Complexity of Analog Modulo Block Codes

Tim Schmitz, Felix Schäfer, Peter Jax, Peter Vary
Institute of Communication Systems (IKS)
RWTH Aachen University, Germany
{schmitz|fschaefer|jax|vary}@iks.rwth-aachen.de

***Abstract*—Analog Modulo Block codes (AMB codes) provide a low-complexity channel coding strategy for discrete-time, continuous-amplitude signals like audio, video, or other sensor data. In contrast to digital coding, the transmission does not suffer from saturation of the quality of the decoded signal with increasing channel quality even if no feedback channel is available.**

In this paper, the complexity of AMB encoding and decoding is analyzed for the Discrete Maximum Likelihood decoder (DML), the Zero Forcing decoder with Lattice Reduction (ZFLR), and the Lattice Maximum Likelihood decoder (LML). A low-complexity algorithm for generating all valid lattice points, which are needed for the DML decoder, is introduced and analyzed. For the complexity reduction of the LML decoder by radius pre-check, further studies of the channel noise probabilities are conducted. Run time measurements qualitatively illustrate the validity of the results.

I. Introduction

If the channel quality is known to the transmitter (e.g., by a feedback channel), digital transmission systems provide a good quality for the transmission of continuous-amplitude (yet discrete-time) signals. A disadvantage of digital source and channel coding is the required quantization of the signal which leads to an irreversible quantization error. Thereby, the digital transmission is designed for worst-case channel conditions, and a better channel quality does not lead to an improved transmission. Hybrid digital-analog (HDA) systems [1–4] avoid this effect by transmitting the quantization error in analog form as side information.

By using a continuous-amplitude, discrete-time channel code the quantization error can be avoided completely. A general analysis of source-channel mappings has been conducted in [5]. A more specialized approach (in analogy to conventional digital block codes) is the multiplication of an information vector with a real-valued generator matrix (Linear Analog Block Codes (LABC) [6]). However, to improve the power efficiency a non-linearity should be applied. Analog Modulo Block Codes (AMB codes), introduced in [7], apply a modulo operation after the multiplication with the generator matrix.

The advantages of AMB codes include bandwidth efficiency and low-delay transmission. Potential fields of application comprise continuous-amplitude systems with low-complexity transmitters which do not adapt to the changing channel quality, e.g., wireless sensors, microphones, or headsets.

The continuous-amplitude samples and symbols are digitally represented and processed (*pseudo-analog*). This implies AD and DA conversion with target precision.

This paper is organized as follows: After a short introduction to the system model and existing approaches for decoders [7,8], the complexity of these decoders is shown in Section II. A novel algorithm for generating all valid lattice points of a code is presented in Section III. In Section IV, the effectiveness of the method to reduce the average decoding complexity for low-noise channels, which was presented in [8], is examined. In Section V, run time measurements are used to compare the different decoders and to illustrate the validity of the results of the previous sections.

II. System Model

This section shortly recaps the system model and the basics[a] of AMB codes [7,8]. Additionally, the asymptotic complexity of the algorithms is given in *big $\mathcal{O}$ notation* [9].

Fig. 1 shows the transmission system as used in [7,8].

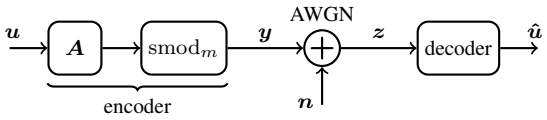


Figure 1. Transmission system [8].

II-A. Encoding

A source vector $\boldsymbol{u} \in \mathbb{R}^M$ with elements $|u_i| \leq m$ is encoded by multiplying it with a code matrix $\boldsymbol{A} \in \mathbb{R}^{M \times N}$ $(N > M)$. Subsequently, a symmetric modulo operation

$$\mathrm{smod}_m(x) = \big((x+m) \mod 2m\big) - m \quad \text{for } x \in \mathbb{R} \tag{1}$$

is applied to the elements of the resulting vector. This function limits the input symbols onto the range $(-m, +m)$ (Fig. 2). Thus, the code words $\boldsymbol{y}$ can be expressed by

$$\boldsymbol{y} = \mathrm{smod}_m(\boldsymbol{u} \cdot \boldsymbol{A})\,. \tag{2}$$

By mapping M source symbols onto N channel symbols, the code rate is $r = \frac{M}{N}$.

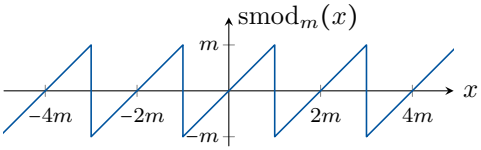


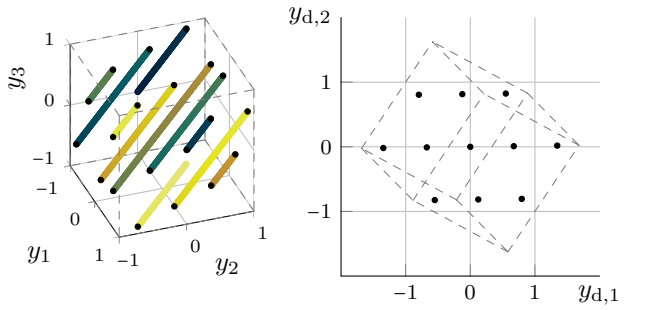
Figure 2. Modulo function [8].

[a]For additional detail, the reader is referred to the original publications.

As the value of m does not affect the performance of an Analog Modulo Block Code (AMB code) if $\boldsymbol{A}$ is scaled accordingly, $m = 1$ is assumed for all codes in this paper. We only use systematic AMB coding with $\boldsymbol{A} = [\mathbb{1}\ \tilde{\boldsymbol{A}}]$. Due to the $M \times M$ identity matrix $\mathbb{1}$, the code words contain the information words. Thus, for the encoding, only $\mathcal{O}(M \cdot D)$ operations are needed, where $D = N - M$ denotes the number of parity symbols added by the generator matrix $\boldsymbol{A}$.

II-B. Code Words and Lattice

Due to the modulo function, all code words are limited to a *modulo cube* with side length $2m$ (Fig. 3a). The valid code words are located on distinguishable lines, which are parallel M-dimensional subspaces of the code space $\mathbb{R}^N$.



(a) Code words $\boldsymbol{y}$ (b) Rotated and projected code words $\boldsymbol{y}\boldsymbol{G}_\mathrm{d}$ ($D = 2$ discrete dimensions)

Figure 3. Valid code words $\boldsymbol{y}$ with $\boldsymbol{A} = [1\ 4\ 6]$ and $m = 1$ ($M = 1$, $N = 3$ $\Rightarrow$ $D = 2$). The dashed lines - - - are the edges of the modulo cube.

By rotating the code words with an $N \times D$ rotation and projection matrix $\boldsymbol{G}_\mathrm{d}$ (derivable from the code matrix $\boldsymbol{A}$, see [7]), $N - M = D$-dimensional discrete-amplitude parts can be separated from the M-dimensional continuous-amplitude parts. After reduction to the D discrete dimensions, the discrete parts $\boldsymbol{y}_\mathrm{d}$ of the *valid* code words form a subset of a lattice (see Fig. 3b) with base matrix $\boldsymbol{L}$:

$$\boldsymbol{y}_\mathrm{d} = \boldsymbol{y} \cdot \boldsymbol{G}_\mathrm{d} = \tilde{\boldsymbol{s}} \cdot \boldsymbol{B} = \tilde{\boldsymbol{s}}' \cdot \boldsymbol{L} \quad \text{with } \tilde{\boldsymbol{s}}, \tilde{\boldsymbol{s}}' \in \mathbb{Z}^D. \tag{3}$$

The base matrix $\boldsymbol{B}$ is a submatrix of $2m\boldsymbol{G}_\mathrm{d}$ and can be reduced [10, 11] to a representation $\boldsymbol{L}$ with preferably shorter base vectors [7].

II-C. Transmission

For the evaluations performed in this paper, it is assumed that the code vector $\boldsymbol{y} \in \mathbb{R}^N$ is transmitted over an Additive White Gaussian Noise (AWGN) channel with the signal-to-noise-ratio

$$CSNR = \frac{\mathrm{E}\left\{\|\boldsymbol{y}\|^2\right\}}{\mathrm{E}\left\{\|\boldsymbol{n}\|^2\right\}} = \frac{N \cdot \sigma_y^2}{N \cdot \sigma_n^2} = \frac{\sigma_y^2}{\sigma_n^2}. \tag{4}$$

The channel is modeled by adding a Gaussian distributed noise vector $\boldsymbol{n} \sim \mathcal{N}(\boldsymbol{0},\ \sigma_n^2 \cdot \mathbb{1})$, $\boldsymbol{n} \in \mathbb{R}^N$, yielding a received vector

$$\boldsymbol{z} = \boldsymbol{y} + \boldsymbol{n}, \tag{5}$$

which is processed by a decoder in order to get an estimate $\hat{\boldsymbol{u}} \in \mathbb{R}^M$ of the source vector $\boldsymbol{u}$.

II-D. Decoding

Rotating and projecting the received vector $\boldsymbol{z}$ with $\boldsymbol{G}_\mathrm{d}$ yields a discrete point $\boldsymbol{z}_\mathrm{d} = \boldsymbol{z}\boldsymbol{G}_\mathrm{d}$, which is used by the decoder to get an estimate $\hat{\boldsymbol{y}}_\mathrm{d}$ of the discrete lattice point $\boldsymbol{y}_\mathrm{d}$. Different concepts to derive this (discrete) estimate will be shown below. Afterwards, an estimation $\hat{\boldsymbol{u}}$ for the information word $\boldsymbol{u}$ is given by

$$\hat{\boldsymbol{u}} = \left(\boldsymbol{z} - 2m\left[\boldsymbol{0} \quad \hat{\boldsymbol{y}}_\mathrm{d} \cdot \boldsymbol{B}^{-1}\right]\right) \cdot \boldsymbol{A}^+, \tag{6}$$

where $\boldsymbol{A}^+ = \boldsymbol{A}^\mathrm{T} \cdot (\boldsymbol{A}\boldsymbol{A}^\mathrm{T})^{-1}$ is the pseudoinverse of the code matrix $\boldsymbol{A}$. The estimation (6) has the complexity $\mathcal{O}(N^2)$.

Here, the different decoding methods[b] from [7, 8] are summarized and analyzed in terms of their complexity.

1) The Discrete Maximum Likelihood (DML) Decoder detects the valid lattice point $\boldsymbol{y}_\mathrm{d}$ which is closest to $\boldsymbol{z}_\mathrm{d}$:

$$\hat{\boldsymbol{y}}_\mathrm{dDML} = \arg\min_{\boldsymbol{y}_\mathrm{d}} \|\boldsymbol{z}_\mathrm{d} - \boldsymbol{y}_\mathrm{d}\|. \tag{7}$$

This approach yields very good results [8] compared to the other methods presented below, but it is computationally complex.

The discrete part $\boldsymbol{z}_\mathrm{d}$ of the received code vector has to be compared to each possible valid lattice point. Therefore, the computing time for (7) increases with the number of valid lattice points. This leads to the complexity $\mathcal{O}(D \cdot l_\mathrm{DML})$, where l_DML denotes the number of valid lattice points and D results from the dimension of the vectors.

Furthermore, all valid lattice points have to be calculated once before the transmission. A low-complexity approach for calculating the valid lattice points is presented in Section III.

2) The Zero Forcing (ZFLR) Decoder with lattice reduction[c] utilizes the lattice structure of the discrete part. An approximation of the transmitted lattice point can be found by using the reduced lattice base matrix $\boldsymbol{L}$:

$$\hat{\boldsymbol{y}}_\mathrm{dZFLR} = \left\lceil \boldsymbol{z}_\mathrm{d} \cdot \boldsymbol{L}^{-1} \right\rfloor \cdot \boldsymbol{L}. \tag{8}$$

As shown in (3), all lattice points have integer values in $\tilde{\boldsymbol{s}}'$, hence, the rounding operation $\lceil \cdot \rfloor$ in (8) is used.

However, the decision regions of this decoder are parallelotopes instead of true Voronoi regions [12, 13] (see Fig. 4), and invalid lattice points outside of the modulo cube can be selected.

This approach has a very low complexity of $\mathcal{O}(D^2)$. Thus, the decoder complexity (including (6) with $\mathcal{O}(N^2)$) only depends on the dimensions of the code matrix $\boldsymbol{A}$.

3) The Lattice Maximum Likelihood (LML) Decoder is a combination of the ZFLR and DML decoder [8]. The decoded lattice point $\hat{\boldsymbol{y}}_\mathrm{dZFLR}$ from (8) is used as a first approximation. Then, the estimation offset

$$\boldsymbol{e} = \boldsymbol{z}_\mathrm{d} - \hat{\boldsymbol{y}}_\mathrm{dZFLR} \tag{9}$$

is calculated, and different candidates for the final ML estimation are chosen with respect to the orientation of $\boldsymbol{e}$ (cf. Fig. 4).

[b] The Minimum Mean Square Error (MMSE) decoder is not considered here, as knowledge about the channel is required and an M-dimensional integral has to be solved for each symbol ($\hat{\boldsymbol{u}} = \mathrm{E}\{\boldsymbol{u}|\boldsymbol{z}\} = \int \boldsymbol{u} \cdot \mathrm{p}(\boldsymbol{u}|\boldsymbol{z})\ \mathrm{d}\boldsymbol{u}$, from [7]), which makes the decoder very complex.

[c] While its complexity is nearly the same, the signal quality of the Zero Forcing (ZF) decoder *without* lattice reduction from [7] is much worse than that of the ZFLR decoder. Thus, the ZF decoder is not considered here.

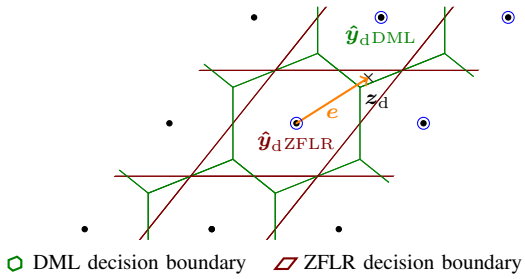


Figure 4. DML and ZFLR decision regions with marked candidates $\mathcal{C}$ ○.

Finally, only $l_{\mathrm{LML}} = 2^D$ candidates $\mathcal{C}$ have to be considered [8] for the ML estimation:

$$\hat{\boldsymbol{y}}_{\mathrm{d}\,\mathrm{LML}} = \underset{\boldsymbol{y}_{\mathrm{d}} \in \mathcal{C}}{\arg\min} \left\| \boldsymbol{z}_{\mathrm{d}} - \boldsymbol{y}_{\mathrm{d}} \right\|. \tag{10}$$

The generation of each candidate in $\mathcal{C}$ needs $\mathcal{O}(D^2)$ operations, as up to D (D-dimensional) base vectors are summed up. Therefore, the complexity is $\mathcal{O}(D^2 \cdot l_{\mathrm{LML}})$. Typically, this complexity is lower than the DML complexity, because for small base vectors (i.e., large coding gains) there are more than 2 lattice points per dimension D and $l_{\mathrm{DML}} > 2^D = l_{\mathrm{LML}}$.

4) Clipping and Truncation: By limiting the received code words to the valid range ($\pm m$) of the code words before decoding (clipping) and limiting the estimated $\hat{\boldsymbol{u}}$ to the input signal range (truncation), the LML decoder nearly achieves DML decoding precision [8]. Because of the lengths of the considered vectors, these methods need $\mathcal{O}(N)$ and $\mathcal{O}(M)$ operations, respectively. Both are applied to all measurements in this paper.

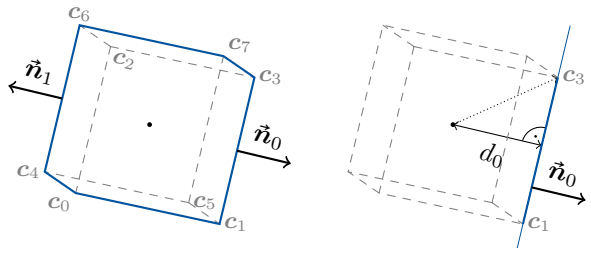
III. Valid Lattice Points

The DML decoder (Section II-D1) needs a set of all valid lattice points. In this section, a new approach for generating this set is presented and its complexity is analyzed. The first step is to generate a set of candidate lattice points that includes (at least) all valid points (Section III-D). Then, this set is limited to the rotated modulo cube, which can be described as a k-DOP (Discrete Oriented Polytope). A k-DOP is a polytope with k limiting (hyper-)planes which is usually used in computer graphics to calculate bounding boxes [14, 15]. It is described by a set of k normal vectors $\vec{\boldsymbol{n}}_i$ and distances d_i (Fig. 5).

III-A. Points in a k-DOP

A point $\boldsymbol{z}_{\mathrm{d}}$ is inside the k-DOP if it is closer to the center than all k limiting (hyper-)planes:

$$\vec{\boldsymbol{n}}_i \cdot \boldsymbol{z}_{\mathrm{d}} \le d_i \quad \forall i \in \{1, \ldots, k\}. \tag{11}$$



(a) k-DOP and normal vectors (b) Distance (projecting $\boldsymbol{c}_3$ on $\vec{\boldsymbol{n}}_0$)

Figure 5. (a) k-DOP —— for $\boldsymbol{A}_1 = [1\ 2.5\ 10.5]$ and (b) corresponding distance calculation: $d_0 = |\boldsymbol{c}_3 \cdot \vec{\boldsymbol{n}}_0|$, see Section III-C.

III-B. Generating k-DOP Normal Vectors

The normal vectors are orthogonal to the limiting (hyper-)planes of the modulo cube. Those planes are spanned by a subset of $D-1$ projected cartesian unit vectors (i.e., edges of the modulo cube). Thus, the D-dimensional normal vectors $\vec{\boldsymbol{n}}_i$ can be determined by

$$\vec{\boldsymbol{n}}_i = \ker(\mathcal{E}) \quad \text{for } i < k/2, \tag{12}$$

where $\mathcal{E} \in \mathbb{R}^{(D-1)\times D}$ is a "set" of $D-1$ edges. Here, the kernel $\ker(\mathcal{E})$ is a single vector orthogonal to all vectors in $\mathcal{E}$. Because of the symmetry of the modulo cube, there is an oppositely pointing vector

$$\vec{\boldsymbol{n}}_{k/2+i} = -\vec{\boldsymbol{n}}_i \tag{13}$$

for each normal vector $\vec{\boldsymbol{n}}_i$.

As there are $\binom{N}{D-1}$ possibilities to choose the set of edges and because of the symmetry, the k-DOP has

$$k = 2 \cdot \binom{N}{D-1} = 2 \cdot \frac{N!}{(D-1)! \cdot (M+1)!} \tag{14}$$

normal vectors $\vec{\boldsymbol{n}}_i$.

Calculating the kernel of a matrix can be done by calculating the singular value decomposition of a matrix, which has the complexity $\mathcal{O}(D^3)$ [16] or less for the $(D-1)\times D$ matrix $\mathcal{E}$. As this has to be done $\binom{N}{D-1}$ times, the computational complexity of this step is $\mathcal{O}\left(D^3 \cdot \binom{N}{D-1}\right)$.

III-C. Generating k-DOP Distances

To fully specify the k-DOP, the distances d_i that correspond to the normal vectors $\vec{\boldsymbol{n}}_i$ are needed. The 2^N projected corners $\boldsymbol{c}_j = [\pm m\ \ \pm m\ \ \ldots\ \ \pm m] \cdot \boldsymbol{G}_{\mathrm{d}}$ of the modulo cube are the furthest points from the origin, so some of them have to be located on the limiting (hyper-)planes of the k-DOP. Thus, the distances

$$d_i = \max_j |\boldsymbol{c}_j \cdot \vec{\boldsymbol{n}}_i| \tag{15}$$

are the maximum[d] of the inner product of the cube corners $\boldsymbol{c}_j$ and the normal vector $\vec{\boldsymbol{n}}_i$.

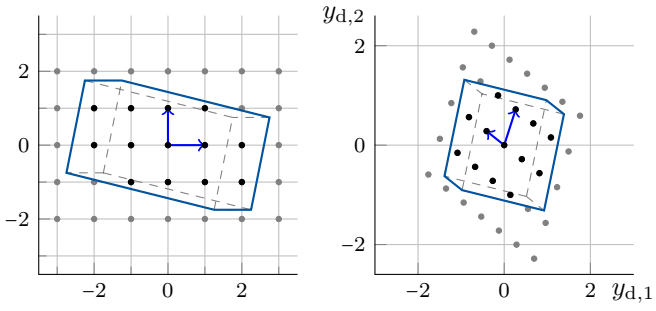
The total complexity of this step is $\mathcal{O}\left(D \cdot 2^N \cdot \binom{N}{D-1}\right)$.

III-D. Generating Valid Lattice Points

In order to generate all valid lattice points, the corners of the rotated modulo cube are transformed into the lattice integer range by multiplication with the inverse $\boldsymbol{L}^{-1}$ of the lattice base matrix: $\boldsymbol{c}'_j = [\pm 1\ \pm 1\ \ldots\ \pm 1] \cdot \boldsymbol{G}_{\mathrm{d}} \cdot \boldsymbol{L}^{-1}$. Afterwards, the minimum and maximum values of the cube corners are calculated for each dimension and rounded to the integer away from 0. The integer points in the range between the rounded cube corners (shown in Fig. 6a) are candidates $\tilde{\boldsymbol{s}}'$ for the set of valid lattice integers.

These candidates $\tilde{\boldsymbol{s}}'$ can be transformed to a set of lattice points candidates by a multiplication with $\boldsymbol{L}$ as in (3). Limiting these candidates to the k-DOP with (11) yields the set of valid lattice points shown in Fig. 6b.

[d] The symmetry $\vec{\boldsymbol{n}}_{k/2+i} \overset{(13)}{=} -\vec{\boldsymbol{n}}_i$ can be exploited. Only one of the distances has to be calculated, and $d_{k/2+i} = d_i$.



(a) Integer lattice (b) Lattice points

Figure 6. Candidate lattice points: valid • and invalid •. Base vectors ⟶ for reference, $\boldsymbol{A}_1$ as in Fig. 5.

If $l'_{\mathrm{DML}} \le \max_j \left\|2 + 2{\cdot}\boldsymbol{c}'_j\right\|_\infty^D$ denotes the number of candidates $\tilde{\boldsymbol{s}}'$, the computational complexity of this step is upper bounded by $\mathcal{O}\left(l'_{\mathrm{DML}}{\cdot}D{\cdot}\binom{N}{D-1}\right)$, as the inequality in (11) has to be evaluated at most $k = \binom{N}{D-1}$ times for each candidate. Usually, l'_{DML} is greater than D^3 and 2^N, so this is also the total complexity of the algorithm presented in this section.

However, the determination of the valid lattice points can be done offline, i.e., it does not have to be done for each received code word, but only once per code.

IV. LML Radius Pre-Check for Complexity Reduction

In Fig. 7, a comparison of the decoding times is presented. The simulations in this work were run on a standard office computer, with a focus on the relative differences between the methods rather than the absolute simulation run time in seconds. For each point in the plots, 1000 simulations were conducted (with 10^5 symbols each), and the minimum of the run time measurements is displayed in order to exclude influences from other processes, e.g., the operating system.

For few discrete dimensions D, the reduced complexity of the LML decoder compared to the DML decoder is recognizable. For larger D, the LML decoding time increases fast due to the exponential dependency on the number of candidates for the ML estimation. The LML decoder considers points that are outside the modulo cube. A method to reduce the LML

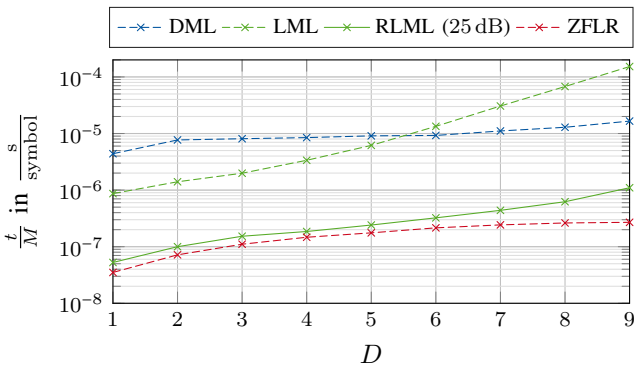


Figure 7. Example: Decoding time per symbol for different decoders with $\boldsymbol{A} = [1\ a\ a^2\ \ldots\ a^D]$, $a = 1.75$.

complexity for good channels, the *radius pre-check* [8], is analyzed in the following sections. An LML decoder which applies this method is called RLML here. For $CSNR = 25\,\mathrm{dB}$ and the codes from Fig. 7, it is much faster than the DML and LML decoders.

IV-A. Common Decision Region

The Maximum Likelihood decision regions share a large common area with the ZFLR decision regions, where both decoders map on the same point (see also Fig. 10). The radius r_{c} of the largest (hyper-)circle in this common region is a property of the code. If

$$\left\|\boldsymbol{z}_{\mathrm{d}} - \hat{\boldsymbol{y}}_{\mathrm{dZFLR}}\right\| \overset{(9)}{=} \|\boldsymbol{e}\| < r_{\mathrm{c}} \tag{16}$$

holds for the estimation offset $\boldsymbol{e}$ of a received code word, an additional ML estimation among the candidates is unnecessary, as $\hat{\boldsymbol{y}}_{\mathrm{dZFLR}} = \hat{\boldsymbol{y}}_{\mathrm{dLML}}$ in that case. Hence, the computational complexity decreases.

In order to estimate the decoding complexity of the LML decoding, the probability $P(\|\boldsymbol{e}\| \le r_{\mathrm{c}})$ that (16) holds has to be determined. However, to calculate this probability, integrals over the circles around all lattice points would have to be evaluated numerically. If the discrete part of the code word is decoded correctly, the estimation offset $\boldsymbol{e}$ equals the discrete dimensions of the channel noise:

$$\hat{\boldsymbol{y}}_{\mathrm{dZFLR}} = \boldsymbol{y}_{\mathrm{d}} \ \Rightarrow\ \underbrace{\boldsymbol{y}_{\mathrm{d}} + \boldsymbol{n}_{\mathrm{d}}}\ \boldsymbol{e} = \boldsymbol{z}_{\mathrm{d}} - \hat{\boldsymbol{y}}_{\mathrm{dZFLR}} = \boldsymbol{n}_{\mathrm{d}} = \boldsymbol{n}\cdot\boldsymbol{G}_{\mathrm{d}}. \tag{17}$$

Therefore, the probability $P(\|\boldsymbol{n}_{\mathrm{d}}\| \le r_{\mathrm{c}})$ is calculated in the next section.

IV-B. Noise Probability Distribution

The noise $\boldsymbol{n}$ is assumed to be Gaussian distributed with mean $\boldsymbol{0}$ and power σ_n^2. As all entries of the vector are uncorrelated (white noise), the probability density of $\boldsymbol{n}_{\mathrm{d}}$ is

$$p(\boldsymbol{n}_{\mathrm{d}}) = \frac{1}{\sqrt{\left(2\pi\sigma_{n_{\mathrm{d}}}^2\right)^D}} \cdot \mathrm{e}^{-\frac{1}{2}\sigma_{n_{\mathrm{d}}}^{-2}\cdot\|\boldsymbol{n}_{\mathrm{d}}\|^2}, \tag{18}$$

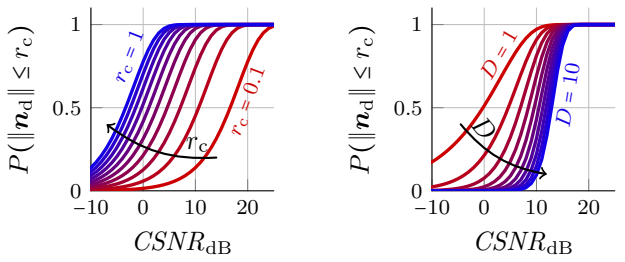
where the power of the discrete noise $\boldsymbol{n}_{\mathrm{d}}$ is $\sigma_{n_{\mathrm{d}}}^2 = \sigma_n^2$.

By integrating the surface $S_D(r) = 2\pi^{D/2}\cdot r^{D-1}\cdot\Gamma^{-1}\left(\frac{D}{2}\right)$ of a D-dimensional hypersphere (weighted with the probability) up to radius $r = r_{\mathrm{c}}$, the distribution function of the magnitude of the discrete noise $\boldsymbol{n}_{\mathrm{d}}$ can be calculated:

$$\begin{aligned} P(\|\boldsymbol{n}_{\mathrm{d}}\| \le r_{\mathrm{c}}) &= \int_{r=0}^{r_{\mathrm{c}}} S_D(r)\cdot p(\|\boldsymbol{n}_{\mathrm{d}}\| = r)\,\mathrm{d}r. \\ &= \frac{2}{\Gamma(\frac{D}{2})}\cdot\frac{1}{\sqrt{(2\sigma_{n_{\mathrm{d}}}^2)^D}}\cdot\int_{r=0}^{r_{\mathrm{c}}} \mathrm{e}^{-\frac{1}{2}\sigma_{n_{\mathrm{d}}}^{-2}\cdot r^2}\cdot r^{D-1}\,\mathrm{d}r. \end{aligned} \tag{19}$$

This integral can be computed numerically as a function of D, r_{c}, and $\sigma_{n_{\mathrm{d}}}^2$ (or $CSNR \overset{(4)}{=} \frac{\sigma_y^2}{\sigma_n^2} \approx \frac{1}{3\sigma_{n_{\mathrm{d}}}^2}$, because in this paper we consider uniformly distributed $\boldsymbol{u}$ with $\sigma_y^2 \overset{[7]}{\approx} \sigma_u^2 = \frac{1}{3}$).

Fig. 8a shows the distribution function (19) for constant D and varied radius r_{c}. For codes with a smaller radius r_{c}, good channels (high $CSNR$) are needed to get a relevant probability



(a) $D = 2$, $r_\mathrm{c} = 0.1, ..., 1.0$ (b) $r_\mathrm{c} = 0.4$, $D = 1, ..., 10$

Figure 8. The distribution function $P(\|\boldsymbol{n}_\mathrm{d}\| \le r_\mathrm{c})$ for a constant number of discrete dimensions D (a) and a constant radius r_c (b).

for receiving a code word inside the hypersphere. In contrast, high probabilities for avoiding the additional ML estimation even for bad channels are achieved by codes with a large radius r_c.

In Fig. 8b, the distribution function (19) is shown for a constant radius r_c and a varied number of discrete dimensions D. For higher-dimensional codes, the probability $P(\|\boldsymbol{n}_\mathrm{d}\| \le r_\mathrm{c})$ decreases.

Fig. 9 shows the calculated probability $P(\|\boldsymbol{n}_\mathrm{d}\| \le r_\mathrm{c})$ in contrast to the simulated probability $P(\|\boldsymbol{e}\| \le r_\mathrm{c})$ for three different AMB codes as a function of $CSNR$.

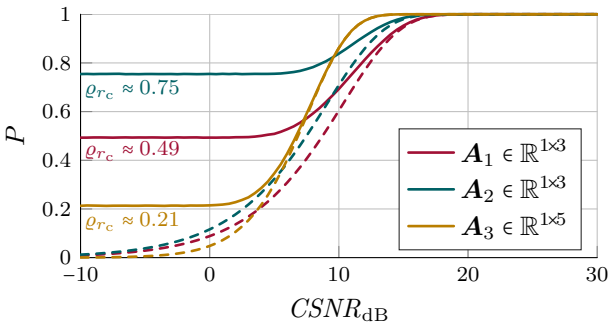


Figure 9. Probabilities $P(\|\boldsymbol{n}_\mathrm{d}\| \le r_\mathrm{c})$ - - - and $P(\|\boldsymbol{e}\| \le r_\mathrm{c})$ —— (measured) for different codes: $\boldsymbol{A}_1 = [1\ 2.5\ 10.5]$, $\boldsymbol{A}_2 = [1\ a\ a^2]$ with $a = 3.29$, and $\boldsymbol{A}_3 = [1\ a\ a^2\ a^3\ a^4]$ with $a = 1.755$. All codes have a similar maximum coding gain. The value ϱ_{r_c} is explained in Section IV-C.

If the noise power is large, it is possible that a received code word is moved into another (neighboring) decision region (Fig. 10). In this case, the assumption $\boldsymbol{e} \overset{(17)}{=} \boldsymbol{n}_\mathrm{d}$ does not hold due to the fact that the ZFLR estimation is wrong. Hence,

$$P(\|\boldsymbol{e}\| \le r_\mathrm{c}) \ge P(\|\boldsymbol{n}_\mathrm{d}\| \le r_\mathrm{c}) \tag{20}$$

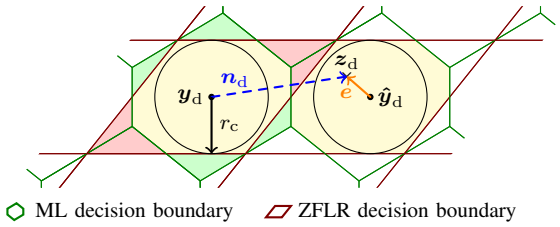


Figure 10. Erroneous decoding. The noise $\boldsymbol{n}_\mathrm{d}$ is large, so the code word is received in a neighboring decision region: $\boldsymbol{e} \neq \boldsymbol{n}_\mathrm{d}$.

and it holds

$$P(\|\boldsymbol{e}\| \le r_\mathrm{c}) = \varrho_{r_\mathrm{c}} \quad \text{for} \quad CSNR \to 0, \text{ i.e., } \|\boldsymbol{n}_\mathrm{d}\| \to \infty, \tag{21}$$

where ϱ_{r_c} is the ratio of a hypersphere with radius r_c and the volume of a decision region, which will be examined in the next section.

IV-C. Ratio Between Volumes of r_c-Hypersphere and Decision Regions

The *volume ratio*

$$\varrho_{r_\mathrm{c}} = \frac{V_D(r_\mathrm{c})}{|\det \boldsymbol{L}|} \tag{22}$$

is the fraction of the volume $V_D(r_\mathrm{c}) = r_\mathrm{c}^D \cdot \frac{\pi^{D/2}}{\Gamma(D/2+1)}$ of the D-dimensional hypersphere and the volume of a decision region[e] $|\det \boldsymbol{L}|$. A high volume ratio ϱ_{r_c} yields a high probability that the estimation offset $\boldsymbol{e}$ is inside the common region with radius r_c even for noisy channels.

A lower bound can be calculated for the volume of the ZFLR decision region, that is given by a D-dimensional parallelotope. The radius r_c is the shortest distance between a lattice point and one of the decision boundaries. Hence, $2r_\mathrm{c}$ is the shortest distance between two parallel decision boundaries and thus the minimal possible height of the parallelotope. Then, the parallelotope volume is at least as large as the volume of a cube with edge length $2r_\mathrm{c}$, which is thus a lower bound for the Voronoi volume

$$|\det(\boldsymbol{L})| \ge (2r_\mathrm{c})^D. \tag{23}$$

Thereby, an upper bound for (22) is given by

$$\varrho_{r_\mathrm{c}} \le \frac{V_D(r_\mathrm{c})}{(2r_\mathrm{c})^D} = \frac{\pi^{\frac{D}{2}}}{2^D \cdot \Gamma(D/2+1)}. \tag{24}$$

This upper bound is shown in Fig. 11 as a function of the discrete code dimensions D. Furthermore, the volume ratios of different randomly generated $M\times(M+D)$ codes are displayed.

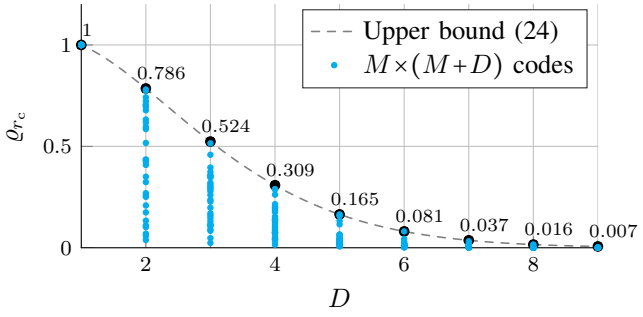


Figure 11. Volume ratio ϱ_{r_c} as a function of discrete Dimensions D: Upper bound - ● - and values • for randomly generated $M\times(M+D)$ codes (30 per value of D).

As the volume ratio decreases with the number of discrete dimensions, the probability of receiving a highly disturbed code word inside the area with radius r_c is small for high dimensional codes (larger D) and the additional ML estimation is frequently needed for channels with a low $CSNR$.

[e] The volume of a ZFLR decision region and the Voronoi volume of an LML decision region are identical.

V. Decoding Time Measurements

Fig. 12 shows simulation results comparing the decoding time of the LML decoder with radius pre-check (RLML) for three different AMB codes. Like in Fig. 7, the focus of the simulations is on the relative time differences.

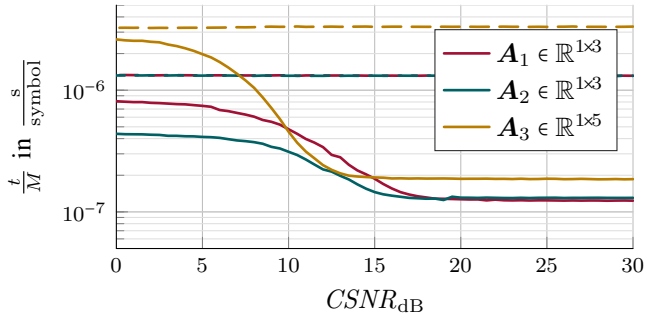


Figure 12. LML decoding times with (——) and without (— —) radius pre-check for three different codes.

In the low-$CSNR$ region all codes have (nearly) constant decoding times due to the (nearly) constant probability $P(\|\boldsymbol{e}\| \leq r_\text{c}) \approx \varrho_{r_\text{c}}$ (see Fig. 9).

Due to the increasing probability of receiving a code word in the common region, the decoding times decrease for increasing $CSNR$. For very good channels (nearly) all code words are inside the hypersphere with radius r_c and only the ZFLR estimation is needed.

Fig. 13 shows a comparison of decoding times for all presented decoders. As expected, all decoding times except the LML decoding time with radius pre-check are independent of the $CSNR$. The LML decoder offers a faster decoding than the DML decoder. Furthermore, the radius pre-check decreases the decoding time significantly, so that ZFLR decoding time is nearly achieved for good channels. In this example, the decoding of the RLML decoder is about 10 to 80 times faster than that of the DML decoder.

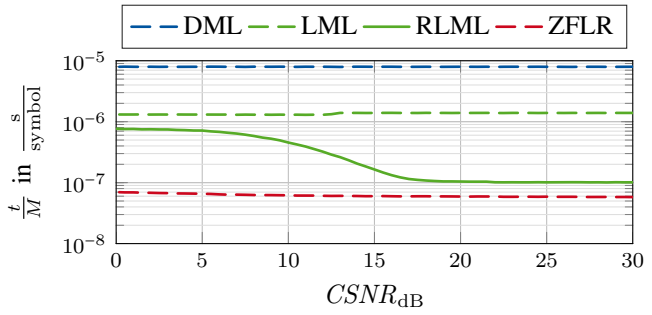


Figure 13. Decoding times for different decoders, $\boldsymbol{A}_1 = [1\ 2.5\ 10.5]$.

VI. Conclusion

In this paper, the complexity of Analog Modulo Block Code encoding and decoding was presented.

A novel low-complexity approach for calculating all valid lattice points for the use with the DML decoder was introduced in Section III.

The complexity reduction by radius pre-check was analyzed with respect to the channel noise and the ratio ϱ_{r_c} of the volumes of the hypersphere and the Voronoi size $|\det \boldsymbol{L}|$. A larger volume ratio ϱ_{r_c} enables a faster decoding due to the frequently skipped ML decoding step. It was shown that, however, a large ratio ϱ_{r_c} cannot be achieved for a large number D of discrete dimensions.

Time measurements in Section V exemplify the impact of the radius pre-check on the decoding time, which is especially beneficial for medium to good channels. The LML decoder combines the decoding precision of the DML decoder [8] and the low complexity (decoding time) of the ZFLR decoder.

References

[1] M. Skoglund, N. Phamdo, and F. Alajaji, "Design and performance of VQ-based hybrid digital-analog joint source-channel codes," *IEEE Transactions on Information Theory*, vol. 48, no. 3, pp. 708–720, Mar. 2002.

[2] U. Mittal and N. Phamdo, "Hybrid digital-analog (HDA) joint source-channel codes for broadcasting and robust communications," *IEEE Transactions on Information Theory*, vol. 48, no. 5, pp. 1082–1102, May 2002.

[3] M. Rüngeler and P. Vary, "Surpassing Purely Digital Transmission: A Simplified Design of Hybrid Digital Analog Codes," in *9th International ITG Conference on Systems, Communications and Coding (SCC'2013)*, Munich, Germany, Jan. 2013.

[4] M. Rüngeler, J. Bunte, and P. Vary, "Design and Evaluation of Hybrid Digital-Analog Transmission Outperforming Purely Digital Concepts," *IEEE Transactions on Communications*, vol. 62, no. 11, pp. 3983–3996, Nov. 2014.

[5] E. Akyol, K. B. Viswanatha, K. Rose, and T. A. Ramstad, "On Zero-Delay Source-Channel Coding," *IEEE Transactions on Information Theory*, vol. 60, no. 12, pp. 7473–7489, Dec. 2014.

[6] M. Rüngeler, B. Schotsch, and P. Vary, "Properties and Performance Bounds of Linear Analog Block Codes," in *Conference Record of Asilomar Conference on Signals, Systems, and Computers (ACSSC)*. IEEE, Nov. 2009, pp. 962–966.

[7] T. Schmitz, M. Rüngeler, and P. Vary, "Analysis of Analog Modulo Block Codes," in *10th International ITG Conference on Systems, Communications and Coding (SCC'2015)*, Hamburg, Germany, Feb. 2015, available at IEEE Xplore.

[8] T. Schmitz, P. Jax, and P. Vary, "Improved Decoding of Analog Modulo Block Codes for Noise Mitigation," in *IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP)*. Shanghai, China: IEEE, Mar. 2016, pp. 3861–3865.

[9] D. E. Knuth, "Big Omicron and Big Omega and Big Theta," *SIGACT News*, vol. 8, no. 2, pp. 18–24, Apr. 1976.

[10] A. K. Lenstra, H. W. Lenstra Jr., and L. Lovász, "Factoring polynomials with rational coefficients," *Mathematische Annalen*, vol. 261, no. 4, pp. 515–534, 1982.

[11] D. Wübben, D. Seethaler, J. Jaldén, and G. Matz, "Lattice Reduction: A Survey with Applications in Wireless Communications," *IEEE Signal Processing Magazine*, vol. 28, no. 3, pp. 70–91, May 2011.

[12] J. H. Conway and N. J. A. Sloane, "Voronoi regions of lattices, second moments of polytopes, and quantization," *IEEE Transactions on Information Theory*, vol. 28, no. 2, pp. 211–226, Mar. 1982.

[13] G. Voronoi, "Nouvelles applications des paramètres continus à la théorie des formes quadratiques. Deuxième mémoire. Recherches sur les parallélloèdres primitifs." *Journal für die reine und angewandte Mathematik*, vol. 134, pp. 198–287, 1908.

[14] T. L. Kay and J. T. Kajiya, "Ray Tracing Complex Scenes," *SIGGRAPH Computer Graphics*, vol. 20, no. 4, pp. 269–278, Aug. 1986.

[15] J. T. Klosowski, M. Held, J. S. B. Mitchell, H. Sowizral, and K. Zikan, "Efficient collision detection using bounding volume hierarchies of k-DOPs," *IEEE Transactions on Visualization and Computer Graphics*, vol. 4, no. 1, pp. 21–36, Jan. 1998.

[16] M. Holmes, A. Gray, and C. Isbell, "Fast SVD for large-scale matrices," in *Workshop on Efficient Machine Learning at NIPS*, vol. 58, 2007, pp. 249–252.