DeEQ: Deep Equalization for Large MIMO Systems

Matthias Hummert, Dirk Wübben and Armin Dekorsy
Department of Communications Engineering
University of Bremen, 28359 Bremen, Germany
Email: {hummert, wuebben, dekorsy}@ant.uni-bremen.de

***Abstract*—Multiple Input Multiple Output (MIMO) and massive MIMO (mMIMO) are key-enabling technologies for 4G and 5G communications systems. mMIMO uses a high number of antennas, where the number of antennas at the base station exceeds in general the number of antennas in the mobiles. For uncorrelated channels, linear equalizers already achieve promising performance in the uplink due to the channel hardening effect. In contrast, we will focus on large symmetrical MIMO systems in this paper, where many antennas are employed at the transmitter and the receiver side resulting in a more challenging task for the receiver. Traditionally, receiver algorithms have been derived based on models for the communications system. Recently, machine learning approaches have been proposed where the design is data driven. In order to overcome the drawbacks of model-based and pure data driven approaches, hybrid approaches combining the benefits of both worlds have emerged. In this paper, we present the novel hybrid approach entitled Deep Equalization (DeEQ) based on model knowledge and a neural network like structure. As demonstrated by simulation results, this novel approach achieves very good performance with the advantage of only a very low error floor.**

***Index Terms*—MIMO, Equalization, Machine Learning, Deep Learning, Neural Networks**

I. Introduction

Multiple Input Multiple Output (MIMO) systems and in particular massive MIMO (mMIMO) become more and more popular due to the demand for higher data rates and more spectral efficiency. For mMIMO the number of antennas at the base station is usually higher than the number of antennas of the mobiles.

Due to the channel hardening effect, linear equalizers already achieve promising performance in the uplink for uncorrelated channel matrices [1]. In this paper we will focus on large symmetrical MIMO systems with many transmit and receive antennas, where the number of transmit antennas is equal or about the number of receive antennas. Linear equalizers as commonly applied for mMIMO will not achieve convincing error rate performance, but the computational complexity for calculating the filter matrix based on the Zero-Forcing (ZF) criterion or Minium Mean Square Error (MMSE) criterion grows cubically with the matrix dimension. Algorithms like the Ordered Succesive Interfernce Cancellation (OSIC) [2] achieve better performance while still being too complex. On the other hand algorithms based on the maximum likelihood (ML) criterion, e.g., the sphere detector (SD) achieve optimal performance at the cost of being far too complex to use in large systems. In order to achieve good performance with reasonable complexity, several new learning based receiver concepts for MIMO systems have been proposed recently.

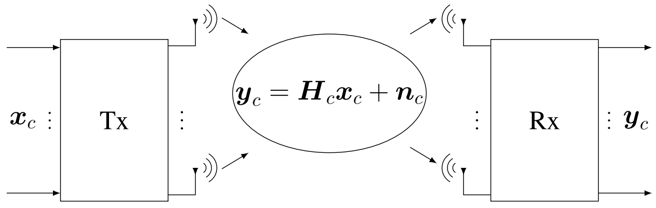
In general, algorithms can be developed using a model-based approach, a data driven approach or a combination of both which is commonly known as hybrid approach [3]. Model-based approaches can have the problem of a model bias since not every model reflets all the effects of the real system precisely. Pure data driven approaches on the other hand have to learn the complete underlying processes purely by data that describes the system, which can fail if the structure of the problem is too complex. The idea of combing these two approaches is therefore promising, as model knowledge is introduced into the training procedure in order to learn complex environments.

To give an example for a classical model-based MIMO approach, the Approxmimate Message Passing (AMP) algorithm [4], [5] is used. This algorithm has been derived under the assumption of infinitely many antennas and therefore suffers from a performance degradation for small system dimensions. In contrast, a pure data driven approach, using deep neural networks, has to learn the underlying MIMO equalizer structure on its own. Therefore, this approach will not be able to generalize for every possible channel matrix and has to be retrained for every channel matrix indivudually [6]. As a reason, this approach requires a huge training overhead and is impractical. Due to the mentioned drawbacks of either these approaches, hybrid approaches have been developed by combining model knowledge and the power of deep learning [6]–[8]. The two examples considered here are Detection Network (DetNet) [6], [7] and MMNet [8]. DetNet is a heuristic approach that incorporates knowledge of the MIMO system model into a neural network like structure and requires many learned weights. MMNet on the other hand is more strict in preserving the incorporated model structure and uses less weights as a consequence. These hybrid approaches seem to be a promising approach as they show excellent performance while being moderatly complex to implement.

In this paper, a new hybrid approach named Deep Equalization (DeEQ) is proposed which incorporates model knowledge into a neural network like structure in order to build an efficient equalizer which shows excellent performance for large MIMO systems with a moderate complexity. The remainder is structured as follows: after discussing the system model in Section II, in Section III the hybrid approaches DetNet and MMNet are revised and the the novel DeEq is proposed. In Section IV the performance is evaluated and conclusions are provided in Section V.

II. System model

A. The Equalization Problem



$\boldsymbol{x}_c$
Tx
$\boldsymbol{y}_c = \boldsymbol{H}_c\boldsymbol{x}_c + \boldsymbol{n}_c$
Rx
$\boldsymbol{y}_c$

Fig. 1. MIMO system model with N_t transmit and N_r receive antennas

We consider the MIMO system in Fig. 1 with N_t transmit and N_r receive antennas where the complex receive signal $\boldsymbol{y}_c \in \mathbb{C}^{N_r}$ is given by

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

with the complex transmit symbols $\boldsymbol{x}_c \in \mathcal{A}_c^{N_t}$, the complex channel matrix $\boldsymbol{H}_c \in \mathbb{C}^{N_r \times N_t}$, and complex white Gaussian noise $\boldsymbol{n}_c \in \mathbb{C}^{N_r}$. The channel matrix $\boldsymbol{H}$ is a realization of a stochastic process, where the real and imaginary part are iid Gaussian distributed. $\mathcal{A}_c$ defines the normalized QAM symbol alphabet and in this paper we restrict ourselves to 4-QAM $\mathcal{A}_c = \left\{\pm\frac{1}{\sqrt{2}} \pm j\frac{1}{\sqrt{2}}\right\}$. In order to enable the application of common deep learning libraries, we will use the equivalent real valued system model

$$\boldsymbol{y} = \boldsymbol{H}\boldsymbol{x} + \boldsymbol{n} \tag{2}$$

of (1). The equivalent real valued vectors and matrices are given by

$$\begin{aligned} \boldsymbol{y} &= \begin{bmatrix} \Re(\boldsymbol{y}_c) \\ \Im(\boldsymbol{y}_c) \end{bmatrix}, \boldsymbol{x} = \begin{bmatrix} \Re(\boldsymbol{x}_c) \\ \Im(\boldsymbol{x}_c) \end{bmatrix}, \boldsymbol{n} = \begin{bmatrix} \Re(\boldsymbol{n}_c) \\ \Im(\boldsymbol{n}_c) \end{bmatrix} \\ \boldsymbol{H} &= \begin{bmatrix} \Re(\boldsymbol{H}_c) & -\Im(\boldsymbol{H}_c) \\ \Im(\boldsymbol{H}_c) & \Re(\boldsymbol{H}_c) \end{bmatrix} \end{aligned} \tag{3}$$

with $\Re(\boldsymbol{x}_c)$ and $\Im(\boldsymbol{x}_c)$ representing the real and imaginary part of $\boldsymbol{x}_c$, respectively. The dimensions double by this procedure, e.g., for the real valued receive vector follows $\boldsymbol{y} \in \mathbb{R}^{2 \cdot N_r}$. Furthermore, we define $\mathcal{A} = \{\pm 1/\sqrt{2}\}$ to be the equivalent real valued symbol alphabet, i.e., BPSK per dimension. In this paper, we restrict ourselves to uncoded systems.

The ML criterion to estimate the transmit vector $\boldsymbol{x}$ is given by

$$\hat{\boldsymbol{x}} = \underset{\boldsymbol{x} \in \mathcal{A}^{2N_t}}{\arg\min} \parallel \boldsymbol{y} - \boldsymbol{H}\boldsymbol{x} \parallel_2^2 \ . \tag{4}$$

It requires exhaustive search over all possible transmit signal vectors $\boldsymbol{x}$, which grows with the cardinality of $\mathcal{A}$ and exponentially with the number of transmit antennas $2N_t$ and becomes too computationally demanding for large system dimensions. An efficient algorithm to solve this problem is the SD which is still to complex for large system dimensions. Due to this fact, several algorithms have been proposed to estimate the transmit signal $\boldsymbol{x}$ in an efficient way while achieving good performance. As the hybrid approaches considered in section III are based on linear equalization, we will introduce them subsequently.

B. Linear MIMO equalizer

In this subsection we revise the linear equalizer (LE) based on the ZF and the MMSE criterion. For the ZF criterion, we relax the demand of $\boldsymbol{x} \in \mathcal{A}^{2N_t}$ in (4) to $\boldsymbol{x} \in \mathbb{R}^{2N_t}$ leading to the estimate

$$\tilde{\boldsymbol{x}}_{\text{ZF}} = \underset{\boldsymbol{x} \in \mathbb{R}^{2N_t}}{\arg\min} \parallel \boldsymbol{y} - \boldsymbol{H}\boldsymbol{x} \parallel_2^2 \ . \tag{5}$$

The estimate vector $\tilde{\boldsymbol{x}}_{\text{ZF}}$ is continuous and by element-wise quantization to the next symbol in $\mathcal{A}$, the estimate $\hat{x}_i = \mathcal{Q}_{\mathcal{A}}\{\tilde{x}_i\}$ for the ith transmit signal is calculated. To get an estimate for (5), we take the derivative of the argument with respect to $\boldsymbol{x}$ and set it to $\boldsymbol{0}$, leading to

$$\begin{aligned} \frac{\partial}{\partial \boldsymbol{x}} \parallel \boldsymbol{y} - \boldsymbol{H}\boldsymbol{x} \parallel_2^2 &= \frac{\partial}{\partial \boldsymbol{x}} \left[(\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x})^T (\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x})\right] \\ &= 2\boldsymbol{H}^T\boldsymbol{H}\boldsymbol{x} - 2\boldsymbol{H}^T\boldsymbol{y} \overset{!}{=} \boldsymbol{0} \ . \end{aligned} \tag{6}$$

Thus, the soft estimate $\tilde{\boldsymbol{x}}_{\text{ZF}}$ for the transmit vector $\boldsymbol{x}$ is given by

$$\tilde{\boldsymbol{x}}_{\text{ZF}} = \left(\boldsymbol{H}^T\boldsymbol{H}\right)^{-1} \boldsymbol{H}^T\boldsymbol{y} = \boldsymbol{H}^+\boldsymbol{y} \ . \tag{7}$$

$\tilde{\boldsymbol{x}}_{\text{ZF}}$ is a soft estimate for the transmit vector $\boldsymbol{x}$ and $\boldsymbol{H}^+$ is the Moore-Penrose pseudoinverse.

The corresponding MMSE solution is given by

$$\tilde{\boldsymbol{x}}_{\text{MMSE}} = \left(\boldsymbol{H}^T\boldsymbol{H} + \sigma_n^2\boldsymbol{I}_{N_t}\right)^{-1} \boldsymbol{H}^T\boldsymbol{y} \ . \tag{8}$$

As the calculation of the inverses in (7) and (8) are costly, especially for higher matrix dimensions, an iterative approach to calculate the ZF solution has been proposed using a Gradient Descent approach [9]. Here, the estimate $\tilde{\boldsymbol{x}}^{k+1}$ in iteration $k+1$ is given by

$$\tilde{\boldsymbol{x}}^{k+1} = \tilde{\boldsymbol{x}}^k - \delta\left(\boldsymbol{H}^T\boldsymbol{H}\tilde{\boldsymbol{x}}^k - \boldsymbol{H}^T\boldsymbol{y}\right) \tag{9}$$

with the non-negative step size δ and the initialization $\tilde{\boldsymbol{x}}^0 = \boldsymbol{0}$. For sufficiently large number of iterations, the estimate $\tilde{\boldsymbol{x}}^{k+1}$ will converge to the closed form solution $\tilde{\boldsymbol{x}}_{\text{ZF}}$ in (7). The main advantage of this approach is the avoidance of the inverse of the closed form solution (7).

Subsequently, (9) will be used as *base equation* to derive the novel DeEQ algorithm. Similarly, this base equation has been used to derive the DetNet [6], [7] and MMNet [8] approaches.

III. Hybrid approaches

In general iterative algorithms can be interpreted as a layer k in a neural network. This layer k can consist of several steps to generate its output, depending on the iterative algorithm structure. Extending this layer k with trainable weights, biases and nonlinearities we arrive at a neural network structure. This general procedure is now applied to the base equation (9) and we introduce $\boldsymbol{z}^k$ as an intermediate variable for it. $\boldsymbol{z}^k$ is hence the first step in the layer k of the resulting neural network. The following steps thereafter add weights, biases and nonlinearities to end up at the hybrid approaches presented here. The open question remains, where to introduce weights and biases. As this question cannot yet be answered with a theoretical foundation, DetNet, MMNet and our approach are based on heuristics and the usage of model knowledge. Subsequently, we note the total number of layers by K.

A. Detection Network (DetNet)

DetNet [6], [7] is a hybrid approach based on the combination of deep neural networks and the model knowledge of the MIMO system in (2). The derivation of DetNet follows [6] and is heuristic, but directly incorporates model knowledge into its update equations. With the help of the base equation (9) and the intermediate variable $\boldsymbol{z}^k$ the update equations read

$$\boldsymbol{z}^k = \tilde{\boldsymbol{x}}^k + \delta_1^k \boldsymbol{H}^T \boldsymbol{y} - \delta_2^k \boldsymbol{H}^T \boldsymbol{H} \tilde{\boldsymbol{x}}^k \tag{10a}$$

$$\boldsymbol{u}^k = \left[\left(\boldsymbol{W}_1^k \begin{pmatrix} \boldsymbol{z}^k \\ \boldsymbol{v}^k \end{pmatrix} + \boldsymbol{b}_1^k \right) \right]_+ \tag{10b}$$

$$\boldsymbol{v}^{k+1} = \boldsymbol{W}_2^k \boldsymbol{u}^k + \boldsymbol{b}_2^k \tag{10c}$$

$$\tilde{\boldsymbol{x}}^{k+1} = \boldsymbol{W}_3^k \boldsymbol{u}^k + \boldsymbol{b}_3^k \tag{10d}$$

where $[\cdot]_+ = \max\{\cdot, 0\}$ is a representation for the well known ReLu function and $\boldsymbol{u}^k \in \mathbb{R}^{8\cdot 2N_t}$ and $\boldsymbol{v}^k \in \mathbb{R}^{2\cdot 2N_t}$ are intermediate variables with chosen dimensions. The design of the dimensions of $\boldsymbol{u}^k$ and $\boldsymbol{v}^k$ is heuristic. The calculation of the intermediate variable $\boldsymbol{z}^k$ in (10a) follows the base equation (9), but introduces two scalar trainable variables δ_1^k and δ_2^k which can be interpreted as a split step size. The intermediate variable $\boldsymbol{z}^k$ is then concatenated with the variable $\boldsymbol{v}^k$ and mapped to a higher dimensional space via $\boldsymbol{W}_1^k \in \mathbb{R}^{8\cdot 2N_t \times 3\cdot 2N_t}$ and the bias $\boldsymbol{b}^k \in \mathbb{R}^{8\cdot 2N_t}$ followed by the ReLu function. The intermediate step $\boldsymbol{v}^{k+1}$ is calculated via the application of a weight matrix $\boldsymbol{W}_2^k \in \mathbb{R}^{2\cdot 2N_t \times 8\cdot 2N_t}$ and a bias $\boldsymbol{b}_2^k \in \mathbb{R}^{2\cdot 2N_t}$. Finally, the estimate for $\tilde{\boldsymbol{x}}^{k+1}$ is given by the last step with the variable $\boldsymbol{u}^k$, the weight matrix $\boldsymbol{W}_3^k \in \mathbb{R}^{2N_t \times 8\cdot 2N_t}$ and the bias $\boldsymbol{b}_3^k \in \mathbb{R}^{2N_t}$, $\delta_1^k \in \mathbb{R}$. The trainable parameters are $\boldsymbol{\theta}^k = \{\boldsymbol{W}_1^k, \boldsymbol{W}_2^k, \boldsymbol{W}_3^k, \boldsymbol{W}_4^k, \boldsymbol{b}_1^k, \boldsymbol{b}_2^k, \boldsymbol{b}_3^k, \delta_1^k, \delta_2^k\}$ and the total number of trainable weights are $192N_t^2 + 22N_t + 2$ in each layer k. DetNet is parametrized with a total number of $K = 3 \cdot 2N_t$ layers and trained offline and used for inference only afterwards.

This more heuristic approach has proven to perform well under iid channels. The disadvantages of DetNet are twofold. On the one hand, due to its heuristic nature its hard to understand how and why it works and on the other hand, the complexity grows rapidly as it uses a large number of layers and weight matrices. Our simulation results of DetNet are based on the source code provided by the authors [10].

B. MMNet

In [8], two low complexity hybrid approaches are presented. An MMNet$_{\text{iid}}$ is discussed which is especially designed for iid channels and has two trainable parameters per layer. MMNet$_{\text{iid}}$ is trained offline and used for inference only afterwards. In addition, MMNet is proposed which is especially designed to cope with correlated channel matrices as it adapts it parameters online for specific channels. MMNet can hence outperform existing algorithms like DetNet while suprisingly still achieving a lower complexitiy than DetNet. In this paper only MMNet$_{\text{iid}}$ is discussed as we restrict the analysis to iid channels.

The idea behind MMNet$_{\text{iid}}$ is to introduce more flexibility into the update equations by using less weights and preserving the overall structure of the intermediate variable $\boldsymbol{z}^k$. The intermediate variable $\boldsymbol{z}^k$ is used with an optimal denoiser $\eta(z;\sigma_k^2)$ for Gaussian noise as a nonlinearity in each layer k. The update equations for iid Gaussian channels are given by

$$\boldsymbol{z}^k = \tilde{\boldsymbol{x}}^k + \theta_1^k \boldsymbol{H}^T \boldsymbol{y} - \theta_1^k \boldsymbol{H}^T \boldsymbol{H} \tilde{\boldsymbol{x}}^k \tag{11a}$$

$$\tilde{\boldsymbol{x}}^{k+1} = \eta^k(\boldsymbol{z}^k; \sigma_k^2) \tag{11b}$$

Using (11a) we observe that MMNet$_{\text{iid}}$ only uses one scalar trainable variable for the intermediate variable $\boldsymbol{z}^k$ and $\boldsymbol{z}^k$ is directly fed into the nonlinearity $\eta^k(z;\sigma_k^2)$ afterwards. The optimal denoiser is given by

$$\eta^k(z;\sigma_k^2) = \mathbb{E}[x|z] = \frac{\sum_{x_i \in \mathcal{A}} x_i \exp\left(-\frac{||z-x_i||^2}{\sigma_k^2}\right)}{\sum_{x_j \in \mathcal{A}} \exp\left(-\frac{||z-x_j||^2}{\sigma_k^2}\right)} \tag{12}$$

which basically resembles the softmax function. This denoiser works on a scalar basis and only uses the ith entry of $\boldsymbol{z}^k$ respectively. The sum in (12) runs over the constellation points $\mathcal{A}$ and can cope with higher order modulation schemes as well. As the denoiser is applied to the intermediate variable $\boldsymbol{z}^k$, an adapted noise variance estimation σ_k^2 is needed in every layer as the noise at the input of the denoiser is corrupted by the residual error in the layers of MMNet$_{\text{iid}}$ and the channel noise. The estimated noise variance is given by

$$\sigma_k^2 = \frac{\theta_2^k}{N_t} \left(\frac{\left\| \boldsymbol{I} - \boldsymbol{H}^T \boldsymbol{H} \right\|_F^2}{\|\boldsymbol{H}\|_F^2} \left[\left\| \boldsymbol{y} - \boldsymbol{H}\tilde{\boldsymbol{x}}^k \right\|_2^2 - N_r \sigma_n^2 \right]_+ + \sigma_n^2 \right) \tag{13}$$

where $||\cdot||_F$ denotes the Frobenius norm and σ_n^2 is the noise variance per real dimension of the additive Gaussian noise in (2). For details of this estimation the authors of [8] refer to the derivation of the AMP algorithm [5] and the orthogonal AMP [11]. The noise variance estimation (13) incorporates the second trainable parameter and thus the trainable parameters per layer k are $\boldsymbol{\theta}^k = \{\theta_1^k, \theta_2^k\}$. They are both scalars, and therefore a total number of two trainable parameters per layer are needed. As a consequence MMNet$_{\text{iid}}$ is the least complex hybrid approach presented here. Our simulation results of MMNet$_{\text{iid}}$ are based on the source code provided by the authors [12].

C. The Deep Equalization algorithm

1) General update equation: The DeEQ algorithm uses the intermediate variable $\boldsymbol{z}^k$ and introduces trainable weight matrices $\boldsymbol{W}_i$, biases $\boldsymbol{b}_i$ and a nonlinearity to it. We propose the DeEq update equations

$$\boldsymbol{z}^k = \tilde{\boldsymbol{x}}^k + \delta_1^k \boldsymbol{H}^T \boldsymbol{y} - \delta_2^k \boldsymbol{H}^T \boldsymbol{H} \tilde{\boldsymbol{x}}^k \tag{14a}$$

$$\tilde{\boldsymbol{x}}^{k+1} = \delta_3^k \tanh\left(\boldsymbol{W}_2^k \left[\boldsymbol{W}_1^k \boldsymbol{z}^k + \boldsymbol{b}_1^k\right] + \boldsymbol{b}_2^k\right) \tag{14b}$$

with $\boldsymbol{W}_1^k \in \mathbb{R}^{w_1 \times 2N_t}$, $\boldsymbol{b}_1^k \in \mathbb{R}^{w_1}$, $\boldsymbol{W}_2^k \in \mathbb{R}^{2N_t \times w_1}$, $\boldsymbol{b}_2^k \in \mathbb{R}^{2N_t}$, $\delta_1^k \in \mathbb{R}$, $\delta_2^k \in \mathbb{R}$ and $\delta_3^k \in \mathbb{R}$ in each iteration k. w_1 forms a choosable parameter for this update equations with the restriction $w_1 \geq 2N_t$, as $w_1 < 2N_t$ would lead to underdetermined systems.

The DeEq forms a neural network by first using the same intermediate variable $\boldsymbol{z}^k$ as DetNet and feds $\boldsymbol{z}^k$ into two weight matrices $\boldsymbol{W}_1^k$, $\boldsymbol{W}_2^k$ and the biases $\boldsymbol{b}_1^k$, $\boldsymbol{b}_2^k$. As one may see in (14b), it is possible to write the two linear steps as one linear step by combining $\boldsymbol{W}_1^k$ and $\boldsymbol{W}_2^k$. However, we found out during training that the choosable dimension w_1 can enhance the training process and we were thus able to achieve better performance for the DeEq algorithm if we leave the two matrices apart. The reason for this behaviour is still under investigation. This whole procedure is followed by the nonlinearity tanh to map the output $\tilde{x}_i^{k+1}$ elementwise between -1 and 1 to get a soft estimate for the equivalent BPSK symbol alphabet $\mathcal{A}$. For higher order modulations the nonlinearity needs to be adapted accordingly. E.g. the nonlinearity of MMNet (12). Finally, the scaling factor δ_3^k is introduced after the tanh operation as we found it helpful during the adaptation of the parameters during learning.

The set of trainable parameters $\boldsymbol{\theta}^k$ in each layer k consists of $\boldsymbol{\theta}^k = \left\{\boldsymbol{W}_1^k, \boldsymbol{W}_2^k, \boldsymbol{b}_1^k, \boldsymbol{b}_2^k, \delta_1^k, \delta_2^k, \delta_3^k\right\}$ and therefore, the total number of parameters per layer k is equal to $4w_1N_t + w_1 + 2N_t + 3$. We initialize $\tilde{\boldsymbol{x}}^0$ with the zero vector. To get the final estimate for the decided symbols $\hat{\boldsymbol{x}}$, we make an element wise hard decision $\hat{x}_i$ by taking the signum function element wise from the output $\tilde{x}_i^K$ at the last layer K. The complexity of the DeEQ algorithm lies in between MMNet$_{\text{iid}}$ and DetNet since MMNet$_{\text{iid}}$ has 2 trainable paramters per layer and DetNet has many more per layer.

2) Training procedure: To optimize the trainable parameters of all K layers $\boldsymbol{\theta} = \left\{\boldsymbol{\theta}^k\right\}_{k=1}^K$ in DeEQ, the deep learning libraries Tensorflow [13] and Keras [14] are used. We use a weighted squared error loss averaged over the iterations given by

$$L = \frac{1}{K} \sum_{k=1}^{K} k \parallel \tilde{\boldsymbol{x}}^k - \boldsymbol{x} \parallel_2^2 \ . \tag{15}$$

We use this loss with the Adam optimizer [15] and a learning rate of 0.001 to train our DeEQ algorithm. A batch size of 500 is applied and we generate new training data $\{\boldsymbol{x}, \boldsymbol{H}, \boldsymbol{y}\}$ for every batch as we have a model (2) at hand and can realize as many new data points as needed. We randomly generate the starting values of our weights $\boldsymbol{W}_1^k, \boldsymbol{W}_2^k$ with the Glorot uniform initializer [16], the biases $\boldsymbol{b}_1^k, \boldsymbol{b}_2^k$ are initialized with all zero vectors $\boldsymbol{0}$ and the scalar values δ_1^k, δ_2^k are initiated with 0.1 and δ_3^k is initialized with 1 in every layer k. The actual training of our DeEQ algorithm is carried out with no noise to enable the focus on equalizing the channel efficiently. The DeEq algorithm is trained offline which means the weights are learned once and afterwards the structure (14a),(14b) is used for inference only.

IV. Performance Evaluation

We evaluate the DeEQ algorithm after training in a MIMO system for 3 different environments. First we will investigate symmetrical MIMO system with $N_t = N_r = 10$ antennas, secondly a larger symmetrical system with $N_t = N_r = 30$ antennas and a mMIMO system with $N_t = 16$ and $N_r = 64$ antennas. All simulations use 4-QAM modulation. We compare the performance of DeEQ, MMNet$_{\text{iid}}$, DetNet, linear equalizer following the ZF and MMSE criterion and the OSIC using the MMSE extension [2]. As a limit, we show the performance of SD.

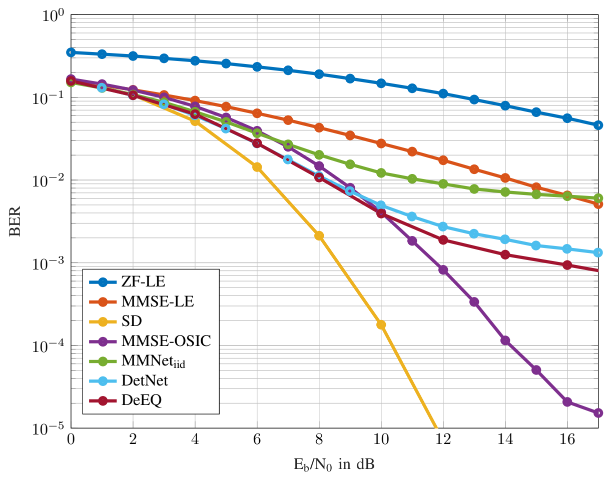


Fig. 2. BER for varying E_b/N_0 for the equalization algorithms in a $N_t = N_r = 10$ MIMO system.

In Fig. 2 we show the Bit-Error-Rates (BER) for varying E_b/N_0 in dB for the $N_t = N_r = 10$ MIMO system. During parametrization, we found a factor of $w_1 = 2N_t$ in combination with a number of $K = 30$ layers to be sufficient for the DeEQ to achieve the best performance. For MMNet$_{\text{iid}}$ we choose $K = 50$ layers as there is no general formula regarding the relation of the system dimension and the number of layers K. DetNet on the other hand has the number of layers chosen according to the MIMO system dimension and uses $K = 6N_t = 60$ layer.

As expected, the linear ZF and MMSE equalizer perform poorly in comparison to the other approaches as we have a diversity degree of only one. All hybrid approaches like DeEQ, MMNet$_{\text{iid}}$ and DetNet perform good at low E_b/N_0, but they suffer from an error floor at higher E_b/N_0. Noticeable is that DeEQ runs into a slightly lower error floor than DetNet, while both error floors are around a BER of 10^{-3}. In comparison, MMNet$_{\text{iid}}$ runs into a higher error floor at a BER of $6 \cdot 10^{-2}$.

The MMSE-OSIC outperforms the deep learning approaches for high E_b/N_0. All schemes witness a large performance gap to SD, e.g., the MMSE-OSIC shows a gap of 3dB at a BER of 10^{-3}.

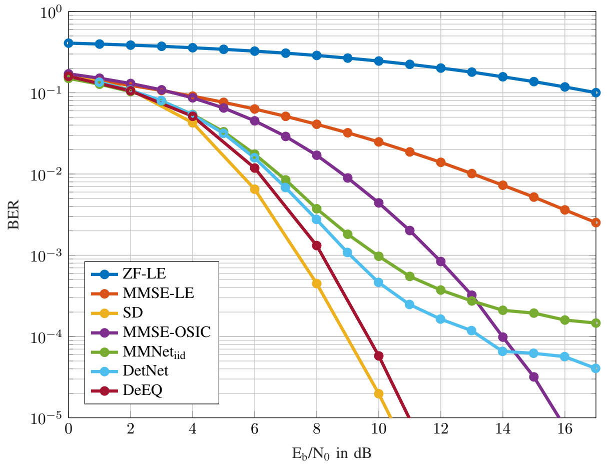


Fig. 3. BER for varying E_b/N_0 for the equalization algorithms in a $N_t = N_r = 30$ MIMO system.

In Fig. 3 the BERs are given for varying E_b/N_0 in dB for the $N_t = N_r = 30$ MIMO system. We use $K = 50$ layers and $w_1 = 6N_t$ (the dimension mapping factor between $\boldsymbol{W}_1^k$ and $\boldsymbol{W}_2^k$) to achieve the best performance for the DeEQ. For $\mathrm{MMNet_{iid}}$ we choose $K = 50$ layers and DetNet uses $K = 6N_t = 180$ layer.

A noticeable point for the hybrid approaches is the fact that the error floor lowers for all of them, especially $\mathrm{MMNet_{iid}}$'s performance increases significantly in comparison to the 10×10 MIMO system. Most surprising is the DeEQ error floor which ends up at a BER of $5 \cdot 10^{-8}$ (not shown in the plot). whereas $\mathrm{MMNet_{iid}}$ error floors at a BER of 10^{-4} and DetNet at $5 \cdot 10^{-5}$. As a consequence, the hybrid approaches outperform the MMSE-OSIC for a wider E_b/N_0 range.

We note that the error floor for the hybrid approaches lowers with the increased system dimension and the reason for this behaviour may be caused by more consistent channel statistics due to the increased matrix dimension. As all the hybrid approaches presented here use the intermediate variable $\boldsymbol{z}^k$ as an input, we may state that they all make a systematic error considering the error floor behaviours witnessed in Fig. 2 and 3. Due to the blackbox nature of machine learning approaches, it is hard to draw theoretical conclusions yet.

For the last part of the simulation section, the performance of a mMIMO system with $N_t = 16$ transmit and $N_r = 64$ receive antennas is shown in Fig. 4. DeEq is parametrized with $w_1 = 6N_t$ in combination with $K = 50$ layers being sufficient to achieve good performance. MMNet uses $K = 50$ layers and DetNet has $K = 6N_t = 96$ layers. The linear equalizers perform close to the SD as the gap is only around 1dB due to the high diversity degree. DetNet, DeEQ and the MMSE-OSIC approach the SD performance with a gap of

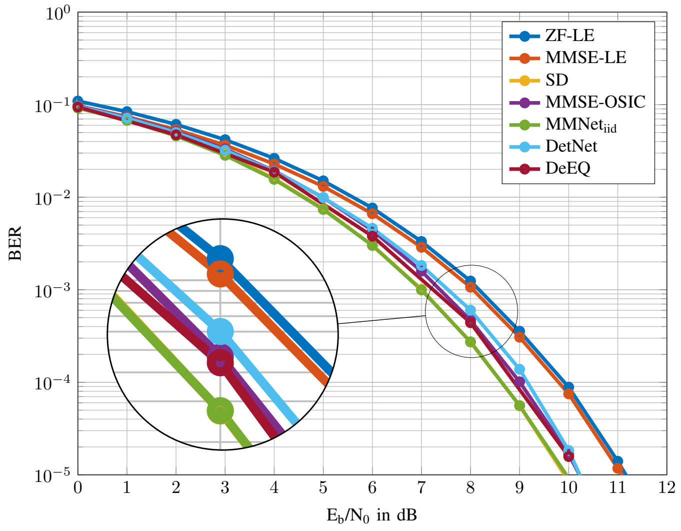


Fig. 4. BER for varying E_b/N_0 for the equalization algorithms in a $N_t = 16$ and $N_r = 64$ mMIMO system.

approximately 0.1dB. $\mathrm{MMNet_{iid}}$ reaches optimal performance while being considerably less complex to handle. A positive effect of the high diversity degree is the fact that the error floor of the hybrid approaches vanishes.

The behaviour of $\mathrm{MMNet_{iid}}$ is interesting as it shows optimal performance for the mMIMO case whereas in the symmetrical MIMO systems, the performance is worse than those of the other hybrid approaches. This behaviour may be explained with the denoiser nonlinearity and the high diversity degree present in the mMIMO case.

Overall, considering mMIMO systems, the performance gap to the SD detector will be relatively small compared to symmetrical systems as the high diversity degree and the channel hardening effect massively increase the performance of these algorithms.

V. Conclusion

We propose a new algorithm "DeEQ" for the MIMO equalization problem. By using a hybrid approach of inserting model knowledge into a neural network like structure we achieve good performance for the presented case of MIMO systems and outperform existing state of the art schemes in some cases. In the future the proposed algorithm will be extended to higher order modulation techniques, the effect of correlated channel matrices is investigated as well as a better understanding of the error floor should be achieved.

References

[1] E. Björnson, J. Hoydis, and L. Sanguinetti, "Massive MIMO networks: Spectral, energy, and hardware efficiency," *Foundations and Trends in Signal Processing*, vol. 11, no. 3-4, pp. 154–655, 2017.

[2] D. Wübben, R. Böhnke, V. Kühn, and K. D. Kammeyer, "MMSE extension of V-BLAST based on sorted QR decomposition," in *2003 IEEE 58th Vehicular Technology Conference. VTC 2003-Fall, Orlando, FL, USA*, vol. 1, Oct 2003, pp. 508–512.

[3] O. Simeone, "A very brief introduction to machine learning with applications to communication systems," *IEEE Transactions Cognitive Communications & Networking*, vol. 4, no. 4, pp. 648–664, 2018.

[4] C. Jeon, R. Ghods, A. Maleki, and C. Studer, "Optimal data detection in large MIMO," *CoRR*, vol. abs/1811.01917, 2018. [Online]. Available: http://arxiv.org/abs/1811.01917

[5] C. Jeon, R. Ghods, A. Maleki, and C. Studer, "Optimality of large mimo detection via approximate message passing," in *2015 IEEE International Symposium on Information Theory (ISIT), Hong Kong, China*, June 2015, pp. 1227–1231.

[6] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2554–2564, May 2019.

[7] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO Detection," in *18th IEEE International Workshop on Signal Processing Advances in Wireless Communications, (SPAWC), Sapporo, Japan*, July 2017.

[8] M. K. Shirkoohi, M. Alizadeh, J. Hoydis, and P. Fleming, "Adaptive neural signal detection for massive MIMO," *CoRR*, vol. abs/1906.04610, 2019. [Online]. Available: http://arxiv.org/abs/1906.04610

[9] B. Widrow and M. Hoff, "Adaptive Switching Circuits," *Neurocomputing.*, pp. 126–134, Jan. 1960.

[10] N. Samuel, T. Diskin, and A. Wiesel, "Deep MIMO Detection, source code," https://github.com/neevsamuel/DeepMIMODetection.

[11] J. Ma and L. Ping, "Orthogonal AMP," *IEEE Access*, vol. 5, pp. 2020–2033, Jan 2017.

[12] M. K. Shirkoohi, M. Alizadeh, J. Hoydis, and P. Fleming, "MMNet Source Code," https://github.com/mehrdadkhani/MMNet.

[13] M. Abadi *et al.*, "TensorFlow: Large-scale machine learning on heterogeneous systems," 2015, software available from tensorflow.org. [Online]. Available: http://tensorflow.org/

[14] F. Chollet *et al.*, "Keras," https://keras.io, 2015.

[15] D. P. Kingma and J. Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR), San Diego, CA, USA*, May 2015.

[16] X. Glorot and Y. Bengio, "Understanding the difficulty of training deep feedforward neural networks," in *Proceedings of the 13th International Conference on Artificial Intelligence and Statistics*, vol. 9, Chia Laguna Resort, Sardinia, Italy, 13–15 May 2010.