An Improved Algorithm for Optimal Noncoherent QAM Detection

Robby G. McKilliam[1], Daniel J. Ryan[2,3], I. Vaughan L. Clarkson[1] and Iain B. Collings[3]

[1] School of Information Technology and Electrical Engineering, University of Queensland, AUSTRALIA, robertm@itee.uq.edu.au
[2]School of Electrical and Information Engineering, University of Sydney, AUSTRALIA
[3] Wireless Technologies Laboratory, CSIRO ICT Centre, AUSTRALIA

***Abstract*—Ryan *et al.* [1] recently described two polynomial time algorithms for noncoherent detection of square QAM in block fading channels with additive white Gaussian noise (AWGN). The first algorithm is optimal with respect to the generalized likelihood ratio test (GLRT) and requires $O(T^3)$ arithmetic computations, where T is the block length of the noncoherent receiver. The second algorithm requires only $O(T^2 \log T)$ arithmetic computations but is statistically suboptimal. This paper derives a new algorithm that is optimal yet requires only $O(T^2 \log T)$ arithmetic computations. The new algorithm has the geometric interpretation of finding the nearest codeword to a plane (2 dimensional subspace). The nearest codeword is found by testing codewords that are near a finite number of lines formed by the intersection of the plane and the nearest neighbour boundaries of the codewords.**

***Index Terms*—Noncoherent detection, lattice decoding, wireless communications.**

I. Introduction

Noncoherent transmission of digital signals over unknown fading channels has recently received significant attention in particular for the case of the block-fading channel model [1]. Noncoherent transmission is applicable to systems exhibiting small coherence intervals where the use of training signals would result in a significant loss in throughput. Such situations typically occur in mobile communications. Moreover, as indicated by Chen *et al.* [2] standard coherent detection is, in a sense, inherently suboptimal because it only uses the energy of a small number of pilot symbols for channel estimation, rather than also exploiting the (typically larger) energy in the unknown data symbols.

Various suboptimal algorithms have been proposed for noncoherent detection for particular transmission schemes [3]–[5]. These techniques, although suboptimal, are still exponential in complexity. The challenge is to develop non coherent detection schemes with practical complexity.

Recently, lattice decoding algorithms have also been applied to noncoherent detection [1], [6]. In particular Ryan *et al.* [1] recently described a polynomial time algorithm for optimal noncoherent reception of square QAM in complex-valued channels with added Gaussian noise. The algorithm is optimal with respect to the generalized likelihood ratio test (GLRT). The algorithm has also found application for efficient beamforming in limited feedback wireless MIMO communications [7]. The algorithm requires $O(T^3)$ computations, where T is the number of QAM symbols used by the noncoherent receiver, the *block-length*. We refer to this algorithm as the *Lattice Based Optimal Detector* (LBOD). The algorithm can be geometrically interpreted as a search for the closest codeword in angle to a plane (2 dimensional subspace).

Ryan *et al.* also describe a suboptimal algorithm that requires $O(T^2 \log T)$ calculations. We refer to the suboptimal algorithm as the *Lattice Based Sub-optimal Detector* (LBSD). The LBSD algorithm also seeks to find the closest codeword in angle to a plane, however, this is achieved by testing codewords that are close to a finite number of lines that lie in the plane and radiate from the origin. The number of lines used is chosen arbitrarily and it is not known how many are needed to guarantee that the optimal codeword is found. In practice, LBSD performs statistically very similarly to LBOD, provided that the number of lines used is sufficient. LBSD is significantly more computationally efficient than LBOD.

In this paper we describe an optimal algorithm that requires only $O(T^2 \log T)$ arithmetic computations compared to the $O(T^3)$ arithmetic computations required for LBOD. We refer to the new algorithm as the *Fast Optimal Detector* (FOD). FOD is a modification of the LBSD algorithm, where, rather than using lines that radiate from the origin, we use lines that are formed by the intersection of the plane with the nearest neighbour boundaries of the codewords. This produces lines that criss-cross the plane. We prove that this approach is optimal whilst still maintaining the complexity of the previous suboptimal algorithm, LBSD. We show via Monte Carlo simulations that the new algorithm is much faster than the previous optimal algorithm (LBOD) and is comparable to that of the suboptimal approach.

The paper is organized into the following sections. Section II describes the signal model and defines the objective function for GLRT-optimal noncoherent detection. A geometric interpretation for the objective function is given and some limitations of the objective function in the form of ambiguities are described. Section III introduces useful definitions and derives some preliminary results. Section IV briefly describes LBOD and LBSD. Section V describes the new algorithm, FOD. Its complexity is shown to be $O(T^2 \log T)$ and pseudocode is given. Section VI provides simulation results that compare FOD, LBOD and LBSD in practice.

II. System Model

A. Signal Model

Following [1] we define a codebook $C(\mathcal{X}, T)$ as the set of all possible sequences of T transmitted symbols, $\boldsymbol{x} = [x_1, ..., x_T]^T$ such that each x_t is in some constellation $\mathcal{X}$. Here, vectors and matrices are written in bold. The vector or matrix transpose is indicated by a superscript T such as $\boldsymbol{x}^T$. This is not to be confused with the variable T that is used to represent the block length of the noncoherent detector. The tth element of a vector is indicated with a subscript like x_t and is not written in bold.

In this paper, $\mathcal{X}$ will always be the M^2-ary square QAM constellation. However, the theory can be extended to rectangular QAM without difficulty. Following [1] we define an M^2-ary QAM symbol to be a Gaussian (complex) integer with odd real and imaginary components in the range $[-M+1, M-1]$.

We consider a codeword $\boldsymbol{x} \in C(\mathcal{X}, T)$ transmitted in the block fading AWGN channel

$$\boldsymbol{y} = h\boldsymbol{x} + \boldsymbol{\eta} \tag{1}$$

where h is a complex scalar, $\boldsymbol{\eta}$ is white complex Gaussian noise and $\boldsymbol{y}$ is the received signal.

B. Objective Function

We use the underscore notation $\underline{\boldsymbol{x}}$ to denote

$$\underline{\boldsymbol{x}} = [\text{Re}\{x_1\}, \text{Im}\{x_1\}, ..., \text{Re}\{x_T\}, \text{Im}\{x_T\}]^T. \tag{2}$$

Hence we can denote the real-valued codebook as

$$C_R(\mathcal{X}', 2T) = \{\underline{\boldsymbol{x}} | \boldsymbol{x} \in C\}$$

For M^2-ary square QAM, we have $\underline{\boldsymbol{x}} \in C_R(\mathcal{X}', 2T)$ where $\mathcal{X}'$ is an M-ary PAM constellation. For the remainder of the paper we abbreviate $C_R(\mathcal{X}', 2T)$ to C_R.

The GLRT-optimal estimate of $\boldsymbol{x}$ is

$$\hat{\underline{\boldsymbol{x}}}^{\text{opt}} = \arg \max_{\hat{\underline{\boldsymbol{x}}} | \hat{\boldsymbol{x}} \in C(\mathcal{X}, T)} \cos^2 \theta(\hat{\underline{\boldsymbol{x}}}, \boldsymbol{P}(\boldsymbol{y})\hat{\underline{\boldsymbol{x}}}) \tag{3}$$

where $\theta(\boldsymbol{x}, \boldsymbol{y})$ is the principal angle between vectors $\boldsymbol{x}$ and $\boldsymbol{y}$, $\boldsymbol{P}(\boldsymbol{y})$ is the projection matrix

$$\boldsymbol{P}(\boldsymbol{y}) = \frac{\underline{\boldsymbol{Y}}\underline{\boldsymbol{Y}}^T}{||\boldsymbol{y}||^2}$$

and $\underline{\boldsymbol{Y}}$ is the matrix

$$\underline{\boldsymbol{Y}} = \begin{bmatrix} \text{Re}\{y_1\} & \text{Im}\{y_1\} & ... & \text{Re}\{y_T\} & \text{Im}\{y_T\} \\ -\text{Im}\{y_1\} & \text{Re}\{y_1\} & ... & -\text{Im}\{y_T\} & \text{Re}\{y_T\} \end{bmatrix}^T. \tag{4}$$

Two forms of ambiguity exist for this noncoherent detection problem. The first is the phase ambiguity which occurs for any constellation that is invariant to a particular phase rotation. For square QAM the following four optimal channel estimates and codeword pairs have the same likelihood, $(h^{\text{opt}}, \hat{\boldsymbol{x}}^{\text{opt}})$, $(-h^{\text{opt}}, -\hat{\boldsymbol{x}}^{\text{opt}})$, $(-ih^{\text{opt}}, i\hat{\boldsymbol{x}}^{\text{opt}})$ and $(ih^{\text{opt}}, -i\hat{\boldsymbol{x}}^{\text{opt}})$. These correspond to rotations of the codeword by $e^{i\pi/2}$. This is called the $\pi/2$ *phase ambiguity*. As in [1], we will assume that this form of ambiguity can be resolved by using the phase of the last symbol from the previous codeword [2], [4], or by using differential encoding [5]. The second form of ambiguity occurs when points in C_R occur on the same 1-dimensional complex subspace eg. $[1, 1, 1, 1]$, $[3, 1, 3, 1]$ and $[3, 3, 3, 3]$ for 16-ary QAM with block length $T = 2$. This produces a lower bound on the noncoherent block detection error rate [8].

The objective function (3) has the following geometric interpretation: $\hat{\underline{\boldsymbol{x}}}^{\text{opt}}$ is the codeword in C_R of minimal angle to the plane $\underline{\boldsymbol{Y}}\mathbb{R}^2$. Using this interpretation Ryan *et al.* developed the LBOD and LBSD algorithms which we describe briefly. We will first reiterate some results from [1] and derive some necessary properties.

III. Preliminary Definitions

Definition 1: Let $NN(\boldsymbol{v})$ be the nearest codeword, or set of codewords, in C_R to $\boldsymbol{v}$. This is: if $\underline{\boldsymbol{x}} \in NN(\boldsymbol{v})$ then $||\underline{\boldsymbol{x}} - \boldsymbol{v}|| \leq ||\boldsymbol{d} - \boldsymbol{v}||$ for any $\boldsymbol{d} \in C_R$.

Definition 2: Let $\mathcal{N}$ be the set of codewords nearest to the subspace $\underline{\boldsymbol{Y}}\mathbb{R}^2$. That is $\underline{\boldsymbol{x}} \in \mathcal{N}$ if and only if there exists $\boldsymbol{\lambda} \in \mathbb{R}^2$ such that $\underline{\boldsymbol{x}} \in NN(\underline{\boldsymbol{Y}}\boldsymbol{\lambda})$.

Definition 3: Let $V(\underline{\boldsymbol{x}}) \subset \mathbb{R}^{2T}$ where $\underline{\boldsymbol{x}} \in C_R$ such that $\boldsymbol{v} \in V(\underline{\boldsymbol{x}}) \Rightarrow \underline{\boldsymbol{x}} \in NN(\boldsymbol{v})$. $V(\underline{\boldsymbol{x}})$ is the Voronoi region of the codeword $\underline{\boldsymbol{x}}$.

Definition 4:

$$G(\underline{\boldsymbol{x}}) = V(\underline{\boldsymbol{x}}) \cap \underline{\boldsymbol{Y}}\mathbb{R}^2$$

$G(\underline{\boldsymbol{x}})$ is region of the plane $\underline{\boldsymbol{Y}}\mathbb{R}^2$ that is closest to the codeword $\underline{\boldsymbol{x}}$. $G(\underline{\boldsymbol{x}}) \neq \emptyset$ if and only if $\underline{\boldsymbol{x}} \in \mathcal{N}$.

We define the set $\mathcal{T} = \{1, ..., 2T\}$ and the set $\mathcal{K} = \{-M + 2, -M + 4, ..., M - 2\}$.

Definition 5: For $t \in \mathcal{T}$ and $k \in \mathcal{K}$ we define the line,

$$L(k, t) = \{\underline{\boldsymbol{Y}}\boldsymbol{\lambda} | \boldsymbol{\lambda} \in \mathbb{R}^2, \lambda_1 \underline{Y}_{t,1} + \lambda_2 \underline{Y}_{t,2} = k\}$$

$L(k, t)$ is the boundary in the plane $\underline{\boldsymbol{Y}}\mathbb{R}^2$ where the tth element in the codeword changes from $\underline{x}_t = k - 1$ to $\underline{x}_t = k + 1$. Figure 1 is an example of the lines $L(k, t)$ and the regions $G(\underline{\boldsymbol{x}})$ in the plane $\underline{\boldsymbol{Y}}\mathbb{R}^2$.

Remark 1: The boundaries of the non-empty $G(\underline{\boldsymbol{x}})$ are defined by some set of the lines $L(k, t)$.

IV. Existing Detectors

A. Lattice Based Optimal Detector (LBOD)

Ryan *et al.* [1] show that the optimal estimate lies in the set $\mathcal{N}$. For LBOD the likelihood of all codewords in the set, $\mathcal{N}$ is evaluated. This is a significant reduction when compared to evaluating the likelihood of *all* codewords in C_R and yet the optimal estimate is still guaranteed to be found. In order to evaluate the codewords in $\mathcal{N}$ the position of the vertices of every non-empty region $G(\underline{\boldsymbol{x}})$ is calculated as the position of the intersection of every pair of lines $L(k, t)$ and $L(k', t')$ for $k, k' \in \mathcal{K}$ and $t, t' \in \mathcal{T}$.

Given the position of a vertex a small translation in an arbitrary direction is added and the nearest codeword is calculated. The translation is required so that a unique nearest codeword exists. Ryan *et al.* state that provided the translation

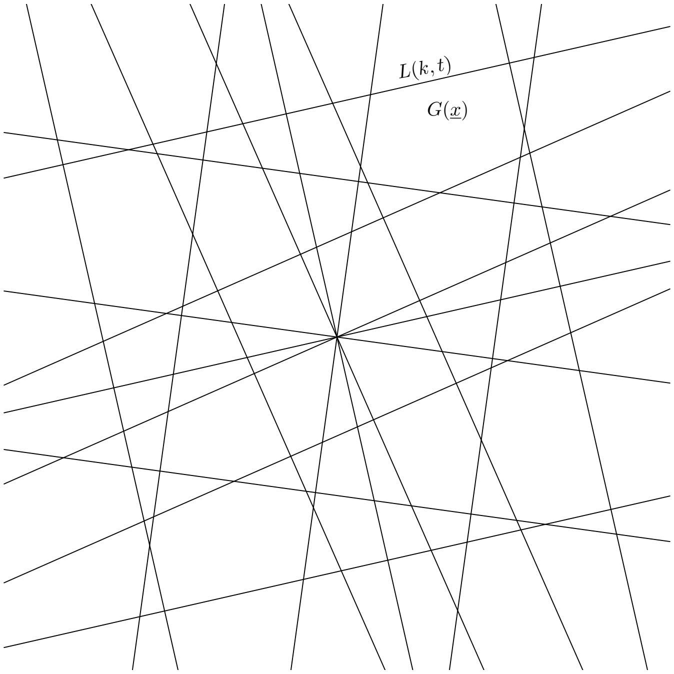

Fig. 1. The lines $L(k,t)$ in the plane $\underline{\boldsymbol{Y}}\mathbb{R}^2$ where $\boldsymbol{y} = [-0.1076 - 0.4728i, -0.7002 - 0.0968i, -1.1228 + 0.4955i]^T$.

is made sufficiently small then the algorithm is guaranteed to find the optimal estimate.

Using the $\pi/2$ phase ambiguity, Ryan *et al.* also note that only 1/4 of the vertices need to be tested. Specifically only codewords in a single quadrant of the plane $\underline{\boldsymbol{Y}}\mathbb{R}^2$ need to be tested.

The algorithm was shown to test $O(M^2T^2)$ vertices. It requires $O(T)$ to test the likelihood of the codewords at a vertex. The overall complexity is $O(M^2T^3)$.

B. Lattice Based Sub-optimal Detector (LBSD)

Rather than evaluating every codeword in $\mathcal{N}$, LBSD considers the codewords that pass close to a finite number of lines lying in the plane $\underline{\boldsymbol{Y}}\mathbb{R}^2$. By using lines the likelihood of each tested codeword can be evaluated in $O(\log T)$ computations, an improvement on the $O(T)$ computations required for LBOD. Ryan *et al.* suggest using lines that radiate outward from the origin and lie in the first quadrant of the plane $\underline{\boldsymbol{Y}}\mathbb{R}^2$.

One difficulty with this approach is that it is not known how many lines are needed so that the optimal codeword will be found. It was conjectured that $O(T)$ lines are needed. For $T = 3$ and 7 and for 16-QAM, Ryan *et al.* indicate that as little as 4 lines gives near optimal results.

It was shown that each line passes through $O(T)$ codewords. Each line therefore requires $O(T \log T)$ calculations. If we assume that $O(T)$ lines are required to give reasonable results then the order of complexity is $O(T^2 \log T)$.

V. Fast Optimal Detector

In LBSD, the position of the line searches performed across the plane were chosen to radiate from the origin of the plane in an attempt to enumerate the optimal codeword. For FOD we instead position the line searches across the plane along the intersection of the plane with the nearest neighbour boundaries of each codeword, this is, we will use the lines $L(k,t)$ for $k \in \mathcal{K}$ and $t \in \mathcal{T}$. This results in a criss-crossing of lines across the plane as seen in Figure 1. We show that this approach is sufficient to guarantee optimality, while maintaining the same complexity as the LBSD algorithm *i.e.* $O(T^2 \log T)$.

A. Theory

Definition 6: We define

$$\vartheta(k,t) = \{\underline{\boldsymbol{x}} | \underline{\boldsymbol{x}} \in NN(\boldsymbol{d}), \boldsymbol{d} \in L(k,t)\}$$

i.e. this is the set of codewords that are nearest neighbours to the line $L(k,t)$.

We now present the following theorem which relates $\vartheta(k,t)$ to $\mathcal{N}$. The relationship will be used in our new algorithm.

Theorem 1:

$$\underline{\boldsymbol{x}} \in \bigcup_{k\in\mathcal{K}} \bigcup_{t\in\mathcal{T}} \vartheta(k,t) \Leftrightarrow \underline{\boldsymbol{x}} \in \mathcal{N}$$

Proof: For the forward implication

$$\underline{\boldsymbol{x}} \in \bigcup_{k\in\mathcal{K}} \bigcup_{t\in\mathcal{T}} \vartheta(k,t) \Rightarrow \underline{\boldsymbol{x}} \in \vartheta(k,t) \Rightarrow \underline{\boldsymbol{x}} \in NN(\boldsymbol{d})$$

for some $\boldsymbol{d} \in L(k,t)$, $k \in \mathcal{K}$ and $t \in \mathcal{T}$. This implies that there exists $\boldsymbol{\lambda} \in \mathbb{R}^2$ such that $\underline{\boldsymbol{x}} \in NN(\underline{\boldsymbol{Y}}\boldsymbol{\lambda})$. Using Definition 2 it follows that $\underline{\boldsymbol{x}} \in \mathcal{N}$.

For the reverse implication, it is a consequence of Definition 4 that

$$\underline{\boldsymbol{x}} \in \mathcal{N}(C_R, \underline{\boldsymbol{Y}}) \Rightarrow G(\underline{\boldsymbol{x}}) \neq \emptyset.$$

From Remark 1 we know that the boundary of $G(\underline{\boldsymbol{x}})$ is defined by the lines $L(k,t)$. Selecting $\boldsymbol{d} \in L(k,t)$ such that $\boldsymbol{d}$ lies on the boundary of $G(\underline{\boldsymbol{x}})$ yields

$$\underline{\boldsymbol{x}} \in NN(\boldsymbol{d}) \Rightarrow \underline{\boldsymbol{x}} \in \bigcup_{k\in\mathcal{K}} \bigcup_{t\in\mathcal{T}} \vartheta(k,t).$$

■

Theorem 1 implies that in order to evaluate the likelihood of all codewords in $\mathcal{N}$, we may instead evaluate the likelihood of the codewords in

$$\bigcup_{k\in\mathcal{K}} \bigcup_{t\in\mathcal{T}} \vartheta(k,t) \tag{5}$$

and the optimal estimate is still guaranteed to be found.

B. Computational Complexity

The number of lines, $L(k,t)$, required is $|\mathcal{K}||\mathcal{T}| = 2T(M-1)$. Each line contains $(M-1)(T-1)$ intersections and so $|\vartheta(k,t)| = M(T-1)$. Therefore for each line $M(T-1)$ codewords need to be evaluated. Each evaluation requires $O(\log T)$ arithmetic computations. The overall complexity is then $2MT(T-1)(M-1)O(\log T) = O(M^2T^2 \log T)$.

By using the $\pi/2$ phase ambiguity the number of computations can be quartered. For brevity we shall not detail how this is achieved here.

C. Pseudocode

The algorithm is implemented similarly to the LBSD algorithm, however, the lines that are searched are now the $L(k,t)$ rather than lines that radiate from the origin.

Pseudocode for the new algorithm is given in Figure 2. We use $\hat{\underline{\boldsymbol{x}}}$ and $\hat{\boldsymbol{x}}$ interchangeably. They are the same variable, but $\hat{\boldsymbol{x}}$ is to be considered as a complex vector and $\hat{\underline{\boldsymbol{x}}}$ the corresponding real vector according to (2).

The function $\texttt{modifiedLBSD}(\underline{\boldsymbol{Y}}, k, t)$ is a modified version of the LBSD line searching algorithm [1]. The function $[L, \boldsymbol{\lambda}, \gamma] = \texttt{modifiedLBSD}(\underline{\boldsymbol{Y}}, k, t)$ evaluates the likelihood of all codewords in $\vartheta(k,t)$ and returns three values L, $\boldsymbol{\lambda}$ and γ. The returned value L is the likelihood of the *best* codeword in $\vartheta(k,t)$. The returned value $\boldsymbol{\lambda}$ is the position on the plane $\underline{\boldsymbol{Y}}\mathbb{R}^2$ where the best codeword in $\vartheta(k,t)$ is a nearest neighbour. Note that $\underline{\boldsymbol{Y}}\boldsymbol{\lambda} \in L(k,t)$ and so $NN(\underline{\boldsymbol{Y}}\boldsymbol{\lambda})$ will contain 2 elements corresponding to codewords from both sides of $L(k,t)$, ie. for $x_t = k+1$ or $k-1$. The returned value γ is used to store either $k+1$ or $k-1$, depending on which side of $L(k,t)$ the best codeword is.

There are two trivial but necessary modifications to the LBSD line searching algorithm for it to perform the operations of $\texttt{modifiedLBSD}(\underline{\boldsymbol{Y}}, k, t)$. Firstly, the original LBSD algorithm only uses lines that pass through the origin. However, the $L(k,t)$ do not necessarily pass though the origin. $\texttt{modifiedLBSD}(\underline{\boldsymbol{Y}}, k, t)$ is generalized to allow lines that do not pass through the origin. Secondly $\texttt{modifiedLBSD}(\underline{\boldsymbol{Y}}, k, t)$ is required to check the codeword on both sides of the line $L(k,t)$ whereas the original LBSD algorithm is not.

Full pseudocode for the $\texttt{modifiedLBSD}(\underline{\boldsymbol{Y}}, k, t)$ function is provided in the appendix.

```
$L^{\text{opt}} = -\infty$
for $t = 1$ to $2T$ step $1$ do
    for $k = 0$ to $M - 2$ step $1$ do
        $[L, \boldsymbol{\lambda}, \gamma] =$ modifiedLBSD($\underline{\boldsymbol{Y}}, k, t$)
        if $L > L^{opt}$ then
            $L^{\text{opt}} = L$
            $\boldsymbol{\lambda}^{\text{opt}} = \boldsymbol{\lambda}$
            $t^{\text{opt}} = t$
            $\gamma^{\text{opt}} = \gamma$
$\hat{\underline{\boldsymbol{x}}}^{\text{opt}} = NN(\underline{\boldsymbol{Y}}\boldsymbol{\lambda}^{\text{opt}})$
$\hat{\underline{x}}^{\text{opt}}_{t^{\text{opt}}} = \gamma^{\text{opt}}$
return $\hat{\boldsymbol{x}}^{\text{opt}}$
```

Fig. 2. GLRT-optimal square QAM Algorithm

VI. Simulation Results

Figure 3 shows the codeword error rate as a function of signal to noise ratio (SNR) for FOD, LBOD and LBSD with varying values of T and M over a block Rayleigh fading channel where h is i.i.d. circularly symmetric complex Gaussian with unit variance. We have assumed that the phase ambiguities have been removed by some technique [2], [5]. It can be seen that all estimators approach the ambiguity lower bound [8].

When $M = 4$ it appears that 4 lines are adequate for LBSD to achieve near-optimal results. When $M = 8$ it appears that 7 lines are required. We used this number of lines to generate the results in Figure 3. The relationship between T, M and the required number of lines for the suboptimal estimator is not well understood.

Figure 4 shows the effect of the number of search lines on the codeword error rate for LBSD when $T = 5$ and $M = 8$. Figure 4 shows that care must be taken to choose a sufficient number of lines or the results can be poor. The advantage of FOD over LBSD is that there is no need to choose the number of search lines. FOD is guaranteed to find the optimal estimate. Still, LBSD is both computationally very efficient and indistinguishable from optimal when a sufficient number of lines is used.

Table VI shows the number of seconds used to perform 100,000 trials for varying values of M and T. The number of search lines used for LBSD is 4 when $M = 4$ and 7 when $M = 8$. The computer used is an Intel Core2 running at 2.13Ghz.

It is evident that FOD is slower than LBSD and significantly faster than the LBOD.

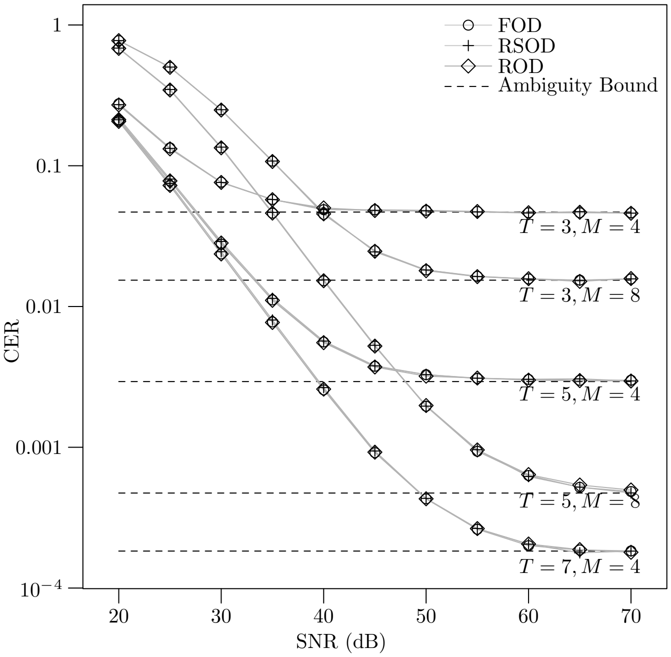

Fig. 3. Codeword error rate (CER) versus signal to noise ratio (SNR) for varying T and M.

VII. Conclusion

The paper describes a GLRT-optimal algorithm for noncoherent reception of square QAM symbols block-fading AWGN channels. The algorithm requires $O(T^2 \log T)$ arithmetic computations. This is an improvement on a previous optimal algorithm that requires $O(T^3)$ arithmetic computations [1].

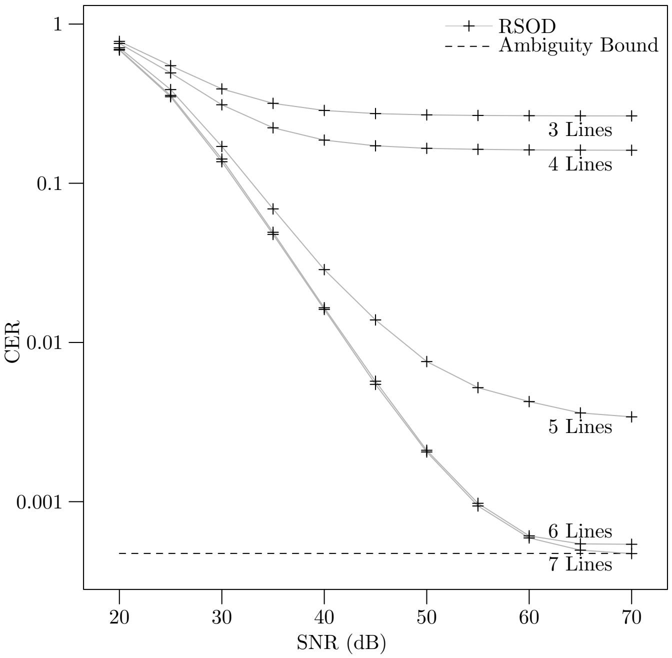

Fig. 4. Codeword error rate (CER) versus signal to noise ratio (SNR) for $T = 5$ and $M = 8$ and for varying number of search lines for LBSD

TABLE I
COMPUTATION TIME IN SECONDS FOR 100,000 TRIALS

	$T = 3$ $M = 4$	$T = 5$ $M = 4$	$T = 7$ $M = 4$	$T = 3$ $M = 8$	$T = 5$ $M = 8$
LBSD	1.08	1.75	2.45	3.38	6.00
FOD	1.45	3.94	7.78	7.69	23.88
LBOD	15.25	70.78	192.92	48.33	227.27

The paper indicates that optimal noncoherent detection for square QAM is possible in practicable time.

REFERENCES

[1] D. J. Ryan, I. B. Collings, and I. V. L. Clarkson, "GLRT-optimal noncoherent lattice decoding," *IEEE Trans. Sig. Process.*, vol. 55, pp. 3773–3786, 2007.

[2] R. Chen, R. Koetter, D. Agrawal, and U. Madhow, "Joint demodulation and decoding for the noncoherent block fading channel: a practical framework for approaching Shannon capacity," *IEEE Trans. Comm.*, vol. 51, no. 10, pp. 1676–1689, October 2003.

[3] C. N. Georghiades, "Blind carrier phase acquisition for QAM constellations," *IEEE Trans. Comm.*, vol. 45, no. 11, pp. 1477–1486, 1997.

[4] D. Warrier and U. Madhow, "Spectrally efficient noncoherent communication," *IEEE Trans. Inform. Th.*, vol. 48, no. 3, pp. 651–668, 2002.

[5] W. J. Weber, III, "Differential encoding for multiple amplitude and phase shift keying systems," *IEEE Trans. Comm.*, vol. 26, pp. 385–391, Mar. 1978.

[6] C. Ling, W. H. Mow, K. H. Li, and A. C. Kot, "Multiple-antenna differential lattice decoding," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 9, pp. 1821–1829, 2005.

[7] D. J. Ryan, I. V. L. Clarkson, I. B. Collings, D. Guo, and M. L. Honig, "QAM Codebooks for Low-Complexity Limited Feedback MIMO Beamforming," *IEEE Int. Conf. Comm.*, pp. 4162–4167, 24-28 June 2007.

[8] D. J. Ryan, I. B. Collings, and I. V. L. Clarkson, "Maximum-likelihood noncoherent lattice decoding of qam," *Proc. Internat. Conf. Acoust. Speech Signal Process.*, vol. 4, pp. 189–192, 2006.

APPENDIX

Full pseudocode for the `modifiedLBSD`($\underline{\boldsymbol{Y}}, k, t$) function is provided in Figure 5. Figure 6 contains the function `updateL`(b) that is used by `modifiedLBSD`($\underline{\boldsymbol{Y}}, k, t$).

```
Input: $\underline{\boldsymbol{Y}}, k, t$
$L = -\infty$ //initialise likelihood
$\boldsymbol{d} = \underline{\boldsymbol{Y}}[1, \frac{\underline{Y}_{t,1}}{\underline{Y}_{t,2}}]^T$ //$\boldsymbol{d}$ = gradient of $L(k,t)$
$\boldsymbol{c} = \underline{\boldsymbol{Y}}[0, \frac{k}{\underline{Y}_{t,2}}]^T$ //$\boldsymbol{c}$ = translation of $L(k,t)$
$\mathcal{V} = \emptyset$ //calculate crossing of line $L(k,t)$
for $j = 0$ to $2T - 1$ do
    if $j \neq t$ then
        for $m = -M + 2$ to $M - 2$ step $2$ do
            $b = (m - c_j)/d_j$
            $\mathcal{V} = \{\mathcal{V}, \{b, j\}\}$
$\mathcal{V}$ = sort($\mathcal{V}$) //sort crossings in ascending order of $b$
$\boldsymbol{s} = \mathtt{sign}(\boldsymbol{d})$
$\underline{\hat{\boldsymbol{x}}} = -\boldsymbol{s}(M - 1)$ //initial codeword
$\underline{\hat{x}}_t = k + 1$
$\alpha = \hat{\boldsymbol{x}}^\dagger \boldsymbol{y}$
$\beta = ||\hat{\boldsymbol{x}}||^2$
updateL($\mathcal{V}_{1,1} - 1.0$)
//for all crossings of $L(k,t)$
for $m = 1$ to $|\mathcal{V}| - 1$ do
    $t' = \mathcal{V}_{m,2}$
    $\beta = \beta + 4s_{t'}\underline{x}_{t'} + 4$
    $\alpha = \alpha + 2s_{t'}\underline{Y}_{t',1} - 2is_{t'}\underline{Y}_{t',2}$
    $\underline{x}_{t'} = \underline{x}_{t'} + 2s_{t'}$
    if $k \neq 0$ *or* $t' \neq 0$ *or* $\mathcal{V}_{m+1,2} \neq 0$ then
        updateL(($\mathcal{V}_{m,1} + \mathcal{V}_{m+1,1}$)/2)
$t' = \mathcal{V}_{|\mathcal{V}|,2}$
$\beta = \beta + 4s_{t'}\underline{x}_{t'} + 4$
$\alpha = \alpha + 2s_{t'}\underline{Y}_{t',1} - 2is_{t'}\underline{Y}_{t',2}$
$\underline{x}_{t'} = \underline{x}_{t'} + 2s_{t'}$
updateL($\mathcal{V}_{1,1} + 1.0$)
return $[L, \boldsymbol{\lambda}, \gamma]$
```

Fig. 5. $[L, \boldsymbol{\lambda}, \gamma]$ = `modifiedLBSD`($\underline{\boldsymbol{Y}}, k, t$) function

```
Input: $b$
if $L < |\alpha|^2/\beta$ then
    $L = |\alpha|^2/\beta$
    $\boldsymbol{\lambda} = [b, \frac{b\underline{Y}_{t,1}+k}{\underline{Y}_{t,2}}]^T$
    $\gamma = k + 1$
if $L < |\alpha - 2\underline{Y}_{t,1} + 2i\underline{Y}_{t,2}|^2/(\beta - 4x_t + 4)$ then
    $L = |\alpha - 2\underline{Y}_{t,1} + 2i\underline{Y}_{t,2}|^2/(\beta - 4x_t + 4)$
    $\boldsymbol{\lambda} = [b, \frac{b\underline{Y}_{t,1}+k}{\underline{Y}_{t,2}}]^T$
    $\gamma = k - 1$
```

Fig. 6. `updateL`(b) function