Robust MIMO Detection using Hypernetworks with Learned Regularizers

Nicolas Zilberstein*, Chris Dick†, Rahman Doost-Mohammady*, Ashutosh Sabharwal*, Santiago Segarra*
*Rice University, USA †Nvidia, USA

***Abstract*—Optimal symbol detection in multiple-input multiple-output (MIMO) systems is known to be an NP-hard problem. Recently, there has been a growing interest to get reasonably close to the optimal solution using neural networks while keeping the computational complexity in check. However, existing work based on deep learning shows that it is difficult to design a generic network that works well for a variety of channels. In this work, we propose a method that tries to strike a balance between symbol error rate (SER) performance and generality of channels. Our method is based on hypernetworks that generate the parameters of a neural network-based detector that works well on a specific channel. We propose a general framework by regularizing the training of the hypernetwork with some pre-trained instances of the channel-specific method. Through numerical experiments, we show that our proposed method yields high performance for a set of prespecified channel realizations while generalizing well to all channels drawn from a specific distribution.**

***Index Terms*—MIMO detection, deep learning, hypernetwork**

I. Introduction

Multiple-input multiple-output (MIMO) systems are an essential part of modern communications [1], [2]. Moreover, they are expected to play a fundamental role in moving from the fifth to the sixth generation of cellular communications by achieving high data rates and spectral efficiency [3]. In MIMO systems, base stations are equipped with multiple antennas, enabling them to handle several users simultaneously. However, these systems entail many challenges such as performing efficient symbol detection, which is the focus of our paper.

Exact MIMO detection is an NP-hard problem [4]. Given a modulation of M symbols and N_u, the exact maximum likelihood (ML) estimator has an exponential complexity $\mathcal{O}(M^{N_u})$. Thus, obtaining this ML estimate is computationally infeasible and becomes intractable even for moderately-sized systems. Several approximate solutions for symbol detection have been proposed in the classical literature including zero forcing (ZF) and minimum mean squared error (MMSE) [5]. Although both (linear) detectors have low complexity and achieve a good performance for small systems, their performance degrades severely for larger systems [6]. Another classical detector is approximate message passing (AMP), which is asymptotically optimal for large MIMO systems with Gaussian channels, but degrades significantly for other (more pratical) channel distributions [7].

Recently, machine learning and, in particular, deep learning have been proposed to solve fundamental problems in wireless communications such as power allocation [8]–[10], link scheduling [11], [12], and random access control [13]. For the particular case of MIMO symbol detection, several solutions have been derived [14]–[17]. At a high level, one can categorize the existing methods into two classes: 1) *channel-specific methods* learn to perform symbol detection for a prespecified channel realization, and 2) *channel-agnostic methods* can perform symbol detection for a wide variety of channels, typically drawn from a distribution of interest. MMNet [14] and the so-called fixed channel version of DetNet [15] are examples of channel-specific methods whereas HyperMIMO [16], RE-MIMO [17], and the varying channel version of DetNet [15] are examples of channel-agnostic methods. Naturally, the first class attains very high performance for the channels in which they were trained but fail in other channels from the same distribution. However, as they have to be trained for each channel realization, they are typically unsuitable for real-time applications. In contrast, the second class generalizes well across a distribution without the need of retraining but cannot match the performance of the first class on the specific channels where they were trained.

Our goal is to combine these two classes of methods to attain a solution that yields very high performance for a set of prespecified channels (and their perturbations) and generalizes well to all channels coming from a distribution of interest without the need to retrain. We achieve this by first constructing a channel-agnostic methods based on a channel-specific one using the concept of hypernetworks [18], and then regularizing the training of the hypernetwork with several pre-trained instances of the channel-specific method. Although the framework proposed is generic in terms of which channel-specific detector to choose, we focus on MMNet [14] and its corresponding hypernetwork extension, the HyperMIMO [16].

Contribution. The contributions of this paper are twofold:
1) We propose a learning-based solution for MIMO detection that yields high accuracy for perturbations of a prespecified set of channels while generalizing to a whole distribution. We attain this via a HyperMIMO architecture whose training is regularized by solutions of the MMNet.
2) Through numerical experiments, we demonstrate that the proposed solution achieves symbol error rates below those obtained by HyperMIMO and MMNet trained separately while maintaining the (forward-pass) computational complexity of HyperMIMO.

II. System model and problem formulation

We consider a communication channel with N_u single-antenna transmitters or users and a receiving base station with

This work was partially supported by Nvidia. Email: {nzilberstein, doost, ashu, segarra}@rice.edu, cdick@nvidia.com.

N_r antennas. The forward model for this MIMO system is given by

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

where $\mathbf{H} \in \mathbb{C}^{N_r \times N_u}$ is the channel matrix, $\mathbf{n} \sim \mathcal{CN}(\mathbf{0}, \sigma^2 \mathbf{I}_{N_r})$ is a vector of complex circular Gaussian noise, $\mathbf{x} \in \mathcal{X}^{N_u}$ is the vector of transmitted symbols, $\mathcal{X}$ is a finite set of constellation points, and $\mathbf{y} \in \mathbb{C}^{N_r}$ is the received vector. In this work, a quadrature amplitude modulation (QAM) is used and each symbol is normalized to unit average power. It is assumed that the constellation is the same for all transmitters and each symbol has the same probability of being chosen by the users N_u. Moreover, perfect channel state information (CSI) is assumed, which means that $\mathbf{H}$ and σ^2 are known at the receiver.[1] Under this setting, the MIMO detection problem can be defined as follows.

Problem 1: *Given perfect CSI and an observed* $\mathbf{y}$ *following* (1)*, find an estimate of* $\mathbf{x}$*.*

Given the stochastic nature of $\mathbf{n}$ in (1), a natural way of solving Problem 1 is to search for the $\mathbf{x}$ that maximizes the probability of observing our given $\mathbf{y}$. Unfortunately, such an ML detector boils down to solving the optimization problem

$$\hat{\mathbf{x}}_{\text{ML}} = \underset{\mathbf{x} \in \mathcal{X}^{N_u}}{\operatorname{argmin}} \; ||\mathbf{y} - \mathbf{H}\mathbf{x}||_2^2, \tag{2}$$

which is NP-hard due to the finite constellation constraint $\mathbf{x} \in \mathcal{X}^{N_u}$ [4], rendering $\hat{\mathbf{x}}_{\text{ML}}$ intractable in practical applications. Consequently, several schemes have been proposed in the last decades to provide efficient approximate solutions to Problem 1, as mentioned in Section I.

The classical body of work (ZF, MMSE, AMP) focuses on solving a single instance of Problem 1 for arbitrary $\mathbf{y}$ and $\mathbf{H}$, which must then be repeated to recompute the detection in successive communication instances. Given that in practice we are interested in solving several instances of Problem 1 across time, a learning-based body of work has gained traction in the past years [14]–[17]. In a nutshell, based on many tuples $(\mathbf{y}, \mathbf{H}, \mathbf{x})$, the idea is to learn a map – a function approximator – from the space of observations and CSI to the corresponding (approximate) transmitted symbols $\mathbf{x}$. In this way, when a new observation $\mathbf{y}$ is received (along with the CSI), $\mathbf{x}$ can be efficiently estimated using the learned map without the need for solving an optimization problem.

Having introduced this framework, we can provide a precise distinction between the families of learning-based methods that we denominated as channel-specific and channel-agnostic. Channel-specific methods like DetNet [15] (for the fixed channel case) and MMNet [14] learn *a different function for every* $\mathbf{H}$, i.e., they learn a function $\Phi_{\mathbf{H}} : \mathbb{C}^{N_r} \to \mathcal{X}^{N_u}$ such that $\Phi_{\mathbf{H}}(\mathbf{y})$ is a good solution to Problem 1 for a specific $\mathbf{H}$ of interest. On the other hand, channel-agnostic methods like HyperMIMO [16] and RE-MIMO [17] consider the CSI as input to their learnable functions, i.e., they look for $\Phi : \mathbb{C}^{N_r} \times \mathbb{C}^{N_r \times N_u} \to \mathcal{X}^{N_u}$ such that $\Phi(\mathbf{y}, \mathbf{H})$ is a good solution to Problem 1. Naturally, such a satisfactory

[1]To avoid notation overload, we adopt the convention that whenever we assume $\mathbf{H}$ to be known, σ^2 is also known.

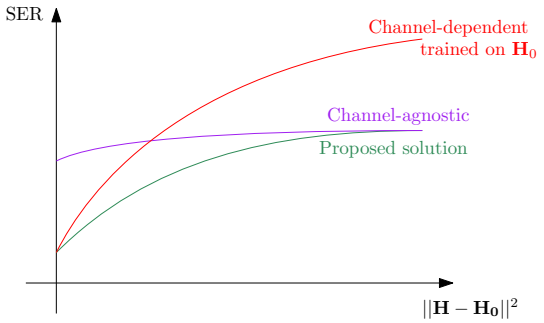


Fig. 1: Our proposed solution seeks to combine the best of both classes of methods by specializing to a channel (or set of channels) of interest while generalizing to a whole channel distribution.

Φ cannot be found for completely arbitrary $\mathbf{H}$ and, rather, channel-agnostic methods focus on channels drawn from some distribution of interest. Moreover, due to the specialized nature of $\Phi_{\mathbf{H}}$, channel-specific models tend to perform better for the particular channel $\mathbf{H}$ but their performance quickly degrades when a different channel is drawn.

In this setting, we are motivated by the following question: *Can we develop a generalizable channel-agnostic method that achieves performance comparable with channel-specific methods for a channel* $\mathbf{H}$ *(or set of channels* $\mathcal{H}$*) of interest?* In essence, we want to keep the best of both classes of methods by performing close to optimal on prespecified channels while generalizing to a whole distribution. Our motivating question is relevant in practice when, e.g., the channel fading varies smoothly with time as in Jakes model [19] (see Section IV for more details). In such a case, we want our learning scheme to perform especially well around the current channel while generalizing satisfactorily to avoid the need for immediate retraining.

At a high level, given some metric in the space of channels, channel-specific solutions yield lower symbol error rate (SER) close to the channel $\mathbf{H}_0$ for which they were trained whereas channel-agnostic methods work better when channels further away from $\mathbf{H}_0$ are drawn; see Fig. 1. Intuitively, we want to derive a method that attains the behavior illustrated in green in Fig. 1. Hence, one can think of our sought solution as a *robust* version of a channel-specific method that gracefully degrades into a channel-agnostic method. Alternatively, one can see the envisioned solution as a channel-agnostic method that has been specially tuned to overperform on a subset of channels of interest. Either way, we propose to achieve this through the use of hypernetworks whose training is regularized by the solutions of channel-specific methods, as we detail next.

III. Hypernetworks with Learned Regularizers

In Section III-A we introduce the notion of a hypernetwork and its use in machine learning whereas in Section III-B we detail how we incorporate hypernetworks in our solution to Problem 1.

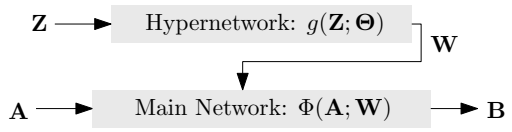


Fig. 2: The general scheme of a hypernetwork. The hypernetwork $g(\cdot;\mathbf{\Theta})$ takes the input $\mathbf{Z}$ and generates the weights $\mathbf{W}$, which are fed into the main network. Then, the main network $\Phi(\cdot;\mathbf{W})$ takes as input $\mathbf{A}$ and returns the output $\mathbf{B}$.

A. Hypernetworks in machine learning and MIMO detection

Hypernetworks are neural networks that have as output the weights of another target (or main) neural network, which performs the learning task [18]; see Fig. 2. More precisely, the interpretation of our main network is that of a classical neural network that learns a parametric function $\Phi(\cdot;\mathbf{W}) : \mathcal{A} \to \mathcal{B}$ from input data $\mathbf{A}$ to some desired target $\mathbf{B}$, being $\mathbf{W} \in \mathcal{W}$ the learnable parameters of this neural network. The goal of the hypernetwork, on the other hand, is to learn a parametric function $g(\cdot;\mathbf{\Theta}) : \mathcal{Z} \to \mathcal{W}$ from the (possibly different) input $\mathbf{Z}$ into the space of parameters $\mathcal{W}$ of the main network. In this way, we are not fixing the parameters $\mathbf{W}$ of our main network but rather making these a function of the input $\mathbf{Z}$, effectively improving the generalizability of Φ. Thus, given inputs $(\mathbf{A}, \mathbf{Z})$, the output of the main network is given by $\Phi(\mathbf{A}; g(\mathbf{Z};\mathbf{\Theta})) \in \mathcal{B}$. It should be noted that only the parameters $\mathbf{\Theta}$ of the hypernetwork need to be learned during training.

The notion of a hypernetwork has been used in several contexts such as object recognition [20] and generation of 3-D point clouds [21]. For example, hypernetworks have been used for 3D shape reconstruction [22] and to learn shared representations of images [23]. In the specific context of MIMO detection, the use of hypernetworks has been already proposed in [16] applied to the MMNet. As the MMNet depends on a particular channel realization, the hypernetwork enables the generalization to a whole distribution of channels.

B. Learning hypernetwork regularizers

Having formally introduced the concept of a hypernetwork, we can now revisit the HyperMIMO [16], which exactly follows the framework in Fig. 2. In particular, we have that the main network – which takes the form of an MMNet [14] – has as input the observation $\mathbf{y}$ and the CSI, i.e., $\mathbf{A} = \{\mathbf{y}, \mathbf{H}\}$. Moreover, the hypernetwork takes the CSI as input $\mathbf{Z} = \mathbf{H}$ and generates the weights for the multiple layers of the main MMNet. Then, the MMNet generates the estimate of the transmitted symbols ($\mathbf{B} = \hat{\mathbf{x}}$). The weights of the hypernetwork are trained to minimize a loss that compares $\hat{\mathbf{x}}$ with the true transmitted symbols $\mathbf{x}$. This flow is depicted by blue arrows in Fig. 3.

We expand the described training procedure to attain a solution to Problem 1 that captures the desirable behavior in Fig. 1. First, we determine the channel or set of channels $\mathcal{H} = \{\mathbf{H}_1, \mathbf{H}_2, \cdots, \mathbf{H}_N\}$ on which we want our solution to achieve especially high detection performance. This choice will be guided by the nature of the system where we anticipate

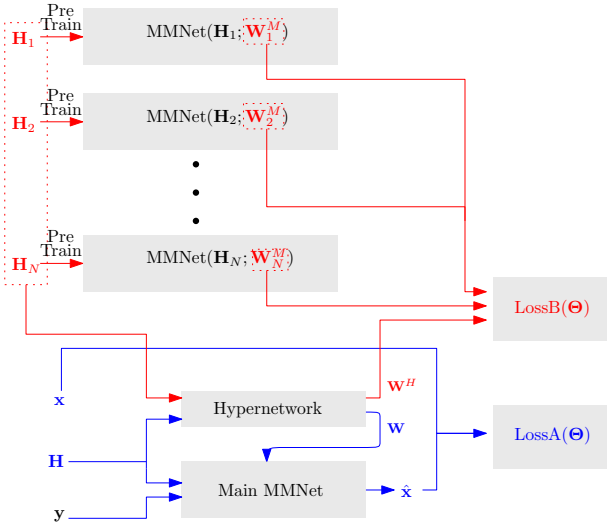


Fig. 3: Scheme of our proposed training architecture. In addition to the more classical hypernetwork training (blue arrows), we propose a regularization term that depends on several pretrained main (MMNet) networks, as depicted by the red arrows.

that our detector will be deployed. Given many realizations $(\mathbf{y}, \mathbf{H}_i, \mathbf{x})$ for the channels $\mathbf{H}_i \in \mathcal{H}$, we train a collection of MMNets $\Phi_{\mathbf{H}_i}(\cdot;\mathbf{W}_i^{\mathrm{M}})$, one per channel $\mathbf{H}_i$. Notice that, given the channel-specific nature of $\Phi_{\mathbf{H}_i}$, the learned weights $\mathbf{W}_i^{\mathrm{M}}$ entail good detection performance for the channel $\mathbf{H}_i$. We use these pretrained weights as regularizers during the training of our hypernetwork; see red arrows in Fig. 3. To be more precise, if we denote by $\mathbf{W}_i^{\mathrm{H}} = g(\mathbf{H}_i;\mathbf{\Theta})$ the weights output by the hypernetwork when we input channel $\mathbf{H}_i$, we define our regularized loss as

$$\mathcal{L}(\mathbf{\Theta}) = \underbrace{\mathbb{E}_{\mathbf{H},\mathbf{x},\mathbf{n}}[||\mathbf{x} - \hat{\mathbf{x}}||_2^2]}_{\text{LossA}(\mathbf{\Theta})} + \beta \underbrace{\sum_{\mathbf{H}_i \in \mathcal{H}} ||\mathbf{W}_i^{\mathrm{M}} - \mathbf{W}_i^{\mathrm{H}}||_1}_{\text{LossB}(\mathbf{\Theta})}. \quad (3)$$

The first term in (3) computes a classical mean square error between the true symbols and the estimated symbols, where the expected value is taken over the channel, input, and noise distributions of interest. The channel distribution used here is the one for which we want our channel-agnostic method to generalize. The second term penalizes the distance between the parameters in the pretrained MMNets and those generated by the hypernetwork when it is fed with channels from $\mathcal{H}$. We measure this discrepancy using an ℓ_1 norm to promote a sparse difference between $\mathbf{W}_i^{\mathrm{M}}$ and $\mathbf{W}_i^{\mathrm{H}}$. This means that the weights $\mathbf{W}_i^{\mathrm{H}}$ generated by the hypernetwork tend to coincide with $\mathbf{W}_i^{\mathrm{M}}$ for a subset of the entries. The relative weight β captures the importance of performing well within the set $\mathcal{H}$. Indeed, when $\beta = 0$ our proposed method boils down to HyperMIMO and completely ignores the prespecified channels $\mathcal{H}$. On the other hand, for $\beta \to \infty$ (and assuming that the hypernetwork is sufficiently expressive) our method should mimic the behavior of MMNet on $\mathcal{H}$ but quickly degrade for generic channels. By selecting an intermediate value of β, we can realize the

desired behavior in Fig. 1, as we demonstrate in Section IV. The model is trained by minimizing the loss in (3) with respect to the hypernetworks parameters $\mathbf{\Theta}$ through stochastic gradient descent.

Before presenting the numerical experiments, two remarks are in order. First, although for concreteness we present our scheme in Fig. 3 for the case where the main network is an MMNet, the same framework can be applied for *any generic channel-specific method* taking the role of our main network. Second, although our proposed scheme incurs an additional training load in comparison with a vanilla hypernetwork, once trained their computational complexities for detection are exactly the same. We will refer to our proposed methodology particularized for the MMNet as HyperMIMO with learned regularizers, or HyperMIMO-LR for short.

IV. Numerical Experiments

In this section we present the results of our proposed method[2]. We start by presenting the channel model, simulation setup, and neural network training process. Then, we present the experimental results and derive insights into the performance of the HyperMIMO-LR.

A. Channel model

The channel model is generated following the Jakes model [19]. We express the small-scale Rayleigh fading component as a first-order complex Gauss-Markov process

$$\mathbf{H}_t = \rho \mathbf{H}_{t-1} + \sqrt{1-\rho^2}\mathbf{E}_t, \tag{4}$$

where $\mathbf{E}_t$ is $= \{\mathbf{e}_{t,i}\}_{i=1}^{N_u} \sim \mathcal{CN}(0, \mathbf{I}_{N_r})$ and $\{\mathbf{E}_0, \mathbf{E}_1, \cdots\}$ are independent and identically distributed random variables. The initial matrix $\mathbf{H}_0$ is generated following the Kronecker correlated channel model

$$\mathbf{H}_0 = \mathbf{R}_r^{1/2}\mathbf{H}_e\mathbf{R}_u^{1/2}, \tag{5}$$

where each column of $\mathbf{H}_e$ is $\{\mathbf{h}_{i,e}\}_{i=1}^{N_u} \sim \mathcal{CN}(0, \mathbf{I}_{N_r})$ and $\mathbf{R}_r$ and $\mathbf{R}_u$ are the spatial correlation matrices at the receiver and transmitter, respectively, generated according to the exponential correlation matrix model with a correlation coefficient ρ_k [24]. In our model, the signal-to-noise ratio (SNR) is given by

$$\text{SNR} = \frac{\mathbb{E}[||\mathbf{Hx}||^2]}{\mathbb{E}[||\mathbf{n}||^2]} = \frac{N_u}{\sigma^2 N_r}. \tag{6}$$

For the experiments, SNRs between 5dB and 10dB are considered.

B. Implementation

Our simulation environment includes a base station with $N_r = 4$ receiver antennas and $N_u = 2$ transmitting single-antenna users. We consider 4-QAM modulation. The architecture of the hypernetwork is composed of three dense layers: the first layer has the same number of units as the input, the second one has 100 units and the third one has the number of units matching the number of parameter that MMNet requires. For the MMNet, we use 6 layers. The activation function for all layers in the hypernetwork is an ELU function; the reason why using an ELU and not a ReLU resides on the nature of the $\mathbf{\Theta}$ parameters, which can take negative values.

Training. We use a batch size of 100 channel matrices generated from (5). The training is performed using ADAM optimizer [25] with a reduce plateau scheduler: we compute the loss every 500 iterations and when the loss stopped improving, the learning rate is reduced by a factor of 0.9. We train for $50{,}000$ iterations[3]. The value of β in (3) was set to 1. For the regularizer, we generate 140 different sequences of length $t = 4$ following (4) with $\rho = 0.98$ and starting from the same initial matrix $\mathbf{H}_0$ from (5) with $\rho_k = 0.6$. In total, we use $N = 561$ pre-trained MMNets. The matrices are chosen randomly, although we tried a few heuristics to select representative samples. However, none of them consistently outperformed the random choise.

C. Simulation results

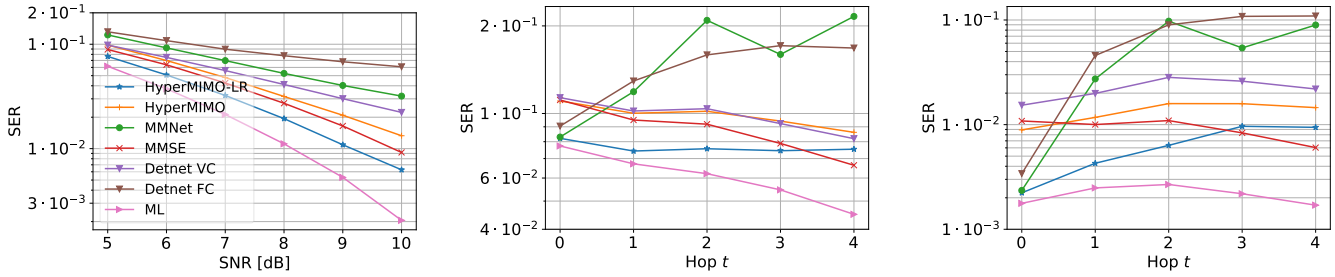
For testing the performance of the detectors, we generate a test set of 100 sequences of the same length $t = 4$ from the same model in (4), also starting from $\mathbf{H}_0$.

We compare the SER achieved by HyperMIMO-LR with respect to the following methods: HyperMIMO, MMNet, DetNet with fixed and varying channel, MMSE and ML (using the Gurobi solver [26]). The comparisons are shown in Fig. 4a. The figure reveals that the performance of HyperMIMO-LR is closest to the optimal ML, and outperforms all the other methods, in particular both HyperMIMO and MMNet. It is particularly interesting to observe that while HyperMIMO-LR consistently outperforms the classical MMSE detector, HyperMIMO has a worse performance than MMSE. This is because the performance of the HyperMIMO decreases significantly when it is tested in perturbed versions of a channel from the distribution, while HyperMIMO-LR performs robustly in those unseen channels.

The performance of the detector as a function of the hops t is represented in Figs. 4b and 4c, for an SNR of 5dB and 10dB, respectively. We use the same test set as in the previous experiment, and for each time t we average the performance of the proposed detector over the 100 sample channels. In both cases, we observe equal SERs at $t = 0$ (initial matrix $\mathbf{H}_0$) for both MMNet and HyperMIMO-LR. This is expected because the parameters of both architectures are similar due to the regularizer, and hence the performance of both has to be the same at the initial hop. We also see that the performance of DetNet-FC at the initial hop is close to HyperMIMO-LR and MMNet, but the performance for both MMNet and DetNet-FC quickly degrades as we increase t. Moreover, HyperMIMO follows a similar trend as HyperMIMO-LR, meaning that its performance does not drop severely with t but nonetheless it is inferior to HyperMIMO-LR. Overall, this behaviour is what we expected from our motivation defined in Fig. 1. Lastly, we see that MMSE performs relatively better for later hops t. This can be explained by looking at the way that on the way that the sequence is constructed following the

[2]Code to replicate the numerical experiments can be found at https://github.com/nzilberstein/HyperMIMO_LR.git.

[3]For HyperMIMO we followed the same scheme as in [16], changing only the lower limit to 10^{-6}.



(a) (b) (c)

Fig. 4: (a) SER as a function of SNR for different detection methods evaluated in a set of channel sequences generated via (4). (b and c) SER as a function of time for SNR = 5dB and SNR = 10dB, respectively (the same legend as Fig. 4a holds).

Jakes model in (4): the initial matrix $\mathbf{H}_0$ is a sample from the Kronecker channel model, while as we move forward, the weight of the term $\mathbf{e}_t$ dominates in the equation. In this way, the channel becomes closer to being i.i.d. Gaussian for larger values of t. For this particular case, given a fixed SNR, MMSE achieves better performance because it may be the case where the correlation is such that more noise power falls in signal space, so the colored noise case tends to be worse, while the learning methods (as they were trained on the Kronecker channel model) tend to perform worse for i.i.d. Gaussian channel because you move out from the original distribution.

V. Conclusions

We proposed a general deep learning based solution for MIMO detection that achieves a high performance for perturbations of a prespecified set of channels while generalizing to the whole distribution. This was done by regularizing the training of the hypernetwork to a deep learning-based detector with solutions for a set of specific channels using that detector. We evaluated this general architecture with an implementation that uses HyperMIMO, a hypernetwork-based solution that incorporates MMNet as its deep learning-based MIMO detector. We demonstrated that our implementation, named HyperMIMO-LR, generalizes well to the whole distribution of channels and outperforms HyperMIMO. Future work include extending to higher-order systems as well as higher-order modulation.

References

[1] Shaoshi Yang and Lajos Hanzo, "Fifty years of MIMO detection: The road to large-scale MIMOs," *IEEE Commun. Surveys Tut.*, vol. 17, no. 4, pp. 1941–1988, 2015.

[2] Arogyaswami J. Paulraj, Dhananjay A. Gore, Rohit U. Nabar, and Helmut Bolcskei, "An overview of MIMO communications - a key to gigabit wireless," *Proc. IEEE*, vol. 92, no. 2, pp. 198–218, 2004.

[3] Khaled B. Letaief, Wei Chen, Yuanming Shi, Jun Zhang, and Ying-Jun Angela Zhang, "The roadmap to 6G: AI empowered wireless networks," *IEEE Commun. Mag.*, vol. 57, no. 8, pp. 84–90, 2019.

[4] Alberto Del Pia, Santanu S. Dey, and M. Molinaro, "Mixed-integer quadratic programming is in NP," *Mathematical Programming*, vol. 162, pp. 225–240, 2017.

[5] John G. Proakis, *Digital Communications 5th Edition*, McGraw Hill, 2007.

[6] Ananthanarayanan Chockalingam and Balaji Sundar Rajan, *Large MIMO Systems*, Cambridge University Press, 2014.

[7] Charles Jeon, Ramina Ghods, Arian Maleki, and Christoph Studer, "Optimality of large MIMO detection via approximate message passing," in *IEEE Intl. Symp. on Info. Theory (ISIT)*, 2015, pp. 1227–1231.

[8] Mark Eisen and Alejandro Ribeiro, "Optimal wireless resource allocation with random edge graph neural networks," *IEEE Trans. Signal Process.*, vol. 68, pp. 2977–2991, 2020.

[9] Arindam Chowdhury, Gunjan Verma, Chirag Rao, Ananthram Swami, and Santiago Segarra, "Unfolding WMMSE using graph neural networks for efficient power allocation," *IEEE Trans. Wireless Commun.*, vol. 20, no. 9, pp. 6004–6017, 2021.

[10] Arindam Chowdhury, Gunjan Verma, Chirag Rao, Ananthram Swami, and Santiago Segarra, "ML-aided power allocation for tactical MIMO," *arXiv preprint arXiv:2109.06992*, 2021.

[11] Zhongyuan Zhao, Gunjan Verma, Chirag Rao, Ananthram Swami, and Santiago Segarra, "Distributed scheduling using graph neural networks," in *IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP)*, 2021, pp. 4720–4724.

[12] Zhongyuan Zhao, Gunjan Verma, Chirag Rao, Ananthram Swami, and Santiago Segarra, "Link scheduling using graph neural networks," *arXiv preprint arXiv:2109.05536*, 2021.

[13] Abhishek Kumar, Gunjan Verma, Chirag Rao, Ananthram Swami, and Santiago Segarra, "Adaptive contention window design using deep Q-learning," in *IEEE Intl. Conf. Acoust., Speech and Signal Process. (ICASSP)*, 2021, pp. 4950–4954.

[14] Mehrdad Khani, Mohammad Alizadeh, Jakob Hoydis, and Phil Fleming, "Adaptive neural signal detection for massive MIMO," *IEEE Trans. Wireless Commun.*, vol. 19, no. 8, pp. 5635–5648, 2020.

[15] Neev Samuel, Tzvi Diskin, and Ami Wiesel, "Deep MIMO detection," in *IEEE Intl. Workshop on Signal Process. Adv. in Wireless Commun. (SPAWC)*, 2017, pp. 1–5.

[16] Mathieu Goutay, Fayçal Ait Aoudia, and Jakob Hoydis, "Deep hypernetwork-based MIMO detection," in *IEEE Intl. Workshop on Signal Process. Adv. in Wireless Commun. (SPAWC)*, 2020, pp. 1–5.

[17] Kumar Pratik, Bhaskar D. Rao, and Max Welling, "RE-MIMO: Recurrent and permutation equivariant neural MIMO detection," *IEEE Trans. Signal Process.*, vol. 69, pp. 459–473, 2021.

[18] David Ha, Andrew Dai, and Quoc V. Le, "Hypernetworks," *arXiv preprint arXiv:1609.09106*, 2016.

[19] Yasar Sinan Nasir and Dongning Guo, "Multi-agent deep reinforcement learning for dynamic power allocation in wireless networks," *IEEE J. Sel. Areas Commun.*, vol. 37, no. 10, pp. 2239–2250, 2019.

[20] Luca Bertinetto, João F. Henriques, Jack Valmadre, Philip H. S. Torr, and Andrea Vedaldi, "Learning feed-forward one-shot learners," in *Advances in Neural Inf. Process. Syst. (NIPS)*, 2016, p. 523–531.

[21] Przemyslaw Spurek, Sebastian Winczowski, Jacek Tabor, Maciej Zamorski, Maciej Zieba, and Tomasz Trzcinski, "Hypernetwork approach to generating point clouds," *arXiv preprint arXiv:2003.00802*, 2020.

[22] Gidi Littwin and Lior Wolf, "Deep meta functionals for shape representation," *arXiv preprint arXiv:1908.06277*, 2019.

[23] Vincent Sitzmann, Julien N. P. Martel, Alexander W. Bergman, David B. Lindell, and Gordon Wetzstein, "Implicit neural representations with periodic activation functions," *arXiv preprint arXiv:2006.09661*, 2020.

[24] Sergey L. Loyka, "Channel capacity of MIMO architecture using the exponential correlation matrix," *IEEE Commun. Lett.*, vol. 5, no. 9, pp. 369–371, 2001.

[25] Diederik P. Kingma and Jimmy Ba, "Adam: A method for stochastic optimization," *arXiv preprint arXiv:1412.6980*, 2017.

[26] Gurobi Optimization LLC., "Gurobi optimizer reference manual," 2021.