Multiple-Symbol Differential Sphere Decoding

Lutz Lampe, Robert Schober
Dept. of Electrical and Computer Engineering
The University of British Columbia
Canada

Volker Pauli, Christoph Windpassinger
Lehrstuhl für Informationsübertragung
Universität Erlangen-Nürnberg
Germany

Abstract — In multiple-symbol differential detection (MSDD) for power-efficient transmission over fading channels without channel state information, blocks of \( N \) received symbols are jointly processed to decide on \( N-1 \) data symbols. The search space for the maximum-likelihood (ML) estimate is therefore (complex) \( (N-1) \)-dimensional, and MSDD quickly becomes computationally intractable as \( N \) grows. Sphere decoding (SD) is a widely used approach to find the ML estimate in such high-dimensional spaces. In this paper, we devise the application of SD to accomplish MSDD, and we refer to the resulting technique as multiple-symbol differential sphere decoding (MSDSD). We present an efficient algorithm for MSDSD, whose excellent performance versus complexity trade-off is verified by various simulation results and by comparisons with other, suboptimum approaches known from the literature.
1 Introduction

Maximum-likelihood (ML) decoding (or detection) algorithms with polynomial expected complexity, collectively referred to as sphere decoding (SD) algorithms, have recently attracted considerable attention in multiple-input multiple-output (MIMO) communication problems, cf. e.g. [1]-[6]. In this context, the problem usually considered is to find the solution

$$\hat{s} = \text{argmin}_{s} ||r - Hs||^2$$

for an $N_s$-dimensional input vector $s$ transmitted over an $(N_r \times N_s)$-dimensional channel $H$ ($N_r \geq N_s$) with $N_r$-dimensional output vector $r$ ($|| \cdot ||$: Euclidean norm). Given that the channel $H$ is known to the receiver, (1) can be rewritten as [6]

$$\hat{s} = \text{argmin}_{s} ||U(s - s_{LS})||^2,$$

where the Cholesky factorization $H^H H = U^H U$ with upper triangular matrix $U$ and the unconstrained least-squares solution $s_{LS} \in \mathbb{C}^{N_s}$ are used ($^H$: Hermitian transpose).

In this paper, we consider the application of SD to multiple-symbol differential detection (MSDD) of differential phase-shift keying (DPSK) over an unknown frequency-nonselective (flat) fading channel. In MSDD, $N$ consecutively received samples are collected for joint detection of $N - 1$ consecutively transmitted data symbols, cf. e.g. [7, 8]. Due to this block processing structure, we can model transmission with MSDD as MIMO system with $N_s = N - 1$, $N_r = N$. It is known that for optimum MSDD with respect to the ML criterion (ML-MSDD) error-rate performance steadily improves with increasing dimension or observation window size $N$. On the other hand, detection complexity grows exponentially with $N$. In order to significantly reduce this detection complexity while maintaining ML performance, we propose to apply SD to MSDD, which we refer to as multiple-symbol differential sphere decoding (MSDSD)$^2$. MSDSD is motivated by the observation that by Cholesky factorization of the channel statistics, a decision metric of the same form as (2) can be obtained. Hence, we can quite straightforwardly extend SD strategies originally designed for (1), i.e., for MIMO detection with channel state information (CSI) at the receiver, to MSDD without CSI.

---

1 Since we use the equivalent complex baseband description, in general all vector and matrix components are complex-valued.

2 In the context of this paper, the terms “decoding” and “detection” refer to the same procedure and are used interchangeably. For “MSDSD” we use the term “decoding” since it is preferred in the SD literature.
In the recent past, several low-complexity implementations for MSDD have been proposed. Mackenthun [9] developed an algorithm with complexity of the order $N \log_2(N)$ per $N$ symbol block. This algorithm is only equivalent to ML-MSDD for channels that are time-invariant over an $N$ symbol period. For the general time-varying flat fading case, linear-prediction based decision-feedback differential detection (DF-DD) receivers have been proposed, cf. e.g. [10, 11, 12]. Although DF-DD does not achieve the performance of ML-MSDD, it offers substantial performance improvements over conventional differential detection (CDD) with $N = 2$ while its per-symbol complexity is only linear in $N$. Other suboptimum *ad hoc* algorithms use a two-step strategy, e.g. [13, 14, 15]. First, a candidate list is set up by using CDD with $N = 2$, and then these candidates are tested using the MSDD metric with increased window size. Again, these detection schemes cannot guarantee that the ML decision is found. This is in contrast to the proposed MSDSD, which in fact performs true ML-MSDD, but keeps detection complexity reasonably low by the application of SD.

This paper is organized as follows. Section 2 briefly introduces the system model for DPSK transmission with MSDD. In Section 3, the new detection technique is derived and the MSDSD algorithm is explained in detail. Simulation results illustrating the favorable performance and complexity properties of MSDSD are presented in Section 4. Finally, Section 5 concludes the paper.

2 System Model

The block diagram of our discrete-time communication system in the equivalent complex low-pass domain is depicted in Figure 1. At the transmitter, we apply the classical $M$-ary DPSK [16]. $\log_2(M)$ binary data symbols $c[\mu]$ are Gray mapped to $M$-ary data-carrying differential symbols $v[k]$ taken from the $M$PSK signal constellation $\mathcal{V} \triangleq \{v = e^{j2\pi m/M} \mid m = 0, 1, \ldots, M - 1\}$ ($\mu, k \in \mathbb{Z}$: binary and $M$-ary symbol discrete-time index, respectively). From $v[k]$ the current transmit symbol $s[k]$ is obtained via differential

![Block diagram of differential transmission and MSDD.](image-url)
The transmission channel is assumed frequency-nonselective and slowly time-varying during one modulation interval of length $T$ and is represented by the complex fading gain $h[k]$. The received sample $r[k]$ can be modelled as

$$r[k] = h[k]s[k] + n[k],$$

where $n[k]$ denotes additive white Gaussian noise (AWGN) with variance $\sigma_n^2$. We consider $h[k]$ to be zero-mean complex Gaussian distributed (Rayleigh fading) with autocorrelation function

$$\varphi_{hh}[\kappa] \triangleq \mathcal{E} \{ h[k + \kappa] h^*[k] \} = J_0(2\pi B_f T \kappa)$$

($\mathcal{E}$: expectation, $J_0(\cdot)$: zeroth order Bessel function of the first kind) according to the widely used Clarke fading model with maximum normalized fading bandwidth $B_f T$.

At the receiver, the multiple-symbol differential sphere decoder processes $N$ consecutively received samples

$$r[k_N] \triangleq [r[Nk - (N - 1)] r[Nk - (N - 2)] \ldots r[Nk]]^T$$

($k_N$: vector symbol discrete-time index) to obtain ML estimates $\hat{s}[k]$ of the corresponding $N$ transmit symbols

$$s[k_N] \triangleq [s[Nk - (N - 1)] s[Nk - (N - 2)] \ldots s[Nk]]^T,$$

and via differential decoding estimates $\hat{v}[k]$ of the $N - 1$ differential symbols

$$v[k_N] \triangleq [v[Nk - (N - 2)] v[Nk - (N - 3)] \ldots v[Nk]]^T.$$

Thereby, no explicit channel state information, i.e., no knowledge of

$$h[k_N] \triangleq [h[Nk - (N - 1)] h[Nk - (N - 2)] \ldots h[Nk]]^T$$

is required. We note that due to differential encoding consecutive blocks $r[k_N]$ overlap by one scalar received symbol [8]. The value of $N$ is referred to as the observation window size, and since with growing $N$ the memory of the fading process is more completely taken into account, the performance of ML-MSDD, and consequently that of MSDSD, shall improve with increasing $N$ [17].
3 Multiple-Symbol Differential Sphere Decoding

For the derivation of the MSDSD algorithm we first review the decision rule of ML-MSDD and adapt it to the application of SD in Section 3.1. Then, Section 3.2 provides the actual MSDSD algorithm, and an interpretation of this MSDSD algorithm in terms of DF-DD is given in Section 3.3. For the sake of compact presentation, we omit the time indices $k_N$ and $k$, and use $x_i$ to refer to the $i$th component of vector $x$.

3.1 ML-MSDD

ML-MSDD delivers the ML estimate $\hat{s}$ based on the observation of $r$. For the case of Rayleigh fading considered in this paper, the ML decision rule reads [17, Eq. (6)]

$$\hat{s} = \arg\min_s \left\{ r^H R_{rr}^{-1} r \right\} ,$$

with the correlation matrix

$$R_{rr} \triangleq \mathcal{E}\{rr^H|s\} .$$

We can further use the relations ($\text{diag}\{x\}$: diagonal matrix with components of $x$ on main diagonal, $^*$: (componentwise) complex conjugate, $I_N$: $N \times N$ identity matrix)

$$R_{rr} = \text{diag}\{s\} C \text{diag}\{s^*\} ,$$

$$C \triangleq \mathcal{E}\{hh^H\} + \sigma_n^2 I_N ,$$

$$\text{diag}\{s^*\} r = \text{diag}\{r\} s^* ,$$

to rewrite decision rule (6) as

$$\hat{s} = \arg\min_s \left\{ (\text{diag}\{r\} s^*)^H C^{-1} \text{diag}\{r\} s^* \right\} .$$

Apparently, expression (11) is a quadratic form in $s$. Therefore, we apply the Cholesky factorization of the inverse matrix

$$C^{-1} = LL^H$$

and further define

$$U \triangleq (L^H \text{diag}\{r\})^* ,$$

where $L$ and $U$ are lower and upper triangular matrices, respectively, to finally arrive at

$$\hat{s} = \arg\min_s \left\{ ||Us||^2 \right\} .$$
We note that decision rule (14) equals (2) with all-zero vector \( s_{LS} \). Hence, ML-MSDD can be regarded as a shortest vector problem, which in many cases can efficiently be solved by SD, cf. [3, 18] for transmission with CSI. Of course, the factorization of \( C^{-1} \) needs to be done only once. We note that the brute-force solution to (14) is to test \( M^{N-1} \) vectors \( s \) corresponding to all possible differential vectors \( v \), and its complexity grows exponentially in \( N \).

It is also worth pointing out that an expression similar to (14) was used as branch metric for ML sequence detection without CSI in [19, 20]. In fact, whereas (14) uses an upper triangular matrix \( U \), a lower triangular matrix is devised in [19, 20] in order to meet causality constraints, i.e., to decide on \( s_i \) prior to \( s_j \) for \( i < j \).

### 3.2 Sphere Decoding Algorithm

The sphere decoder only examines those candidate vectors \( s \) that lie inside a sphere of radius \( R \) [3]:

\[
||Us||^2 \leq R^2.
\]  

(15)

Due to the upper triangular form of \( U \), condition (15) can be checked componentwise, i.e., having found (preliminary) decisions \( \hat{s}_l \) for the last \( N - l \) components \( s_l, i + 1 \leq l \leq N \), we obtain a condition for the \( i \)th component \( s_i, 1 \leq i \leq N \). To see this, let \( u_{il} \) denote the entry of \( U \) in row \( i \) and column \( l \), \( 1 \leq i, l \leq N \), and introduce the squared length

\[
d_{i+1}^2 = \sum_{l=i+1}^{N} \left| \sum_{\ell=l}^{N} u_{\ell l} \hat{s}_\ell \right|^2
\]

(16)

accounting for the last \( N - i \) components. Then, possible values \( s_i \) have to satisfy the length criterion

\[
d_i^2 = \left| u_{ii} s_i + \sum_{l=i+1}^{N} u_{il} \hat{s}_l \right|^2 + d_{i+1}^2 \leq R^2.
\]

(17)

Once a valid vector \( \hat{s} \) is found, i.e., \( i = 1 \) is reached, the radius \( R \) is dynamically updated by

\[
R := ||Us||
\]

and sphere decoding is repeated starting with \( i = N \) and new radius \( R \). If for some index \( i \) condition (17) cannot be met, \( i \) is incremented and another value \( s_i \) is tested. MSDSD is finished if no vector is found inside the sphere with radius \( R \).

The proposed MSDSD algorithm performs the search for the ML solution \( \hat{s} \) as outlined above and is summarized by the pseudocode in Appendix A. The organization agrees in principle with the SD algorithm.
proposed in [3, Section III.B] for the closest point search in lattices. In particular, code lines 11–15, 16–22, and 24–31 correspond to parts “Case A”, “Case B”, and “Case C” in [3, Section III.B]. In the following, we comment on the specific details of our MSDSD algorithm.

**Phase Ambiguities:** Since the MSDD metric is invariant to a phase shift common to all components of \( s \) such that the same differential vector \( v \) results, without loss of generality we fix \( s_N = 1 \) (line 2) and start sphere decoding with \( i = N - 1 \) (line 3).

**Search Strategy:** The ordering of hypothetical symbols \( s_i \) examined for the \( i \)th component is critical for a low SD complexity, i.e., for early finding the ML estimate \( \hat{s} \). As suggested in [3] we apply the Schnorr-Euchner search strategy [21], i.e., the symbols \( s_i \) are ordered according to the length \( d_i \) starting with the smallest value. In this way, the search can be safely terminated as soon as \( d_i \) exceeds the current sphere radius \( R \). The appropriate ordering of \( s_i \) is accomplished by first finding the phase index \( m_i \) (\( s_i = e^{j2\pi m_i/M} \)) of the best candidate point which minimizes \( d_i \) in (17), and then zigzagging through the remaining \( M - 1 \) phase indices such that \( d_i \) increases monotonically (functions “findBest()” and “findNext()” in Appendix A). By dealing with the phase index \( m_i \) instead of the signal point \( s_i \) itself, the Schnorr-Euchner search strategy for PSK constellations becomes similar to that for pulse-amplitude modulation (PAM) signal sets, cf. e.g. [3, 22]. Furthermore, by introducing (line F1-3) and incrementing (line F2-3) the counter \( n_i \) for the number of signal points already examined for component \( i \), the finite constellation size is taken into account, i.e., the search is terminated if \( n_i = M \) is true (line 28). We note that this counter is a necessity, since in contrast to PAM signal sets no boundary region exists for the phase index \( m_i \in \mathbb{Z} \).

**Initial Radius:** When using the Schnorr-Euchner strategy no initial radius is needed, i.e., \( R \to \infty \) can be assumed. Clearly, this initialization is appealing, since the likely event of setting the value for \( R \) too small, i.e., no point lies in the sphere, is bypassed. For this reason, we chose to initialize our MSDSD algorithm with a reasonably high value of \( R \) for the following simulation results.

### 3.3 MSDSD vs. DF-DD

It is interesting to relate MSDSD to well-known DF-DD. A closer look at the elements of \( U \) reveals that

\[
 u_{ii} = r_i^* \cdot p_{i,N-i} / \sigma_{e,N-i},
\]  

(18)
where \( p_i^j \) and \( \sigma^2_{e,i} \), respectively, are the \( i \)th coefficient and the error variance of an \( i \)th order linear backward minimum mean-squared error (MMSE) predictor for the discrete-time random process \( h[k] + n[k] \), and \( p_0^j \triangleq -1 \), cf. [19, 20]. Accordingly, the estimation of \( s_i \) based on (tentative) decisions \( s_l, i + 1 \leq l \leq N \), can be interpreted as linear-prediction based DF-DD with observation window size \( N - i + 1 \), \( 1 \leq i \leq N - 1 \). Since we fix \( s_N = 1 \), MSDSD involves \( N - 1 \) decision-feedback decoders with increasing window sizes of \( 2, 3, \ldots, N \) to obtain one estimate \( \hat{s} \). It is interesting to note that the computational complexity required to obtain the first vector \( \hat{s} \) and hence, the first estimate of \( \hat{v} \), is less than that for conventional DF-DD with fixed window size \( N \). We therefore expect that in good channel conditions, where the first decision is already the ML decision, MSDSD is even less complex than DF-DD. This anticipation will be confirmed by simulation results.

4 Results and Discussion

In this section, we present simulation results and discuss the performance of MSDSD. In particular, we consider the bit-error rate (BER) and the computational complexity of MSDSD and compare them with those of Mackenthun’s algorithm (MA) and DF-DD. As exemplary system parameters, 4DPSK transmission and a fading bandwidth of \( B_f T = 0.03 \) are assumed.

4.1 Bit-Error Rate

Figure 2 shows the BER curves for MSDSD, DF-DD, and MA, respectively, with observation window sizes \( N = 6 \) and \( N = 10 \) as functions of the signal-to-noise ratio (SNR) \( 10 \log_{10}(E_b/N_0) \) (\( E_b \) and \( N_0 \) denote the average received energy per information bit and the two-sided equivalent baseband noise power density, respectively.) As reference curves, BERs for CDD (\( N = 2 \)) and coherent detection with perfect CSI are also plotted.

From Figure 2 we can observe a substantial gain in power efficiency of MSDSD and DF-DD with \( N > 2 \), respectively, compared to CDD with \( N = 2 \). MSDSD approaches quite closely the performance of coherent detection and, depending on the target BER, outperforms DF-DD by several dBs for the same value of \( N \). Interestingly, the performance of MA deteriorates with larger \( N \) due to the increasing mismatch between the blockwise time-invariant channel assumption and the actually time-varying channel coefficients. This, in fact, nicely illustrates the necessity for a true ML detector with affordable complexity for the general flat fading case.
4.2 Computational Complexity

The application of SD to MSDD is certainly interesting in itself, but for practical implementation its computational complexity is of paramount importance. Therefore, we implemented the different algorithms in floating point C and consider the number of real multiplications required per $N$ symbols as measure for complexity, cf. e.g. [23, 22]. Although neglecting computational costs of memory accesses, loops etc., this measure provides quite meaningful statements and is independent of particular processor architecture or programming skills. As lower bound for the complexity of MSDSD we use the number of multiplications required to find the first vector $\hat{s}$ and to examine only a single additional signal point for each component $s_i$, $1 \leq i \leq N - 1$, i.e., MSDSD terminates as early as possible.

As it was to be expected from other applications of SD, e.g. in multiple-antenna systems [2, 4, 5, 6, 22], MSDSD leads to a tremendous improvement over brute-force ML-MSDD in terms of computational complexity. Depending on $N$, the number of multiplications is reduced by orders of magnitudes. For example, we observed that for MSDSD with $N = 10$ and $10 \log_{10}(E_b/N_0) \geq 10$ dB the expected number of visited symbols $s_i$ for the $i$th component, $1 \leq i \leq N - 1$, is less than four, whereas $M^{N-i}$, i.e., on average $\frac{1}{N-1} \sum_{i=1}^{N-1} 4^{N-i} = 38836$, symbols $s_i$ are tested for brute-force ML-MSDD.

Figure 3 compares the expected complexity of MSDSD with that of DF-DD and MA, respectively, for $N = 10$ as function of the SNR. For the implementation of DF-DD and MA we followed closely the description in [12] and the MATLAB program in [24, Section 5], respectively. For the sake of comparison, we also included the lower bound for the complexity of MSDSD. As can be seen, the complexity of MSDSD decreases rapidly and approaches the lower bound with increasing SNR, since the search quickly terminates for small enough noise. Especially for $10 \log_{10}(E_b/N_0) \gtrsim 15$ dB, which corresponds to BER $\lesssim 2 \cdot 10^{-2}$, it is well in the order of or even below the complexity of DF-DD, which is independent of the SNR. This emphasizes on the practical relevance of MSDSD, as the performance gain shown in Figure 2 does not entail an increase in complexity or is even accompanied by a complexity reduction. MA requires about a third of the multiplications of DF-DD, but as concluded from the results in Figure 2 is only an appropriate alternative for rather time-invariant fading channels.

To gain more insight into the properties of MSDSD, it is quite illustrative to consider the complexity exponent [23, 22] $E_C(N) \triangleq \log_{N-1}(\text{average number of multiplications})$.

$^3$We use $N-1$ instead of $N$ as basis of the logarithm, since only $N-1$ components of $s$ are searched for.
In Figure 4, $E_C(N)$ is plotted as a function of $N$ for different values of $10 \log_{10}(E_b/N_0)$. The respective curves for DF-DD, MA, and the lower MSDSD bound are also included for comparison. We can observe that the behavior of $E_C(N)$ heavily depends on the particular adjusted SNR. In particular, it appears that for the larger values of $10 \log_{10}(E_b/N_0)$ the complexity exponent approaches a constant, which in turn implies polynomial expected complexity of MSDSD [23]. For $10 \log_{10}(E_b/N_0) = 40$ dB the measured complexity of MSDSD is quite close to the lower complexity bound. From Figure 4 we can also conclude that the performance gain of MSDD over DF-DD comes at practically no cost in complexity considering usual observation window sizes of $N \leq 10 \ldots 20$.

So far, we have considered expected complexity of MSDSD. Of course, in a practical implementation the maximum complexity will be limited, and we are interested in the implications of this on the performance of MSDSD. Therefore, Figure 5 depicts the achieved BERs for the case that the maximum number of multiplications is limited to a certain value. MSDSD with $N = 8$ and $10 \log_{10}(E_b/N_0) = 10, 20, 30$ dB are assumed. For a proper comparison, the BERs achieved and, respectively, the maximum and the average number of multiplications required for MSDSD without any complexity limitation are also plotted. As can be seen, the maximum number of multiplications can safely be limited to about $2 \cdot 10^3$ multiplications per $N$ symbol block without sacrificing performance in this example. Especially for low SNRs this implies a tremendous reduction in worst-case complexity, which then is well in the order of the average complexity. For high SNRs we have more generally observed that setting a reasonable complexity limit does practically not affect performance, since the probability of exceeding this limit is far below the achievable BER without any limit. Hence, MSDSD is an attractive solution also in this regard.

5 Conclusions

In this paper, we have proposed the application of SD to accomplish MSDD for Rayleigh fading channels. Following the footsteps of the SD literature we derived an efficient MSDSD algorithm, whose expected complexity is orders of magnitudes below that of brute-force search typically used in MSDD. The presented simulation results verify the excellent performance versus complexity trade-off of the proposed MSDSD. It was shown that the gains in power efficiency compared to suboptimum low-complexity multiple-symbol detectors come almost for free. Furthermore, imposing reasonable limitations on the maximum MSDSD complexity did practically not affect performance.
References

A Pseudocode for MSDSD

```
function MSDSD(U, M, N, R)
Input: N x N upper triangular matrix U, constellation size M, window length (dimension) N, initial radius R
Output: Maximum-likelihood decision \( \hat{s} \)

1. \( d_N := 0 \) // initialize length
2. \( s_N := 1 \) // fix last component of \( \hat{s} \)
3. \( i := N - 1 \) // start with component \( i = N - 1 \)
4. \( k := \sum_{i=i+1}^{N} u_{ii} s_i \) // add last \( N - i \) components, see (16)
5. \([m_i, \text{step}_i, n_i] = \text{findBest}(k, u_{ii}, M)\) // find best candidate point for \( i \)th component
6. \( \{ \text{loop} \) // subfunction
7. \( d_{\text{new}} := |k + u_{ii} \cdot e^{\frac{2\pi}{M} m_i}|^2 + d_{i+1}^2 \) // update squared length, see (17)
8. \( \text{if } d_{\text{new}} < R \text{ and } n_i \leq M \{ \) // check sphere radius (17) and constellation size
9. \( d_i := d_{\text{new}} \) // store squared length
10. \( s_i := e^{\frac{2\pi}{M} m_i} \) // store candidate component
11. \( \text{if } i \neq 1 \{ \) // component 1 not reached yet
12. \( i := i - 1 \) // move down
13. \( k := \sum_{i=i+1}^{N} u_{ii} s_i \) // add last \( N - i \) components, see (16)
14. \( [m_i, \text{step}_i, n_i] := \text{findBest}(k, u_{ii}, M) \) // find best candidate point for \( i \)th component
15. \} // first component reached
16. \( \hat{s} := s \) // best point so far
17. \( R := d_i \) // update sphere radius
18. \( i := i + 1 \) // move up
19. \( k := \sum_{i=i+1}^{N} u_{ii} s_i \) // add last \( N - i \) components, see (16)
20. \( [m_i, \text{step}_i, n_i] := \text{findNext}(m_i, \text{step}_i, n_i) \) // next point examined for component \( i \)
21. \} // while all constellation points examined
22. \( \{ \text{loop} \) // subfunction
23. \( \text{if } i == N - 1 \text{ return } \hat{s} \text{ and exit} \) // outside sphere and no component left
24. \( i := i + 1 \) // move up
25. \} while \( n_i == M \)
26. \( k := \sum_{i=i+1}^{N} u_{ii} s_i \) // add last \( N - i \) components, see (16)
27. \( [m_i, \text{step}_i, n_i] := \text{findNext}(m_i, \text{step}_i, n_i) \) // next point examined for component \( i \)
28. \) goto (loop)

\textbf{subfunction} \([m_i, \text{step}_i, n_i] = \text{findBest}(k, u_{ii}, M)\) // Finds MPSK signal point that minimizes (16)

F1-1 \( p := \frac{M}{2\pi} \text{ (angle (} \frac{k}{u_{ii}} \text{) - } \pi) \) // unconstrained phase index \((p \in \mathbb{R})\)
F1-2 \( m_i := \lfloor p \rfloor \) // constrained phase index \((m_i \in \mathbb{Z})\)
F1-3 \( n_i := 1 \) // counter of examined candidates for component \( i \)
F1-4 \( \text{step}_i := \text{sign}(p - m_i) \) // step size for phase index

\textbf{subfunction} \([m_i, \text{step}_i, n_i] = \text{findNext}(m_i, \text{step}_i, n_i)\) // Selects next MPSK signal point

F2-1 \( m_i := m_i + \text{step}_i \) // zig-zag through MPSK constellation
F2-2 \( \text{step}_i = -\text{step}_i - \text{sign} (\text{step}_i) \) // update step size
F2-3 \( n_i := n_i + 1 \) // count examined candidates for component \( i \)
Figure 2: BER vs. $10 \log_{10}(E_b/N_0)$ for 4DPSK and Rayleigh fading with $B_fT = 0.03$. Comparison of MSDSD, DF-DD, and MA. As reference: CDD ($N = 2$) and coherent detection with CSI.

Figure 3: Average number of multiplications vs. $10 \log_{10}(E_b/N_0)$ for 4DPSK and Rayleigh fading with $B_fT = 0.03$. Comparison of MSDSD, DF-DD, and MA. Observation window size $N = 10$. 


Figure 4: Complexity exponent vs. observation window size $N$. 4DPSK and Rayleigh fading with $B_fT = 0.03$.

Figure 5: BER vs. maximum allowed number of multiplications for MSDSD with $N = 8$. 4DPSK and Rayleigh fading with $B_fT = 0.03$. 