s) \sim (M_T L, M_T L)$. This implies that we can interpolate $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$ from a single set of base points of cardinality B_{M_T}. The corresponding algorithm can be formulated as follows:

Algorithm II: Single interpolation step

1) Interpolate $\mathbf{H}(s)$ from $\mathcal{S}(\mathcal{E})$ to $\mathcal{S}(\mathcal{I}_{M_T})$.
2) For each $n \in \mathcal{I}_{M_T}$, perform QR decomposition on $\mathbf{H}(s_n)$ to obtain $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$.
3) For each $n \in \mathcal{I}_{M_T}$, apply $\mathcal{M} : (\mathbf{Q}(s_n), \mathbf{R}(s_n)) \mapsto (\tilde{\mathbf{Q}}(s_n), \tilde{\mathbf{R}}(s_n))$.
4) Interpolate $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$ from $\mathcal{S}(\mathcal{I}_{M_T})$ to $\mathcal{S}(\mathcal{D} \backslash \mathcal{I}_{M_T})$.
5) For each $n \in \mathcal{D} \backslash \mathcal{I}_{M_T}$, apply $\mathcal{M}^{-1} : (\tilde{\mathbf{Q}}(s_n), \tilde{\mathbf{R}}(s_n)) \mapsto (\mathbf{Q}(s_n), \mathbf{R}(s_n))$.

This formulation of Algorithm II assumes that $\mathbf{H}(s_n)$ has full rank for all $n \in \mathcal{D} \backslash \mathcal{I}_{M_T}$, which allows to perform all inverse mappings $\mathcal{M}^{-1}$ in Step 5 using (6)–(8) only. If, however, for a given $n \in \mathcal{D} \backslash \mathcal{I}_{M_T}$, $\mathbf{H}(s_n)$ is rank-deficient with ordered column rank $K < M_T$, we have $\tilde{\mathbf{Q}}_{K+1,M_T}(s_n) = \mathbf{0}$ and $\tilde{\mathbf{R}}^{K+1,M_T}(s_n) = \mathbf{0}$. Hence, according to the definition of $\mathcal{M}^{-1}$ in Section IV, $\mathbf{Q}_{K+1,M_T}(s_n)$ and $\mathbf{R}^{K+1,M_T}(s_n)$ must be computed through QR decomposition of $\mathbf{H}_{K+1,M_T}(s_n) - \mathbf{Q}_{1,K}(s_n)\mathbf{R}^{1,K}_{K+1,M_T}(s_n)$ for $K > 0$ or of $\mathbf{H}(s_n)$ for $K = 0$. This, in turn, requires $\mathbf{H}_{K+1,M_T}(s_n)$ to be obtained by interpolating $\mathbf{H}_{K+1,M_T}(s)$ from $\mathcal{S}(\mathcal{E})$ to the single target point s_n in an additional step. For simplicity of exposition, in the remainder of the paper we will assume that $\mathbf{H}(s_n)$ has full rank for all $n \in \mathcal{D}$.

Departing from Algorithm II, which interpolates $\tilde{\mathbf{q}}_k(s)$ and $\underline{\tilde{\mathbf{r}}}_k(s)$ from B_{M_T} base points, we next present a more sophisticated algorithm that involves interpolation of $\tilde{\mathbf{q}}_k(s)$ and $\underline{\tilde{\mathbf{r}}}_k(s)$ from $B_k \leq B_{M_T}$ base points ($k = 1, 2, \ldots, M_T$), in agreement with (9). The resulting Algorithm III consists of M_T iterations. In the first iteration, the tones $n \in \mathcal{I}_1$ are considered. At each of these tones, QR decomposition is performed on $\mathbf{H}(s_n)$, resulting in $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$, which are then mapped to $(\tilde{\mathbf{Q}}(s_n), \tilde{\mathbf{R}}(s_n))$ by applying $\mathcal{M}$. Next, $\tilde{\mathbf{q}}_1(s)$ and $\underline{\tilde{\mathbf{r}}}_1(s)$ are interpolated from the tones $n \in \mathcal{I}_1$ to the remaining tones $n \in \mathcal{D} \backslash \mathcal{I}_1$. In the kth iteration ($k = 2, 3, \ldots, M_T$), the tones $n \in \mathcal{I}_k \backslash \mathcal{I}_{k-1}$ are considered. At each of these tones, $\mathbf{Q}_{1,k-1}(s_n)$ and $\mathbf{R}^{1,k-1}(s_n)$ are obtained[2] by applying $\mathcal{M}^{-1}$ to $(\tilde{\mathbf{Q}}_{1,k-1}(s_n), \tilde{\mathbf{R}}^{1,k-1}(s_n))$, already known from the previous iterations, whereas the submatrices $\mathbf{Q}_{k,M_T}(s_n)$ and $\mathbf{R}^{k,M_T}_{k,M_T}(s_n)$ are obtained by

[2]The mapping $\mathcal{M}$ and its inverse $\mathcal{M}^{-1}$ are defined on submatrices of $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$ according to (3)–(8).

performing QR decomposition on the matrix $\mathbf{H}_{k,M_T}(s_n) - \mathbf{Q}_{1,k-1}(s_n)\mathbf{R}_{k,M_T}^{1,k-1}(s_n)$, in accordance with Proposition 4, and $\mathbf{R}^{k,M_T}(s_n)$ is given, for $k > 1$, by $[\,\mathbf{0} \quad \mathbf{R}_{k,M_T}^{k,M_T}(s_n)\,]$. Next, the submatrices $\tilde{\mathbf{Q}}_{k,M_T}(s_n)$ and $\tilde{\mathbf{R}}^{k,M_T}(s_n)$ are computed by applying $\mathcal{M}$ to $(\mathbf{Q}_{k,M_T}(s_n), \mathbf{R}^{k,M_T}(s_n))$. Since the samples $\tilde{\mathbf{q}}_k(s_n)$ and $\underline{\tilde{\mathbf{r}}}_k(s_n)$ are now known at all tones $n \in \mathcal{I}_k$, $\tilde{\mathbf{q}}_k(s)$ and $\underline{\tilde{\mathbf{r}}}_k(s)$ can be interpolated from the tones $n \in \mathcal{I}_k$ to the remaining tones $n \in \mathcal{D}\backslash\mathcal{I}_k$, thereby completing the kth iteration. After M_T iterations, we know $\tilde{\mathbf{Q}}(s_n)$ and $\tilde{\mathbf{R}}(s_n)$ at all tones $n \in \mathcal{D}$, as well as $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$ at the tones $n \in \mathcal{I}_{M_T}$. The last step consists of applying $\mathcal{M}^{-1}$ to $(\tilde{\mathbf{Q}}(s_n), \tilde{\mathbf{R}}(s_n))$ to obtain $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$ at the remaining tones $n \in \mathcal{D}\backslash\mathcal{I}_{M_T}$. The algorithm is formulated as follows:

Algorithm III: Multiple interpolation steps

1) Set $k \leftarrow 1$.
2) Interpolate $\mathbf{H}_{k,M_T}(s)$ from $\mathcal{S}(\mathcal{E})$ to $\mathcal{S}(\mathcal{I}_k\backslash\mathcal{I}_{k-1})$.
3) If $k = 1$, go to Step 5. Otherwise, for each $n \in \mathcal{I}_k\backslash\mathcal{I}_{k-1}$, apply $\mathcal{M}^{-1} : (\tilde{\mathbf{Q}}_{1,k-1}(s_n), \tilde{\mathbf{R}}^{1,k-1}(s_n)) \mapsto (\mathbf{Q}_{1,k-1}(s_n), \mathbf{R}^{1,k-1}(s_n))$.
4) For each $n \in \mathcal{I}_k\backslash\mathcal{I}_{k-1}$, overwrite $\mathbf{H}_{k,M_T}(s_n)$ by $\mathbf{H}_{k,M_T}(s_n) - \mathbf{Q}_{1,k-1}(s_n)\mathbf{R}_{k,M_T}^{1,k-1}(s_n)$.
5) For each $n \in \mathcal{I}_k\backslash\mathcal{I}_{k-1}$, perform QR decomposition on $\mathbf{H}_{k,M_T}(s_n)$ to obtain $\mathbf{Q}_{k,M_T}(s_n)$ and $\mathbf{R}_{k,M_T}^{k,M_T}(s_n)$, and, if $k > 1$, construct $\mathbf{R}^{k,M_T}(s_n) = [\,\mathbf{0} \quad \mathbf{R}_{k,M_T}^{k,M_T}(s_n)\,]$.
6) For each $n \in \mathcal{I}_k\backslash\mathcal{I}_{k-1}$, apply $\mathcal{M} : (\mathbf{Q}_{k,M_T}(s_n), \mathbf{R}^{k,M_T}(s_n)) \mapsto (\tilde{\mathbf{Q}}_{k,M_T}(s_n), \tilde{\mathbf{R}}^{k,M_T}(s_n))$.
7) Interpolate $\tilde{\mathbf{q}}_k(s)$ and $\underline{\tilde{\mathbf{r}}}_k(s)$ from $\mathcal{S}(\mathcal{I}_k)$ to $\mathcal{S}(\mathcal{D}\backslash\mathcal{I}_k)$.
8) If $k = M_T$, proceed to the next step. Otherwise, set $k \leftarrow k + 1$ and go back to Step 2.
9) For each $n \in \mathcal{D}\backslash\mathcal{I}_{M_T}$, apply $\mathcal{M}^{-1} : (\tilde{\mathbf{Q}}(s_n), \tilde{\mathbf{R}}(s_n)) \mapsto (\mathbf{Q}(s_n), \mathbf{R}(s_n))$.

In comparison to Algorithm II, Algorithm III performs QR decompositions on increasingly smaller matrices. The corresponding computational complexity savings are, however, traded against an increase in interpolation effort and the computational overhead associated with Step 4, which will be referred to as the *reduction step* in what follows. Moreover, the complexity of applying $\mathcal{M}$ and $\mathcal{M}^{-1}$ differs for the two algorithms. A detailed complexity analysis provided in the next section will show that, depending on the system parameters, Algorithm III can exhibit smaller complexity than Algorithm II.

So far, we assumed that the matrices $\mathbf{H}(s_n), n \in \mathcal{E}$, are known perfectly. Under this assumption, Algorithms I–III produce the same result, namely QR factors $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$ of $\mathbf{H}(s_n), n \in \mathcal{D}$. If the matrices $\mathbf{H}(s_n), n \in \mathcal{E}$, are perturbed by additive noise due to, e.g., channel estimation errors, interpolation of $\mathbf{H}(s)$ (as performed in Step 1 of Algorithms I and II, or in Step 2 of Algorithm III) will produce samples of an LP matrix $\hat{\mathbf{H}}(s) \sim (0, L)$ that is not equal to $\mathbf{H}(s)$ but is the same in all three algorithms (cf. the discussion at the end of Section II-C in the context of a generic LP $a(s)$). Consequently, Algorithms I–III will yield QR factors of the matrices $\hat{\mathbf{H}}(s_n), n \in \mathcal{D}$, and are hence all equally affected by channel estimation errors. The results presented in the remainder of the paper are based on the assumption of perfect knowledge of $\mathbf{H}(s_n), n \in \mathcal{E}$.

We note that the conditions $|\mathcal{E}| \geq L + 1$ and $B_k \geq 2kL + 1$, $k = 1, 2, \ldots, M_T$, guarantee that all instances of interpolation in Algorithms I–III can be carried out exactly. In Section IX-A, we will argue that savings in the complexity of the interpolation-based algorithms can be obtained by performing, instead, inexact interpolation from a small number of base points. Although this leads to errors in the algorithm output $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$, $n \in \mathcal{D}$, the numerical results in Section IX-A demonstrate that large complexity savings can be obtained without significantly degrading MIMO-OFDM detection performance. Throughout Sections VI–VIII, we assume exact interpolation unless specified otherwise.

VI. Complexity Analysis

We are next interested in assessing under which circumstances the interpolation-based Algorithms II and III offer computational complexity savings over the brute-force approach in Algorithm I. To this end, we propose a simple computational complexity metric, representative of VLSI circuit complexity as quantified by the product of chip area and processing delay [22]. We note that other important aspects of VLSI design, including, e.g., wordwidth requirements, memory access strategies, and datapath architecture, are not accounted for in our analysis. Nevertheless, the proposed metric is indicative of the complexity of Algorithms I–III and allows to quantify the impact of the system parameters on the potential savings of interpolation-based QR decomposition over brute-force per-tone QR decomposition. In the remainder of the paper, unless explicitly stated otherwise, the term *complexity* refers to computational complexity according to the metric defined in Section VI-A below.

A. Complexity Metric

In the VLSI implementation of a given algorithm, a wide range of trade-offs between silicon area A and processing delay τ can, in general, be realized [22]. Parallel processing reduces τ at the expense of a larger A, whereas resource sharing reduces A at the expense of a larger τ. However, the corresponding circuit transformations typically do not affect the area-delay product $A\tau$ significantly. For this reason, the area-delay product is considered a relevant indicator of algorithm complexity [22]. In the definition of the specific complexity metric that will be used subsequently, we only take into account the arithmetic operations with a significant impact on $A\tau$. More specifically, we divide the operations underlying the algorithms under consideration into three classes, namely i) multiplications, ii) divisions and square roots, and iii) additions and subtractions. Class iii) operations will not be counted as they typically have a significantly lower VLSI circuit complexity than Class i) and Class ii) operations.

In all algorithms presented in this paper, the number of Class i) operations is significantly larger than the number of

Class ii) operations.[3] Moreover, the data dependences in these algorithms allow Class i) operations and Class ii) operations to be performed in parallel. In an architecture where Class i) operations are carried out sequentially, a single Class ii) operation can be distributed over the time required by multiple Class i) operations, as demonstrated in [23]. Therefore, the arithmetical unit in charge of the Class ii) operation can be designed to consume a significantly smaller silicon area than the multiplier carrying out the Class i) operations. Based on this observation, we conclude that the impact of Class ii) operation on the overall complexity can be neglected.

Within Class i), we distinguish between *full multiplications* (i.e., multiplications of two variable operands) and *constant multiplications* (i.e., multiplications of a variable operand by a constant operand[4]). We define the cost of a full multiplication as the unit of computational complexity. We do not distinguish between real-valued full multiplications and complex-valued full multiplications, as we assume that both are performed by multipliers designed to process two variable complex-valued operands. The fact, discussed in detail in Section VIII-A, that a constant multiplication can be implemented in VLSI at significantly smaller cost than a full multiplication, will be accounted for through a weighting factor smaller than one.

B. Per-Tone Complexity of Individual Computational Tasks

In order to simplify the notation, in the remainder of this section we drop the dependence of all quantities on s_n. We furthermore introduce the auxiliary variable

$$J_k \triangleq M_R k + M_T k - \frac{(k-1)k}{2}, \qquad k = 1, 2, \ldots, M_T$$

which specifies the maximum total number of nonzero entries in $\mathbf{Q}_{1,k}$ and $\mathbf{R}^{1,k}$, and, hence, also in $\tilde{\mathbf{Q}}_{1,k}$ and $\tilde{\mathbf{R}}^{1,k}$, in accordance with the fact that $\mathbf{R}$ and $\tilde{\mathbf{R}}$ are upper triangular.

Interpolation: We quantify the complexity of interpolating an LP to one target point through an equivalent of c_{IP} full multiplications. The dependence of interpolation complexity on the underlying VLSI implementation and on the number of base points is assumed to be incorporated into c_{IP}. Specific strategies for efficient interpolation along with the corresponding values of c_{IP} are presented in Section VIII. Since interpolation of an LP matrix is performed entrywise, the complexity of interpolating $\mathbf{H}_{k,M_T}(s)$ to one target point is given by $c_{\text{IP},\mathbf{H}}^{k,M_T} \triangleq M_R(M_T-k+1)c_{\text{IP}}$ $(k = 1, 2, \ldots, M_T)$. Similarly, interpolation of $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$ to one target point has complexity $c_{\text{IP},\tilde{\mathbf{Q}}\tilde{\mathbf{R}}} \triangleq J_{M_T} c_{\text{IP}}$ and the complexity of interpolating $\tilde{\mathbf{q}}_k(s)$ and $\underline{\tilde{\mathbf{r}}}_k(s)$ to one target point is given by $c_{\text{IP},\tilde{\mathbf{q}}\tilde{\mathbf{r}}}^{(k)} \triangleq (M_R + M_T - k + 1)c_{\text{IP}}$ $(k = 1, 2, \ldots, M_T)$.

QR decomposition: To keep our discussion independent of the QR decomposition method used, we denote the cost of performing QR decomposition on an $M_R \times k$ matrix by $c_{\text{QR}}^{M_R \times k}$ $(k = 1, 2, \ldots, M_T)$. Specific expressions for $c_{\text{QR}}^{M_R \times k}$ will only be required in the numerical complexity analysis in Section IX.

Mapping $\mathcal{M}$: We denote the overall cost of mapping $(\mathbf{Q}_{k,M_T}, \mathbf{R}^{k,M_T})$ to $(\tilde{\mathbf{Q}}_{k,M_T}, \tilde{\mathbf{R}}^{k,M_T})$ $(k = 1, 2, \ldots, M_T)$ by $c_{\mathcal{M}}^{k,M_T}$. In the case $k = 1$, application of the mapping $\mathcal{M}$ requires computation of $[\mathbf{R}]_{1,1}$, $[\mathbf{R}]_{1,1}^2$, $[\mathbf{R}]_{1,1}^2[\mathbf{R}]_{2,2}$, $[\mathbf{R}]_{1,1}^2[\mathbf{R}]_{2,2}^2, \ldots, \prod_{i=1}^{M_T}[\mathbf{R}]_{i,i}^2$, at the cost of $2M_T - 1$ full multiplications. This step yields both the scaling factors $\Delta_{k'-1}[\mathbf{R}]_{k',k'}$, $k' = 1, 2, \ldots, M_T$, and the diagonal entries of $\tilde{\mathbf{R}}$. Now, the first column of $\tilde{\mathbf{Q}}$ is equal to the first column of $\mathbf{H}$ and is hence obtained at zero complexity. The remaining entries of $\tilde{\mathbf{Q}}$ and the entries of $\tilde{\mathbf{R}}$ above its main diagonal are obtained by scaling the corresponding entries of $\mathbf{Q}$ and $\mathbf{R}$ according to (4) and (5), respectively, which requires $J_{M_T} - M_R - M_T$ full multiplications. Hence, we obtain $c_{\mathcal{M}}^{1,M_T} = J_{M_T} - M_R + M_T - 1$. Next, we consider the case $k > 1$, which only occurs in Step 3 of Algorithm III, where $\Delta_{k-1} = [\tilde{\mathbf{R}}]_{k-1,k-1}$ is already available from the previous iteration which involves interpolation of $\underline{\tilde{\mathbf{r}}}_{k-1}(s)$. The application of the mapping $\mathcal{M}$ first requires computation of $\Delta_{k-1}[\mathbf{R}]_{k,k}$, $\Delta_{k-1}[\mathbf{R}]_{k,k}^2$, $\Delta_{k-1}[\mathbf{R}]_{k,k}^2[\mathbf{R}]_{k+1,k+1}, \ldots,$ $\Delta_{k-1}\prod_{i=k}^{M_T}[\mathbf{R}]_{i,i}^2$, at the cost of $2(M_T - k + 1)$ full multiplications. Then, the entries of $\mathbf{Q}_{k,M_T}$ and the entries of $\mathbf{R}^{k,M_T}$ above the main diagonal of $\mathbf{R}$ are scaled according to (4) and (5), which requires $J_{M_T} - J_{k-1} - (M_T - k + 1)$ full multiplications. In summary, we obtain $c_{\mathcal{M}}^{k,M_T} = J_{M_T} - J_{k-1} + M_T - k + 1$ for $k = 2, 3, \ldots, M_T$.

Inverse mapping $\mathcal{M}^{-1}$: We denote the overall cost of mapping $(\tilde{\mathbf{Q}}_{1,k}, \tilde{\mathbf{R}}^{1,k})$ to $(\mathbf{Q}_{1,k}, \mathbf{R}^{1,k})$ $(k = 1, 2, \ldots, M_T)$ by $c_{\mathcal{M}^{-1}}^{1,k}$. Since $\Delta_0 = 1$ and $[\tilde{\mathbf{R}}]_{1,1} = [\mathbf{R}]_{1,1}^2$, by first computing $([\tilde{\mathbf{R}}]_{1,1})^{1/2}$ and then its inverse, we can obtain both $[\mathbf{R}]_{1,1}$ and the scaling factor $(\Delta_0[\mathbf{R}]_{1,1})^{-1} = 1/[\mathbf{R}]_{1,1}$ at the cost of one square root operation and one division. For $k' = 2, 3, \ldots, k$, the scaling factors $(\Delta_{k'-1}[\mathbf{R}]_{k',k'})^{-1}$ can be obtained according to (8) by computing $([\tilde{\mathbf{R}}]_{k'-1,k'-1}[\tilde{\mathbf{R}}]_{k',k'})^{-1/2}$, at the cost of $k-1$ full multiplications, $k-1$ square root operations, and $k-1$ divisions. The entries of $\mathbf{Q}_{1,k}$ and the remaining entries of $\mathbf{R}^{1,k}$ on and above the main diagonal of $\mathbf{R}$ are obtained by scaling the corresponding entries of $\tilde{\mathbf{Q}}_{1,k}$ and $\tilde{\mathbf{R}}^{1,k}$ according to (6) and (7), respectively, at the cost of $J_k - 1$ full multiplications. Since we neglect the impact of square root operations and divisions on complexity, we obtain $c_{\mathcal{M}^{-1}}^{1,k} = J_k + k - 2$ for $k = 1, 2, \ldots, M_T$.

Reduction step: Since matrix subtraction has negligible complexity, for a given $k \in \{2, 3, \ldots, M_T\}$, the complexity associated with the computation of $\mathbf{H}_{k,M_T} - \mathbf{Q}_{1,k-1}\mathbf{R}_{k,M_T}^{1,k-1}$, denoted by $c_{\text{red}}^{(k)}$, is given by the complexity associated with the multiplication of the $M_R \times (k-1)$ matrix $\mathbf{Q}_{1,k-1}$ by the $(k-1) \times (M_T - k + 1)$ matrix $\mathbf{R}_{k,M_T}^{1,k-1}$. Hence, we obtain $c_{\text{red}}^{(k)} = M_R(k-1)(M_T - k + 1)$ for $k = 2, 3, \ldots, M_T$.

[3] We assume that division of an M-dimensional vector $\mathbf{a}$ by a scalar α, such as the divisions in (6) or (7), is implemented by first computing the single division $\beta \triangleq 1/\alpha$ and then multiplying the M entries of $\mathbf{a}$ by β, at the cost of one Class ii) operation and M Class i) operations, respectively.

[4] In the context of the interpolation-based algorithms considered in this paper, all operands that depend on $\mathbf{H}(s)$ are assumed to be variable. The coefficients of interpolation filters, e.g., are treated as constant operands. For a detailed discussion on the difference between full multiplications and constant multiplications, we refer to Section VIII-A.

Table I
TOTAL COMPLEXITY ASSOCIATED WITH THE INDIVIDUAL COMPUTATIONAL TASKS

Computational task	Symbol [a]	Algorithm I	Algorithm II	Algorithm III
Interpolation of $\mathbf{H}(s)$	$c_{\text{IP},\mathbf{H},\text{A}}$	$Dc_{\text{IP},\mathbf{H}}^{1,M_T}$	$B_{M_T}c_{\text{IP},\mathbf{H}}^{1,M_T}$	$B_1c_{\text{IP},\mathbf{H}}^{1,M_T}+2L\sum_{k=2}^{M_T}c_{\text{IP},\mathbf{H}}^{k,M_T}$
Interpolation of $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$	$c_{\text{IP},\tilde{\mathbf{Q}}\tilde{\mathbf{R}},\text{A}}$	0	$(D-B_{M_T})c_{\text{IP},\tilde{\mathbf{Q}}\tilde{\mathbf{R}}}$	$\sum_{k=1}^{M_T}\big(D-B_k\big)c_{\text{IP},\tilde{\mathbf{q}}\tilde{\mathbf{r}}}^{(k)}$
QR decomposition	$c_{\text{QR,A}}$	$Dc_{\text{QR}}^{M_R\times M_T}$	$B_{M_T}c_{\text{QR}}^{M_R\times M_T}$	$B_1c_{\text{QR}}^{M_R\times M_T}+2L\sum_{k=2}^{M_T}c_{\text{QR}}^{M_R\times(M_T-k+1)}$
Mapping $\mathcal{M}$	$c_{\mathcal{M},\text{A}}$	0	$B_{M_T}c_{\mathcal{M}}^{1,M_T}$	$B_1c_{\mathcal{M}}^{1,M_T}+2L\sum_{k=2}^{M_T}c_{\mathcal{M}}^{k,M_T}$
Inverse mapping $\mathcal{M}^{-1}$	$c_{\mathcal{M}^{-1},\text{A}}$	0	$(D-B_{M_T})c_{\mathcal{M}^{-1}}^{1,M_T}$	$2L\sum_{k=2}^{M_T}c_{\mathcal{M}^{-1}}^{1,k-1}+\big(D-B_{M_T}\big)c_{\mathcal{M}^{-1}}^{1,M_T}$
Reduction	$c_{\text{red,A}}$	0	0	$2L\sum_{k=2}^{M_T}c_{\text{red}}^{(k)}$

[a] The index A is a placeholder for the algorithm number (I, II, or III).

C. Total Complexity of Algorithms I–III

The contribution of a given computational task to the overall complexity of a given algorithm is obtained by multiplying the corresponding per-tone complexity, computed in the previous section, by the number of relevant tones. For simplicity of exposition, in the ensuing analysis we restrict ourselves to the case where the sets of OFDM tones used as base points satisfy $\mathcal{I}_1\subseteq\mathcal{I}_2\subseteq\ldots\subseteq\mathcal{I}_{M_T}\subset\mathcal{D}$ and $|\mathcal{I}_k|=B_k=2kL+1$ ($k=1,2,\ldots,M_T$), and consequently also $|\mathcal{I}_k\backslash\mathcal{I}_{k-1}|=2L$ and $|\mathcal{D}\backslash\mathcal{I}_k|=D-2kL-1$ ($k=1,2,\ldots,M_T$). With the total complexity of the individual tasks summarized in Table I, the complexity associated with Algorithms I–III is obtained as

$$C_{\text{I}}=c_{\text{IP},\mathbf{H},\text{I}}+c_{\text{QR,I}}$$
$$C_{\text{II}}=c_{\text{IP},\mathbf{H},\text{II}}+c_{\text{IP},\tilde{\mathbf{Q}}\tilde{\mathbf{R}},\text{II}}+c_{\text{QR,II}}+c_{\mathcal{M},\text{II}}+c_{\mathcal{M}^{-1},\text{II}} \quad (10)$$
$$C_{\text{III}}=c_{\text{IP},\mathbf{H},\text{III}}+c_{\text{IP},\tilde{\mathbf{Q}}\tilde{\mathbf{R}},\text{III}}+c_{\text{QR,III}}+c_{\mathcal{M},\text{III}}+c_{\mathcal{M}^{-1},\text{III}}+c_{\text{red,III}}. \quad (11)$$

D. Complexity Comparisons

In the following, we identify conditions on the system parameters and on the interpolation cost c_{IP} that guarantee that Algorithms II and III exhibit smaller complexity than Algorithm I. We start by comparing Algorithms I and II and note that

$$C_{\text{I}}-C_{\text{II}}=\Big(D-B_{M_T}\Big)\Big(c_{\text{QR}}^{M_R\times M_T}-c_{\mathcal{M}^{-1}}^{1,M_T}-\frac{M_T\left(M_T+1\right)}{2}c_{\text{IP}}\Big)-B_{M_T}c_{\mathcal{M}}^{1,M_T}. \quad (12)$$

Hence, if c_{IP} satisfies

$$c_{\text{IP}}<c_{\text{IP,max,II}}\triangleq\frac{2\big(c_{\text{QR}}^{M_R\times M_T}-c_{\mathcal{M}^{-1}}^{1,M_T}\big)}{M_T\left(M_T+1\right)} \quad (13)$$

then there exists a $D_{\min}$ such that $C_{\text{II}}<C_{\text{I}}$ for $D\geq D_{\min}$, i.e., Algorithm II exhibits a lower complexity than Algorithm I for a sufficiently high number of data-carrying tones D. Moreover, for $c_{\text{IP}}<c_{\text{IP,max,II}}$, increasing B_{M_T} reduces $C_{\text{I}}-C_{\text{II}}$. If the inequality (13) is met, as a consequence of $B_{M_T}=2M_TL+1$, (12) implies that for increasing L and with all other parameters fixed, Algorithm II exhibits smaller savings. For larger $c_{\text{QR}}^{M_R\times M_T}$, again with all other parameters fixed, Algorithm II exhibits larger savings.

In order to compare Algorithms II and III, we start from (10) and (11) and rewrite $C_{\text{II}}-C_{\text{III}}$ as

$$C_{\text{II}}-C_{\text{III}}=\Delta c_{\text{QR}}+\Delta c_{\mathcal{M},\mathcal{M}^{-1}}+\Delta c_{\text{IP},\mathbf{H}\tilde{\mathbf{Q}}\tilde{\mathbf{R}}}-c_{\text{red,III}} \quad (14)$$

where we have introduced the auxiliary variables $\Delta c_{\text{QR}}\triangleq c_{\text{QR,II}}-c_{\text{QR,III}}$, $\Delta c_{\mathcal{M},\mathcal{M}^{-1}}\triangleq c_{\mathcal{M},\text{II}}+c_{\mathcal{M}^{-1},\text{II}}-c_{\mathcal{M},\text{III}}-c_{\mathcal{M}^{-1},\text{III}}$, and $\Delta c_{\text{IP},\mathbf{H}\tilde{\mathbf{Q}}\tilde{\mathbf{R}}}\triangleq c_{\text{IP},\mathbf{H},\text{II}}+c_{\text{IP},\tilde{\mathbf{Q}}\tilde{\mathbf{R}},\text{II}}-c_{\text{IP},\mathbf{H},\text{III}}-c_{\text{IP},\tilde{\mathbf{Q}}\tilde{\mathbf{R}},\text{III}}$. From the results in Table I we get

$$\Delta c_{\text{QR}}=2L\sum_{k=2}^{M_T}\Big(c_{\text{QR}}^{M_R\times M_T}-c_{\text{QR}}^{M_R\times(M_T-k+1)}\Big) \quad (15)$$

which is strictly positive since, obviously, $c_{\text{QR}}^{M_R\times M_T}>c_{\text{QR}}^{M_R\times(M_T-k+1)}$ ($k=2,3,\ldots,M_T$). Furthermore, again employing the results in Table I, straightforward calculations yield

$$\Delta c_{\text{IP},\mathbf{H}\tilde{\mathbf{Q}}\tilde{\mathbf{R}}}=-2L\sum_{k=2}^{M_T}k(k-1)c_{\text{IP}}=-\frac{2}{3}LM_T\big(M_T^2-1\big)c_{\text{IP}} \quad (16)$$

and

$$\Delta c_{\mathcal{M},\mathcal{M}^{-1}}=\big(B_1-B_{M_T}\big)\big(M_R-1\big)=-2L\big(M_R-1\big)\big(M_T-1\big). \quad (17)$$

We observe that (14)–(17), along with the expression for $c_{\text{red,III}}$ in Table I, imply that $C_{\text{II}}-C_{\text{III}}$ does not depend on D and is proportional to L. Moreover, it follows from (14) and (16) that $C_{\text{III}}<C_{\text{II}}$ is equivalent to $c_{\text{IP}}<c_{\text{IP,max,III}}$ with

$$c_{\text{IP,max,III}}\triangleq\frac{\Delta c_{\text{QR}}+\Delta c_{\mathcal{M},\mathcal{M}^{-1}}-c_{\text{red,III}}}{\frac{2}{3}LM_T(M_T^2-1)}. \quad (18)$$

We note that the right-hand side (RHS) of (18) depends solely on M_T and M_R, since Δc_{QR}, $\Delta c_{\mathcal{M},\mathcal{M}^{-1}}$, and $c_{\text{red,III}}$ are proportional to L. Hence, if $\Delta c_{\text{QR}}+\Delta c_{\mathcal{M},\mathcal{M}^{-1}}-c_{\text{red,III}} > 0$ and for c_{IP} sufficiently small, Algorithm III has lower complexity than Algorithm II.

We conclude this section with a comment on the memory requirements of Algorithms I and II. If interpolation of $\bar{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$ in Step 4 of Algorithm II is implemented such that the samples of $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$ at the B_{M_T} base points are overwritten by samples of $\bar{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$ at B_{M_T} out of the $D-B_{M_T}$ target points, Algorithms I and II exhibit comparable memory requirements. We can therefore conclude that the complexity savings of interpolation-based QR decomposition over brute-force per-tone QR decomposition need not come at the cost of increased memory requirements.

VII. The MMSE Case

In this section, we modify the QR decomposition algorithms described in Section V to obtain corresponding algorithms that compute the MMSE-QR decomposition of the channel matrices $\mathbf{H}(s_n), n \in \mathcal{D}$.

We recall from Section IV that the definitions of $\mathcal{M}$ and $\mathcal{M}^{-1}$, as well as the statement of Theorem 8, carry over to regularized QR decomposition. Moreover, MMSE-QR decomposition was defined in Section III-B to be a special case of regularized QR decomposition. With this insight, the modification of Algorithms I and II to the MMSE case is straightforward and simply amounts to replacing, in Step 2 of both algorithms, QR decomposition by MMSE-QR decomposition. The resulting algorithms are referred to as Algorithm I-MMSE and Algorithm II-MMSE, respectively.

We next discuss the extension of Algorithm III to the MMSE case. As a starting point, we consider the straightforward approach of applying Algorithm III to the *MMSE-augmented channel matrix* $\bar{\mathbf{H}}(s_n) \triangleq [\,\mathbf{H}^T(s_n) \quad \sqrt{M_T}\sigma_w\mathbf{I}_{M_T}\,]^T$ to produce $\bar{\mathbf{Q}}(s_n)$ and $\mathbf{R}(s_n)$ for all $n \in \mathcal{D}$. In the following, we denote by $\tilde{\bar{\mathbf{Q}}}(s_n)$ and $\tilde{\mathbf{R}}(s_n)$ the matrices resulting from the application of the mapping $\mathcal{M}$ to $(\bar{\mathbf{Q}}(s_n), \mathbf{R}(s_n))$. We observe that the straightforward approach under consideration is inefficient, since we are only interested in obtaining $\mathbf{Q}(s_n) = \bar{\mathbf{Q}}^{1,M_R}(s_n)$ and $\mathbf{R}(s_n)$ for all $n \in \mathcal{D}$. Consequently, we would like to avoid computing the last M_T rows of $\bar{\mathbf{Q}}(s_n)$ at as many tones as possible. Now, the reduction step (i.e., Step 4) in the kth iteration of Algorithm III requires knowledge of $\bar{\mathbf{Q}}_{1,k-1}(s_n)$ at the tones $n \in \mathcal{I}_k\backslash\mathcal{I}_{k-1}$ $(k = 2, 3, \ldots, M_T)$. Hence, at the tones $n \in \mathcal{I}_k\backslash\mathcal{I}_{k-1}$ we must compute all $M_R + M_T$ rows of $\bar{\mathbf{Q}}_{1,k-1}(s_n)$ anyway. In contrast, at the tones $n \in \mathcal{D}\backslash\mathcal{I}_{M_T}$ we can restrict interpolation and inverse mapping to $\tilde{\mathbf{Q}}(s_n) = \tilde{\bar{\mathbf{Q}}}^{1,M_R}(s_n)$ and $\tilde{\mathbf{R}}(s_n)$. With the definitions $\tilde{\mathbf{q}}_k(s_n) \triangleq \tilde{\bar{\mathbf{Q}}}^{1,M_R}_{k,k}(s_n)$ and $\check{\mathbf{q}}_k(s_n) \triangleq \tilde{\bar{\mathbf{Q}}}^{M_R+1,M_R+M_T}_{k,k}(s_n)$, $k = 1, 2, \ldots, M_T$, the resulting algorithm can be formulated as follows:

Algorithm III-MMSE

1) Set $k \leftarrow 1$.
2) Interpolate $\mathbf{H}_{k,M_T}(s)$ from $\mathcal{S}(\mathcal{E})$ to $\mathcal{S}(\mathcal{I}_k\backslash\mathcal{I}_{k-1})$.
3) For each $n \in \mathcal{I}_k\backslash\mathcal{I}_{k-1}$, construct $\bar{\mathbf{H}}_{k,M_T}(s_n) = ([\,\mathbf{H}^T(s_n) \quad \sqrt{M_T}\sigma_w\mathbf{I}_{M_T}\,]^T)_{k,M_T}$.
4) If $k = 1$, go to Step 6. Otherwise, for each $n \in \mathcal{I}_k\backslash\mathcal{I}_{k-1}$, apply $\mathcal{M}^{-1} : (\tilde{\bar{\mathbf{Q}}}_{1,k-1}(s_n), \tilde{\mathbf{R}}^{1,k-1}(s_n)) \mapsto (\bar{\mathbf{Q}}_{1,k-1}(s_n), \mathbf{R}^{1,k-1}(s_n))$.
5) For each $n \in \mathcal{I}_k\backslash\mathcal{I}_{k-1}$, overwrite $\bar{\mathbf{H}}_{k,M_T}(s_n)$ by $\bar{\mathbf{H}}_{k,M_T}(s_n) - \bar{\mathbf{Q}}_{1,k-1}(s_n)\mathbf{R}^{1,k-1}_{k,M_T}(s_n)$.
6) For each $n \in \mathcal{I}_k\backslash\mathcal{I}_{k-1}$, perform QR decomposition on $\bar{\mathbf{H}}_{k,M_T}(s_n)$ to obtain $\bar{\mathbf{Q}}_{k,M_T}(s_n)$ and $\mathbf{R}^{k,M_T}_{k,M_T}(s_n)$, and, if $k > 1$, construct $\mathbf{R}^{k,M_T}(s_n) = [\,\mathbf{0} \quad \mathbf{R}^{k,M_T}_{k,M_T}(s_n)\,]$.
7) For each $n \in \mathcal{I}_k\backslash\mathcal{I}_{k-1}$, apply $\mathcal{M} : (\bar{\mathbf{Q}}_{k,M_T}(s_n), \mathbf{R}^{k,M_T}(s_n)) \mapsto (\tilde{\bar{\mathbf{Q}}}_{k,M_T}(s_n), \tilde{\mathbf{R}}^{k,M_T}(s_n))$.
8) Interpolate $\tilde{\mathbf{q}}_k(s)$ and $\tilde{\underline{\mathbf{r}}}_k(s)$ from $\mathcal{S}(\mathcal{I}_k)$ to $\mathcal{S}(\mathcal{D}\backslash\mathcal{I}_k)$.
9) If $k = M_T$, proceed to Step 11. Otherwise, interpolate $\check{\mathbf{q}}_k(s)$ from $\mathcal{S}(\mathcal{I}_k)$ to $\mathcal{S}(\mathcal{I}_{M_T}\backslash\mathcal{I}_k)$.
10) Set $k \leftarrow k + 1$ and go back to Step 2.
11) For each $n \in \mathcal{D}\backslash\mathcal{I}_{M_T}$, apply $\mathcal{M}^{-1} : (\tilde{\mathbf{Q}}(s_n), \tilde{\mathbf{R}}(s_n)) \mapsto (\mathbf{Q}(s_n), \mathbf{R}(s_n))$.

For an in-depth discussion of the computational complexity of Algorithms I-MMSE through III-MMSE, the interested reader is referred to [24].

VIII. Efficient Interpolation

Throughout this section, we consider interpolation of a generic LP $a(s) \sim (V_1, V_2)$ of maximum degree $V = V_1 + V_2$ from the set of base points $\mathcal{B}$ to the set of base points $\mathcal{T}$, where $|\mathcal{B}| = B$ and $|\mathcal{T}| = T$. We note that in the context of interpolation in MIMO-OFDM systems, relevant for the algorithms presented in this paper, all base points and all target points correspond to OFDM tones. Therefore, in the following we assume that $\mathcal{B}$ and $\mathcal{T}$ satisfy the condition

$$\mathcal{B} \cup \mathcal{T} \subseteq \{s_0, s_1, \ldots, s_{N-1}\}. \tag{19}$$

The complexity analysis in Section VI showed that interpolation-based QR decomposition algorithms yield savings over the brute-force approach only if the interpolation complexity per target point c_{IP} is sufficiently small. Straightforward interpolation of $a(s)$, which corresponds to direct evaluation of (2), is performed by carrying out the multiplication of the $T \times B$ interpolation matrix $\mathbf{T}\mathbf{B}^\dagger$ by the $B \times 1$ vector $\mathbf{a}_\mathcal{B}$. The corresponding complexity is given by TB, which results in $c_{\text{IP}} = B$ full multiplications per target point. Since exact interpolation of $\tilde{\mathbf{q}}_k(s) \sim (kL, kL)$ and $\tilde{\underline{\mathbf{r}}}_k(s) \sim (kL, kL)$ requires $B \geq 2kL + 1$ $(k = 1, 2, \ldots, M_T)$, with the worst case being $B \geq 2M_TL + 1$, this complexity may be too high to realize savings over brute-force QR decomposition. In this section, we present interpolation methods characterized by significantly smaller values of c_{IP}. As demonstrated by the numerical results in Section IX, this can then lead to

significant savings of the interpolation-based approaches for QR decomposition over the brute-force approach.

A. Interpolation with Dedicated Multipliers

As already noted, the interpolation matrix $\mathbf{TB}^\dagger$ is a function of $\mathcal{B}$, $\mathcal{T}$, V_1 and V_2, but not of the realization of the LP $a(s)$ to be interpolated. Hence, as long as $\mathcal{B}$, $\mathcal{T}$, V_1 and V_2 do not change, multiple LPs can be interpolated using the same interpolation matrix $\mathbf{TB}^\dagger$, which can be computed off-line. This observation leads to the first strategy for efficient interpolation, which consists of carrying out the matrix-vector product $(\mathbf{TB}^\dagger)\mathbf{a}_\mathcal{B}$ in (2) through TB constant multiplications, where the entries of $\mathbf{TB}^\dagger$ are constant and the entries of $\mathbf{a}_\mathcal{B}$ are variable.

In the context of VLSI implementation, full multiplications and constant multiplications differ significantly. Whereas a full multiplication must be performed by a *full multiplier* which processes two variable operands, in a constant multiplication, the fact that one of the operands, and more specifically its binary representation, is known a priori, can be exploited to perform binary logic simplifications that result in a drastically lower circuit complexity [22]. The resulting multiplier, called a *dedicated multiplier* in the following, consumes only a fraction of the silicon area (down to $1/9$, as reported in [14] for complex-valued dedicated multipliers) required by a full multiplier, and exhibits the same processing delay.

In the remainder of the paper, $\chi_\mathbb{C}$ and $\chi_\mathbb{R}$ denote the complexity associated with a constant multiplication of a complex-valued variable operand by a complex-valued and by a real-valued constant coefficient, respectively. Since $\mathbf{TB}^\dagger$ is real-valued for $V_1 = V_2$ and complex-valued otherwise, interpolation through constant multiplications with dedicated multipliers has a complexity per target point of

$$c_{\text{IP}} = \begin{cases} \chi_\mathbb{R} B, & V_1 = V_2 \\ \chi_\mathbb{C} B, & V_1 \neq V_2. \end{cases}$$

By leaving a cautionary implementation margin from the best-effort value of $1/9$ reported in [14], we assume that $\chi_\mathbb{C} = 1/4$ in the remainder of the paper. Since the multiplication of two complex-valued numbers requires (assuming straightforward implementation) four real-valued multiplications, whereas multiplying a real-valued number by a complex-valued number requires only two real-valued multiplications, we henceforth assume that $\chi_\mathbb{R} = \chi_\mathbb{C}/2$, which leads to $\chi_\mathbb{R} = 1/8$.

B. Equidistant Base Points

In the following, we say that the points in a set $\{u_0, u_1, \ldots, u_{K-1}\} \subset \mathcal{U}$ are *equidistant on* $\mathcal{U}$ if $u_k = u_0 e^{j2\pi k/K}$ for $k = 1, 2, \ldots, K-1$. So far, we discussed interpolation of $a(s) \sim (V_1, V_2)$ for generic sets $\mathcal{B}$ and $\mathcal{T}$. In the remainder of Section VIII we will, however, focus on the following special case. Given integers $B, R > 1$, we consider the set of B base points $\mathcal{B} = \{b_k = e^{j2\pi k/B} : k = 0, 1, \ldots, B-1\}$ and the set of $T = (R-1)B$ target points $\mathcal{T} = \{t_{(R-1)k+r-1} = b_k e^{j2\pi r/(RB)} : k = 0, 1, \ldots, B-1, r = 1, 2, \ldots, R-1\}$. We note that both the B points in $\mathcal{B}$ and the RB points in $\mathcal{B} \cup \mathcal{T} = \{e^{j2\pi l/(RB)} : l = 0, 1, \ldots, RB-1\}$ are equidistant on $\mathcal{U}$. Hence, interpolation of $a(s)$ from $\mathcal{B}$ to $\mathcal{T}$ essentially amounts to an R-fold increase in the sampling rate of $a(s)$ on $\mathcal{U}$, and will therefore be termed *upsampling of $a(s)$ from B equidistant base points by a factor of R* in the remainder of the paper. The corresponding base point matrix $\mathbf{B}$ and target point matrix $\mathbf{T}$ are constructed according to their definitions in Section II-C. We note that for $B \geq V + 1$, $\mathbf{B}$ satisfies $\mathbf{B}^H\mathbf{B} = B\mathbf{I}_B$ and hence $\mathbf{B}^\dagger = (1/B)\mathbf{B}^H$.

We recall that the number of OFDM tones N is typically a power of two. Therefore, in order to have RB equidistant points on $\mathcal{U}$ while satisfying the condition (19), both B and R are constrained to be powers of two. In order to satisfy the condition $B \geq V + 1$ mandated by the requirement of exact interpolation, we set $B = 2^{\lceil \log(V+1) \rceil}$.

Finally, we mention that samples of $a(s)$ at a set of equidistant base points on $\mathcal{U}$ can be obtained from samples of $a(s)$ at any set of base points on $\mathcal{U}$ (of cardinality at least $V + 1$) through an additional interpolation step. This idea is discussed in detail in the context of MIMO-OFDM channel estimation in [14].

C. Interpolation by FIR Filtering

This section discusses upsampling of $a(s)$ from B equidistant base points by a factor of R by finite impulse response (FIR) filtering. We mention that an alternative FFT-based approach is presented in [24].

Proposition 9. *In the context of upsampling from B equidistant base points by a factor of R, the $B(R-1) \times B$ interpolation matrix $\mathbf{TB}^\dagger$ satisfies the following properties:*

1) *There exists an $(R-1) \times B$ matrix $\mathbf{F}_0$ such that $\mathbf{TB}^\dagger$ can be written as*

$$\mathbf{TB}^\dagger = \left[(\mathbf{F}_0\mathbf{C}_B)^T \ \ (\mathbf{F}_0\mathbf{C}_B^2)^T \ \cdots \ (\mathbf{F}_0\mathbf{C}_B^B)^T\right]^T \quad (20)$$

with the $B \times B$ circulant matrix

$$\mathbf{C}_B \triangleq \begin{bmatrix} \mathbf{0} & \mathbf{I}_{B-1} \\ 1 & \mathbf{0} \end{bmatrix}.$$

2) *The matrix $\mathbf{F}_0$, as implicitly defined in (20), satisfies*

$$\left[\mathbf{F}_0\right]_{r,k+1} = \left[\mathbf{F}_0\right]^*_{R-r,B-k}$$

for $r = 1, 2, \ldots, R-1$ and $k = 0, 1, \ldots, B-1$.

Proof: Since $\mathbf{B}^\dagger = (1/B)\mathbf{B}^H$, the entries of $\mathbf{TB}^\dagger$ are given by

$$\left[\mathbf{TB}^\dagger\right]_{k(R-1)+r,k'+1} = \frac{1}{B} \sum_{v=-V_1}^{V_2} e^{-j2\pi v \frac{R(k-k')+r}{RB}} \quad (21)$$

for $k, k' = 0, 1, \ldots, B-1$ and $r = 1, 2, \ldots, R-1$. The two properties are now established as follows:

1) The RHS of (21) remains unchanged upon replacing k and k' by $(k+1) \bmod B$ and $(k'+1) \bmod B$, respectively. Hence, for a given $r \in \{1, 2, \ldots, R-1\}$, the $B \times B$ matrix obtained by stacking the rows indexed by $r, (R-1)+r, \ldots, (B-1)(R-1)+r$ (in this order)

of $\mathbf{T}\mathbf{B}^\dagger$ is circulant. By taking $\mathbf{F}_0$ to consist of the last $R-1$ rows of $\mathbf{T}\mathbf{B}^\dagger$, and using $\mathbf{C}_B^B = \mathbf{I}_B$, along with the fact that for $b \in \mathbb{Z}$, the multiplication $\mathbf{F}_0\mathbf{C}_B^b$ corresponds to circularly shifting the columns of $\mathbf{F}_0$ to the right by $b \bmod B$ positions, we obtain (20).

2) The entries of $\mathbf{F}_0$ are obtained by setting $k = B-1$ in (21) and are given by

$$[\mathbf{F}_0]_{r,k'+1} = \frac{1}{B}\sum_{v=-V_1}^{V_2} e^{-j2\pi v\frac{r-R(k'+1)}{RB}}$$

for $r = 1, 2, \ldots, R-1$ and $k' = 0, 1, \ldots, B-1$. Hence, we obtain

$$\begin{aligned}[\mathbf{F}_0]^*_{R-r,B-k'} &= \frac{1}{B}\sum_{v=-V_1}^{V_2} e^{j2\pi v\frac{R-r-R(B-k')}{RB}} \\ &= \frac{1}{B}\sum_{v=-V_1}^{V_2} e^{-j2\pi v\frac{r-R(k'+1)}{RB}} \\ &= [\mathbf{F}_0]_{r,k'+1}\end{aligned}$$

for $r = 1, 2, \ldots, R-1$ and $k' = 0, 1, \ldots, B-1$.

■

We note that Property 1 in Proposition 9 implies that the matrix-vector multiplication $(\mathbf{T}\mathbf{B}^\dagger)\mathbf{a}_\mathcal{B}$ in (2) can be carried out through the application of $R-1$ FIR filters. Specifically, for $r = 1, 2, \ldots, R-1$, the entries $r, r+R, \ldots, r+(B-1)R$ of $\mathbf{a}_\mathcal{T}$ can be obtained by computing the circular convolution of $\mathbf{a}_\mathcal{B}$ with the impulse response of length B contained in the rth row of $\mathbf{F}_0$. By allocating B dedicated multipliers per FIR filter (one per impulse response tap), we would need a total of $(R-1)B$ dedicated multipliers. We will next see that the complex-conjugate symmetry in the rows of $\mathbf{F}_0$, formulated as Property 2 in Proposition 9, allows to reduce the number of dedicated multipliers and the interpolation complexity by a factor of two.

In the following, we assume that the multiplications of a variable complex-valued operand by a constant $\gamma \in \mathbb{C}$ and by its complex conjugate γ^* can be carried out using the same dedicated multiplier, and that the resulting complexity is comparable to the complexity of multiplication by γ alone. This is justified as the multiplication by γ^*, compared to the multiplication by γ, involves the same four underlying real-valued multiplications and only requires two additional sign flips, which have significantly smaller complexity than the real-valued multiplications. Thus, we can perform multiplication by the coefficients $[\mathbf{F}_0]_{r,k+1}$ and $[\mathbf{F}_0]_{R-r,B-k} = [\mathbf{F}_0]^*_{r,k+1}$ through a single dedicated multiplier ($r = 1, 2, \ldots, R/2$, $k = 0, 1, \ldots, B/2-1$). This resource sharing approach leads to

$$c_{\text{IP}} = \begin{cases} \frac{\chi_\mathbb{R}}{2}B, & V_1 = V_2 \\ \frac{\chi_\mathbb{C}}{2}B, & V_1 \neq V_2. \end{cases} \tag{22}$$

So far, we assumed that $a(s)$ is interpolated from the $B = 2^{\lceil \log(V+1)\rceil}$ base points in $\mathcal{B}$, resulting in c_{IP} as in (22). We will next show that the interpolation complexity can be further reduced by using a smaller number of base points $B' < B$. Interpolation will be exact as long as the condition $B' \geq V+1$ is satisfied.

As done above, we assume knowledge of the B samples $a(s), s \in \mathcal{B}$. In the following, however, we require that for a given target point t_r, the sample $a(t_r)$ is obtained by interpolation from only B' base points, picked from the B elements of $\mathcal{B}$ as a function of t_r. For simplicity of exposition, we assume that B' is even, and for every $t_r \in \mathcal{T}$ we choose the B' elements of $\mathcal{B}$ that are located closest to t_r on $\mathcal{U}$. We next show that the resulting interpolation of $a(s)$ from $\mathcal{B}$ to $\mathcal{T}$ can be performed by FIR filtering.

In the following, we define B disjoint subsets $\mathcal{T}_k$ of $\mathcal{T}$ (satisfying $\bigcup_{k=0}^{B-1}\mathcal{T}_k = \mathcal{T}$) and consider the corresponding subsets $\mathcal{B}_k$ of $\mathcal{B}$, defined such that for all points in $\mathcal{T}_k$, the B' closest base points are given by the elements of $\mathcal{B}_k$ ($k = 0, 1, \ldots, B-1$). We next show that the interpolation matrix corresponding to interpolation of $a(s)$ from $\mathcal{B}_k$ to $\mathcal{T}_k$ is independent of k. To this end, we first consider the set of target points $\mathcal{T}_0 \triangleq \{t_{(B-1)(R-1)+r-1} : r = 1, 2, \ldots, R-1\}$, containing the $R-1$ target points located on $\mathcal{U}$ between the base points b_{B-1} and b_0. The subset of $\mathcal{B}$ containing the B' points that are closest to every point in $\mathcal{T}_0$ is given by $\mathcal{B}_0 \triangleq \{b_0, b_1, \ldots, b_{B'/2}, b_{B-B'/2}, b_{B-B'/2+1}, \ldots, b_{B-1}\}$. Interpolation of $a(s)$ from $\mathcal{B}_0$ to $\mathcal{T}_0$ involves the base point matrix $\mathbf{B}_0$, the target point matrix $\mathbf{T}_0$, and the interpolation matrix $\mathbf{T}_0\mathbf{B}_0^\dagger$, constructed as described in Section II-C. Next, for $k = 1, 2, \ldots, B-1$, we denote by $\mathcal{B}_k$ and $\mathcal{T}_k$ the sets obtained by multiplying all elements of $\mathcal{B}_0$ and $\mathcal{T}_0$, respectively, by $e^{j2\pi k/B}$. We note that $\mathcal{T}_k$ contains the $R-1$ target points located on $\mathcal{U}$ between the base points b_{k-1} and b_k, and that $\mathcal{B}_k$ is the subset of $\mathcal{B}$ containing the B' points that are closest to every point in $\mathcal{T}_k$. With the unitary matrix $\mathbf{S}_k \triangleq \operatorname{diag}((e^{j2\pi k/B})^{V_1}, (e^{j2\pi k/B})^{V_1-1}, \ldots, (e^{j2\pi k/B})^{-V_2})$, interpolation of $a(s)$ from $\mathcal{B}_k$ to $\mathcal{T}_k$ involves the base point matrix $\mathbf{B}_k = \mathbf{B}_0\mathbf{S}_k$, with pseudoinverse $\mathbf{B}_k^\dagger = \mathbf{S}_k^{-1}\mathbf{B}_0^\dagger$, the target point matrix $\mathbf{T}_k = \mathbf{T}_0\mathbf{S}_k$, and the interpolation matrix $\mathbf{T}_k\mathbf{B}_k^\dagger = \mathbf{T}_0\mathbf{S}_k\mathbf{S}_k^{-1}\mathbf{B}_0^\dagger = \mathbf{T}_0\mathbf{B}_0^\dagger$ ($k = 1, 2, \ldots, B-1$). Hence, the interpolation matrix is independent of k and is the same as in the interpolation of $a(s)$ from $\mathcal{B}_0$ to $\mathcal{T}_0$.

Now, interpolation of $a(s)$ from $\mathcal{B}$ to $\mathcal{T}$, with the constraint that the sample of $a(s)$ at every target point is computed only from the samples of $a(s)$ at the B' closest base points, amounts to performing interpolation of $a(s)$ from $\mathcal{B}_k$ to $\mathcal{T}_k$ for all $k = 0, 1, \ldots, B-1$, and can be written in a single equation as $\mathbf{a}_\mathcal{T} = \mathbf{F}\mathbf{a}_\mathcal{B}$. Here, the $(R-1)B \times B$ interpolation matrix $\mathbf{F}$ is equal to the RHS of (20), with the $(R-1) \times B$ matrix

$$\mathbf{F}_0 = \begin{bmatrix} (\mathbf{T}_0\mathbf{B}_0^\dagger)_{1,B'/2} & \mathbf{0} & (\mathbf{T}_0\mathbf{B}_0^\dagger)_{B-B'/2+1,B} \end{bmatrix} \tag{23}$$

which contains an all-zero submatrix of dimension $(R-1) \times (B-B')$. Hence, $\mathbf{F}$ satisfies Property 1 of Proposition 9, with $\mathbf{F}_0$ given by (23). In addition, we state without proof that $\mathbf{F}_0$ in (23) satisfies Property 2 of Proposition 9. We can therefore conclude that interpolation from the closest B' base points maintains the structural properties of interpolation from all B base points and, as above, can be performed by FIR filtering using $R-1$ filters with dedicated multipliers that exploit the conjugate symmetry in the rows of $\mathbf{F}_0$. Since the rows of $\mathbf{F}_0$ in (23) contain $B-B'$ zeros, the $R-1$ impulse

responses now have length B', and we obtain

$$c_{\text{IP}} = \begin{cases} \frac{\chi_{\mathbb{R}}}{2} B', & V_1 = V_2 \\ \frac{\chi_{\mathbb{C}}}{2} B', & V_1 \neq V_2. \end{cases} \tag{24}$$

IX. Numerical Results

The results presented so far do not depend on a specific QR decomposition method. For the numerical complexity comparisons presented in this section, we become more specific and assume UT-based QR decomposition performed through Givens rotations and coordinate rotation digital computer (CORDIC) operations [25], [26], which is the method of choice in VLSI implementations [23], [27]. For a generic matrix $\mathbf{A} \in \mathbb{C}^{P \times M}$ with $P \geq M$, it was shown in [23] that the complexity of UT-based QR decomposition of $\mathbf{A}$, as required in Algorithms I–III, is given by $c_{\text{QR}}^{P \times M} = \frac{3}{2}(P^2M + PM^2) - M^3 - \frac{1}{2}(P^2 - P + M^2 + M)$ and that the complexity of UT-based MMSE-QR decomposition of $\mathbf{A}$, as required in Algorithms I-MMSE and II-MMSE, is given[5] by $c_{\text{MMSE-QR}}^{P \times M} \triangleq \frac{3}{2}(P^2M + PM^2) - \frac{1}{2}P^2 + \frac{1}{2}P$. The results in [23] carry over, in a straightforward fashion, to UT-based QR decomposition of the augmented matrix $[\,\mathbf{A}^T \quad \alpha \mathbf{I}_M\,]^T$, as required in Algorithm III-MMSE, to yield $c_{\text{QR,III-MMSE}}^{P \times M} \triangleq c_{\text{MMSE-QR}}^{P \times M} + \frac{3}{2}PM^2 + \frac{1}{2}PM$.

A. Efficient Interpolation and Performance Degradation

The interpolation complexity (24) of the approach described in Section VIII-C can be further reduced by choosing the number of base points B' to be smaller than the number of coefficients $V + 1$ that determine an LP of maximum degree V. This violates the condition $B' \geq V + 1$ for exact interpolation and therefore leads to a systematic interpolation error. In the following, we evaluate the resulting trade-off between interpolation complexity and interpolation accuracy in the context of an interpolation-based MIMO-OFDM receiver, as we gradually reduce B', and hence also c_{IP} as in (24), in the interpolation of $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$ as required by Algorithm II. The corresponding analysis for the interpolation of $\tilde{\mathbf{q}}_k(s)$ and $\tilde{\mathbf{r}}_k^T(s)$, $k = 1, 2, \ldots, M_T$, as required by Algorithm III, is more involved and does not yield any additional insights into the trade-off under consideration. In order to simplify the simulation setup, we interpolate the channel transfer function matrix $\mathbf{H}(s)$ exactly and perform inexact interpolation only on the LP matrices $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$. The numerical results presented in the following demonstrate that for Algorithm II to have smaller complexity than Algorithm I, setting B' to a value smaller than $V + 1$, and hence accepting a systematic error in the interpolation of $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$, may be necessary. On the other hand, we will also see that the resulting degradation in detection performance, in terms of both coded and uncoded bit error rate (BER), can be negligible even for values of B' that are significantly smaller than $V + 1$.

In the following, we consider a MIMO-OFDM system with $D = N = 512$, $M_R = 4$, and either $M_T = 2$ or $M_T = 4$, operating over a frequency-selective channel with $L = 15$. The data symbols are drawn from a 16-QAM constellation. In the coded case, a rate $1/2$ convolutional code with constraint length 7 and generator polynomials $[133_o\ 171_o]$ is used. The receiver performs maximum-likelihood detection through hard-output sphere decoding. Our results are obtained through Monte Carlo simulation, where averaging is performed over the channel impulse response taps $\mathbf{H}_0, \mathbf{H}_1, \ldots, \mathbf{H}_L$, assumed to have entries i.i.d. $\mathcal{CN}(0, 1/(L+1))$. This assumption on the channel statistics, along with the average transmit power being given by $\mathbb{E}[\mathbf{c}_n^H \mathbf{c}_n] = 1$ and the noise variance σ_w^2, implies that the per-antenna receive signal-to-noise ratio (SNR) is $1/\sigma_w^2$. The receiver employs either Algorithm I or Algorithm II to compute $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$ at all tones. Interpolation is assumed to be performed through FIR filtering with dedicated multipliers. As discussed in Section VIII-A, we assume that a multiplication of a variable operand by a complex-valued and by a real-valued FIR filter coefficient has a complexity corresponding to $\chi_{\mathbb{C}} = 1/4$ and to $\chi_{\mathbb{R}} = 1/8$ full multiplications, respectively. In Step 1 of both algorithms, the LP matrix $\mathbf{H}(s) \sim (0, L)$ with maximum degree $V = L$ is interpolated exactly from $B = L + 1 = 16$ equidistant base points by FIR filtering. Since $0 = V_1 \neq V_2 = L$, the corresponding interpolation complexity per target point is obtained from (22) as $c_{\text{IP},\mathbf{H}} \triangleq (L + 1)\chi_{\mathbb{C}}/2 = 2$. In Step 4 of Algorithm II, we interpolate $\tilde{\mathbf{Q}}(s) \sim (M_T L, M_T L)$ and $\tilde{\mathbf{R}}(s) \sim (M_T L, M_T L)$, with maximum degree $V = 2M_T L$, through FIR filtering from $B' \leq B = 2^{\lceil \log(V+1) \rceil}$ base points. With $V_1 = V_2 = M_T L$, the corresponding interpolation complexity per target point is obtained from (24) as $c_{\text{IP},(\tilde{\mathbf{Q}},\tilde{\mathbf{R}})} \triangleq \chi_{\mathbb{R}} B'/2$. Here, the FIR filters for exact interpolation derived in Section VIII-C are replaced by FIR filters of length $B' < V + 1$ obtained using an ad-hoc method described in [24]. Finally, since at the tones $n \in \mathcal{D} \backslash \mathcal{I}_{M_T}$, the matrices $\mathbf{Q}(s_n)$ and $\mathbf{R}(s_n)$ are computed through mapping, interpolation, and inverse mapping in Algorithm II rather than directly from $\mathbf{H}(s_n)$ as in Algorithm I, in fixed-point implementations Algorithm II can be expected to suffer more from error propagation than Algorithm I. In order to compensate for this, Algorithm II may require the use of larger wordwidths, implying an increase in its overall complexity C_{II}. This aspect will not be taken into account in the following. In the numerical results shown below, we ensure that systematic errors in the interpolation of $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$, arising from B' being smaller than $V + 1$, are the sole source of detection performance degradation by using double-precision floating-point arithmetic.

Table II summarizes the simulation parameters, along with the corresponding values of the interpolation complexity per target point $c_{\text{IP},(\tilde{\mathbf{Q}},\tilde{\mathbf{R}})}$ and the resulting algorithm complexity ratio $C_{\text{II}}/C_{\text{I}}$, which quantifies the savings of Algorithm II over Algorithm I. We note that for $M_T = 4$, exact interpolation results in $C_{\text{II}} > C_{\text{I}}$. Hence, in this case inexact interpolation is necessary to obtain complexity savings of Algorithm II over Algorithm I. In contrast, for $M_T = 2$, Algorithm II exhibits lower complexity than Algorithm I even in the case of exact interpolation.

Figs. 1a and 1b show the resulting BER performance for $M_T = 2$ and $M_T = 4$, respectively, both for the coded and the

[5] In [23], the last term in the expression for $c_{\text{MMSE-QR}}^{P \times M}$ was erroneously specified as $-(1/2)P$.

Table II
SIMULATION PARAMETERS

$M_T = 2$			$M_T = 4$			Interpolation of
B'	$c_{\text{IP},(\tilde{\mathbf{Q}},\tilde{\mathbf{R}})}$	$C_{\text{II}}/C_{\text{I}}$	B'	$c_{\text{IP},(\tilde{\mathbf{Q}},\tilde{\mathbf{R}})}$	$C_{\text{II}}/C_{\text{I}}$	$\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$
64	4	0.82	128	8	1.52	exact
32	2	0.55	32	2	0.71	inexact
16	1	0.41	24	1.5	0.64	inexact
12	0.75	0.37	16	1	0.57	inexact
8	0.5	0.34	8	0.5	0.51	inexact

Common to all simulations are the parameters $D = N = 512$, $L = 15$, $M_R = 4$, and $c_{\text{IP},\mathbf{H}} = 2$.

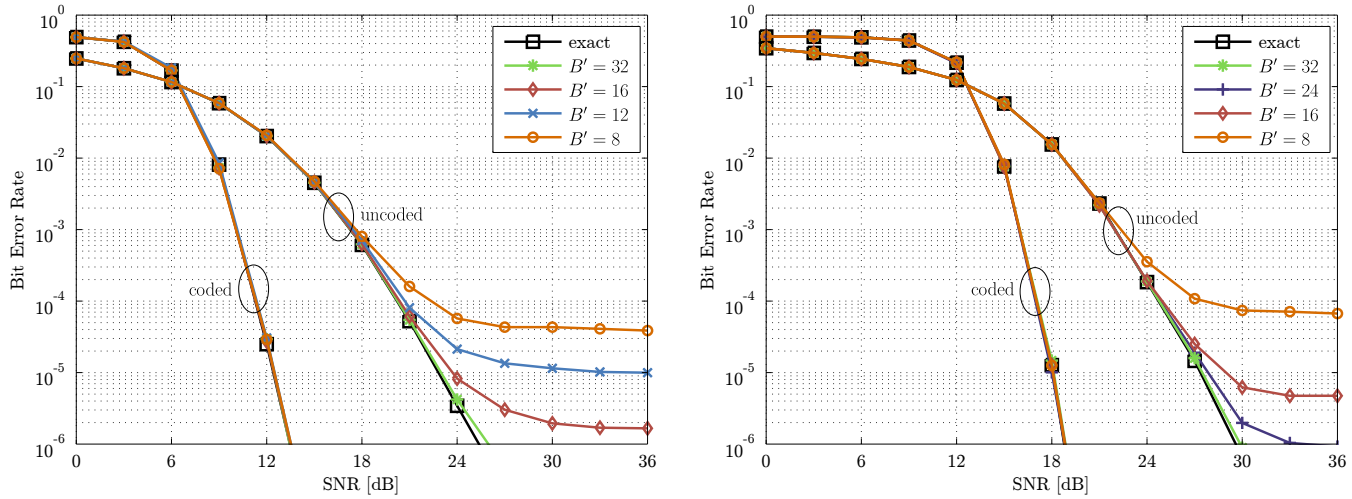


Figure 1. Bit error rates as a function of SNR for different interpolation filter lengths, with and without channel coding, for (a) $M_T = 2$ and (b) $M_T = 4$. The results corresponding to exact QR decomposition are provided for reference.

uncoded case. For uncoded transmission and inexact interpolation, we observe an error floor at high SNR which rises with decreasing B'. For $M_T = 2$ and uncoded transmission, we can see in Fig. 1a and Table II, respectively, that an interpolation filter length of $B' = 8$ results in negligible performance loss for SNR values of up to 18 dB, and yields complexity savings of Algorithm II over Algorithm I of 66%. Choosing $B' = 16$ yields close-to-optimum performance for SNR values of up to 24 dB and complexity savings of 59%. For $M_T = 4$ and uncoded transmission, Fig. 1b and Table II show that the interpolation filter length can be shortened from $B' = 128$ to $B' = 8$, leading to complexity savings of Algorithm II over Algorithm I of 49%, at virtually no performance loss in the SNR range of up to 21 dB. Setting $B' = 32$ results in a performance loss, compared to exact interpolation, of less than 1 dB at BER $= 10^{-6}$ and in complexity savings of 29%. In the coded case, both for $M_T = 2$ and $M_T = 4$, we can see in Figs. 1a and 1b that the BER curves for Algorithm II, for all values of B' under consideration, essentially overlap with the corresponding curves for Algorithm I for BERs down to 10^{-6}. This observation suggests that the use of channel coding allows to employ significantly shorter FIR filters for the interpolation of $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$ (corresponding to a smaller $c_{\text{IP},\tilde{\mathbf{Q}}\tilde{\mathbf{R}}}$ and hence to a lower C_{II}, which in turn implies higher savings of Algorithm II over Algorithm I) than in the uncoded case. We conclude that in the practically relevant case of coded transmission, complexity savings of Algorithm II over Algorithm I can be obtained at negligible detection performance loss. Further numerical results indicate that the latter statement continues to hold even in the presence of channel estimation errors induced by training-based estimation of the channel matrices $\mathbf{H}(s_n), n \in \mathcal{E}$. Specifically, here we applied the method described in [28] (in the context of flat-fading MIMO) on a per-tone basis.

B. Algorithm Complexity Comparisons

The discussion in Section VIII and the numerical results in Section IX-A demonstrate that for the case of upsampling from equidistant base points, small values of c_{IP} can be achieved and inexact interpolation does not necessarily induce a significant detection performance loss. Therefore, in the following, we assume that for all $k = 1, 2, \ldots, M_T$, the set $\mathcal{I}_k$, containing indices of OFDM tones used as base points, is such that $\mathcal{S}(\mathcal{I}_k)$ contains $B_k = |\mathcal{I}_k| = 2^{\lceil \log_2(2kL+1) \rceil}$ points that are equidistant on $\mathcal{U}$, and we set $c_{\text{IP}} = 2$.

For $D = 500$, $L = 15$, and different values of M_T and M_R, Fig. 2a shows the complexity of Algorithms II and III as percentage of the complexity of Algorithm I. We observe savings of Algorithms II and III over Algorithm I as high as 48% and 53%, respectively. Furthermore, we can see that Algorithm III exhibits a lower complexity than Algorithm II in all considered configurations. This is a consequence of the small value of c_{IP} and of Algorithm III, with respect to Algorithm II, trading

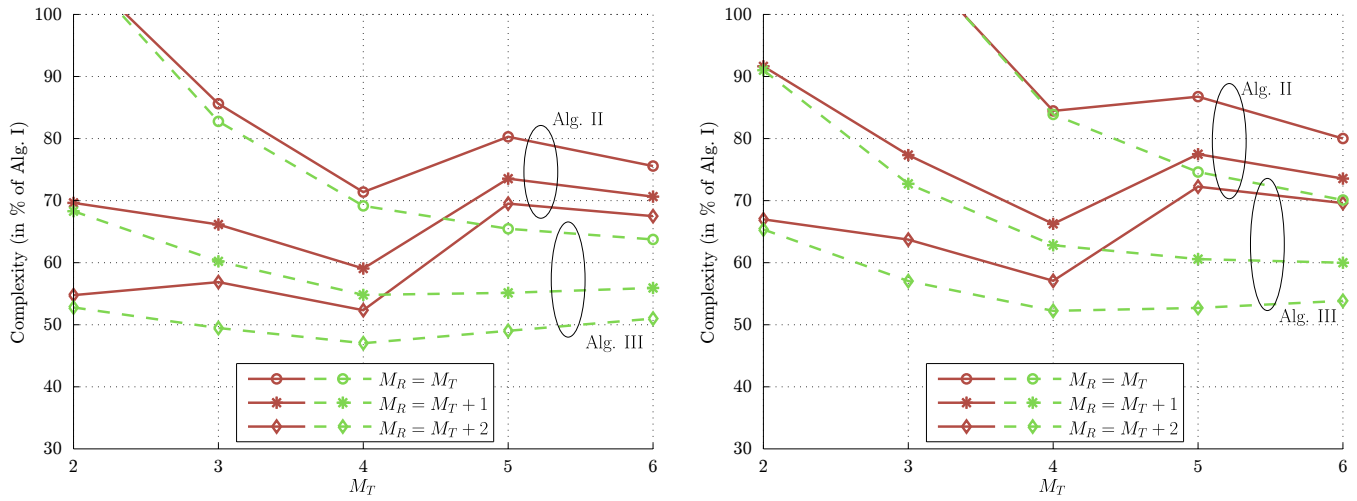


Figure 2. Complexity of Algorithms II and III as percentage of complexity of Algorithm I for $D = 500$, and $L = 15$, (a) including and (b) excluding the complexity of interpolation of $\mathbf{H}(s)$.

lower QR decomposition cost against higher interpolation cost. Moreover, we observe that the savings of Algorithms II and III over Algorithm I are more pronounced for larger $M_R - M_T$. For the special case $\mathcal{E} = \mathcal{D}$ implying i) that $\mathbf{H}(s_n)$ is known at all data-carrying tones $n \in \mathcal{D}$ and ii) that Algorithm I simplifies to the computation of D QR decompositions, Fig. 2b shows that the relative savings of Algorithms II and III over Algorithm I are somewhat reduced, but still significant.

For $D = 500$, $M_T = M_R$, and different values of L, Fig. 3a shows the complexity of Algorithms II-MMSE and III-MMSE as percentage of the complexity of Algorithm I-MMSE. The fact (which also carries over to the savings of Algorithms II and III over Algorithm I) that the savings of Algorithms II-MMSE and III-MMSE over Algorithm I-MMSE are more pronounced for smaller values of L is a consequence of the number of base points B_k being an increasing function of L. In Fig. 3a, we can see that despite the low interpolation complexity implied by $c_{\text{IP}} = 2$, Algorithm III-MMSE exhibits a higher complexity than Algorithm II-MMSE for all considered values of $M_T = M_R$ and L. This is a consequence of the fact that the overall complexity of the UT-based QR decompositions required in Step 6 of Algorithm III-MMSE can be larger than the overall complexity of the UT-based MMSE-QR decompositions required in Step 2 of Algorithm II-MMSE. We mention that if (MMSE-)QR decomposition is carried out by GS-based algorithms rather than by UT-based algorithms, Algorithm III-MMSE can exhibit a lower complexity than Algorithm II-MMSE for a nonempty range of values of the parameters $M_T = M_R$ and L.

Finally, Fig. 3b shows the absolute complexity of Algorithms I–III and I-MMSE through III-MMSE as a function of D, for $M_T = 3$, $M_R = 4$, and $L = 15$. We observe that the complexity savings of Algorithms II and III over Algorithm I and the savings of Algorithms II-MMSE and III-MMSE over Algorithm I-MMSE grow linearly in D. This behavior was predicted for Algorithms I and II by the analysis in Section VI-D, where we showed that $C_{\text{I}} - C_{\text{II}}$ is an affine function of D and is positive for small c_{IP} and large D.

X. Concluding Remarks

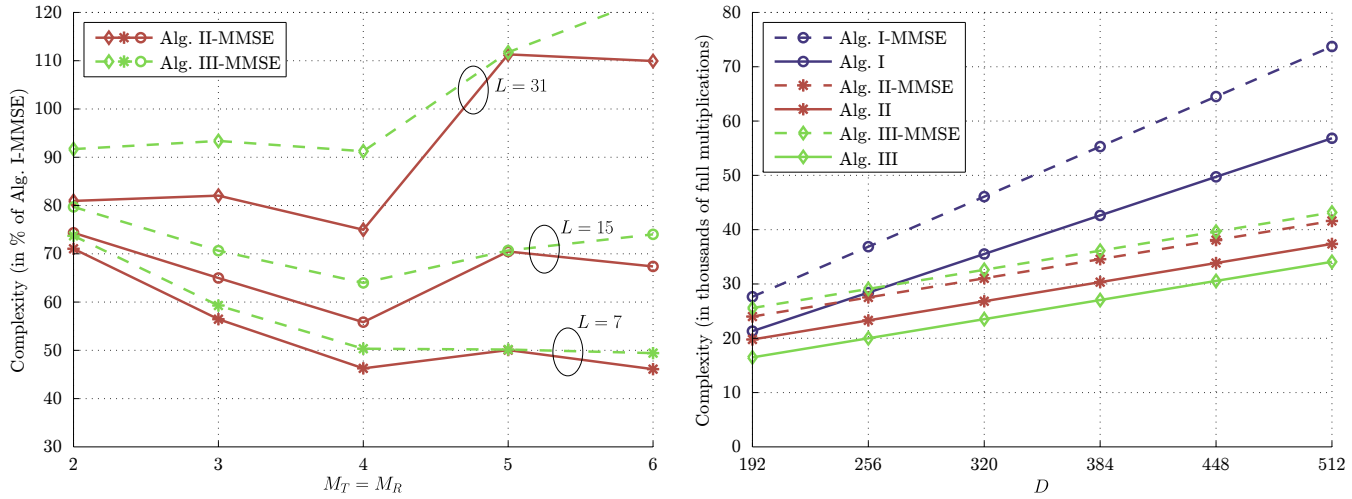
We demonstrated that, for a wide range of system parameters, the proposed interpolation-based QR decomposition algorithms can yield significant complexity savings over brute-force per-tone QR decomposition, provided that the interpolation complexity is sufficiently small. Nevertheless, since $2M_TL + 1$ base points are required for the interpolation of $\tilde{\mathbf{Q}}(s)$ and $\tilde{\mathbf{R}}(s)$, and since the worst case $L = L_{\text{CP}}$ must be accounted for, these algorithms can only be applied if $N > 2M_TL_{\text{CP}} + 1$. Consequently, the proposed algorithms are not suitable for use in, e.g., the IEEE 802.11n standard and some of the configurations of the IEEE 802.16e standard, as the corresponding channel oversampling ratios N/L_{CP} are not large enough. This motivates further development of the methods proposed in this paper, with the goal of reducing the number of base points required for interpolation-based QR decomposition. Specifically, open problems include i) determining whether the mapping $\mathcal{M}$ can be replaced by a different mapping producing LP matrices with maximum degree smaller than $2M_TL$, and ii) investigating whether methods for interpolation under unitarity constraints [29], applied to the interpolation of $\mathbf{Q}(s)$, would allow to further reduce the number of base points required.

Acknowledgments

The authors would like to thank Andreas Burg and Simon Haene for many inspiring and helpful discussions, and Moritz Borgmann for his contributions in early stages of this work.

References

[1] A. J. Paulraj, R. U. Nabar, and D. A. Gore, *Introduction to Space-Time Wireless Communications*. Cambridge, U.K.: Cambridge Univ. Press, 2003.



(a) (b)

Figure 3. (a) Complexity of Algorithms II-MMSE and III-MMSE as percentage of complexity of Algorithm I-MMSE for $D = 500$. (b) Absolute complexity of Algorithms I–III and I-MMSE through III-MMSE, for $M_T = 3$, $M_R = 4$, and $L = 15$.

[2] P. Wolniansky, G. Foschini, G. Golden, and R. Valenzuela, "V-BLAST: An architecture for realizing very high data rates over the rich-scattering wireless channel," in *Proc. URSI Symp. Signals, Syst., Electron. (ISSSE)*, Pisa, Italy, Oct. 1998, pp. 295–300.

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

[4] E. Viterbo and E. Biglieri, "A universal decoding algorithm for lattice codes," in *Proc. GRETSI Symp. Signal and Image Process.*, Juan-les-Pins, France, Sep. 1993, pp. 611–614.

[5] D. Perels, S. Haene, P. Luethi, A. Burg, N. Felber, W. Fichtner, and H. Bölcskei, "ASIC implementation of a MIMO-OFDM transceiver for 192 Mbps WLANs," in *Proc. IEEE Eur. Solid-State Circuits Conf. (ESSCIRC)*, Grenoble, France, Sep. 2005, pp. 215–218.

[6] M. Borgmann and H. Bölcskei, "Interpolation-based efficient matrix inversion for MIMO-OFDM receivers," in *Proc. Asilomar Conf. Signals, Syst., Comput.*, Pacific Grove, CA, Nov. 2004, pp. 1941–1947.

[7] D. Cescato and H. Bölcskei, "QR decomposition of Laurent polynomial matrices sampled on the unit circle," *IEEE Trans. Inf. Theory*, vol. 56, no. 9, pp. 4754–4761, Sep. 2010.

[8] J. A. Foster, J. G. McWhirter, M. R. Davies, and J. A. Chambers, "An algorithm for calculating the QR and singular value decompositions of polynomial matrices," *IEEE Trans. Signal Process.*, vol. 58, no. 3, pp. 1263–1274, Mar. 2010.

[9] C. Windpassinger, R. F. H. Fischer, T. Vencel, and J. B. Huber, "Precoding in multi-antenna and multi-user communication," *IEEE Trans. Wireless Commun.*, vol. 3, no. 4, pp. 1305–1316, July 2004.

[10] G. Caire and S. Shamai (Shitz), "On the achievable throughput of a multiantenna Gaussian broadcast channel," *IEEE Trans. Inf. Theory*, vol. 49, no. 7, pp. 1691–1706, July 2003.

[11] J. Choi and R. W. Heath Jr., "Interpolation-based transmit beamforming for MIMO-OFDM with limited feedback," in *Proc. IEEE Int. Conf. Commun. (ICC)*, vol. 1, Paris, France, June 2004, pp. 249–259.

[12] O. Edfors, M. Sandell, J.-J. van de Beek, S. K. Wilson, and P. O. Börjesson, "OFDM channel estimation by singular value decomposition," *IEEE Trans. Commun.*, vol. 46, no. 7, pp. 931–939, July 1998.

[13] Y. Li, L. Cimini, and N. Sollenberger, "Robust channel estimation for OFDM systems with rapid dispersive fading channels," *IEEE Trans. Commun.*, vol. 46, no. 7, pp. 902–915, July 1998.

[14] S. Haene, A. Burg, N. Felber, and W. Fichtner, "OFDM channel estimation algorithm and ASIC implementation," in *Proc. IEEE Int. Conf. Circuits and Syst. Commun. (ICCSC)*, Bucharest, Romania, July 2006, pp. 270–275.

[15] D. Cescato, M. Borgmann, H. Bölcskei, J. Hansen, and A. Burg, "Interpolation-based QR decomposition in MIMO-OFDM systems," in *Proc. IEEE Int. Workshop Signal Process. Advances Wireless Commun. (SPAWC)*, New York, NY, Jun. 2005, pp. 945–949.

[16] D. Wübben and K.-D. Kammeyer, "Interpolation-based successive interference cancellation for per-antenna-coded MIMO-OFDM systems using P-SQRD," in *Proc. IEEE Workshop Smart Antennas*, Ulm, Germany, Mar. 2006.

[17] G. H. Golub and C. F. Van Loan, *Matrix Computations*, 3rd ed. Baltimore, MD: Johns Hopkins Univ. Press, 1996.

[18] R. A. Horn and C. R. Johnson, *Matrix Analysis*. Cambridge, U.K.: Cambridge Univ. Press, 1985.

[19] G. G. Raleigh and J. M. Cioffi, "Spatio-temporal coding for wireless communication," *IEEE Trans. Commun.*, vol. 46, no. 3, pp. 357–366, 1998.

[20] B. Hassibi, "An efficient square-root algorithm for BLAST," in *Proc. IEEE Int. Conf. Acoust., Speech, Signal Process. (ICASSP)*, vol. 2, Istanbul, Turkey, June 2000, pp. 737–740.

[21] 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.

[22] H. Kaeslin, *Digital Integrated Circuit Design*. Cambridge, U.K.: Cambridge Univ. Press, 2008.

[23] A. Burg, *VLSI Circuits for MIMO Communication Systems*, ser. in Microelectronics. Konstanz, Germany: Hartung-Gorre, 2006, vol. 169, Ph.D. thesis, ETH Zurich.

[24] D. Cescato, *Interpolation-Based Matrix Arithmetics for MIMO-OFDM Systems*, ser. in Communication Theory. Konstanz, Germany: Hartung-Gorre, 2010, vol. 7, Ph.D. thesis, ETH Zurich.

[25] J. Volder, "The CORDIC trigonometric computing technique," *IRE Trans. Electron. Comput.*, vol. EC-8, no. 3, pp. 330–334, Sep. 1959.

[26] J. S. Walther, "The story of unified CORDIC," *Kluwer J. VLSI Signal Process.*, vol. 25, no. 2, pp. 107–112, June 2000.

[27] G. Lightbody, R. Woods, and R. Walke, "Design of a parameterizable silicon intellectual property core for QR-based RLS filtering," *IEEE Trans. VLSI Syst.*, vol. 11, no. 4, pp. 659–678, Aug. 2003.

[28] T. L. Marzetta, "BLAST training: Estimating channel characteristics for high capacity space-time wireless," in *Proc. Allerton Conf. Commun., Contr., Comput.*, Monticello, IL, Sep. 1999, pp. 958–966.

[29] A. Edelman, T. A. Arias, and S. T. Smith, "The geometry of algorithms with orthogonality constraints," *SIAM J. Matrix Anal. Appl.*, vol. 20, no. 2, pp. 303–353, 1998.