UNIVERSIDAD
NACIONAL
DE COLOMBIA

Direct Detection with phase recovery in optical communications

Diego Alejandro Figueroa Del Castillo

Universidad Nacional de Colombia
Facultad de Ingenieria , Departamento de Eléctrica y Electrónica
Bogotá, Colombia
2023

This page intentionally left blank.

Acknowledgments

This page intentionally left blank.

Abstract

This page intentionally left blank.

Contents

List of Figures

This page intentionally left blank.

List of Tables

This page intentionally left blank.

List of symbols, abbreviations and notation

$\mathbf{0}_n$	Ceros vector of length n
$\boldsymbol{I}_n$	Identity matrix of size $n \times n$
$\mathcal{N}(\boldsymbol{x};\boldsymbol{\mu},\boldsymbol{C})$	Multivariate real Gaussian distributions, with mean vector μ and covariance matrix $\boldsymbol{C}$.
$\mathcal{N}(x;\mu,\sigma^2)$	Univariate real Gaussian distributions, with mean μ and variance σ^2.
$\mathcal{F}\{\cdot\}$	Fourier transform
$\mathcal{F}^{-1}\{\cdot\}$	Inverse fourier transform
$\mathbb{E}[X]$	Expected value of X
f	Frequency
j	Imaginary unit
$X \sim \mathcal{N}(\mu,\sigma^2)$	X has a Gaussian distribution with mean μ and variance σ^2
APP	a posteriori probability
AWGN	Additive white Gaussian noise
BER	Bit error rate
DD	Direct Detection
diag($\boldsymbol{x}$)	Diagonal matrix with the elements of $\boldsymbol{x}$ along the diagonal
i.i.d.	independent and identically distributed
Im$\{z\}$	imaginary part of z
IM-DD	Intensity modulation - Direct detection
MAP	Maximum a posteriori probability
MIMO	Multiple input multiple output
PDF	Probability density function
Re$\{z\}$	Real part of z
SER	Symbol error rate
sinc(t)	$\frac{\sin(\pi t)}{\pi t}$
SNR	Signal to noise ratio
SQL	Square law

This page intentionally left blank.

Chapter 1

Introduction

Square-law detection, also known as Direct detection (DD), is a detection scheme that measures the square magnitude of a complex wave form, clearly this scheme is a nonlinear waveform detection because of the square operation. Direct detection is widely used in different scientific fields, for example in crystallography, radio astronomy, biomedical spectroscopy, among others [1]. In particular this scheme is often used in optical communications, specially in short-haul systems with length up to 50 km [2] due to its simplicity, or even in shorter links up to 10 km for example in rack to rack communications in big data centers [1].

In the recent years the interest of studying systems with DD is getting bigger again, because of the simplicity of the receivers, which are a promising low-cost alternative compared to the coherent detection systems [3]. In this context two questions or problems arise. First, how good is a system with DD compared to a system with coherent detection in terms of the information capacity of the channel. And second, how to design a system that exploits the information capacity of the DD channel in the best possible way.

In this work we cover briefly two answers given to the first question, and then we review two systems proposed for a channel with DD. Then we propose a new decoder for the second system, with reduced complexity at the expenses of a slightly worse performance.

1.1 Direct Detection

In optical communication DD is performed with a single photodiode, that convert the optical signal to an electric signal trough the photoelectric effect according to the next equation [4]:

$$I_p = R_d \cdot P_{in} \tag{1.1}$$

where I_p is the photocurrent, P_{in} is the incident optical power (which is proportional to the square of the magnitud of the electric field, that is where the square-law term comes from), and R_d is the so called responsivity of the photodetector, with units of A/W.

The noise in the photodiode are generated primarily by two mechanisms, in the first place the shot noise, and in the second place thermal noise.

The shot noise models the fact that the photocurrent consist of a stream of electrons generated at random times. Mathematically the current corresponding to the shot noise $i_s(t)$ is a stationary random process with Poisson distribution, but is usually approximated by a Gaussian distribution with variance given by [4]:

$$\sigma_s^2 = 2qI_pB \tag{1.2}$$

where q is the electron charge, I_p the photocurrent, and B the bandwidth of the system. σ_s can be interpreted as the RMS value of the shot noise current $i_s(t)$.

The termal noise is generated by the movement of electrons due to the ambient temperature, this kind of noise is Gaussian distributed, and its a variance is given by [4]:

$$\sigma_{th}^2 = \frac{4k_BT}{R_L}F_nB \tag{1.3}$$

where k_B is the Boltzmann constant, T is the temperature given in kelvin, R_L is the load resistance, B the bandwidth, and F_n is the amplifier noise figure.

With this in mind the output current of the direct detection process is given by:

$$I(t) = I_p + i_s(t) + i_{th}(t) \tag{1.4}$$

with $i_s(t) \sim \mathcal{N}(0, \sigma_s^2)$ and $i_{th}(t) \sim \mathcal{N}(0, \sigma_{th}^2)$.

1.2 Capacity under direct detection

A communications channel that uses DD can retrieve only the information about the magnitude of the signal, in contrast a system with coherent detection can retrieve the magnitud and phase of the signal. This means that DD scheme ignores one of the two degrees of freedom, and hence it is reasonable to think that the capacity of this systems should be approximately half that of the system with coherent detection [1], [3], [5].

However in [3] it is shown that the spectra efficiency of a band limited system under DD is at most 1 bit/s/Hz less than the same system under coherent detection. Also in [5] it is proven that for time limited signals the capacity is also at most one bit less than the coherent case. This means that contrary to intuition, the loss in the capacity of a system under DD is not a half of the coherent system, but just 1 bit/s/Hz.

The results of this papers show that the systems with DD have a big potential, because the detector is cheaper and easier to implement (basically just one photodiode) and the loss in the capacity may not be as big as thought. However the problem to find a system simple enough that uses the potential of the DD is still open.

1.3 Basic principle of phase recovery

The key to retrieve the phase information of the transmitted symbols when using DD is to make use of the ISI. As a toy example one can think on the problem where given two complex numbers z_1 and z_2; from $|z_1|^2$, $|z_2|^2$ and $|z_1+z_2|^2$ (an ISI term), it is possible to determine the phase difference between z_1 and z_2 up to a sign ambiguity [1].

To show this, notice the following:

$$\begin{aligned} |z_1+z_2|^2 &= (z_1+z_2)(z_1+z_2)^* \\ &= z_1z_1^* + z_1z_2^* + z_2z_2^* + z_1^*z_2 \\ &= |z_1|^2 + |z_2|^2 + z_1z_2^* + \left(z_1z_2^*\right)^* \\ &= |z_1|^2 + |z_2|^2 + 2\mathrm{Re}\{z_1z_2^*\} \end{aligned}$$

under the convention that $z_1 = a \cdot e^{j\alpha}$ and $z_2 = b \cdot e^{j\beta}$

$$|z_1+z_2|^2 = |z_1|^2 + |z_2|^2 + 2|z_1||z_2|\cos(\alpha-\beta) \tag{1.5}$$

Clearly if $|z_1|^2$, $|z_2|^2$ and $|z_1+z_2|^2$ are known, one can solve the equation 1.5 for $\cos(\alpha-\beta)$ and get the information about the phase difference between z_1 and z_2. However, since cosine is an even function $\cos(\alpha-\beta) = \cos(-\alpha+\beta)$, hence there is still an ambiguity on the sign of the phase difference.

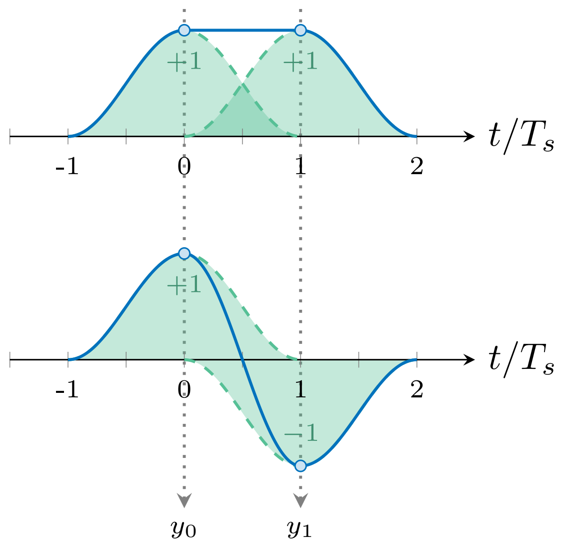
To visualize this principle in a real system (jet not even near to optimal due to the big bandwidth of the pulse), see the figure 1.1. There is shown with a solid line the wave form of a simple transmission under coherent detection (see figure 1.1a) and under direct detection (see figure 1.1b), and also the wave form of the individual symbols in dashed lines.

For the coherent detection case the samples at each symbol time are sufficient to determine the transmitted symbols. In contrast, for the direct detection case the samples at each symbol time (circles) only carry information about the magnitude of the symbols (always one in this example); and the samples at intermediate symbol time (squares) carry information about the phase difference [6].

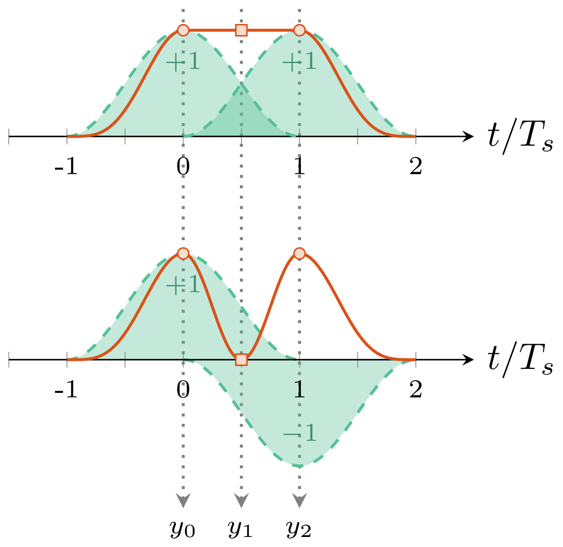
Figure 1.1c show the DD system for a longer sequence, there it is possible to notice that the even samples always carry information about the magnitude, whereas the odd samples carry information about the differential phase of the symbols. This toy example shows the principle behind the phase recovery in DD systems.

It is important to highlight two things, first an oversampling factor of two is needed, and second only information on the differential phase (up to a sign ambiguity) is retrieved, hence it is beneficial to use differential coding in the transmission.

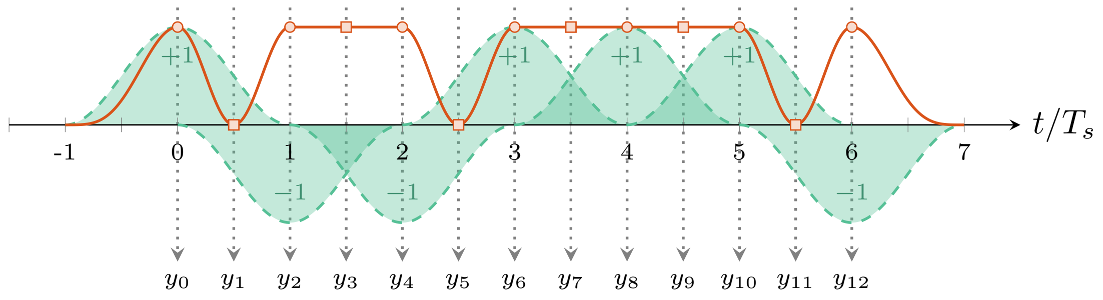
An other way to explain the previous conclusions is to take a look at the simple system



(a) Coherent detection



(b) Direct detection



(c) Direct detection of the sequence $\{+1, -1, -1, +1, +1, +1, -1\}$

Figure 1.1: Comparative of the wave forms under coherent detection and direct detection. Based on [6], [7].

given by:

$$g(t) = \sum_{k=0}^{m} g_k \text{sinc}(t-k)$$

where g_k is a complex number representing the k-th transmitted symbol, and $g(t)$ is the transmitted signal. Note that after the DD the bandwidth of the signal is twice as big as the bandwidth of $g(t)$, since the received signal is given by the product $|g(t)|^2 = g(t)g^*(t)$. Hence to recover the data one should use an oversampling factor of two, so the sampled signal becomes [1]:

$$\left|g\left(\frac{n}{2}\right)\right|^2 = \begin{cases} \left|g_{\frac{n}{2}}\right|^2 & \text{if } n \text{ is even} \\ \left|\sum\limits_{k=0}^{m} g_k \text{sinc}\left(\frac{n}{2}-k\right)\right|^2 & \text{if } n \text{ is odd} \end{cases} \qquad \text{for } n = 0, \cdots, 2m$$

Once again, it is clear that the even samples carry the information about the magnitude of the symbols, whereas the odd samples carry some how the information about the phase of the symbols, but now notice that all the g_k contribute to the odd samples. As we will see in later chapters, this makes that retrieving the phase becomes quickly an intractable problem as m grows [1].

This page intentionally left blank.

Chapter 2

Tukey signaling

One of the solutions given to the problem of phase recovery under DD is the so called Tukey signaling proposed in [1]. The main idea in this paper is to introduce controlled ISI in the transmission and use the interference between symbols to recover the phase of the symbols in a block of length n symbols. The system is proposed for short-haul systems, specifically for links with lengths less than 10 km.

The system model proposed in the paper is shown in figure 2.1. In the following sections each block will be explain and discussed when needed. Nevertheless the blocks will be explained in a different order, to facilitate understanding. First the dispersion precompensation, the transmission medium and the photodiode will be explained, then the signaling block, integrate and dump, class representative, and finally the decoder.

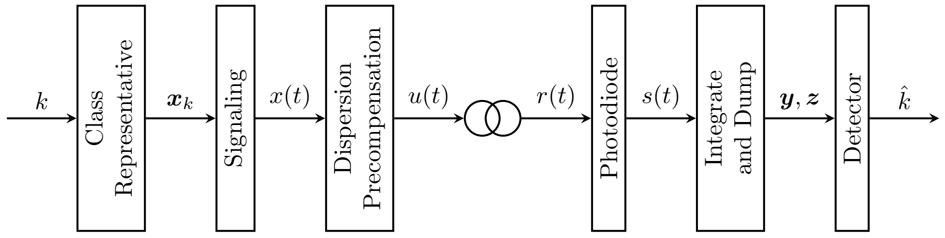


Figure 2.1: System model for Tukey signaling. Based on [1]

2.1 System model

2.1.1 Dispersion precompensation

In this block the chromatic dispersion of the optical fiber is precompensated, by an all-pass filter with transfer function

$$H(f) = e^{j2\beta_2 L\pi^2 f^2} \tag{2.1}$$

where β_2 is the group-velocity dispersion parameter and L is the fiber length [1].

Clearly from this description of the block, the following relations between $x(t)$ and $u(t)$ are given:

$$u(t) = \mathcal{F}^{-1}\big\{X(f)H(f)\big\} \tag{2.2}$$

where $X(f)$ is the Fourier transform of $x(t)$

2.1.2 Transmission medium

Since the optical fiber length is less than 10 km it is possible to ignore all the transmission impairments, except for power loss and chromatic dispersion [1]. With this in mind, and taking into account the dispersion pre compensation, we have the following relation:

$$r(t) = \rho x(t) \tag{2.3}$$

where $0 < \rho \leq 1$ is the attenuation constant, that depends on the length of the fiber. The spacial case $\rho = 1$, i.e. without attenuation, is called back-to-back transmission.

2.1.3 Photodiode

For this system an avalanche photodiode (APD) is used, and this diode is considered as the only source of noise. In this case both shoot and thermal noise are considered. The behavior of the diode is given by [1]:

$$s(t) = \big|r(t)\big|^2 + \big|r(t)\big|n_{sh}(t) + n_{th}(t) \tag{2.4}$$

where $n_{sh}(t)$ and $n_{th}(t)$ are gaussian distributed and the variance of each noise is given by [1]:

$$\sigma_{th}^2 = \frac{4k_B TB}{R_L} \tag{2.5}$$

$$\sigma_{sh}^2 = 2qM_{\text{APD}}^2 FR_{\text{APD}}B \tag{2.6}$$

Notice that the difference of between the σ_{sh}^2 presented here and the equation 1.2 is due to the gain of the APD.

2.1.4 Signaling

This block receive n complex numbers $x_0, \dots, x_{n-1}$, which are the transmitted symbols, and produces continuous time complex signal given by:

$$x(t) = \sum_{i=0}^{n-1} x_i w(t - iT) \tag{2.7}$$

where T is the inverse of the baud rate and $w(t)$ is a wave form.

Usually in communications $w(t)$ is a root raised-cosine waveform, however the ISI of this pulse at times different to the symbol time is to complicated, because a lot of symbols interfere if not all of them, and that make the problem of recovering the phase to complicated. That is why in the paper is proposed a time limited waveform with a simpler "ISI pattern".

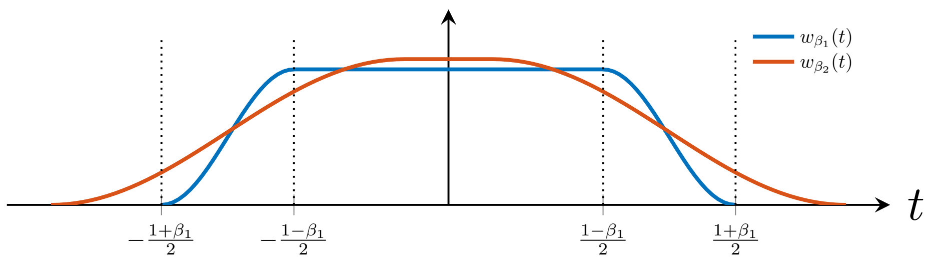
The proposed waveform is the Tukey window, which is equivalent to the Fourier transform of a raised-cosine but in time domain. This waveform is given by:

$$w_\beta(t) = \begin{cases} \frac{2}{\sqrt{4-\beta}} & \text{if } |t| \le \frac{1-\beta}{2} \\ \frac{1}{\sqrt{4-\beta}}\left(1-\sin\left(\frac{\pi(2|t|-1)}{2\beta}\right)\right) & \text{if } \left||t|-\frac{1}{2}\right| \le \frac{\beta}{2} \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

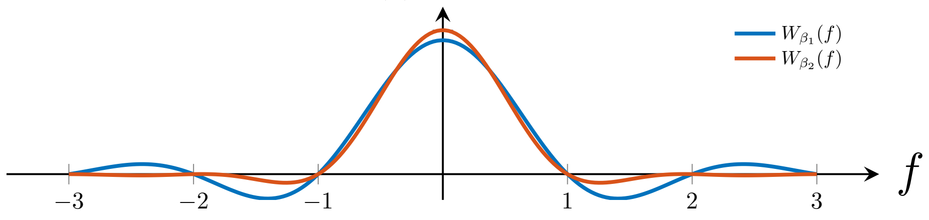
$$W_\beta(f) = \begin{cases} \frac{\pi}{2\sqrt{4-\beta}} \cdot \operatorname{sinc}\left(\frac{1}{2\beta}\right) & \text{if } f = \pm\frac{1}{2\beta} \\ \frac{2}{\sqrt{4-\beta}} \cdot \operatorname{sinc}(f) \cdot \frac{\cos(\pi\beta f)}{1-(2\beta f)^2} & \text{otherwise} \end{cases} \tag{2.9}$$

Notice that the factor $2/\sqrt{4-\beta}$ is to normalize the energy of the waveform.

The waveform can be seen in figure 2.2 both in time domain and frequency domain for two different β.



(a) Time domain



(b) Frequency domain

Figure 2.2: Tukey window for different β, $\beta_1 < \beta_2$. Based on [1]

When we select $w(t) = w_\beta(t/T)$ in equation 2.7, the resulting signal have an important property: at any given time, either only one or only two symbols contribute to the value of the signal, here we see the meaning of *controlled* ISI. According to this property we define two types of intervals, an ISI-free interval $\mathcal{Y}_i$, where $x(t)$ depends only on x_i; and an ISI-present interval $\mathcal{Z}_i$, where $x(t)$ depends only on x_i and x_{i+1}. Those intervals are shown in the figure 2.3 and are given by:

$$\mathcal{Y}_i : \left[\left(i - \frac{1-\beta}{2}\right)T, \left(i + \frac{1-\beta}{2}\right)T\right] \qquad i \in \{0, \dots, n-1\} \tag{2.10}$$

$$\mathcal{Z}_i : \left[\left(i + \frac{1-\beta}{2}\right)T, \left(i + \frac{1+\beta}{2}\right)T\right] \qquad i \in \{0, \dots, n-2\} \tag{2.11}$$

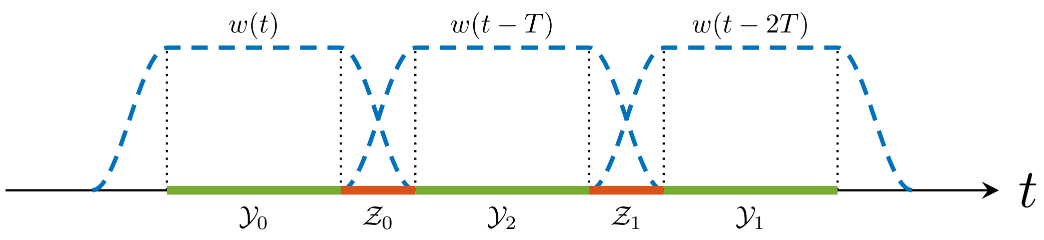


Figure 2.3: ISI-free and ISI-pressent intervals for $n = 3$

Other consequence of choosing this wave form is that, in frequency domain, the signal is not band limited, however the amplitude of $W_\beta(f)$ tends to 0 as the frequency increase, so it is possible to define the bandwidth of the waveform as the smallest interval containing the 95% of the signal energy [1]. With this definition we get the bandwidths seen in table 2.1 for different β.

β	Bandwith [Hz]	overhead
0.1	1.477	195.4%
0.3	0.788	57.6%
0.5	0.668	33.6%
0.7	0.613	22.6%
0.8	0.592	18.4%
0.9	0.575	15.0%

Table 2.1: Bandwidth containing 95%of the waveform energy, and the overhead compared to the minimum bandwidth for Nyquist signaling. Taken from [1]

2.1.5 Integrate and dump

This block integrate the incoming signal $s(t)$ in the different intervals $\mathcal{Y}_i$ and $\mathcal{Z}_i$ to produce the real valued samples y_i and z_i given by:

$$y_i = \int_{\mathcal{Y}_i} s(t)dt \qquad i \in \{0, \dots, n-1\} \tag{2.12}$$

$$z_i = \int_{\mathcal{Z}_i} s(t)dt \qquad i \in \{0, \dots, n-2\} \tag{2.13}$$

It can be shown that y_i is given by [1]:

$$y_i = \alpha^2(1-\beta)T|x_i|^2 + \alpha|x_i|n_i + m_i \tag{2.14}$$

where $\alpha = 2/\sqrt{4-\beta}$, $n_i \sim \mathcal{N}(0, \sigma_{sh}^2(1-\beta)T)$ and $m_i \sim \mathcal{N}(0, \sigma_{th}^2(1-\beta)T)$.

Also defining:

$$\psi(v, w) = \frac{1}{4}|v+w|^2 + \frac{1}{8}|v-w|^2 \tag{2.15}$$

which can be expanded using equation 1.5 as:

$$\psi(ae^{j\alpha}, be^{j\beta}) = \frac{3}{8}\left(a^2 + b^2\right) + \frac{1}{4}ab\cos(\alpha - \beta) \tag{2.16}$$

it is possible to show that z_i is given by [1]:

$$z_i = \alpha^2 \beta T \psi(x_i, x_{i+1}) + \alpha\sqrt{\psi(x_i, x_{i+1})}p_i + q_i \tag{2.17}$$

where $p_i \sim \mathcal{N}(0, \sigma_{sh}^2 \beta T)$ and $q_i \sim \mathcal{N}(0, \sigma_{th}^2 \beta T)$.

Notice that n_i, m_i, p_i and q_i are mutually independent, and that all the y_i carry information about the magnitude of the transmitted symbols, and all the z_i carry information about the interface between adjacent symbols, i.e. the phase difference between symbols.

2.1.6 Class representative

When transmitting symbol blocks (n consecutive symbols) it turns out that there are some ambiguities, that means that two different symbol blocks produces the same output of the system. Of course we want to avoid this situation to achieve a reliable communication, and that is what the Class representative block of figure 2.1 does.

To formalize the problem let define the function $\Upsilon : \mathbb{C}^n \to (\mathbb{R}^n, \mathbb{R}^{n-1})$ that maps a vector $\boldsymbol{x}$ at the input of the signaling block, to the output of the integrate and dump block, in the absence of noise [1]. That means:

$$\Upsilon(\boldsymbol{x}) = (\boldsymbol{y}, \boldsymbol{z}) \tag{2.18}$$

where $\boldsymbol{x} \in \mathbb{C}^n$, $\boldsymbol{y} \in \mathbb{R}^n$, $\boldsymbol{z} \in \mathbb{R}^{n-1}$ and

$$y_i = \int_{\mathcal{Y}_i} \left|x(t)\right|^2 dt \qquad i \in \{0, \ldots, n-1\} \tag{2.19}$$

$$z_i = \int_{\mathcal{Z}_i} \left|x(t)\right|^2 dt \qquad i \in \{0, \ldots, n-2\} \tag{2.20}$$

Now we define an equivalence relation in $\mathbb{C}^n$, where two vectors $\boldsymbol{x}$ and $\tilde{\boldsymbol{x}}$ are square law equivalent if and only if $\Upsilon(\boldsymbol{x}) = \Upsilon(\tilde{\boldsymbol{x}})$, and we denote the equivalence by $\boldsymbol{x} \equiv \tilde{\boldsymbol{x}}$ [1].

The Class representative block consist of a set $\mathcal{S}$ of cardinality M, whose elements belong to $\mathbb{C}^n$ and are all square law distinct [1], that is:

$$\mathcal{S} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_M\} \subset \mathbb{C}^n \qquad \text{such that} \qquad \boldsymbol{x}_i \not\equiv \boldsymbol{x}_l \Leftrightarrow k \neq l \tag{2.21}$$

and for an input $k \in \{1, \ldots, M\}$ the block outputs $\boldsymbol{x}_k$ [1]:

$$k \longrightarrow \text{Class Representative} \longrightarrow \boldsymbol{x}_k \qquad k \in \{1, \ldots, M\} \tag{2.22}$$

Notice that this scheme, can transmit M different symbol blocks using n symbols, thus the spectral efficiency of this channel can not exceed $\frac{1}{n}\log_2(M)$bits/s/Hz [1].

2.1.6.1 Analysis on ambiguities

There are two key points to understand the origin of the ambiguities. The first, and most important, is to notice that the detection of a symbol block is based only on the information about the magnitude of each symbol (present on each y_i) and the information about the phase difference between adjacent symbols (present on each z_i). That means that every two vectors whose elements have the same magnitude, and the phase difference between adjacent elements is the same, are square law equivalent.

The second key point is to notice that the symbols of the symbol blocks belong to a constellation $\mathcal{K}$ (some examples of possible constellation are shown on figure 2.8), that means that a $\boldsymbol{x} \in \mathcal{K}^n \subset \mathbb{C}^n$, in other words $\boldsymbol{x}$ belongs to a finite subset of $\mathbb{C}^n$.

Summarizing, all n-tuple of points in a given constellation that have the same magnitudes and same phase difference between adjacent symbols are square law equivalent and generate ambiguities. Lets define equivalence class as the set of all such vectors, so an equivalence class $\mathcal{E}$ of size L is given by:

$$\mathcal{E} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_L\} \subset \mathbb{C}^n \qquad \text{such that} \qquad \boldsymbol{x}_i \equiv \boldsymbol{x}_j \quad \forall i, j \in \{1, \ldots, L\} \tag{2.23}$$

To understand better the equivalence classes, lets look an example of equivalence class when the symbol block length is $n = 4$ and the constellation is a 5-ring 5-ary phase (see figure 2.8d) and how to construct it.

The first step is to select 4 random points of the constellation to create a symbol block. Take for example the points shown in the figure 2.4a, there the transmitted symbols are the vertices of the graph, and the order of transmission is given by the small number in each point.

Now to create a new equivalent symbol block one can reflect the third and fourth symbol along the symmetry axis that pass through the second symbol, as shown in the figure 2.4b. Notice that the firs and second symbols do not change, hence their magnitudes are the same, and the phase difference between they also. The third and fourth symbols change, but since the symmetry axis passes through the origin their magnitudes do not change, and neither does the phase difference between them. The phase difference between the second and third symbol in both symbol blocks is also the same as shown in figure 2.4b. Finally, all the points of the new symbol block lay on a valid constellation point, due to the high symmetry of the constellation. Hence the first symbol block and the new one are valid (belong to $\mathcal{K}^4$) and square law equivalent so they belong to the same equivalence class.

The same process can be applied to the fourth symbol, and using as symmetry axis the axis that passes trough the third symbol as shown in the figure 2.4c. Using the same arguments the new symbol block is equivalent to the previous two and belongs to the same equivalence class. Now from this new symbol block, reflecting the third and fourth symbol as shown in the figure 2.4d one can create a new fourth symbol block that belongs to the same equivalence class.

Now we have 4 different, but square law equivalent symbol blocks, clearly if we reflect all the 4 symbols blocks along the real axis, we create 4 new different symbol blocks that belong

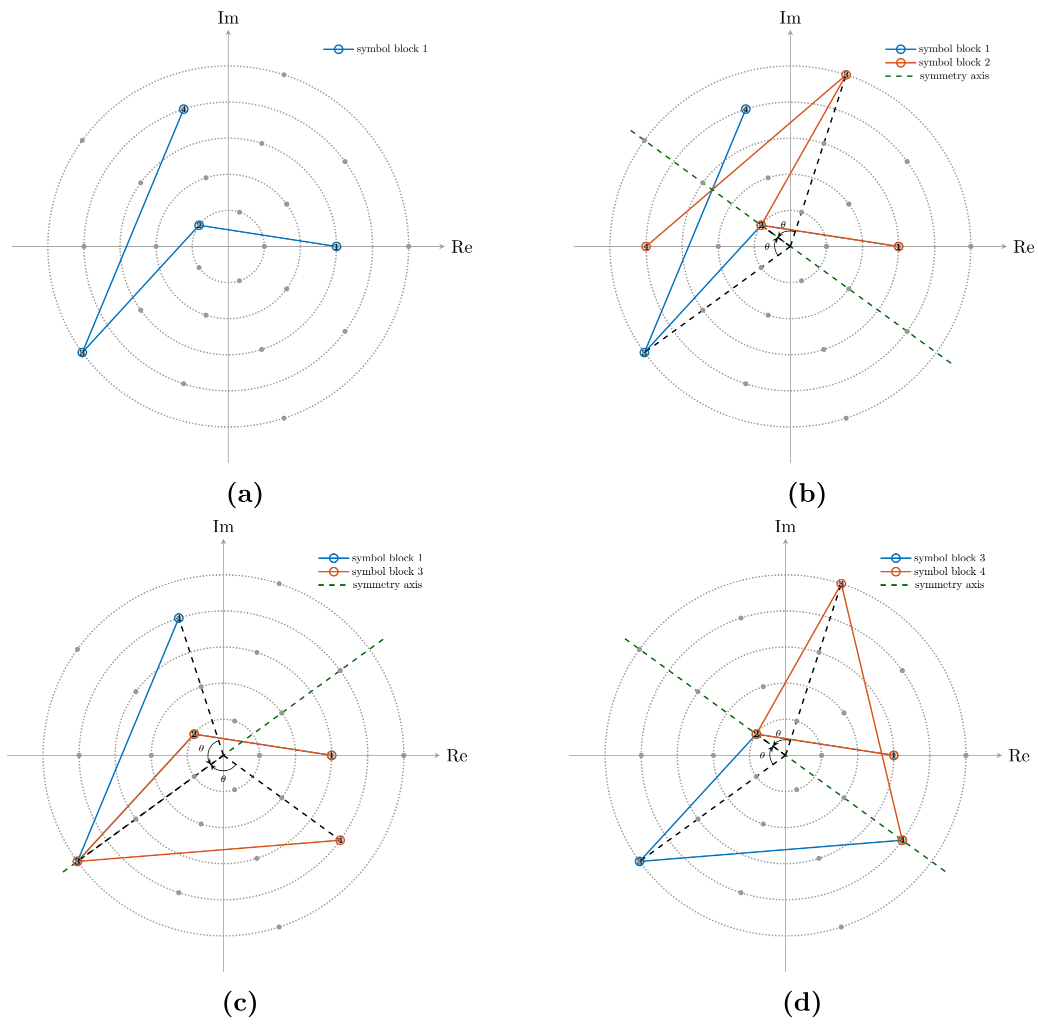
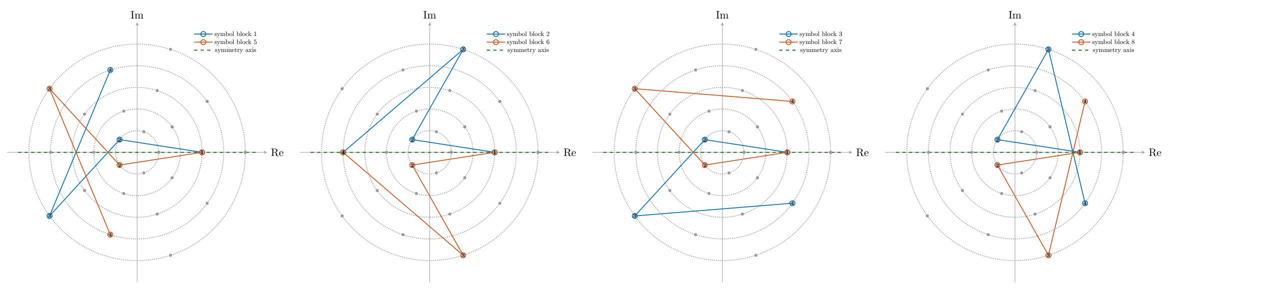


Figure 2.4: Construction of 4 different but square law equivalent symbol blocks.

to the equivalence class, as shown in figure 2.5.

Finally we can rotate all the 8 different, but square law equivalent symbol blocks, 4 times around the origin to create 32 new different symbol blocks that belong to the equivalence class, as shown in figure 2.6. As a result we get all the symbol block that belong to the same equivalence class, those symbol blocks are shown on figure 2.7.

When looking all the symbol blocks of the equivalence class (see figure 2.7), it is possible to notice the high symmetry of the plot, this shows that constellations with high symmetry allow a lot of ambiguities and reduce the achievable rate of the system.



(a) **(b)** **(c)** **(d)**

Figure 2.5: Construction of 4 new different but square law equivalent symbol blocks by reflection.

2.1.7 Detector

For the detection, the maximum-likelihood (ML) criterion is chosen, that means that from $\boldsymbol{y} = (y_0, \ldots, y_{n-1})$ and $\boldsymbol{z} = (z_0, \ldots, z_{n-2})$, the detector outputs $\hat{k}$ if and only if [1]:

$$\hat{k} = \underset{d \in \{1,\ldots,M\}}{\arg\max}\ f\big(\boldsymbol{y}, \boldsymbol{z} | \boldsymbol{x}_d\big) \tag{2.24}$$

Where the PDF of $\boldsymbol{y}, \boldsymbol{z}$ given $\boldsymbol{x}_d$ is given by [1]:

$$f\big(\boldsymbol{y}, \boldsymbol{z} | \boldsymbol{x}_d\big) = f\big(\boldsymbol{y} | \boldsymbol{x}_d\big) f\big(\boldsymbol{z} | \boldsymbol{x}_d\big) = \prod_{i=0}^{n-1} f\big(y_i | \boldsymbol{x}_d[i]\big) \prod_{i=0}^{n-2} f\big(z_i | \boldsymbol{x}_d[i], \boldsymbol{x}_d[i+1]\big) \tag{2.25}$$

From equations 2.14 and 2.17 it is clear that y_i and z_i given x_i and x_{i+1} are distributed as [1]:

$$y_i \sim \mathcal{N}\left(\alpha^2(1-\beta)T|x_i|^2, (1-\beta)T(\alpha^2|x_i|^2\sigma_{sh}^2 + \sigma_{th}^2)\right) \tag{2.26}$$
$$z_i \sim \mathcal{N}\left(\alpha^2\beta T\psi(x_i, x_{i+1}), \beta T(\alpha^2\psi(x_i, x_{i+1})\sigma_{sh}^2 + \sigma_{th}^2)\right) \tag{2.27}$$

2.2 Numerical simulation

2.2.1 System parameters

For the numerical simulation the system is simulated in base band, the baud rate is 10 Gb/s, no attenuation in the optical fiber is considered, and the parameter of the diode are taken from [1] and can be seen on table 2.2.

2.2.2 Class representative creation

To define the Class representative block one have to specify a constellation and a block length n, then look for all the equivalence classes exhaustively, to finally select one random symbol block from each equivalence class to conform the class representative. In this step several constellations are considered with different block length, six examples of constellations are shown in figure 2.8.

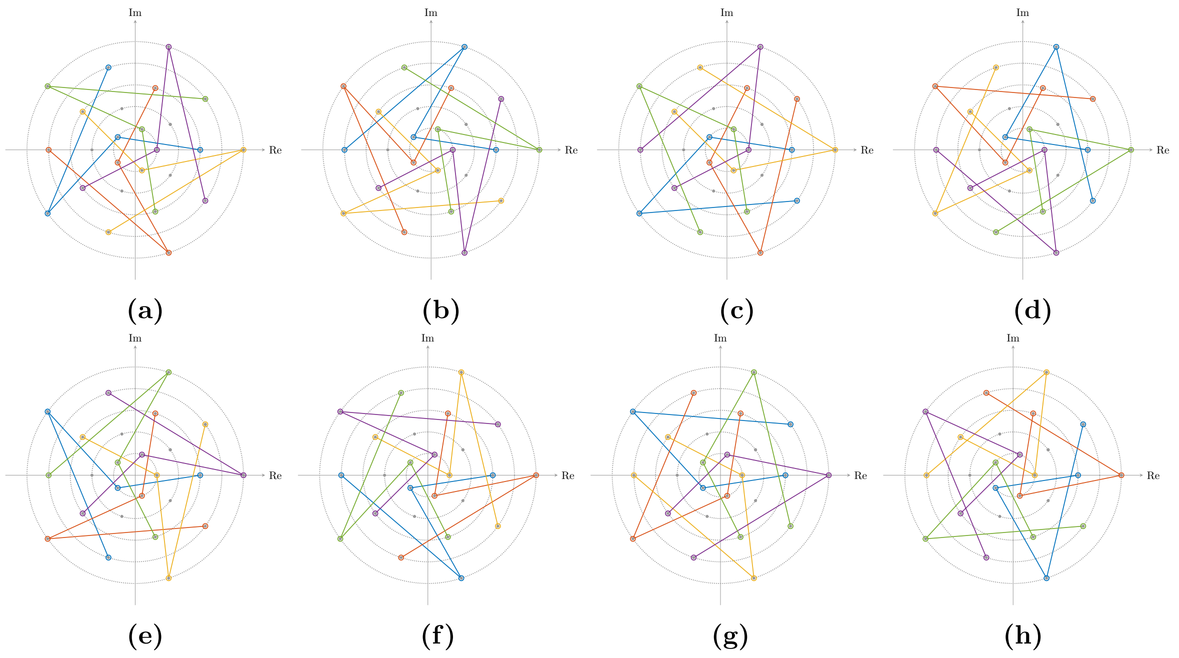


Figure 2.6: Construction of 4 new different but square law equivalent symbol blocks by rotation.

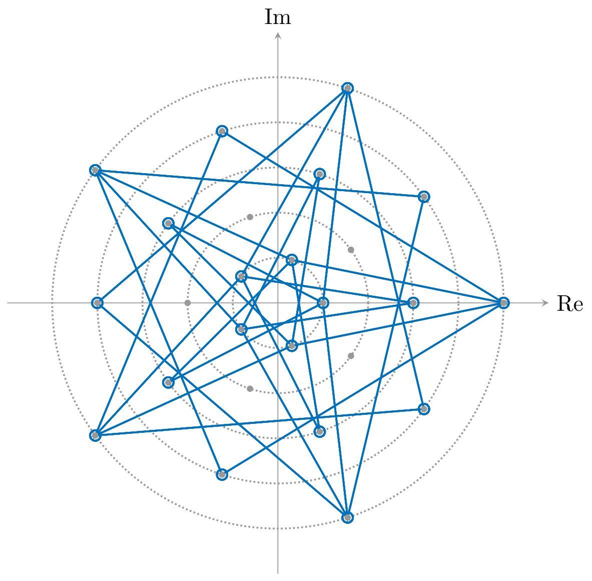


Figure 2.7: All the symbol blocks in an equivalence class shown at the same time

To calculate those equivalence classes we use a Python script, that is available on the GitHub repository Direct_detection_under_Tukey_Signaling, all the related codes are in the folder **Equivalence_classes**. From this code we get all the equivalence classes for the studied constellations and block length, the summary can be seen in the tables 2.3 to 2.6 that show the number of equivalence classes sorted by the class size L and the total of equivalence

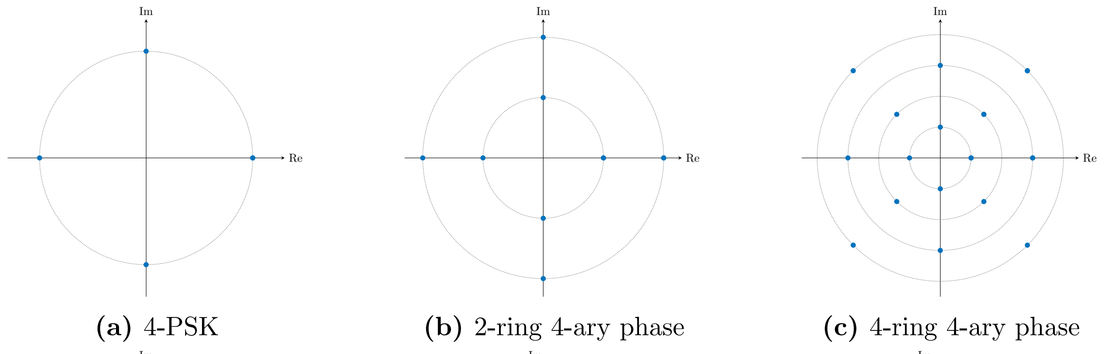
Parameter	Value	Typical range
Temperature (T)	300 K	
Load resistance (R_L)	15 Ω	
APD Gain (M_{APD})	20	10 - 40
Enhanced Responsivity ($M_{\text{APD}} R_{\text{APD}}$)	10 mA/mW	5 - 20
k-factor (k_{APD})	0.6	0.5 - 0.7
Excess noise factor (F)	12.78	a function of M_{APD} and k_{APD}

Table 2.2: Parameter of the photodiode used in the simulations. Taken from [1]

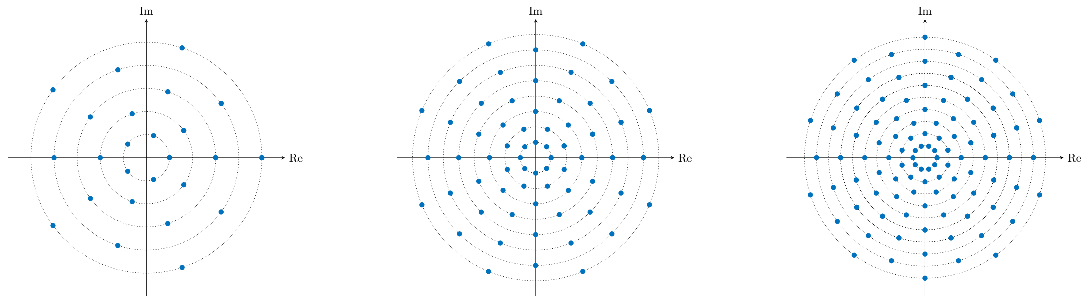
classes for different block lengths n, finally is also shown the rate loss, calculated as:

$$R_{\text{loss}} = \frac{1}{n} \log_2 \left(\frac{|\mathcal{K}|^n}{N_{\text{Eq class}}} \right) \tag{2.28}$$

where $|\mathcal{K}|$ denotes the number of points of the constellation, and $N_{\text{Eq class}}$ the total of equivalence classes.



(a) 4-PSK **(b)** 2-ring 4-ary phase **(c)** 4-ring 4-ary phase



(d) 5-ring 5-ary phase **(e)** 8-ring 8-ary phase **(f)** 10-ring 10-ary phase

Figure 2.8: Different possible constellations $\mathcal{K}$.

2.2.3 Mutual information estimation

One of the figure of merit of the system is the Mutual information I of the channel which is given by [8]:

$$I = \sum_{x,y} P(x,y) \log_2 \left(\frac{P(x,y)}{P(x)P(y)} \right) \tag{2.29}$$

Class size \ Block length	3	4	5	6	7	8
4	4	8	16	32	64	128
8	4	12	32	80	192	448
16	1	6	24	80	240	672
32		1	8	40	160	560
64			1	10	60	280
128				1	12	84
256					1	14
512						1
Total	9	27	81	243	729	2187
Rate loss (bit/sym)	0.94	0.81	0.73	0.68	0.64	0.61

Table 2.3: Number of equivalence classes for 4-PSK constellation.

Class size \ Block length	3	4	5	6	7
4	32	128	512	2048	8192
8	32	192	1024	5120	24576
16	8	96	768	5120	30720
32		16	256	2560	20480
64			32	640	7680
128				64	1536
256					128
Total	72	432	2592	15552	93312
Rate loss (bit/sym)	0.94	0.81	0.73	0.68	0.64

Table 2.4: Number of equivalence classes for 2-ring 4-ary Phase constellation.

To estimate the mutual information we use a Python script to implement a Montecarlo simulation with 100 000 symbol blocks (code available on the GitHub repository Direct_detection_under_Tukey_Signaling, in the folder **skripts** file **main.py**). In this case for the simulation all the possible square law distinct symbol blocks are used.

For the 2-ring 4-ary phase constellation and symbol block length $n = 3$ we simulated for 4 different β, and get the results shown in figure 2.9. Notice that for midium and high SNR the best choice of beta is $\beta = 0.9$, also is important to observe that the mutual information tends to 2.06 bit/sym approximately, which is coherent with the rate loss calculated (see table 2.4)

$$2.06\,\text{bit/sym} = 3\,\text{bit/sym} - 0.94\,\text{bit/sym}$$

Also is important to notice that the obtained results are coherent with the results reported by [1], which show that the implementation of the system made by us is well done.

Block length / Class size	3	4
5	125	625
10	500	3750
20	500	7500
40		5000
Total	1125	16875
Rate loss (bit/sym)	1.27	1.13

Table 2.5: Number of equivalence classes for 5-ring 5-ary Phase constellation.

Block length / Class size	3
8	512
16	3584
32	6272
Total	10368
Rate loss (bit/sym)	1.55

(a) 8-ring 8-ary Phase

Block length / Class size	3
10	100
20	9000
40	20250
Total	30250
Rate loss (bit/sym)	1.68

(b) 10-ring 10-ary Phase

Table 2.6: Number of equivalence classes for different constellations.

2.2.4 BER estimation

To estimate the BER, once again a Montecarlo simulation is used, but this time not all the square law distinct symbols are used, instead we use the biggest power of two [1]. For example for the 2-ring 4-ary phase constellation and symbol block length $n = 4$ we use 256 square law distinct symbol blocks instead of 432.

Again we simulated for different β, the 2-ring 4-ary phase constellation and symbol block length $n = 4$ to get the results shown in figure 2.10. The results show the typical behavior of a BER vs SNR graph, again the case of $\beta = 0.9$ have the best performance, and the results are coherent with the ones presented by [1].

2.3 Discussion

After implementing the system of Tukey signaling, we noticed that the system, at least in simulation, and an ideal situation should work good and is able to retrieve some information about the phase of the complex symbols using only DD. However there are 3 down sides of the system: the complexity of the decoder, the bandwidth efficiency and the implementation.

To decrease the rate loss in this system one can increase the block length, but the number of equivalence classes grows exponentially with the block length, and so does the complexity of the decoder, that is why the system becomes impractical really quickly, that is why the

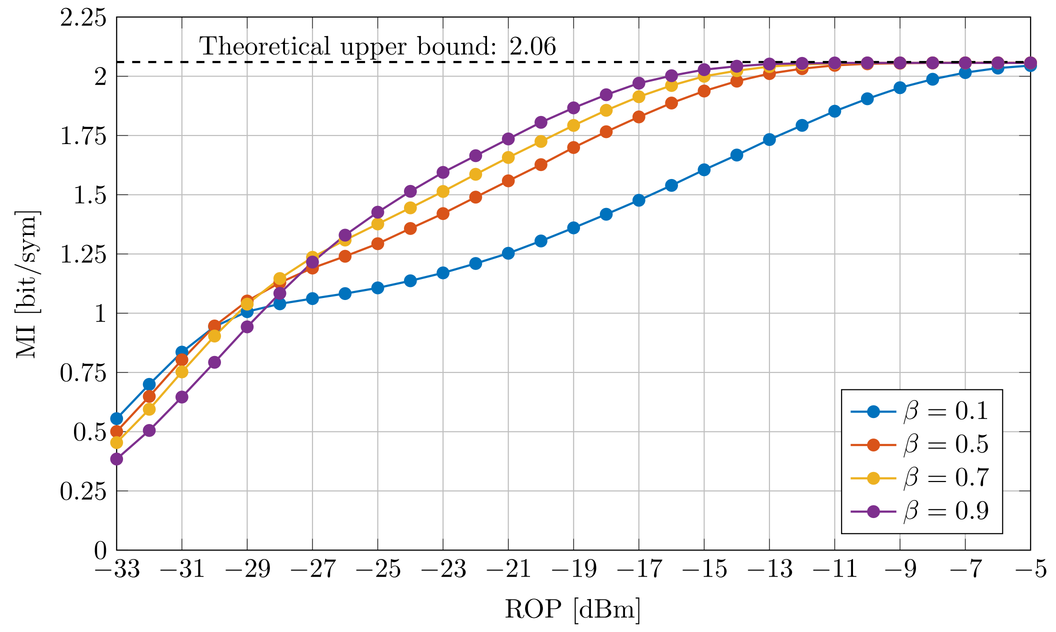


Figure 2.9: Mutual information of the system with 2-ring 4-ary phase constellation and symbol block length $n = 3$.

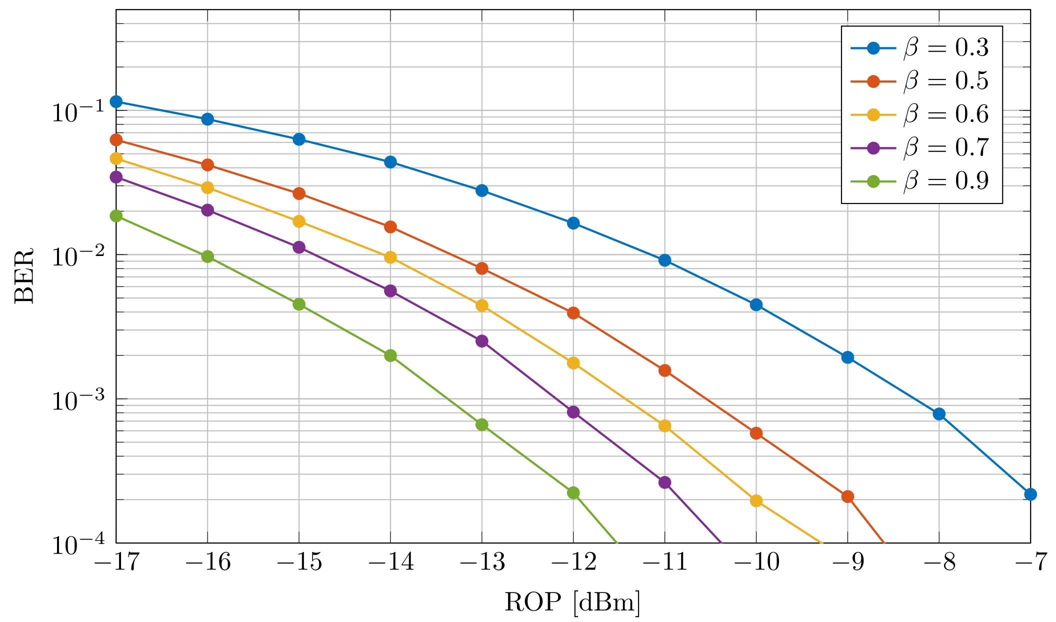


Figure 2.10: Mutual information of the system with 2-ring 4-ary phase constellation and symbol block length $n = 4$.

complexity of the decoder is a big drawback of the system.

An other drawback is the bandwidth efficiency, since the Tukey signal is time limited, its bandwidth is in theory unlimited, and in practice really big compared to others wave forms, such as a sinc pulse. To solve this problem one should increase the duration of the pulse, but doing so the idea of the integrate and dump block is no longer valid, because at some points in time more than 2 symbols may influence the signal, and the idea of the system was based on the assumption that any time at most 2 symbols influence the signal.

Finally the integrate and dump is an analog block by default, which may be complicated to implement, and a digital approximation of the block requires a big oversampling factor which is to expensive in terms of the hardware to do that.

For this three drawbacks we decide to look for other solutions in the literature and we stop the work with the system proposed by Tasbihi in [1].

Chapter 3

Generalized Direct Detection with phase recovery

The other solution considered to the problem of phase recovery is the proposed by Plabst in the paper [7]. There the authors try to improve the two drawbacks of the Tukey signaling scheme. First they considered an arbitrary pulse wave form, and in particular they experimented with raised cosine waveforms, with the objective to improve the spectral efficiency. Second they propose a discrete time model, which allows the use of a digital system with oversampling factor of only two.

The proposed system model can be seen in figure 3.1. In the following sections we will explain the system model, both in continuous time and in discrete time, as well as the detector used to determine an upper bound limit of the performance of the system.

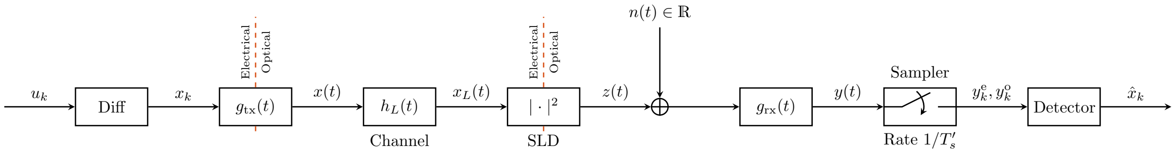


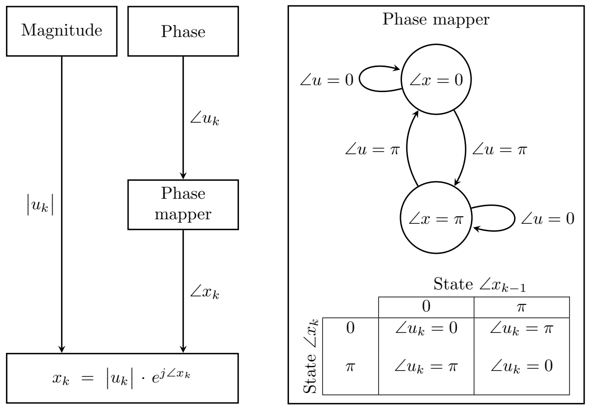
Figure 3.1: System model proposed in [7].

3.1 System model

3.1.1 Continuous time Model

3.1.1.1 Differential encoder

As it was mentioned in chapter 2 the DD generate some ambiguities, and one way to eliminate them is by using differential encoding [6], [7], [9]. As suggested in [9] the phase encoding is done by a function that receive as an input a vector of symbols $\boldsymbol{u}$ and outputs a vector of symbols $\boldsymbol{x} = f_{\text{dif}}(\boldsymbol{u})$ with the same magnitude as the symbols in $\boldsymbol{u}$, and a given phase based on a set of conditions, the phase of the input symbol and the phase of the last output symbol and, as shown for example in the figure 3.2 for a M-ASK constellation.



	0	π
0	$\angle u_k = 0$	$\angle u_k = \pi$
π	$\angle u_k = \pi$	$\angle u_k = 0$

Figure 3.2: Differential phase mapper example for M-ASK. Based on [9]

3.1.1.2 Transmitter

The transmitter receives as input a sequence $\boldsymbol{x} = \{x_0, \ldots, x_{n-1}\}$ of independent and identically distributed (i.i.d.) symbols x_i taken from a finite constellation $\mathcal{K} = \{a_1, \ldots, a_q\}$ with q elements, and outputs a signal given by [7]:

$$x(t) = \sum_{k=0}^{n-1} x_k \cdot g_{\text{tx}}(t - iT) \tag{3.1}$$

where T is the inverse of the symbol rate, $x_i \in \mathcal{K}$, and $g_{\text{tx}}(t)$ is the pulse waveform.

Common pulse waveforms are raised cosine pulses, of particular interest is the case for roll off factor $\alpha = 0$, that is:

$$g_{\text{tx}} = \frac{t}{T_s}\text{sinc}\left(\frac{1}{T_s}\right) \tag{3.2}$$

$$G_{\text{tx}} = \begin{cases} 1 & \text{if } |f| \leq \frac{1}{2T_s} \\ 0 & \text{otherwise} \end{cases} \tag{3.3}$$

3.1.1.3 Fiber Optic Link

The channel considered is a optic fiber that only presents chromatic dispersion, characterized in frequency domain by [7]:

$$H_L(f) = e^{j2\beta_2 L\pi^2 f^2} \tag{3.4}$$

where β_2 is the group-velocity dispersion parameter and L is the fiber length.

3.1.1.4 Receiver

The receiver consist of a photodiode, whose output is given by:

$$z(t) = |x_L(t)|^2 \tag{3.5}$$

The photodiode noise is modeled as a real valued white gaussian process with spectral density $N_0/2$ [7], representing the thermal noise. Finally the receiver has a band limited sampler, with impulse response

$$g_{\text{rx}} = 2B\text{sinc}\,(2Bt) \tag{3.6}$$

$$G_{\text{rx}} = \begin{cases} 1 & \text{if } |f| \leq B \\ 0 & \text{otherwise} \end{cases} \tag{3.7}$$

where B is the symbol rate given by $B = 1/T_s$ [7].

3.1.2 Discrete time Model

3.1.2.1 Discrete signal notation

For the discrete time model, a receiver with sampling rate $1/T_s' = 2B$ is considered (in this section the super index $'$ denotes a variable of the oversampled system), that is an oversampling factor of $N_{\text{os}} = T_s/T_s' = 2$, meaning 2 samples per transmitted symbol [7].

Let $\boldsymbol{x}' = \{0, x_0, \dots 0, x_{n-1}\}$ and $\boldsymbol{y}' = \{y_0, \dots, y_{2n-1}\}$ be the upsampled sequence at the transmitter and receiver, they are related by [7]:

$$y_k' = z_k' + n_k' \tag{3.8}$$

$$z_k' = \left(|x_L(t)|^2 * g_{\text{rx}}(t)\right)_{t=kT_s'} \tag{3.9}$$

where $n_k \sim \mathcal{N}(0, \sigma_N^2)$ and $\sigma_N^2 = N_0 B$.

Now define the combined impulse response of the pulse shaping filter, and the channel response as $\psi(t) = g_{\text{tx}}(t) * h_L(t)$ and the discrete version of it $\psi_k = \psi(kT_s')$, then [7]:

$$x_L(kT_s') = \sum_{m=0}^{2n-1} \psi_m x_{k-m}' \tag{3.10}$$

Finally notice that, for the special case when the pulse shape is the sinc pulse in equations 3.2 and 3.3, $|x_L(t)|^2 * g_{\text{rx}} = |x_L(t)|^2$, so:

$$z_k' = \left| \sum_{m=0}^{2n-1} \psi_m x_{k-m}' \right|^2 \tag{3.11}$$

3.1.2.2 Vector matrix notation

Now if we think in the signals as column vectors:

$$\begin{aligned} \boldsymbol{x}' &= \left[0, x_0, \ldots, 0, x_{n-1}\right]^T && \in \mathbb{C}^{2n\times 1} \\ \boldsymbol{z}' &= \left[z_0, \ldots, z_{2n-1}\right]^T && \in \mathbb{R}^{2n\times 1} \\ \boldsymbol{n}' &= \left[n_0, \ldots, n_{2n-1}\right]^T && \in \mathbb{R}^{2n\times 1} \\ \boldsymbol{y}' &= \left[y_0, \ldots, y_{2n-1}\right]^T && \in \mathbb{R}^{2n\times 1} \end{aligned}$$

Also consider that ψ is time limited, so ψ_m is zero outside some interval $[0, M-1]$, and define the Toeplitz matrix $\boldsymbol{\Psi} \in \mathbb{C}^{2n\times(2n+M-1)}$ as:

$$\boldsymbol{\Psi} = \begin{bmatrix} \psi_{M-1} & \psi_{M-2} & \cdots & \psi_0 & & & & \\ & \psi_{M-1} & \cdots & \psi_1 & \psi_0 & & & \\ & & \ddots & & & \ddots & \text{\Large 0} & \\ & & & \psi_{M-1} & \cdots & \cdots & \psi_0 & \\ & \text{\Large 0} & & & \ddots & & & \ddots \\ & & & & \psi_{M-1} & \cdots & \cdots & \psi_0 \end{bmatrix} \tag{3.12}$$

Finally define the channel state $\boldsymbol{s}_0$ as:

$$\boldsymbol{s}_0' = [0, x_{-\widetilde{M}}, 0, x_{1-\widetilde{M}}, \ldots, 0, x_{-1}]^T \in \mathbb{C}^{(M-1)\times 1} \tag{3.13}$$

where $\widetilde{M}$ is the memory channel in terms of the transmitted symbols and is given by $\widetilde{M} = (M-1)/2$.

With the previous definitions the output of the square law detection is given by:

$$\boldsymbol{z}' = \left|\boldsymbol{\Psi}\begin{bmatrix} \boldsymbol{s}_0' \\ \boldsymbol{x}' \end{bmatrix}\right|^{\circ 2} = \left|\boldsymbol{\Psi}\tilde{\boldsymbol{x}'}\right|^{\circ 2} \qquad \in \mathbb{R}^{2n}$$

where $|\cdot|^{\circ 2}$ is the element wise $|\cdot|^2$ operator.

Finally the complete discrete time system model including the Gaussian noise is given by [7]:

$$\boldsymbol{y}' = \boldsymbol{z}' + \boldsymbol{n}' = \left|\boldsymbol{\Psi}\tilde{\boldsymbol{x}'}\right|^{\circ 2} + \boldsymbol{n}' \qquad \in \mathbb{R}^{2n}$$

and the channel's conditional probability density is Gaussian[7]:

$$p(\boldsymbol{y}'|\boldsymbol{x}') = \mathcal{N}\left(\boldsymbol{y} - \left|\boldsymbol{\Psi}\tilde{\boldsymbol{x}'}\right|^{\circ 2}; \boldsymbol{0}_{2n}, \sigma_N^2 \boldsymbol{I}_{2n}\right) \tag{3.14}$$

3.1.2.3 Even and odd samples

For convenience one can concatenate the k-th and $(k+1)$-th sample in a new vector as follows:

$$\boldsymbol{z}_k = \left[z_{2k}', z_{2k+1}'\right] = \left[z_k^{\mathrm{e}}, z_k^{\mathrm{o}}\right] \tag{3.15}$$

$$\boldsymbol{y}_k = \left[y_k^{\mathrm{e}}, y_k^{\mathrm{o}}\right] = \boldsymbol{z}_k + \left[n_k^{\mathrm{e}}, n_k^{\mathrm{o}}\right] \tag{3.16}$$

The reason for group two samples can be understood by looking at the figure 3.3. There the convolution and the SQL detector is represented for an even and odd sample, in the case of link length $L = 0$. For this choice of wave form, the even sample is an ISI-free sample, and carries information about the magnitude of a past symbol; in contrast the odd sample experience ISI, and hence somehow carries information about the phases of the symbols.

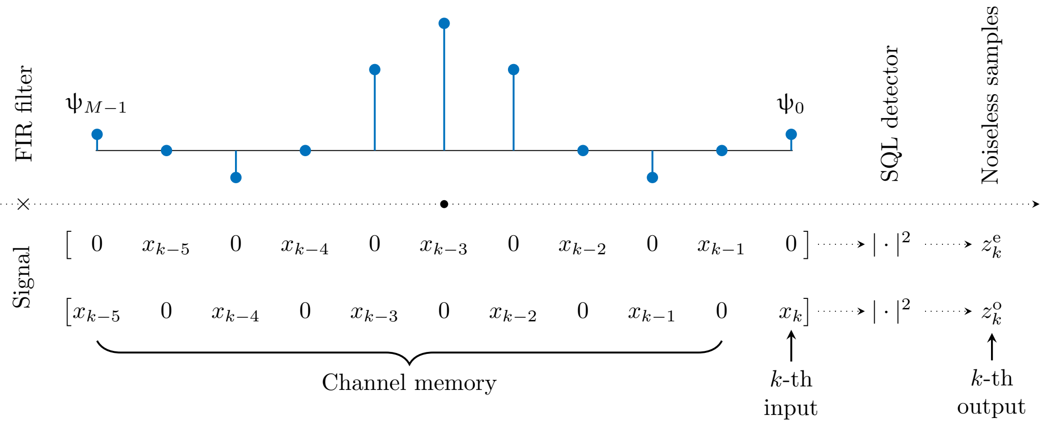

FIR filter
ψ_{M-1}
ψ_0
SQL detector
Noiseless samples
×
Signal
$[\ 0\quad x_{k-5}\quad 0\quad x_{k-4}\quad 0\quad x_{k-3}\quad 0\quad x_{k-2}\quad 0\quad x_{k-1}\quad 0\ ]$
$|\cdot|^2$
z_k^{e}
$[x_{k-5}\quad 0\quad x_{k-4}\quad 0\quad x_{k-3}\quad 0\quad x_{k-2}\quad 0\quad x_{k-1}\quad 0\quad x_k]$
$|\cdot|^2$
z_k^{o}
Channel memory
k-th input
k-th output

Figure 3.3: Signals for oversampling factor $N_{\mathrm{os}} = 2$. Based on [7]

3.1.3 Symbol-wise MAP detection

To evaluate the optimal performance of the system, in [7] is proposed a decoder based on the maximum a posteriori that decides the transmitted symbol based on the a posteriori probabilities (APP) $p(x_k|\boldsymbol{y}')$. Since the APPs are proportional to $p(x_k, \boldsymbol{y}')$ because $p(\boldsymbol{y}')$ is constant with respect to the decision, the MAP rule becomes [7]:

$$\hat{x}_k = \arg\max_{x_k} p(x_k, \boldsymbol{y}') \tag{3.17}$$

as a reminder, $\boldsymbol{y}'$ is the hole vector of received samples, i.e. $\boldsymbol{y}' = \left[y_0, \ldots, y_{2n-1}\right]^T$, and $p(x_k, \boldsymbol{y}')$ is given by [7]:

$$p(x_k, \boldsymbol{y}') = \sum_{s_0} \sum_{\boldsymbol{x}\backslash x_k} p(s_0, \boldsymbol{x}, \boldsymbol{y}') \tag{3.18}$$

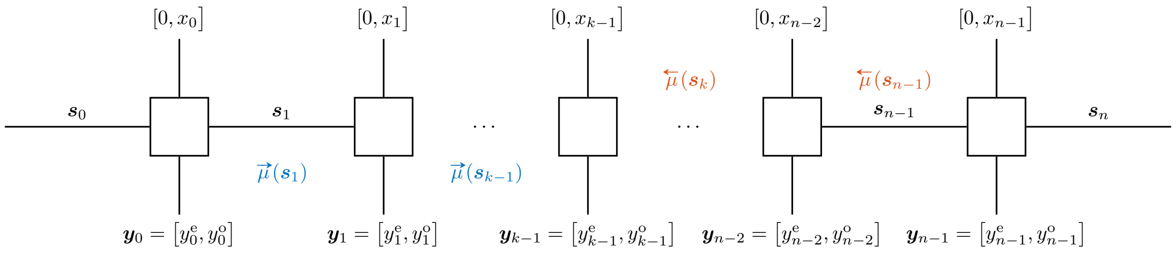
and $\sum_{\boldsymbol{x}\backslash x_k}$ denotes a sum over all possible vectors $\boldsymbol{x}$ but where the k-th position is fixed.

This APPs can be efficiently computed with the Forward-Backward Algorithm or BCJR algorithm with the factor graph shown in figure 3.4 and the following recursive equations [7]:

$$\overrightarrow{\mu}(\boldsymbol{s}_k) = \sum_{x_k, \boldsymbol{s}_{k-1}} p(x_k, \boldsymbol{y}_k, \boldsymbol{s}_k|\boldsymbol{s}_{k-1}) \cdot \overrightarrow{\mu}(\boldsymbol{s}_{k-1}) \qquad \text{FW path} \tag{3.19}$$

$$\overleftarrow{\mu}(\boldsymbol{s}_{k-1}) = \sum_{x_k, \boldsymbol{s}_k} p(x_k, \boldsymbol{y}_k, \boldsymbol{s}_k|\boldsymbol{s}_{k-1}) \cdot \overleftarrow{\mu}(\boldsymbol{s}_k) \qquad \text{BW path} \tag{3.20}$$

$$p(x_k, \boldsymbol{y}') = \sum_{\boldsymbol{s}_{k-1}} \sum_{\boldsymbol{s}_k} \overrightarrow{\mu}(\boldsymbol{s}_{k-1}) \cdot p(x_k, \boldsymbol{y}_k, \boldsymbol{s}_k|\boldsymbol{s}_{k-1}) \cdot \overleftarrow{\mu}(\boldsymbol{s}_k) \qquad \text{APP} \tag{3.21}$$



$[0, x_0]$
$[0, x_1]$
$[0, x_{k-1}]$
$[0, x_{n-2}]$
$[0, x_{n-1}]$
$\overleftarrow{\mu}(\boldsymbol{s}_k)$
$\overleftarrow{\mu}(\boldsymbol{s}_{n-1})$
$\boldsymbol{s}_0$
$\boldsymbol{s}_1$
$\boldsymbol{s}_{n-1}$
$\boldsymbol{s}_n$
$\vec{\mu}(\boldsymbol{s}_1)$
$\vec{\mu}(\boldsymbol{s}_{k-1})$
$\boldsymbol{y}_0 = [y_0^{\mathrm{e}}, y_0^{\mathrm{o}}]$
$\boldsymbol{y}_1 = [y_1^{\mathrm{e}}, y_1^{\mathrm{o}}]$
$\boldsymbol{y}_{k-1} = [y_{k-1}^{\mathrm{e}}, y_{k-1}^{\mathrm{o}}]$
$\boldsymbol{y}_{n-2} = [y_{n-2}^{\mathrm{e}}, y_{n-2}^{\mathrm{o}}]$
$\boldsymbol{y}_{n-1} = [y_{n-1}^{\mathrm{e}}, y_{n-1}^{\mathrm{o}}]$

Figure 3.4: Factor graph to implement the BCJR. Based on [7]

where $k = 1, \dots, n$, $\vec{\mu}(\boldsymbol{s}_0) = p(\boldsymbol{s}_0)$ and $\overleftarrow{\mu}(\boldsymbol{s}_n) = p(\boldsymbol{s}_n)$.

Finally the likelihood is given by [7]:

$$p(x_k, \boldsymbol{y}_k, s_k | \boldsymbol{s}_{k-1}) = \underbrace{p(\boldsymbol{y}_k | x_k, \boldsymbol{s}_{k-1})}_{(a)} \underbrace{p(\boldsymbol{s}_k | x_k, \boldsymbol{s}_{k-1})}_{(b)} \underbrace{p(x_k | \boldsymbol{s}_{k-1})}_{(c)} \tag{3.22}$$

where (a) is:

$$p(\boldsymbol{y}_k | x_k, \boldsymbol{s}_{k-1}) = p(y_k^{\mathrm{e}} | \boldsymbol{s}_{k-1}) \cdot p(y_k^{\mathrm{o}} | x_k, \boldsymbol{s}_{k-1}) \tag{3.23}$$

$$p(y_k^{\mathrm{e}} | \boldsymbol{s}_{k-1}) = p_N \left(y_k^{\mathrm{e}} - \left| [0, x_{k-\widetilde{M}}, \cdots, 0, x_{k-1}, 0] \cdot \psi \right|^2 \right) \tag{3.24}$$

$$p(y_k^{\mathrm{o}} | x_k, \boldsymbol{s}_{k-1}) = p_N \left(y_k^{\mathrm{o}} - \left| [x_{k-\widetilde{M}}, 0, \cdots, x_{k-1}, 0, x_k] \cdot \psi \right|^2 \right) \tag{3.25}$$

The term (b) is 1 if the state transition from $\boldsymbol{s}_{k-1}$ to $\boldsymbol{s}_k$ is possible and 0 otherwise, and (c) is $p(x_k | \boldsymbol{s}_{k-1}) = p(x_k)$ since the symbols are i.i.d. [7].

Additionally, since using differential encoding is recommended to avoid ambiguities, one can include the differential decoding in the BCJR algorithm by including the fact that $\boldsymbol{x} = f_{\text{diff}}(\boldsymbol{u})$ in the factor graph [7]. That is done by doing the summations in equations 3.19, 3.20 and 3.21 over all u_k instead of x_k (which in terms of the algorithm makes no change), and defining the states of equations 3.24 and 3.25 in terms of the $\boldsymbol{u}$, that means replacing $\boldsymbol{x}$ by $f_{\text{diff}}(\boldsymbol{u})$ (for more details see the algorithm used in [10]).

3.2 Numerical simulation

For the numerical simulations we implemented the proposed system model in a Python simulation, which can be found in the git repository Direct_Detection_with_Phase_Recovery. As the system parameters we use the settings found in table 3.1, and we estimate the SER using a Montecarlo simulation with 20 000 transmitted symbols. We set the noise power to $\sigma_N^2 = 1$ so the SNR is given by [7]:

$$\text{SNR} = P_{\text{tx}} = \frac{\mathbb{E}\big[||x(t)||\big]}{nT_s} \tag{3.26}$$

Paramater	Value
β_2	$-2.168 \times 10^{-23}\,\mathrm{s}^2/\mathrm{km}$
attenuation factor	0.2 dB/km
link lengths L	0 km
Baud rate B	35 Gbaud

Table 3.1: Simulation parameters. Taken from [7]

For the pulse shaping filter we used a sinc pulse with sampling rate of $2B$. Since the complexity of the decoder grows exponentially with the length of the filter we could not use in the decoder the complete signal $\psi(t)$, instead of that we use an auxiliary channel for the decoding process, that considers only a window of the real channel as shown in figure 3.5. That means, we simulated the transmission with a channel with memory (in terms of the transmitted symbols) of $\widetilde{M}$, i.e. $M = 2\widetilde{M} + 1$ taps, but decode with an auxiliary channel with memory $\widetilde{N}$ i.e. $N = 2\widetilde{N} + 1$ taps.

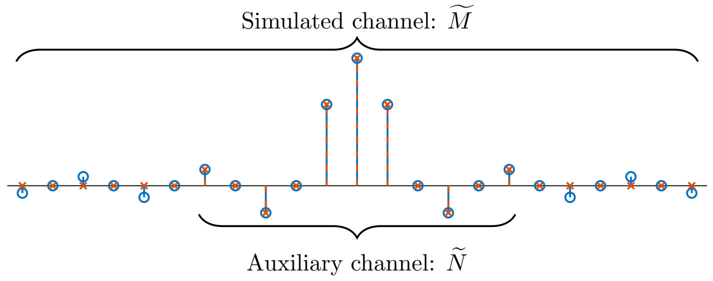

Simulated channel: $\widetilde{M}$
Auxiliary channel: $\widetilde{N}$

Figure 3.5: Simulated and auxiliary channel. Based on [7]

For the constellations we choose 3 different constellations, BPSK, 4-QAM and DD-SQAM, a specially designed constellation, shown in figure 3.6, with the objective to mitigate the ambiguities that arise from the DD. As pointed out in section 2.1.6.1, constellations with high symmetry produce a lot of ambiguities, and the DD process only distinguish on the phase difference between symbols trough the cosine of the phase difference, so when designing the DD-SQAM constellation we reduced the symmetry, and separate the cosine of the angle between symbols as much as possible, that is why we use the special angle $\phi = \arccos(1/3)$.

Finally we simulate for each constellation two scenarios, one when the auxiliary channel is much shorter than the actual channel, and the other when the auxiliary channel and the actual channel have the same length. The length of the auxiliary channel is chosen based on complexity criteria, we choose a relative large length, that still is computable in reasonable time. The results can be seen in figure 3.7.

Looking at the figure 3.7 we can notice that the simulations with a channel miss-match ($\widetilde{N} < \widetilde{M}$) have an error floor, this is due to the ISI that is not considered by the BCJR decoder, so even increasing the SNR the SER do not improve, because the main source of "noise" is the ISI.

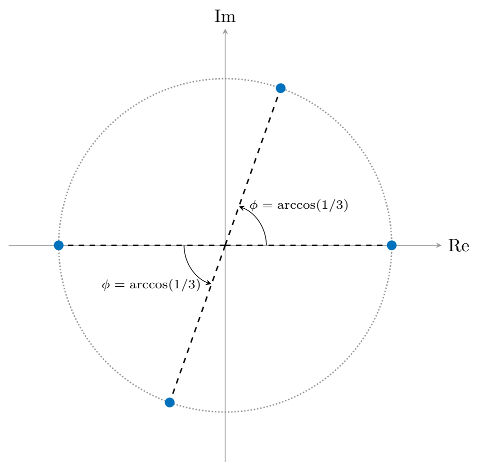


Figure 3.6: DD-SQAM constellation.

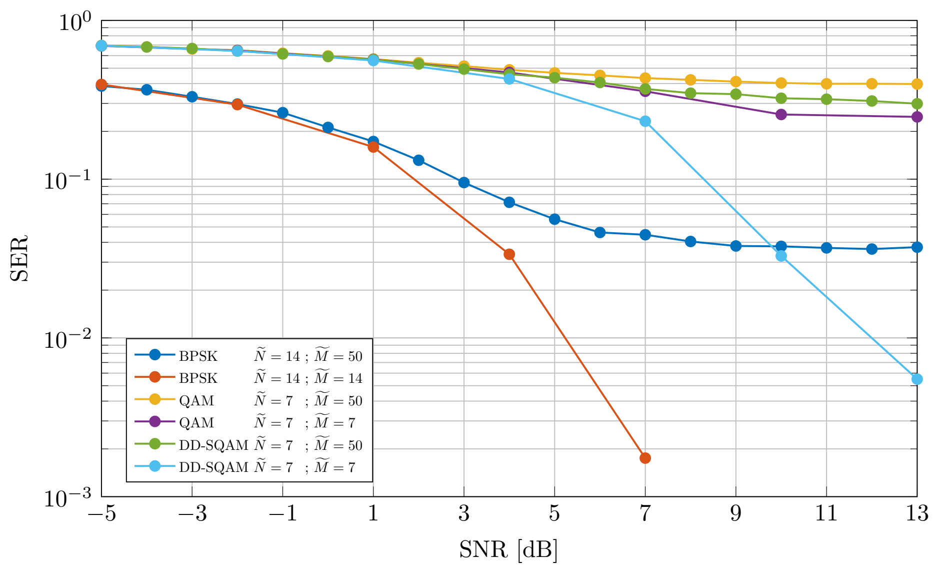


Figure 3.7: Symbol error rate vs SNR for different constellations, and different channel lengths.

It is also important to notice that the QAM constellation with $\widetilde{N} = \widetilde{M}$ also have an error floor at 2.5 approximately, this is caused by the ambiguities generated by the DD for the case of the QAM constellation. Even with differential encoding there is still ambiguities 1/4 of the time, because a phase change of π or 0 can be distinguished, but a phase change of $\pm\pi/2$ can not. Finally it is important to notice that the BPSK and DD-SQAM constellations do not have an error floor when $\widetilde{N} = \widetilde{M}$, this shows that this constellations when used with differential encoding do not generate ambiguities.

3.3 Discussion

The system proposed in [7] compared with the system proposed in [1] has the advantages of using a wave form with smaller bandwidth, that increase the spectral efficiency [7], and also represent a simpler system compatible with a digital system with oversampling factor of only 2, which is very desirable. However the problem of finding a good decoder with feasible complexity is still open.

For this reason we use this system as a base to formulate a machine learning based decoder which will be discussed in the next chapter.

This page intentionally left blank.

Chapter 4

MagPhase-DetNet

The decoder we try to implement is based on the DetNet decoder proposed in [11], [12], there the authors propose a machine learning based architecture for a MIMO AWGN linear channel. In those papers the authors noticed that the detection problem is equivalent to the following minimization problem in the MIMO case [11]:

$$\hat{\boldsymbol{x}} = \arg \min_{\boldsymbol{x} \in \mathcal{K}^n} \|\boldsymbol{y} - \boldsymbol{H}\boldsymbol{x}\|^2 \tag{4.1}$$

and hence it may be solved by gradient descent. However the solution found by gradient descent could be invalid, because the vector $\hat{\boldsymbol{x}}$ may not belong to $\mathcal{K}^n$. Thats way the authors propose a projected gradient descent, where in every iteration the new estimated vector is somehow projected in $\mathcal{K}^n$. With this in mind, the authors in [11], [12] proposed to unfold the iterations of the gradient descent, and perform each iteration with a layer of a neural network, and at the end have a estimation of the vector $\hat{\boldsymbol{x}}$.

As reported in [11], [12], the performance of the architecture is as gut as the state of the art decoders, but at least 30 times faster, what makes architecture promising.

The reason for trying this architecture, is the similitude if the MIMO problem (equation 4.1) ant the DD problem:

$$\hat{\tilde{\boldsymbol{x}}}' = \arg \min_{\tilde{\boldsymbol{x}}' \in \mathbb{K}^{2n}} \left\| \boldsymbol{y} - \left| \boldsymbol{\Psi} \tilde{\boldsymbol{x}}' \right|^{\circ 2} \right\|^2 \tag{4.2}$$

where the main difference is the non linear $|\cdot|^{\circ 2}$ operator. Which is not an impediment to implementing the DetNet architecture.

In the following sections we will follow the steps done in [12] to propose an architecture for the DD problem based on the system model presented in chapter 3, and present some numerical simulation results.

4.1 Separation in even and odd subchannel

The first thing we do to cast the problem for the DetNet architecture is to separate the hole system into two subchannels. For doing so we take advantage of the oversampling factor of 2, and distinction of even and odd samples done in equations 3.15 and 3.16.

For doing that start from the not upsampled $\boldsymbol{x}$ and $\boldsymbol{s}_0$ vector:

$$\boldsymbol{x} = \begin{bmatrix} x_0 & x_1 & \cdots & x_{n-1} \end{bmatrix}^T \qquad \in \mathbb{C}^n \tag{4.3}$$

$$\boldsymbol{s}_0 = \begin{bmatrix} x_{-\widetilde{M}} & x_{1-\widetilde{M}} & \cdots & x_{-1} \end{bmatrix}^T \qquad \in \mathbb{C}^{\widetilde{M}} \tag{4.4}$$

$$\tilde{x} = \begin{bmatrix} \boldsymbol{s}_0 & \boldsymbol{x} \end{bmatrix}^T \qquad \in \mathbb{C}^{n+\widetilde{M}} \tag{4.5}$$

and define the following subchannel matrices:

$$\boldsymbol{\Psi}_{\text{o}} = \begin{bmatrix} \psi_{M-1} & \psi_{M-3} & \cdots & \psi_0 & & & & \\ & \psi_{M-1} & \cdots & \psi_2 & \psi_0 & & & \\ & & \ddots & & & \ddots & \Large 0 & \\ & & & \psi_{M-1} & \cdots & \cdots & \psi_0 & \\ & \Large 0 & & & \ddots & & & \ddots \\ & & & & & \psi_{M-1} & \cdots & \cdots & \psi_0 \end{bmatrix} \in \mathbb{C}^{n\times(n+\widetilde{M})} \tag{4.6}$$

$$\boldsymbol{\Psi}_{\text{e}} = \begin{bmatrix} \psi_{M-2} & \psi_{M-4} & \cdots & \psi_1 & 0 & & & & \\ & \psi_{M-2} & \cdots & \psi_3 & \psi_1 & 0 & & & \\ & & \ddots & & & \ddots & \ddots & \Large 0 & \\ & & & \psi_{M-2} & \cdots & \cdots & \psi_1 & 0 & \\ & \Large 0 & & & \ddots & & & \ddots & \ddots \\ & & & & & \psi_{M-2} & \cdots & \cdots & \psi_1 & 0 \end{bmatrix} \in \mathbb{C}^{n\times(n+\widetilde{M})} \tag{4.7}$$

with this definitions the problem transforms into:

$$\begin{bmatrix} \boldsymbol{y}_{\text{e}} & \boldsymbol{y}_{\text{o}} \end{bmatrix} = \begin{bmatrix} \boldsymbol{z}_{\text{e}} & \boldsymbol{z}_{\text{o}} \end{bmatrix} + \begin{bmatrix} \boldsymbol{n}_{\text{e}} & \boldsymbol{n}_{\text{o}} \end{bmatrix} \tag{4.8}$$

$$= \begin{bmatrix} |\boldsymbol{\Psi}_{\text{e}}\tilde{\boldsymbol{x}}|^{\circ 2} & |\boldsymbol{\Psi}_{\text{o}}\tilde{\boldsymbol{x}}|^{\circ 2} \end{bmatrix} + \begin{bmatrix} \boldsymbol{n}_{\text{e}} & \boldsymbol{n}_{\text{o}} \end{bmatrix} \tag{4.9}$$

$$\tag{4.10}$$

which can be separated into the tow independent and fictitious channels:

$$\boldsymbol{y}_{\text{e}} = \boldsymbol{z}_{\text{e}} + \boldsymbol{n}_{\text{e}} = |\boldsymbol{\Psi}_{\text{e}}\tilde{\boldsymbol{x}}|^{\circ 2} + \boldsymbol{n}_{\text{e}} \qquad \in \mathbb{R}^n \tag{4.11}$$

$$\boldsymbol{y}_{\text{o}} = \boldsymbol{z}_{\text{o}} + \boldsymbol{n}_{\text{o}} = |\boldsymbol{\Psi}_{\text{o}}\tilde{\boldsymbol{x}}|^{\circ 2} + \boldsymbol{n}_{\text{o}} \qquad \in \mathbb{R}^n \tag{4.12}$$

4.2 Real and imaginary reparametrization

In order to be able to work with the Pytorch framework, we cast the original problem, that have complex numbers, in one problem that is completely real, for doing that we redefine

$\boldsymbol{\Psi}_{\mathrm{e,ML}}$, $\boldsymbol{\Psi}_{\mathrm{o,ML}}$ and $\tilde{\boldsymbol{x}}_{\mathrm{ML}}$ as suggested in [12] (where the subindex "ML" stands for machine learning):

$$\boldsymbol{\Psi}_{\mathrm{e,ML}} = \begin{bmatrix} \mathrm{Re}\{\boldsymbol{\Psi}_{\mathrm{e}}\} & -\mathrm{Im}\{\boldsymbol{\Psi}_{\mathrm{e}}\} \\ \mathrm{Im}\{\boldsymbol{\Psi}_{\mathrm{e}}\} & \mathrm{Re}\{\boldsymbol{\Psi}_{\mathrm{e}}\} \end{bmatrix} \qquad \in \mathbb{R}^{2n\times 2(n+\widetilde{M})} \tag{4.13}$$

$$\boldsymbol{\Psi}_{\mathrm{o,ML}} = \begin{bmatrix} \mathrm{Re}\{\boldsymbol{\Psi}_{\mathrm{o}}\} & -\mathrm{Im}\{\boldsymbol{\Psi}_{\mathrm{o}}\} \\ \mathrm{Im}\{\boldsymbol{\Psi}_{\mathrm{o}}\} & \mathrm{Re}\{\boldsymbol{\Psi}_{\mathrm{o}}\} \end{bmatrix} \qquad \in \mathbb{R}^{2n\times 2(n+\widetilde{M})} \tag{4.14}$$

$$\tilde{\boldsymbol{x}}_{\mathrm{ML}} = \begin{bmatrix} \mathrm{Re}\{\tilde{\boldsymbol{x}}\} \\ \mathrm{Im}\{\tilde{\boldsymbol{x}}\} \end{bmatrix} \qquad \in \mathbb{R}^{2(n+\widetilde{M})} \tag{4.15}$$

$$\tag{4.16}$$

We also redefine the $|\cdot|^{\circ 2}_{\mathrm{ML}}$ operator as:

$$|\cdot|^{\circ 2}_{\mathrm{ML}} : \mathbb{R}^{2n} \to \mathbb{R}^{n} \tag{4.17}$$

$$\left|\begin{bmatrix} x_1 \\ \vdots \\ x_{2n} \end{bmatrix}\right|^{\circ 2}_{\mathrm{ML}} = \begin{bmatrix} x_1^2 + x_{1+n}^2 \\ \vdots \\ x_n^2 + x_{2n}^2 \end{bmatrix} \tag{4.18}$$

In this way the system model is completely reparametrized into an analogous and completely real valued problem:

$$\boldsymbol{y}_{\mathrm{e}} = |\boldsymbol{\Psi}_{\mathrm{e,ML}}\tilde{\boldsymbol{x}}_{\mathrm{ML}}|^{\circ 2}_{\mathrm{ML}} + \boldsymbol{n}_{\mathrm{e}} \qquad \in \mathbb{R}^{n} \tag{4.19}$$

$$\boldsymbol{y}_{\mathrm{o}} = |\boldsymbol{\Psi}_{\mathrm{o,ML}}\tilde{\boldsymbol{x}}_{\mathrm{ML}}|^{\circ 2}_{\mathrm{ML}} + \boldsymbol{n}_{\mathrm{o}} \qquad \in \mathbb{R}^{n} \tag{4.20}$$

4.3 MagPhase-DetNet architecture

Now we are going to construct the architecture of the decoder based on the machine Learning solution presented in [12], hence from now on all, all the matrices and the $|\cdot|^{\circ 2}$ operator are to be understand as the reparametrized version of the problem, that means as if they have the subscript "ML" even if they do not.

The detection problem, according to de maximum likelihood criterion, is given by:

$$\hat{\tilde{\boldsymbol{x}}}' = \arg\min_{\tilde{\boldsymbol{x}}' \in \mathbb{K}^{2n}} \left\| \boldsymbol{y} - \left|\boldsymbol{\Psi}\tilde{\boldsymbol{x}}'\right|^{\circ 2} \right\|^2 \tag{4.21}$$

which, using the separation of channels, is equivalent to:

$$\hat{\tilde{\boldsymbol{x}}} = \arg\min_{\tilde{\boldsymbol{x}} \in \mathbb{K}^{n}} \left\{ \left\| \boldsymbol{y}_{\mathrm{e}} - |\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}|^{\circ 2} \right\|^2 + \left\| \boldsymbol{y}_{\mathrm{o}} - |\boldsymbol{\Psi}_{\mathrm{o}}\tilde{\boldsymbol{x}}|^{\circ 2} \right\|^2 \right\} \tag{4.22}$$

Given this objective function, in [12] is recommended to use the gradient of the expression above in the architecture, so each layer of the DetNet should mimic one projected gradient descent step, that is:

$$\tilde{\boldsymbol{x}}_{k+1} = \Pi\left(\tilde{\boldsymbol{x}}_k + \delta_k \nabla_{\tilde{\boldsymbol{x}}} \boldsymbol{f}(\tilde{\boldsymbol{x}})\big|_{\tilde{\boldsymbol{x}}=\tilde{\boldsymbol{x}}_{\boldsymbol{k}}} \right)$$

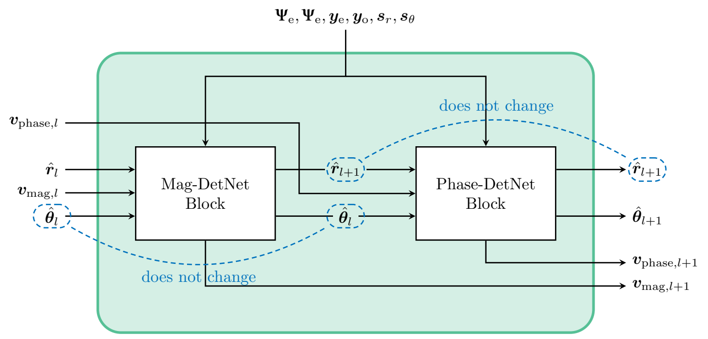
$$\text{whith} \quad \boldsymbol{f}(\tilde{\boldsymbol{x}}) = \left\| \boldsymbol{y}_{\mathrm{e}} - |\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}|^{\circ 2} \right\|^2 + \left\| \boldsymbol{y}_{\mathrm{o}} - |\boldsymbol{\Psi}_{\mathrm{o}}\tilde{\boldsymbol{x}}|^{\circ 2} \right\|^2$$

where $\Pi(\cdot)$ is a nonlinear function that forces the gradient descent to be a possible solution, that means $\hat{\boldsymbol{x}}_{k+1} \in \mathbb{K}^n$.

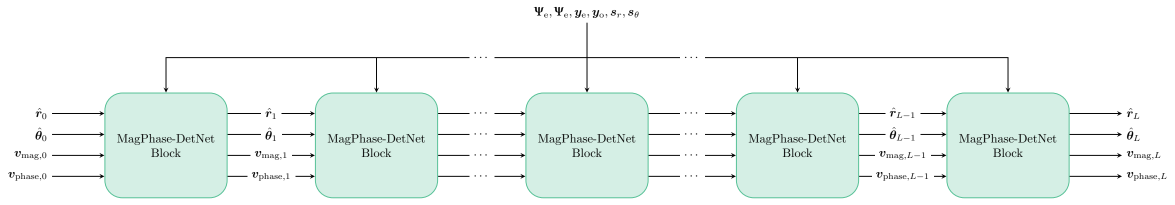
However we try this approach with several small variations in the loss function and the projection function, and we get no good results; this shows that the approach is not the best, so we had to find a better one.

We noticed that for the DD system it is really easy to detect the magnitude of the symbols (that is what a normal IM-DD system does) and the phase detection is the big problem. Also along all the studied papers is present the idea that one sample carries the information about the magnitude and the other carries, somehow, the information about the phase, even in [6] the magnitude detection, and phase detection are treated almost as separated tasks.

With this in mind we decided to separate a little bit the two tasks, so we propose the architecture shown on figure 4.1. The idea is that on each layer we use one DetNet-based block to improve the magnitude estimate (without changing the phase) followed by a DetNet-based block to improve the phase estimate (without changing the magnitude) and concatenate many layers to get the final architecture.


$\boldsymbol{\Psi}_e, \boldsymbol{\Psi}_e, \boldsymbol{y}_e, \boldsymbol{y}_o, \boldsymbol{s}_r, \boldsymbol{s}_\theta$
$\boldsymbol{v}_{\text{phase},l}$
$\hat{\boldsymbol{r}}_l$
$\boldsymbol{v}_{\text{mag},l}$
$\hat{\boldsymbol{\theta}}_l$
Mag-DetNet Block
$\hat{\boldsymbol{r}}_{l+1}$
$\hat{\boldsymbol{\theta}}_l$
does not change
Phase-DetNet Block
does not change
$\hat{\boldsymbol{r}}_{l+1}$
$\hat{\boldsymbol{\theta}}_{l+1}$
$\boldsymbol{v}_{\text{phase},l+1}$
$\boldsymbol{v}_{\text{mag},l+1}$

(a) MagPhase-DetNet block


$\boldsymbol{\Psi}_e, \boldsymbol{\Psi}_e, \boldsymbol{y}_e, \boldsymbol{y}_o, \boldsymbol{s}_r, \boldsymbol{s}_\theta$
$\hat{\boldsymbol{r}}_0$
$\hat{\boldsymbol{\theta}}_0$
$\boldsymbol{v}_{\text{mag},0}$
$\boldsymbol{v}_{\text{phase},0}$
MagPhase-DetNet Block
$\hat{\boldsymbol{r}}_1$
$\hat{\boldsymbol{\theta}}_1$
$\boldsymbol{v}_{\text{mag},1}$
$\boldsymbol{v}_{\text{phase},1}$
$\hat{\boldsymbol{r}}_{L-1}$
$\hat{\boldsymbol{\theta}}_{L-1}$
$\boldsymbol{v}_{\text{mag},L-1}$
$\boldsymbol{v}_{\text{phase},L-1}$
$\hat{\boldsymbol{r}}_L$
$\hat{\boldsymbol{\theta}}_L$
$\boldsymbol{v}_{\text{mag},L}$
$\boldsymbol{v}_{\text{phase},L}$

(b) MagPhase-DetNet layer conections

Figure 4.1: MagPhase-DetNet architecture.

4.3.1 Magnitude phase reparametrization

Before defining the architecture in detail, we have to do a small reparametrization of $\boldsymbol{x}$ form cartesian to polar form as follows:

$$\tilde{\boldsymbol{x}} = \begin{bmatrix} \boldsymbol{r} \\ \boldsymbol{r} \end{bmatrix} \odot \begin{bmatrix} \cos(\boldsymbol{\theta}) \\ \sin(\boldsymbol{\theta}) \end{bmatrix} \tag{4.23}$$

with $\boldsymbol{r}, \boldsymbol{\theta} \in \mathbb{R}^{n+\widetilde{M}}$ contains the magnitude and phase of each symbol, the cosine and sine are applied element wise, and the $\odot$ operator is the elementwise multiplication.

If we define $\boldsymbol{R}(\boldsymbol{r}) = \begin{bmatrix}\boldsymbol{r} & \boldsymbol{r}\end{bmatrix}^T$ and $\boldsymbol{w}(\boldsymbol{\theta}) = \begin{bmatrix}\cos(\boldsymbol{\theta}) & \sin(\boldsymbol{\theta})\end{bmatrix}^T$, the element wise product is equivalent to the next two matrix vector multiplications:

$$\tilde{X} = \boldsymbol{R}(\boldsymbol{r}) \odot \boldsymbol{w}(\boldsymbol{\theta}) = \operatorname{diag}\big(\boldsymbol{R}(\boldsymbol{r})\big) \cdot \boldsymbol{w}(\boldsymbol{\theta}) = \operatorname{diag}\big(\boldsymbol{w}(\boldsymbol{\theta})\big) \cdot \boldsymbol{R}(\boldsymbol{r}) \tag{4.24}$$

which are useful to calculate the gradients.

4.3.2 Magnitude DetNet block

For the magnitude block, each block should perform a gradient descent step with respecto to the magnitude of the symbols, and assuming the phases constant, hence the first step to define the block structure is to calculate the following gradient:

$$\nabla_{\boldsymbol{r}} \left\| \boldsymbol{y}_{\mathrm{e}} - |\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}|^{\circ 2} \right\|^2 + \left\| \boldsymbol{y}_{\mathrm{o}} - |\boldsymbol{\Psi}_{\mathrm{o}}\tilde{\boldsymbol{x}}|^{\circ 2} \right\|^2 \tag{4.25}$$

$$= \nabla_{\boldsymbol{r}} \left\| \boldsymbol{y}_{\mathrm{e}} - \left|\boldsymbol{\Psi}_{\mathrm{e}} \cdot \operatorname{diag}\big(\boldsymbol{w}(\boldsymbol{\theta})\big) \cdot \boldsymbol{R}(\boldsymbol{r})\right|^{\circ 2} \right\|^2 + \left\| \boldsymbol{y}_{\mathrm{o}} - \left|\boldsymbol{\Psi}_{\mathrm{o}} \cdot \operatorname{diag}\big(\boldsymbol{w}(\boldsymbol{\theta})\big) \cdot \boldsymbol{R}(\boldsymbol{r})\right|^{\circ 2} \right\|^2 \tag{4.26}$$

which we will name $\mathrm{grad}_{\mathrm{mag}}$, and after some calculations we get:

$$\mathrm{grad}_{\mathrm{mag}} = \boldsymbol{A}_{\mathrm{e}} \cdot \left(2\boldsymbol{y}_{\mathrm{e}} - 2\,|\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}|^{\circ 2}\right) + \boldsymbol{A}_{\mathrm{o}} \cdot \left(2\boldsymbol{y}_{\mathrm{o}} - 2\,|\boldsymbol{\Psi}_{\mathrm{o}}\tilde{\boldsymbol{x}}|^{\circ 2}\right) \tag{4.27}$$

where

$$\boldsymbol{A}_{\mathrm{e},l} = \begin{bmatrix}\operatorname{diag}\big(\cos(\boldsymbol{\theta})\big) & \operatorname{diag}\big(\sin(\boldsymbol{\theta})\big)\end{bmatrix} \cdot \boldsymbol{\Psi}_{\mathrm{e}}^T \cdot \begin{bmatrix}\operatorname{diag}\left(\left[2\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}_l\right]_0^n\right) \\ \operatorname{diag}\left(\left[2\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}_l\right]_n^{2n}\right)\end{bmatrix} \tag{4.28}$$

$$\boldsymbol{A}_{\mathrm{o},l} = \begin{bmatrix}\operatorname{diag}\big(\cos(\boldsymbol{\theta})\big) & \operatorname{diag}\big(\sin(\boldsymbol{\theta})\big)\end{bmatrix} \cdot \boldsymbol{\Psi}_{\mathrm{o}}^T \cdot \begin{bmatrix}\operatorname{diag}\left(\left[2\boldsymbol{\Psi}_{\mathrm{o}}\tilde{\boldsymbol{x}}_l\right]_0^n\right) \\ \operatorname{diag}\left(\left[2\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}_l\right]_n^{2n}\right)\end{bmatrix} \tag{4.29}$$

where $[\cdot]_a^b$ denotes the the elements from position a to b (not included) of a vector, and the l subindex denotes the l-th layer of the network.

Now following the procedure from [12], we give the architecture of the magnitude network, where each layer is given by:

$$\boldsymbol{r}_{l-1} = \begin{bmatrix} \boldsymbol{s}_r \\ \hat{\boldsymbol{r}}_{l-1} \end{bmatrix} \tag{4.30}$$

$$\boldsymbol{\theta}_{l-1} = \begin{bmatrix} \boldsymbol{s}_\theta \\ \hat{\boldsymbol{\theta}}_{l-1} \end{bmatrix} \tag{4.31}$$

$$\tilde{\boldsymbol{x}}_{l-1} = \boldsymbol{R}(\boldsymbol{r}_{l-1}) \odot \boldsymbol{w}(\boldsymbol{\theta}_{l-1}) \tag{4.32}$$

$$\begin{aligned}\boldsymbol{q}_l = \boldsymbol{r}_{l-1} - \delta_{1l}\boldsymbol{A}_{\mathrm{e},l-1}\boldsymbol{y}_\mathrm{e} + \delta_{2l}\boldsymbol{A}_{\mathrm{e},l-1}|\boldsymbol{\Psi}_\mathrm{e}\tilde{\boldsymbol{x}}_{l-1}|^{\circ 2} - \\ \delta_{3l}\boldsymbol{A}_{\mathrm{o},l-1}\boldsymbol{y}_\mathrm{o} + \delta_{4l}\boldsymbol{A}_{\mathrm{o},l-1}|\boldsymbol{\Psi}_\mathrm{o}\tilde{\boldsymbol{x}}_{l-1}|^{\circ 2}\end{aligned} \tag{4.33}$$

$$\boldsymbol{z}_l = \rho\left(\boldsymbol{W}_{1l}\begin{bmatrix} \boldsymbol{q}_l \\ \boldsymbol{v}_{l-1} \end{bmatrix} + \boldsymbol{b}_{1l}\right) \tag{4.34}$$

$$\boldsymbol{r}_{\mathrm{oh},l} = \boldsymbol{r}_{\mathrm{oh},l-1} + \boldsymbol{W}_{2l}\boldsymbol{z}_l + \boldsymbol{b}_{2l} \tag{4.35}$$

$$\hat{\boldsymbol{r}}_l = \left[\boldsymbol{f}_\mathrm{oh}(\boldsymbol{r}_{\mathrm{oh},l})\right]_{\widetilde{M}}^{\mathrm{end}} \tag{4.36}$$

$$\boldsymbol{v}_{\mathrm{mag},l} = \boldsymbol{v}_{\mathrm{mag},l-1} + \boldsymbol{W}_{3l}\boldsymbol{z}_l + \boldsymbol{b}_{3l} \tag{4.37}$$

$$\hat{\boldsymbol{r}}_0 = \boldsymbol{0} \tag{4.38}$$

$$\boldsymbol{v}_{\mathrm{mag},0} = \boldsymbol{0} \tag{4.39}$$

$$\tag{4.40}$$

where $\rho(\cdot)$ is the ReLu activation function, $\boldsymbol{W}$ a matrix and $\boldsymbol{b}$ a vector that together apply a linear transformation, the function $\boldsymbol{f}_\mathrm{oh}(\cdot)$ is as defined in [12] and $[\boldsymbol{c}]_m^{\mathrm{end}}$ denotes the elements of $\boldsymbol{c}$ from m to the last.

The trainable parameters of the model are:

$$\boldsymbol{\Theta}_\mathrm{mag} = \left\{\boldsymbol{W}_{1l}, \boldsymbol{b}_{1l}, \boldsymbol{W}_{2l}, \boldsymbol{b}_{2l}, \boldsymbol{W}_{3l}, \boldsymbol{b}_{3l}, \delta_{1l}, \delta_{2l}, \delta_{3l}, \delta_{4l}\right\}_{l=1}^{L} \tag{4.41}$$

where L is the number of layers, and loss function used for training is:

$$\mathrm{loss}(\boldsymbol{r}, \hat{\boldsymbol{r}}(\boldsymbol{\Psi}_\mathrm{e}, \boldsymbol{\Psi}_\mathrm{e}, \boldsymbol{y}_\mathrm{e}, \boldsymbol{y}_\mathrm{o}; \boldsymbol{\Theta}_\mathrm{mag})) = \sum_{l=1}^{L} \log(l)\|\boldsymbol{r} - \hat{\boldsymbol{r}}_l\|^2 \tag{4.42}$$

4.3.3 Phase DetNet block

For the phase block, each block should perform a gradient descent step with respecto to the phase of the symbols, so we follow the same procedure as for the magnitude block, and calculate the following gradient:

$$\nabla_{\boldsymbol{\theta}} \left\|\boldsymbol{y}_\mathrm{e} - |\boldsymbol{\Psi}_\mathrm{e}\tilde{\boldsymbol{x}}|^{\circ 2}\right\|^2 + \left\|\boldsymbol{y}_\mathrm{o} - |\boldsymbol{\Psi}_\mathrm{o}\tilde{\boldsymbol{x}}|^{\circ 2}\right\|^2 \tag{4.43}$$

$$= \nabla_{\boldsymbol{\theta}} \left\|\boldsymbol{y}_\mathrm{e} - \left|\boldsymbol{\Psi}_\mathrm{e} \cdot \mathrm{diag}\big(\boldsymbol{R}(\boldsymbol{r})\big) \cdot \boldsymbol{w}(\boldsymbol{\theta})\right|^{\circ 2}\right\|^2 + \left\|\boldsymbol{y}_\mathrm{o} - \left|\boldsymbol{\Psi}_\mathrm{o} \cdot \mathrm{diag}\big(\boldsymbol{R}(\boldsymbol{r})\big) \cdot \boldsymbol{w}(\boldsymbol{\theta})\right|^{\circ 2}\right\|^2 \tag{4.44}$$

which we will name $\mathrm{grad}_\mathrm{phase}$, and after some calculations we get:

$$\mathrm{grad}_\mathrm{phase} = \boldsymbol{A}_\mathrm{e} \cdot \left(2\boldsymbol{y}_\mathrm{e} - 2\,|\boldsymbol{\Psi}_\mathrm{e}\tilde{\boldsymbol{x}}|^{\circ 2}\right) + \boldsymbol{A}_\mathrm{o} \cdot \left(2\boldsymbol{y}_\mathrm{o} - 2\,|\boldsymbol{\Psi}_\mathrm{o}\tilde{\boldsymbol{x}}|^{\circ 2}\right) \tag{4.45}$$

where

$$\boldsymbol{A}_{\mathrm{e},l} = \left[-\mathrm{diag}\left(\left[\tilde{\boldsymbol{x}}_l\right]_n^{2n}\right) \quad \mathrm{diag}\left(\left[\tilde{\boldsymbol{x}}_l\right]_0^n\right)\right] \cdot \boldsymbol{\Psi}_{\mathrm{e}}^T \cdot \begin{bmatrix} \mathrm{diag}\left(\left[2\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}_l\right]_0^n\right) \\ \mathrm{diag}\left(\left[2\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}_l\right]_n^{2n}\right) \end{bmatrix} \tag{4.46}$$

$$\boldsymbol{A}_{\mathrm{e},l} = \left[-\mathrm{diag}\left(\left[\tilde{\boldsymbol{x}}_l\right]_n^{2n}\right) \quad \mathrm{diag}\left(\left[\tilde{\boldsymbol{x}}_l\right]_0^n\right)\right] \cdot \boldsymbol{\Psi}_{\mathrm{e}}^T \cdot \begin{bmatrix} \mathrm{diag}\left(\left[2\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}_l\right]_0^n\right) \\ \mathrm{diag}\left(\left[2\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}_l\right]_n^{2n}\right) \end{bmatrix} \tag{4.47}$$

where $[\cdot]_a^b$ denotes the the elements from position a to b (not included) of a vector.

Now following the procedure from [12], we give the architecture of the magnitude network, where each layer is given by:

$$\boldsymbol{r}_{l-1} = \begin{bmatrix} \boldsymbol{s}_r \\ \hat{\boldsymbol{r}}_{l-1} \end{bmatrix} \tag{4.48}$$

$$\boldsymbol{\theta}_{l-1} = \begin{bmatrix} \boldsymbol{s}_\theta \\ \hat{\boldsymbol{\theta}}_{l-1} \end{bmatrix} \tag{4.49}$$

$$\tilde{\boldsymbol{x}}_{l-1} = \boldsymbol{R}(\boldsymbol{r}_{l-1}) \odot \boldsymbol{w}(\boldsymbol{\theta}_{l-1}) \tag{4.50}$$

$$\begin{aligned} \boldsymbol{q}_l = \boldsymbol{\theta}_{l-1} - \delta_{1l}\boldsymbol{A}_{\mathrm{e},l-1}\boldsymbol{y}_{\mathrm{e}} + \delta_{2l}\boldsymbol{A}_{\mathrm{e},l-1}|\boldsymbol{\Psi}_{\mathrm{e}}\tilde{\boldsymbol{x}}_{l-1}|^{\circ 2} - \\ \delta_{3l}\boldsymbol{A}_{\mathrm{o},l-1}\boldsymbol{y}_{\mathrm{o}} + \delta_{4l}\boldsymbol{A}_{\mathrm{o},l-1}|\boldsymbol{\Psi}_{\mathrm{o}}\tilde{\boldsymbol{x}}_{l-1}|^{\circ 2} \end{aligned} \tag{4.51}$$

$$\boldsymbol{z}_l = \rho\left(\boldsymbol{W}_{1l}\begin{bmatrix} \boldsymbol{q}_l \\ \boldsymbol{v}_{l-1} \end{bmatrix} + \boldsymbol{b}_{1l}\right) \tag{4.52}$$

$$\boldsymbol{\theta}_{\mathrm{oh},l} = \boldsymbol{\theta}_{\mathrm{oh},l-1} + \boldsymbol{W}_{2l}\boldsymbol{z}_l + \boldsymbol{b}_{2l} \tag{4.53}$$

$$\hat{\boldsymbol{\theta}}_l = \left[\boldsymbol{f}_{\mathrm{oh}}(\boldsymbol{\theta}_{\mathrm{oh},l})\right]_{\widetilde{M}}^{\mathrm{end}} \tag{4.54}$$

$$\boldsymbol{v}_{\mathrm{phase},l} = \boldsymbol{v}_{\mathrm{phase},l-1} + \boldsymbol{W}_{3l}\boldsymbol{z}_l + \boldsymbol{b}_{3l} \tag{4.55}$$

$$\hat{\boldsymbol{\theta}}_0 = \boldsymbol{0} \tag{4.56}$$

$$\boldsymbol{v}_{\mathrm{phase},0} = \boldsymbol{0} \tag{4.57}$$

$$\tag{4.58}$$

where $\rho(\cdot)$ is the ReLu activation function, $\boldsymbol{W}$ a matrix and $\boldsymbol{b}$ a vector that together apply a linear transformation, and the function $\boldsymbol{f}_{\mathrm{oh}}(\cdot)$ is as defined in [12].

The trainable parameters of the model are:

$$\boldsymbol{\Theta}_{\mathrm{phase}} = \left\{\boldsymbol{W}_{1l}, \boldsymbol{b}_{1l}, \boldsymbol{W}_{2l}, \boldsymbol{b}_{2l}, \boldsymbol{W}_{3l}, \boldsymbol{b}_{3l}, \delta_{1l}, \delta_{2l}, \delta_{3l}, \delta_{4l}\right\}_{l=1}^{L} \tag{4.59}$$

where L is the number of layers, and loss function used for training is:

$$\mathrm{loss}(\boldsymbol{\theta}, \hat{\boldsymbol{\theta}}(\boldsymbol{\Psi}_{\mathrm{e}}, \boldsymbol{\Psi}_{\mathrm{e}}, \boldsymbol{y}_{\mathrm{e}}, \boldsymbol{y}_{\mathrm{o}}; \boldsymbol{\Theta}_{\mathrm{phase}})) = \sum_{l=1}^{L} \log(l)\|\boldsymbol{\theta} - g_{\text{diff deco}}(\hat{\boldsymbol{\theta}}_l)\|^2 \tag{4.60}$$

where $g_{\text{diff deco}}(\cdot)$ is a function that implements the the differential decoding, and depends on the constellation used.

4.4 Numerical simulation

To implement the proposed architecture we used the Pytorch framework, the code can be found in Direct_Detection_with_Phase_Recovery. For training as well as evaluating the network we use the simulation parameters shown in table 3.1, the constellation used is DD-SQAM, and we focused on the case when the auxiliary channel and real channel are equal, i.e.$\widetilde{N} = \widetilde{M}$, in particular for $\widetilde{M} = 1, 3, 5$. Finally we considered the SNR range from 0 dB to 20 dB and for each SNR and $\widetilde{M}$ a different model is trained.

4.4.1 Training process

4.4.1.1 Hyper parameters

For the architecture we used the following lengths of the vectors and number of layers:

$$\mathrm{len}(\tilde{\boldsymbol{x}}) = 2\widetilde{M} + 1 \tag{4.61}$$

$$\mathrm{len}(\boldsymbol{v}) = 2 \cdot (2\widetilde{M} + 1) \tag{4.62}$$

$$\mathrm{len}(\boldsymbol{z}) = 4 \cdot (2\widetilde{M} + 1) \tag{4.63}$$

$$L = \max(3 \cdot (2\widetilde{M} + 1), 30) \tag{4.64}$$

where len(·) denotes the length of the vector and max() returns the biggest element of the argument.

4.4.1.2 Training set

The training set consist of the output of the system after simulating the transmission of $2\widetilde{M} + 1$ random symbols, where the first $\widetilde{M}$ represent the state of the channel, and the last $\widetilde{M} + 1$ symbols are transmitted symbols. As a result the network should be trained for all possible states and transmitted symbols.

For the training we use a variable batch size that take each of the following values in order

$$[100, 400, 1000, 2000, 5000, 10000]$$

and we use 300 batches for each batch size.

4.4.2 Evaluation process

4.4.2.1 Normal testing

For the evaluation process we perform two evaluation, the first one is a evaluation equal to training, that means using many small transmissions of $2\widetilde{M} + 1$ symbols, where the first $\widetilde{M}$ symbols represent the state of the channel and are passed to the MagPhase-DetNet as an input (that means assuming perfect channel state knowledge), and we evaluate the SER of the next symbols as shown in figure **??**. We do this because we noticed that the reliability of the detection decreases for the last symbols.

After doing this and plotting the results we get the graph of figure 4.2, there is shown the performance, in terms of SER, of the models trained for each channel memory and SNR. The graph is to be interpreted as the best performance that the architecture can achieve when trained for a specific SNR.

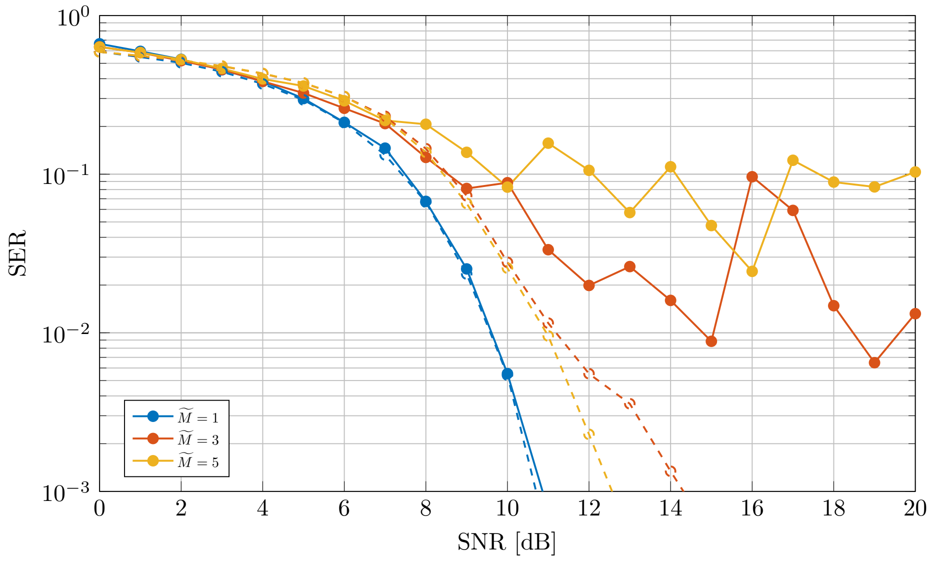


Figure 4.2: Performance of the MagPhase-DetNet trained for different SNR and channel memory. Dotted line is the reference optimal performance.

Analysing the results it is easy to notice that for the $\widetilde{M} = 1$ case the performance of the MagPhase-DetNet is almost optimal. In contrast for $\widetilde{M} = 3$ and 5 the performance is near to the optimal only for slow SNR, at bigger SNR the architecture tends to have an error floor. Also is important to notice that the graph is not smooth, this could happen because the training process was not sufficient.

4.4.2.2 Sequential testing

The next testing done tries to implement the network as it would be implemented in a practical application, where one do not have access to different segments of transmission with perfect channel knowledge, but only to a long string of samples. That is why we use the network as shown in figure **??**, we start with an initial channel knowledge and use the network to decode the next symbol. Then with the decoded symbol we use it to determine the new state of the channel and with it decode the next symbol, and the process keeps going until the end of the transmission.

Clearly the performance for the $\widetilde{M} = 1$ case is again almost optimal, however the other two cases are really bad, they show an error floor pretty big, nevertheless the error floor is below 3/4 which means that the decoding system is better than randomly guess the symbols. This suggest that the system try to work but there may be a severe error propagation that degrades the performance. This shows that the proposed architecture is very dependent of the quality of the channel state knowledge, and a small error can propagate a lot.

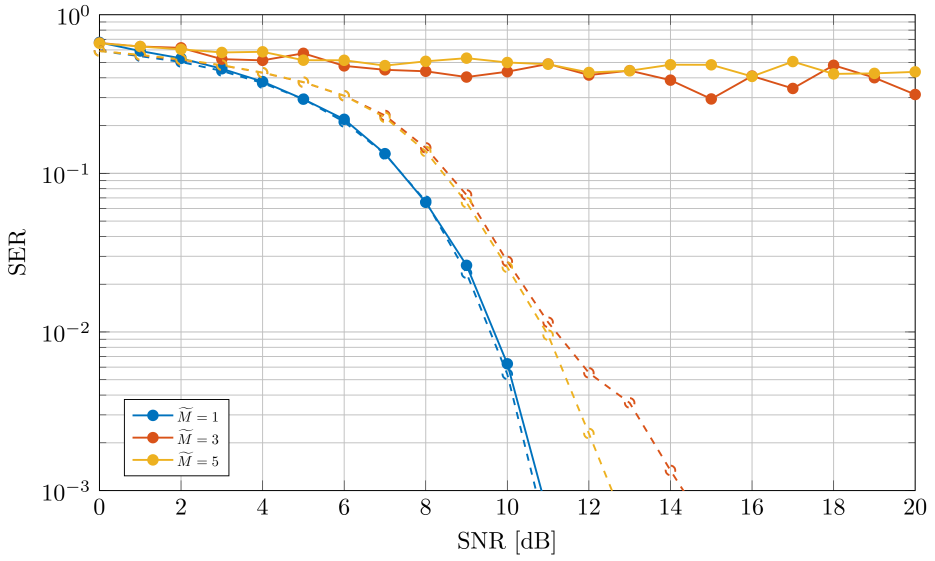


Figure 4.3: default

Bibliography

[1] A. Tasbihi and F. R. Kschischang, "Direct detection under tukey signalling," *Journal of Lightwave Technology*, vol. 39, no. 21, pp. 6845–6857, 2021. DOI: 10.1109/JLT.2021.3109852.

[2] G. P. Agrawal, "Introduction," in *Fiber-Optic Communication Systems*. John Wiley & Sons, Ltd, 2010, ch. 1, pp. 1–23, ISBN: 9780470918524. DOI: https://doi.org/10.1002/9780470918524.ch1. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470918524.ch1. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470918524.ch1.

[3] A. Mecozzi and M. Shtaif, "Information capacity of direct detection optical transmission systems," *Journal of Lightwave Technology*, vol. 36, no. 3, pp. 689–694, 2018. DOI: 10.1109/JLT.2017.2777188.

[4] G. P. Agrawal, "Optical receivers," in *Fiber-Optic Communication Systems*. John Wiley & Sons, Ltd, 2010, ch. 4, pp. 128–181, ISBN: 9780470918524. DOI: https://doi.org/10.1002/9780470918524.ch4. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/9780470918524.ch4. [Online]. Available: https://onlinelibrary.wiley.com/doi/abs/10.1002/9780470918524.ch4.

[5] A. Tasbihi and F. R. Kschischang, "On the capacity of waveform channels under square-law detection of time-limited signals," *IEEE Transactions on Information Theory*, vol. 66, no. 11, pp. 6682–6687, 2020. DOI: 10.1109/TIT.2020.2999889.

[6] M. Secondini and E. Forestieri, "Direct detection of bipolar pulse amplitude modulation," *Journal of Lightwave Technology*, vol. 38, no. 21, pp. 5981–5990, 2020. DOI: 10.1109/JLT.2020.3007584.

[7] D. Plabst, T. Prinz, T. Wiegart, *et al.*, "Achievable rates for short-reach fiber-optic channels with direct detection," *Journal of Lightwave Technology*, vol. 40, no. 12, pp. 3602–3613, 2022. DOI: 10.1109/JLT.2022.3149574.

[8] D. J. C. MacKay, *Information theory, inference, and learning algorithms*, 15th print. Cambridge [u.a.]: Cambridge University Press, 2015, ISBN: 9780521642989. [Online]. Available: http://bvbr.bib-bvb.de:8991/F?func=service&doc_library=BVB01&doc_number=020774035&line_number=0001&func_code=DB_RECORDS&service_type=MEDIA.

[9] T. Prinz, D. Plabst, T. Wiegart, S. Calabrò, N. Hanik, and G. Kramer, *Successive interference cancellation for bandlimited channels with direct detection*, 2023. arXiv: 2212.07761 [cs.IT].

[10] T. Wang, T. Lv, H. Gao, and S. Zhang, "Joint multiple symbol differential detection and channel decoding for noncoherent uwb impulse radio by belief propagation," *IEEE Transactions on Wireless Communications*, vol. 16, no. 1, pp. 293–306, 2017. DOI: 10.1109/TWC.2016.2623301.

[11] N. Samuel, T. Diskin, and A. Wiesel, "Deep mimo detection," in *2017 IEEE 18th International Workshop on Signal Processing Advances in Wireless Communications (SPAWC)*, 2017, pp. 1–5. DOI: 10.1109/SPAWC.2017.8227772.

[12] N. Samuel, T. Diskin, and A. Wiesel, "Learning to detect," *IEEE Transactions on Signal Processing*, vol. 67, no. 10, pp. 2554–2564, 2019. DOI: 10.1109/TSP.2019.2899805.