This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

Adaptive Neural Signal Detection for Massive MIMO

Mehrdad Khani, Mohammad Alizadeh,
Jakob Hoydis, *Senior Member, IEEE*, Phil Fleming, *Senior Member, IEEE*

***Abstract*—Traditional symbol detection algorithms either perform poorly or are impractical to implement for Massive Multiple-Input Multiple-Output (MIMO) systems. Recently, several learning-based approaches have achieved promising results on simple channel models (e.g., i.i.d. Gaussian channel coefficients), but as we show, their performance degrades on real-world channels with spatial correlation. We propose MMNet, a deep learning MIMO detection scheme that significantly outperforms existing approaches on realistic channels with the same or lower computational complexity. MMNet's design builds on the theory of iterative soft-thresholding algorithms, and uses a novel training algorithm that leverages temporal and spectral correlation in real channels to accelerate training. These innovations make it practical to train MMNet online for every realization of the channel. On i.i.d. Gaussian channels, MMNet requires two orders of magnitude fewer operations than existing deep learning schemes but achieves near-optimal performance. On spatially-correlated channels, it achieves the same error rate as the next-best learning scheme (OAMPNet) at 2.5dB lower signal-to-noise ratio (SNR), and with at least 10$\times$ less computational complexity. MMNet is also 4–8dB better overall than a classic linear scheme like the minimum mean square error (MMSE) detector.**

***Index Terms*—Massive MIMO, signal detection, deep learning, online adaptation, spatial channel correlation**

I. Introduction

The fifth generation of cellular communication systems (5G) promises an order of magnitude higher spectral efficiency (measured in bits/s/Hz) than legacy standards such as Long Term Evolution (LTE) [1]. One of the key enablers of this better efficiency is Massive Multiple-Input Multiple-Output (MIMO) [2], in which a base station (BS) equipped with a very large number of antennas (around 64–256) simultaneously serves multiple single-antenna user equipments (UEs) on the same time-frequency resource.

Legacy systems already use MIMO [3], but this is the first time it will be deployed on such a large scale, creating significant challenges for *signal detection*. The goal of signal detection is to infer the transmitted signal vector $\mathbf{x}$ from the vector $\mathbf{y} = \mathbf{H}\mathbf{x} + \mathbf{n}$ received at the BS antennas, where $\mathbf{H}$ is the channel matrix and $\mathbf{n}$ is Gaussian noise. Traditional MIMO signal detection schemes with strong performance [4, 5, 6, 7] are feasible only for small systems and have prohibitive complexity for massive MIMO deployments. Thus, there is a need for low-complexity signal detection schemes that can both perform well and scale to large system dimensions.

M. Khani and M. Alizadeh are with the Computer Science & Artificial Intelligence Laboratory, MIT, Cambridge, MA 02139, USA (khani@mit.edu, alizadeh@csail.mit.edu).

J. Hoydis is with Nokia Bell Labs, Paris-Saclay, 91620 Nozay, France (jakob.hoydis@nokia-bell-labs.com).

P. Fleming is with Ann Arbor Analytics, LLC, Ann Arbor, MI, USA (phil.fleming@annarbor-analytics.com).

In recent work, researchers have proposed several learning approaches for MIMO signal detection. Samuel et al. [8, 9] achieved impressive results with a deep neural network architecture called DetNet, e.g., matching the performance of a semidefinite relaxation (SDR) baseline for independent and identically distributed (i.i.d.) Gaussian channel matrices while running 30$\times$ faster. He et al. [10] introduced OAMPNet, a model inspired by the Orthogonal AMP algorithm [11], and demonstrated strong performance on both i.i.d. Gaussian and small-sized correlated channel matrices based on the Kronecker model [12]. DetNet and OAMPNet are both trained offline: they try to learn a single detector during training for a family of channel matrices (e.g., i.i.d. Gaussian channels).

In this paper we show that neither approach is effective in practice. We conduct extensive experiments using a dataset of channel realizations from the 3GPP 3D MIMO channel [13], as implemented in the QuaDRiGa channel simulator [14]. Our results show that DetNet's training is unstable for realistic channels, while OAMPNet suffers a large performance gap (4–7dB at symbol error rate of 10^{-3}) compared to the optimal Maximum-Likelihood detector on these channels. Both models (as well as several classical baselines) perform well in simpler settings used for evaluation in prior work (e.g., i.i.d. Gaussian channels, low-order modulation schemes). Our results demonstrate the difficulty of learning a single detector that generalizes across a wide range of realistic channel matrices (esp. poorly-conditioned channels that are difficult to invert).

Motivated by these findings, we revisit MIMO detection from an online learning perspective. We ask: *Can a receiver optimize its detector for every realization of the channel matrix?* Intuitively, such an approach could perform better than using a fixed detector for a wide variety of channel matrices. However, conventional wisdom suggests that training a MIMO detector online is impossible because of the stringent performance requirements [8].

Our design, MMNet, overcomes this challenge with two key ideas. First, it uses a neural network architecture that strikes a balance between flexibility and complexity. Prior neural network architectures for MIMO detection are poorly suited to online training. DetNet is a large model with 1-10 million parameters depending on the system size and modulation scheme, making it prohibitively expensive to train online. OAMPNet, on the other hand, is very restrictive, adding only 2 trainable parameters per iteration to the OAMP algorithm. Since the OAMP algorithm requires strong assumptions about channel matrices (unitarily-invariant channels [11]), OAMPNet, even with online training, performs poorly on channels that deviate from the assumptions.

MMNet's neural network is based on iterative soft-

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

thresholding algorithms, a popular class of solutions to "linear-inverse" problems [15, 16, 17] like MIMO detection. These algorithms repeatedly refine an estimate of the signal by alternating between a linear detector and a non-linear denoising step. By preserving the core components of these algorithms in MIMO detection, such as a simple denoiser that is optimal for uncorrelated Gaussian noise, MMNet avoids the pitfalls of overly general neural network architectures like DetNet. At the same time, unlike OAMPNet, MMNet provides adequate flexibility in the architecture through trainable parameters that can be optimized for each channel realization.

MMNet's second key idea is an online training algorithm that exploits the locality of channel matrices at a receiver in both the frequency and time domains. By leveraging spectral and temporal locality, MMNet accelerates training by more than two orders of magnitude compared to naively retraining the neural network from scratch for each channel realization.

Taken together, these ideas enable MMNet to achieve performance within ∼2dB of the optimal Maximum-Likelihood detector with 10-15× less computational complexity than the second-best scheme, OAMPNet. On random i.i.d. Gaussian channels, an even simpler version of MMNet (MMNet-iid) with 100× less complexity than OAMPNet and DetNet achieves near-optimal performance without any retraining.

We empirically analyze the dynamics of errors across different layers of MMNet and OAMPNet to understand how MMNet achieves higher detection accuracy. Our analysis reveals that MMNet "shapes" the distribution of noise at the input of the denoisers to ensure they operate effectively. In particular, as signals propagate through the MMNet neural network, the noise distribution at the input of the denoiser stages approaches a Gaussian distribution, create precisely the conditions in which the denoisers can attenuate noise maximally.

The rest of this paper is organized as follows. Section II provides background on classical and learning-based detection schemes, and introduces a general iterative framework that can express many of these algorithms. Section IV introduces the MMNet design in addition to a simple variant for i.i.d. channels. Section V shows performance results of detection algorithms on i.i.d. Gaussian and 3GPP MIMO channels for different modulations. Section VI discusses the error dynamics of MMNet and empirically studies why it performs better than OAMPNet. Section VII introduces MMNet's online training algorithm.

The code to reproduce our results is available at https://github.com/mehrdadkhani/MMNet.

II. Background

A. Notation

We will use lowercase symbols for scalars, bold lowercase symbols for column vectors and bold uppercase symbols to denote matrices. Symbols $\{\theta, \boldsymbol{\theta}, \boldsymbol{\Theta}\}$ are used to represent the parameters of trainable models. The transpose and pseudo-inverse of matrix $\boldsymbol{A}$ are denoted by $\boldsymbol{A}^H$ and $\boldsymbol{A}^+ = (\boldsymbol{A}^H\boldsymbol{A})^{-1}\boldsymbol{A}^H$ respectively. $\mathbf{I}_n$ stands for identity matrix of size n.

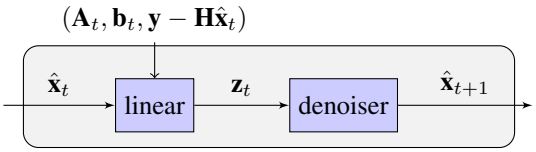


Fig. 1: A block of an iterative detector in our general framework. Each block contains a linear transformation followed by a denoising stage.

B. The MIMO Signal Detection Problem

We consider a communication channel from N_t single-antenna transmitters to a receiver equipped with N_r antennas. The received vector $\mathbf{y} \in \mathbb{C}^{N_r}$ is given as

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

where $\mathbf{H} \in \mathbb{C}^{N_r \times N_t}$ is the channel matrix, $\mathbf{n} \sim \mathcal{CN}(0, \sigma^2\mathbf{I}_{N_r})$ is complex Gaussian noise, and $\mathbf{x} \in \mathcal{X}^{N_t}$ is the vector of transmitted symbols. $\mathcal{X}$ denotes the finite set of constellation points. We assume that each transmitter chooses a symbol from $\mathcal{X}$ uniformly at random, and all transmitters use the same constellation set. Further, as is standard practice, we assume that the constellation set $\mathcal{X}$ is given by a quadrature amplitude modulation (QAM) scheme [18]. All constellations are normalized to unit average power (e.g., the QAM4 constellation is $\{\pm\frac{1}{\sqrt{2}} \pm j\frac{1}{\sqrt{2}}\}$).

The channel matrix $\mathbf{H}$ is assumed to be known at the receiver. The goal of the receiver is to compute the maximum likelihood (ML) estimate $\hat{\mathbf{x}}$ of $\mathbf{x}$:

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{x} \in \mathcal{X}^{N_t}} ||\mathbf{y} - \mathbf{H}\mathbf{x}||_2. \tag{2}$$

The optimization problem in (2) is NP-hard due to the finite-alphabet constraint $\mathbf{x} \in \mathcal{X}^{N_t}$ [19]. Therefore, over the last three decades, researchers have proposed a variety of detectors with differing levels of complexity. We refer the interested reader to [5, 6] for a comprehensive overview of MIMO detection schemes.

C. An iterative framework for MIMO detection

We focus on a class of iterative estimation algorithms for solving (2) shown in Fig. 1. Each iteration of these algorithms comprises the following two steps:

$$\text{General Iteration:} \quad \begin{aligned} \mathbf{z}_t &= \hat{\mathbf{x}}_t + \mathbf{A}_t(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}_t) + \mathbf{b}_t \\ \hat{\mathbf{x}}_{t+1} &= \eta_t\left(\mathbf{z}_t\right). \end{aligned} \tag{3}$$

The first step takes as input $\hat{\mathbf{x}}_t$, a current estimate of $\mathbf{x}$, and the *residual* error $\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}_t$, and applies a *linear* transformation to obtain an intermediate signal $\mathbf{z}_t$. In the second step, a *non-linear* "denoiser" is applied to $\mathbf{z}_t$ to produce $\hat{\mathbf{x}}_{t+1}$, a new estimate of $\mathbf{x}$ that is used as the input for the next iteration. Together, the linear and denoising steps aim to improve the quality of the estimate $\hat{\mathbf{x}}_t$ from one iteration to the next. In this paper we use the terms *iteration*, *layer*, and *block* interchangeably to refer to one complete iteration (the linear step followed by the non-linear denoiser). All algorithms discussed assume $\hat{\mathbf{x}}_0 = 0$.

The denoiser is a non-linear function $\eta_t \colon \mathbb{C}^{N_t} \to \mathbb{C}^{N_t}$ in general, however, most algorithms apply the same denoising

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

function $\beta_t : \mathbb{C} \to \mathbb{C}$ to each element of $\mathbf{z}_t$. A natural choice for the denoising function is the minimizer of $\mathbb{E}[\|\hat{\mathbf{x}} - \mathbf{x}\|_2 | \mathbf{z}_t]$, which is given by:

$$\eta_t(\mathbf{z}_t) = \mathbb{E}[\mathbf{x}|\mathbf{z}_t]. \tag{4}$$

Optimal denoiser for Gaussian noise: Several existing MIMO detection schemes assume that the noise at the input of the denoiser $\mathbf{z}_t - \mathbf{x}$ has an i.i.d. Gaussian distribution with diagonal covariance matrix $\sigma_t^2 \mathbf{I}_{N_t}$. In this case, the optimal element-wise denoising function derived from (4) has the form

$$\beta_t(z; \sigma_t^2) = \frac{1}{Z} \sum_{x_i \in \mathcal{X}} x_i \exp\left(-\frac{\|z - x_i\|^2}{\sigma_t^2}\right), \tag{5}$$

where $Z = \sum_{x_j \in \mathcal{X}} \exp\left(-\frac{\|z-x_j\|^2}{\sigma_t^2}\right)$. The standard deviation of input noise at the denoisers, σ_t, generally varies from iteration to iteration, and depends on the linear steps in each iteration. Different algorithms use different methods to estimate σ_t. In the rest of this paper, $\eta_t(\cdot; \sigma_t)$ refers to a denoiser which applies (5) to each element of its input vector.

III. Related Work

In this section we briefly describe several algorithms for MIMO detection. We begin with traditional, non-learning approaches (Section III-A) and then discuss recent deep learning proposals (Section III-B). We show how many of these algorithms can be expressed in the iterative framework discussed above.

A. Classical MIMO detection algorithms

1) Linear: The simplest method to approximately solve (2) is to relax the constraint of $\mathbf{x} \in \mathcal{X}^{N_t}$ to $\mathbf{x} \in \mathbb{C}^{N_t}$ and then round the relaxed solution to the closest point on the constellation:

$$\text{Linear:} \quad \begin{aligned} \mathbf{z} &= \arg\min_{\mathbf{x} \in \mathbb{C}^{N_t}} \|\mathbf{y} - \mathbf{H}\mathbf{x}\|_2 = \mathbf{H}^{+}\mathbf{y} \\ \hat{\mathbf{x}} &= \arg\min_{\mathbf{x} \in \mathcal{X}^{N_t}} \|\mathbf{x} - \mathbf{z}\|_2. \end{aligned} \tag{6}$$

Rounding each component of $\mathbf{z}$ to the closest point in the constellation set $\hat{\mathbf{x}}$ leads to the well-known zero-forcing (ZF) detector, which is equivalent to a single step of (3) with initial condition of $\hat{\mathbf{x}}_0 = 0$, $\mathbf{A}_0 = \mathbf{H}^{+}$, $\mathbf{b}_0 = 0$, and a hard-decision denoiser with respect to the points in the constellation. Other widely-used single-step linear detectors include the matched filter and the minimum mean square error (MMSE) detectors [2] with $\mathbf{A}_0 = \mathbf{H}^H$ and $\mathbf{A}_0 = (\mathbf{H}^H\mathbf{H} + \sigma^2 \mathbf{I}_{N_t})^{-1}\mathbf{H}^H$, respectively. Linear detectors are attractive for practical implementation because of their low complexity, but they perform substantially worse than the optimal detector.

We can also perform the optimization in (6) in multiple iterations using gradient descent. The gradient of the objective function in the first equation of (6) with respect to $\mathbf{x}$ is $-2\mathbf{H}^H(\mathbf{y} - \mathbf{H}\mathbf{x})$. Hence, if we set $\mathbf{A}_t$ to $2\alpha\mathbf{H}^H$ and $\mathbf{b}_t = 0$, the linear step of (3) is equivalent to minimizing $||\mathbf{y} - \mathbf{H}\mathbf{x}||_2$ using gradient descent with step size α. This is followed by a projection onto the constellation set in the denoising step. If we had a compact convex constellation set, this projected gradient descent procedure is guaranteed to converge to the global optimum. Discrete constellation sets, however, are not compact convex. Nonetheless, solving the linear least squares problem in (6) iteratively may be desirable to avoid the cost of computing the pseudo-inverse of the channel matrix.

2) Approximate Message Passing (AMP): MIMO detection can, in principle, be solved through belief propagation (BP) if we consider a bipartite graph representation of the model in (1) [20]. BP on this graph requires $\mathcal{O}(N_r N_t)$ update messages in each iteration, which would be prohibitive for large system dimensions. In the large system limit, Jeon et al. [17] introduce approximate message passing (AMP) as a lower complexity inference algorithm for solving (2) for i.i.d. Gaussian channels. AMP reduces the number of messages in each iteration to $\mathcal{O}(N_r + N_t)$. The algorithm performs the following sequence of updates:

$$\text{AMP:} \quad \begin{aligned} \mathbf{z}_t &= \hat{\mathbf{x}}_t + \mathbf{H}^H(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}_t) + \mathbf{b}_t \\ \mathbf{b}_t &= \alpha_t \left(\mathbf{H}^H(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}_{t-1}) + \mathbf{b}_{t-1}\right) \\ \hat{\mathbf{x}}_{t+1} &= \eta_t\left(\mathbf{z}_t; \sigma_t\right). \end{aligned} \tag{7}$$

To express AMP in our iterative framework, use $\mathbf{A}_t = \mathbf{H}^H$ as the linear operator. The vector $\mathbf{b}_t$ is known as the *Onsager* term. The scalar sequences σ_t and α_t are computed analytically given the signal-to-noise ratio (SNR) and system parameters (constellation, number of transmitters and receivers); see [21] for details. The denoising function $\eta_t(\cdot; \sigma_t)$ applies the optimal denoiser for Gaussian noise in (5) to each element of the vector $\mathbf{z}_t$. Jeon et al. [17] prove that AMP is asymptotically optimal for large i.i.d. Gaussian channel matrices.

Orthogonal AMP (OAMP) [11] is another scheme proposed to relax the i.i.d. Gaussian channel assumption in the original AMP algorithm. OAMP assumes unitarily-invariant channel matrices [22] and operates as follows:

$$\text{OAMP:} \quad \begin{aligned} \mathbf{z}_t &= \hat{\mathbf{x}}_t + \gamma_t \mathbf{H}^H \left(v_t^2 \mathbf{H}\mathbf{H}^H + \sigma^2 \mathbf{I}\right)^{-1} (\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}_t) \\ \hat{\mathbf{x}}_{t+1} &= \eta_t\left(\mathbf{z}_t; \sigma_t^2\right) \end{aligned} \tag{8}$$

where $\gamma_t = N_t / \text{trace}\left(v_t^2 \mathbf{H}^H \left(v_t^2 \mathbf{H}\mathbf{H}^H + \sigma^2 \mathbf{I}\right)^{-1} \mathbf{H}\right)$ is a normalizing factor and v_t^2 is proportional to the average noise power at the denoiser output at iteration t and can be computed given the SNR and system dimensions [11]. Notice that OAMP requires a matrix inverse operation in each iteration, making it more expensive computationally than AMP.

3) Other techniques: Several detection schemes relax the lattice constraint ($\mathbf{x} \in \mathcal{X}^{N_t}$) in (2). For example, Semi-Definite Relaxation (SDR) [7] formulates the problem as a semi-definite program. Sphere decoding [4] conducts a search over solutions $\hat{\mathbf{x}}$ such that $||\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}||_2 \leq r$. Increasing r covers a larger set of possible solutions, but this comes at the cost of increased complexity, approaching that of brute-force search. There is a large body of work on improvements to this idea which can be found in [5, 6]. While these approaches can perform well, their computational complexity is prohibitive for Massive MIMO systems with currently available hardware.

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

Another class of detector applies several stages of linear detection followed by interference subtraction from the observation $\mathbf{y}$. The V-BLAST scheme [23] does this by detecting the strongest symbols, which are then successively removed from $\mathbf{y}$. The drawbacks of this approach are error propagation of early symbol decisions and high complexity due to the N_t required stages, as well as the necessary reordering of transmitters after each step. Parallel interference cancellation (PIC) has been proposed to circumvent these problems. PIC jointly detects all transmitted symbols and then attempts to create an interference-free channel for each transmitter through the cancellation of all other transmitted symbols [24, 25]. A large system approximation of this approach was recently developed in [26] based on [27]. However, it is currently limited to binary phase shift keying (BPSK) modulation and leads to unsatisfactory performance for realistic system dimensions.

In summary, most existing techniques are too complex to be implemented at the scale required by next-generation Massive MIMO systems. On the other hand, light-weight techniques like AMP cannot handle correlated channel matrices. These limitations have motivated a number of learning-based proposals for MIMO detection, which we discuss next.

B. Learning-based MIMO detection schemes

1) DetNet: Inspired by iterative projected gradient descent optimization, Samuel et al. [8, 9] propose DetNet, a deep neural network architecture for MIMO detection. This architecture performs very well in case of i.i.d. complex Gaussian channel matrices and achieves the performance of state-of-the-art algorithms for lower-order modulation schemes, such as BPSK and QAM4. However, it is far more complex. The neural network is described by the following equations:

$$\text{DetNet:} \quad \begin{aligned} \mathbf{q}_t &= \hat{\mathbf{x}}_{t-1} - \theta_t^{(1)}\mathbf{H}^H\mathbf{y} + \theta_t^{(2)}\mathbf{H}^H\mathbf{H}\hat{\mathbf{x}}_{t-1} \\ \mathbf{u}_t &= \left[\boldsymbol{\Theta}_t^{(3)}\mathbf{q}_t + \boldsymbol{\Theta}_t^{(4)}\mathbf{v}_{t-1} + \boldsymbol{\theta}_t^{(5)}\right]_+ \\ \mathbf{v}_t &= \boldsymbol{\Theta}_t^{(6)}\mathbf{u}_t + \boldsymbol{\theta}_t^{(7)} \\ \hat{\mathbf{x}}_t &= \boldsymbol{\Theta}_t^{(8)}\mathbf{u}_t + \boldsymbol{\theta}_t^{(9)} \end{aligned} \tag{9}$$

where $[x]_+ = \max(x, 0)$, which is also known as ReLU activation function [28], is applied element-wise [8].

Although DetNet's performance is promising, it has two main limitations. First, its heuristic nature makes it difficult to reason about how the neural network works, and how to extend its architecture, for example, to support spatially correlated channel matrices. Second, DetNet's architecture does not incorporate known properties of iterative methods and is thus unnecessarily complex. For example, similar to other iterative schemes, DetNet's neural network can be viewed to be performing a linear transformation followed by a non-linear projection in each iteration. DetNet's linear step computes $\boldsymbol{q}_t$, and the non-linear projection computes $\hat{\mathbf{x}}_t$ from $\boldsymbol{q}_t$.[1] However, unlike other iterative schemes that use the simple non-linear denoiser in (5), DetNet's non-linear projection is a fully-connected 2-layer neural network that operates on an N_t-dimensional input vector. In fact, DetNet uses parameter matrices $\boldsymbol{\Theta}_t^{(3)}$ and $\boldsymbol{\Theta}_t^{(4)}$ to map the input of the projection to an even larger space before mapping it back to a N_t-dimensional vector.[2].

2) OAMPNet: He et al. [10] designed a learning-based iterative scheme based on the OAMP algorithm. OAMPNet adds two tuning parameters per iteration to the OAMP algorithm, as follows:

$$\text{OAMPNet:} \quad \begin{aligned} \mathbf{z}_t &= \hat{\mathbf{x}}_t + \theta_t^{(1)}\mathbf{H}^H\left(v_t^2\mathbf{H}\mathbf{H}^H + \sigma^2\mathbf{I}\right)^{-1}(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}_t) \\ \hat{\mathbf{x}}_{t+1} &= \eta_t\left(\mathbf{z}_t; \sigma_t^2\right). \end{aligned} \tag{10}$$

In the above equations, OAMPNet uses the second trainable parameter , $\theta_t^{(2)}$, in order to balance the estimate of denoisers input noise variance σ_t^2. OAMPNet uses the same denoisers used by AMP.

OAMPNet shows very good performance in the case of i.i.d. Gaussian channels, but it does not generalize to realistic channels with spatial correlations, as our experiments in Section V show. The reason is that OAMPNet bases its architecture on OAMP, which requires a strict assumption about the system: unitarily-invariant channel matrices. Therefore, its performance degrades on realistic channel matrices that do not conform to this assumption. Further, like OAMP, OAMPNet must compute a matrix pseudo-inverse in each iteration and, therefore, its complexity is still quite high compared to schemes like AMP.

IV. MMNet Design

MMNet is a neural-network-based signal detection scheme inspired by the iterative framework described in Section II-C. Unlike prior approaches that use a single model for all channel matrices, MMNet is designed to be trained online for each realization of $\mathbf{H}$. In this approach, the receiver continually adapts its parameters as it observes new channel matrices. We demonstrate that online training is feasible in practice with a suitable neural network architecture by exploiting the fact that realistic channels exhibit locality in both the frequency and time domains. We introduce the neural network architecture in this section and discuss the training algorithm in Section VII.

The main idea behind MMNet's architecture is to strike a balance between flexibility and complexity in the linear and denoising components of each layer of the neural network. In the following, we present two different neural network architectures for (1) i.i.d. Gaussian and (2) arbitrary channels.

i.i.d. Gaussian channels: In the i.i.d. Gaussian case, the model is extremely simple:

$$\text{MMNet-iid:} \quad \begin{aligned} \mathbf{z}_t &= \hat{\mathbf{x}}_t + \theta_t^{(1)}\mathbf{H}^H(\mathbf{y} - \mathbf{H}\hat{\mathbf{x}}_t) \\ \hat{\mathbf{x}}_{t+1} &= \eta_t\left(\mathbf{z}_t; \sigma_t^2\right). \end{aligned} \tag{11}$$

Here, the denoiser is the optimal denoiser for Gaussian noise given in (5). MMNet-iid assumes the same distribution of

[1]The role of $\mathbf{v}_t$ in DetNet is unclear, and is not explained in [8].

[2]In the specific networks evaluated in [8], the $\mathbf{u}_t$ vector has up to $6N_t$ dimensions depending on the constellation.

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

noise at the input of the denoiser for all transmitted symbols and estimates its variance σ_t^2 according to

$$\sigma_t^2 = \frac{\theta_t^{(2)}}{N_t}\left(\frac{\|\mathbf{I}-\mathbf{A}_t\mathbf{H}\|_F^2}{\|\mathbf{H}\|_F^2}\left[\|\mathbf{y}-\mathbf{H}\hat{\mathbf{x}}_t\|_2^2 - N_r\sigma^2\right]_+ + \frac{\|\mathbf{A}_t\|_F^2}{\|\mathbf{H}\|_F^2}\sigma^2\right), \tag{12}$$

where $\mathbf{A}_t = \theta_t^{(1)}\mathbf{H}^H$. The intuition behind (12) is that the noise at the input of the denoiser at step t is comprised of two parts: (i) the residual error caused by deviation of $\hat{\mathbf{x}}_t$ from the true value of $\mathbf{x}$, and (ii) the contribution of the channel noise $\mathbf{n}$. The first component is amplified by the linear transformation $(\mathbf{I}-\mathbf{A}_t\mathbf{H})$, and the second component is amplified by $\mathbf{A}_t$. See [11, 17] for further details on this method for estimating noise variance.

This model has only two parameters per layer: $\theta_t^{(1)}$ and $\theta_t^{(2)}$. We discuss this model merely to illustrate that, for the i.i.d. Gaussian channel matrix case (which most prior work focused on for evaluation), a simple model that adds a small amount of flexibility to existing algorithms like AMP can perform very well. In fact, our results will show that, in this case, we do not even need to train the parameters of the model online for each channel realization; training offline over randomly sampled i.i.d. Gaussian channel suffices.

Arbitrary channels: The MMNet neural network for arbitrary channel matrices is as follows:

$$\text{MMNet:} \qquad \begin{aligned} \mathbf{z}_t &= \hat{\mathbf{x}}_t + \mathbf{\Theta}_t^{(1)}(\mathbf{y}-\mathbf{H}\hat{\mathbf{x}}_t) \\ \hat{\mathbf{x}}_{t+1} &= \eta_t\left(\mathbf{z}_t;\boldsymbol{\sigma}_t^2\right) \end{aligned} \tag{13}$$

where $\mathbf{A}_t = \mathbf{\Theta}_t^{(1)}$ is an $N_t \times N_r$ complex-valued trainable matrix. In order to enable the model to handle cases in which different transmitted symbols have different levels of noise, we estimate the noise variance at the input of the denoiser corresponding to each transmitted signal as:

$$\boldsymbol{\sigma}_t^2 = \frac{\boldsymbol{\theta}_t^{(2)}}{N_t}\left(\frac{\|\mathbf{I}-\mathbf{A}_t\mathbf{H}\|_F^2}{\|\mathbf{H}\|_F^2}\left[\|\mathbf{y}-\mathbf{H}\hat{\mathbf{x}}_t\|_2^2 - N_r\sigma^2\right]_+ + \frac{\|\mathbf{A}_t\|_F^2}{\|\mathbf{H}\|_F^2}\sigma^2\right), \tag{14}$$

where the parameter vector $\boldsymbol{\theta}_t^{(2)}$ of size $N_t \times 1$ scales the noise variance by different amounts for each symbol.

MMNet concatenates T layers of the above form. We use the average L2-loss over all T layers in order to train the model, which is given by

$$L = \frac{1}{T}\sum_{t=1}^{T}\|\hat{\mathbf{x}}_t - \mathbf{x}\|_2^2. \tag{15}$$

MMNet's architecture provides many more degrees of freedom than the highly-constrained OAMPNet, but it is also much simpler than DetNet. In particular, MMNet uses a flexible linear transformation (which does not need to be linear in $\mathbf{H}$) to construct the intermediate signal z_t, but it applies the standard optimal denoiser for Gaussian noise in (5). Further, unlike OAMPNet, MMNet does not require any matrix inverse operations.

V. Experiments

In this section, we evaluate and compare the performance of MMNet with state-of-the-art schemes for both i.i.d. Gaussian and realistic channel matrices. These are our main findings:

1) On i.i.d. Gaussian channels, most schemes perform very well. MMSE, SDR, V-BLAST and DetNet are 1–3dB worse than the Maximum-Likelihood solution. AMP performance degrades for higher-order modulations in high SNRs. MMNet-iid and OAMPNet are both very close to Maximum-Likelihood in all experiment on these channels. MMNet-iid, however, has two orders of magnitude lower complexity than other learning-based schemes, OAMPNet and DetNet.
2) On realistic, spatially-correlated channel matrices, the performance of all existing learning-based approaches degrades significantly. MMNet consistently achieves the smallest gap with Maximum-Likelihood. MMSE has a 8–10dB gap with Maximum-Likelihood on 64×16 channels. OAMPNet reduces this gap to 5dB, and MMNet closes the gap further to less than 2.2dB. DetNet and AMP perform poorly on realistic channels; with QAM4, for example, AMP achieves an SER of only 0.36 at 7dB while MMSE can bring down the SER to 0.033 in the same setting. The best SER that we achieve in this setting with DetNet before running into stability problems is 0.045, which is worse than MMSE. Finally, MMNet's performance is robust and degrades more gracefully to channel estimation errors than OAMPNet and MMSE

A. Methodology

We first briefly discuss the details of detection schemes used for comparison. Since some of these schemes (including MMNet) require training, we then discuss the process of generating data and training/testing on this data.

1) Compared Schemes: In our experiments, we compare the following schemes on QAM modulation:

- **MMSE:** Linear decoder that applies the SNR-regularized channel's pseudo inverse and rounds the output to the closest point on the constellation.
- **SDR:** Semidefinite programming using a rank-1 relaxation interior point method [29].
- **V-BLAST:** Multi-stage interference cancellation BLAST algorithm using Zero-Forcing as the detection stage introduced in [25].
- **AMP:** AMP algorithm for MIMO detection from Jeon et al. [17]. AMP runs 50 iterations of the updates described in (7). We verified that adding more iterations does not improve the results.
- **DetNet:** The deep learning approach introduced in [8]. The DetNet paper describes instantiations of the architecture for BPSK, QAM4 and QAM16; these neural networks have, on the order of 1–10M, trainable parameters depending on the size of the system and constellation set.
- **OAMPNet:** The OAMP-based architecture [10] implemented in 10 layers with 2 trainable parameters per layer and an inverse matrix computation at each layer.

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

- **MMNet-iid:** The simple mode described in (11). This scheme has only 2 scalar parameters per layer and does not require any matrix inversions. We implement this neural network with 10 layers.
- **MMNet:** Our design described in (13). It has 10 layers, and the total number of trainable parameters is $2N_t(N_r + 1)$ per layer, independent of constellation size. In the systems evaluated, this results in 20K-41K trainable parameters.
- **Maximum-Likelihood:** The optimal solver for (2) using a highly-optimized Mixed Integer Programming package Gurobi [30].

2) Dataset: Training and test data are generated through the model described in (1). In this model, there are three sources of randomness: the signal $\mathbf{x}$, the channel noise $\mathbf{n}$ and the channel matrix $\mathbf{H}$. Each transmitted signal $\mathbf{x}$ is generated randomly and uniformly over the corresponding constellation set. All transmitters are assumed to use the same modulation. The channel noise $\mathbf{n}$ is sampled from a zero-mean i.i.d. normal distribution with a variance that is set according to the operating SNR, defined as $\mathrm{SNR(dB)} = 10\log\left(\mathbb{E}[\|\mathbf{Hx}\|_2^2]/\mathbb{E}[\|\mathbf{n}\|_2^2]\right)$. For every training batch, the SNR(dB) is chosen uniformly at random in the desired operating SNR interval. This interval depends on the modulation scheme. For each modulation in each experiment, the SNR regime is chosen such that the best scheme other than Maximum-Likelihood can achieve a symbol error rate (SER) of 10^{-3}–10^{-2}.

The channel matrices $\mathbf{H}$ are either sampled from an i.i.d. Gaussian distribution (i.e., each column of $\mathbf{H}$ is a complex-normal $\mathcal{CN}(0, (1/N_r)\mathbf{I}_{N_r})$), or they are generated via the realistic channel simulation described below.

We study two system size ratios (N_t/N_r) of 0.25 and 0.5, with the total number of receivers fixed at $N_r = 64$. These are typical values for 4G/5G base stations in urban cellular deployments. For the case of realistic channels, we generate a dataset of channel realizations from the 3GPP 3D MIMO channel model [13], as implemented in the QuaDRiGa channel simulator [14].[3] We consider a base station (BS) equipped with a rectangular planar array consisting of 32 dual-polarized antennas installed at a height of 25 m. The BS is assumed to cover a 120°-cell sector of radius 500 m within which $N_t \in \{16, 32\}$ single-antenna users are dropped randomly. A guard distance of 10 m from the BS is kept. Each user is then assumed to move along a linear trajectory with a speed of 1 m/s. Channels are sampled every $\lambda/4$ m at a center frequency of 2.53 GHz to obtain sequences of length 100. Each channel realization is then converted to the frequency domain assuming a bandwidth of 20 MHz and using 1024 sub-carriers from which only every fourth is kept, resulting in $F = 256$ effective sub-carriers. We gather a total of 40 user drops, resulting in 40×256 length 100 sequences of channel matrices (i.e., 1M channel matrices in total). Since the path loss can vary dramatically between different users, we assume perfect power control, which normalizes the average received power across antennas and sub-carriers to one. Denote $\mathbf{H}[f, k]$ as the kth column of $\mathbf{H}$ on sub-carrier f. Our normalization ensures that

$$\frac{1}{N_r N_t F}\sum_{k=1}^{N_t}\sum_{f=1}^{F}\|\mathbf{H}[f,k]\|^2 = 1.$$

3) Training: MMNet, DetNet, and OAMPNet require training and were implemented in TensorFlow [31]. We have converted (2) to its equivalent real-valued representation for TensorFlow implementations (cf. [32, Sec. II]). DetNet and OAMPNet are both trained as described in the corresponding publications (i.e., batch size of 5K samples). We trained each of the latter two algorithms for 50K iterations.

To train MMNet, we use the Adam optimizer [33] with a learning rate of 10^{-3}. Each training batch has a size of 500 samples. We train MMNet for 1K iterations on each realization of $\mathbf{H}$ in the naive implementation. In Section VII-B, we exploit frequency and time domain correlations to reduce the training requirement to 9 iterations per channel matrix.

In spatially correlated channels, we perform an additional 5K iterations of training with a batch size of 5K samples for each realization of $\mathbf{H}$ on the pre-trained OAMPNet model, in order to have a fair comparison against MMNet's online training. However, as we found that this extra training does not meaningfully improve the performance of OAMPNet, we do not count this re-training overhead in the complexity of OAMPNet algorithm in Section VII.

For i.i.d. Gaussian channels, MMNet-iid is not trained per channel realization $\mathbf{H}$. Instead, we use 10K iterations with a batch size of 500 samples to train a single MMNet-iid neural network, which we then test on new channel samples.

B. Results

We compare different schemes along two axes: performance and complexity. In this section, we focus on performance, leaving a comparison of complexity to Section VII. We use the SNR required to achieve an SER of 10^{-3} as the primary performance metric. In practice, most error correcting schemes operate around an SER of 10^{-3}–10^{-2}, so this is the relevant regime for MIMO detection.

1) i.i.d. Gaussian channels: Fig. 2 shows the SER vs. SNR of the state-of-the-art MIMO schemes on i.i.d. channels for two system sizes: 32 and 16 transmitters (Fig. 2a and Fig. 2b, respectively), and 64 receivers.

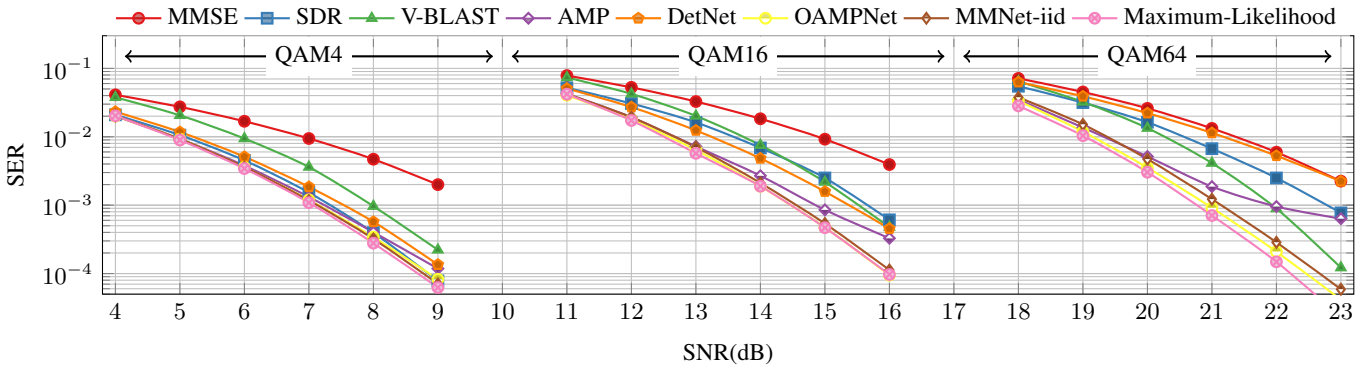
We make the following observations:

1) The SNR required to achieve a certain SER increases by 2–3dB as we double the number of transmitters (notice the different range of x-axes). The reason is that there is higher interference with more transmitters.
2) There is a 2–3dB performance gap between Maximum-Likelihood and MMSE across all modulations for $N_t = 32$. However, this gap decreases to 1dB for $N_t = 16$, because of the lower interference in this case.
3) Multiple schemes perform similarly to Maximum-Likelihood, especially at lower-order modulations like QAM4. As we move to QAM64, the performance of several schemes degrades compared to Maximum-Likelihood.

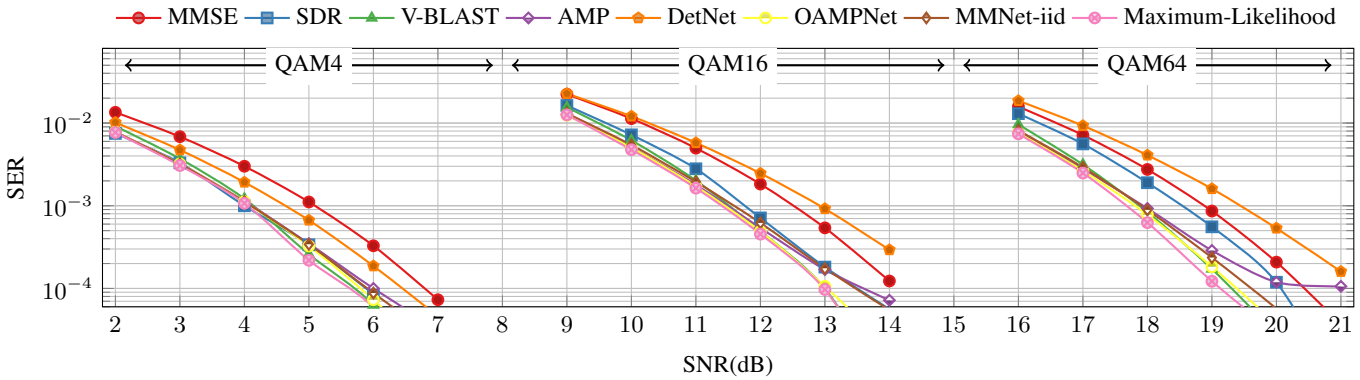
[3]The simulation results were generated using QuaDRiGa Version 2.0.0-664.

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications



(a) 32 Transmit, 64 Receive Antennas



(b) 16 Transmit, 64 Receive Antennas

Fig. 2: SER vs. SNR of different schemes for three modulations (QAM4, QAM16 and QAM64) and two system sizes (32 and 16 transmitters, 64 receivers) with i.i.d. Gaussian channels.

4) SDR performs better than MMSE, but its gap with Maximum-Likelihood increases with modulation order.
5) V-BLAST achieves almost the optimal performance across all modulations when we have 16 transmit antennas. However, its performance is sensitive to system size and degrades when we increase the number of transmitters to 32.
6) AMP is near-optimal in many cases (recall that, theoretically, AMP is asymptotically optimal for i.i.d. Gaussian channels as the system size increases). However, it suffers from robustness issues at higher SNR levels, especially with higher-order modulations like QAM64.
7) DetNet has a good performance on QAM4, but its gap with Maximum-Likelihood increases as we move to QAM16 and QAM64. With QAM64, DetNet performs even worse than MMSE for $N_t = 16$.
8) OAMPNet and the simple MMNet-iid approach are both very close to Maximum-Likelihood across different modulations over a wide range of SNRs, even though these models have only two parameters per layer.

In summary, these results show that for i.i.d. Gaussian channel matrices, adding just a small amount of flexibility via tuning parameters to existing iterative schemes like AMP can result in equivalent or improved performance over much more complex deep learning models like DetNet. Further, these models even outperform classical algorithms like SDR.

2) Realistic channels: Fig. 3 shows the results for the realistic channels derived using the 3GPP 3D MIMO channel model. We consider only MMSE (as a baseline), OAMPNet, MMNet and Maximum-Likelihood. As shown in the i.i.d. Gaussian case, schemes like SDR, V-BLAST and DetNet do not perform as well as the OAMPNet baseline.[4] Also, AMP is not designed for correlated channels and is known to perform poorly on them (see discussion in Section VI).

We make the following observations:

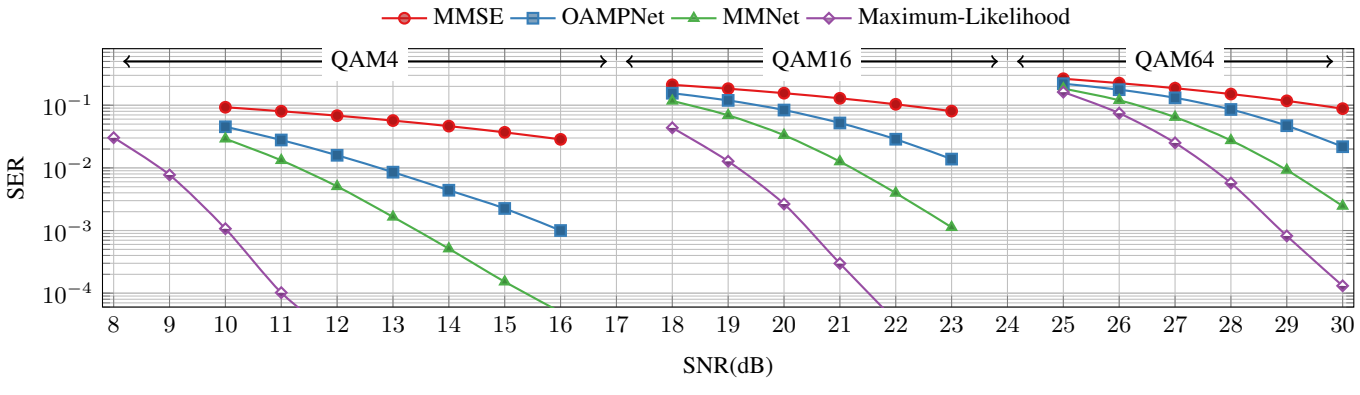
1) There is a much larger gap with Maximum-Likelihood for all detection schemes on these channels compared to the i.i.d. case.
2) To achieve comparable SER, we require 4–7dB increase in SNR relative to the i.i.d. case in Fig. 2. Also, doubling the number of transmitters from 16 to 32 incurs about a 5dB penalty in SNR for each scheme in this case (compare with 2–3dB in i.i.d. case.)
3) MMSE exhibits a relatively flat SER vs. SNR curve. For example, it requires 4dB SNR increase on QAM16 to go from an SER of 20% to 10% on 64×32 channels.
4) OAMPNet's performance improves faster than MMSE as the SNR increases. Compared to MMSE, OAMPNet can achieve a similar SER at 2–3dB lower SNR.
5) MMNet outperforms MMSE and OAMNet schemes for both system sizes and in all modulations.

In Fig. 4, we plot the performance gap with Maximum-Likelihood for these three detection schemes. For this purpose,

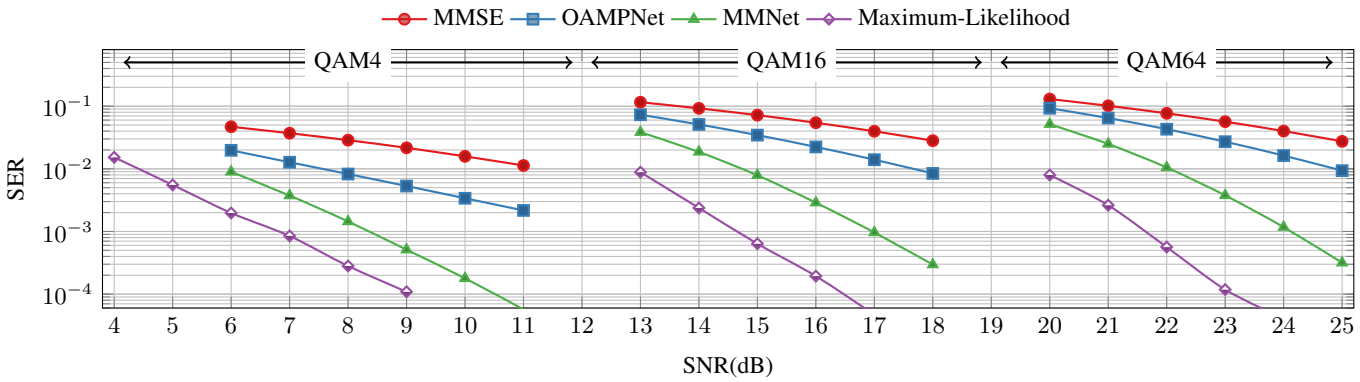
[4]We tried to train DetNet for realistic channels and ran into significant difficulty with stability and convergence in training.

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

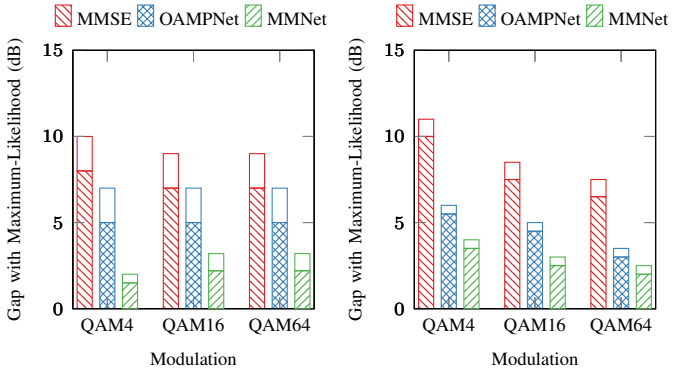


(a) 32 Transmit, 64 Receive Antennas



(b) 16 Transmit, 64 Receive Antennas

Fig. 3: SER vs. SNR of different schemes for three modulations (QAM4, QAM16 and QAM64) and two system sizes (32 and 16 transmitters, 64 receivers) with 3GPP MIMO channels.



(a) 3GPP MIMO $N_t = 16$ **(b)** 3GPP MIMO $N_t = 32$

Fig. 4: SNR requirement gap with Maximum-Likelihood at SER of 10^{-3}. The total bar height shows the 90th-percentile gap (over different channels) while the hatched section depicts the average.

we measure the difference in the minimum SNR level that is required to have SER of 10^{-3}. In this figure, we also show the 90th-percentile of the SNR gap for different channels. We observe that MMNet reduce the SNR requirement by up to 5dB and 8dB, respectively, over OAMPNet and MMSE for realistic channels.

3) Robustness to channel estimation errors: In order to evaluate MMNet's robustness against channel estimation errors [34], we consider a least-squares channel estimator which is equivalent to adding complex Gaussian noise to the channel

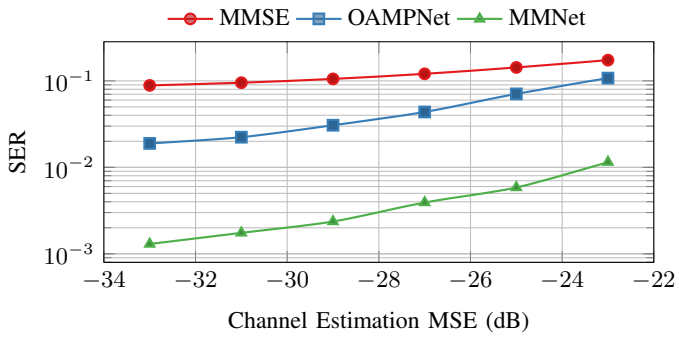


Fig. 5: SER for QAM16 versus channel estimation MSE.

matrix, i.e., : $\hat{\mathbf{H}}_{ij} = \mathbf{H}_{ij} + \boldsymbol{\delta}_{ij}$, where $\boldsymbol{\delta}_{ij} \sim \mathcal{CN}(0, \sigma_c^2)$. We provide $\hat{\mathbf{H}}$ to the signal detector instead of $\mathbf{H}$. Fig. 5 compares the impact of imperfect channel estimation on 3GPP MIMO channels for QAM16 modulation at different values of σ_c. Here we have 32 transmit and 64 receive antennas operating at 23dB SNR. We observe that MMNet is able to achieve the same SER as OAMPNet at 10dB higher channel estimation error at a fixed SNR

VI. Why MMNet works

In this section, we examine why MMNet performs better than the best previous learning-based scheme, OAMPNet. By analyzing the dynamics of the error $(\hat{\mathbf{x}}_t - \mathbf{x})$, we find that

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

MMNet's denoisers are significantly more effective than those in OAMPNet. We show that this occurs because the linear stages learned by MMNet create favorable conditions for the denoisers. Specifically, we find that the distribution of noise at the input of MMNet's denoisers is nearly Gaussian, whereas the noise distribution for OAMPNet is far from Gaussian. Since the denoisers in both architectures are optimized for Gaussian noise, they perform much better in MMNet.

A. Error dynamics

Define the error at the outputs of the linear and denoiser stages at iteration t as $\mathbf{e}_t^{lin} = \mathbf{z}_t - \mathbf{x}$ and $\mathbf{e}_t^{den} = \hat{\mathbf{x}}_{t+1} - \mathbf{x}$, respectively. For algorithms such as MMNet and OAMPNet with $\mathbf{b}_t = 0$, we can rewrite the update equations of (3) in terms of these two errors in the form:

$$\mathbf{e}_t^{lin} = (\mathbf{I} - \mathbf{A}_t\mathbf{H})\mathbf{e}_{t-1}^{den} + \mathbf{A}_t\mathbf{n} \tag{16a}$$

$$\mathbf{e}_t^{den} = \eta_t(\mathbf{x} + \mathbf{e}_t^{lin}) - \mathbf{x}. \tag{16b}$$

Eq. (16a) includes two terms, corresponding to two sources of error that contribute to $\mathbf{e}_t^{lin}$: (1) the error in the output signal from the previous iteration, and (2) the channel noise. Different choices of $\mathbf{A}_t$ impact these two terms differently. For example, if we set $\mathbf{A}_t$ to $\mathbf{H}^+$ (the pseudo-inverse of the channel matrix), we are only left with the term $\mathbf{H}^+\mathbf{n}$ in $\mathbf{e}_t^{lin}$, thus eliminating the first error term entirely. However, this comes at a price: we are left with Gaussian noise with covariance matrix $\sigma^2\mathbf{H}^+\mathbf{H}^{+H}$. This presents two potential problems: (i) if $\mathbf{H}$ is ill-conditioned, it might amplify the channel noise (e.g., inversely proportional to the smallest singular value of $\mathbf{H}$ in some directions); (ii) the resulting noise, $\mathbf{e}_t^{lin}$, may have correlated elements, and therefore applying an element-wise denoising function to it as per Eq. (16b) may be suboptimal.

Another naive option is to eliminate the channel noise term entirely by setting $\mathbf{A}_t$ to zero. This would effectively remove the linear stages from the architecture, reducing it to a cascade of denoising stages. However, applying a (well-chosen) denoiser multiple times in a cascade should be no better than applying it once.

It is also instructive to consider the error dynamics in the special case of i.i.d. channels. In this case, if we set $\mathbf{A}_t = \mathbf{H}^H$, the factor $\mathbf{I} - \mathbf{A}_t\mathbf{H}$ asymptotically goes to zero as we increase N_r [21], and the auto-covariance of $\mathbf{A}_t\mathbf{n}$, $\sigma^2\mathbf{H}^H\mathbf{H}$, is approximately equal to $\sigma^2\mathbf{I}_{N_t}$. This means that the linear stage does not amplify the channel noise or make it correlated. On the other hand, the error from the previous iteration, $\mathbf{e}_{t-1}^{den}$, is attenuated significantly via $\mathbf{I}-\mathbf{A}_t\mathbf{H}$. These calculations explain why AMP has great performance on i.i.d. Gaussian channels. However, in case of correlated channels, neither $\mathbf{I} - \mathbf{A}_t\mathbf{H}$ is close to zero, nor is $\mathbf{A}_t\mathbf{n}$ uncorrelated, and therefore AMP does not perform well on realistic channels.

B. Analysis

Based on the above discussion of the error dynamics, we identify two desirable properties for picking $\mathbf{A}_t$:

1) *Noise reduction property:* $\mathbf{A}_t$ must reduce the magnitude of $\mathbf{e}_t^{lin}$. This requires striking a balance between the two terms in (16a), because attenuating one term may amplify the other and vice-versa.
2) *Uncorrelated Gaussian noise property:* $\mathbf{A}_t$ must "shape" the distribution of $\mathbf{e}_t^{lin}$ to make it suitable for the subsequent denoising stage. In particular, the denoisers in most iterative schemes (e.g., MMNet and OMAPNet) are specifically designed for uncorrelated Gaussian noise. Thus, ideally, the linear stage should avoid outputting correlated or non-Gaussian noise.

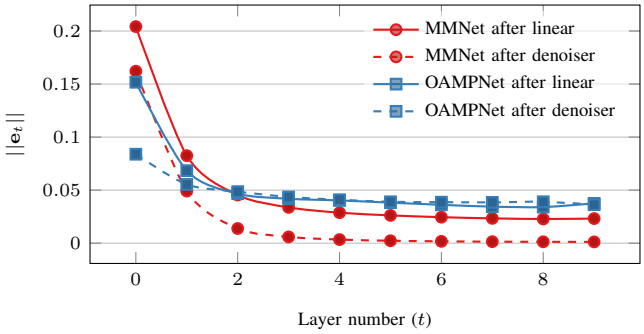


Fig. 6: Noise power after the linear and denoiser stages at different layers of OAMPNet and MMNet. The OAMPNet denoisers become ineffective after the third layer on 3GPP MIMO channels.

Fig. 6 shows the average noise power at the output of the linear and denoiser stages across different layers, for both OAMPNet and MMNet on 64×16 3GPP MIMO channels. We observe that for OAMPNet, the average noise power after the linear stage (before the denoiser) and after the denoiser is the same from the third layer ($t = 2$) onwards. In other words, in OAMPNet, the denoisers are unable to reduce the noise in their input signal after a few layers. However, with MMNet, the noise power is significantly lower at the output of the denoisers compared to their input at all layers.

We hypothesize that the reason OAMPNet's denoisers become ineffective is that the noise distribution for OAMPNet is not Gaussian, whereas MMNet is able to provide near-Gaussian noise to its denoisers. We evaluate how close the noise distribution is to Gaussian for both schemes using the Anderson test [35]. In order to measure this score, we generate 10,000 samples of $\mathbf{x}$ and $\mathbf{y}$ per channel realization $\mathbf{H}$. We run each sample through both MMNet and OAMPNet, and we calculate the Anderson score for the noise distribution at the output of the linear stages, for each transmitter and channel matrix. If this score is below a threshold of 0.786, it indicates that the noise comes from a Gaussian distribution with a significance of 5%, i.e. the probability of false rejection of a Gaussian distribution is less than 5%. In Fig. 7, we plot the average fraction of transmitters that have Gaussian distributed noise at the output of the linear stage according to this test. Since in both schemes we start with $\hat{\mathbf{x}}_0 = 0$, the output of the linear stage at layer $t = 0$ is $\mathbf{A}_0\mathbf{n}$, which is Gaussian. Thus, the fraction of transmitters with Gaussian noise is 1 in layer $t = 0$ for both schemes. However, both schemes deviate from Gaussian noise in layer $t = 1$, while at the same time sharply reducing the total noise power as seen in Fig. 6. However, the noise for fewer transmitters in MMNet deviate from a Gaussian distribution. Unlike OAMPNet, in which the noise

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

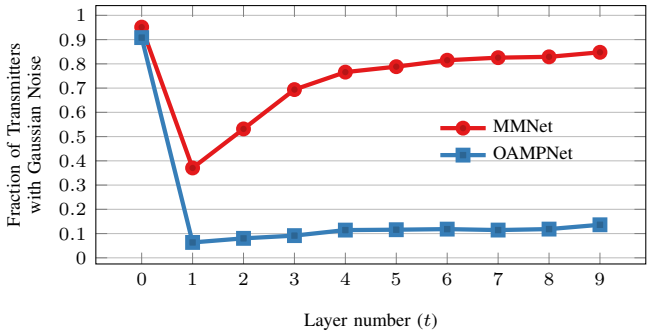


Fig. 7: Fraction of transmitters that have Gaussian error distribution after the linear block for each layer with significance level of 5%. MMNet produces Gaussian distributed errors at the output of linear blocks, while OAMPNet fails to achieve the Gaussian property.

for 95% of the transmitters is not Gaussian at layer $t = 1$, for MMNet nearly 40% of the transmitters exhibit Gaussian noise. On the other hand, MMNet reduces the noise power slightly less than OAMPNet in layer $t = 1$.

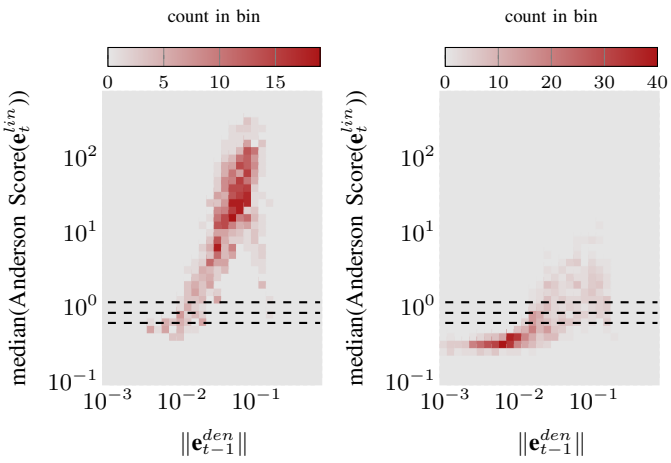
In subsequent layers, the noise distribution for MMNet becomes increasingly Gaussian, with nearly 90% of transmitter passing the Anderson test by layer $t = 9$. By contrast, most transmitters in OAMPNet continue to exhibit non-Gaussian noise in subsequent layers, though the fraction of transmitter with Gaussian noise marginally increases.

Next, we measure the effect of input noise power on the distribution of noise at the output of the linear stages in both schemes. In other words, we are interested to know how $\|\mathbf{e}_{t-1}^{den}\|$ impacts the Gaussian distribution property of $\mathbf{e}_t^{lin}$. For this purpose, we choose the median of Anderson scores as a measure of the linear stage's ability to maintain the Gaussian property at its output. In Fig. 8, we show the 2D histogram of this median score for different values of $\|\mathbf{e}_{t-1}^{den}\|$. For reference, we also plot three thresholds corresponding to 1%, 5% and 15% significance for the normality test as dashed horizontal lines. To be Normally distributed with 1%, 5%, or 15% confidence, the Anderson scores must fall below the respective line.

We notice that the median score in both schemes increases with the norm of the error from the previous iteration. In other words, the linear stages that have a higher input noise power produce outputs that deviate more significantly from a Gaussian. However, MMNet is $100\times$ better in terms of the median Andersen score for large values of $\|\mathbf{e}_{t-1}^{den}\|$. This figure also suggests that OAMPNet's inability to achieve the Gaussian property is not limited to the first couple of layers, in which it aggressively reduces the noise power. The later linear stages, for which $\|\mathbf{e}_{t-1}^{den}\|$ is fairly small, are also not very good at achieving the Gaussian property.

C. Impact of channel condition number

Finally, we evaluate the impact of the channel *condition number* on MMNet and OAMPNet. A channel's condition number is defined as the ratio of its largest singular value to the smallest. It is well-known that symbol detection is difficult for channel matrices with higher condition number.



(a) OAMPNet **(b)** MMNet

Fig. 8: Median of Anderson score for the noise at the output of linear stage vs. the power of noise at the input of the stage. MMNet achieves the Gaussian property much more effectively for all noise power levels. Dashed horizontal lines show the thresholds for 1%, 5% and 15 significance level.

In Fig. 9a, we show a scatterplot depicting the fraction of transmitters with the Gaussian noise property at output of the linear stage in layer $t = 4$ vs. channel condition number for the 3GPP MIMO dataset. We show the behavior at layer $t = 4$ as an example of what occurs in the first few iterations of the detection process. We see that, for OAMPNet, the fraction of transmitters satisfying the Gaussian noise property decreases sharply for channels with higher condition number. By contrast, MMNet maintains the Gaussian noise property for a much broader range of channels. The consequence of the failure to achieve the Gaussian property on SER is shown in Fig. 9b. Although the performance of all schemes degrades as the channel condition number increases, MMNet is able to achieve an SER of less than 10^{-3} for a wider range of channel conditions.

VII. Online Training Algorithm

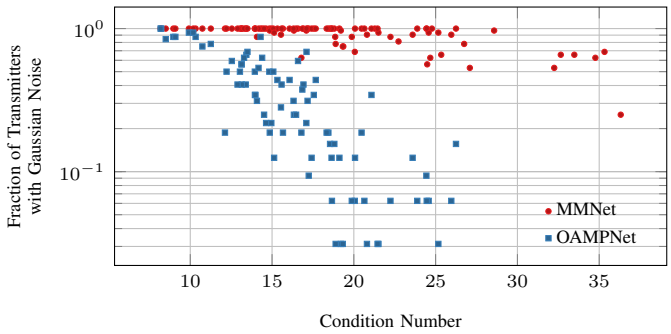
Training deep learning models is a computationally intensive task, often requiring hours or even days for large models. The computation overhead depends on two factors: (i) the total number of required training samples, and (ii) the size of the model. For example, to train a model like DetNet with about 1M parameters, we need 50K iterations with a batch size of 5K samples, i.e., 250M training samples. If we assume each parameter of the model shows up in at least one floating-point operation per training sample, we require a minimum of 2.5×10^{14} floating-point operations for the entire training process. This computational complexity makes training such a large model online for each realization of $\mathbf{H}$ impossible.[5]

In comparison, MMNet has only ∼40K parameters, and training it from scratch requires about 1000 iterations with batch size of 500. Further, we show that by taking advantage of locality of the channels observed at a receiver, we can further

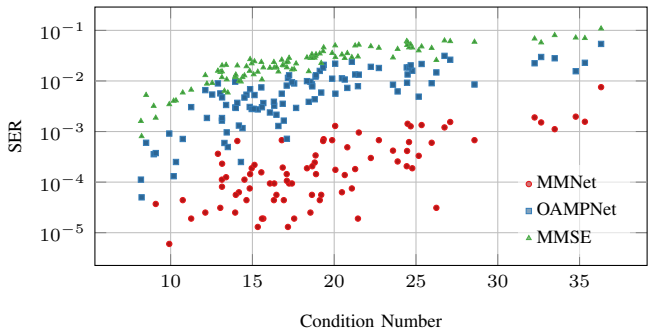
[5]Of course, DetNet was specifically designed for offline training, where computational complexity is less of a concern.

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications



(a) Fraction of transmitters with Gaussian noise at layer $t = 4$



(b) Symbol error rate

Fig. 9: Effect of channel condition number on performance in the 3GPP MIMO dataset. (a) MMNet is more robust in maintaining the Gaussian noise property for channels with large condition number. (b) SER is directly affected by the condition number.

reduce training cost to 9 iterations (with batch size 500) on average per channel realization. All in all, training MMNet has six orders of magnitude lower computational overhead than DetNet, making online training for each realization of $\mathbf{H}$ practical.

In this section, we first discuss the temporal and spectral locality of realistic channels. Then, we show how we can exploit these localities to accelerate online training.

A. Channel locality

The channel matrices measured at a base station exhibit two forms of locality:

- *Temporal:* Channel matrices change gradually over time as user devices move within a cell or the multipath environment changes. The samples of $\mathbf{H}$ at nearby points in time are thus correlated.
- *Spectral*: The base station needs to recover signals from several frequency subcarriers (1024 in our 3GPP MIMO model). The channels for subcarriers in nearby frequencies are also strongly correlated. The frequency affects the phase of the multipath signal components incident on receiver antennas. For a path of length l, the phase difference for two subcarriers Δf apart in frequency is $\Delta\phi = \frac{2\pi l \Delta f}{c}$. Therefore the received signal and the channels for adjacent subcarriers will be similar at each receiver antenna.

Both forms of locality reduce the complexity of training for each channel realization, because (i) the cost of channel-specific computations can be amortized across multiple correlated channel realizations across time and frequency, (ii) the trained model for one channel realization can serve as strong initialization for training for adjacent channel realizations in the time-frequency plane.

Fig. 10 shows both forms of locality by plotting the correlation among the 3GPP MIMO channel samples across time and frequency (subcarriers). To compute these correlations, we take the average of the inner-product of each channel matrix with its neighboring channel matrices separated in time or frequency by different number of steps. A shift of one step in the time domain corresponds to two channel matrices at the same subcarrier frequency that are 118ms apart in time.

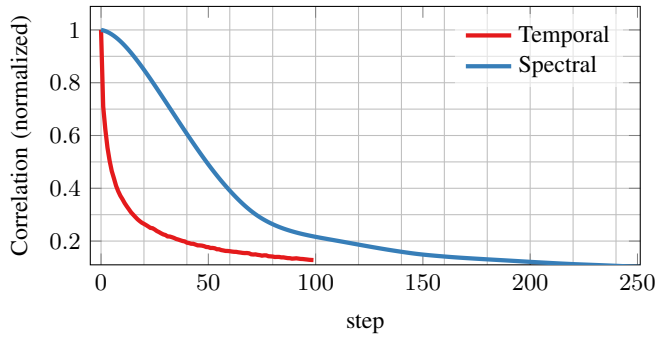


Fig. 10: Correlation of channel samples across time and frequency dimensions. The correlation decays relatively quickly in the time dimension, but the channel matrices show strong locality across subcarriers in frequency dimension.

A shift of 1 step in the frequency domain corresponds to two channel matrices at adjacent subcarriers (78.1KHz apart) at the same time. We normalize the inner-product by the norm of the matrices, such that the correlation of a channel matrix with itself is 1. As the figure show, we observe a stronger locality in the frequency domain than in the time domain in the 3GPP MIMO channels.

B. Training algorithm and results

In this section, we show how channel locality can help reduce the total number of operations MMNet needs to decode each received signal $\mathbf{y}$ at the BS. The computational complexity of MMNet is mostly dominated by the cost of online training for each new realization of the channel $\mathbf{H}$. This cost in turn depends on the channel *coherence time*. In the case of a quasi-static channel, as expected for instance in fixed-wireless access or backhaul solutions such as 5G home wireless (see Section 7.6.2 in [2]), the channel between the transmitter and receiver does not change for extended periods of time. In such cases, MMNet does not require frequent retraining and can reuse the same model until the communication channel changes significantly. However, MMNet can also operate at reasonable computation cost when the channel is changing fairly frequently. For example, our 3GPP MIMO channel samples were generated assuming all devices constantly move at a speed of 1 m/s, and after about 500ms, the channel correlation is less than 0.5. However, even in this scenario,

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

Algorithm 1 MMNet online training

$\mathcal{M} \leftarrow$ Construct MMNet with parameters $\vartheta = \{\boldsymbol{\Theta}_t^{(1)}, \boldsymbol{\theta}_t^{(2)}\}_{t=1}^{T}$
$\vartheta \leftarrow$ Initialize model parameters randomly
for $n \in \{1,\ 2,\ ...\}$ **do** ▷ n keeps the time step
  for $f \in \{1,\ 2,\ ...,\ F\}$ **do**
    $\mathbf{H}[f] \leftarrow$ Measured channel at time step n and frequency f
    #TrainIterations $\leftarrow$ Set to Φ if $f \neq 1$ and Ψ otherwise
    for it$\in \{1, 2, ...,$#TrainIterations$\}$ **do**
      $\mathbf{D} \leftarrow$ Generate random $(\mathbf{x}, \mathbf{y})$ batch on $\mathbf{H}[f]$ using (1)
      $L \leftarrow$ Find the loss in (15) for $\mathcal{M}$ over the samples in $\mathbf{D}$
      $\vartheta \leftarrow$ Compute the model updates using $\nabla_\vartheta L$
    end for
    $\mathcal{M}_n[f] \leftarrow \mathcal{M}$.copy( ) ▷ Store the parameters ϑ
  end for
end for

we require only 9 training iterations on average per channel realization, as explained next.

To see how, note that a receiver at a BS must simultaneously decode signals from different subcarriers. Since channels exhibit strong correlations across sub-carriers (Fig. 10), training the MMNet detector on $\mathbf{H}$ for one subcarrier produces a detector that will achieve very similar performance on adjacent subcarriers. The performance of this detector will however decay for more distant subcarriers in the frequency domain.

Based on this observation, we propose the online training scheme in Algorithm 1. We start from a random initialization of the MMNet neural network model $\mathcal{M}$. We define n as an index for time intervals in which we can assume that channels do not change substantially. For each interval n, we measure a channel matrix $\mathbf{H}[f]$ for each subcarrier frequency f. The basic idea in the algorithm is to train the model for Ψ iterations (with a batch size of 500) for the first subcarrier ($f = 1$), then retrain the model using only Φ additional training iterations per subcarrier for all subsequent subcarriers. Typical values for the hyperparameters Ψ and Φ are respectively 1000 and 3–10 in our experiments. For each channel matrix $\mathbf{H}[f]$, we generate $(\mathbf{x}, \mathbf{y})$ training data pairs using (1). Once the model has trained for subcarrier f, we save a copy of the model as $\mathcal{M}_n[f]$ for detecting all signals received in time interval n on that subcarrier. We repeat the entire training algorithm in each time interval. In Fig. 11, we show the result of our online training method on 3GPP MIMO channels for the QAM16 modulation with 16 transmit and 64 receive antennas at 18dB SNR. In this experiment, we set $\Psi = 1000$ and compare MMNet with other schemes on a range of Φ values. We see that while MMSE and OAMPNet achieve a SER of 0.03 and 0.009 respectively, MMNet can bring the SER down to 0.0007 (below the 10^{-3} threshold). With this approach, MMNet performs 9.19K iterations of training with batch size 500 in order to learn a detector for all 1024 subcarriers in total at each time interval n. Therefore the cost of online training is less than 9 iterations on average per channel realization, yet MMNet delivers better performance than other schemes, like OAMPNet and MMSE.

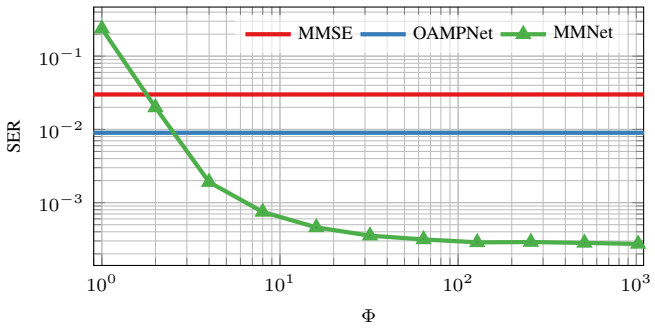


Fig. 11: SER vs. Φ using Algorithm 1 with $\Psi = 10^3$ for training MMNet on QAM16 modulation. MMNet requires only 9 overall iterations of batch size 500 per channel realization to train to a reasonable performance with $(\Psi, \Phi) = (10^3, 8)$.

C. Computational complexity

One iteration of training MMNet on a batch of size of b has a complexity of $\mathcal{O}(bN_r^2)$, as detection takes $\mathcal{O}(N_r^2)$ in MMNet. To put this in perspective, a light-weight algorithm like AMP has a complexity of $\mathcal{O}(N_r^2)$ dominated by the multiplication of the channel matrix and residual vectors. The MMSE scheme has a higher complexity of $\mathcal{O}(N_r^3)$ because it needs to invert a matrix. OAMPNet similarly requires a matrix inversion, resulting in a complexity of $\mathcal{O}(N_r^3)$.

Moving beyond $\mathcal{O}(\cdot)$ analysis, Fig. 12 shows the average number of multiplication operations required per signal detection on 3GPP MIMO channels for learning-based algorithms in addition to two classic baselines, MMSE and AMP. For these numbers, all algorithms reuse computations whenever feasible. In particular, in every channel coherence interval in the 3GPP MIMO model, each algorithm receives ∼100 signals to detect [2, Definition 2.2 on page 220]. MMSE calculates the required channel matrix inverse only once for all 100 received signals in the coherence interval. This reduces computation complexity for MMSE by a factor of 100×, resulting in 5–7× fewer multiplications than AMP, which cannot reuse computation but has modest complexity. MMNet reuses the weights it trains (with 9 iterations of batch size 500 on average) for all 100 received signals in a coherence interval. MMNet, with its online training and detection operations combined, places after AMP with 2–5× fewer multiplications than pre-trained DetNet. However, as we have seen, neither AMP nor DetNet extend to realistic, spatially correlated channels. OAMPNet's computational complexity is higher than the other models, because it has to calculate a new matrix inverse in each layer for every received signal, as v_t^2 in (10) depends on $\hat{\mathbf{x}}_t$.

Consequently, the cost of MMNet with its online training algorithm is 10–15× less than OAMPNet depending on the system size. MMNet has 41× higher computational complexity than a light-weight iterative approach like AMP, which only works for i.i.d. Gaussian channels.

VIII. Conclusion and Future Work

This paper proposed a deep learning architecture for Massive MIMO detection, and an online training algorithm to optimize it for every realization of the channel matrix at a base station. MMNet outperforms state-of-the-art detection

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

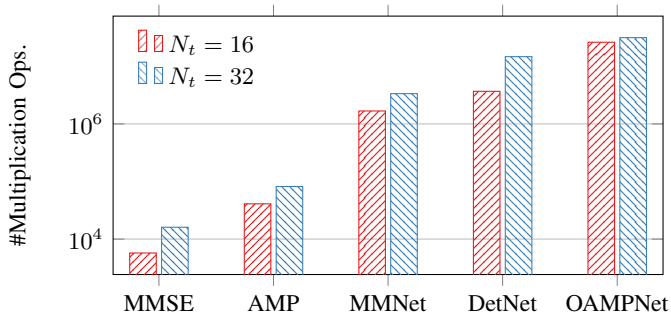


Fig. 12: Number of multiplication operations per signal detection for different algorithms on QAM16 with $N_r = 64$ receive antennas in 3GPP MIMO model. Detection with MMNet, including its online training process, requires fewer multiplication operations than detection with pre-trained DetNet and OAMPNet models.

algorithms on realistic channels with spatial correlation. We designed MMNet as an iterative algorithm and showed that a carefully chosen degree of flexibility in the model, in addition to leveraging the channel's spectral and temporal correlation, can enable online training at a less or equal computation complexity than other deep-learning based schemes. MMNet is 4–8dB better overall than the classic MMSE detector and it requires 2.5dB lower SNR at the same SER, relative to the second-best detection scheme, OAMPNet, at 10–15$\times$ less computational complexity. Many extensions of MMNet are possible to support, for example, a varying number of transmitters with possibly different modulation schemes.

From a hardware perspective, implementing MMNet has its own challenges and requires an in-depth study. For example, the sequential online training algorithm introduced in this paper incurs latency, which may be traded off with parallel training of multiple sub-carriers at the cost of more training iterations and hence increased complexity. The optimal trade-off depends on the channel coherence time.

Our results show that evaluations based on simple channel models such as i.i.d. Gaussian channels can lead to misleading conclusions for MIMO detection performance. Future work should therefore evaluate on realistic channel models, from either simulation, ray-tracing, or measurements. We have released the simulated 3GPP MIMO channel dataset used in this work in hope that it will serve as a useful benchmark for the community.

IX. Acknowledgements

This work was funded in part by National Science Foundation grants CNS-1563826 and CNS-1751009, and the Alfred P. Sloan Research Fellowship.

References

[1] Afif Osseiran, Jose F Monserrat, and Patrick Marsch. *5G Mobile and Wireless Communications Technology*. Cambridge University Press, 2016.

[2] Emil Björnson, Jakob Hoydis, and Luca Sanguinetti. Massive MIMO networks: Spectral, energy, and hardware efficiency. *Foundations and Trends® in Signal Processing*, 11(3-4):154–655, 2017. ISSN 1932-8346. doi: 10.1561/2000000093. URL http://dx.doi.org/10.1561/2000000093.

[3] Emre Telatar. Capacity of Multi-antenna Gaussian Channels. *Transactions on Emerging Telecommunications Technologies*, 10(6):585–595, 1999.

[4] Emanuele Viterbo and Joseph Boutros. A universal lattice code decoder for fading channels. *IEEE Transactions on Information theory*, 45(5): 1639–1642, 1999.

[5] Shaoshi Yang and Lajos Hanzo. Fifty years of mimo detection: The road to large-scale mimos. *IEEE Communications Surveys & Tutorials*, 17(4):1941–1988, 2015.

[6] Erik G Larsson. MIMO detection methods: How they work. *IEEE signal processing magazine*, 26(3), 2009.

[7] Ami Wiesel, Yonina C Eldar, and Shlomo Shamai. Semidefinite relaxation for detection of 16-qam signaling in mimo channels. *IEEE Signal Processing Letters*, 12(9):653–656, 2005.

[8] Neev Samuel, Tzvi Diskin, and Ami Wiesel. Learning to detect. *IEEE Transactions on Signal Processing*, 67(10):2554–2564, 2019.

[9] Neev Samuel, Tzvi Diskin, and Ami Wiesel. Deep mimo detection. In *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, pages 1–5. IEEE, 2017.

[10] Hengtao He, Chao-Kai Wen, Shi Jin, and Geoffrey Ye Li. A model-driven deep learning network for mimo detection. *arXiv preprint arXiv:1809.09336*, 2018.

[11] Junjie Ma and Li Ping. Orthogonal amp. *IEEE Access*, 5:2020–2033, 2017.

[12] Sergey L Loyka. Channel capacity of mimo architecture using the exponential correlation matrix. *IEEE Communications letters*, 5(9):369–371, 2001.

[13] 3GPP TR 36.873. 2015. "Study on 3D channel model for LTE". Technical report.

[14] Stephan Jaeckel, Leszek Raschkowski, Kai Börner, and Lars Thiele. QuaDRiGa: A 3-D multi-cell channel model with time evolution for enabling virtual field trials. *IEEE Trans. Antennas Propag.*, 62(6):3242–3256, 2014.

[15] Ingrid Daubechies, Michel Defrise, and Christine De Mol. An iterative thresholding algorithm for linear inverse problems with a sparsity constraint. *Communications on Pure and Applied Mathematics: A Journal Issued by the Courant Institute of Mathematical Sciences*, 57 (11):1413–1457, 2004.

[16] Amir Beck and Marc Teboulle. A fast iterative shrinkage-thresholding algorithm for linear inverse problems. *SIAM journal on imaging sciences*, 2(1):183–202, 2009.

[17] Charles Jeon, Ramina Ghods, Arian Maleki, and Christoph Studer. Optimality of large mimo detection via approximate message passing. In *Information Theory (ISIT), 2015 IEEE International Symposium on*, pages 1227–1231. IEEE, 2015.

[18] Lajos Hanzo, Soon Xin Ng, WT Webb, and T Keller. *Quadrature amplitude modulation: From basics to adaptive trellis-coded, turbo-equalised and space-time coded OFDM, CDMA and MC-CDMA systems*. IEEE Press-John Wiley, 2004.

[19] Alberto Del Pia, Santanu S Dey, and Marco Molinaro. Mixed-integer quadratic programming is in np. *Mathematical Programming*, 162(1-2): 225–240, 2017.

[20] Amine Mezghani and Josef A Nossek. Belief propagation based mimo detection operating on quantized channel output. In *Information Theory Proceedings (ISIT), 2010 IEEE International Symposium on*, pages 2113–2117. IEEE, 2010.

[21] Mohsen Bayati and Andrea Montanari. The dynamics of message passing on dense graphs, with applications to compressed sensing. *IEEE Transactions on Information Theory*, 57(2):764–785, 2011.

[22] Antonia M Tulino, Sergio Verdú, et al. Random matrix theory and wireless communications. *Foundations and Trends® in Communications and Information Theory*, 1(1):1–182, 2004.

[23] Peter W Wolniansky, Gerard J Foschini, GD Golden, and Reinaldo A Valenzuela. V-blast: An architecture for realizing very high data rates over the rich-scattering wireless channel. In *Signals, Systems, and Electronics, 1998. ISSSE 98. 1998 URSI International Symposium on*, pages 295–300. IEEE, 1998.

[24] Mahesh K Varanasi and Behnaam Aazhang. Multistage detection in asynchronous code-division multiple-access communications. *IEEE Transactions on communications*, 38(4):509–519, 1990.

[25] WH Chin, AG Constantinides, and DB Ward. Parallel multistage detection for multiple antenna wireless systems. *Electronics Letters*, 38(12):597–599, 2002.

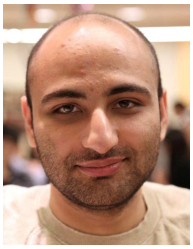
[26] Ori Shental, Sivarama Venkatesan, Alexei Ashikhmin, and Reinaldo A Valenzuela. Massive blast: An architecture for realizing ultra-high data rates for large-scale mimo. *arXiv preprint arXiv:1708.05405*, 2017.

[27] Toshiyuki Tanaka and Masato Okada. Approximate belief propagation, density evolution, and statistical neurodynamics for cdma multiuser detection. *IEEE Transactions on Information Theory*, 51(2):700–706, 2005.

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.

This article has been accepted for publication in a future issue of this journal, but has not been fully edited. Content may change prior to final publication. Citation information: DOI 10.1109/TWC.2020.2996144, IEEE Transactions on Wireless Communications

[28] Vinod Nair and Geoffrey E Hinton. Rectified linear units improve restricted boltzmann machines. In *Proceedings of the 27th international conference on machine learning (ICML-10)*, pages 807–814, 2010.
[29] Wing-Kin Ma, Chao-Cheng Su, Joakim Jaldén, and Chong-Yung Chi. Some results on 16-qam mimo detection using semidefinite relaxation. In *2008 IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2673–2676. IEEE, 2008.
[30] Inc. Gurobi Optimization. Gurobi optimizer reference manual, 2016. URL http://www.gurobi.com.
[31] Martín Abadi, Ashish Agarwal, Paul Barham, Eugene Brevdo, Zhifeng Chen, Craig Citro, Greg S. Corrado, Andy Davis, Jeffrey Dean, Matthieu Devin, Sanjay Ghemawat, Ian Goodfellow, Andrew Harp, Geoffrey Irving, Michael Isard, Yangqing Jia, Rafal Jozefowicz, Lukasz Kaiser, Manjunath Kudlur, Josh Levenberg, Dandelion Mané, Rajat Monga, Sherry Moore, Derek Murray, Chris Olah, Mike Schuster, Jonathon Shlens, Benoit Steiner, Ilya Sutskever, Kunal Talwar, Paul Tucker, Vincent Vanhoucke, Vijay Vasudevan, Fernanda Viégas, Oriol Vinyals, Pete Warden, Martin Wattenberg, Martin Wicke, Yuan Yu, and Xiaoqiang Zheng. TensorFlow: Large-scale machine learning on heterogeneous systems, 2015. URL https://www.tensorflow.org/. Software available from tensorflow.org.
[32] Wing-Kin Ma, Chao-Cheng Su, Joakim Jaldén, Tsung-Hui Chang, and Chong-Yung Chi. The equivalence of semidefinite relaxation mimo detectors for higher-order qam. *IEEE Journal of Selected Topics in Signal Processing*, 3(6):1038–1052, 2009.
[33] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
[34] Hengtao He, Chao-Kai Wen, Shi Jin, and Geoffrey Ye Li. Model-driven deep learning for joint mimo channel estimation and signal detection. *arXiv preprint arXiv:1907.09439*, 2019.
[35] Theodore W Anderson, Donald A Darling, et al. Asymptotic theory of certain" goodness of fit" criteria based on stochastic processes. *The annals of mathematical statistics*, 23(2):193–212, 1952.



Mehrdad Khani is a Ph.D. candidate in the Computer Science and Artificial Intelligence Lab of Massachusetts Institute of Technology, advised by Professor Mohammad Alizadeh. His research interests are in computer systems, machine learning, and signal processing spanning adaptive algorithms, video streaming, and large-scale machine learning systems. He received his B.Sc. degrees in two majors of Electrical Engineering and Computer Science from Sharif University of Technology, Iran, in 2016.

Mohammad Alizadeh is an Associate Professor of Computer Science at the Massachusetts Institute of Technology. His research interests are in the areas of computer networks, systems, and applied machine learning. His current research focuses on learning-augmented systems, algorithms and protocols for datacenter and wide-area networks, and video streaming. Mohammad's research on datacenter transport protocols has been implemented in Linux and Windows, and has been deployed by large network operators; his work on adaptive network load balancing algorithms has been implemented in Cisco's flagship datacenter switching products. Mohammad received his Ph.D. from Stanford University in 2013, and then spent two years at Insieme Networks (a datacenter networking startup) and Cisco Systems before joining MIT. He is a recipient of several awards, including the Microsoft Research Faculty Fellowship (2019), VMware Systems Research Award (2019), NSF CAREER Award (2018), SIGCOMM Rising Star Award (2017), Alfred P. Sloan Research Fellowship (2017), and multiple best paper awards.

Jakob Hoydis (S'08–M'12) received the diploma degree (Dipl.-Ing.) in electrical engineering and information technology from RWTH Aachen University, Germany, and the Ph.D. degree from Supélec, Gif-sur-Yvette, France, in 2008 and 2012, respectively. He is currently head of a research department at Nokia Bell Labs, France, focusing on radio systems and artificial intelligence. Prior to this, he was co-founder and CTO of the social network SPRAED and worked for Alcatel-Lucent Bell Labs in Stuttgart, Germany. His research interests are in the areas of machine learning, cloud computing, SDR, large random matrix theory, information theory, signal processing, and their applications to wireless communications. He is a co-author of the textbook "Massive MIMO Networks: Spectral, Energy, and Hardware Efficiency" (2017). He is recipient of the 2019 VTG IDE Johann-Philipp-Reis Prize, the 2019 IEEE SEE Glavieux Prize, the 2018 IEEE Marconi Prize Paper Award, the 2015 IEEE Leonard G. Abraham Prize, the IEEE WCNC 2014 best paper award, the 2013 VDE ITG Foerderpreis Award, and the 2012 Publication Prize of the Supélec Foundation. He has received the 2018 Nokia AI Innovation Award, the 2018 Nokia France Top Inventor Award, and has been nominated as an Exemplary Reviewer 2012 for the IEEE Communication Letters. He is currently chair of the IEEE COMSOC Emerging Technology Initiative on Machine Learning as well as Editor of the IEEE Transactions on Wireless Communications.

Phil Fleming (S'18–M'13) received his Ph.D in Mathematics from the University of Michigan in 1981. He currently heads Ann Arbor Analytics LLC doing research and consulting in applications of digital signal processing, applied probability and machine learning to technical challenges in wireless communications and adjacent fields. In 2017 he was named a Bell Labs Fellow and in 2018 was appointed Co-Chief AI Officer in Nokia Bell Labs. Before that he held the title of CTO of North America and Head of the Network Advanced Technologies group in Nokia Network's CTO organization. He is a winner of the 2014 Inaugural Nokia Innovation Prize, the 2014 Leading Lights Award for Best New Product (Mobile) and the 2014 CTIA Emerging Technology Award for Wide Area Networks. He earned the Dan Noble Fellow award in Motorola in 2007. He has 21 patents, 12 journal papers and 21 conference papers on wireless technologies and related areas.

1536-1276 (c) 2020 IEEE. Personal use is permitted, but republication/redistribution requires IEEE permission. See http://www.ieee.org/publications_standards/publications/rights/index.html for more information.
Authorized licensed use limited to: MIT Libraries. Downloaded on July 07,2020 at 23:17:28 UTC from IEEE Xplore. Restrictions apply.