Insights into Maximum Likelihood Detection for One-bit Massive MIMO Communications

Aditya Sant, *Member, IEEE,* and Bhaskar D. Rao, *Life Fellow, IEEE*

***Abstract*—One-bit massive MIMO has gained much attention in the areas of wireless communication and sensing. Among the various receiver designs, the maximum-likelihood-based receivers achieve state-of-the-art performance. Through this work we provide both analytical insight into the likelihood formulation, and develop a one-bit MIMO receiver, motivated specifically from this analysis. In particular, *(i)* Properties of the original Gaussian CDF based likelihood function are analyzed, culminating in an improved gradient descent (GD) algorithm for one-bit MIMO. *(ii)* This improved GD update rule is further enhanced through an accelerated GD method, improving convergence performance. *(iii)* The likelihood analysis is extended to an effective surrogate function for the Gaussian CDF, i.e., the logistic regression (LR). The presented analytical framework for the CDF also serves as a robust mathematical model to explain the enhanced performance of the LR, when utilized as a surrogate likelihood. *(iv)* Detection from a finite M-QAM constellation is incorporated by introducing a Gaussian denoiser to project the detected symbols onto the M-QAM subspace. This is implemented as a novel, unfolded, DNN architecture for one-bit detection. Through our experimental validation we demonstrate results on par with the current state-of-the-art methods for one-bit MIMO detection.**

***Index Terms*—Massive MIMO, one-bit ADCs, convex optimization, accelerated gradient descent, unfolded DNNs.**

I. Introduction

THE advent of massive MIMO communications has brought in a new era of high speed communication systems and interconnected devices [1], [2]. However, one of the key challenges facing massive MIMO deployment is the ensuing system cost and complexity. In this context, the use of high-resolution and high-speed analog-to-digital converters (ADCs) significantly contributes to the overall cost and power consumption within the MIMO communication system [3], [4]. Addressing these challenges, accompanied by several advances in algorithm design and machine learning techniques, research into robust communication system design is being developed on the backbone of few-bit or low resolution ADCs [3], [5]–[9]. A specific type of low-resolution ADC, the one-bit ADC, has garnered significant attention in communication system design and sensing due to its simple design and ease of implementation.

Advances in DNN technologies have enabled robust detector designs for these few-bit MIMO receivers. The application of DNNs to wireless communication systems has significantly enhanced receiver performance and robustness. The general parametric structure of DNNs, coupled with their advantage as universal functional approximators [10], [11], makes these an integral part of the future of robust wireless communication, exploited for a variety of applications from beamformer design [12]–[14], channel estimation [15]–[17] as well as end-to-end detection [18]–[22]. In this work, we begin by analyzing the detection process for one-bit MIMO receivers. Following this, a robust detector utilizing a DNN-aided unfolded network is developed for multi-user one-bit massive MIMO systems.

A. Prior work

One-bit MIMO was first used for sensing and channel estimation algorithms [23]–[25]. Going beyond this, the main focus of research into one-bit MIMO has been on receiver design. One-bit MIMO data detection gained a valuable advance with the application of Bussgang's theorem to linearize the input-output relation [26]. Through means of this relation, a class of linear receivers was developed for detection from one-bit data [27]–[29]. Several works utilized this linearization to characterize the one-bit system and evaluate the overall system performance and capacity [30]–[32]. Additional robust model-based detectors improving on the Bussgang linear detectors have also been proposed in some works [33], [34].

However, presently, the state-of-the-art class of receivers utilizes the nonlinear optimization of the likelihood function. The one-bit maximum likelihood (ML) optimization was derived using the Gaussian cumulative distribution function (CDF) [35]. Utilizing this formulation, the work in [36] introduced a near maximum likelihood (n-ML) detector based on a two step iterative algorithm - gradient descent (GD) followed by projection onto the unit sphere. Other works applying the Gaussian CDF likelihood formulation have also been used extending this idea [37], [38]. However, one of the limitations of applying the GD iteration on the Gaussian CDF is its numerical instability at high signal-to-noise ratio (SNR) values [39]. One of the approaches to address this was through a surrogate function for the Gaussian CDF, i.e., the logistic regression (LR). The authors in [40] designed the detector, the OBMNet, as an unfolded DNN, implementing the GD algorithm for this approximate likelihood. Both the n-ML algorithm as well as the OBMNet were limited in performance due to the sub-optimal projection step onto the M-QAM constellation. The work in [5] improved on the OBMNet

This work is financially supported in part by ONR Grant No. N00014-18-1-2038, NSF grant CCF-2124929, NSF Grant CCF-2225617, and the UCSD Center for Wireless Communications.

Aditya Sant was with the Department of Electrical and Computer Engineering, University of California San Diego, La Jolla, CA, USA. He is currently working with InterDigital Communications, Inc., Los Altos, CA, USA (email: aditya.sant@interdigital.com).

Bhaskar D. Rao is with the Department of Electrical and Computer Engineering, University of California at San Diego, La Jolla, CA, USA (e-mail: brao@ucsd.edu).

by introducing a learnable M-QAM projection over the GD iterations. The resulting unfolded DNN, the FBM-DetNet, is the current state-of-the-art detector for one-bit MIMO systems.

Extending the model-based methods to learning-based methods, DNNs have also been used to design robust detectors for one-bit MIMO receivers. The OBMNet [40] and FBM-DetNet [5] were implemented as unfolded DNNs, learning the parameters for the GD iterations and the M-QAM projection, respectively. The work in [41] utilized model-based unrolling to learn the GD algorithm with a generalized Newton update. Alternate DNN architectures for one-bit detection, not relying on the likelihood framework, have also been developed [42]–[45]. The general parametric structure of DNNs can also enhance the GD update step by enforcing a general regularization at the end of each GD iteration. The framework in [46] utilizes two unique networks - an unfolded DNN, the ROBNet, as well as a recurrent network, the OBiRIM, to implement a regularized GD algorithm. The mmWave extension of the regularized GD, i.e., the mmW-ROBNet [47], demonstrates the utility of the regularized GD framework for mmWave channels. Here, the regularized framework, along with a novel hierarchical training strategy is able to generate equitable user performance for the mmWave one-bit MIMO receiver.

Although different strategies for detection of one-bit received signals have been proposed, no work, to the best of the authors' knowledge, comprehensively looks at the properties and convergence for the recovery algorithms. Bridging this gap, this work aims to generate useful insights into the ML framework for the one-bit MIMO receivers.

B. Contributions of this work

Through this work, we endeavor to bridge the gap between theory and algorithm design for the one-bit MIMO receiver. In particular, the following contributions are enumerated.

1) *Characterizing ML optimization:* We characterize the properties of the CDF-based likelihood, namely, the convexity, smoothness, and the nature of the solution space. Different from prior works, this analysis enables structured algorithm design as well as convergence analysis.
2) *Stabilizing CDF-based GD update:* Implementing the GD update for the CDF-based likelihood is shown to run into computational instabilities. Utilizing the properties of the CDF, a robust approximation of the gradient is implemented, preserving the first order properties of the CDF (necessary for GD).
3) *Introduce accelerated GD update:* This stabilized GD update is utilized in the design of an accelerated GD algorithm for faster convergence. To the best of the authors' knowledge, this is the first work to utilize AGD in signal recovery for one-bit MIMO receivers. The convergence of the algorithms is analyzed using the properties of the likelihood function.
4) *Analysis of a robust CDF surrogate, LR:* Prior works have demonstrated the utility of the logistic regression (LR) as an effective surrogate to the CDF for the one-bit likelihood [5], [39], [40]. The insights from the CDF-based likelihood are extended to explain the improved performance of the LR-based likelihood.
5) *DNN-aided Gaussian denoising:* In order to address the constrained optimization over the M-QAM symbols, we extend and generalize the quantization-based M-QAM projection from [5]. To this end, we expound the role of the M-QAM projection step and develop a general *learnable* two-tier projection framework for robust M-QAM symbol recovery. This framework is implemented as an unfolded DNN referred to as the A-PrOBNet.

Organization: This manuscript is organized as follows - Sec. II introduces the system model and formulation of the one-bit likelihood optimization. Sec. III analyzes the different properties of the CDF-based likelihood. This section also introduces the improved GD algorithm and the AGD algorithm, as well as the related convergence analysis for these algorithms. Sec. IV analyzes the surrogates for the CDF-based likelihood, in particular, the LR-based likelihood. Sec. V introduces the general Gaussian denoising for projection onto the M-QAM symbol space. Sec. VI provides the experimental validation and Sec. VII provides concluding remarks and future directions.

Notation: The abbreviation ML is used for maximum likelihood, as opposed to machine learning. The latter has not been abbreviated wherever utilized. We use lower-case boldface letters $\mathbf{a}$ and upper case boldface letters $\mathbf{A}$ to denote complex valued vectors and matrices respectively. The i^{th} element of the vector $\mathbf{a}$ is denoted by a_i. The notation $\mathfrak{Re}(\cdot)$ and $\mathfrak{Im}(\cdot)$ denote the real and imaginary parts, respectively. The operation $(\cdot)^{\text{T}}$ denotes the transpose of the array or matrix. Unless otherwise specified, all scalar functions like $\tanh(\cdot)$ or $\text{sign}(\cdot)$, when applied to arrays or matrices, imply element-wise operation. The diagonalization operator, denoted by $\text{diag}(\cdot)$, when applied to an array $\mathbf{a}$, creates a diagonal matrix with the entries given by the elements of $\mathbf{a}$. The notation $\mathbf{x}^{(t)}$ is used to denote the value of the variable $\mathbf{x}$ at iteration t of the algorithm. For the DNN training, the size of the training set is given by N_{train} and the notation $\hat{\mathbf{x}}_{n,\text{train}}$ denotes the n^{th} sample from this set. Unless otherwise specified, the norm $||\cdot||$ represents the ℓ_2-norm for a vector or matrix.

II. System Model and general one-bit likelihood

Through this section the multi-user uplink MIMO model is introduced, with one-bit ADCs at the base station (BS) receiver. This is followed by the formulation of the ML optimization that forms the basis of the detection algorithm.

A. One-bit MIMO system model

The Rayleigh fading channel with block flat-fading, as used in most past works, e.g. [48], [49] is utilized here. The K single antenna users transmit to a multi-antenna base-station (BS) with N receive antennas. The MIMO channel $\bar{\mathbf{H}} \in \mathbb{C}^{N\times K}$ consists of i.i.d entries drawn from $\mathcal{CN}(0,1)$. This work assumes perfect unquantized channel state information (CSI) at the BS.

As a part of the multi-user uplink, the k^{th} user transmits the signal $\bar{x}_k$ drawn from the M-QAM constellation. The multi-user transmitted signal is $\bar{\mathbf{x}} = \left[\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_K\right]^{\text{T}}$. The unquantized received signal at the BS is given by

$$\bar{\mathbf{r}} = \bar{\mathbf{H}}\bar{\mathbf{x}} + \bar{\mathbf{n}}, \tag{1}$$

where $\bar{\mathbf{n}}$ is the AWCGN[1] with noise variance depending on the system signal-to-noise ratio (SNR) $\rho = \frac{\mathbb{E}(||\bar{\mathbf{H}}\bar{\mathbf{x}}||^2)}{\mathbb{E}(||\bar{\mathbf{n}}||^2)}$. The transformed signal due to the one-bit quantization is given by

$$\bar{\mathbf{y}} = \operatorname{sign}\big(\mathfrak{Re}(\bar{\mathbf{r}})\big) + j\operatorname{sign}\big(\mathfrak{Im}(\bar{\mathbf{r}})\big). \tag{2}$$

In order to express the algorithm design as a function of real-valued inputs, we convert the received signal and the observed channel matrix into real-valued forms as

$$\begin{aligned}\mathbf{H} &= \begin{bmatrix}\mathfrak{Re}(\bar{\mathbf{H}}) & -\mathfrak{Im}(\bar{\mathbf{H}})\\ \mathfrak{Im}(\bar{\mathbf{H}}) & \mathfrak{Re}(\bar{\mathbf{H}})\end{bmatrix},\ \mathbf{x} = \begin{bmatrix}\mathfrak{Re}(\bar{\mathbf{x}})\\ \mathfrak{Im}(\bar{\mathbf{x}})\end{bmatrix},\\ \mathbf{r} &= \begin{bmatrix}\mathfrak{Re}(\bar{\mathbf{r}})\\ \mathfrak{Im}(\bar{\mathbf{r}})\end{bmatrix},\ \mathbf{y} = \begin{bmatrix}\mathfrak{Re}(\bar{\mathbf{y}})\\ \mathfrak{Im}(\bar{\mathbf{y}})\end{bmatrix},\ \mathbf{n} = \begin{bmatrix}\mathfrak{Re}(\bar{\mathbf{n}})\\ \mathfrak{Im}(\bar{\mathbf{n}})\end{bmatrix}.\end{aligned} \tag{3}$$

Thus, the modified received one-bit signal at the BS is

$$\mathbf{y} = \operatorname{sign}(\mathbf{H}\mathbf{x} + \mathbf{n}). \tag{4}$$

The detection algorithm recovers the M-QAM transmitted symbols $\mathbf{x}$ from the one-bit received data $\mathbf{y}$.

B. Signal detection - Maximum likelihood framework

The signal detection for one-bit MIMO is formulated as the maximum likelihood (ML) optimization, derived in [35] as

$$\hat{\mathbf{x}}_{\mathrm{ML}} = \underset{\mathbf{x}\in\mathcal{M}^{2K}}{\operatorname{argmax}} \prod_{i=1}^{2N} \Phi\big(\sqrt{2\rho}\, y_i \mathbf{h}_i^{\mathrm{T}}\mathbf{x}\big). \tag{5}$$

Expressing the maximization (5) as the minimization of the negative log-likelihood gives the optimization of the form

$$\hat{\mathbf{x}}_{\mathrm{ML}} = \underset{\mathbf{x}\in\mathcal{M}^{2K}}{\operatorname{argmin}} \sum_{i=1}^{2N} -\log\Phi\big(\sqrt{2\rho}\, y_i \mathbf{h}_i^{\mathrm{T}}\mathbf{x}\big), \tag{6}$$

where $\Phi(\cdot)$ is the Gaussian cumulative distribution function (CDF) for $\mathcal{N}(0,1)$ and $\mathcal{M}^{2K}$ represents the set of the $2K$-dimensional vectors, consisting of the real-valued representation (see eq. (3)) of the K-dimensional vectors of M-QAM symbols. In addition, the vector $\mathbf{h}_i$ denotes the i^{th} row of $\mathbf{H}$.

Remark 1. Since the factor $\sqrt{2\rho}$ is a positive scalar and does not affect the convergence of the optimization over the constrained set $\mathcal{M}^{2K}$, we can eliminate this factor for ease of representation. Thus, for all subsequent expressions and analysis, the likelihood is expressed as a general function by the form $\mathcal{L} = \sum_i f(y_i \mathbf{h}_i^{\mathrm{T}}\mathbf{x})$.

In order to delve deeper into the analysis of the likelihood function, and subsequent algorithm development, we consider two key features with respect to the optimization (6).

1) Generalization of likelihood: In order to understand the broader class of likelihood functions, including all surrogate measures, a general likelihood formulation is presented as

$$\mathcal{L}(\mathbf{x}) = \sum_{i=1}^{2N} \zeta(y_i \mathbf{h}^{\mathrm{T}}\mathbf{x}). \tag{7}$$

The scalar function $\zeta(\cdot)$ can take different values, depending on the exact or surrogate value of the likelihood. Based on this, two separate lines of analysis are presented.

- By substituting the CDF, we attain the original likelihood expression (6). We provide detailed analysis into the CDF-based likelihood expression in Sec. III.
- We can also substitute appropriate surrogates for the CDF-based likelihood to overcome the limitations of the former. This is elaborated in more detail in Sec. IV.

The general gradient $\nabla_{\mathbf{x}}$ and Hessian $\mathcal{H}_{\mathbf{x}}$ expressions will be utilized in the analysis later. For the general likelihood $\zeta(\cdot)$, these expressions are given as

$$\nabla_{\mathbf{x}} = \mathbf{G}^{\mathrm{T}}\,\zeta'(\mathbf{G}\mathbf{x}) \tag{8a}$$

$$\mathcal{H}_{\mathbf{x}} = \mathbf{H}^{\mathrm{T}}\operatorname{diag}(\zeta''(\mathbf{G}\mathbf{x}))\,\mathbf{H}, \tag{8b}$$

where $\mathbf{G} = \operatorname{diag}(\mathbf{y})\mathbf{H}$ and the $\operatorname{diag}(\cdot)$ operator notation for both matrices and arrays is explained in Sec. I (see Notation).

2) Constrained vs unconstrained optimization: Since the transmitted symbols are drawn from an M-QAM constellation, a constrained optimization is performed over the set of M-QAM symbols. However, for understanding the properties of the likelihood framework and development of robust recovery algorithms, unconstrained optimization over the entire set $\mathbb{R}^{2K}$ is initially considered. Specifically, we analyze

$$\hat{\mathbf{x}}_{\mathrm{ML}} = \underset{\mathbf{x}\in\mathbb{R}^{2K}}{\operatorname{argmin}} \sum_{i=1}^{2N} -\log\Phi\big(y_i \mathbf{h}_i^{\mathrm{T}}\mathbf{x}\big). \tag{9}$$

The CDF-based likelihood and the different CDF surrogates will first be analyzed via the unconstrained optimization framework (9) in Sec. III-IV. Constrained optimization over $\mathcal{M}^{2K}$ is then detailed in Sec. V.

III. Insights into the CDF-based one-bit likelihood

This section begins with the analysis of the CDF-based likelihood. This is followed by the design of a robust approximate GD algorithm and accelerated GD algorithm, along with the convergence analysis for both.

Substituting the CDF-based likelihood for the general expressions (7)-(8) gives

$$\zeta(z) = -\log\Phi(z) \tag{10a}$$

$$\zeta'(z) = -\frac{\phi(z)}{\Phi(z)} \tag{10b}$$

$$\zeta''(z) = \frac{\phi(z)}{\Phi(z)}\Big(z + \frac{\phi(z)}{\Phi(z)}\Big), \tag{10c}$$

where $\phi(\cdot)$ is the probability density function (PDF) of the standard normal distribution $\mathcal{N}(0,1)$. Utilizing this in (7) and evaluating the gradient gives the GD update, as derived in [36],

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \alpha^{(t)}\,\mathbf{G}^{\mathrm{T}}\frac{\phi(\mathbf{G}\mathbf{x})}{\Phi(\mathbf{G}\mathbf{x})}, \tag{11}$$

where $\alpha^{(t)}$ is the step size at the t^{th} iteration.

One of the limitations of applying unconstrained GD to the CDF-based likelihood function is the evaluation of the gradient (10b) (explained in Sec. III-B). We construct a more robust GD algorithm to overcome these limitations.

A. Characterizing the CDF-based likelihood

The various properties of the CDF-based one-bit likelihood function, useful for deriving the different GD-based algorithms

[1] additive white complex Gaussian noise

This article has been accepted for publication in IEEE Transactions on Wireless Communications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TWC.2024.3439648

and analyzing the convergence properties, are enumerated. Before enumerating these properties, the following inequalities for the Gaussian CDF are provided, which will be frequently utilized in the subsequent analysis.

Lemma 1. For the scalar argument $z < 0$, the CDF-gradient $\zeta'(z)$, given by (10b), can be bounded by utilizing

$$-z < \frac{\phi(z)}{\Phi(z)} < -z - \frac{1}{z}. \tag{12}$$

Proof. If $z > 0$, then the following holds for the CDF,

$$\frac{1}{\sqrt{2\pi}}\exp\Big(-\frac{z^2}{2}\Big)\frac{z}{z^2+1} < 1 - \Phi(z) < \frac{1}{\sqrt{2\pi}}\exp\Big(-\frac{z^2}{2}\Big)\frac{1}{z}. \tag{13}$$

Rearranging the terms above gives the inequalities in (12). □

We now enumerate the various characteristic properties of the cdf-based likelihood function.

1) Convexity: The Hessian for the CDF-based likelihood (10c) is substituted in (8b). This can be separately analyzed for both positive and negative arguments. For $z > 0$ the expression (8b) is always positive. For $z < 0$, Lemma 1 is utilized to show

$$0 < \zeta''(z) < 1. \tag{14}$$

Since each element of the matrix $\operatorname{diag}(\zeta''(\mathbf{G}\mathbf{x}))$ in (8b) is always positive, for the CDF-based likelihood, the Hessian is positive semi-definite (PSD). Therefore, the CDF-based likelihood is a convex function of $\mathbf{x}$. Convexity of the CDF-based likelihood is crucial for providing insights into the solution space of (9), as explained next.

2) Solution space: Having showed the convexity of the likelihood, the minimizer is now analyzed. Just as a set of data points in an N-dimensional space can be linearly separable or non-separable for binary classification, the one-bit MIMO detection problem can also be analyzed as linearly separable or non-separable, depending on the SNR. The set of linearly separating hyperplanes for the data points $\{\mathbf{h}_i, y_i\}$, i.e., $\mathcal{X}_1 = \{\mathbf{x}\,|y_i\mathbf{h}_i^{\mathrm{T}}\mathbf{x} > 0, \forall i = 1, 2, \ldots, 2N\}$ is used to analyze the solution space. The separable and non-separable cases are further elaborated below.

- Case 1: Separable solution - There exists at least one finite $\mathbf{x}$, for which $y_i\mathbf{h}_i^{\mathrm{T}}\mathbf{x} > 0$, $\forall i = 1, 2, \ldots, 2N$, i.e., the set $\mathcal{X}_1$ is not empty. This corresponds to operating in a high SNR regime; the power of the AWGN added for the received signal in (1) is low enough such that there are no sign flips compared to the noiseless data, i.e.,

$$\operatorname{sign}(\mathbf{H}\mathbf{x} + \mathbf{n}) = \operatorname{sign}(\mathbf{H}\mathbf{x}). \tag{15}$$

- Case 2: Non-separable solution - There exists no $\mathbf{x}$, such that $y_i\mathbf{h}_i^{\mathrm{T}}\mathbf{x} > 0$, $\forall i = 1, 2, \ldots, 2N$, i.e., $\mathcal{X}_1$ is a null set. This corresponds to low SNR operation, where the noise added has sufficiently high power, such that there are sign flips compared to the noiseless data, i.e.,

$$\operatorname{sign}(\mathbf{H}\mathbf{x} + \mathbf{n}) \neq \operatorname{sign}(\mathbf{H}\mathbf{x}). \tag{16}$$

With regards to analyzing the optimal value for $\mathbf{x}$ for both the cases above, the following analysis is presented.

- Case 1: Separable solution - Consider a value $\mathbf{x}^*$, such that $y_i\mathbf{h}_i\mathbf{x}^* > 0\,\forall\, i$. For a scaling constant $1 < \alpha < \infty$, the CDF-based likelihood, given by (9), decreases as

$$0 < \mathcal{L}(\alpha\mathbf{x}^*) < \mathcal{L}(\mathbf{x}^*). \tag{17}$$

  Further, as $\alpha \to \infty$, $\mathcal{L}(\alpha\mathbf{x}^*) \to 0$. Thus, there does not exist a finite $\mathbf{x}^*$, with $||\mathbf{x}^*|| < \infty$, such that $\mathcal{L}(\mathbf{x}^*) = 0$. Therefore, the minimum value of the likelihood cannot be attained by any finite $\mathbf{x}$. The high-SNR saturation of the performance the GD algorithms is analyzed by operating in this case, as seen in later sections.
- Case 2: Non-separable solution - For this case it follows that for any $\tilde{\mathbf{x}} \in \mathbb{R}^{2K}$ and $\alpha > 0$, as $\alpha \to \infty$, $\mathcal{L}(\alpha\tilde{\mathbf{x}}) \to \infty$. Since the negative log-likelihood $\mathcal{L}(\cdot)$ is convex, the minimizing value $\mathbf{x}^*$ is bounded, i.e., $||\mathbf{x}^*|| < \infty$. The significance of analyzing this case is presented after the smoothness analysis of the likelihood (see Remark 2).

3) Smoothness: The function $\mathcal{L}(\mathbf{x})$ is β-smooth if

$$\mathcal{L}_\beta(\mathbf{x}) = \frac{\beta}{2}\,||\mathbf{x}||^2 - \mathcal{L}(\mathbf{x}) \tag{18}$$

is convex [50]. Utilizing (8b) and (10c), the Hessian for $\mathcal{L}_\beta(\mathbf{x})$, $\mathcal{H}_{\mathbf{x}}^{\beta}$, is given by

$$\mathcal{H}_{\mathbf{x}}^{\beta} = \beta\,\mathbf{I} - \mathbf{H}^{\mathrm{T}}\operatorname{diag}(\zeta''(\mathbf{G}\mathbf{x}))\,\mathbf{H}. \tag{19}$$

In order to show the Hessian to be PSD, consider any vector $\mathbf{z} \in \mathbb{R}^{2K}$. It follows that

$$\begin{aligned} \mathbf{z}^{\mathrm{T}}\mathcal{H}_{\mathbf{x}}^{\beta}\mathbf{z}^{\mathrm{T}} &= \beta\,\mathbf{z}^{\mathrm{T}}\mathbf{I}\mathbf{z} - \mathbf{z}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\operatorname{diag}(\zeta''(\mathbf{G}\mathbf{x}))\,\mathbf{H}\mathbf{z} \\ &> \beta\,\mathbf{z}^{\mathrm{T}}\mathbf{I}\mathbf{z} - \mathbf{z}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathbf{H}\mathbf{z} \\ &= \beta\,||\mathbf{z}||^2 - ||\mathbf{H}\mathbf{z}||_2^2 \\ &\geq ||\mathbf{z}||^2(\beta - ||\mathbf{H}||_2^2), \end{aligned} \tag{20}$$

where $||\mathbf{H}||_2$ is the ℓ_2-norm of the matrix $\mathbf{H}$. The inequalities in (14) and the Cauchy-Schwartz inequality are utilized in deriving the inequalities in (20). The Hessian is PSD if

$$\beta \geq ||\mathbf{H}||_2^2. \tag{21}$$

Thus, the cdf-based likelihood is a smooth function with the smoothness parameter β lower bounded by $||\mathbf{H}||^2$. Based on this, the following points are presented.

- The smoothness parameter thus depends on the chosen channel matrix. This captures the dimensionality of the problem, i.e., the number of users and MIMO antennas.
- The optimal step size for the improved GD method, i.e., $\alpha^{(t)}$, is given by $1/\beta$ [50]. If the number of users or antenna elements increases, the optimal step size reduces.

Remark 2. Note that this smoothness characterization is valid for the likelihood, irrespective of the solution being drawn from Case 1 or Case 2. The optimal value $\mathbf{x}^*$ is bounded for Case 2; hence the existing results for smooth functions [50] can be applied to this case. For Case 1 however, the choice of GD parameters and subsequent convergence analysis, in the absence of a finite minimizer warrants, explicit analysis. This case presents the high-SNR saturation regime of receiver algorithm. Thus, for the remainder of this work, all subsequent analysis and algorithm design is performed from the perspective of operating under Case 1.

B. Improved Gradient Descent for log-CDF likelihood

One of the limitations of applying GD to the CDF-based likelihood (9) is the evaluation of the gradient (10b) for large negative arguments. The Gaussian CDF quickly decreases to zero for decreasing negative values of z and thus the numerical evaluation of the gradient runs into instabilities due to inability of capturing such low values within floating point precision. The significance of implementing an algorithm, which computes the gradient in a numerically stable manner, is the utilization of the same for one-bit MIMO detection on practical hardware using finite precision arithmetic like fixed point or floating point operations. Such finite precision arithmetic is unable to capture the rapid decay of the Gaussian CDF for low negative values, i.e., $[\mathbf{G}\mathbf{x}]_i \to -\infty$, which is always rounded off to zero. One of the possible fixes to address the gradient computation instability involves a scalar denominator regularization, generating the gradient expression of the form

$$\nabla_{\mathbf{x}} = -\mathbf{G}^{\mathrm{T}} \frac{\phi(\mathbf{G}\mathbf{x})}{\Phi(\mathbf{G}\mathbf{x}) + \epsilon}, \tag{22}$$

with ϵ as a fixed small scalar value, to prevent numerical errors of dividing by zero. However, since the Gaussian PDF $\phi(\cdot)$ also decays to zero for decreasing negative arguments, this results in $\nabla_{\mathbf{x}} \to 0$ as $[\mathbf{G}\mathbf{x}]_i \to -\infty \, \forall \, i$. This regularization approach does not correctly compute the value of $\nabla_{\mathbf{x}}$ when $[\mathbf{G}\mathbf{x}]_i \to -\infty$, as seen in the subsequent analysis. Thus, there is a need to utilize an improved surrogate gradient that is both numerically stable, and accurately computes the gradient value for the GD algorithm.

A key observation here is that the gradient computation of (10b) does not necessarily require the computation of the individual PDF and CDF terms $\phi(z)$ and $\Phi(z)$, respectively; only the ratio of the two terms is essential. The core principle to improve robustness for the CDF-based GD algorithm thus involves a numerically efficient method to evaluate the ratio $\phi(z)/\Phi(z)$ for $z < z_{\text{thresh}}$. Here z_{thresh} is an empirically evaluated threshold such that the CDF cannot be numerically evaluated accurately to floating point precision for negative values beyond this value.

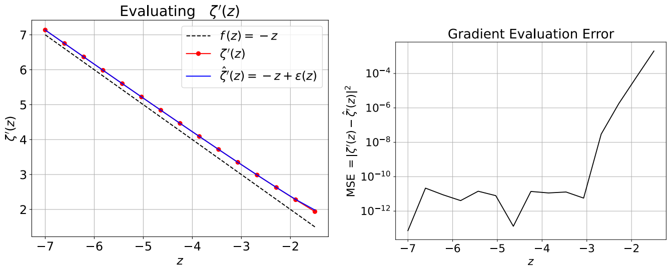
Lemma 1 derives an upper and lower bound for the ratio $\zeta'(z)$ in (10b) for $z < 0$. Based on this lemma, it is evident that the value of $-\zeta'(z)$ asymptotically approaches the linear function $f(z) = -z$ as $z \to -\infty$. For negative values below a threshold z_{thresh}, a surrogate value $\hat{\zeta}'(z)$, using an empirically evaluated residual $\epsilon(z)$, is evaluated as

$$\hat{\zeta}'(z) = -(-z + \epsilon(z)), \text{ for } z < z_{\text{thresh}}. \tag{23}$$

As seen in Lemma 1, the value of $\zeta'(z)$ is sandwiched between $-z$ and $-z-1/z$ for $z < 0$. Thus $\epsilon(z) \to 0$ as $z \to -\infty$. This residual is empirically evaluated, utilizing the series expansion

$$\epsilon(z) = -\frac{1}{z} + \frac{c_2}{z^2} + \frac{c_3}{z^3} + \frac{c_4}{z^4} + \ldots \tag{24}$$

Using the least squares fit, the coefficient values are evaluated as $c_2 = -0.09, c_3 = 1.80, c_4 = 1.95$. Further, we observe that the computation of the residual up to 4 orders, i.e., $\frac{c_4}{z^4}$ is sufficient for the desired accuracy in gradient evaluation. The plots in Fig. 1 illustrate the fit of the gradient using (23)-(24).



(a) (b)

Fig. 1. Accuracy of the numerically stable gradient of the CDF-based likelihood (a) Comparing the curve fit of (10b) and (25), (b) Mean square error of using the approximation (25).

Based on Fig. 1, the evaluation of the gradient using (23) approximates the actual gradient to a high degree of accuracy for large negative values. Thus, the surrogate value $\hat{\zeta}'(z)$ to compute the ratio $\zeta'(z) = -\frac{\phi(z)}{\Phi(z)}$ in (10b) is given by

$$\hat{\zeta}'(z) = \begin{cases} -\Big(-z - \frac{1}{z} + \frac{c_2}{z^2} + \frac{c_3}{z^3} + \frac{c_4}{z^4} \Big), & \text{for } z < z_{\text{thresh}} \\ -\frac{\phi(z)}{\Phi(z)}, & \text{for } z \geq z_{\text{thresh}}. \end{cases} \tag{25}$$

Using the value of (25), the surrogate gradient is computed using (8a). Since the gradient computation for large negative arguments is now computed using linear and rational polynomial terms, (25) presents an improved framework for CDF-based gradient computation. In particular,

- *Numerical stability:* Deviating from the use of the Gaussian CDF and PDF, the gradient computation can now be computed for negative values with large magnitudes, without running into divide-by-zero instabilities.
- *Gradient value:* As evidenced by the curve fitting and MSE plots in Fig. 1, this surrogate gradient closely matches the actual value, with this gap reducing as the magnitude of z gets larger.

An improved GD algorithm for the log-CDF based likelihood is presented in Algorithm 1, with the following salient features.

- The vector $\mathbf{G}\mathbf{x}^{(t)}$ for the t^{th} iteration is evaluated.
- Based on a pre-determined threshold [2] $z_{\text{thresh}} = -5$, each index of the vector $\mathbf{G}\mathbf{x}^{(t)}$ is classified as $\mathcal{I}^+$ or $\mathcal{I}^-$ (see line 2-3 in Algorithm 1).
- Depending on the classification of each index, $\zeta'(\big[\mathbf{G}\,\mathbf{x}^{(t)}\big]_i)$ is evaluated using (25).
- The final output $\mathbf{x}^{(T)}$ is normalized to the M-QAM magnitudes, as required.

C. Accelerated Gradient Descent for faster convergence

The general accelerated gradient descent (AGD) method for a convex β-smooth function was first introduced in [51] as an algorithm to attain the optimum oracle complexity for smooth

[2] empirically chosen threshold based on numerical results (see Fig. 1(b))

Algorithm 1 Improved GD for log-CDF likelihood

```
Input: $T$, $\mathbf{G}$, $\mathbf{x}^{(0)} = 0$, $\{\alpha^{(t)}\}_{t=0}^{T-1}$, $z_{\text{thresh}}$
Output: $\mathbf{x}^{(T)}$
for $t = 0$ to $T$ do
    $\mathcal{I}^+ = \{i \mid \big[\mathbf{G}\,\mathbf{x}^{(t)}\big]_i \ge z_{\text{thresh}}\}$
    $\mathcal{I}^- = \{i \mid \big[\mathbf{G}\,\mathbf{x}^{(t)}\big]_i < z_{\text{thresh}}\}$
    if $i \in \mathcal{I}^-$,  evaluate $\zeta'\big(\big[\mathbf{G}\,\hat{\mathbf{x}}^{(t)}\}\big]_i\big)$ as (25), case 1
    if $i \in \mathcal{I}^+$,  evaluate $\zeta'\big(\big[\mathbf{G}\,\hat{\mathbf{x}}^{(t)}\}\big]_i\big)$ as (25), case 2
    Evaluate $\nabla_{\mathbf{x}}^{(t)}$ using (8a)
    Update $\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} - \alpha^{(t)}\nabla_{\mathbf{x}}^{(t)}$
end for
$\mathbf{x}^{(T)} \leftarrow \eta\,\frac{\mathbf{x}^{(T)}}{||\mathbf{x}^{(T)}||}$
```

Algorithm 2 Accelerated GD for log-CDF likelihood

```
Input: $T$, $\mathbf{G}$, $\mathbf{x}^{(0)} = 0$, $\mathbf{d}^{(0)} = 0$ $\{\alpha^{(t)}\}_{t=0}^{T-1}$, $z_{\text{thresh}}$, $\gamma$
Output: $\mathbf{x}^{(T)}$
for $t = 0$ to $T$ do
    Evaluate $\hat{\mathbf{x}}^{(t)} = \mathbf{x}^{(t)} + \mathbf{d}^{(t)}$
    $\mathcal{I}^+ = \{i \mid \big[\mathbf{G}\,\hat{\mathbf{x}}^{(t)}\}\big]_i \ge z_{\text{thresh}}\}$
    $\mathcal{I}^- = \{i \mid \big[\mathbf{G}\,\hat{\mathbf{x}}^{(t)}\}\big]_i < z_{\text{thresh}}\}$
    if $i \in \mathcal{I}^-$,  evaluate $\zeta'\big(\big[\mathbf{G}\,\hat{\mathbf{x}}^{(t)}\}\big]_i\big)$ as (25), case 1
    if $i \in \mathcal{I}^+$,  evaluate $\zeta'\big(\big[\mathbf{G}\,\hat{\mathbf{x}}^{(t)}\}\big]_i\big)$ as (25), case 2
    Evaluate $\nabla_{\hat{\mathbf{x}}^{(t)}}^{(t)}$ using (8a)
    Update $\mathbf{x}^{(t+1)} = \hat{\mathbf{x}}^{(t)} - \alpha^{(t)}\nabla_{\hat{\mathbf{x}}}^{(t)}$
    Update $\mathbf{d}^{(t+1)} = \gamma\,(\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)})$
end for
$\mathbf{x}^{(T)} \leftarrow \eta\,\frac{\mathbf{x}^{(T)}}{||\mathbf{x}^{(T)}||}$
```

convex functions, and has since been widely applied to various applications in signal processing [52].

Applying AGD to the CDF-based one-bit likelihood optimization (9), gives the update

$$\mathbf{d}^{(t)} = \gamma^{(t)}\,(\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}) \tag{26a}$$

$$\hat{\mathbf{x}}^{(t)} = \mathbf{x}^{(t)} + \mathbf{d}^{(t)} \tag{26b}$$

$$\mathbf{x}^{(t+1)} = \hat{\mathbf{x}}^{(t)} - \alpha^{(t)}\nabla_{\mathbf{x}}\mathcal{L}(\hat{\mathbf{x}}^{(t)}). \tag{26c}$$

Here, $\mathbf{d}^{(t)}$ is the momentum update at the t^{th} iteration, which is a step taken in addition to the gradient step. The scalar $\gamma^{(t)}$ is the weighting coefficient for the momentum. The gradient $\nabla_{\mathbf{x}}\mathcal{L}(\hat{\mathbf{x}}^{(t)})$ is evaluated using the improved gradient method described in Sec. III-B.

The AGD algorithm, utilizing the improved GD update for the CDF likelihood, is presented in Algorithm 2. Different from the GD, i.e., Algorithm 1, AGD is able to modify the update step with an additional correction from the gradient direction, determined by the previous estimates. The momentum $\mathbf{d}^{(t)}$ in (26a) accumulates the gradients from the previous iterations, preventing the algorithm slowdown due to vanishing gradient [52]. The momentum endows a “speed" to the GD algorithm, preventing saturation in such regions of very low gradient values. This is particularly effective for speeding up the likelihood decrease for the CDF-based likelihood optimization without any finite minimizer, as detailed next.

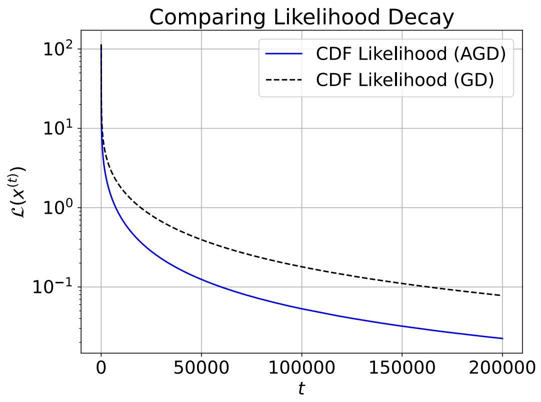


Fig. 2. Comparing decrease in CDF-based likelihood for AGD vs GD.

D. Likelihood decay for the GD-based algorithms

Through this section we analyze the likelihood decay performance for the unconstrained GD Algorithms 1 and 2.

1) Likelihood decay for GD: The likelihood decay for smooth functions with a finite minimizer and minima has been extensively analyzed [50]. In particular, the GD iterations decrease the likelihood function at the rate $1/t$. However, as stated in Sec III-A, there is no value that achieves the infimum of the likelihood, i.e., $\mathcal{L}(\mathbf{x}) > 0\,\forall\,\mathbf{x} \in \mathbb{R}^{2K}$ (see Remark 2). The convergence is thus analyzed to a surrogate minimum ϵ of the likelihood. The following theorem provides the likelihood decay after T iterations of the improved GD algorithm.

Theorem 1. *For a given surrogate minimum ϵ, the likelihood decay after T steps of the GD algorithm, i.e., Algorithm 1, with step size $\alpha^{(t)} = \frac{1}{\beta}$, is given by*

$$\mathcal{L}(\mathbf{x}^{(T)}) - \epsilon \le \frac{\lambda_0(\epsilon)}{T + \lambda_1(\epsilon)}, \tag{27}$$

where the scalars $\lambda_0(\epsilon)$ and $\lambda_1(\epsilon)$ are dependent on ϵ.

The proof[3] follows the same steps as the general decay rate analysis using a finite minimizer [50], with the surrogate minimum ϵ for the likelihood appropriately chosen.

Theorem 1 provides the best possible convergence rate for GD, utilizing the optimally chose step size, i.e., $\alpha^{(t)} = 1/\beta$, for a finite horizon GD algorithm.

2) Comparing likelihood decay for GD and AGD: For a general β-smooth function $f(\mathbf{x})$, the AGD has been proven to converge to the minimum as [50], [51]

$$f(\mathbf{x}^{(T)}) - f(\mathbf{x}^*) < \frac{c_1}{T^2}. \tag{28}$$

Applying the AGD to the recovery of symbols by minimizing the one-bit likelihood (9) shows a similar gain in convergence rate. This is empirically illustrated in Fig. 2, comparing the likelihood convergence rate to the infimum for GD vs AGD. As seen from the plots, the likelihood decays to a much lower value for AGD, with the gap to the GD-based likelihood decay

[3] The derivation of the proof for Theorem 1 requires the definition of a surrogate minimizer $\mathbf{x}_\epsilon^*$ that attains the surrogate minima ϵ. The complete proof of likelihood decay, utilizing the analysis of the surrogate minimizer, has been provided in [53].

This article has been accepted for publication in IEEE Transactions on Wireless Communications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TWC.2024.3439648

increasing with T. This empirically illustrates the strength of using the AGD for the unconstrained optimization (9).

Through theoretical bounds on likelihood decay, as well as the empirical results for AGD, we illustrate that unconstrained GD-based techniques will be able to converge arbitrarily close to the infimum, provided that there are no constraints on the number of iterations. This greatly scales the GD horizon T, proving infeasible for practical receivers, operating to minimize computational complexity. Resilient and simplified GD for practical receivers is added through *(i)* Improved surrogate likelihoods to allow for larger step sizes and thereby speed up convergence, and *(ii)* Projected GD to efficiently converge to the solution within the constrained set $\mathcal{M}^{2K}$. Each of these is elaborated in Sections IV and V, respectively.

IV. Improved CDF Surrogates for Modeling One-bit Likelihood

This section explores surrogate functions of the CDF, focusing primarily on the logistic regression (LR), to model an approximate one-bit likelihood for signal recovery. Insights into the improved likelihood decay for the LR are provided, followed by the GD algorithms for this likelihood.

A. Modeling one-bit likelihood through logistic regression

The approximation for the Gaussian CDF using the sigmoid function was first proposed in [54]. This was initially applied to the one-bit MIMO receiver in [40], where, motivated by the utilization of the sigmoid function as a prevalent nonlinear activation in DNNs, the GD-based receiver was implemented as an unfolded DNN, i.e., the OBMNet.

The LR-based likelihood expression is evaluated by substituting the value of the sigmoid function $\sigma(z)$ for the general $\zeta(z)$ in (7), generating the following expressions

$$\zeta(z) = -\log\sigma(z) \tag{29a}$$

$$\zeta'(z) = -\sigma(-z) \tag{29b}$$

$$\zeta''(z) = \sigma(z)(1-\sigma(z)). \tag{29c}$$

The unconstrained ML optimization is given by

$$\hat{\mathbf{x}}_{\mathrm{ML}} = \underset{\mathbf{x}\in\mathbb{R}^{2K}}{\operatorname{argmin}} \sum_{i=1}^{2N} -\log\sigma\big(y_i\mathbf{h}_i^{\mathrm{T}}\mathbf{x}\big). \tag{30}$$

The following is the analysis the LR-based likelihood.

1) Convexity: On substituting the Hessian for the LR-based likelihood (31) in (8b), it is evident that each element of the matrix $\mathrm{diag}(\zeta''(\mathbf{Gx}))$ is always positive. Thus the Hessian is positive semi-definite (PSD). Therefore, the LR-based likelihood is a convex function of $\mathbf{x}$.

2) Smoothness: Analogous to (19), $\mathcal{H}_\beta$ is evaluated as

$$\mathcal{H}_\beta = \beta\,\mathbf{I} - \mathbf{H}^{\mathrm{T}}\mathrm{diag}(\sigma(\mathbf{Gx})(1-\sigma(\mathbf{Gx})))\mathbf{H}. \tag{31}$$

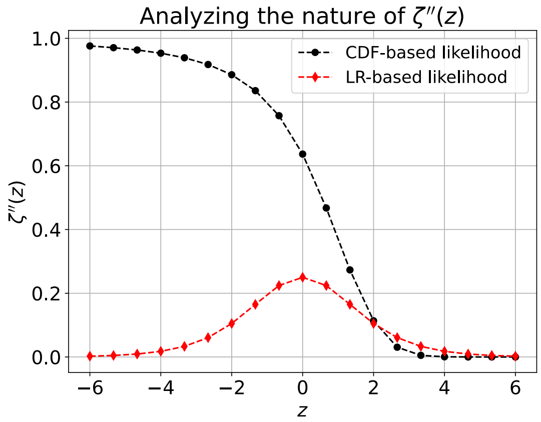


Fig. 3. Comparing the values of $\zeta''(z)$ for the LR and CDF-based likelihoods.

For any vector $\mathbf{z}\in\mathbb{R}^{2K}$. We have

$$\begin{aligned}\mathbf{z}^{\mathrm{T}}\mathcal{H}_{\mathbf{x}}^{\beta}\mathbf{z}^{\mathrm{T}} &= \beta\,||\mathbf{z}||^2 - \mathbf{z}^{\mathrm{T}}\mathbf{H}^{\mathrm{T}}\mathrm{diag}(\sigma(\mathbf{Gx})(1-\sigma(\mathbf{Gx}))\mathbf{Hz}\\ &\geq \beta\,||\mathbf{z}||^2 - ||\mathrm{diag}\big(\sqrt{\sigma(\mathbf{Gx})(1-\sigma(\mathbf{Gx}))}\,\big)\mathbf{Hz}||^2\\ &> \beta\,||\mathbf{z}||^2 - \frac{1}{4}||\mathbf{Hz}||_2^2\\ &\geq ||\mathbf{z}||^2(\beta - \frac{1}{4}||\mathbf{H}||_2^2),\end{aligned} \tag{32}$$

where we utilize $\sigma(z)(1-\sigma(z)) \leq 1/4\ \forall\, z$, and the Cauchy-Schwartz inequality. The Hessian is PSD if

$$\beta \geq \frac{||\mathbf{H}||_2^2}{4}. \tag{33}$$

Comparing this to (21) gives $\beta_{\mathrm{LR}} = \frac{1}{4}\beta_{\mathrm{CDF}}$. Following the model-based selection of the step size $\alpha^{(t)} = 1/\beta$, the LR enables an increase of the step size by a factor of four.

B. Step size robustness of LR for GD

In addition to better smoothness characterization over the CDF, the LR offers additional robustness to larger step sizes $\alpha^{(t)} >> 1/\beta_{\mathrm{LR}}$, resulting in faster likelihood decay without diverging to the incorrect solution. This is attributed to the properties of the Hessian matrix; the plots in Fig. 3 pictorially show different behavior, which translates to increased robustness for LR. This is further elaborated below.

Consider the general likelihood expression $\mathcal{L}(\mathbf{x})$, given by (7). As described earlier, smoothness parameter β determines the step size for the GD algorithm. Applying the analysis from (20) and (32) to a general $\zeta(z)$, we evaluate the following bound on β

$$\beta \geq ||\mathrm{diag}\big(\sqrt{\zeta''(\mathbf{Gx})}\,\big)\,\mathbf{H}||_2^2. \tag{34}$$

Further, the maximum value for the RHS of (34) is given by

$$\beta_{\max} = \max_{\mathbf{x}\in\mathbb{R}^{2K}} ||\mathrm{diag}\big(\sqrt{\zeta''(\mathbf{Gx})}\big)\,\mathbf{H}||_2^2 = \Big[\max_{z\in\mathbb{R}}\zeta''(z)\Big]||\mathbf{H}||_2^2. \tag{35}$$

Choosing the step size $\alpha^{(t)}$ utilizing the value $\beta = \beta_{\max}$ is sufficient to guarantee convergence of GD over all $\mathbf{x}\in\mathbb{R}^{2K}$. This is the model-based limit chosen for the CDF and LR-based likelihoods, as given in Sec III-A and IV-A, respectively.

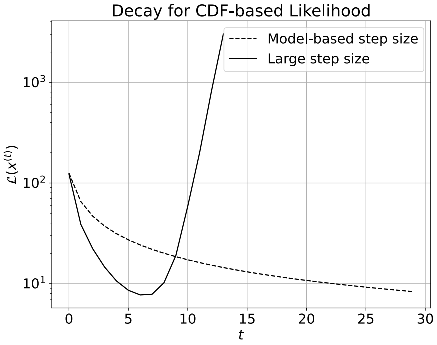


Fig. 4. Illustrating likelihood decay via GD for CDF-based likelihood with large step size, when compared to the model-based step size $\alpha^{(t)} = 1/\beta$.

However, a step size larger than this limit can be utilized by analyzing the value of the $\mathbf{x}^{(t)}$ for the attainment of $\beta = \beta_{\max}$. This, in turn, depends on the value of $\zeta''(z^{(t)})$, which is compared for both the LR and CDF-based likelihoods in Fig. 3. Based on the plots, we analyze this further.

1) LR-based likelihood: As seen by the curve for the LR-based likelihood in Fig. 3, the value for $\beta_{\max}$ is attained at $z = 0$, corresponding to the case $\mathbf{x} = 0$. However, the practical GD trajectory is considered via the attainment of the values of $\zeta''(z^{(t)})$ for two zones, as seen in Fig. 3: *(i)* The convergence zone, corresponding to the positive z-axis, and *(ii)* The divergence zone, corresponding to the negative z-axis.

- Convergence zone, i.e., $[\mathbf{Gx}]_i > 0 \ \forall i$: It has been shown that $||\mathbf{x}^{(t)}||$ increases unbounded with each GD iteration[4]. Thus the gap of the expression $\zeta''(z^{(t)})$ to the maximum value of 0.25 increases monotonically. This, in turn allows a much larger step size, i.e., $\alpha^{(t)} >> 1/\beta_{\max}$.
- Divergence zone, i.e., $[\mathbf{Gx}]_i < 0 \ \forall i$: The symmetry of the plot for $\zeta''(z^{(t)})$ plays an important role for the divergence zone as well. With increasing divergent behavior, i.e., increasing negative values of $z^{(t)}$, the gap of $\zeta''(z^{(t)})$ to the maximum value also increases. This further increases the maximum value of the step size that can be taken to move in the convergence direction. Since the value of $\zeta''(z^{(t)})$ can decrease to zero, there will always exist a point after which the GD algorithm (with a fixed step size) will move in the direction of convergence.

The same logic will also hold for intermediate behavior between convergence and divergence zones of the GD algorithm, wherein the GD algorithm will never indefinitely diverge.

2) CDF-based likelihood: For the CDF-based likelihood plots in Fig. 3, the same robustness as the LR will hold for the convergence zone. However, for the GD algorithm dynamics in the divergence zone, the dependence on the step size is inverted compared to the LR-based likelihood. With increasing negative values of $\zeta''(z^{(t)})$, the gap to $\beta_{\max}$ decreases. This implies the need to take smaller step sizes for convergence; using a larger step size will result in indefinite divergence of the GD algorithm. The comparison of the likelihood decay using the model based step size of $\alpha^{(t)} = 1/\beta$ and a large step size, i.e., $\alpha^{(t)} = 10/\beta$ is illustrated in Fig. 4. As seen by these plots, implementing GD with a large step size results in an unstable increase in the negative log-likelihood (6).[5] This increased sensitivity to step sizes larger than $1/\beta_{\max}$ results in use of step sizes smaller than LR, resulting in greater GD iterations for convergence.

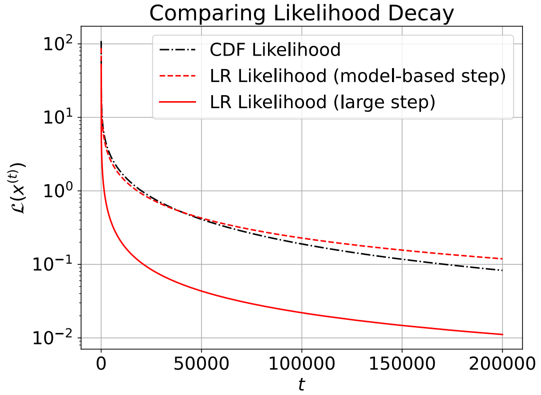


Fig. 5. Comparing decrease in CDF-based likelihood vs LR-based likelihood, with model-based step size and large step size, due to GD.

C. GD for LR-based likelihood and algorithm convergence

Applying GD to the likelihood (30) gives the GD update

$$\mathbf{x}^{(t+1)} = \mathbf{x}^{(t)} + \alpha^{(t)}\mathbf{G}^{\mathrm{T}}\sigma(-\mathbf{Gx}^{(t)}). \tag{36}$$

Similar to the application of the GD for the CDF-based likelihood, the choice of the step size parameter $\alpha^{(t)}$ is dependent on β_{LR}. In order to guarantee convergence of GD, the step size is chosen such that, $\alpha^{(t)} = 1/\beta_{\max}$, as explained in IV-A. However, the LR is more resilient to larger step sizes, allowing for faster convergence, as explained in Sec. IV-B.

The convergence analysis for GD algorithm of the LR follows the same analysis as the CDF-based likelihood, i.e., Theorem 1. However the specific constants will differ for the LR, owing to the different likelihood function. This convergence of the GD algorithm for the LR-based likelihood is illustrated in Fig. 5. The plots compare the GD convergence of the LR-based likelihood, using both the model-based step size $\alpha^{(t)} = 1/\beta_{\max}$ and the large step size $\alpha^{(t)} >> 1/\beta_{\max}$, to the CDF-based likelihood. All the GD-based algorithms decay as $1/t$, validating Theorem 1. The similar decay performance for the GD algorithms with the model-based step size $\alpha^{(t)} = 1/\beta$ is attributed to the fact that $\alpha_{\mathrm{LR}}^{(t)} = 4\,\alpha_{\mathrm{CDF}}^{(t)}$. Specifically, the chosen GD step sizes $\alpha_{\mathrm{LR}}^{(t)}$ and $\alpha_{\mathrm{CDF}}^{(t)}$ are within the same order of magnitude. However, the larger step size resilience for the LR-based likelihood is clearly seen by the significantly improved convergence.

Remark 3. Although the OBMNet [40] learns the step sizes $\alpha^{(t)}$ at each GD iteration, these do not need to be explicitly learnt. The evaluation of the Lipschitz constant β, theorizes the the required optimal step size. Additionally, empirical comparison of the unconstrained GD algorithm (OBMNet)

[4] Proof of Theorem 1 also proves monotonic decrease of $||\mathbf{x}^{(t)} - \mathbf{x}^{(t-1)}||$.

[5] This unstable increase in the negative log-likelihood has also been validated by the work in [39].

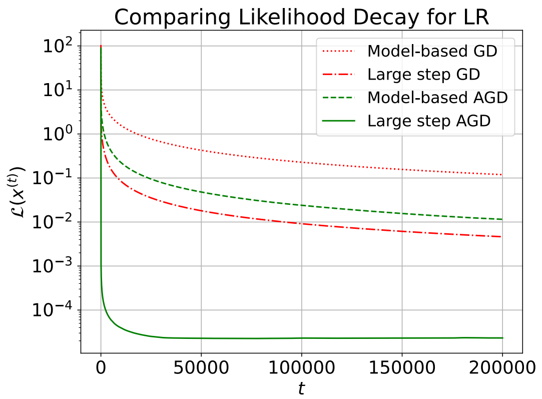


Fig. 6. Comparing decrease in LR-based likelihood for different variants of the GD algorithms.

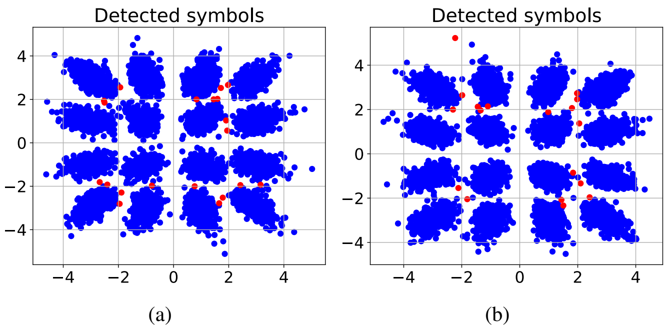


Fig. 7. Recovered 16-QAM constellation plots using unconstrained GD for M-QAM constellations with $K = 8$ users and $N = 128$ BS antennas (blue - correctly detected and red are incorrectly detected symbols). (a) CDF-based likelihood (9) (b) LR-based likelihood (30).

with learnt step sizes, and pre-defined step sizes, that aren't learnt, shows no difference in performance.

D. AGD for LR-based likelihood and algorithm convergence

The AGD algorithm, introduced in III-C, is applied to the LR-based likelihood. The resulting GD update step is

$$\hat{\mathbf{x}}^{(t)} = (1+\gamma^{(t)})\mathbf{x}^{(t)} - \gamma^{(t)}\mathbf{x}^{(t-1)} \tag{37a}$$

$$\mathbf{x}^{(t+1)} = \hat{\mathbf{x}}^{(t)} + \alpha^{(t)}\mathbf{G}^{\mathrm{T}}\sigma(-\mathbf{G}\hat{\mathbf{x}}^{(t)}). \tag{37b}$$

The entire T-step AGD (37) can also be equivalently implemented as a T-layer unfolded DNN, i.e., the accelerated-OBMNet (A-OBMNet). Different from the original OBMNet with individual disjoint subnetworks to implement (36), the A-OBMNet also additionally links the each subsequent OBMNet sub-network stage and adds increased robustness to the signal recovery. At each iteration, we can learn the scalar coefficients $\alpha^{(t)}$ and $\gamma^{(t)}$ using the loss function (49). As stated in Remark 3, for the A-OBMNet as well, we empirically observe no difference in performance for learning the parameters $\{\alpha^{(t)}, \gamma^{(t)}\}_{t=1}^{T}$ or choosing and fixing the values through the smoothness properties of the LR-based likelihood. The performance comparison of the A-OBMNet to the original OBMNet is given in Sec. VI.

V. Projected Gradient Descent - DNN-Aided Optimization for M-QAM Symbols

This section begins by elucidating the significance of the projection step. This is followed by the general two-tier projection strategy employed for the M-QAM constellation symbols. Finally, the entire projected AGD algorithm is implemented as an unfolded DNN, the A-PrOBNet.

A. Significance of M-QAM projection for GD

One of the main limitations of of applying the unconstrained GD algorithm, optimizing over $\mathbb{R}^{2K}$, for the recovery of symbols generated from the M-QAM constellation is symbol recovery with large cluster spread. The recovered symbols are illustrated in Fig. 7 for both the CDF-based as well as the LR-based likelihoods. The consequences of this large cluster spread on the unconstrained GD-based symbol recovery, specifically Algorithms 1 and 2, are explained below.

1) Slow rate of gradient decay: We begin by first understanding the road to convergence, specifically through the gradient decay. Consider the expression for the gradient at the t^{th} iteration for a general likelihood, i.e., (8a),

$$\nabla_{\mathbf{x}}^{(t)} = \mathbf{G}^{\mathrm{T}}\,\zeta'(\mathbf{G}\mathbf{x}) = \sum_{i=1}^{2N}\mathbf{g}_i\,\zeta'(y_i\mathbf{h}_i^{\mathrm{T}}\mathbf{x}^{(t)}), \tag{38}$$

where $\mathbf{g}_i$ and $\mathbf{h}_i$ are the i^{th} rows of the matrices $\mathbf{G}$ and $\mathbf{H}$, respectively. Firstly, the function $\zeta'(\cdot)$ is strictly positive-valued and the rows are drawn from a normal distribution, hence the gradient decays to zero if $\zeta'(y_i\mathbf{h}_i^{\mathrm{T}}\mathbf{x}^{(t)}) \to 0, \forall i$. Secondly, the elements of the input vector $\mathbf{x}$ are drawn from the M-QAM constellation points. Both these factors imply that for all i, $y_i\mathbf{h}_i^{\mathrm{T}}\mathbf{x}^{(t)}$ should be large positively-scaled constellation symbols, with very low cluster spread, in order for the gradient to decay to zero.

The presence of large symbol cluster spread affects the positivity of the expression $y_i\mathbf{h}_i^{\mathrm{T}}\mathbf{x}^{(t)}$ for some indices, even though the recovered symbols are within the right symbol boundaries. This is an induced negative bias due to large cluster spread. Due to this negative bias, the GD is significantly slowed down, correcting for both incorrectly detected symbols as well as reducing the cluster spread of correctly detected symbols. This makes the GD process very slow and inefficient, if applied by itself, as seen from the different convergence results of Sec III and IV. The slow convergence is corrected through the use of projected GD, as explained below.

2) GD step – projection step positive feedback: The effect of the GD-step - projection step positive feedback is demonstrated by applying the per-iteration projection on the AGD algorithm for the CDF-based likelihood, i.e., Algorithm 2, owing to its optimal performance for the unconstrained optimization (9). The improvement in convergence via projected AGD is pictorially illustrated in Fig. 8, portraying the symbol error rate (SER) reduction over the AGD iterations. In the absence of any projection step at each iteration, the SER quickly saturates and further reduction is very slow, i.e., the rate of symbol error correction doesn't follow the rate of likelihood decay. In the absence of projection, the AGD iteration itself works towards reducing the cluster spread, which does not have any bearing on the SER. On the other hand, the the two-tier

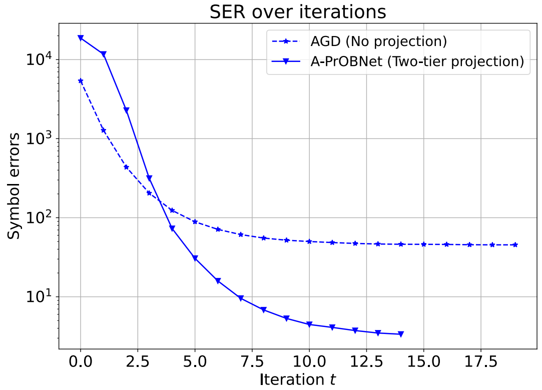


Fig. 8. Iteration dynamics of SER: Comparing CDF-based AGD with and without projection. Recovery of 16-QAM symbols received from $K = 8$ users at a BS antenna with $N = 128$ antennas for SNR = 25 dB.

projection (explained in Sec. V-B) improves performance and speeds up convergence. The projection step, by itself, does not help correct symbol errors; it is only responsible for improved regularization of the recovered symbols into smaller clusters. This reduces the negative bias and the AGD iteration is able to efficiently correct the M-QAM symbol errors in the subsequent step, which is further helpful to better regularize the recovered M-QAM symbols, and so on. This creates positive feedback with the projection step helping the GD step, and the AGD step helping the projection step, to greatly speed up convergence.

Remark 4. Although the above analyzes symbol recovery for Algorithms 1 and 2 for the CDF-based likelihood, these observations are general to the unconstrained optimization and also apply to the surrogate likelihood based on the LR.

B. Two-tier projected GD framework

The use of a learnt M-QAM projection has been applied for one-bit MIMO in [5], which utilizes quantizers based on the rectified linear unit (ReLU) function. However, one of the major limitations of this approach is the absence of a structure for the projection, causing the detection to undergo unstable initial GD iterations, before being stabilized in the later stages. Differently, this work introduces a two-tier structured projection applied to the GD and AGD algorithms. This is explained below.

1) Tier 1 - Hypercube projection: The Tier 1 projection maps each GD iterand to the M-QAM $2K$-dimensional hypercube, defined as $\mathcal{S}_{\text{cube}} \in \mathbb{R}^{2K}$ such that

$$\mathcal{S}_{\text{cube}} = \{\mathbf{x} \,|\, |\mathbf{x}[i]| \leq s_{\max},\, \forall i = 1, 2, \ldots, 2K\}, \tag{39}$$

where $s_{\max}$ is the maximum value of the M-QAM quadrature component. We define the projection operation $\mathcal{P}_{\text{cube}} : \mathbb{R}^{2K} \to \mathcal{S}_{\text{cube}}$ through the element-wise transformation

$$\big[\mathcal{P}_{\text{cube}}(\mathbf{x})\big]_i = \begin{cases} \mathbf{x}[i], \text{ if } |\mathbf{x}[i]| \leq s_{\max} \\ s_{\max}, \text{ otherwise.} \end{cases}, \forall i = 1, 2, \ldots 2K. \tag{40}$$

Applying this, we have the following projected GD update

$$\hat{\mathbf{x}}^{(t+1)} = \mathbf{x}^{(t)} - \alpha^{(t)} \nabla_{\mathbf{x}}^{(t)} \tag{41a}$$

$$\mathbf{x}^{(t+1)} = \mathcal{P}_{\text{cube}}(\hat{\mathbf{x}}^{(t+1)}). \tag{41b}$$

Similarly, applying this projection to the AGD method gives

$$\hat{\mathbf{x}}^{(t+1)} = \mathbf{x}^{(t)} + \mathbf{d}^{(t)} - \alpha^{(t)} \nabla_{\mathbf{x}+\mathbf{d}}^{(t)} \tag{42a}$$

$$\mathbf{x}^{(t+1)} = \mathcal{P}_{\text{cube}}(\hat{\mathbf{x}}^{(t+1)}) \tag{42b}$$

$$\mathbf{d}^{(t+1)} = \gamma^{(t)}(\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}). \tag{42c}$$

The improvement to Algorithms 1 and 2 through Tier 1 projection is elaborated through the points below.

- Bounding each $\mathbf{x}^{(t)}[i]$ as $-s_{\max} \leq \mathbf{x}^{(t)}[i] \leq s_{\max}$, the GD update (41) converges faster due to the larger value of the smoothness parameter β over the set $\mathcal{S}_{\text{cube}}$.
- The Tier 1 projection is linear inside the M-QAM hypercube, which is a soft projection and hence not too restrictive. This allows for more flexible symbol recovery and error correction in the initial stages of the GD algorithm. This flexibility in projection enables the formation of the initial M-QAM constellation clusters for the recovered symbols, which are efficiently fined-tuned using the subsequent projection method.

2) Tier 2 - Gaussian denoiser: The Tier 2 projection $\mathcal{P}_{\text{QAM}}$ maps from the set $\mathcal{S}_{\text{cube}} \to \mathcal{S}_{\text{cube}}$ through an exhaustive weighted sum of all the symbols in $\mathcal{M}^{2K}$. This requires modeling the posterior distribution of the transmitted symbols.

The vector of M-QAM transmitted symbols from the K different users is given by $\mathbf{s}$. Each GD iterand $\mathbf{x}^{(t)}$ after the Tier 1 projection (40) is modeled as

$$\mathbf{x}^{(t)} = \mathbf{s} + \triangle\mathbf{s}^{(t)}, \tag{43}$$

where the residual $\triangle\mathbf{s}^{(t)}$ is the deviation from the transmitted symbols, drawn from the Gaussian distribution $\mathcal{N}(\mathbf{0}, (\sigma^{(t)})^2\mathbf{I})$. We assume that this residual component at the t^{th} iteration $\triangle\mathbf{s}^{(t)}$ is independent of the previous residuals $\{\triangle\mathbf{s}^{(t)}\}_{t=0}^{t-1}$. Further, we consider a uniform non-informative prior over all the symbols in $\mathcal{M}^{2K}$. The Tier 2 projection $\mathcal{P}_{\text{QAM}}$ is the MMSE estimate of the transmitted symbols using this estimation model for $\mathbf{x}^{(t)}$. Hence, using the modeled Gaussian distribution with the independent increment assumption, the Tier 2 projection at each iteration is given as

$$\hat{\mathbf{s}}^{(t)} = \mathcal{P}_{\text{QAM}}(\mathbf{x}^{(t)}) = \mathbb{E}_{\mathbf{s}|\mathbf{x}^{(t)}}(\mathbf{s}), \tag{44}$$

which is the posterior mean of the distribution $\Pr(\mathbf{s}|\mathbf{x}^{(t)})$. Using the Gaussian likelihood $f(\mathbf{x}^{(t)}|\mathbf{s})$ and the uniform prior $\Pr(\mathbf{s}) = 1/M^{2K}$, the MMSE estimate is given by

$$\hat{\mathbf{s}}^{(t)} = c^{(t)} \sum_{i=1}^{M^K} \mathbf{s}_i \exp\Big(-\frac{||\mathbf{x}^{(t)} - \mathbf{s}_i||^2}{2(\sigma^{(t)})^2}\Big), \tag{45}$$

where $c^{(t)} = \Big(\sum_{i=1}^{M^K} \exp\Big(-\frac{||\mathbf{x}^{(t)} - \mathbf{s}_i||^2}{2(\sigma^{(t)})^2}\Big)\Big)^{-1}$ is the normalization constant and $\mathbf{s}_i$ is the i^{th} element of $\mathcal{M}^{2K}$. The parameter $\sigma^{(t)}$ is a learnt over each iteration (see Sec. V-C). Since $\mathbf{s}^{(t)}$ consists of $2K$ independent components, corresponding to the real and imaginary parts of K users, the element-wise

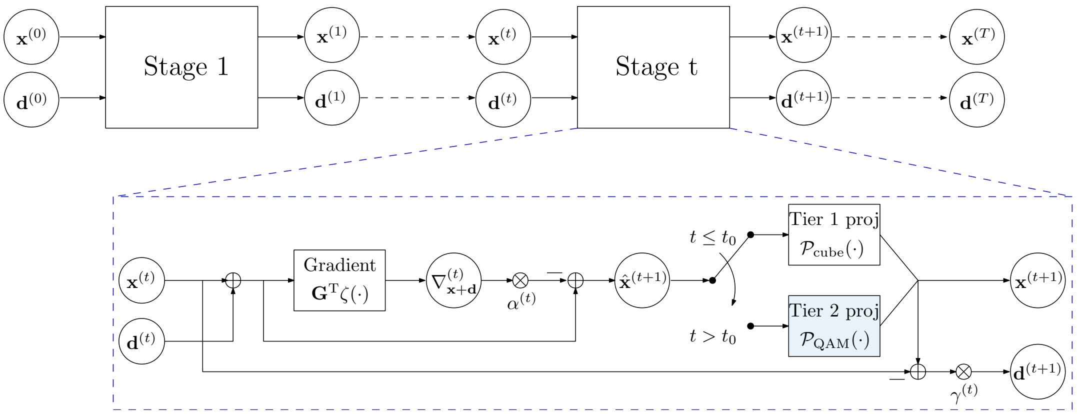


Fig. 9. Block diagram for the A-PrOBNet - Unfolded DNN to implement the projected AGD update (42). The blue shaded blocks in each stage represent the learnable parameters in the unfolded DNN.

evaluation of the Tier 2 projection is given by

$$\hat{\mathbf{s}}[i] = c^{(t)} \sum_{k=1}^{\sqrt{M}} s_k \exp\Big(- \frac{(\mathbf{x}^{(t)}[i] - s_k)^2}{2(\sigma^{(t)})^2} \Big), \tag{46}$$

where s_k is the k^{th} quadrature component of the M-QAM constellation. The equation (46) is the Gaussian denoiser, formed by a convex summation of all the elements in $\mathcal{M}^{2K}$. This convex projection clearly also maps to a point in the hypercube $\mathcal{S}_{\mathrm{cube}}$. Based on this projection, we have the following.

- Different from the Tier 1 projection (40), the Tier 2 projection is weighted by the ℓ_2 distance of the iterand $\mathbf{x}^{(t)}$ to each constellation point, via a Gaussian kernel. Thus, the values $\mathbf{x}^{(t)}[i]$ close to the constellation points $\{s_k\}$ are compactly clustered around these points. This enables reducing the cluster spread of the recovered constellation symbols.
- The iteration-dependent parameter $(\sigma^{(t)})^2$ quantifies the cluster spread of the recovered symbols. The initial iterations begin with a large value of $(\sigma^{(t)})^2$, allowing for flexible symbol error correction. The value of this parameter reduces with iterations, due to increasing confidence in detected symbol values, resulting in more compact clusters. This trend over the GD iterations is learnt from training data, as explained in the subsequent sub-section.

A threshold value t_0 represents the empirically evaluated iteration index to switch from the Tier 1 to the Tier 2 projection. Thus, the overall two-tier projected GD update is given by

$$\hat{\mathbf{x}}^{(t+1)} = \mathbf{x}^{(t)} - \alpha^{(t)} \nabla^{(t)}_{\mathbf{x}} \tag{47a}$$

$$\mathbf{x}^{(t+1)} = \mathcal{P}(\hat{\mathbf{x}}^{(t+1)}) = \begin{cases} \mathcal{P}_{\mathrm{cube}}(\hat{\mathbf{x}}^{(t+1)}), & \text{if } t \le t_0 \\ \mathcal{P}_{\mathrm{QAM}}(\hat{\mathbf{x}}^{(t+1)}), & \text{if } t > t_0. \end{cases} \tag{47b}$$

Similarly, the AGD update with the two-tier projection is

$$\hat{\mathbf{x}}^{(t+1)} = \mathbf{x}^{(t)} + \mathbf{d}^{(t)} - \alpha^{(t)} \nabla^{(t)}_{\mathbf{x+d}} \tag{48a}$$

$$\mathbf{x}^{(t+1)} = \mathcal{P}(\hat{\mathbf{x}}^{(t+1)}) = \begin{cases} \mathcal{P}_{\mathrm{cube}}(\hat{\mathbf{x}}^{(t+1)}), & \text{if } t \le t_0 \\ \mathcal{P}_{\mathrm{QAM}}(\hat{\mathbf{x}}^{(t+1)}), & \text{if } t > t_0 \end{cases} \tag{48b}$$

$$\mathbf{d}^{(t+1)} = \gamma^{(t)}(\mathbf{x}^{(t+1)} - \mathbf{x}^{(t)}). \tag{48c}$$

The unfolded DNN implementing the AGD algorithms with the two-tier projection is explained next.

C. Unfolded DNN implementation of projected AGD

Unfolding algorithms using DNNs have found much applicability in different areas of signal processing [55], [56]. These DNN frameworks leverage model-based information and update equations to address model mismatches or enhance algorithm performance through a DNN-aided step in the original algorithm. A key advantage of unfolded DNNs is their ability to significantly reduce the number of trainable parameters and the training time compared to other DNN frameworks. This efficiency makes unfolded DNNs particularly attractive for real-time and resource-constrained applications.

In the projected AGD update in (48), the constraint of projecting each GD iteration on the M-QAM subspace is addressed by learning the parameters of the Tier 2 Gaussian denoiser. As seen in Sec. III, the accelerated GD improves on the convergence over conventional GD for the unconstrained optimization (see Fig. 2). This advantage can also be utilized for the constrained optimization. The combination of the well tested framework for accelerated GD, combined with the learnable Gaussian denoiser for M-QAM projection makes the unfolded DNN framework ideally suited for implementation of the projected GD.

The proposed accelerated projected one-bit network (A-PrOBNet) is illustrated in Fig. 9. The following are the salient features for this framework.

- The T-step AGD algorithm is unfolded as a T-stage DNN, with each Stage t denoting a distinct sub-network.
- The initial inputs are provided as $\mathbf{x}^{(0)} = \mathbf{d}^{(0)} = \mathbf{0}$, empirically shown to have a well-conditioned initial gradient value to start the GD.
- Within each Stage t, the gradient is evaluated using a shallow neural network, with the two static weight matrices $\mathbf{G}$ and $\mathbf{G}^{\mathrm{T}}$ and the hidden layer nonlinearity $\zeta'(z)$. For the A-ProbNet, we implement the CDF-based likelihood and hence the element wise nonlinearity $\zeta'(z)$ is evaluated using the improved gradient method (25).
- The learnable parameters (denoted by the blue shaded box in Fig. 9) for the network are the scalers $\{\sigma^{(t)}\}_{t=t_0+1}^{T}$

This article has been accepted for publication in IEEE Transactions on Wireless Communications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TWC.2024.3439648

for the Tier 2 projection (46). The values of the different static parameters $\{\alpha^{(t)}, \gamma^{(t)}\}$ are chosen differently for different M-QAM constellations. This is elaborated in Sec. VI.

- Learning the Gaussian denoiser parameters $\{\sigma^{(t)}\}_t$ specializes each stage of the A-ProbNet to gradually reduce the cluster spread of the recovered symbols. As explained through Fig. 8 this has a significant effect on improving the rate of convergence.
- The A-ProbNet parameters are trained in an end-to-end manner, using the MSE loss for the ideal constellation symbols, given by

$$\mathcal{L} = \frac{1}{N_{\text{train}}} \sum_{n=1}^{N_{\text{train}}} ||\tilde{\mathbf{x}}_n - \tilde{\mathbf{x}}_{\text{train},n}||^2. \tag{49}$$

We now present some finer points through means of a brief discussion on the overall projected AGD framework.

D. Discussion

1) Significance of $s_{\max}$ for Tier-1 hypercube projection: The Tier 1 hypercube projection acts as a preconditioning step for the Gaussian denoising. The role of the Tier 2 Gaussian denoising is to reduce the cluster spread of the recovered symbols centered around the M-QAM points. Thus the most efficient utilization of this projection is observed for symbols with clusters centered around the M-QAM points. To this end, the use of projected GD or AGD with Tier 1 hypercube projection restricts the recovered symbols to large clusters around the M-QAM symbols. This simplifies the subsequent reduction in cluster spread using Gaussian denoising. If the value of $s_{\max}$ is increased beyond the maximum M-QAM quadrature value, the clusters are no longer centered around the M-QAM symbols, which affects the subsequent Gaussian denoising step (46). The Gaussian denoising, when applied to symbols outside the hypercube boundaries, will itself map the iterates $\mathbf{x}^{(t)}$ to the hypercube boundary. Particularly, the first few iterations of Gaussian denoising, applied after the Tier 1 projection, will be utilized in projecting to the hypercube and subsequently reducing the cluster spread, which isn't an efficient use of the Gaussian denoising. Thus the optimal choice for $s_{\max}$ for the Tier 1 hypercube projection is the maximum value of the M-QAM quadrature component.

2) Generalization of the learnt quantization-based projection: As stated earlier, the work in [5] also introduced a learnt quantization-based denoiser for M-QAM projection, utilizing the nearest neighbor. The general Gaussian denoiser for the proposed two-tier projection weights the symbol against all the constellation symbol values, adding more robustness and flexibility, especially in the initial iterations.

3) Loss function for end-to-end learning: The work in [46] introduced a novel regularized DNN loss function that captured both the MSE and symbol errors. This loss implicitly captured the effect of projection during DNN training. However, differently, this work does not utilize this regularized loss due to the explicit use of the projection operation. Further, the application of the loss on the final symbols with sharp Gaussian denoisers results in the MSE capturing the symbol errors exclusively. However, the use of a regularized loss is still relevant for iteration-dependent loss functions, utilizing and fine-tuning all the intermediate estimates $\{\mathbf{x}\}_{t=1}^{T}$[6].

4) Generalized Gaussian denoising: Through the Gaussian denoiser introduced in this work, a single scalar parameter $\sigma^{(t)}$ per iteration t denoises all the user symbols $x^{(t)}[i]$ of the multi-user recovered signal estimate $\mathbf{x}^{(t)}$. This has the potential to be generalized further, incorporating separate per-user denoising.

TABLE I
COMPUTATIONAL COMPLEXITY COMPARISON OF THE A-PROBNET WITH OTHER BENCHMARK ALGORITHMS.

Method	Complexity
n-ML	$\mathcal{O}(KNT)$
OBMNet	$\mathcal{O}(KNT)$
FBM-DetNet	$\mathcal{O}(KNT)$
A-PrOBNet	$\mathcal{O}(KNT) + \mathcal{O}(\sqrt{M}KT)$

E. Computation complexity

The comparison of the computational complexities for the different GD-based algorithms for one-bit MIMO detection is given in Table. I. As seen from this table, the A-PrOBNet possesses similar complexity as the existing benchmark algorithms. However, there are are two key differences.

1) Tier-2 Gaussian denoising: The complexity of $\mathcal{O}(\sqrt{M}KT)$ depends on the modulation order. However, with the use of massive MIMO systems, with large N, the complexity of gradient gradient evaluation, i.e., $\mathcal{O}(KNT)$, is the main rate-determining step.
2) Scalar operations: Optimizing the scalar operations in gradient computation plays a key role in reducing the execution time of the detection algorithm. In particular, the OBMNet and FBM-DetNet utilize the sigmoid scalar operation whereas the A-PrOBNet utilizes the evaluation of the Gaussian CDF and PDF. Numerical optimization of these scalar operations, or use of efficient approximations, will be key in improving the computational complexity of the A-PrOBNet.

VI. Experimental Results

A. Simulation setup

All the different prior works for one-bit MIMO receivers (see Sec. I) benchmark the algorithm for lower and higher order M-QAM constellations, i.e., QPSK and 16-QAM. However, all these approaches perform comparably for QPSK symbols. Hence, in order to show true robustness to higher order M-QAM, we perform detailed testing and benchmarking of this work for the 16-QAM constellation symbols.

The 16-QAM constellation symbols are transmitted from $K = 8$ users, $N = 128$ BS antennas with SNR $= \frac{\mathbb{E}(||\mathbf{Hx}||^2)}{\mathbb{E}(||\mathbf{n}||^2)}$ in the range 10 to 45 dB. This setup follows the standard multi-user 16-QAM simulations conducted in [36], [39], [40], [46]. The Rayleigh fading channel $\mathbf{H}$ is considered with each entry independently chosen from the $\mathcal{CN}(0, 1)$ distribution.

[6] outside the scope of this work

1) Performance benchmarks: We compare the proposed algorithm against the different model-based and learning based frameworks. *(i)* The N-ML algorithm from [36] is used to establish the original benchmark using the CDF-based likelihood. *(ii)* The OBMNet in [40] forms the original LR-based likelihood benchmark. *(iii)* The FBM-DetNet from [5] is the existing state-of-the-art benchmark, utilizing the learnt quantization-based projection to the M-QAM set.

2) Benchmark algorithm and network parameters: The n-ML [36] is executed for a maximum of $T = 500$ iterations, with a step size of 0.001, (to ensure convergence). Consistent with the benchmarks established in [40], the OBMNet is run for $T = 15$ iterations. The same parameters are also taken for the FBM-DetNet [5].

3) Improved GD, AGD and A-PrOBNet: The following are the parameters chosen for the different algorithms and networks introduced in this work in Sec. III-B, III-C and V-C.

- The improved GD, i.e., Algorithm 1, is run for $T = 100$ iterations, to ensure convergence of the likelihood. The step size $\alpha = 0.03$.
- The AGD, i.e., Algorithm 2, is run for $T = 20$ iterations. The momentum parameter γ is taken as 0.63 and step size $\alpha = 0.03$, based on empirical testing.
- The A-PrOBNet is run for $T = 15$ iterations. The momentum parameter $\gamma = 0.63$ and step size $\alpha = 0.03$. The denoiser parameters $\{\sigma^{(t)}\}_{t=0}^{T-1}$ are the only learnable parameters. The training for the DNN is similar to the training strategy in [46]. The network training is carried out via minibatch gradient descent, with the chosen batch size $N_{\text{train}} = 32$. In order to train the A-PrOBNet on the set of randomly generated Rayleigh channel matrices, each minibatch is generated from a different channel matrix $\mathbf{H}$, denoted by $\mathcal{B}_{\mathbf{H}}$. Based on the described system model (1)-(2), the minibatch set is generated as $\mathcal{B}_{\mathbf{H}} = \{\bar{\mathbf{x}}_n, \bar{\mathbf{z}}_n, \bar{\mathbf{y}}_n\}_{n=1}^{N_{\text{train}}}$. We utilize the MSE loss function (49). We practically implement minibatch gradient descent with the Adam update [57] for each training minibatch to keep a check on the learning rate. For regularization of DNN weights, we utilize weight decay to further increase resilience by preventing exploding network weights.

B. Intrinsic testing

In this sub-section the algorithms and DNNs proposed in this work are tested by varying the different parameters.

1) CDF-based likelihood performance: The performance for the improved GD and AGD, Algorithms 1 and 2 respectively, is evaluated for different number of total iterations in Fig 10. As seen from these plots, the improved GD performance saturates beyond $T = 50$ iterations. In addition, the the momentum-based GD clearly outperforms the GD, with significantly fewer iterations. The performance of both Algorithms 1 and 2 are limited due to the M-QAM amplitude-scaled unit sphere normalization. Further improvement is only possible by modifying the projection step as seen in the subsequent tests.

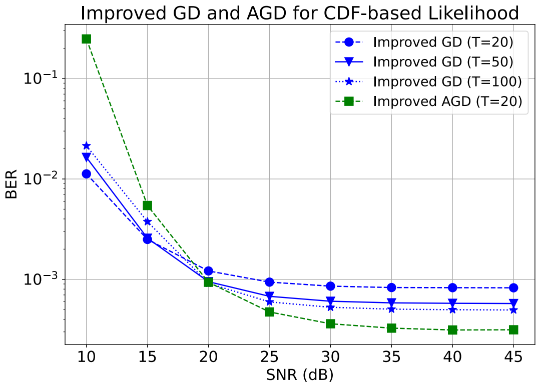


Fig. 10. Intrinsic comparison of improved GD and AGD performance for CDF-based likelihood for given simulation setup.

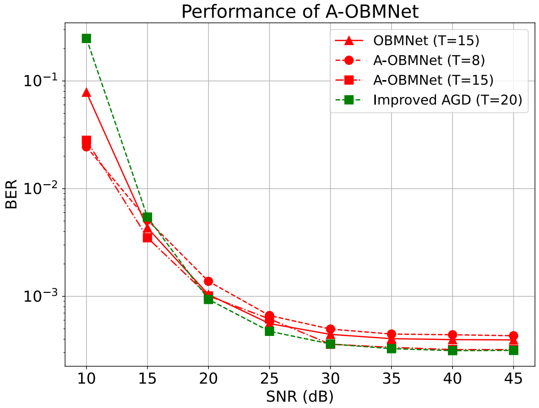


Fig. 11. Testing the performance of AGD on surrogate likelihood using LR, i.e., (37) for the given simulation setup.

2) Evaluating surrogate likelihoods: The performance plots of the surrogate likelihood, i.e., LR-based likelihood, using both GD and AGD update (36) and (37), respectively, are given in Fig. 11. As seen by the results, the LR-based likelihood converges in a fewer number of steps using AGD (see Fig. 6). The BER performance for the AGD update is comparable to the GD update using half the number of iterations. This is attributed to the step size robustness for the LR-based likelihood. However, as seen by the plots, increasing the number of iterations for AGD doesn't improve BER significantly. This shows that in addition to the robustness in step size as well as the advantages of accelerated GD, projection plays a vital role in improving BER performance.

3) Performance of projected AGD framework: We evaluate the role of the different projection strategies on the better performing AGD algorithm. The role of the different projection strategies is highlighted through the results in Fig. 12. As seen from these plots, Tier 1 projection is a significantly better strategy compared to projection on the M-QAM amplitude scaled unit sphere. The two-tier learnt strategy of the A-PrOBNet further improves on the BER by directly reducing the cluster spread.

C. Detection for general channel

We now compare the performance of the A-PrOBNet to the state-of-the-art recovery algorithms for a general channel matrix drawn from the distribution of Rayleigh distributed

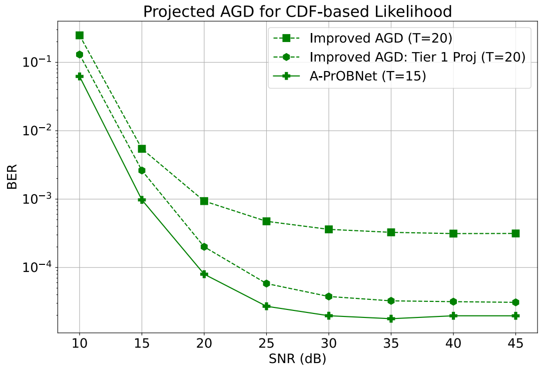


Fig. 12. Testing the role of different projection strategies on the CDF-based AGD for given simulation setup.

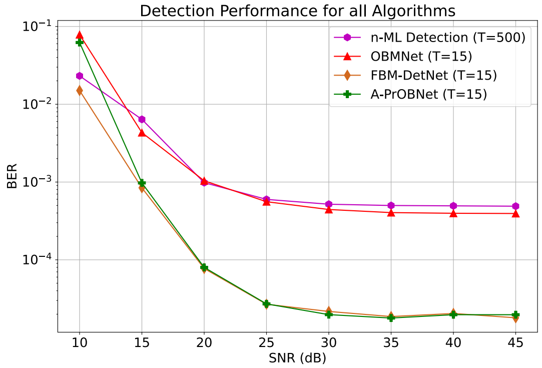


Fig. 13. Testing state of the art detection performance of all algorithms for given simulation setup.

channels. The recovery performance is given in Fig. 13. As can be seen from these plots, the performance of the proposed A-PrOBNet matches the current state-of-the-art performance of the FBM-DetNet with the same number of iterations (outperforming the OBMNet and n-ML using unit sphere normalization). However, differently, this algorithm does not make any additional approximations on the likelihood like utilization of a surrogate function. The A-PrOBNet thus establishes the limit of optimum performance for the original CDF-based likelihood without any additional approximations. Further, the two-tier projection is developed as a generalization of the quantization-based projection. The latter is clearly a better strategy at lower SNR values owing to weighting by a fewer M-QAM neighbors.

VII. Conclusions

This work provides insights into the ML optimization for one-bit MIMO receivers, enabling a better understanding of the GD-based signal recovery algorithm. The accelerated GD, with faster convergence, is introduced into the class of different algorithms. These insights are extended to the surrogate likelihood function, the logistic regression, explaining the improved robustness and speed of convergence. Finally, the significance of an effective per-iteration projection step is highlighted in the GD-based recovery. The accelerated GD, with a novel two-tier projection is unfolded into a T-stage DNN, the A-PrOBNet, to achieve state of the art performance.

Future work in this area involves the extension of this work to mmWave channels. The challenge of non-uniform power distribution among the different users makes joint-detection especially challenging for one-bit MIMO systems.

References

[1] A.-S. Bana, E. De Carvalho, B. Soret, T. Abrao, J. C. Marinello, E. G. Larsson, and P. Popovski, "Massive mimo for internet of things (iot) connectivity," *Physical Communication*, vol. 37, p. 100859, 2019.

[2] K. Shafique, B. A. Khawaja, F. Sabir, S. Qazi, and M. Mustaqim, "Internet of things (iot) for next-generation smart systems: A review of current challenges, future trends and prospects for emerging 5g-iot scenarios," *IEEE Access*, vol. 8, pp. 23 022–23 040, 2020.

[3] J. Mo, A. Alkhateeb, S. Abu-Surra, and R. W. Heath, "Hybrid architectures with few-bit adc receivers: Achievable rates and energy-rate tradeoffs," *IEEE Transactions on Wireless Communications*, vol. 16, no. 4, pp. 2274–2287, 2017.

[4] B. Murmann *et al.*, "Adc performance survey 1997-2020," in *IEEE Int. Solid-State Circuits Conf.(ISSCC) Dig. Tech. Papers VLSI Symp*, 2020.

[5] L. V. Nguyen, D. H. Nguyen, and A. L. Swindlehurst, "Deep learning for estimation and pilot signal design in few-bit massive mimo systems," *IEEE Transactions on Wireless Communications*, 2022.

[6] D. H. Nguyen, "Neural network-optimized channel estimator and training signal design for mimo systems with few-bit adcs," *IEEE Signal Processing Letters*, vol. 27, pp. 1370–1374, 2020.

[7] J. Mo, P. Schniter, and R. W. Heath, "Channel estimation in broadband millimeter wave mimo systems with few-bit adcs," *IEEE Transactions on Signal Processing*, vol. 66, no. 5, pp. 1141–1154, 2017.

[8] L. Fan, S. Jin, C.-K. Wen, and H. Zhang, "Uplink achievable rate for massive mimo systems with low-resolution adc," *IEEE Communications Letters*, vol. 19, no. 12, pp. 2186–2189, 2015.

[9] S. Jacobsson, G. Durisi, M. Coldrey, U. Gustavsson, and C. Studer, "Throughput analysis of massive mimo uplink with low-resolution adcs," *IEEE Transactions on Wireless Communications*, vol. 16, no. 6, pp. 4038–4051, 2017.

[10] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural networks*, vol. 2, no. 5, pp. 359–366, 1989.

[11] I. Goodfellow, Y. Bengio, and A. Courville, *Deep learning*. MIT press, 2016.

[12] F. Sohrabi, Z. Chen, and W. Yu, "Deep active learning approach to adaptive beamforming for mmwave initial alignment," *IEEE Journal on Selected Areas in Communications*, 2021.

[13] Y. Zhang, M. Alrabeiah, and A. Alkhateeb, "Reinforcement learning of beam codebooks in millimeter wave and terahertz mimo systems," *IEEE Transactions on Communications*, vol. 70, no. 2, pp. 904–919, 2021.

[14] A. Sant, A. Abdi, and J. Soriaga, "Deep sequential beamformer learning for multipath channels in mmwave communication systems," in *ICASSP 2022-2022 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2022, pp. 5198–5202.

[15] H. Ye, G. Y. Li, and B.-H. Juang, "Power of deep learning for channel estimation and signal detection in ofdm systems," *IEEE Wireless Communications Letters*, vol. 7, no. 1, pp. 114–117, 2017.

[16] C.-K. Wen, W.-T. Shih, and S. Jin, "Deep learning for massive mimo csi feedback," *IEEE Wireless Communications Letters*, vol. 7, no. 5, pp. 748–751, 2018.

[17] M. Soltani, V. Pourahmadi, A. Mirzaei, and H. Sheikhzadeh, "Deep learning-based channel estimation," *IEEE Communications Letters*, vol. 23, no. 4, pp. 652–655, 2019.

[18] T. J. O'Shea, T. Erpek, and T. C. Clancy, "Deep learning based mimo communications," *arXiv preprint arXiv:1707.07980*, 2017.

[19] T. Diskin, N. Samuel, and A. Wiesel, "Deep mimo detection," in *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2017, pp. 1–5.

[20] H. He, C.-K. Wen, S. Jin, and G. Y. Li, "Model-driven deep learning for mimo detection," *IEEE Transactions on Signal Processing*, vol. 68, pp. 1702–1715, 2020.

[21] M. Khani, M. Alizadeh, J. Hoydis, and P. Fleming, "Adaptive neural signal detection for massive mimo," *IEEE Transactions on Wireless Communications*, vol. 19, no. 8, pp. 5635–5648, 2020.

[22] K. Pratik, B. D. Rao, and M. Welling, "Re-mimo: Recurrent and permutation equivariant neural mimo detection," *IEEE Transactions on Signal Processing*, vol. 69, pp. 459–473, 2020.

This article has been accepted for publication in IEEE Transactions on Wireless Communications. This is the author's version which has not been fully edited and content may change prior to final publication. Citation information: DOI 10.1109/TWC.2024.3439648

[23] C. Stöckle, J. Munir, A. Mezghani, and J. A. Nossek, "Channel estimation in massive mimo systems using 1-bit quantization," in *2016 IEEE 17th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*. IEEE, 2016, pp. 1–6.

[24] C.-L. Liu and P. Vaidyanathan, "One-bit sparse array doa estimation," in *2017 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2017, pp. 3126–3130.

[25] A. Sant and B. D. Rao, "Doa estimation in systems with nonlinearities for mmwave communications," in *ICASSP 2020-2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2020, pp. 4537–4541.

[26] J. Bussgang, "Crosscorrelation functions of amplitude-distorted gaussian signals," *MIT Res. Lab. Elec. Tech. Rep.*, vol. 216, pp. 1–14, 1952.

[27] A. Mezghani, M.-S. Khoufi, and J. A. Nossek, "A modified mmse receiver for quantized mimo systems," *Proc. ITG/IEEE WSA, Vienna, Austria*, pp. 1–5, 2007.

[28] D. K. Ho and B. D. Rao, "Antithetic dithered 1-bit massive mimo architecture: Efficient channel estimation via parameter expansion and pml," *IEEE Transactions on Signal Processing*, vol. 67, no. 9, pp. 2291–2303, 2019.

[29] Q. Wan, J. Fang, H. Duan, Z. Chen, and H. Li, "Generalized bussgang lmmse channel estimation for one-bit massive mimo systems," *IEEE Transactions on Wireless Communications*, vol. 19, no. 6, pp. 4234–4246, 2020.

[30] S. Jacobsson, G. Durisi, M. Coldrey, U. Gustavsson, and C. Studer, "One-bit massive mimo: Channel estimation and high-order modulations," in *2015 IEEE International Conference on Communication Workshop (ICCW)*. IEEE, 2015, pp. 1304–1309.

[31] C. Mollen, J. Choi, E. G. Larsson, and R. W. Heath, "Uplink performance of wideband massive mimo with one-bit adcs," *IEEE Transactions on Wireless Communications*, vol. 16, no. 1, pp. 87–100, 2016.

[32] Y. Li, C. Tao, G. Seco-Granados, A. Mezghani, A. L. Swindlehurst, and L. Liu, "Channel estimation and performance analysis of one-bit massive mimo systems," *IEEE Transactions on Signal Processing*, vol. 65, no. 15, pp. 4075–4089, 2017.

[33] L. V. Nguyen, A. L. Swindlehurst, and D. H. Nguyen, "Svm-based channel estimation and data detection for one-bit massive mimo systems," *IEEE Transactions on Signal Processing*, vol. 69, pp. 2086–2099, 2021.

[34] S. S. Thoota and C. R. Murthy, "Variational bayes' joint channel estimation and soft symbol decoding for uplink massive mimo systems with low resolution adcs," *IEEE Transactions on Communications*, vol. 69, no. 5, pp. 3467–3481, 2021.

[35] J. Choi, D. J. Love, D. R. Brown, and M. Boutin, "Quantized distributed reception for mimo wireless systems using spatial multiplexing," *IEEE Transactions on Signal Processing*, vol. 63, no. 13, pp. 3537–3548, 2015.

[36] J. Choi, J. Mo, and R. W. Heath, "Near maximum-likelihood detector and channel estimator for uplink multiuser massive mimo systems with one-bit adcs," *IEEE Transactions on Communications*, vol. 64, no. 5, pp. 2005–2018, 2016.

[37] Y.-S. Jeon, N. Lee, S.-N. Hong, and R. W. Heath, "One-bit sphere decoding for uplink massive mimo systems with one-bit adcs," *IEEE Transactions on Wireless Communications*, vol. 17, no. 7, pp. 4509–4521, 2018.

[38] Y.-S. Jeon, N. Lee, and H. V. Poor, "Robust data detection for mimo systems with one-bit adcs: A reinforcement learning approach," *IEEE Transactions on Wireless Communications*, vol. 19, no. 3, pp. 1663–1676, 2019.

[39] D. Ho, "Channel estimation and data detection methods for 1-bit massive mimo systems," Ph.D. dissertation, University of California San Diego, 2022.

[40] L. V. Nguyen, A. L. Swindlehurst, and D. H. Nguyen, "Linear and deep neural network-based receivers for massive mimo systems with one-bit adcs," *IEEE Transactions on Wireless Communications*, vol. 20, no. 11, pp. 7333–7345, 2021.

[41] S. Khobahi, N. Shlezinger, M. Soltanalian, and Y. C. Eldar, "Lord-net: Unfolded deep detection network with low-resolution receivers," *IEEE Transactions on Signal Processing*, vol. 69, pp. 5651–5664, 2021.

[42] E. Balevi and J. G. Andrews, "One-bit ofdm receivers via deep learning," *IEEE Transactions on Communications*, vol. 67, no. 6, pp. 4326–4336, 2019.

[43] S. Kim, M. So, N. Lee, and S. Hong, "Semi-supervised learning detector for mu-mimo systems with one-bit adcs," in *2019 IEEE International Conference on Communications Workshops (ICC Workshops)*. IEEE, 2019, pp. 1–6.

[44] S. Kim, J. Chae, and S.-N. Hong, "Machine learning detectors for mu-mimo systems with one-bit adcs," *IEEE Access*, vol. 8, pp. 86 608–86 616, 2020.

[45] Y.-S. Jeon, D. Kim, S.-N. Hong, N. Lee, and R. W. Heath, "Artificial intelligence for physical-layer design of mimo communications with one-bit adcs," *IEEE Communications Magazine*, 2022.

[46] A. Sant and B. D. Rao, "Regularized neural detection for one-bit massive mimo communication systems," *arXiv e-prints*, pp. arXiv–2305, 2023.

[47] ——, "Regularized neural detection for millimeter wave massive mimo communication systems with one-bit adcs," in *ICASSP 2023-2023 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, 2023, pp. 1–5.

[48] D. Tse and P. Viswanath, *Fundamentals of wireless communication*. Cambridge university press, 2005.

[49] P. Barsocchi, "Channel models for terrestrial wireless communications: a survey," *CNR-ISTI technical report*, vol. 83, 2006.

[50] Y. Nesterov, *Introductory lectures on convex optimization: A basic course*. Springer Science & Business Media, 2003, vol. 87.

[51] Y. E. Nesterov, "A method of solving a convex programming problem with convergence rate o\bigl(k^2\bigr)," in *Doklady Akademii Nauk*, vol. 269, no. 3. Russian Academy of Sciences, 1983, pp. 543–547.

[52] A. Beck and M. Teboulle, "A fast iterative shrinkage-thresholding algorithm for linear inverse problems," *SIAM journal on imaging sciences*, vol. 2, no. 1, pp. 183–202, 2009.

[53] A. Sant, "Towards model-based synergistic learning for robust next-generation mimo systems," Ph.D. dissertation, UC San Diego, 2024.

[54] S. R. Bowling, M. T. Khasawneh, S. Kaewkuekool, and B. R. Cho, "A logistic approximation to the cumulative normal distribution," *Journal of Industrial Engineering and Management*, vol. 2, no. 1, pp. 114–127, 2009.

[55] A. Balatsoukas-Stimming and C. Studer, "Deep unfolding for communications systems: A survey and some new directions," in *2019 IEEE International Workshop on Signal Processing Systems (SiPS)*. IEEE, 2019, pp. 266–271.

[56] V. Monga, Y. Li, and Y. C. Eldar, "Algorithm unrolling: Interpretable, efficient deep learning for signal and image processing," *IEEE Signal Processing Magazine*, vol. 38, no. 2, pp. 18–44, 2021.

[57] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2014.