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 of AMB codes [7,8]. Additionally, the asymptotic complexity of the algorithms is given in big O notation [9].

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

II-A. Encoding

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

$$smod_m(x) = ((x + m) \mod 2m) - m \quad \text{for } x \in \mathbb{R}$$

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 $y$ can be expressed by

$$y = smod_m(u \cdot A).$$

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

Fig. 2. Modulo function [8].
As the value of \( m \) does not affect the performance of an Analog Modulo Block Code (AMB code) if \( A \) is scaled accordingly, \( m = 1 \) is assumed for all codes in this paper. We only use systematic AMB coding with \( A = [1 \ A] \). Due to the \( M \times M \) identity matrix \( I \), the code words contain the information words. Thus, for the encoding, only \( O(M \cdot D) \) operations are needed, where \( D = N - M \) denotes the number of parity symbols added by the generator matrix \( A \).

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 \( \mathbb{R}^N \).

![Code words](image)

Figure 3. Valid code words \( y \) with \( A = [1 \ 4] \) 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 \( G_d \) (derivable from the code matrix \( A \), 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 \( y_d \) of the valid code words form a subset of a lattice (see Fig. 3b) with base matrix \( L \):

\[
y_d = y \cdot G_d = \tilde{s} \cdot B = \tilde{s}' \cdot L \quad \text{with} \quad \tilde{s}, \tilde{s}' \in \mathbb{Z}^D.
\]

The base matrix \( B \) is a submatrix of \( 2mG_d \) and can be reduced \([10, 11]\) to a representation \( L \) with preferably shorter base vectors \([7]\).

II-C. Transmission

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

\[
CSNR = \frac{E\{ y^2 \}}{E\{n^2 \}} = \frac{N \cdot \sigma_y^2}{N \cdot \sigma_n^2} = \frac{\sigma_y^2}{\sigma_n^2},
\]

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

\[
z = y + n,
\]

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

II-D. Decoding

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

\[
\tilde{u} = (z - 2m[0 \ y_d \cdot B^{-1}]) \cdot A^T,
\]

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

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

The valid lattice point \( y_d \) which is closest to \( z_d \):

\[
\hat{y}_{d\text{DML}} = \arg\min_{y_d} \|z_d - y_d\|.
\]

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

The discrete part \( z_d \) 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 \( O(D \cdot l_{\text{DML}}) \), where \( l_{\text{DML}} \) 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

Utilizing the lattice structure of the discrete part. An approximation of the transmitted lattice point can be found by using the reduced lattice base matrix \( L \):

\[
\hat{y}_{d\text{ZFLR}} = \left[ z_d \cdot L^{-1} \right] \cdot L.
\]

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

However, the decision regions of this decoder are paralleloptopes 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 \( O(D^2) \). Thus, the decoder complexity (including (6) with \( O(N^2) \)) only depends on the dimensions of the code matrix \( A \).

3) The Lattice Maximum Likelihood (LML) Decoder

is a combination of the ZFLR and DML decoder \([8]\). The decoded lattice point \( \hat{y}_{d\text{ZFLR}} \) from (8) is used as a first approximation. Then, the estimation offset

\[
e = z_d - \hat{y}_{d\text{ZFLR}}
\]

is calculated, and different candidates for the final ML estimation are chosen with respect to the orientation of \( e \) (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 \( (\hat{u} = E\{u|z\} = \int f \cdot p(u|z) \, du, \text{from } [7]) \), 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.
Finally, only \( l_{\text{LML}} = 2^D \) candidates \( C \) have to be considered [8] for the ML estimation:

\[
\hat{y}_{\text{dLML}} = \arg\min_{y_d \in C} \| z_d - y_d \|.
\]

(10)

The generation of each candidate in \( C \) needs \( \mathcal{O}(D^2) \) operations, as up to \( D \) (\( D \)-dimensional) base vectors are summed up. Therefore, the complexity is \( \mathcal{O}(D^2 : l_{\text{LML}}) \). 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 \( l_{\text{DML}} > 2^D = l_{\text{LML}} \).

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 \( \hat{u} \) 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 \( \hat{n}_i \) and distances \( d_i \) (Fig. 5).

III-A. Points in a k-DOP

A point \( z_d \) is inside the k-DOP if it is closer to the center than all \( k \) limiting (hyper-)planes:

\[
\hat{n}_i \cdot z_d \leq d_i \quad \forall i \in \{1, \ldots, k\}.
\]

(11)

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 \( \hat{n}_i \) can be determined by

\[
\hat{n}_i = \ker(\mathcal{E}) \quad \text{for } i < k/2,
\]

(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

\[
\hat{n}_{k/2+i} = -\hat{n}_i
\]

(13)

for each normal vector \( \hat{n}_i \).

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

\[
k = 2 \left( \frac{N}{D - 1} \right) = 2 \frac{N!}{(D - 1)! (M + 1)!}
\]

(14)

normal vectors \( \hat{n}_i \). 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 \( \mathcal{O}\left(D^3 \binom{N}{D-1}\right) \).

III-C. Generating k-DOP Distances

To fully specify the k-DOP, the distances \( d_i \) that correspond to the normal vectors \( \hat{n}_i \) are needed. The \( 2^N \) projected corners \( c_j = [\pm m \pm m \ldots \pm m] G_d \) 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

\[
d_i = \max_j |c_j \cdot \hat{n}_i|
\]

(15)

are the maximum \(^d\) of the inner product of the cube corners \( c_j \) and the normal vector \( \hat{n}_i \).

The total complexity of this step is \( \mathcal{O}(D \cdot 2^N \cdot \binom{N}{D-1}) \).
If \( t'_{\text{DML}} \leq \max_j \| 2 + 2 \cdot c'_j \|_D \) denotes the number of candidates \( s'_j \), the computational complexity of this step is upper bounded by \( \mathcal{O}(t'_{\text{DML}} \cdot D^{D-(D-1)}) \), as the inequality in (11) has to be evaluated at most \( k = \binom{N}{D-1} \) times for each candidate. Usually, \( t'_{\text{DML}} \) 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 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 \( \text{CSNR} = 25 \) 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

\[
\| z_d - \hat{y}_{d_{\text{ZFLR}}} \| = \min \| e \| < r_c
\]

holds for the estimation offset \( e \) of a received code word, an additional ML estimation among the candidates is unnecessary, as \( \hat{y}_{d_{\text{ZFLR}}} = \hat{y}_{d_{\text{LML}}} \) in that case. Hence, the computational complexity decreases.

In order to estimate the decoding complexity of the LML decoding, the probability \( P(\| e \| \leq r_c) \) 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 \( e \) equals the discrete dimensions of the channel noise:

\[
\hat{y}_{d_{\text{ZFLR}}} = y_d \implies e = z_d - \hat{y}_{d_{\text{ZFLR}}} = n_d = n \cdot G_d.
\]

Therefore, the probability \( P(\| n_d \| \leq r_c) \) is calculated in the next section.

IV-B. Noise Probability Distribution

The noise \( n \) is assumed to be Gaussian distributed with mean 0 and power \( \sigma_n^2 \). As all entries of the vector are uncorrelated (white noise), the probability density of \( n_d \) is

\[
p(n_d) = \frac{1}{\sqrt{(2\pi \sigma_n^2)^D}} e^{-\frac{1}{2} \sigma_n^2 \| n_d \|_2^2}.
\]

where the power of the discrete noise \( n_d \) is \( \sigma_{n_d}^2 = \sigma_n^2 \).

By integrating the surface \( S_D(r) = 2\pi^{D/2} \cdot r^{D-1} \cdot \Gamma^{-1}(\frac{D}{2}) \) of a \( D \)-dimensional hypersphere (weighted with the probability) up to radius \( r = r_c \), the distribution function of the magnitude of the discrete noise \( n_d \) can be calculated:

\[
P(\| n_d \| \leq r_c) = \int_{r=0}^{r_c} S_D(r) \cdot p(\| n_d \| = r) \, dr.
\]

This integral can be computed numerically as a function of \( D \), \( r_c \), and \( \sigma_{n_d}^2 \) (or \( \text{CSNR} = \frac{1}{2} \sigma_n^2 = \sigma_{n_d}^2 \approx \frac{1}{3} \)).

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 \( \text{CSNR} \)) are needed to get a relevant probability
and it holds
\[ P(\|e\| \leq r_c) = \varrho_{r_c} \quad \text{for} \quad \text{CSNR} \rightarrow 0, \ i.e., \ \|n_d\| \rightarrow \infty, \ (21) \]
where \( \varrho_{r_c} \) 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**
\[ \varrho_{r_c} = \frac{V_D(r_c)}{|\det L|} \quad (22) \]
is the fraction of the volume \( V_D(r_c) = r_c^D \cdot \frac{\pi^{D/2}}{\Gamma(D/2 + 1)} \) of the \( D \)-dimensional hypersphere and the volume of a decision region \( |\det L| \). A high volume ratio \( \varrho_{r_c} \) yields a high probability that the estimation offset \( e \) 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, \( 2r_c \) 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 \( 2r_c \), which is thus a lower bound for the Voronoi volume
\[ |\det(L)| \geq (2r_c)^D. \quad (23) \]

Thereby, an upper bound for (22) is given by
\[ \varrho_{r_c} \leq \frac{V_D(r_c)}{(2r_c)^D} = \frac{\pi^{D/2}}{2^D \cdot \Gamma(D/2 + 1)}. \quad (24) \]
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.

\[ - - - \text{Upper bound} \ (24) \quad \& \quad M \times (M + D) \text{ codes} \]
\[ D \]
\[ 0.0786 \quad 0.0524 \quad 0.0309 \quad 0.0165 \quad 0.0081 \quad 0.0037 \quad 0.0016 \quad 0.0007 \]

Figure 11. Volume ratio \( \varrho_{r_c} \) 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.

\( ^5 \)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.

The complexity reduction by radius pre-check was analyzed with respect to the channel noise and the ratio $\varrho_r$, of the volumes of the hypersphere and the Voronoi size $|\det L|$. A larger volume ratio $\varrho_r$ enables a faster decoding due to the frequently skipped ML decoding step. It was shown that, however, a large ratio $\varrho_r$ 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.

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.

REFERENCES