Low Complexity Golden Code Analytical and Deep Learning-Based Sphere-Decoders For High-Density M-QAM

BHEKISIZWE MTHETHWA AND HONGJUN XU
School of Engineering, University of KwaZulu-Natal, Durban 4041, South Africa
Corresponding author: Bhekisizwe Mthethwa (tshomie2020@yahoo.com)

ABSTRACT In this paper, we develop low complexity Golden code sphere-decoding (SD) algorithms for high-density M-ary quadrature amplitude modulation (M-QAM). We define the high-density M-QAM as having modulation orders \( M \) of at least 64, i.e. \( M \geq 64 \). High-density M-QAM symbols deliver high data rates under good wireless channels. Future wireless systems must deliver high data rates and simultaneously low end-to-end latency. However, higher M-QAM modulation orders increase the Golden code SD search breadth, thus increasing decoding latency. We, therefore, propose two forms of low complexity Golden code SD to achieve low decoding latency while maintaining the near-optimal SD bit-error rate (BER). The proposed low complexity SD algorithms are based on the SD with sorted detection subsets (SD-SDS). The literature shows the SD-SDS to achieve lower detection complexity relative to the Schnorr-Euchner SD (SE-SD). The first form of the proposed Golden code SD is the SD-SDS-Ascend algorithm with instantaneously varying subset lengths and a search tree search order sorted based on the worst-first search strategy. The second form of the proposed Golden code SD is an SD-SDS algorithm called SD-SDS-ES-DNN with a deep learning-based early stopping search criterion. Our proposed algorithms achieve at most 57% and 70% reduction in Golden code decoding latency relative to SD-SDS, at low SNR, for 64-QAM and 256-QAM, respectively. At high SNR, the proposed algorithms achieve 40% and 37% in Golden code decoding latency reduction relative to the SD-SDS for 64-QAM and 256-QAM, respectively. The decoding latency reduction is achieved while maintaining near-optimal BER performances.

INDEX TERMS Deep Learning, Golden Code, Space-Time Coding, Sphere-Decoding, Supervised Learning.
I. INTRODUCTION

With the high demand for communication services that require high data throughputs and low end-to-end latency, coupled with the sharp increase of mobile devices depending on wireless communications, the wireless communication literature proposes various multiple-input multiple-output (MIMO) architectures to cater to these demands. Wireless MIMO technology offers high data throughputs and link reliability through spatial multiplexing and spatial diversity, respectively [1]. The spatial diversity offers the benefit of creating high-reliability wireless links. However, the wireless link reliability is enhanced using space-time block coding (STBC) signal processing at the transmitter, which adds time diversity to MIMO spatial diversity. These STBC schemes create full-diversity wireless links by transmitting replicas of the data symbols over two or more timeslots. One such scheme is the non-orthogonal STBC Golden code [2].

Golden code is a full-rate full-diversity STBC scheme relevant to meet the demands for high data throughput and link reliability. Golden code offers not only an added time diversity over the space diversity of MIMO, but it also adds spatial multiplexing gain over and above the spatial multiplexing inherent in wireless MIMO. Golden code further improves the data throughputs offered by wireless MIMO over and above enhancing the link reliability. It, however, has the drawback, which other non-orthogonal STBC schemes have, of having nonlinear maximum likelihood (ML) detection at the receiver side [3]. In the case of the Golden code, the optimal ML detector has a detection complexity of $O(M^4)$ [4]. This nonlinear detection complexity of order $O(M^4)$, where $M$ is the M-QAM modulation order, has the negative effect of high decoding latency, increasing end-to-end latency. Golden code has low adoption in modern wireless standards relative to the orthogonal STBC scheme called Alamouti [5]. Another non-orthogonal STBC scheme is the half-rate full-diversity uncoded space-time labeling diversity (USTLD) with nonlinear optimal ML detection [6].

The orthogonal STBC schemes exploit their orthogonality property to deliver optimal linear ML, $O(M)$, detection complexity at the receiver side in block fading channels [3]. For example, the Alamouti is a half-rate full-diversity orthogonal STBC scheme, yet it is implemented in the various WiFi [7] and LTE [8] wireless standards. Golden code is incorporated in the now-defunct WiMAX standard [9]. Despite the Golden code having spatial multiplexing gain over the Alamouti scheme, its popularity in wireless standards is low, possibly due to the higher detection complexity than the Alamouti linear ML detector.

The literature proposes various novel lower detection complexity algorithms to lower the Golden code detection complexity, hence decoding latency. In [4], the authors propose a fast ML detection of Golden code using a sphere-decoder (SD) with a search tree with reduced dimensions. In [10], the authors manage to reduce the Golden code detection complexity to $O(M^{1.5})$ at the expense of losing 1 dB signal-to-noise ratio (SNR) relative to the optimal ML detector. The authors introduced fast ML detection in [11], and their detection scheme achieves a detection complexity order of $O(M^2)$ with near-optimal bit-error-rate (BER) performance. In [12], the authors propose the low complexity Schnorr-Euchner SD (SE-SD) as an SD variant for the Golden code detection. The SE-SD algorithm does not require the re-adjusting or increasing of the search radius compared to the traditional SD [13], which increases its search radius when there are no lattice points found inside the hypersphere. From the literature, it is known that SD detection complexity depends on the search tree’s search breadth and depth [14]. The authors in [15] reduce the search breadth of the Golden code SD by creating detection subsets (SD-DS) of the full M-QAM signal cardinality. The SD-DS detection strategy is shown to achieve lower detection complexity relative to the SE-SD algorithm while exhibiting near-optimal BER performances. The SD with sorted detection subsets (SD-SDS) is ventilated in [16], where the M-QAM signal constellation candidate symbols are sorted in ascending order based on which symbols are closest to the estimated symbols detected by the sub-optimal QR decoder. The furthest symbols from the estimated M-QAM symbols are the least likely transmitted symbols. Hence the SD-SDS algorithm creates the detection subsets by rejecting the candidate symbols furthest away from the estimated M-QAM symbols. This strategy reduces the signal cardinality, hence SD search breadth of SD-SDS, and achieves a detection complexity that is 1 order lower than the SD-DS algorithm in [15].

Recently, deep learning has been applied to lower the detection complexity of the SD algorithm for large MIMO architectures. A deep learning algorithm is introduced in [17] that predicts the number of lattice points inside the SD hypersphere. The prediction is based on the SD initial radius, which is reduced until the number of predicted lattice points inside the hypersphere is small. With this sufficiently small initial radius, the SD algorithm is initiated, and hence lower detection complexity is achieved since the SD complexity also depends on the value of the initial radius. In [18], the authors propose a deep learning-based initial radius predictor algorithm that uses the instantaneous wireless channel conditions and noise statistics to predict the initial radius for SD in large MIMO. The traditional SD algorithm calculates the initial radius based on average channel conditions. The calculation of the initial radius based on the average channel conditions has a disadvantage. When the instantaneous channel is good, the SD will have many lattice points inside the hypersphere due to the fixed initial radius, which depends on average channel conditions. The authors in [19] propose a low complexity deep learning-based SD for large MIMO. This deep learning-based SD algorithm provides low complexity offline training and online decoding compared to the deep learning-based SD algorithms in the literature. In
[20], the authors propose a deep learning-based SD minimum path metric predictor for the sub-trees. These minimum path metrics are used for early search termination for candidates on the SD search tree. The algorithm is developed for a large MIMO architecture and achieves considerable low detection complexity while achieving near-optimal BER performances.

A. MOTIVATION

The low complexity SD-SDS in [16] is shown to achieve lower detection complexity relative to the SD-DS in [15] and the SE-SD variant described in [12]. However, the SD-SDS detection subset lengths are set based on the average SNR values, as shown in [16, Table 2]. Therefore, the SD-SDS fixed-length detection subsets leave room for further reduction in subset lengths based on the instantaneous channel and noise statistics. Good instantaneous channel and noise statistics may prompt even shorter subset lengths relative to the fixed lengths, as we do not need to search through as many symbol candidates under such conditions. Therefore, we are motivated to propose instantaneously varying subset lengths that vary based on the instantaneous channel quality. The subset lengths can be shortened relative to the average SNR-based subset lengths determined in [16, Table 2] at high instantaneous SNR. Shorter subset lengths shorten the SD-SDS search breadth and thus reduce decoding latency. The SD-SDS algorithm also exhibits another opportunity for reduction in detection complexity by ordering the search order of the SD-SDS search tree. The SD-SDS in [16] currently has a search order that is not ordered based on any instantaneous channel quality. As shown in the paper, a fixed search order has a disadvantage at low SNR since the SD-SDS architecture has a detection complexity dominated by the detection complexity at search layer 1 of the search tree. It thus motivates us to propose a worst-first search strategy that ensures that the candidate subset with the most petite subset length is always used at search layer 1. This lowers the decoding latency at low SNR, as shown in the paper.

The SE-SD variant described in [12] orders the search tree search order or wireless channel matrix columns in ascending order, based on the instantaneous wireless fading power. In the search order, the M-QAM symbol that experiences the highest instantaneous wireless fading power is detected first in the SD search tree. However, the metric used to sort the search tree search order is based on the instantaneous wireless fading power and is not an accurate metric for determining instantaneous channel quality. The instantaneous noise statistics are not factored in by [12]. The approach in [12] is valid for high SNR as the wireless fading power dominates the performance at high SNR since the average noise power is very low. However, noise statistics dominate the system performance at low SNR as the average noise power is very large. Therefore, we are motivated to propose a metric that factors in the instantaneous noise power and wireless fading power to sort the SD-SDS search tree search order.

Deep learning-based low complexity SD algorithms are proposed in [17-20]. However, these deep learning algorithms are specifically designed to lower the detection complexity of large MIMO SD systems where $N_t = N_r \geq 8$. The number of transmit and receive antennas in a MIMO configuration are defined as $N_t$ and $N_r$, respectively. The deep learning-based SD algorithms also reduce complexity in the traditional SD which is a high complexity decoding algorithm. The proposed deep neural networks (DNN) in [19-20] are very complex for small MIMO environments, such as $N_r = 2$. Further, in [20], the authors rely on the large MIMO property of channel hardening to design the DNN architecture that predicts the minimum path metrics for the sub-trees. These predicted minimum path metrics are used to initiate early termination of the SD search. However, this solution will not apply to small MIMO channels as the assumption of channel hardening does not hold. The other drawback of [20] is the DNN architecture complexity. The hidden layer is set to have $2N_t + 2M$ neurons, and the output layer has $M$ neurons. It is easy to see that for the high-density M-QAM, $M \geq 64$ contexts, the DNN complexity will increase the decoding latency for the small MIMO low complexity SD-SDS-based decoders. Therefore, we are motivated to propose an SD-SDS search tree early stopping deep learning-based algorithm with a low inference time DNN architecture that is invariant to the M-QAM modulation order. This DNN algorithm prematurely terminates the SD-SDS search under learned channel conditions. This has the advantage of lowering the decoding latency of the Golden code SD-SDS search tree.

B. CONTRIBUTIONS

This paper proposes two forms of low complexity Golden code SD-based algorithms. We present analytical algorithms that further reduce the decoding latency of the low complexity SD-SDS. We propose a deep learning-based early stopping algorithm that prematurely terminates the SD-SDS search under specific instantaneous channel conditions. Based on the literature survey, none of the research has attempted to reduce the decoding latency of the low complexity Golden code SD-SDS algorithm in a small MIMO environment i.e. $N_t = 2$ and $N_r \in [N_r;8]$. The reduction in decoding latency is necessary for high-density M-QAM modulation as future wireless standards will require the use of high-density M-QAM for faster data rates but at the same time with a low end-to-end latency constraint. High-density M-QAM increases the search breadth of the Golden code SD-SDS search tree, increasing decoding latency and negatively affecting the end-to-end latency. Our main contributions of the paper are listed as follows:

- We propose a simple metric that more accurately describes the channel quality compared to the instantaneous wireless fading power described in [12]. Not only does our proposed metric consider the wireless
fading power gain, but it also indirectly considers the instantaneous noise power.

- We propose a heuristic approach to instantaneously set the sorted candidate symbol subset lengths based on the proposed simple metric used to measure the instantaneous channel quality. The sorted candidate symbol subset lengths are not necessarily identical for each of the estimated M-QAM symbols, \( \hat{x}_q \) \( \forall q \in [1:4] \), as they experience different wireless fading power and noise power. The instantaneously varying subset lengths are shorter than the average SNR-based fixed subset lengths at high instantaneous SNR. This lowers the decoding latency of the SD-SDS search tree.

- We exploit to our advantage the instantaneously varying wireless channel quality for each estimated M-QAM symbol to sort the search tree search order either in ascending or descending order based on the proposed channel quality metric. In literature, the search tree search order sorting has mainly been in ascending order or best-first strategy [12]. We show in this paper that the worst-first strategy is beneficial in the low SNR regions and further reduces decoding latency relative to the best-first strategy.

- We finally propose a novel early stopping deep learning-based SD-SDS algorithm that takes advantage of the sorted candidate symbols in the subsets. The candidate symbols are sorted from the most likely transmitted symbol to the least likely transmitted symbol for each search layer in the search tree. We thus take advantage of this and the depth-first search strategy to prematurely terminate the search on the first lattice point found inside the hypersphere. This termination only happens when the instantaneous channel conditions are good. The DNN developed in this paper detects when these channel conditions are good enough to perform early stopping. In literature, an early termination DNN algorithm is developed, for large MIMO [20], using the property of channel hardening. This property does not apply in our context of small MIMO. Over and above that, the DNN architecture in [20] is too complex for our small MIMO low complexity SD-SDS high-density M-QAM environment.

The remainder of this paper is organized as follows: in Section II, the system model of the paper is presented. In Section III, we present the theoretical overview of the SD-SDS algorithm. In Section IV, we present the proposed low complexity analytical SD based algorithm. Section V presents the low complexity Deep Learning-based SD algorithm. Section VI presents the Simulations results and discussion. Section VII concludes the paper.

**Notation:** Bold lowercase letters are used for vectors and bold uppercase for matrices. \( (\cdot)^T \), \( (\cdot)^H \), \( |\cdot| \), \( \|\cdot\| \) and \( \|\cdot\|_F \) represent the Transpose, Hermitian, Absolute Value, Euclidean norm and Frobenius norm operations, respectively. The functions \( \mathcal{R}(\cdot) \) and \( \mathcal{I}(\cdot) \) are the real and imaginary components of a complex number, respectively. \( j \) is a complex number. The statistical average is represented by the expectation function \( \mathbb{E}(\cdot) \). The function \( (\cdot)^* \) is the complex conjugate of a complex number. The function \( \text{vec}(\cdot) \) is a matrix vectorization function that stacks the column vectors of a matrix on top of each other to form a single column vector.

**Acronyms:** The salient algorithm acronyms used in this paper are stated as follows together with their definitions:

- **SD-SDS:** Is a Golden code SD algorithm with fixed candidate symbol subset lengths and fixed search tree search order.
- **SD-SDS-Ascend:** Is a Golden code SD algorithm with instantaneously varying subset lengths and a search tree search order sorted based on the worst-first search strategy.
- **SD-SDS-Descend:** Is a Golden code SD algorithm with instantaneously varying subset lengths and a search tree search order sorted based on the best-first search strategy.
- **SD-SDS-ES-DNN:** Is a Golden code SD-SDS algorithm with a deep learning-based early stopping search criterion.

**II. SYSTEM MODEL**

In this paper we consider an \( N_t \times N_r \) wireless MIMO channel with the transmit and receive antenna constraints which are governed by \( N_t = 2 \) and \( N_r \in [N_t;8] \). The Golden code wireless channel matrix for timeslot \( i \) is defined as \( H_i \in \mathbb{C}^{N_r \times N_t} \), \( \forall i \in [1:2] \). The wireless channel is fast frequency-flat fading which implies that the wireless channel matrix entries change for each transmission timeslot. The wireless channel matrix entries are drawn from an independent and identically distributed (i.i.d) zero-mean complex Gaussian distribution \( CN(0,1) \). This implies that each entry's wireless channel fading gain is drawn from a Rayleigh distribution. The wireless channel matrix is assumed to be known at the receiver side.

Each Golden code super symbol is formed from a pair of M-QAM symbols that carry the \( \log_2 M \) information data bits. The way the transmission works is that the random data bit streams, at the physical layer, are packaged into 4 independent M-QAM complex symbols that each carry \( \log_2 M \) data bits. Then 2 of the 4 complex M-QAM symbols are selected to form the first Golden code super symbol, and the remaining 2 M-QAM symbols are used to form the second Golden code super symbol. In transmission timeslot 1, the first Golden code super symbol, \( x_{1_{11}} \), is sent from transmit antenna 1, and the second Golden code super symbol, \( x_{1_{2}} \), is sent from transmit antenna 2. In transmission timeslot 2, the M-QAM symbol pairs used to construct the Golden code super symbol 1 are used to construct the third Golden code super symbol, \( x_{211} \), transmitted from transmit antenna 1. In the same transmission timeslot 2, a fourth
Golden code super symbol is created from the same M-QAM symbol pairs used to construct the second Golden code super symbol. The fourth Golden code super symbol, \( x_{22} \), is transmitted in timeslot 2 from transmit antenna 2. The Golden code super symbols are constructed from the complex M-QAM symbol pairs as follows: \( x_{11} = \frac{a}{\sqrt{5}}(x_1 + x_2 \theta) \), \( x_{12} = \frac{a}{\sqrt{5}}(x_3 + x_4 \theta) \), \( x_{21} = \frac{a}{\sqrt{5}}(x_1 + x_2 \bar{\theta}) \), and \( x_{22} = \frac{a}{\sqrt{5}}(x_3 + x_4 \bar{\theta}) \). The scalar parameters \( a, \bar{a}, \theta \) and \( \bar{\theta} \) are defined as follows: \( a \Theta 1 + j \bar{\theta} \), \( \bar{a} \Theta 1 + j \theta \), \( \theta = \frac{1 + \sqrt{5}}{2} \) and \( \bar{\theta} = \frac{1 - \sqrt{5}}{2} \). The complex M-QAM symbols are \( x_1, x_2, x_3, x_4 \in \Omega_M \), where \( \Omega_M \) is an arbitrary square M-QAM signal constellation.

At the receiver side, the distorted Golden code super symbols are perturbed using the following system equation in (1) for timeslot \( i, \forall i \in [1:2] \):

\[
y_i = H_i x_i + n_i
\]  

where \( x_i = [x_{1i} x_{2i}]^T \) is the Golden code super symbol transmission vector for timeslot \( i \), \( y_i \in \mathbb{C}^{N_r \times 1} \) is the received perturbed Golden code super symbol signal vector for timeslot \( i \) and \( n_i \in \mathbb{C}^{N_r \times 1} \) is the noise vector for timeslot \( i \). The Golden code super symbol power is constrained to unity, i.e., \( E(|x_{1i}|^2) = E(|x_{2i}|^2) = 1 \). The noise vectors \( n_i \) noise entries are drawn from an i.i.d zero-mean complex Gaussian distribution \( \mathcal{C}N(0, \sigma^2) \). The average received SNR per receive antenna is \( \bar{y} \). The average noise power \( \sigma^2 \) is assumed to be known at the receiver side.

In our paper, we rely on the alternative representation of the system model in (1). As per [12] and [16], the transmission vector is based on the complex M-QAM symbols instead of the Golden code super symbols. This is achieved by rearranging the system model in (1) using the following rules in (2)

\[
y = \tilde{H}u + n
\]  

where \( \tilde{H} = [\tilde{H}_1 \tilde{H}_2]^T \in \mathbb{C}^{2N_r \times 2N_t} \), \( y = [y_1 y_2]^T \in \mathbb{C}^{2N_r \times 1} \), \( u = [x_1 x_2 x_3 x_4]^T \), \( n = [n_1 n_2]^T \in \mathbb{C}^{2N_r \times 1} \), \( \tilde{H}_1 = \frac{1}{\sqrt{5}}H_1 \) (\( \alpha \alpha \theta 0 0 \)), \( \tilde{H}_2 = \frac{1}{\sqrt{5}}H_2 \) (\( \bar{\alpha} \bar{\alpha} \bar{\theta} 0 0 \)), \( H_1 \) and \( H_2 \) are the wireless channel matrices, for timeslot \( i \), that includes the Golden code super symbol constants \( \alpha, \theta, \bar{\alpha} \) and \( \bar{\theta} \). The rest of the paper will use the system model in (2) for the Golden code sphere-decoding based detection algorithms.

III. GOLDEN CODE SD-SDS OVERVIEW

The authors in [16] introduced the SD-SDS algorithm, a modified version of the SE-SD algorithm described in [12]. The SD-SDS algorithm is slightly different from the SE-SD algorithm because it does not perform the SD search over the full signal cardinality of the M-QAM constellations. In [12], the SE-SD algorithm performs the SD search over the full sorted M-QAM symbol candidates. The candidate symbols in SE-SD, just like in SD-SD, are sorted in ascending order from the closest complex M-QAM symbol to the furthest. In SD-SDS, the furthest sorted candidate symbols are discarded as they are least likely to have been the transmitted symbols.

In [16], the search tree tree search order of the SD-SDS algorithm is not sorted using the best-first search strategy as described in [12]. The SD-SDS algorithm search order execution is fixed and not sorted based on channel conditions. The SD-SDS algorithm is shown in Algorithm 1.

Algorithm 1: SD-SDS [16]

Input: \( y, \tilde{H}, \tilde{y}, N_t \)

Output: \( i_1, i_2, i_3, i_4 \)

1. global result \( \leftarrow \) [], \( r^2 \)
2. global lattice_point_found_flag \( \leftarrow \) False
3. \( z, \tilde{R}, \tilde{z} \leftarrow \) qrDecoder(\( y, \tilde{H} \))
4. subsetList[ ] \( \leftarrow \) getSortedSubsets(\( \tilde{x} [1], \tilde{y} \))
5. \( \sigma^2 \leftarrow \frac{N_t}{\bar{y}} \), \( K = 10, N = 2N_t \)
6. \( r^2 \leftarrow 2\sigma^2KN - z^H(I - \tilde{R}(\tilde{R}^H \tilde{R})^{-1} \tilde{R}^H)z \) [13, Eq(28)]
7. \( I \) is a \( 2N_r \times 2N_r \) identity matrix.
8. lattice_point_found_flag \( \leftarrow \) False,
9. layer \( \leftarrow 2N_t, \) dist \( \leftarrow 0 \)
10. sphere_decoder \( (z, \tilde{R}, \text{subsetList}[1], \) dist, layer) \( \leftarrow \) False,
11. if lattice_point_found_flag = False then
12. \( i_4 \leftarrow \) result[4], \( i_3 \leftarrow \) result[3], \( i_2 \leftarrow \) result[2], \( i_1 \leftarrow \) result[1]
13. else
14. \( z, \tilde{R}, \tilde{z} \leftarrow \) qrDecoder(\( y, \tilde{H} \))
15. for \( q \leftarrow 1 \) to \( 4 \) do
16. \( d \leftarrow [ ] \)
17. for \( i \leftarrow 1 \) to \( M \) do
18. \( d[i] \leftarrow ||\tilde{z}[q] - u_i||^2, \forall u_i \in \Omega_m \)
19. end for
20. \( i_q \leftarrow \) argmin(\( d[] \))
21. end for
22. end if
23. function qrDecoder(\( y, \tilde{H} \))
24. \( \tilde{Q}, \tilde{R} \leftarrow qr(\tilde{H}) \)
25. \( z \leftarrow \tilde{Q}^Hy, \tilde{z} \leftarrow [ ] \)
26. for \( q \leftarrow 4 \) to \( 1 \) do
27. \( \tilde{z}[q] \leftarrow \frac{[z[q] - \Sigma_{i=q+1}^M R[i][q]u_i]}{R[q][q]} \)
28. end for
29. return \( z, \tilde{R}, \tilde{x} \]
30. function getSortedSubsets(\( \hat{x}[\cdot], \hat{y} \))
31. subsetList \( \leftarrow [ ] \)
32. if \( \| y \|_F^2 \leq \gamma_F^M \) then
33. \( L \leftarrow L_{\text{low}} \)
34. else
35. \( L \leftarrow L_{\text{high}} \)
36. end if
37. for \( q \leftarrow 1 \) to \( 4 \) do
38. \( \text{dict} \leftarrow [ ] \)
39. for \( i \leftarrow 1 \) to \( M \) do
40. \( \text{dict}[i] \leftarrow | \hat{x}[q] - u_i |^2, \forall u_i \in \Omega_M \)
41. end for
42. indices[\( q \)] \( \leftarrow \) sort(dict[\( q \)]
43. subsetList[\( q \)] \( \leftarrow \) indices[1: \( L \)]
44. end for
45. return subsetList[\( ] \)

It is evident from Algorithm 1 that the sorted subsets are generated by sorting the candidate symbols, in ascending order, using the square of the Euclidean distance between the estimated complex M-QAM symbols from the QR decoder and the exact complex M-QAM symbols from the signal constellation. The sorted symbol candidates are then truncated by taking the first \( L \) symbol indices of the sorted candidates. Each SD-SDS search layer will have its own \( L \)-dimensional sorted detection subset list for the candidate symbols since each of the M-QAM symbols in the set \( \{ x_1, x_2, x_3, x_4 \} \) are perturbed by different instantaneous fading channels and noise conditions. The order of execution of the SD-SDS search tree order is shown in the following one-to-one correspondence \( [\hat{x}_4 \mapsto 4, \hat{x}_3 \mapsto 3, \hat{x}_2 \mapsto 2, \hat{x}_1 \mapsto 1] \). The correspondence shows the mapping of which symbols are estimated in each search layer, numbered 1 to 4, and which order are the layers executed in the search tree. This order does not change, unlike in the case of the SE-SD algorithm, in which the search order changes based on the instantaneous wireless channel fading power. The SD-SDS search tree searches for lattice points, \( \hat{R}u \), that lie inside the hypersphere, \( \| z - \hat{R}u \|_F^2 \leq r^2 \), with radius \( r \). The vector \( z = \hat{Q}^H y = \hat{R}u + \hat{Q}^H n \in \mathbb{C}^{2N_r \times 1} \) is the received signal vector, \( \hat{R} \in \mathbb{C}^{2N_t \times 2N_r} \) is a random upper triangular matrix related to the wireless channel matrix, \( \hat{H} \), via the reduced QR decomposition \( \hat{H} = \hat{Q}\hat{R} \). The matrix \( \hat{Q} \in \mathbb{C}^{2N_r \times 2N_r} \) is a unitary matrix.

If no lattice point lies inside the hypersphere, the sub-optimal QR decoder is used to output the estimated transmitted symbol indices directly. If a lattice point lies inside the hypersphere, then the closest lattice point to the received signal vector is found using the Schnorr-Euchner (SE) search strategy as described in [12]. This closest lattice point contains the indices of the estimated transmitted symbols.

Looking at Figure 1, we see the SD-SDS search tree with each of the 4 search layers having a fixed length \( L \)-dimensional sorted candidate symbol subset shown as nodes 1 to \( L \). The search for the closest lattice point to the received signal vector is performed using a search tree that combines the depth-first search strategy with the SE strategy. The depth-first search strategy ensures that the SD-SDS algorithm finds the closest lattice point to the received signal vector earlier in the search as possible. This is especially true for SE-SD and SD-SDS as the most likely transmitted candidate symbols are placed first in the candidate symbol subset and therefore are used first in the search on all search layers. This limits the number of lattice points found inside the hypersphere, lowering the detection complexity. The limitation occurs because after finding the first lattice point candidate inside the hypersphere, the SD-SDS adjusts the radius of the hypersphere to a smaller radius based on the distance between the lattice point and the received signal vector. It means only lattice points closer to the received signal vector will be considered going forward. Since the best candidate symbols are placed first at each search layer, it implies that at high instantaneous SNR, we can expect that the first lattice point found inside the hypersphere is the closest lattice point to the received signal vector. However, despite finding the closest lattice point, the SD-SDS algorithm continues searching all the unvisited nodes of the search tree, using the SE strategy and testing if they possibly lie inside the hypersphere. Therefore, this is an opportunity to lower the decoding latency by prematurely terminating the SD-SDS search the moment the first lattice point is found inside the hypersphere, under good instantaneous channel conditions.

The best-case scenario for the search tree is that the average SNR will be as high as possible, i.e. \( \sigma^2 \rightarrow 0 \), such that there is a very high occurrence of high instantaneous SNR channel conditions. This leads to a smaller Hypersphere radius since \( r^2 \propto \sigma^2 \) [13]. The sorted candidate subsets, for each search layer, and the depth-first search strategy can be relied upon to find the closest lattice point to the received signal vector as the first lattice point inside the Hypersphere. The first lattice point exists on the far left of the search tree in Figure 1. This is because at high instantaneous SNR, the transmitted symbols experience minimal perturbation, and thus, any candidate...
symbol closest to the estimated M-QAM symbol is most likely the transmitted symbol. This makes the sorted candidate symbol subset a reliable subset containing the most likely transmitted symbols as the first symbols in the subset.

The worst-case scenario exists when the average SNR is low, i.e. $\sigma^2 \rightarrow \infty$, and the search radius becomes very large. A low average SNR also implies a high occurrence of low instantaneous SNR channel conditions. This makes the sorted candidate symbol subset unreliable as it no longer holds that the first symbols in the sorted candidate symbol subset are the most likely transmitted symbols. It then becomes possible that the closest lattice point to the received signal vector exists at the far right of the search tree in Figure 1, i.e. last lattice point. Under the worst-case scenario, it is obvious to see that the search tree detection complexity is dominated by the detection complexity at search layer 1. For the search tree to find the lattice point at the far right of the search tree, it will have to compute the Euclidean distance calculations in layer 1 $L^4$ times. For layer 2 up to layer 4, the search tree computes the Euclidean distance calculations $L^3, L^2$, and $L$ times, respectively. The worst-case scenario makes the search tree equivalent to the ML detector, with detection subset length of $L$, as the order of execution of the search tree approaches $O(L^4)$.

The next Section IV presents the proposed low complexity analytical modified SD-SDS algorithm. The proposed algorithm exploits the inherent weaknesses of the SE-SD and the SD-SD algorithms to offer a detection algorithm with lower Golden code decoding latency compared to SD-SDS and SE-SD.

**IV. PROPOSED LOW COMPLEXITY ANALYTICAL SPHERE-DECODER**

In this Section we propose a low detection complexity analytical modified Golden code SD-SDS based algorithm. This algorithm is presented in Algorithm 2. We will explain the new concepts as we go along as we explain the workings of Algorithm 2. Before we discuss the workings of Algorithm 2, we will illuminate the salient differences between Algorithm 1 and 2.

- Since the search tree search order for Algorithm 2 is dynamic, unlike in Algorithm 1 where it is fixed, we need the candidate symbol subsets to follow the search tree search order of the M-QAM symbols. The function $sortSubsetDataOrder(\cdot)$ in Algorithm 2 makes sure that the candidate symbol subsets follow the M-QAM symbol search order. The $searchOrder[\cdot]$ array in Algorithm 2 is used to track the search tree search order of the M-QAM symbols.

- Algorithm 2 sorts the wireless channel based on the dynamic search tree search order. Algorithm 1 does not sort the wireless channel since its search tree search order is fixed. The function $sortChannel(\cdot)$ in Algorithm 2 is responsible for sorting the wireless channel based on the search tree search order.

- Algorithm 2 uses the function $unSortOutput(\cdot)$ to restore the M-QAM symbol order to prior sorting so that the decoded output order of M-QAM symbols is predictable. Algorithm 1 has a fixed order of M-QAM symbols output from the sphere decoder; hence it does not need this function.

Algorithm 2 is presented below:

**Algorithm 2: SD-SDS-Descend**

*Input: y, $\hat{H}, \hat{y}, N_t$*

*Output: $i_1, i_2, i_3, i_4$*

1. $\text{global result} \leftarrow [], r^2$
2. $\text{global lattice_point_found_flag}$
3. $x[\cdot] \leftarrow \text{qrDecoder}(y, \hat{H})$
4. $\sigma^2 \leftarrow N_e / \bar{y}$
5. $\text{subsetList}[], \text{searchOrder}[] \leftarrow \text{getSortedSubsets}(x[\cdot], y, \sigma^2)$
6. $\text{orderedSubsets}[] \leftarrow \text{sortSubsetDataOrder(subsetList[], searchOrder[])}$
7. $z', \hat{R}' \leftarrow \text{sortChannel}(y, \hat{H}, \text{searchOrder[]})$
8. $K = 10, N = 2N_t$
9. $r^2 \leftarrow 2\sigma^2 K N - z'^H (I - \hat{R}' \hat{R}')^{-1} \hat{R}'^H) z'$ [13, Eq(28)]
10. $I$ is a $2N_t \times 2N_t$ identity matrix
11. $\text{lattice_point_found_flag} \leftarrow \text{False}$
12. layer $\leftarrow 2N_t, \text{dist} \leftarrow 0$
13. $\text{sphere_decoder}(z', \hat{R}', \text{orderedSubsets}[], \text{dist}, \text{layer})$
14. if $\text{lattice_point_found_flag} \neq \text{False}$ then
15. output $\leftarrow \text{unSortOutput(result[]), searchOrder[]}$
16. $i_4 \leftarrow \text{output}[4], i_3 \leftarrow \text{output}[3], i_2 \leftarrow \text{output}[2], i_1 \leftarrow \text{output}[1]$
17. else
18. $z, R, x[\cdot] \leftarrow \text{qrDecoder}(y, \hat{H})$
19. for $q \leftarrow 1 \text{ to } 4$ do
20. $d \leftarrow []$
21. for $i \leftarrow 1 \text{ to } M$ do
22. $d[i] \leftarrow |x[q] - u_i|^2, \forall u_i \in \Omega_m$
23. end for
24. $i_q \leftarrow \text{argmin}(d[\cdot])$
25. end for
26. end if
27. function $\text{qrDecoder}(y, \hat{H})$
28. $\hat{Q}, \hat{R} \leftarrow \text{qr}(\hat{H})$

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/ACCESS.2022.3141626, IEEE Access
29. \( z \leftarrow \hat{Q}^H y, \hat{x} \leftarrow [ ] \)
30. for \( q \leftarrow 4 \) to 1 do
31. \( \hat{x}[q] \leftarrow \frac{\langle z[q]\rangle - \Sigma_{4}^{q} R[q][i][i]}{R[q][q]} \)
32. end for
33. return \( \hat{x} [ ] \)
34. function getSortedSubsets(\( \hat{x} [ ] \), \( y \), \( \sigma^2 \))
35. subsetList \( \leftarrow [ ] \), dMinList \( \leftarrow [ ] \)
36. grad \( \leftarrow [2.4,1.8,1.6,1.4,1.2,1.0,0.8,0.4] \)
37. intercept \( \leftarrow [5.4,3.2,1.0,-1,-2] \)
38. order \( \leftarrow \text{"descend"} \)
39. for \( q \leftarrow 1 \) to 4 do
40. dict \( \leftarrow [ ] \)
41. for \( i \leftarrow 1 \) to \( M \) do
42. \( \text{dict}[i] \leftarrow \langle \hat{x}[q] \rangle - u_i^2, \forall u_i \in \Omega_M \)
43. end for
44. indices, distances \( \leftarrow \text{sort(dict[ ])} \)
45. \( d_{\min} \leftarrow \text{distances[1]} \)
46. if \( \gamma \leq \gamma_M \) then
47. \( L \leftarrow L_{M_{\text{low}}} \)
48. else
49. \( L \leftarrow L_{M_{\text{high}}} \)
50. end if
51. \( S = L \)
52. for \( vv \leftarrow 1 \) to 8 do
53. if \( d_{\min} < \text{grad[vv]} \sigma^2 \) then
54. \( S \leftarrow L + \text{intercept[vv]} \)
55. end if
56. end for
57. subsetList[q] \( \leftarrow \text{indices[1:S]} \)
58. dMinList[q] \( \leftarrow d_{\min} \)
59. end for
60. qList[ ] \( \leftarrow \text{sort(dMinList[ ], order)} \)
61. searchOrder[ ] \( \leftarrow qList[ ] \)
62. return subsetList[ ], searchOrder[ ]

Algorithm 2 explains the workings of the worst-first search algorithm denoted by SD-SDS-Descend. Like in Algorithm 1, the SD-SDS-Descend algorithm determines the suboptimal M-QAM estimates from the QR decoder. These M-QAM estimates are then used to sort the candidate M-QAM symbols, from the M-QAM constellation, in ascending order from the closest candidate symbol to the estimated M-QAM symbol to the furthest. The candidate symbols are sorted for each M-QAM symbol estimate. \( \hat{x}_q, \forall q \in [1:4] \). The metric used for sorting in ascending order is the Euclidean distance between the estimated or noisy M-QAM symbol and the exact M-QAM symbol from the M-QAM constellation, i.e. \( d_i^q = |\hat{x}_q - u_i|^2, \forall u_i \in \Omega_M, \forall q \in [1:4], \forall i \in [1:M] \). Now, to set the candidate symbol subset lengths based on the instantaneous channel conditions experienced by each estimated M-QAM symbol, we first define a metric that considers the instantaneous fading power and noise power. The metric is the distance between the closest M-QAM complex symbol to the estimated M-QAM complex symbol, i.e. \( d_{\min}^q \triangleq \min |\hat{x}_q - u_i|^2, \forall u_i \in \Omega_M, \forall q \in [1:4] \). This minimum distance metric, \( d_{\min}^q, \forall q \in [1:4] \), is defined for each estimated M-QAM symbol. We make the following assumptions to show that the distance metric, \( d_{\min}^q \), is a function of the instantaneous wireless fading power and the noise power. We will carry out the proof assuming that the previously estimated M-QAM symbols from the QR decoder are estimated without error. In the QR decoder, we estimate the M-QAM symbols in the order, \( \hat{x}_q, \forall q \in [4:1] \). \( \hat{x}_1, \hat{x}_3, \ldots, \hat{x}_q \). To estimate the complex M-QAM symbol, \( \hat{x}_q \) using the QR decoder, we use the following expression in (3)

\[
\hat{x}_q = \frac{1}{\overline{\overline{\overline{R_2q}}}q} \left( z_q - \sum_{l=q+1}^{4} R_2q[i] \hat{x}_i \right) \tag{3}
\]

where \( z_q \) is the \( q^\text{th} \) scalar element in the received vector \( z \) and \( \overline{\overline{\overline{R_2q}}}q \) is the scalar element in row \( q \) and column \( q \) of the upper triangular matrix \( \overline{\overline{\overline{R}}} \). It is obvious to see from (3) that the subsequent M-QAM symbol estimates are dependent on the previously estimated M-QAM symbols. It then follows that if the previously estimated M-QAM symbols are estimated without error, then the currently estimated M-QAM symbol has its estimation errors dependent only on its own instantaneous wireless channel and noise perturbation, as shown in (4)

\[
\hat{x}_q = \frac{\overline{R_2q} x_q + n_q}{\overline{\overline{\overline{R_2q}}}q} = x_q + \left| \frac{n_q}{\overline{\overline{\overline{R_2q}}}q} e^{i \theta_q} \right| \forall q \in [1:4] \tag{4}
\]

where \( n_q \) is the \( q^\text{th} \) complex scalar element of the noise vector \( \overline{\overline{\overline{Q}}}n \) and \( \overline{\overline{\overline{R_2q}}}q \) is the scalar element in row \( q \) and column \( q \) of the upper triangular matrix \( \overline{\overline{\overline{R}}} \). The terms \( |n_q| e^{i \theta_q} \) and \( |\overline{\overline{\overline{R_2q}}}q| e^{i \theta_q} \) are the complex exponential version of the terms \( n_q \) and \( \overline{\overline{\overline{R_2q}}}q \), respectively. The complex M-QAM symbol \( x_q \) is the exact transmitted symbol. If we take the best-case scenario, which is at high instantaneous SNR, then the closest candidate symbol to the estimated M-QAM symbol is the transmitted symbol. If we assume that the instantaneous SNR is high for all QR decoder estimated M-QAM symbols, then the metric \( d_{\min}^q \) is represented mathematically in (5)

\[
d_{\min}^q = |\hat{x}_q - x_q|^2 = \frac{\left| n_q \right|^2}{\overline{\overline{\overline{R_2q}}}q}, \forall q \in [1:4] \tag{5}
\]

From (5) we can see that at high instantaneous SNR, for all estimated M-QAM symbols, \( d_{\min}^q = f(R_2q, \overline{\overline{\overline{R_2q}}}q) \). Therefore, the metric \( d_{\min}^q \) is a function of the instantaneous noise power and the instantaneous wireless fading power. The random upper triangular matrix entries represent the wireless channel fading. For the low instantaneous SNR scenario with estimation errors in the previously estimated M-QAM symbols, the simplification in (5) does not apply. However,
the metric \( d^q_{\text{min}} \) will still be a function of the noise and wireless channel fading. The noise will be compounded from the previously estimated M-QAM symbols via error propagation.

The metric \( d^q_{\text{min}} \) is used in Algorithm 2 to set the instantaneously varying subset length of the candidate symbol subset for each estimated M-QAM symbol. The subset lengths, \( S_q \), are set based on the heuristic method shown in (6)

\[
S_q = L \quad \text{if} \quad d^q_{\text{min}} < k\sigma^2 \quad \forall q \in [1:4] \tag{6}
\]

where \( L \) is the initial subset length which is set as \( L = 20 \) when the average SNR is at most 16dB, else it is set as \( L = 30 \) for an average SNR above 16dB for the case of 64-QAM. The initial subset length values for 64-QAM are extracted from the SD-SDS algorithm in [16, Table 2]. For the case of 256-QAM, the initial subset length is set to \( L = 80 \) for an average SNR, which is at most 21dB and \( L = 120 \) for an average SNR above 21dB. The constants are set as follows:

\[
k \in \Delta = \{2,4,1.8,1.6,1.4,1.2,1.0,0.8,0.4\}
\]

and \( c \in \beta = \{5,4,3,2,1,0,-1,-2\} \). The function \( \lfloor \cdot \rfloor \) returns the integer nearest to \( \lfloor \cdot \rfloor \) or equal to it. The intuition behind this method is that if the instantaneous SNR is sufficiently high, we set the instantaneous subset lengths to a very short length. If the instantaneous SNR worsens, we increase the instantaneous length of the subset. This is intuitive as we can expect that at high instantaneous SNR, we do not need to search through many candidate symbols as compared to when the instantaneous SNR is low. The gradual shortening of the instantaneous subset lengths as the instantaneous SNR increases, lowers the average decoding latency relative to the SD-SDS algorithm using fixed-length subsets that depend on average channel conditions.

Algorithm 2 sets the instantaneous subset lengths, for each estimated M-QAM symbol, by looping through the set \( \Delta \) to set the constant \( k \) and comparing the metric \( d^q_{\text{min}} \) to each threshold level set as \( k\sigma^2 \). If the metric \( d^q_{\text{min}} \) is not less than any threshold level, then the subset length is set as \( S_q = L \) else if it falls into one of the threshold levels, then the corresponding constant \( c \) value from set \( \beta \) is used to set \( S_q = \lfloor \frac{L}{2} + c \rfloor \). The loop index is used to extract the value of \( c \) from the set \( \beta \).

It is obvious to see that each estimated M-QAM symbol candidate subset has a possibility of having a different subset length, \( S_q \), to the other candidate symbol subsets of the other estimated M-QAM symbols. Using this property of differing instantaneous subset lengths for each candidate symbol subset, we can easily see that it will be beneficial at low SNR to have the subset with the smallest length being used at Layer 1 of the search tree. The layer 1 detection complexity dominates the search tree complexity at low SNR. We, therefore, propose the worst-first search strategy of sorting the search order of the search tree using the descending order of the metric \( d^q_{\text{min}} \). From Algorithm 2 we can see that \( d_{\text{min}} \) is calculated for each value of \( q \in [1:4] \). This means we get the following unordered set of values of the metric \( d_{\text{min}}; \rho = \{d^1_{\text{min}}, d^2_{\text{min}}, d^3_{\text{min}}, d^4_{\text{min}}\} \). We then sort the set \( \rho \) in descending order to get an ordered set \( \delta = \text{sort}(\rho,"descend") \). The descend string is used to signify that we are sorting the items in descending order. For sorting items in ascending order, we use the ascend string. Let us illustrate with an example. Let us assume the unordered set \( \rho = \{d^1_{\text{min}} = 0.5, d^2_{\text{min}} = 2.5, d^3_{\text{min}} = 0.15, d^4_{\text{min}} = 0.9\} \).

The one-to-one correspondence which shows the mapping between the metric and the estimated symbol is as follows

\[
[\mathbf{x}_4 \mapsto d^4_{\text{min}}, \mathbf{x}_3 \mapsto d^3_{\text{min}}, \mathbf{x}_2 \mapsto d^2_{\text{min}}, \mathbf{x}_1 \mapsto d^1_{\text{min}}].
\]

Based on the SD-SDS [16] algorithm, the wireless channel matrix \( \mathbf{H} \) columns are unordered as follows together with the unordered M-QAM transmission vector \( \mathbf{u} \) entries:

\[
\mathbf{H}u = [\mathbf{H}^1 \mathbf{H}^2 \mathbf{H}^3 \mathbf{H}^4][x_1 \quad x_2 \quad x_3 \quad x_4]^T \tag{7}
\]

where \( \mathbf{H}^q \in \mathbb{C}^{2N \times 1} \) are the column vectors of the wireless channel matrix from (2). After sorting the unordered set \( \rho \) in descending order, we get the following ordered set \( \delta = \{d^1_{\text{min}} = 2.5, d^4_{\text{min}} = 0.9, d^3_{\text{min}} = 0.5, d^2_{\text{min}} = 0.15\} \). As we can see, the estimated M-QAM symbol with a metric with the highest value will be searched for first in the search tree. We call it the worst-first search strategy because the metric \( d^q_{\text{min}} \) in (5) appears as an approximate inverse of the instantaneous SNR. Therefore, a good instantaneous SNR will yield a smaller \( d^q_{\text{min}} \) value relative to a bad instantaneous SNR. Using the sorted set \( \delta \) we sort the wireless channel matrix \( \mathbf{H} \) column vectors and the corresponding transmission vector \( \mathbf{u} \) entries as shown in (8)

\[
\mathbf{H}u' = [\mathbf{H}^1 \mathbf{H}^2 \mathbf{H}^3 \mathbf{H}^4][x_3 \quad x_1 \quad x_4 \quad x_2]^T \tag{8}
\]

From (8) we can see that the one-to-one correspondence mapping that maps the estimated M-QAM symbols and the search layers becomes \([\mathbf{x}_2 \mapsto 4, \mathbf{x}_4 \mapsto 3, \mathbf{x}_1 \mapsto 2, \mathbf{x}_3 \mapsto 1]\). Search layer 1 estimates the M-QAM symbol \( \mathbf{x}_3 \) and this is because \( \mathbf{x}_3 \) is mapped to a metric \( d^3_{\text{min}} \) with the smallest value. Based on the heuristic method in (6), there is a high probability that the subset length, \( S_3 \), of the candidate symbol subset for \( \mathbf{x}_3 \) is the smallest amongst the other estimated M-QAM candidate symbol subset lengths. For this example, the subset lengths are sorted as follows \( S_4 \leq S_1 \leq S_3 \leq S_2 \). This then shows why the worst-first search strategy is best at the low average SNR region since the search layer 1 complexity dominates the search tree complexity. The sorted channel matrix \( \mathbf{H}' \) is then used to produce the modified received signal vector \( \mathbf{z}' = \mathbf{Q}'\mathbf{y} \) since \( \mathbf{H}' = \mathbf{Q}'\mathbf{R}' \). The SD-SDS search is then performed on the sorted search tree search order to find
the lattice points, \( \tilde{\mathbf{R}} \mathbf{u}' \), that lie inside the hypersphere \( \| \mathbf{z}' - \tilde{\mathbf{R}} \mathbf{u}' \|^2 \leq r^2 \). When at least 1 lattice point is found inside the hypersphere, the lattice_point_found_flag in Algorithm 2 is set to True. The result array, result[], is populated with the M-QAM symbol indices derived from the most optimal lattice point found inside the hypersphere. The one-to-one correspondence \([\hat{x}_2 \leftrightarrow 4, \hat{x}_4 \leftrightarrow 3, \hat{x}_1 \leftrightarrow 2, \hat{x}_3 \leftrightarrow 1]\) or search order is used to unsort the SD-SDS M-QAM symbol output accordingly. If no lattice points are found inside the hypersphere, then the suboptimal QR decoder is used to determine the M-QAM symbol estimates.

V. PROPOSED DEEP LEARNING-BASED SPHERE-DECODER

This Section proposes a deep learning-based SD-SDS search early stopping algorithm. The early stopping criteria of the SD-SDS search is performed on the early stopping algorithm 1. The idea is that we want to find a suitable mapping between the input and output of a deep neural network (DNN) that can predict when the SD-SDS search must be terminated prematurely. This DNN mapping will take on a structure like a typical DNN shown in Figure 2.

![Typical DNN structure](image)

Figure 2, “Typical DNN structure” [21].

In our case, we want to find a function approximator that maps the instantaneous channel conditions and noise to a binary state that determines whether early termination should take place. When early termination is deemed appropriate, it is carried out as soon as the first lattice point is found inside the hypersphere. When the first lattice point is found inside the hypersphere, the SD-SDS search tree is terminated, and we immediately output the estimated M-QAM symbol indices. There is no need to visit all the unvisited nodes in the search tree to determine if there is a closer lattice point to the received signal vector. This is because we take advantage of the fact that the candidate symbol subsets are sorted so that the most likely transmitted symbols are placed first in the candidate symbol subset. We also take advantage of the depth-first search strategy that produces lattice points quicker than the breadth-first search. The job of the DNN is to learn the channel conditions that necessitate early termination.

We define the DNN function approximator as shown in (9)

\[
\mu = \Phi(\omega, \tau)
\]

where \( \mu \) is the probability of initiating early termination, \( \omega \) is the input vector of the DNN function approximator, and \( \tau \) is the vector of the DNN model parameters that need tuning during offline training. We further define the input vector of the DNN as \( \omega \triangleq [\mathbf{2}_R, \mathbf{2}_I, \mathbf{r}_1^T, \mathbf{r}_2^T] \in \mathbb{R}^{2B} \) where \( \mathbf{2}_R \triangleq (\Re(\mathbf{R(z)}))^T \), \( \mathbf{2}_I \triangleq (\Im(\mathbf{R(z)}))^T \), \( \mathbf{r}_1 \triangleq \text{vec}(\Re(\mathbf{R}))^T \) and \( \mathbf{r}_2 \triangleq \text{vec}(\Im(\mathbf{R}))^T \). The input vector is a 28-dimensional vector in our case since we will simulate over a 2 \( \times \) 4 MIMO wireless channel. For a generic \( N_t \times N_r \) MIMO wireless channel, the input vector dimension will vary depending on the number of non-zero entries found in the input vector. The input vector-only considers non-zero entries. As can be seen, the DNN uses instantaneous wireless channel fading and noise statistics to determine when early termination is suitable. The instantaneous noise statistics are indirectly catered for via the received signal vector \( \mathbf{z} \). The instantaneous wireless channel fading is represented using the random upper triangular matrix \( \mathbf{R} \), which is related to the wireless channel matrix \( \mathbf{H} \) via the reduced QR factorization \( \mathbf{H} = \mathbf{QR} \).

The DNN function approximator \( \Phi(\cdot) \) has an architecture shown in Table 1.

| Layer | Parameter Description |
|-------|-----------------------|
| Input (28) | 28 dimensional input vector \( \omega \) |
| (Input Layer) | 128 neurons. LeakyReLU slope of 0.3 |
| Dense+LeakyReLU | void |
| BatchNormalization | Probability of 0.5 |
| Dropout | (Hidden Layer 1) 128 neurons. LeakyReLU slope of 0.3 |
| Dense+LeakyReLU | void |
| BatchNormalization | Probability of 0.5 |
| Dropout | (Output Layer) 1 neuron |
| Dense+Sigmoid | 1 neuron |

For 64-QAM: Batch Size=1428, Learning Rate=0.00312, Pseudo random seed value=70 and Maximum Epochs=2000

For 256-QAM: Batch Size=8000, Learning Rate=0.0001, Pseudo random seed value=70 and Maximum Epochs=2000

The 64-QAM and 256-QAM DNN training learning rate, pseudo-random seed value, and batch size are determined using a meta-heuristic Genetic algorithm [22] with a fitness function dependent on the validation accuracy metric. As shown in Table 1, the DNN architecture does not increase in size based on the M-QAM modulation order \( M \) unlike in [20]. Our architecture is thus suitable for a high-density M-QAM Golden code environment as the DNN architecture inference time will not increase as the M-QAM modulation order increases. The DNN architecture in Table 1 is only valid for the 2 \( \times \) 4 MIMO wireless configuration implying that any other MIMO configuration will require the re-designing and training of a new architecture.
A. OFFLINE TRAINING OF THE DNN

The DNN architecture in Table 1 is trained once offline but separately for the 64-QAM and 256-QAM scenarios. The training sample sizes are set to 20000 and 80000 for the case of 64-QAM and 256-QAM, respectively. The training samples are collected by logging the values of the instantaneous wireless fading channel and noise statistics using the definition of the input vector \( \omega \equiv [\tilde{z}, \tilde{z}, \tilde{r}, \tilde{r}] \) and logging the corresponding output labels since this is supervised learning. The input vector features are each normalized into the range of [0,1] to aid good training performance by putting all features into the same scale. We prematurely terminate the SD-SDS search for each wireless channel realization when we find the first lattice point inside the hypersphere. This sole lattice point is used to deduce the estimated M-QAM symbols. If all the estimated M-QAM symbols match the transmitted M-QAM symbols, i.e \( \hat{x}_q = x_q \) \( \forall q \in [1:4] \), then the output label is set to an integer value 1 else it is set to 0. We then use this training data, with a training-to-test ratio split of 75:25, to train and evaluate the DNN architecture in Table 1. However, before training the DNN architecture in Table 1, we observe that the output labels will be unbalanced. At high SNR, we can expect most of the output labels to be 1 and at low SNR to be 0. This is because at high average SNR, the high instantaneous SNR is more frequent, and thus we can expect that the first lattice point, found inside the hypersphere, produces the correct estimates of the transmitted symbols. At low average SNR, we can expect that the high instantaneous SNR frequency is low, and hence the first lattice point found inside the hypersphere, will rarely produce reliable symbol estimates. Figures 3 and 4 illustrate this observation for the distribution of output label values for the 64-QAM and 256-QAM scenarios.

![Figure 3, “64-QAM output label distribution”](image)

![Figure 4, “256-QAM output label distribution”](image)

B. ONLINE DECODING PROCESS

The tuned model parameters in vector \( \tau \) and the DNN architecture in Table 1 are then saved and deployed in the simulation environment for online SD-SDS search tree early termination prediction. To determine whether early termination of the SD-SDS search is desirable, under specific wireless channel and noise conditions, we feed the trained DNN function approximator with the online input vector \( \omega \) with normalized features. The input vector \( \omega \) contains the instantaneous wireless channel realization. The output of the DNN will be a probability value \( \mu \in [0,1] \in \mathbb{R} \) that is then compared to a fixed threshold value \( \mu_t \). The threshold values for 64-QAM and 256-QAM are documented in Table 2 and are found using a heuristic approach that balances the decoding latency reduction and the BER performance at different SNR values.
TABLE 2: 64-QAM AND 256-QAM PROBABILITY_THRESHOLDS

| Probability Thresholds | 64-QAM Parameters | 256-QAM Parameters |
|------------------------|-------------------|--------------------|
| \( \tilde{\gamma} \leq 16 \text{ dB} \) | \( \mu_1 = 0.2 \) | \( \mu_1 = 0.25 \) |
| \( \tilde{\gamma} > 16 \text{ dB} \) | \( \mu_1 = 0.5 \) | \( \mu_1 = 0.65 \) |
| \( \tilde{\gamma} \leq 21 \text{ dB} \) | \( \mu_2 = 0.2 \) | \( \mu_2 = 0.25 \) |
| \( \tilde{\gamma} > 21 \text{ dB} \) | \( \mu_2 = 0.5 \) | \( \mu_2 = 0.65 \) |

The rule for prematurely terminating the SD-SDS search is straightforward. For every probability value \( \mu \geq \mu_t \) we prematurely terminate the SD-SDS search on the first encounter of a lattice point that lies inside the hypersphere. If the probability \( \mu < \mu_t \), then we continue the SD-SDS search despite finding the first lattice point inside the hypersphere. This early termination technique prevents the unnecessary execution of the Euclidean distance calculations in each of the unvisited nodes in the search tree. This has the obvious effect of lowering the decoding latency of the SD-SDS search while achieving near-optimal error rate performances. The algorithm described in this section is named as SD-SDS-ES-DNN.

VI. SIMULATION RESULTS AND DISCUSSION

The Monte-Carlo simulation for uncorrelated wireless channels is performed for the \( 2 \times 4 \) wireless MIMO configuration. The number of transmit antennas in the MIMO configuration is \( N_t = 2 \), and the number of receive antennas is \( N_r = 4 \). The high-density Golden code M-QAM modulation orders considered for this simulation are the 64-QAM and 256-QAM variants. The average M-QAM symbol power and Golden code super symbol power is set to 1. The Monte-Carlo simulation only shows the low complexity Golden code SD-based detection algorithms from literature versus our proposed SD-based algorithms. The performance comparison between the SE-SD [12], SD-SDS [16], and our proposed algorithms is done using the error rate performance, average decoding latency, and the average number of lattice points found inside the hypersphere. The average decoding latency is measured for each algorithm under the same computer platform.

A. COMPLEXITY ANALYSIS

The detection complexity is assessed using the simulated average decoding latency and the average number of lattice points found inside the hypersphere. From Figure 5, we see that the SD-SDS-ES-DNN algorithm has the lowest average number of lattice points found inside the hypersphere at lower SNR. This is because the early termination algorithm terminates the SD-SDS search under good instantaneous channel conditions after finding exactly 1 lattice point inside the hypersphere. There is no difference in performance at high SNR because all the algorithms find their most optimal lattice point as the first lattice point inside the hypersphere at high instantaneous SNR. This is because the candidate symbol subset has M-QAM symbols sorted so that the most likely transmitted symbols are placed first in the subset. This coupled with the depth-first search strategy, yields a high probability, at high SNR, of finding the most optimal lattice point as the first lattice point inside the hypersphere. We must remember that the moment the SD-SDS search tree finds a lattice point closer to the received signal vector, the search tree updates the hypersphere radius to the distance of this lattice point to the received signal vector. If the first lattice point found inside the hypersphere is the most optimal or closest lattice, no other lattice points will be found inside the hypersphere.

Figure 5, "64-QAM Golden code SD-based detection algorithms average number of lattice points inside hypersphere for \( 2 \times 4 \) MIMO''.

Figure 6, "64-QAM Golden code SD-based detection algorithms average decoding latency for \( 2 \times 4 \) MIMO''.

Figure 6 exhibits the Golden code SD-based detection algorithms' average decoding latency. At low SNR, we observe that the SD-SDS-Descend algorithm, which is the proposed Algorithm 2 in this paper, has the lowest average decoding latency with at most 57% reduction in decoding latency relative to the SD-SDS from literature. At high SNR, the SD-SDS-Descend and SD-SDS-Ascend algorithms produce similar decoding latency reduction of at most 40% relative to SD-SDS. At high SNR, the source of decoding latency reduction is the smaller instantaneous subset lengths generated by the heuristic method in Eq.(6) of this paper. This is because the SD-SDS-Descend and SD-SDS-Ascend algorithms use the instantaneous subset lengths, which will
be in the range \( S_q \in [13:20] \) for 64-QAM at high SNR. Despite the narrow instantaneous subset length range, it is approximately 66% of the subset length used by the SD-SDS algorithm, which uses fixed-length subsets of 30 candidate symbols for 64-QAM at high SNR. Because at high instantaneous SNR the instantaneous subset lengths are shorter than that required at high average SNR by the SD-SDS with fixed-length subsets, the search tree search breadth is shortened and thus decoding latency is lowered. The BER performance is not affected because this shortening of subset lengths only occurs at sufficiently high instantaneous SNR as per Eq. (6). The SD-SDS-Ascend algorithm is just the reverse of the SD-SDS-Descend because the search tree search order is sorted in ascending order using the defined metric in this paper. Despite the SD-SDS-ES-DNN having the lowest average number of lattice points inside the hypersphere, at low SNR, it has a greater decoding latency than the SD-SDS-Descend algorithm because of the effect of its DNN architecture inference time. The SD-SDS-Descend algorithm has lower decoding latency at low SNR than the other analytical decoding algorithms because it orders the search tree search order such that search tree layer 1 has the smallest candidate symbol subset. The search tree layer 1 dominates the search tree complexity at low SNR, therefore, assigning it a subset with the smallest length lowers the decoding latency of the search tree. The SE-SD algorithm has the worst decoding latency because it uses the full signal cardinality of the M-QAM constellation to search for the optimal solution. The other algorithms use candidate symbol subsets shorter than the M-QAM signal constellation cardinality. The SD algorithms are known to have their decoding complexity dependent on the search signal cardinality and search depth of the search tree [14].

Figure 5. However, at high SNR, the SD-SDS-ES-DNN algorithm has the same performance as the analytical SD algorithms. This is because, at high SNR, the SD-SDS-ES-DNN is virtually not prematurely terminating the SD-SDS search because of the error rate performance sensitivity at high SNR. The SD-SDS-ES-DNN algorithm relies on a DNN output probability to activate the early termination based on the wireless channel quality. Because of the inevitable prediction errors present in the DNN output, there are times when the DNN erroneously outputs a probability that enables premature termination of the SD-SDS search in unfavorable channel conditions. This will negatively impact the error rate performance at high SNR. To counter this, the SD-SDS-ES-DNN algorithm infrequently prematurely terminates the SD-SDS search to maintain the near-optimal error rate performance at high SNR.

Figure 8 exhibits that the proposed SD-SDS-Descend algorithm has the lowest decoding latency among the analytical SD-based algorithms, at low SNR, for the same reasons advanced in the case of 64-QAM modulation in Figure 6.

![Figure 7. "256-QAM Golden code SD-based detection algorithms average number of lattice points inside hypersphere for 2 × 4 MIMO".](image)

The difference here is that for the 256-QAM case, the SD-SDS-ES-DNN algorithm has the lowest decoding latency, at low SNR, compared to all other algorithms. A decoding latency reduction of 70% is achieved relative to the SD-SDS algorithm at low SNR. This is because, for 256-QAM, the analytical SD-based algorithms visit all the unvisited nodes in the large search tree. The 256-QAM search tree is larger than the 64-QAM tree because the signal cardinality or search breadth is larger for 256-QAM. Whether the instantaneous SNR is good or not, the analytical SD-based decoding algorithms visit unvisited tree nodes to determine if a more optimal lattice point can be found inside the hypersphere. The SD-SDS-ES-DNN algorithm prematurely terminates the search when it finds 1 lattice point inside the hypersphere under good instantaneous SNR conditions. This lowers the decoding latency. Over and above this, for the 256-QAM case, the DNN architecture remains the same as
that for the 64-QAM case. This implies that the DNN inference time has a marginal effect on the decoding latency for the case of 256-QAM. At high SNR, the SD-SDS-ES-DNN algorithm has a decoding latency that matches the SD-SDS algorithm decoding latency. This is because, at high SNR, the SD-SDS-ES-DNN algorithm performs virtually no early termination of the SD-SDS search. After all, it needs to maintain a near-optimal BER performance at the expense of increased decoding latency. At high SNR, the SD-SDS-Descend and SD-SDS-Ascend algorithms achieve approximately 37% decoding latency reduction relative to the SD-SDS algorithm from the literature. This is because the SD-SDS-Descend and SD-SDS-Ascend algorithms use the instantaneous subset lengths in the range $S_q \in [58:65]$ for 256-QAM at high SNR. Despite the narrow instantaneous subset length range, it is approximately 50% of the subset length used by the SD-SDS algorithm, which uses fixed-length subsets of 120 candidate symbols for 256-QAM at high SNR. Because at high instantaneous SNR the instantaneous subset lengths are shorter than that required at high average SNR by the SD-SDS with fixed-length subsets, the search tree search breadth is shortened and thus decoding latency is lowered. The BER performance is not affected because this shortening of subset lengths only occurs at sufficiently high instantaneous SNR as per Eq.(6). The SE-SD [12] algorithm still exhibits the worst decoding latency performance relative to the proposed algorithms and the SD-SDS [16] algorithm.

**B. ERROR RATE PERFORMANCE**

![Figure 9](image9.png) Figure 9, "64-QAM Golden code SD-based detection algorithms error rate performance for 2 × 4 MIMO".

Figures 9 and 10 show that the proposed SD-SDS-Descend and SD-SDS-ES-DNN algorithms achieve near-optimal BER performances despite reducing decoding latency relative to the state-of-the-art low complexity Golden code detection algorithms SE-SD [12] and SD-SDS [16].

![Figure 10](image10.png) Figure 10, "256-QAM Golden code SD-based detection algorithms error rate performance for 2 × 4 MIMO".

**VII. CONCLUSION**

In this paper, we successfully proposed a more appropriate channel quality metric to sort the SD-SDS search tree search order. The channel quality metric considered both the instantaneous wireless channel fading power and the instantaneous noise power. The SE-SD search order sorting metric, in literature, considered only the instantaneous wireless channel fading power. This is not an accurate assessment of channel quality as noise statistics dominate the SNR performance at low SNR instead of fading. We also proposed instantaneously varying candidate symbol subset lengths per search layer. The candidate symbol subset lengths varied with the instantaneous channel conditions for each estimated M-QAM symbol and allowed the search tree search order to be sorted based on the subset lengths. This led to the proposal of the worst-first search strategy, which was employed by the detection algorithm SD-SDS-Ascend. The SD-based search trees have their detection complexity dominated by the search layer 1 detection complexity at low SNR. The worst-first search strategy ensured that the candidate symbol subset with the smallest subset length always got assigned to search layer 1. This assisted the SD-SDS-Descend algorithm to achieve a reduction in decoding latency of 57% relative to the SD-SDS algorithm for the case of 64-QAM modulation at low SNR. The paper also proposed a deep learning-based early termination algorithm, i.e. SD-SDS-ES-DNN, for low complexity SD-SDS small MIMO. For 256-QAM, the SD-SDS-ES-DNN algorithm achieved 70% reduction in decoding latency at low SNR relative to the SD-SDS algorithm proposed in the literature. The SD-SDS-Descend algorithm achieved 40% and 37% decoding latency reduction relative to SD-SDS, at high SNR, for the case of 64-QAM and 256-QAM, respectively. All these gains were shown to be achieved without losing any error rate performance relative to the near-optimal BER performances of SE-SD and SD-SDS.
REFERENCES

[1] L. Zheng and D. N. C. Tse, “Diversity and Multiplexing: A Fundamental Trade-off in Multiple-Antenna Channels,” IEEE Trans. Inform. Theory, vol. 49, no. 5, pp. 1073-1096, May 2003.

[2] J. Belfiore, G. Rekaya and E. Viterbo, “The golden code: a 2 x 2 full-rate space-time code with non-vanishing determinants”, International Symposium on Information Theory, ISIT 2004. Proceedings., Chicago, IL, 2004, pp. 310-310, 2004.

[3] V. Torakh, H. Jafarkhani, A. R. Calderbank, “Space–Time Block Codes from Orthogonal Designs”, IEEE Transactions on Information Theory, vol. 45, no. 5, July 1999.

[4] M. O. Sinnokrot and J. R. Barry, “Fast maximum-likelihood decoding of the golden code”, IEEE Trans.Wireless Commun., vol. 9, no. 1, pp. 26-31, 2010.

[5] S. M. Alamouti, “A simple transmit diversity technique for wireless communications”, IEEE Journ. Sel. Areas Commun., vol. 16, no. 8, pp.1451-1458, 1998.

[6] H. Xu, K. Govindasamy, and N. Pillay, “Uncoded space-time labelling diversity”, IEEE Commun. Lett., vol. 20, no. 8, pp. 1511-1514, 2016.

[7] F. Riera-Palou and G. Femenias, “Improving STBC Performance in IEEE 802.11n Using Group-Orthogonal Frequency Diversity”, IEEE Wireless Communications and Networking Conference, Las Vegas, NV, pp. 193-198, 2008.

[8] N. Sharma, “Space Time Block Code for Next Generation Multi-user MIMO System”, 9th International Conference on Future Networks and Communications, Procedia Computer Science, pp. 172-179, 2014.

[9] IEEE 802.16e-2005: IEEE Standard for Local and Metropolitan Area Network - Part 16: Air Interface for Fixed and Mobile Broadband Wireless Access Systems - Amendment 2: Physical Layer and Medium Access Control Layers for Combined Fixed and Mobile Operation in Licensed Bands, Feb. 2006.

[10] S. Kahraman and M. E. Çelebi, "Dimensionality reduced decoding for the golden code with the worst-case complexity of O(m^3) for low range of SNR," IEEE Wireless Communications and Networking Conference (WCNC), 2012, pp. 246-250.

[11] S. Sirinaunpiboon, A. R. Calderbank, and S. D. Howard, “Fast essentially maximum likelihood decoding of the Golden code”, IEEE Trans. Inf. Theory, vol. 57, no. 6, pp. 3537-3541, 2011.

[12] L. Zhang, B. Li, T. Yuan, X. Zhang, and D. Yang, “Golden code with low complexity sphere decoder”, in Proc. 18th Int. Symp. Pers. Indoor Mobile Radio Commun., pp. 1-5, 2007.

[13] B. M. Hochwald and S. ten Brink, “Achieving near-capacity on a multiple-antenna channel”, IEEE Trans. on Commun., vol. 51, no. 3, pp. 389-399, 2003.

[14] J. Jaldén and B. Ottersten, “On the complexity of sphere decoding in digital communications”, IEEE Trans. Signal Process., vol. 53, no. 4, pp. 1474-1484, 2005.

[15] H. Xu and N. Pillay, “Reduced Complexity Detection Schemes for Golden Code Systems”, IEEE Access, vol. 7, pp. 139140-139149, 2019.

[16] H. Xu and N. Pillay, “Multiple Complex Symbol Golden Code”, IEEE Access, vol. 8, pp. 103576-103584, 2020.

[17] A. Askri and G. R. Othman, “DNN assisted Sphere Decoder”, 2019 IEEE Int. Symposium on Inf. Theory (ISIT), Paris, France, pp. 1172-1176, 2019.

[18] M. Mohammadkarimi, M. Mehrabi, M. Ardakani and Y. Jing, “Deep Learning-Based Sphere Decoding”, IEEE Trans. on Wireless Commun., vol. 18, no. 9, pp. 4368-4378, 2019.

[19] N. T. Nguyen, K. Lee and H. Dai, "Application of Deep Learning to Sphere Decoding for Large MIMO Systems," in IEEE Transactions on Wireless Communications, doi: 10.1109/TWC.2021.3076527.

[20] D. Weon and K. Lee, "Learning-Aided Deep Path Prediction for Sphere Decoding in Large MIMO Systems," in IEEE Access, vol. 8, pp. 70870-70877, 2020.

[21] M.A. Nielsen, “Neural Networks and Deep Learning”, Determination Press, 2015.

[22] D. E. Goldberg, “Genetic Algorithms in Search, Optimization and Machine Learning”, Reading, MA, USA: Addison-Wesley, 1989.

[23] N. V. Chawla, K. W. Bowyer, L. O. Hall, W. P. Kegelmeyer, “SMOTE: Synthetic Minority Over-sampling Technique”, Journal of Artificial Intelligence Research, Vol. 16, 2002.

[24] D. B. J. Kingma, “Adam, a method for stochastic optimization”, 2014.

MR BHEKISIZWE MTHETHWA received the BScEng Electronics degree from the University of KwaZulu-Natal in 2008 and the MScEng Electronics degree in 2013 from the same institution. He has over 10 years industrial experience ranging from Mining consulting as a Control and Instrumentation Engineer to a senior system engineer within the Research and Development space. He has been involved in projects from designing and constructing furnaces, software application development and system engineering work. He is a registered Professional Engineer with the Engineering Council of South Africa (ECSA). Mr Mthethwa’s research interests include Spatial Modulation, MIMO Space-Time Block Coding schemes and the application of Deep Learning to the design of wireless systems. Mr Mthethwa is currently pursuing his PhD degree under the guidance of Dr. H. Xu. Mr Mthethwa is also a reviewer for the Physical Communication Journal and IEEE ACCESS.
DR HONGJUN XU (MIEEE, 07) received the B.Sc. degree from the Guilin Institute of Electronic Technology (nowadays it is called Guilin University of Electronic Technology), Guilin, China, in 1984; the M.Sc. degree from the Institute of Telecontrol and Telemeasure, Shi Jian Zhuang, China, in 1989; and the Ph.D. degree from the Beijing University of Aeronautics and Astronautics, Beijing, China, in 1995. From 1997 to 2000, Dr. Xu was a Postdoctoral Fellow with the University of Natal (Nowadays it is called University of KwaZulu-Natal), Durban, South Africa, and Inha University, Incheon, Korea. Dr. Xu joined the University of Natal in 2002 as a lecturer. Dr. Xu has been a full Professor with the School of Engineering at the University of KwaZulu-Natal since 2011. Dr. Xu is also a rated scientist of the national research foundation in South Africa.