POLITECNICO
MILANO 1863

SCUOLA DI INGEGNERIA INDUSTRIALE
E DELL'INFORMAZIONE

Executive Summary of the Thesis

Deep Learning based Approximate Message Passing for MIMO Detection in 5G

Laurea Magistrale in Computer Science Engineering - Ingegneria Informatica

Author: Andrea Pozzoli

Advisor: Prof. Umberto Spagnolini

Academic year: 2020-2021

Abstract

Massive Multiple Input Multiple Output (MIMO) is the main technology in the Fifth Generation (5G) mobile communication system. It consists in several receiver antennas that serve multiple transmitters, and it increases link reliability and information throughput. However, one of the main problem for MIMO systems is MIMO detection, so retrieving the original messages at the receiver side from a noisy signal. The optimal technique to solve the problem is called Maximum Likelihood (ML), but it does not scale. Several sub-optimal techniques have been tested in order to solve MIMO detection problem, trying to balance the complexity-performance trade-off. In recent years, Approximate Message Passing (AMP) based techniques brought interesting results. Moreover, deep learning (DL) has been tested with promising results. In the thesis, two new techniques AMP and DL based, called OAMP-Net2 and LVAMP, have been tested and compared with the state of art, using channel matrices that differ in channel model and size. OAMP-Net2 revealed to be a consistent technique that can be used in solving MIMO detection problem. It not only provides really interesting results on both i.i.d Gaussian and Kronecker channel models, but it adapts easily to different models, providing good results on Kronecker channel models also when the network is trained only with i.i.d. Gaussian matrices. LVAMP instead has performances that are similar to MMSE, but with a lower complexity. Also LVAMP adapts well to complex channel such as OAMP-Net2.

Keywords: **5G, MIMO detection, Approximate Message Passing, OAMP-Net2, LVAMP, MMNet, Deep Learning**

1. Introduction

During the last years, the demand in wireless communication data transferring in a rapid and reliable way is increased and continues to increase drastically. In order to face the quantity of data and the speed of communication, Massive Multiple Input Multiple Output (MIMO) technology is the key technology that is used the 5G mobile communication system [3]. This technique enhances a large number of antennas at

both the transmitter and the receiver side [15]. In 5G systems, a Base Station (BS) equipped with a large number of receiver antennas (64-256) serves multiple single-antenna transmitters, called User Equipment (UE), simultaneously on the same frequency [18]. MIMO increases the information throughput and link reliability by exploiting link diversity.
With a high number of antennas, trying to recover the transmitted information in a massive MIMO up-link receiver is more computationally complex, because the transmitted signals interfere with each other. The channel is time-varying and shared. The thesis focuses on signal detection in a multi-user communication scenario, in particular at the receiver side. This means that the focus is on trying to detect correctly the signals transmitted by the UEs, from the signal that arrives at the BSs.
The optimal MIMO detector in terms of performance is the Maximum Likelihood (ML) detector, but it has a complexity that is exponentially increasing with the number of transmitters. It consists of an exhaustive search over all the possible symbols for each UE. Therefore it is necessary to find sub-optimal algorithms with the best performance/complexity trade-off. One of the most used sub-optimal algorithm is Minimum Mean Squared Error (MMSE) algorithm, that reduces ML complexity, scarifying accuracy.
Promising detectors with excellent performance and reduced complexity are iterative detectors based on Approximate Message Passing (AMP) algorithm [7]. The AMP based detector approximates the posterior distribution on a dense factor graph by using the central limit theorem and the Taylor expansion [5]. AMP based detector works well with channel matrices with independent elements and identically sub-Gaussian distributions.
Recently, Deep Learning (DL) reported promising results in signal detection showing that it can reduce the prediction complexity during training. Therefore Section 2 focuses on some DL and AMP based techniques for solving MIMO detection.

2. Related work

In [13] the Generalised AMP (GAMP) algorithm is presented. GAMP extends AMP methods by including arbitrary distribution on both the input and the output of the transform. Its advantage is an efficient approximate implementation of max-sum and sum-problem loopy Belief Propagation (BP). Moreover, the algorithm is computationally simple involving only scalar estimation and linear transforms. In [21] a deep learning based Trainable AMP (TAMP) algorithm is proposed. It consists of a neural network composed by a preprocessing layer that acts like a learnable filter followed by detection layers derived by unfolding the iterations of GAMP. TAMP uses parameters that are trainable in order to control the prior mean and variance of MMSE denoiser. TAMP uses backpropagation for parameter tuning.
A method based on the approximation of the original discrete messages sent in an AMP based factor graph with continuous Gaussian messages through Kullback-Leiber divergence criterion has been proposed in [17]. The principle of expectation propagation is applied in order to compute the approximate Gaussian messages and then the approximate message is computed from the Gaussian approximate belief. Moreover, it uses a posteriori probabilities fed back from channel decoders, and the central limit theorem. The algorithm presents low complexity and good performances for small MIMO systems.
The authors of [20] combined different techniques in order to complete channel estimation, MIMO detection and noise level estimation simultaneously. They proposed an algorithm called variational approximate message passing that exploits the advantages of AMP, in particular using Bilinear Generalized AMP, and Variational Bayesian Inference.
Another deep learning neural network, this time base on belief propagation, has been proposed in [11] and it is called DLBP detector. The network is composed by multiple units that consist of a four layer neural network derived by unfolding the BP algorithm. The output of each unit is designed so that it is possible to speed up the training, that is conducted using cross entropy loss function.

3. Problem and research question

Every years, new methods are studied, presented and tested in order to find the one that best deals with performance-complexity tradeoff. Recently two new algorithms based on deep learning and AMP have been proposed. The first one is called LVAMP and it is a deep leaning based neural network that is obtained by unfolding the Vectorized AMP algorithm (VAMP). The second one instead is a new network obtained by unfolding Orthogonal AMP (OAMP) algorithm, called OAMPNet2. A comparison between these two algorithms in different situations and with a common baseline is missing, therefore the thesis is focused on that.
From the problem described, the research question emerges easily and can be stated as follow.

Research Question: *Can LVAMP and OAMPNet2 be considered as sub-optimal solutions to MIMO detection problem and which one is the best network in terms of complexity and SER on different MIMO scenarios?*

Sub-optimal solution stands for a working algorithm with a possible practical implementation that can handle the MIMO detection problem with a timing that respects the communication speed and with low errors during the detection. In particular, the thesis considers as sub-optimal solution a MIMO detector that obtains at least the same performances of MMSE detector, with a complexity that is polynomial respect to the dimension of the MIMO system. The complexity of the algorithm is the time complexity, not the space one, and it will affect the timing of the algorithm. The Symbol Error Rate (SER) metric is the measure of the performance used in the thesis and that is explained later.

4. MIMO

MIMO system can be represented trough a matrix mathematical approach. Each transmitter antenna i sends a x_i message, that follows different paths with different channel properties to reach the receiver antennas. The different paths are represented by h_{ij} that is the channel that links the transmitter i with the receiver j. Each receiver antenna j collects a message y_j that is the combination of signals that arrives to the antenna j. with three transmitters and three receivers for example, the system can be expressed as in 1:

$$\begin{aligned} y_1 &= h_{11}x_1 + h_{21}x_2 + h_{31}x_3 \\ y_2 &= h_{12}x_1 + h_{22}x_2 + h_{32}x_3 \\ y_3 &= h_{13}x_1 + h_{23}x_2 + h_{33}x_3 \end{aligned} \tag{1}$$

In general, the system is modelled through the MIMO system 2 in matrix form :

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

where $\bar{y}$ is a vector of complex values of length $\bar{N}_r$, with $\bar{N}_r$ that is the number of receiver antennas, and represents the signal received by the BS. $\bar{x}$ represents the signal transmitted by the UEs and it is a vector of complex values, taken from a discrete alphabet $\bar{\mathcal{A}}$, of length $\bar{N}_t$, with $\bar{N}_t$ that is the number of transmitter antennas. $\bar{n}$ represents the noise and it is a vector of complex numbers of length $\bar{N}_r$. Finally, $\bar{H}$ is called channel matrix, it has shape $\bar{N}_r \times \bar{N}_t$ and $h_{ij} \in \mathcal{C}$ is the channel gain that derives from $\bar{x}_j$ and $\bar{x}_i$ antennas. The channel matrix can assume different structures, called channel models. In particular in this thesis the focus is on the iid Gaussian channel model and the Kronecker channel model.
The MIMO model that is used in the thesis follows the following assumptions:

- Noise is complex zero-mean Gaussian $n_i \sim \mathcal{CN}(0, \sigma^2)$ and the Covariance matrix is $\sigma^2 I_{\bar{N}_r}$.
- The column of $\bar{H}$ are normalized, $\|\bar{h}_i\|_2 = 1$.
- $SNR = 10\log_{10}\frac{\mathbf{E}(\|\bar{H}\bar{x}\|_2^2)}{\bar{N}_r\sigma^2}$.
- The channel matrix $\bar{H}$ is known by the receiver.

4.1. i.i.d Gaussian channel model

The common channel model, used in most of the literature in order to test and compare MIMO signal detection algorithms, is the iid Gaussian one. This channel model is so common because it is simple but it is far from the real scenario. In fact, it assumes that each $\bar{h}_{ij}$ is independent from all the others and $\bar{h}_{ij} \sim \mathcal{CN}(0, 1/\bar{N}_r)$. The columns of $\bar{H}$ are normalized such that $\|\bar{h}_i\|_2 = 1$ in this thesis. Due to the fact that the channels in a real scenario are spatially correlated and so dependent each other, also the Kronecker channel model is considered.

4.2. Kronecker channel model

As said before, the Kronecker channel model simulates the spatial correlation among channels [12]. It is modelled as: $\bar{H} = R_R^{1/2} K R_T^{1/2}$ where $k_{ij} \sim \mathcal{CN}(0, 1/\bar{N}_r)$ and R_R and R_T are the receiver and transmitter Correlation matrices respectively. These matrices are generated according to an exponential correlation model, depending on a parameter of correlation that assumes values from 0 (iid Gaussian) to 1 when all the channels interfere with each other. The coefficient at receiver side is indicated as ρ_r while the one at transmitter side is ρ_t. Also in this case, the column of $\bar{H}$ are normalized such that $\|\bar{h}_i\|_2 = 1$.

4.3. From Complex to Real MIMO System

Due to the fact that working with complex values is more difficult than working with real numbers, the MIMO model previously described can be convert from a complex-valued system to a real-valued system. The conversion works by treating the real and the imaginary parts of a complex number separately. A new vector is defined for each variable of the model, in particular the transmitted symbol vector becomes

$$x = \begin{bmatrix} \Re(\bar{x}^T) & \Im(\bar{x}^T) \end{bmatrix}^T \tag{3}$$

where $\Re(\cdot)$ is the real part of the complex number between brackets and $\Im(\cdot)$ is its imaginary part. Following the same approach, also the other variables are modified and in particular

$$\begin{aligned} y &= \begin{bmatrix} \Re(\bar{y}^T) & \Im(\bar{y}^T) \end{bmatrix}^T \\ n &= \begin{bmatrix} \Re(\bar{n}^T) & \Im(\bar{n}^T) \end{bmatrix}^T \\ H &= \begin{bmatrix} \Re(\bar{H}^T) & -\Im(\bar{H}^T) \\ \Im(\bar{H}) & \Re(\bar{H}) \end{bmatrix} \end{aligned} \tag{4}$$

With these new variables, it is possible to define differently the MIMO system in 5

Definition 1 (Real-valued MIMO):

$$\begin{aligned} y &= Hx + n \\ N_r &= 2\bar{N}_r \\ N_t &= 2\bar{N}_t \end{aligned} \tag{5}$$

The last modification that must be applied concerns the alphabet $\mathcal{A}$. With these new definition of MIMO system, x can assume values coming from $\mathcal{A}$, where $\mathcal{A} = \Re(\bar{\mathcal{A}}) = \Im(\bar{\mathcal{A}})$ in the case of QAM modulation.

4.4. QAM Modulation

Modulation is an operation applied on the periodic waveform called carrier signal that varies its phase and/or its amplitude and/or its frequency in order to transmit information. The modulation technique used in the thesis is the Quadrature Amplitude Modulation (QAM), that changes the amplitude and the phase of the carrier signal. It consists of producing a signal in which two carriers with the same frequency shifted in phase by 90 degrees (they are in quadrature or orthogonal) are modulated and combined. At the receiver side the signal can be divided thank to the orthogonality property. A basic signal can transmit only a 0 or a 1 since it can exhibit only two positions. Thanks to QAM it is possible to enhance different points that differs for phase and amplitude. The QAM points are spaced in a squared grid with equal horizontal and vertical spacing, that is called constellation diagram. Due to the fact that digital communications use binary data, usually the number of points that compose the constellation is a power of 2. The most used forms of QAM are QAM-4, QAM-16, QAM-64, QAM-256 as QAM is square. Different binary values are assigned to different symbols in the constellation and in this way it is possible to transfer data with a single signal in a much higher rate. In QAM-M the points have values along each axis equal to $\pm(\sqrt{M} - 1)d/2$ where M is a power of two and square number and d is the minimum distance between two different points in the constellation.

5. MIMO Detection

Definition 2 (MIMO Detection): *Given a MIMO system a MIMO Detection method can be defined as the problem of retrieving the transmitted signal vector* $x \in \mathcal{R}^{N_t}$ *from a noisy linear measurement that can be expressed as* $y = Hx + n \in \mathcal{R}^{N_r}$ *where* H *is a known matrix* $\in \mathcal{R}^{N_r \times N_t}$ *and* n *is an unknown unstructured noise vector* $\in \mathcal{R}^{N_r}$.

This problem can be also called standard lin-

ear regression or, in the signal processing literature, linear inverse problem (or compressive sensing if $N_r << N_t$ and x is sparse). The multiple symbols can be detected separately or jointly. In joint detection in order to detect a symbol it is required to consider also the characteristics of the other symbols, while in separate detection each symbol is detected independently. Typically joint detection achieves better performances than separate detection despite an higher complexity. The channel matrix can be known from explicit channel estimation, in this case the detection of x is called coherent detection, or the explicit estimation can be avoided and in this case the detection is said incoherent detection.
The optimal method for solving the MIMO detection problem is represented by the ML detector in 6.

Definition 3 (Maximum Likelihood Detector):

$$\hat{x} = \arg\max_{x\in\mathcal{A}^{N_t}} prob(y|x,H) = \arg\min_{x\in\mathcal{A}^{N_t}} \|y - Hx\|^2 \tag{6}$$

The best estimation $\hat{x}$ for x is the one that maximize the likelihood $p(y|x,H)$, but this approach suffers from a high computational complexity due to the exhaustive search over all possible values of $x \in \mathcal{A}^{N_t}$. This optimization problem is a NP-hard problem due to the finite alphabet constraint.
To solve the MIMO detection problem it is required to find a sub optimal solution trying to handle the performance/complexity trade-off. There are methods that are computationally cheap but with low accuracy, and methods with high accuracy but that are computationally expensive. The accuracy and the complexity of some methods can vary with the dimension of the MIMO system.
Several methods have been tested in order to solve this problem during the years, and the most common is Minimum Mean Squared Error (MMSE) [14]. MMSE finds $\hat{x}_{MMSE}$ through

$$\hat{x}_{MMSE} = \arg\min_{x\in\mathcal{A}^{N_t}} \int \|x - \bar{x}\|^2 p(x|y)dx \tag{7}$$

that is equal to compute $\mathbf{E}[x|y]$. MMSE can be expressed also as

$$\hat{x} = (H^T H + \sigma^2 I_{N_t})^{-1} H^T y \approx z + w \tag{8}$$

In practice MMSE provides good results but the performance are still far from optimal and it has some difficulties at scaling when the number of antennas increases. MMSE is used as benchmark in most of the comparative research for MIMO detection problem, therefore also in the thesis it is used as benchmark.
If the noise w can be expressed as $w \sim \mathcal{N}(0, \gamma_w^{-1} I)$, it is called AWGN, with $\gamma_w > 0$.

5.1. Iterative framework

An approach that can be followed in order to solve the MIMO detection problem is an iterative estimation of the transmitted signal. This class of algorithms is based on a number T of iterations that comprise two steps (9)

$$\begin{aligned} z_t &= \hat{x}_t + A_t(y - H\hat{x}_t) + b_t \\ \hat{x}_{t+1} &= \eta_t(z_t) \end{aligned} \tag{9}$$

The first step has as input $\hat{x}_t$, that is the current estimation of the transmitted signal x, the channel matrix H and the received signal y, and it computes z_t that is a linear transformation. The second step instead is a non-linear denoiser that is applied to z_t in order to produce the new estimation $\hat{x}_{t+1}$ of x, that is used for the first step of the next iteration. The goal of each iteration is to improve the estimation $\hat{x}_t$ of x respect to the previous iterations. For the first iteration, $\hat{x}_0 = 0$. The term $y - H\hat{x}_t$ is called residual term. The denoiser $\eta_t(\cdot)$ can be any non-linear function, but usually it applies the same thresholding function to each element. A element-wise η function can reduce the complexity of the denoiser. Usually the parameters required by the denoising function are indicated with σ_t, and they can change for each iteration. A common choice for the denoising function is the minimizer of $\mathbf{E}[\|\hat{x} - x\|_2|z_t]$ that is given by $\eta_t(z_t) = \mathbf{E}[x|z_t]$.
Optimal denoiser for Gaussian noise: if the noise at the input of the denoiser $z_t - x$ has an iid Gaussian distribution with a covariance matrix that is diagonal with value $\sigma_t^2 I_{N_t}$, the element-wise thresholding function derived from the previous formula is

$$\beta_t^g(z;\sigma_t^2) = \frac{1}{Z}\sum_{x_t\in\mathcal{A}} x_t \exp\left(-\frac{\|z - x_t\|^2}{\sigma_t^2}\right) \tag{10}$$

where $Z = \sum_{x_t\in\mathcal{A}} \exp(-\frac{\|z-x_t\|^2}{\sigma_t^2})$. In this case σ_t represents the standard deviation of the Gaussian noise.

5.2. AMP

Another approach for approximately solving the MIMO detection problem is through BP if the problem is seen as a bipartite graph [16]. On this type of graph, BP requires a number of update messages that is in $\mathcal{O}(N_r N_t)$ for each iteration, that is not feasible for large MIMO systems. To face this limit, Jeon et al. [9] proposes Approximate Message Passing (AMP) for solving MIMO detection problem in iid Gaussian scenario with a lower complexity. In fact, AMP uses $\mathcal{O}(N_r + N_t)$ messages for each iteration. The AMP algorithm performs the steps:

$$\begin{aligned} z_t &= \hat{x}_t + H^H(y - H\hat{x}_t) + b_t \\ b_t &= \alpha_t(H^H(y - H\hat{x}_{t-1}) + b_{t-1}) \\ \hat{x}_{t+1} &= \eta_t(z_t; \sigma_t) \end{aligned} \tag{11}$$

AMP is an iterative algorithm that uses $A_t = H^H$ and a b_t term that is called Onsanger correction term. Both σ_t and α_t can be computed using Signal Noise Ratio (SNR) and system parameters such as the dimension of the system or the constellation. The denoising function is the optimal one described in formula optimal for each element of z_t. AMP is asymptotically optimal for large iid Gaussian channel matrices [4].

5.3. OAMP

A variant of AMP that relaxes the iid Gaussian channel assumption is the Orthogonal AMP (OAMP) that works for unitarily invariant channel matrices [10]. OAMP is an optimal estimator with excellent convergence properties [19]. The principle of OAMP is to decouple the posterior probability $p(x|y, H)$ into a series of probabilities $p(x_i|y, H)_{i=1,2,\dots,N_t}$ in an iterative way. The OAMP detector can be written as Algorithm 1 [6]

The algorithm is divided into two modules, the linear estimator used to compute r_t and the non linear estimator to estimate x_{t+1}. v_t^2 and τ_t^2 are instead the average variance of $q_t = \hat{x}_t - x$ and $p_t = r_t - x$ respectively. The vector p_t is used to measure the accuracy of the output of the linear estimator, while q_t is used for the nonlinear estimator. v_t^2 and τ_t^2 are defined as

$$\begin{aligned} v_t^2 &= \frac{\mathbf{E}[\|q_t\|_2^2]}{N_t} \\ \tau_t^2 &= \frac{\mathbf{E}[\|p_t\|_2^2]}{N_t} \end{aligned} \tag{12}$$

Algorithm 1 OAMP

Require: received signal y, channel matrix H, noise covariance matrix $R_{\hat{n}\hat{n}}$
Output: Estimated signal $\hat{x}_{T+1}$
Initialize: $\tau_0 \leftarrow 1, \hat{x}_0 \leftarrow 0$
for $t = 1, \dots, T-1$ **do**
  $r_t = \hat{x}_t + W_t(y - H\hat{x}_t)$
  $\hat{x}_{t+1} = \mathbf{E}\{x|r_t, \tau_t\}$
  $v_t^2 = \frac{\|y - H\hat{x}_t\|_2^2 - \mathbf{tr}(R_{\hat{n}\hat{n}})}{\mathbf{tr}(H^H H)}$
  $\tau_t^2 = \frac{1}{N_t}\mathbf{tr}(B_t B_t^H)v_t^2 + \frac{1}{N_t}\mathbf{tr}(W_t R_{\hat{n}\hat{n}} W_t^H)$
end for
Return $\hat{x_T}$

The linear estimator presents the matrix W_t that can assume different values such as the transpose of H, its pseudo inverse or the LMMSE matrix, but the optimal definition is

$$W_t = \frac{N_t}{\mathbf{tr}(\hat{W}_t H)}\hat{W}_t \tag{13}$$

where $\hat{W}_t$ is the LMMSE matrix and it can be defined as

$$\hat{W}_t = v_t^2 H^H (v_t^2 H H^H + R_{\hat{n}\hat{n}})^{-1} \tag{14}$$

where $R_{\hat{n}\hat{n}}$ is the covariance matrix of the noise in signal detector $\hat{n}$. The matrix W_t is decorrelated when $tr(B_t) = 0$, where $B_t = I - W_t H$ (p_t uncorrelated with x and mutually uncorrelated with zero-mean and identical variance).

The non linear estimator instead is MMSE estimate of x which is in relation to r_t through

$$r_t = x_t + w_t \tag{15}$$

where $w_t \sim \mathcal{NC}(0, \tau_t^2 I)$. Due to the fact that the values of x are taken from a constellation of symbols and that the estimation is based on MMSE, in order to estimate $\hat{x}$ it is used

$$\hat{x}_{t+1} = \mathbf{E}\{x_i|r_i, \tau_i\} = \frac{\sum_{s_i} s_i \mathcal{N}_{\mathcal{C}}(s_i; r_i, \tau_t^2)p(s_i)}{\sum_{s_i} \mathcal{N}_{\mathcal{C}}(s_i; r_i, \tau_t^2)p(s_i)} \tag{16}$$

where $p(s_i)$ is the prior distribution of the symbol x_t and is defined as $p(x_i) = \sum_{j \in N_r} \frac{1}{\sqrt{N_r}}\delta(x_i - s_j)$.

It is important to notice that the prior mean of MMSE estimator is r_t and its variance is τ_t^2 and they control the accuracy and convergence of $\hat{x}_{t+1}$.

5.4. VAMP

In [14] the Vector Approximate message passing (VAMP) algorithm is proposed, showing that it holds under a bigger class of channel matrices respect to AMP, those that are right-orthogonally invariant. Moreover, it keeps the same desirable properties of AMP, such as low periteration complexity, convergence in few iterations, and shrinkage inputs r_t that can be modelled through the AWGN model [2]. The VAMP algorithm is based on the "economy" SVD of the channel matrix: $H = \bar{U}\mathbf{diag}(\bar{s})\bar{V}^T$ where $\bar{s} \in \mathcal{R}^R for R := \mathbf{rank}(H) \leq \min(N_r, N_t)$. Algorithm 2 is the SVD form of the VAMP algorithm.

Algorithm 2 VAMP in SVD form

Require: received signal y, channel matrix H, denoiser function $g_1(\cdot, \gamma_t)$, noise precision $\gamma_w \geq 0$, number of iterations T, $r_0 \geq 0$, $\gamma_0 \geq 0$

Output: Estimated signal $\hat{x}_T$

Compute economy SVD $H = \bar{U}\mathbf{diag}(\bar{s})\bar{V}^T$

Compute preconditioned $\tilde{y} = \mathbf{diag}(\bar{s})^{-1}\bar{U}^T y$

for $t = 0, 1, \ldots, T$**do do**

$\hat{x}_t = g_1(r_t, \gamma_t)$

$\alpha_t = \langle g_1'(r_t, \gamma_t)\rangle$

$\tilde{r}_t = (\hat{x}_t - \alpha_t r_t)/(1 - \alpha_t)$

$\tilde{\gamma}_t = \gamma_t(1 - \alpha_t)/\alpha_t$

$d_t = \gamma_w \mathbf{diag}(\gamma_w \bar{s}^2 + \tilde{\gamma}_t \mathbf{1})^{-1}\bar{s}^2$

$\gamma_{t+1} = \tilde{\gamma}_t\langle d_t\rangle/(\frac{N_t}{R} - \langle d_t\rangle)$

$r_{t+1} = \tilde{r}_t + \frac{N_t}{R}\bar{V}\mathbf{diag}(d_t/\langle d_t\rangle)(\tilde{y} - \bar{V}^T\tilde{r}_t)$

end for

Return$\hat{x}_T$

In the algorithm, $g_1(r_t, \gamma_t) : \mathcal{R}^{N_t} \to \mathcal{R}^{N_t}$ is defined as $g_1 = \arg\min_{x\in\mathcal{A}}[\frac{\gamma_t}{2}|x - r_t|^2 - \ln p(x)]$ where $p(x)$ is the prior distribution of x, while $g_1'(r_t, \gamma_t)$ as $g_1'(r_t, \gamma_t) = diag[\frac{\partial g_1(r_t,\gamma_t)}{\partial r_t}]$, and $\langle\cdot\rangle$ is the empirical averaging operation $\langle u\rangle := \frac{1}{N_t}\sum_{n=1}^{N_t} u_n$. Moreover, $r_t \in \mathcal{R}^{N_t}$ is called residual term at iteration $t-th$ and γ_t represents the reciprocal of its variance. Finally, R is defined as $R = rank(H)$.

As it possible to see, the VAMP algorithm is very similar to the AMP one [8]. In fact, the denoising and the divergence steps are identical, and also the VAMP algorithm presents an Onsanger term $\alpha_k r_k$. Finally, another similarity, not visible from the comparison of the algorithm, is that for both the algorithms for certain large random H, r_k behaves like a white Gaussian noise corrupted version of x, $r_k = x + \mathcal{N}(0, \tau_k I)$ for some variance $\tau_k > 0$.

There is also another approach to write the VAMP algorithm without using the SVD form. In this second variant, the linear MMSE and the trace of the covariance matrix must be computed at each iteration, involving the inverse of a $N_t \times N_t$ matrix. Differently from AMP that uses a loopy factor graph with scalar valued nodes, VAMP uses a non loopy graph with vector valued nodes and this is the reason for the name Vector AMP. The second version of the VAMP algorithm is called LMMSE form and it follows the steps of Algorithm 3.

Algorithm 3 VAMP in LMMSE form

Require: LMMSE estimator $g_2(r_{2t}, \gamma_{2t})$, denoiser function $g_1(\cdot, \gamma_{1t})$, number of iterations T, r_{10} and $\gamma_{10} \geq 0$

Output: Estimated signal $\hat{x}_T$

for $t = 0, 1, \ldots, T$ **do**

Denoising

$\hat{x}_{1t} = g_1(r_{1t}, \gamma_{1t})$

$\alpha_{1t} = \langle g_1'(r_{1t}, \gamma_{1t})\rangle$

$\eta_{1t} = \gamma_{1t}/\alpha_{1t}$

$\gamma_{2t} = \eta_{1t} - \gamma_{1t}$

$r_{2t} = (\eta_{1t}\hat{x}_{1t} - \gamma_{1t}r_{1t})/\gamma_{2t}$

LMMSE

$\hat{x}_{2t} = g_2(r_{2t}, \gamma_{2t})$

$\alpha_{2t} = \langle g_2'(r_{2t}, \gamma_{2t})\rangle$

$\eta_{2t} = \gamma_{2t}/\alpha_{2t}$

$\gamma_{1,t+1} = \eta_{2t} - \gamma_{2t}$

$r_{1,t+1} = (\eta_{2t}\hat{x}_{2t} - \gamma_{2t}r_{2t})/\gamma_{1,t+1}$

end for

Return$\hat{x}_{1T}$

where $g_2(r_{2t}, \gamma_{2t}) = (\gamma_w H^T H + \gamma_{2t}I)^{-1}(\gamma_w H^T y + \gamma_{2t}r_{2t})$ is a MMSE estimate linear in r_{2k} (that's why it is called LMMSE form) of a random vector x_2 under likelihood $\mathcal{N}(y; Hx_2, \gamma_w^{-1}I)$ and prior $\mathcal{N}(r_{2k}, \gamma_{2k}^{-1}I)$. Instead, $\langle g_2'(r_{2k}, \gamma_{2k})\rangle$ can be defined as $\langle g_2'(r_{2k}, \gamma_{2k})\rangle = \frac{\gamma_{2t}}{N_r}\mathbf{tr}[(\gamma_w H^T H + \gamma_{2t}I)^{-1}]$. For what concerns g_1 and g_1', they are defined as in Algorithm 2.

VAMP alternates between two stages that can be summarized as:

- a MMSE inference of x under likelihood $\mathcal{N}(y; Hx, \sigma_w^2 I)$ and pseudo prior $\mathcal{N}(x; \tilde{r}_t, \tilde{\sigma}_t^2 I)$

- a MMSE inference of x under pseudolikelihood $\mathcal{N}(r_t; x, \sigma_t^2 I)$ and prior $x \sim p(x)$.

The OAMP algorithm is similar to VAMP but they differ in the approximation of certain variance terms, and on the reliance on matrix inversion.

6. Deep learning

Deep learning is a subset of machine learning that from a dataset composed by pairs (feature, label) $\{(y^{(d)}, x^{(d)})\}_{d=1}^{D}$ where D represents the number of pairs, tries to learn some parameters of a Artificial Neural Network (ANN) aiming to predict the unknown label $\hat{x}$ associated to new data y. The network knows y and uses it in many layers of processing, where each layer is composed by a linear transformation and a component-wise non-linearity.
MLP is a type of (ANN) that concatenates T basic blocks also called layers

$$\begin{aligned} V_t &\in \mathcal{R}^{M_t} \times \mathcal{R}^{M_{t-1}} \\ y_t = \Omega(V_t y_{t-1}) &: \mathcal{R}^{M_{t-1}} \to \mathcal{R}^{M_t} \end{aligned} \tag{17}$$

where V_t is a linear operator, Ω a non linear function, and M_t the dimension of the vector in block $t-th$. In order to predict $\hat{x}$ it is necessary to create a net composed by the concatenation of T layers:

$$\hat{x} = g(y) = g(y_0) = \Omega(V_t \Omega(V_{t-1} - \ldots \Omega(V_1 y_0) \ldots)) \tag{18}$$

During the training phase. The values of $V_1, \ldots, V_T$ are learnt with the goal of reducing the prediction error on the labels in the training data. For quantifying the prediction error, it is required to define a loss function $\mathcal{L}$, that brings the problem to be defined as a minimization problem

$$\arg \min_{V_1, \ldots, V_T} \mathcal{L}([x^{(1)}, \ldots, x^{(D)}], [\hat{x}^{(1)}, \ldots, \hat{x}^{(D)}])$$

.

In the thesis, to achieve the values of the parameters $V_1, \ldots, V_T$ that minimize the loss function L two steps are needed. First, the gradients of L with respect to the parameters are computed. Second, the parameters are updated through gradient descent, and, in the case of the thesis, with Adam optimizer as update rule [15].
Deep learning is usually used in MIMO Detection by unfolding an iterative MIMO Detection models and adding some trainable parameters. Deep learning can increase significantly the speed of convergence with respect to the traditional iterative algorithms. Moreover, the DL methods can decrease the average recovery error respect to the iterative version, because they do not need to model the problem but they learn a mapping from the input to the output directly [1].

6.1. OAMP-Net2

In [7] the authors proposed a model driven deep learning network based on OAMP, named OAMPNet2, with four trainable parameters in order to adapt to various channel environments and take channel estimation error into consideration. OAMPNet2 performs considerably better than OAMP and is more robust with respect to SNR, channel correlation, modulation symbol and MIMO configuration mismatches.
The OAMPNet2 algorithm performs signal detection with channel estimation error. In the MIMO system, it is possible to express the channel matrix H as $H = \hat{H} - \Delta H$ where $\hat{H}$ is the estimated channel and ΔH is the error on the channel estimation. If the MIMO detection problem uses the estimated channel matrix $\hat{H}$, $\hat{n}_d = n_d - \Delta H x_d$ is the noise in signal detector that includes the channel estimation error and the AWGN vector. This noise is supposed to be Gaussian distributed.
OAMPNet2 is a deep learning network composed by T cascade layers with the same architecture but different parameters. The inputs of the network are y and $\hat{H}$, while the output is $\hat{x}_{T+1}$. For each layer, the input is the $\hat{x}_t$ estimation of x computed in the previous layer. The OAMPNet2 detector follows these steps at each layer:

$$\begin{aligned} r_t &= \hat{x}_t + \gamma_t W_t (y - \hat{H}\hat{x}_t) \\ \hat{x}_{t+1} &= \eta_t(r_t, \tau_t^2; \phi_t, \xi_t) \\ v_t^2 &= \frac{\|y - \hat{H}\hat{x}_t\|_2^2 - \mathbf{tr}(R_{\hat{n}\hat{n}})}{\mathbf{tr}(\hat{H}^H \hat{H})} \\ \tau_t^2 &= \frac{1}{N_t}\mathbf{tr}(C_t C_t^H) v_t^2 + \frac{\theta_t^2}{N_t}\mathbf{tr}(W_t R_{\hat{n}\hat{n}} W_t^H) \end{aligned} \tag{19}$$

As it possible to see, the difference between OAMP and OAMPNet2 is represented by the presence of the learnable parameters $\Omega_t = \{\gamma_t, \phi_t, \xi_t, \theta_t\}$ in each layer. When $\gamma_t = \theta_t =$

$\phi_t = 1$ and $\xi_t = 0$ the OAMPNet2 is reduced to the OAMP detector, while optimizing the values of the parameters, the performance can be improved. The matrix $C_t = I - \theta_t W_t \hat{H}$ is similar to B_t of OAMP algorithm adding the trainable parameter θ_t in order to regulate τ_t^2. Also the OAMPNet2 algorithm is divided into two modules, a linear and a nonlinear estimator. For what concerns the linear estimator, the trainable parameter γ_t is added to the formula for updating r_t and it can be considered as the step size of the update. The nonlinear estimator η_t for estimating $\hat{x}_{t+1}$ instead is revised and it is constructed by the divergence free estimator

$$\eta_t(r_t, \tau_t^2, \phi_t, \xi_t) = \phi_t(\mathbf{E}\{x|r_t, \tau_t\} - \xi_t r_t) \quad (20)$$

where $\mathbf{E}\{x|r_t, \tau_t\}$ is computed as for OAMP. The formula can be seen as a linear combination between the priori mean r_t and the posteriori mean $\mathbf{E}\{x|r_t, \tau_t\}$, and it uses the learning parameters ϕ_t and ξ_t. For what concerns the variance estimators v_t^2 and τ_t^2, v_t^2 remains the same as for OAMP, while τ_t^2 is computed using C_t instead of B_t and using γ_t and θ_t parameters. v_t^2 can be replaced with $\max(v_t^2, \xi)$ for a small positive constant $\xi = 5 \cdot 10^{-13}$ in order to avoid stability problems. OAMP is far from optimal performance when there are strong spatial correlation and channel estimation error. With the help of trainable variables, OAMPNet2 tries to avoid this problem and to adapt to various channel environments. γ_t and θ_t are used to adjust the linear estimator and to find the optimal step size for updating r_t and τ_t^2. ϕ_t and ξ_t instead are important in order to construct the divergence free estimator $\eta_t(\cdot)$.
In Figure 1 the network is shown.

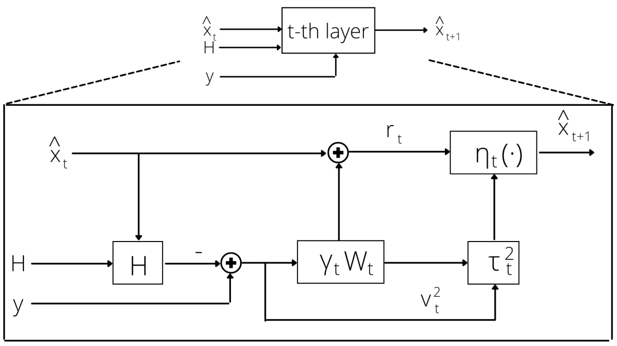

Figure 1: OAMPNet2 network layer

6.2. LVAMP

LVAMP is the neural network that results from the unfolding of the iterations of VAMP. The LVAMP network consists in two modules as VAMP that can be divided in two parts each. Therefore, four steps compose the LVAMP network, a LMMSE estimation, decoupling stage, shrinkage estimation, an another identical decoupling stage. The LMMSE stage uses as parameters $\tilde{\theta} = \{U_t, s_t, V_t, \sigma_{wt}^2\}$ for each iterations t. When the channel is not iid Gaussian and there are correlations, it is important to consider the covariance matrix. In this case the parameters of the network becomes $\tilde{\theta} = \{G_t, K_t\}$ and the LMMSE stage is defined as

$$\tilde{\eta}(\tilde{r}_t; \tilde{\sigma}_t, \tilde{\theta}_t) = G_t \tilde{r}_t + K_t y \quad (21)$$

where $G_t \in \mathcal{R}^{N_t \times N_t}$ and $K_t \in \mathcal{R}^{N_t \times N_r}$. The shrinkage stage instead has as parameter θ_t that is used in the denoising function $\eta(\cdot)$. Therefore the parameters to be learnt are expressed as $\{\tilde{\theta}_t, \theta_t\}_{t=0}^{T}$. It is suggested to inizialize U, s, V as the SVD values of H and σ_w^2 at the average value of $N_t^{-1}\|y\|^2$.
In Figure 2 it is possible to see how the network is built.

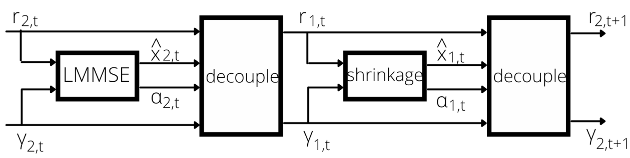

Figure 2: LVAMP network layer

6.3. MMNet

In [10] the authors propose MMNet detector, a MIMO detection scheme based on deep learning and on the theory of iterative soft-thresholding algorithms. Thanks to a novel training algorithm that leverages temporal and spectral correlation to accelerate training, MMNet outperforms existing approaches on realistic channel with the same or lower computational complexity. MMNet adds the right degree of freedom into the iterative framework, balancing model flexibility and complexity. On a iid Gaussian channel, MMNet achieves the same performances of sub-optimal detectors, with a two order less of complexity respect to other deep leaning approaches. It is also better than a classic linear scheme such as MMSE detector. The advantage of MMNet is that the algorithm is trained online, and in this way it can adapt to different channel models. There are two version of MMNet neural network, one for Gaussian channel matrices and one for arbitrary channels. For the iid Gaussian channel, the network has the following architecture:

$$\begin{aligned} z_t &= \hat{x}_t + \theta_t^{(1)} H^H (y - H\hat{x}_t) \\ \hat{x}_{t+1} &= \eta_t(z_t; \sigma_t^2) \end{aligned} \tag{22}$$

the denoiser is the same of AMP, in fact it is the optimal denoiser for Gaussian noise. One of the properties of MMNet is the assumption that the noise at the input of the denoiser follows the same distribution for all the transmitted symbols. The estimation of the noise variance σ_t^2 is given by

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

MMNet assumes that the noise is composed by two parts, the residual error caused by the estimation of $\hat{x}_t$ respect to the real value of x, and by the noise of the channel n. The first composed is amplified by the linear transformation $(I - A_t H)$ while the second by A_t. For the iid Gaussian channel model, only two parameters per iteration ($\theta_t^{(1)}$ and $\theta_t^{(2)}$) are sufficient, and the type of channel does not require an online training. In fact, MMNet can reach good performance on iid Gaussian channels by being trained offline on randomly sampled iid Gaussian matrices. The MMNet architecture for arbitrary channel matrices is structured as follows:

$$\begin{aligned} z_t &= \hat{x}_t + \Theta_t^{(1)} (y - H\hat{x}_t) \\ \hat{x}_{t+1} &= \eta_t(z_t; \sigma_t^2) \end{aligned} \tag{23}$$

where $\Theta_t^{(1)}$ is a $N_t \times N_r$ complex-valued trainable matrix. In order to add a degree of freedom to the estimation of the noise per transmitter, the trainable parameter $\theta_t^{(2)}$ in formula σ_t^2 becomes a vector of shape $N_t \times 1$. In this way, the model can handle cases where the different transmitted symbols have different levels of noise, by scaling the noise variance by different values for each symbol. MMNet uses a flexible linear transformation for computing z_t but at the same time using the optimal denoiser for Gaussian noise. Moreover, it does not need matrix inversion that can raise the complexity.
The MMNet $t-th$ layer block is shown in Figure 3.

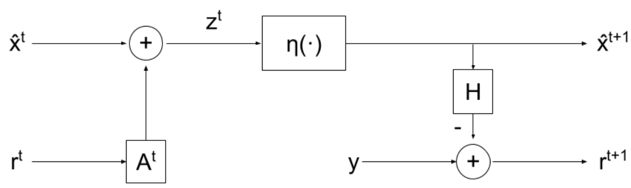

Figure 3: MMNet network layer

7. Method

The thesis is based on an experimental and quantitative methodology. The algorithms that are taken in consideration for the thesis and therefore for the experiments are MMSE, VAMP, OAMP, LVAMP, OAMPNet2, and also MMNet. While the first three algorithms are ready for being used, LVAMP, OAMPNet2 and MMNet, that are based on DL, require a training phase.
The experiments consist in different simulations based on 5G scenarios. The simulations differ for size of the MIMO system, channel model, QAM size, SNR values and type of training.
The sizes of MIMO system that are used are: 4×4, 32×32, 64×32, where for each couple the first value represents the number of receiver antennas while the second one the number of transmitters.
The channel models on which the algorithms are trained and tested are the iid Gaussian and the Kronecker ones. For what concerns the Kronecker channel model, different correlation values at both receiver and transmitter side are considered. In particular, the correlation values that are considered are $0.1, 0.3, 0.5, 0.7$ for both sides and the simulations differ for different combinations of correlation at receivers and correlation at transmitters.
The modulation scheme that are used in the experiments are different types of QAM. In particular QAM-16 and QAM-64 are the values used

to create different simulations.
For each experiment, the same range of SNR values are considered. The range that has been selected is from 18 dB to 23 dB, that are the values that are common in the comparison of MIMO detectors. Therefore, the same configuration of parameters that composes a simulation is tested on this range of SNR with a step size of 1 dB.
Finally, the algorithms, in particular the ones that are DL-based, are trained and tested in three different ways:

- The training phase is conducted with iid Gaussian channel matrices, and tested with matrices generated from the same channel model.
- The training phase uses Kronecker channel matrices with different correlation parameters, and the testing is conducted with Kronecker channel matrices.
- The training is based on iid Gaussian channel model, but the testing is done with Kronecker matrices in order to verify the adaptability of the algorithms.

The metric that is used to compare the algorithms in the experiments is the SER performance metric.

7.1. Performance Metrics

The performance metrics that are usually used while working with MIMO detection problem are the Bit Error Rate (BER) and the SER, at different SNR values. Both the metrics are a division between the number of errors in the estimated message $\hat{x}$ compared to the original transmitted message x and the number of values transmitted. In particular, the BER is defined as

$$BER = \frac{no.ofbitsinerror}{totalno.oftransmittedbits} \tag{24}$$

While SER is defined as

$$SER = \frac{no.ofsymbolsinerror}{totalno.oftransmittedsymbols} \tag{25}$$

Therefore BER works on the bit level, while SER on the constellation symbols. In the thesis, only the SER metric is used.

7.2. Dataset

The training dataset is split in parts of equal size called batches before the start of the learning procedure. When the training phase starts the algorithm iterates over epochs. For each epoch, a training step is repeated for every batch that compose the training dataset. For each experiment, the dataset is composed by samples with three sources of randomness: the signal x, the channel matrix H and the noise n. The signal x is sampled form the constellation randomly and uniformly. The channel matrix is sampled following the structure of the channel model selected for the experiment. The noise is derived from the sampling of the standard deviation sigma that is derived from the SNR of the experiment. For each sample in the batch, the SNR value is chosen randomly in the range 18-23 dB. Therefore, each batch that is used during training is a tuple of four elements $\{(y^{(d)}, H^{(d)}, \sigma^{(d)}, x^{(d)})\}_{d=1}^{D}$ where $x^{(d)}$ are the values that have to be predicted. In this thesis, the algorithms are trained offline with randomly sampled iid Gaussian channel matrices. Then they are tested on both iid Gaussian and Kronecker channel matrices with same source of randomness and different random seed. Finally, the algorithms are trained also on Kronecker channel matrices and tested using the same channel model for generating different matrices.
The generated training dataset is split in two parts in order to create the validation set, that can be used for early stopping and for cross validation. The size of the validation set is 25% of the generated dataset. In order to avoid overfitting, early stopping, dropout and cross validation are used. The algorithms have been trained for 2000 epochs, with Adam optimizer and learning rate of 0.001. Each batch has size of 1000 samples.

7.2.1 Early Stopping

Early stopping is a criteria that decides when to stop the training of the model according to the prediction error on the validation set. During the training of the model, the prediction error on the training set continues to decrease because this is the goal of the phase. However, the prediction error on the validation set initially decreases but after a while it stops to go down and starts to raise. When the prediction error on the validation dataset starts to increase, this means that the model is overfitting on the training data, therefore it is better to stop the train-

Table 1: Experiments summary

Experiment	N_r	N_t	Shape	Modulation	C.M. training	C.M. testing
1	4	4	Squared	QAM-16	i.i.d Gaussian	i.i.d Gaussian
2	32	32	Squared	QAM-64	i.i.d Gaussian	i.i.d Gaussian
3	32	32	Squared	QAM-64	Kronecker	Kronecker
4	32	32	Squared	QAM-64	i.i.d Gaussian	Kronecker
5	64	32	Rectangular	QAM-64	i.i.d Gaussian	i.i.d Gaussian
6	64	32	Rectangular	QAM-64	Kronecker	Kronecker
7	64	32	Rectangular	QAM-64	i.i.d Gaussian	Kronecker

ing procedure.

7.2.2 Droupout

Dropout is applicable to any layer t of the neural network excepts for the last layer, so the output layer. At every stage of the training phase, each input y_t can drop out temporarily with a probability p, that is called as dropout rate. When an input drops out, its value is set to zero and it will not have a contribution during that training step.

7.2.3 Cross Validation

During the training phase, with deep learning, not only the values of the parameters $V_1, \ldots, V_T$ have to be optimized, but also the hyperparameters of the model. These parameters cannot be optimized with the backpropagation step but they require to be tuned in order to achieve better performance. The hyperparameters are for example the number of layers, the learning rate, the hidden dimension of each layer, the dropout rate. The cross validation technique consists in validating the performance of different models composed by predefined combinations of hyperparameters on the validation set, at the end of the training. The model with the combination of hyperparameters that achieves the best performances on the validation set is the best candidate to be used for testing.

7.3. Experiments

The number of combinations that can be generated and that have been tested during the thesis work are hundreds. In the thesis only the most significant ones are reported. Seven experiments have been described in the thesis and they can be summarized through Table 1.

The first column of the table states the number of the experiment. N_r and N_t indicates the number of receivers and the number of transmitters respectively. The fourth column states if the MIMO system has a squared or rectangular shape. The column named modulation indicates which modulation scheme the experiment adopts. The last two columns represents the channel models (indicated with the acronym C.M.) that are used for the training and the testing phase respectively.

8. Results and Analysis

In this section, the results of the experiments are shown and analysed. Due to the fact that the 32 transmitters and 64 receivers configuration is the most common and realistic for comparing algorithms in a MIMO scenario, only the results of Experiment 5, 6 and 7 are shown in this executive summary. The results are presented in a graphical way where the horizontal axis represents the different values of SNR expressed in dB on which the algorithms are tested and the vertical axis instead expresses the SER metric in a logarithmic scale.

8.1. Experiment 5

The number of receivers is 64 and the transmitters are 32, therefore the shape is rectangular. The training phase is conducted through iid Gaussian channel matrices, and the same channel model is used for the testing phase. This experiment is the most used when dealing with comparison among MIMO detectors. In Figure 4 the results of the experiment are presented. As it possible to see, all the algorithms have very good performances, with SER values that are always (a part from 18 dB of SNR) under 0.1. The OAMPNet2 algorithm outperforms the others, being the only one having all SER under 0.1, and reaching 10^{-4} for 23dB of SNR. VAMP and LVAMP algorithms instead does not perform well in this configuration, with values that are very similar to the one of MMSE and curves that overlap with the baseline. The OAMP algorithm is the second best algorithm with great results despite it is not a DL-based algorithm. For what concerns MMNet instead, it has a particular curve, very different from the others. In fact, at the beginning the curve and the SER values are very close to the ones of OAMP, but when the SNR values increase, it does not follow the OAMP curve, having an improvement of accuracy lower. Interesting is the case of 23 dB, where the performance of MMNet is worse than all the other algorithms, including MMSE. The MMNet algorithm, compared to the others, seems to be less adaptable to changes of SNR, while the others have a more regular shape.

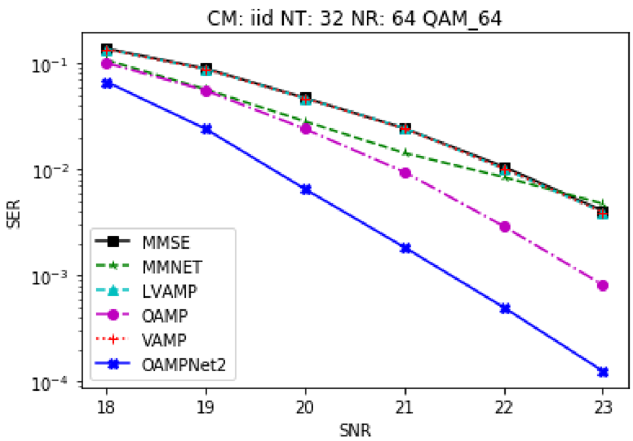

Figure 4: Experiment 5 graphical results: the graph represents the SER values for the different SNR values for each algorithm, in a 64 receivers and 32 transmitters MIMO system, trained and tested on i.i.d. Gaussian channel matrices using QAM-64 as modulation scheme

8.2. Experiment 6

Experiment 6 is divided in two cases, that are based on training and testing phases conducted with Kronecker channel matrices.

8.2.1 Experiment 6a

The first case consists in an experiment where the DL-based algorithms are trained with Kronecker channel matrices and then all the algorithms are tested on a a Kronecker channel model with $\rho_R = \rho_T = 0.3$. In Figure 5 the results of the experiment are reported. At first sight, what is evident is the strange behaviour of MMNet. In fact, the MMNet algorithm performs very badly, with SER values that are very high. This algorithm, as said before, is fragile when trained offline and this experiments shows the difficulties for MMNet to adapt to a more complex scenario. For what concerns the other algorithms, they behave similarly to experiment 5. In fact, also in this case VAMP and LVAMP have performances very close to the ones of MMSE, and graphically they appear overlapped. OAMP provides again good results, while OAMPNet2 outperforms the other algorithms. As it possible to see, OAMPNet2 is the only algorithm that reaches a SER value close to 10^{-4} with a SNR value of 23 dB. The SER values are a bit worse than the ones in experiment 5, but this is due to a more complex and realistic MIMO system in which the experiment is conducted.

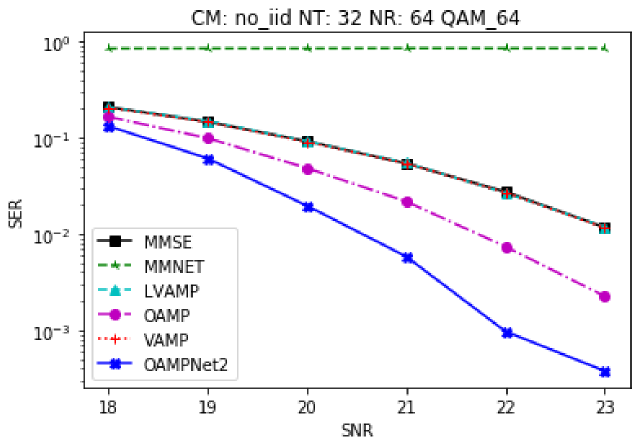

Figure 5: Experiment 6a graphical results: the graph represents the SER values for the different SNR values for each algorithm, in a 64 receivers and 32 transmitters MIMO system, trained with Kronecker channel matrices and tested on a Kronecker channel model with $\rho_R = \rho_T = 0.3$ using QAM-64 as modulation scheme

8.2.2 Experiment 6b

The second case changes the correlation parameters respect to the experiment 6a. In fact in this case $\rho_R = \rho_T = 0.5$. The result of the experiment are presented in Figure 6. This time MMNet has SER values very close to 1, meaning that quite all the estimations are wrong. With an highly correlated Kronecker channel model, MMNet is not able to estimate the transmitted signal if trained offline. An interesting result of the experiment is represented by OAMPNet2. In fact for the first time, it has a behaviour very close to the one of OAMP for SNR equal to 18, 19 and 20 dB. For the last three values of SNR, instead, OAMPNet2 performs better than the others. Again, VAMP and LVAMP have accuracy very close to the one of MMSE, overlapping graphically. Also OAMP is closer to MMSE for 18dB of SNR, but then its performances improve.

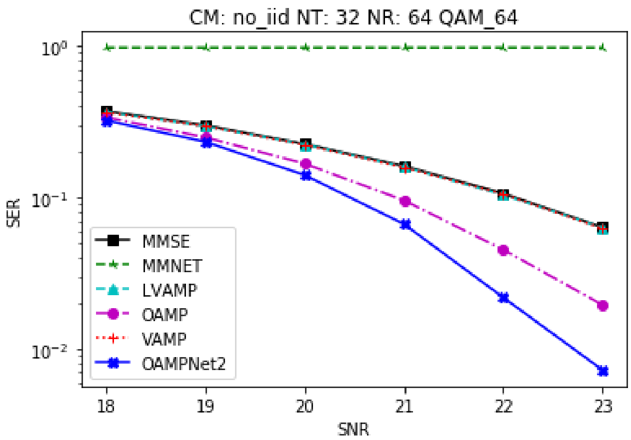

Figure 6: Experiment 6b graphical results: the graph represents the SER values for the different SNR values for each algorithm, in a 64 receivers and 32 transmitters MIMO system, trained with Kronecker channel matrices and tested on a Kronecker channel model with $\rho_R = \rho_T = 0.5$ using QAM-64 as modulation scheme

8.3. Experiment 7

The final experiment is based on a training phase conducted with iid Gaussian channel matrices and then the algorithms are tested on a Kronecker channel model. This represents the innovative part of the thesis. Thanks to this experiment it is possible to analyse the adaptability of the DL algorithms, trained with a simple channel model, and tested on a more complex one. Also in this experiment, two cases are analysed.

8.3.1 Experiment 7a

The first case uses $\rho_R = \rho_T = 0.3$ as correlation parameters for the Kronecker channel model. The results of the experiment are shown in Figure 7. The results are very similar to the ones obtained in experiment 6a, with some SER values that are also better. Again, MMNet is not able to adapt to a more complex channel model, having high SER values for each SNR considered. VAMP and LVAMP performs as MMSE as seen in the previous experiments with the 32×64 MIMO configuration. OAMPNet2, despite the training phase conducted with iid Gaussian channel matrices, outperforms the other algorithms. Again it reaches a value of SER in order of 10^{-4} for SNR equal to 22 dB and 23 dB.

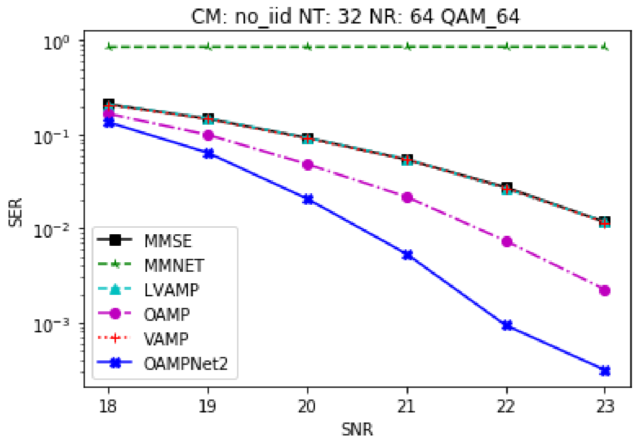

Figure 7: Experiment 7a graphical results: the graph represents the SER values for the different SNR values for each algorithm, in a 64 receivers and 32 transmitters MIMO system, trained with i.i.d. Gaussian channel matrices and tested a Kronecker channel model with $\rho_R = \rho_T = 0.3$ using QAM-64 as modulation scheme

8.4. Experiment 7b

The second case and last experiment is conducted testing the algorithms on a Kronecker channel model with correlation parameters $\rho_R = \rho_T = 0.5$, therefore an highly correlated channel. The DL-based algorithms are again trained with iid Gaussian channel matrices. The results of the experiment are very similar to the ones of experiment 6b and they can be seen in and Figure 8. MMNet again misses the estimations of all the points, achieving a SER very close to 1 for all the

SNR values. Also in this case, VAMP, LVAMP and MMSE perform the same, with overlapping curves in the graph. OAMPNet2 is also in this case the best algorithm, but for low SNR values it performs like OAMP. For higher SNR values instead it improves its performances. Also OAMP has performances that are a bit worse for the lowest SNR values, and better when SNR increases.

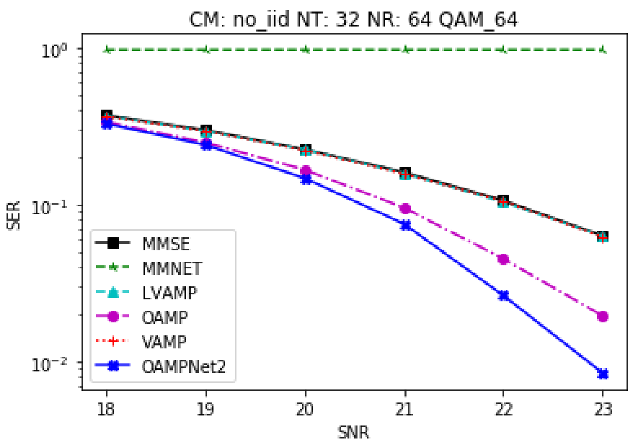

Figure 8: Experiment 7b graphical results: the graph represents the SER values for the different SNR values for each algorithm, in a 64 receivers and 32 transmitters MIMO system, trained with i.i.d. Gaussian channel matrices and tested a Kronecker channel model with $\rho_R = \rho_T = 0.5$ using QAM-64 as modulation scheme

9. Complexity Analysis

The algorithms that are taken in consideration in the experiments have different complexity. MMSE has a complexity in $\mathcal{O}(N_r^3)$ because it is dominant by the matrix inversion of the channel matrix. VAMP and LVAMP have a complexity in $\mathcal{O}(TN_r^3)$ due to the SVD decomposition, but with some new implementations they can be in $\mathcal{O}(2TN_tN_r)$ using an economy-SVD. For VAMP, T represents the number of iterations, while for LVAMP it is the number of layers of the network. For what concerns OAMP and OAMPNet2, the matrix inversion is again dominant in the detection, therefore the complexity is in $\mathcal{O}(TN_r^3)$, where T are the number of iterations for OAMP, and the number of layers of the network for OAMPNet2. Finally, MMNet has a complexity in $\mathcal{O}(TN_r^2)$ for the detection. The number of iterations or layers in order to the algorithms to converge is different, impacting the complexity. For MMSE, there is only an iteration since it is not an iterative algorithm. VAMP and LVAMP converge in 5 or 6 iterations/layers. OAMP and OAMPNet2 converge faster, since they converge in 4 or 5 iterations/layers. MMNet is the slower algorithm of this group. In fact it needs from 10 to 14 layers to converge, impacting the complexity of the algorithm. A summary of this complexity analysis is shown in Table 2 where each column represents an algorithm, the first row the computational complexity and the second row the number of iterations/layers T for convergence.

10. Conclusions

Both LVAMP and OAMPNet2 algorithms can be considered sub-optimal solutions for the MIMO detection problem. In fact, LVAMP can be built with a complexity lower than MMSE and it obtains SER values that are equal or better than MMSE in all the scenarios considered for the experiments. Also OAMPNet2 is a sub-optimal solution to the problem because it outperforms MMSE in all the scenarios considered, having a complexity that is just a few greater than MMSE, thanks also to a very fast convergence.

In terms of complexity LVAMP is better than OAMPNet2, and also of MMNet. The performances are very interesting for this complexity and convergence speed. Therefore, it is perfect for a MIMO system in which the complexity should be low, but with an accuracy at least equal to the one of MMSE.

For what concerns the SER performances, OAMPNet2 has incredible performances, outperforming all the algorithms considered in every scenarios. This network is perfect for a MIMO system in which the detection does not represents the complexity bottleneck. It is a very accurate algorithm with performances often near also to ML, with a complexity only four or five times the one of MMSE. It could improve the MIMO detection problem, becoming a serious candidate as detector.

Another important insight discovered thanks to the experiments is that the two deep networks are able to adapt very well to different conditions. In fact they perform well on both iid Gaussian and Kronecker channel models. They

Table 2: Complexity and number of iterations/layers for convergence of each algorithm

	MMSE	MMNet	VAMP	LVAMP	OAMP	OAMPNet2
Complexity	$\mathcal{O}(N_r^3)$	$\mathcal{O}(TN_r^2)$	$\mathcal{O}(2TN_rN_t)$	$\mathcal{O}(2TN_rN_t)$	$\mathcal{O}(TN_r^3)$	$\mathcal{O}(TN_r^3)$
T	1	10-14	5-6	5-6	4-5	4-5

are also able to perform well on realistic channel models like the Kronecker one, with both medium and high correlation parameters. Moreover, they can perform well on this type of channel, also with a simple training phase conducted with iid Gaussian channel matrices.
The drawbacks of this work is that in order to obtain the performances achieved by OAMPNet2, an high complexity is still needed despite a deep learning approach. At the same time, lowering the complexity thanks to LVAMP does not bring a strong improvement of the performances. Therefore, despite deep learning can be an interesting and promising solution for MIMO detection, a deep learning that can face better the complexity-performance trade-off is still missing.

11. Future work

In this section, the future works and directions that can follow this thesis work, are presented.
The first future work that should be done is to try to find new methods in order to solve the MIMO detection problem.
A second direction can be to compare and verify that all the different version of VAMP and LVAMP performs the same or if there is a version that provides better results than the others.
Trying to rethink the OAMPNet2 algorithm in order to lowering its complexity can be an hard but useful future work for trying to have a very competitive detector.
Another interesting future work is to apply all the algorithms presented in the thesis in the 3GPP channel model, that is the most realistic scenario for simulating link layer transmission in MIMO systems.
As modulation scheme and SNR values, in the thesis, only limited values have been tested. A simple future work can be testing the algorithms also using other QAM types as modulation scheme. Moreover, the experiment can be conducted on a wider range of SNR values in order to discover strange behaviour of the curves of the algorithms. On the same direction, also different sizes of MIMO system can be tested, testing the algorithms also on bigger systems and with different $\frac{N_t}{N_r}$ ratios.
Another direction can be changing the loss function during the training phase of the DL-based algorithms. Different loss functions can bring different results and new insights on LVAMP and OAMPNet2.
Finally, even if in the thesis it has already been done, trying different numbers of iterations/layers for the four proposed algorithms can help in finding the best trade-off between performances and complexity.

References

[1] Yanna Bai, Wei Chen, Jie Chen, and Weisi Guo. Deep learning methods for solving linear inverse problems: Research directions and paradigms. *Signal Processing*, 177:107729, Dec 2020.

[2] Mark Borgerding, Philip Schniter, and Sundeep Rangan. Amp-inspired deep networks for sparse linear inverse problems. *IEEE Transactions on Signal Processing*, 65(16):4293–4308, Aug 2017.

[3] Mischa Dohler and Takehiro Nakamura. *5G Mobile and Wireless Communications Technology*. Cambridge University Press, 2016.

[4] David L. Donoho, Arian Maleki, and Andrea Montanari. Message passing algorithms for compressed sensing: II. analysis and validation. *CoRR*, abs/0911.4222, 2009.

[5] David L. Donoho, Arian Maleki, and Andrea Montanari. Message passing algorithms for compressed sensing: I. motivation and construction. In *2010 IEEE Information Theory Workshop on Information Theory (ITW 2010, Cairo)*, pages 1–5, 2010.

[6] Hengtao He, Chao-Kai Wen, Shi Jin, and Geoffrey Y. Li. A model-driven deep learning network for mimo detection. *2018 IEEE Global Conference on Signal and Information Processing (GlobalSIP)*, pages 584–588, 2018.

[7] Hengtao He, Chao-Kai Wen, Shi Jin, and Geoffrey Ye Li. A model-driven deep learning network for mimo detection. *IEEE Transactions on Signal Processing*, 68:1702–1715, Feb 2020.

[8] Seyyed Akbar Hodudi Atigh, Jafar Pourrostam, and Behzad Mozaffary Tazeh kand. Uplink massive mimo detector based on vector approximate message passing. In *2020 10th International Conference on Computer and Knowledge Engineering (ICCKE)*, pages 388–391, 2020.

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

[10] Mehrdad Khani, Mohammad Alizadeh, Jakob Hoydis, and Phil Fleming. Adaptive neural signal detection for massive mimo. *IEEE Transactions on Wireless Communications*, 19(8):5635–5648, 2020.

[11] Xiangfeng Liu and Ying Li. Deep mimo detection based on belief propagation. In *2018 IEEE Information Theory Workshop (ITW)*, pages 1–5, 2018.

[12] Song Noh, Michael D. Zoltowski, Youngchul Sung, and David J. Love. Pilot beam pattern design for channel estimation in massive mimo systems. *IEEE Journal of Selected Topics in Signal Processing*, 8(5):787–801, 2014.

[13] Sundeep Rangan. Generalized approximate message passing for estimation with random linear mixing, 2012.

[14] Sundeep Rangan, Philip Schniter, and Alyson K. Fletcher. Vector approximate message passing. *CoRR*, abs/1610.03082, 2016.

[15] Andrea Scotti. Graph neural networks and learned approximate message passing algorithms for massive mimo detection, 2020.

[16] Xiaosi Tan, Weihong Xu, Yair Be'ery, Zaichen Zhang, Xiaohu You, and Chuan Zhang. Improving massive mimo belief propagation detector with deep neural network, 2018.

[17] Sheng Wu, Linling Kuang, Zuyao Ni, Jianhua Lu, Defeng Huang, and Qinghua Guo. Low-complexity iterative detection for large-scale multiuser mimo-ofdm systems using approximate message passing. *IEEE Journal of Selected Topics in Signal Processing*, 8(5):902–915, 2014.

[18] 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.

[19] Suchun Zhang, Mengjiao Zhang, Shi Jin, and Chao-Kai Wen. Low-complexity detection for mimo c-fbmc using orthogonal approximate message passing. *IEEE Signal Processing Letters*, 26(1):34–38, 2019.

[20] Zhaoyang Zhang, Xiao Cai, Chunguang Li, Caijun Zhong, and Huaiyu Dai. One-bit quantized massive mimo detection based on variational approximate message passing. *IEEE Transactions on Signal Processing*, PP:1–1, 12 2017.

[21] Peicong Zheng, Yuan Zeng, Zhenrong Liu, and Yi Gong. Deep learning based trainable approximate message passing for massive mimo detection. *ICC 2020 - 2020 IEEE International Conference on Communications (ICC)*, 2020.