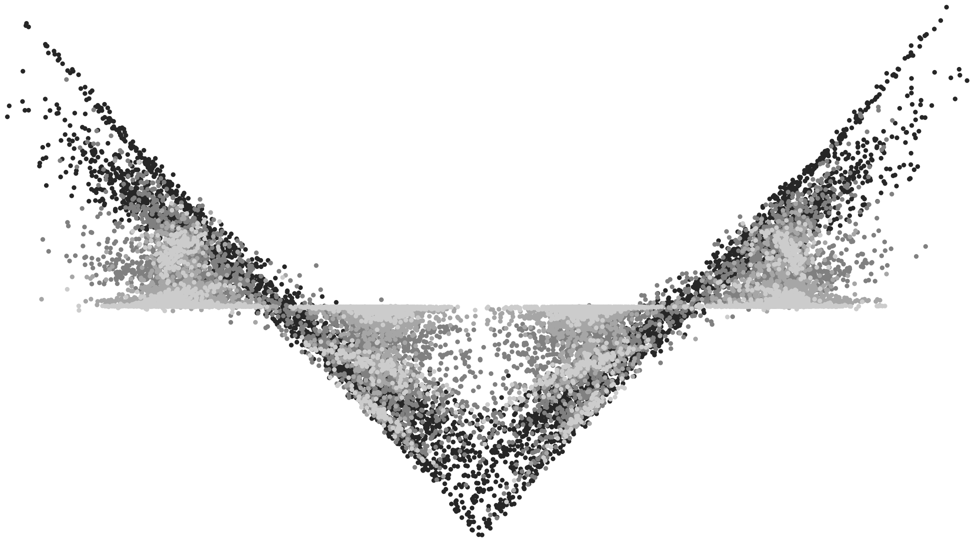
CHALMERS
UNIVERSITY OF TECHNOLOGY



Improving log-likelihood ratio calculation for LDPC decoding in presence of residual phase noise

Master's thesis in Information and Communication Technology

SIMON IMARK & THEODOR HULT BERÉNYI

DEPARTMENT OF ELECTRICAL ENGINEERING

CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2023
www.chalmers.se

Master's thesis 2023

Improving log-likelihood ratio calculation for LDPC decoding in presence of residual phase noise

SIMON IMARK & THEODOR HULT BERÉNYI

Department of Electrical Engineering
Division of Communication systems
Chalmers University of Technology
Gothenburg, Sweden 2023

Improving log-likelihood ratio calculation for LDPC decoding in presence of residual phase noise
SIMON IMARK & THEODOR HULT BERÉNYI

Supervisor: Christoffer Fougstedt, Ericsson,
Examiner: Alexandre Graell i Amat, Chalmers University of Technology, Department of Electrical Engineering

Master's Thesis 2023
Department of Electrical Engineering
Division of Communication Systems
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: log-likelihood ratios for the least significant bit in a 16-QAM SISO system for different in-phase amplitudes under increasing SNR.

Typeset in LaTeX
Gothenburg, Sweden 2023

Improving log-likelihood ratio calculation for LDPC decoding in presence of residual phase noise

SIMON IMARK & THEODOR HULT BERÉNYI
Department of Electrical Engineering
Chalmers University of Technology

Abstract

In the ever-evolving landscape of communication systems, the primary objective is to ensure efficient and reliable transmission of information across a physical medium, commonly referred to as the channel. The pioneering work of Claude E. Shannon showed that channel coding can harness the full potential of information transfer, enabling the attainment of channel capacity. A coding scheme extensively used in digital communication is low-density parity-check (LDPC) codes that provides near-optimal error-correction at low complexity. The most common decoding algorithm for LDPC codes is the iterative belief propagation algorithm. This iterative process involves exchanging messages in the form of log-likelihood ratios (LLR), e.g. information about the probability of the decoded bit being either a 0 or a 1. Typically, the LLR calculation assumes an additive white Gaussian noise (AWGN) channel. However, there may be other types of noise affecting the received signal, such as residual phase noise due to imperfect phase estimation. This thesis investigates the performance of LDPC codes if the LLR calculation is extended with the information about phase noise over a single-input single-output (SISO) and a 2×2 multiple-input multiple-output (MIMO) channel. It is shown that it is possible to extend the conventional LLR calculation to consider phase noise in addition to AWGN and consequently improve the coding gain in the presence of residual phase noise. Hardware synthesis simulations showed that the proposed PN-LLR scheme significantly increased the hardware resource usage compared to the conventional AWGN-LLR scheme. However, the absolute majority of the increase is attributed to the high cost of divisions in hardware.

Keywords: channel coding, LDPC, LLR, SISO, MIMO, phase noise

Acknowledgements

We would like to express our gratitude to Christoffer, our supervisor, for his contribution, guidance, and unwavering support. We would also like to thank Ericsson for the opportunity to conduct this thesis project. Finally, we acknowledge the contributions of all those who supported us during the thesis project. Thank you.

Simon Imark & Theodor Hult Berényi, June 1, 2023

List of Acronyms

Below is the list of acronyms that have been used throughout this thesis listed in alphabetical order:

AWGN	additive white Gaussian noise
BER	bit error rate
CN	check node
DSP	digital signal processing
FF	flip-flops
FPGA	field programmable gate array
HLS	high-level synthesis
LDPC	low-density parity-check
LLR	log-likelihood ratio
LUT	look-up table
MAP	maximum a posteriori
MIMO	multiple-input multiple-output
ML	maximum likelihood
PN	phase noise
RPN	residual phase noise
RTL	register-transfer level
SISO	single-input single-output
SPC	single parity-check code
SNR	signal-to-noise ratio
VN	variable node
QAM	quadrature amplitude modulation

Contents

1

Introduction

1.1 Background

In the ever-evolving landscape of communication systems, the primary objective is to ensure efficient and reliable transmission of information across a physical medium, commonly referred to as the channel. The pioneering work of Claude E. Shannon in his seminal paper "A Mathematical Theory of Communication" [1] laid the foundation for understanding the representation of information as binary sequences and the subsequent operations necessary for successful transmission and reception. A key finding of Shannon's work was the realisation that channel coding can harness the full potential of information transfer, enabling the attainment of channel capacity. The series of subsequent blocks of operations of a communication system are often illustrated with Shannon's Communication model, which is shown in figure 1.1.

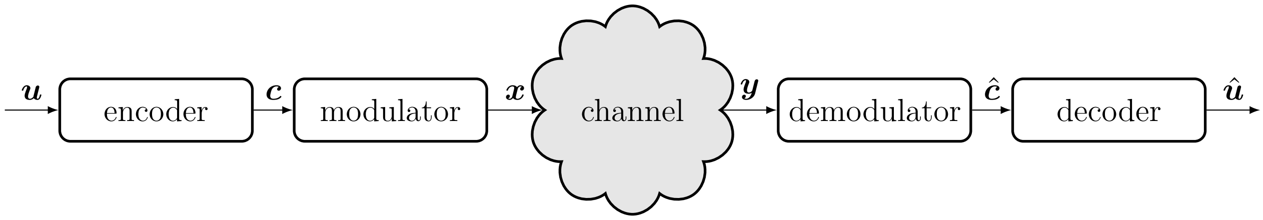


Figure 1.1: A communication model illustrating the subsequent blocks of operation contained in a classical communication system.

From the information source binary information is fed into the system by the vector $\boldsymbol{u}$. On the transmitter side the binary sequence, also called bit stream, is encoded into the codeword $\boldsymbol{c}$ and then mapped to (complex) symbols based on the used constellation scheme. The constellation determines the mapping from bits to symbols. The mapping is part of the modulation block, which may also include pulse shaping and adding frequency carrier to place the transmitted signal $\boldsymbol{x}$ in the desired frequency band depending on the application. The modulated signal is transmitted over the physical channel and then demodulated at the receiver, i.e. the received signal $\boldsymbol{y}$ is fed through a sequence of operations and mapped back to the codeword bits. Since this operation will be an estimation, the received codeword is denoted $\hat{\boldsymbol{c}}$. The signal is then decoded into the estimation $\hat{\boldsymbol{u}}$ of the transmitted bits. When sending information over the channel, the physical medium will introduce noise which distorts the original signal and may create errors in the received signal. A common way of modelling the noisy channel is as a memoryless discrete-time additive white Gaussian noise (AWGN) channel. This simple channel model can be extended by introducing e.g. phase noise as we will see later on in chapter 2. However, before that we will restrict our self to the AWGN channel to lay the necessary foundation. For the AWGN model, the received signal can be expressed as

$$\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n} \tag{1.1}$$

where $\boldsymbol{n}$ is the zero-mean white Gaussian noise, which is independent from the transmitted signal $\boldsymbol{x}$. Depending on the quality of the physical channel the power of the noise will vary, which will limit the amount of information that can be transmitted reliably over the channel for a given signal power. Shannon's channel coding theorem shows that it is possible to transmit information nearly error-free if the communication rate R is below the channel capacity C. The channel capacity over the continuous-time AWGN channel can be expressed as

$$C = W \cdot \log_2(1 + \mathsf{SNR}) \tag{1.2}$$

where W is the signal bandwidth. When designing a communication system, it is desired to maximise the rate at which information can be transmitted reliably while also minimising the signal power. However, there is a trade off between the so called signal-to-noise ratio (SNR) and the capacity of the channel. The SNR can be written as $R \cdot E_b/N_0$ where R is the data rate and E_b/N_0 the bit energy over noise spectral density. The minimum E_b/N_0 needed to ensure reliable communication at rate R over a complex AWGN channel is given by

$$\frac{E_b}{N_0} \geq \frac{2^R - 1}{R}. \tag{1.3}$$

This shows that a larger rate R require larger energy per bit. In contrary, if $R \to 0$, the minimum SNR to achieve reliable transmission over an AWGN channel is given by

$$\lim_{R \to 0} \frac{2^R - 1}{R} = \ln 2 = -1.59 \text{ dB}. \tag{1.4}$$

The reliability of the system can be measured by the bit-error-rate (BER), i.e., the fraction of bits erroneously received. With limited SNR available, the signal distortion will increase, hence the probability of errors in the received message will also increase. In practice, uncoded transmission performs far from capacity meaning that for a given BER, much larger SNR is required than stated in eq. 1.2. It is desirable to find schemes that can transmit reliably at a given BER for as low SNR as possible, i.e. as close to the limit of channel capacity as possible. As previously stated, this limit can be achieved by the use of channel coding.

1.1.1 Optimal decoding rules

Before feeding the channel output to the decoder, the received signal y have to be demodulated, which for a wireless communication system typically involves removing the frequency carrier and filtering the signal. After that the signal y, consisting of noisy (complex) symbols, must be mapped back to bits. This requires a decision rule that specifies which original symbol x that is most likely to have been transmitted given the received channel output y and hence minimises the probability of error. It can be shown that the optimal decoding rule for the AWGN channel is the maximum a posteriori (MAP) rule where the estimated symbol $\hat{x}$ from the set symbols χ is given by

$$\hat{\boldsymbol{x}}_{\mathsf{MAP}} = \arg\max_{\boldsymbol{x} \in \chi} p(\boldsymbol{x}|\boldsymbol{y}) = \arg\max_{\boldsymbol{x} \in \chi} p(\boldsymbol{y}|\boldsymbol{x})p(\boldsymbol{x}), \tag{1.5}$$

where the posterior probability $p(\boldsymbol{x}|\boldsymbol{y})$ is rewritten as the product of the likelihood $p(\boldsymbol{y}|\boldsymbol{x})$ and prior $p(\boldsymbol{x})$ by using Bayes' theorem. If the prior $p(\boldsymbol{x})$ is uniform, i.e. if the system have equiprobable symbols, the MAP-rule simplifies to the maximum-likelihood (ML) rule

$$\hat{\boldsymbol{x}}_{\mathsf{ML}} = \arg\max_{\boldsymbol{x}\in\mathcal{X}} p(\boldsymbol{y}|\boldsymbol{x}). \tag{1.6}$$

1.1.2 Error correcting codes

The general idea of coding is to add redundancy to the bit stream in a controlled and clever way, by mapping blocks of information bits u into a set of codewords $\boldsymbol{c}_i \in \mathcal{C}$, so that the receiver can detect and correct errors in the received message. With more added redundancy, the error correcting capability of the code improves at the cost of reduced effective throughput of information bits u. As the error correction capabilities increases with channel coding, less E_b/N_0 is required to achieve the same BER. This is known as the coding gain. The redundancy of a code is measured through the code rate R_c, which is defined as

$$R_c = \frac{k}{n} < 1 \tag{1.7}$$

where k is the number of information bits that are mapped into codewords $\boldsymbol{c}$ of length n. Naturally, it is desirable to design codes that can detect many errors, while adding little redundancy. There exist both linear and non-linear codes, which describes how the information bits are mapped into the codewords. However, most codes used in practice today are linear codes since they can be designed to reach capacity while still being relatively efficient to encode and decode [2]. An important subgroup of linear codes are linear block codes.

1.1.3 Linear block codes

A linear block code is a block code $\mathcal{C}(n,k)$ with 2^k codewords $\boldsymbol{c}_1, ..., \boldsymbol{c}_k$ that together span a k-dimensional subspace of the n-dimensional space $\{0,1\}^n$. Linear block codes follow three important properties. Firstly, linear block codes always contain the all-zero codeword. Secondly, the set of codewords is closed under modulo-2 addition, i.e., $\boldsymbol{c} + \tilde{\boldsymbol{c}} \in \mathcal{C}, \forall\ \boldsymbol{c}, \tilde{\boldsymbol{c}} \in \mathcal{C}$. Thirdly, the minimum Hamming distance $d_{h,\min}$ of $\mathcal{C}$ equals the minimum Hamming weight of the code words, excluding the all-zero codeword, which can be expressed as

$$d_{H,\min}(\mathcal{C}) = \arg\min_{\boldsymbol{c},\tilde{\boldsymbol{c}}\in C} d_H(\boldsymbol{c},\tilde{\boldsymbol{c}}) = \arg\min_{\boldsymbol{c}\in\mathcal{C}} w_H(\boldsymbol{c}) = w_{H,\min}(\mathcal{C}), \tag{1.8}$$

where the Hamming weight w_H is defined as the number of non-zero elements for a binary sequence and the Hamming distance d_H is the number of positions with different values when comparing two binary vectors of equal length. The minimum Hamming distance $d_{H,\min}(\mathcal{C})$ is an important metric for linear codes since it determines the error correction capabilities of a linear code. A code with $d_{H,\min}(\mathcal{C}) = d_{\min}$ can correct up to

$$t = \left\lfloor \frac{d_{\min} - 1}{2} \right\rfloor \tag{1.9}$$

number of errors and detect $d_{\min} - 1$ errors. Hence, a large $d_{\min}$ is desirable. Another feature of linear block codes is that the encoding can be performed as a matrix multiplication according to

$$\boldsymbol{c} = \boldsymbol{u}\boldsymbol{G}, \tag{1.10}$$

where $\boldsymbol{G}$ is the generator matrix that spans the code $\mathcal{C}$. In addition to the generator matrix, there is also the parity check matrix $\boldsymbol{H}$ which is used for decoding the linear block code. The parity check matrix is in fact the generator matrix of the null space $\mathcal{C}_{\perp}$ of $\mathcal{C}$, hence $\boldsymbol{H}$ is related to the generator matrix according to

$$\boldsymbol{G}\boldsymbol{H}^T = \emptyset_{k\times(n-k)}, \tag{1.11}$$

which means that if the received codeword is error free, i.e., $\hat{\boldsymbol{c}} = \boldsymbol{c}_i, \forall\ \boldsymbol{c}_i \in \mathcal{C}$, then

$$\hat{\boldsymbol{c}}\boldsymbol{H}^T = \hat{\boldsymbol{u}}\boldsymbol{G}\boldsymbol{H}^T = \emptyset. \tag{1.12}$$

If $\hat{\boldsymbol{c}}$ is not equal to any codeword, then equation 1.12 is non-zero. An example of a parity check matrix $\boldsymbol{H}$ is

$$\boldsymbol{H} = \begin{bmatrix} 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}. \tag{1.13}$$

1.1.4 Hard and soft decisions

In the communication model shown in Figure 1.1 the symbol mapper is outputting the estimated codewords $\hat{\boldsymbol{c}}$ as a *hard* decision. This means that the output is strictly quantised to either a 0 or a 1 based on the optimal decoding rule and the given constellation. It is then easy to check by eq. (1.12) if the estimated codeword is correct as the product must result in the zero vector. However, as a hard decision is made a lot of information is lost when the result is quantised. Alternatively, by instead incorporating the *soft* information in the decoding, which includes the probability associated with each received code word bit, the decoder can provide a more accurate prediction. A class of linear block codes using soft information for decoding, and also of particular interest in this thesis, are low-density parity-check (LDPC) codes, which provides excellent, although sub-optimal, performance close to capacity.

1.1.5 LDPC codes

LDPC codes were first introduced by Robert G. Gallager in 1962 [3] but were forgotten due to the limited computational power available at that time. In 1981 Michael Tanner provided a graphical representation of LDPC codes through a bipartite graph [4], later called Tanner graphs, and generalised the ideas by Gallager. Nevertheless, it was not until the mid 90's when the true potential of the LDPC-codes was going to be recognised when David J. C. MacKay and others reinvented them, apparently independent of Gallager's previous work [5]. MacKay showed that the LDPC codes could match the performance of the current state-of-the-art turbo codes. Today, LDPC codes are used in current 5G NR cellular networks [6].

As LDPC codes are linear block codes they can fully be represented by their $n \times m$ parity-check matrix $\boldsymbol{H}$. From the name, this matrix is sparse in an LDPC code, which means that the Hamming weight $w_{\mathsf{r},i}$ of row i is $w_{\mathsf{r},i} \ll m$ and similarly the Hamming weight $w_{\mathsf{c},i}$ of column i is $w_{\mathsf{c},i} \ll n$. This is a especially important property that ensures low complexity [2]. Furthermore,

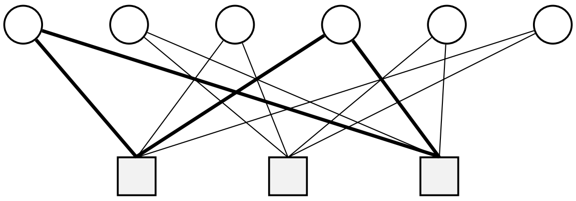


Figure 1.2: Tanner graph of the parity-check matrix $\boldsymbol{H}$ in eq. (1.13). The circles represents the VN:s while squares represents the CN:s. The bold lines shows one of the minimum cycles in the graph.

if the weight of the rows and columns are equal, that is $w_{\mathsf{r},i} = w_r$ and $w_{\mathsf{c},i} = w_{\mathsf{c}}$, the code is said to be regular,. If the weight of all rows or columns are not equal, then the code is said to be irregular. The code rate of the regular and irregular LDPC code is

$$R_c^{\mathsf{reg}} = 1 - \frac{w_{\mathsf{c}}}{w_{\mathsf{r}}} \quad \text{and} \quad R_c^{\mathsf{irreg}} = 1 - \frac{\tilde{w}_{\mathsf{c}}}{\tilde{w}_{\mathsf{r}}} \tag{1.14}$$

respectively, where $\tilde{w}_{\mathsf{c}}$ and $\tilde{w}_{\mathsf{r}}$ are the average column and row weights of the irregular parity-check matrix.

Tanner introduced in his 1981 paper that a parity-check matrix can be represented by a bipartite graph, a *Tanner graph*, whose nodes are separated into two categories: *variable nodes* (VN:s) and *check nodes* (CN:s). The VN:s corresponds to the n columns of $\boldsymbol{H}$ and the code bits while the CN:s corresponds to the m rows of $\boldsymbol{H}$. The Tanner graph is then constructed by drawing an edge, or connecting, each VN i and CN j where the corresponding element of $\boldsymbol{H}$ has a one. By example, drawing the Tanner graph of the parity-check matrix in eq. (1.13) is shown in Figure 1.2.

Associated with the Tanner graph are a few properties. The *girth* of the graph is the length of the minimum cycle, which in the case of Figure 1.2 is 4, with an example highlighted by the bold edges. This is an important measure as the decoding algorithm can struggle to find the global optimum if it gets stuck in a cycle. Here the importance of a low-density graph comes in play. The higher the density, the more short cycles the graph will contain. The *degree* of a node is the number of branches departing from it. Thus, the VN v_i has degree $d_{\mathsf{v}} = \deg(v_i) = w_{\mathsf{c},i}$. Since all the branches departing from the VN conveys the same information, the VN:s can be seen as a repetition code. Moreover, the values of all the VN:s connected to the same CN must sum to zero under modulo 2 since they participate in a parity-check equation, following from eq. (1.12). The CN:s can therefore be seen as a single parity-check code (SPC).

As previously stated, the optimum decoding rule using MAP is

$$\hat{c_i} = \begin{cases} 1 & \text{if } P(c_i = 1|\boldsymbol{y}) > P(c_i = 0|\boldsymbol{y}) \\ 0 & \text{if } P(c_i = 1|\boldsymbol{y}) < P(c_i = 0|\boldsymbol{y}). \end{cases} . \tag{1.15}$$

By convenience this is usually written using the log-likelihood ratio (LLR), defined as

$$L(c_i|\boldsymbol{y}) \triangleq \ln \frac{P(c_i = 0|\boldsymbol{y})}{P(c_i = 1|\boldsymbol{y})}, \tag{1.16}$$

which outputs information about if c_i is more likely to be a 0 or 1. The optimum decoding rule in eq. (1.15) can then be rewritten as

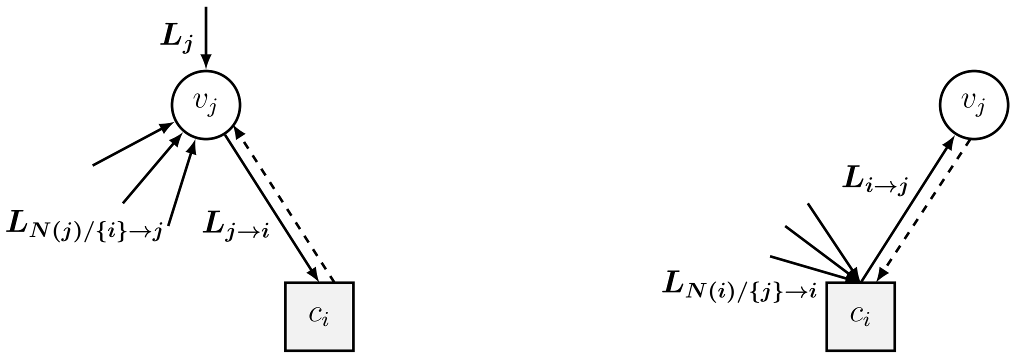
$$\hat{c_i} = \begin{cases} 1 & \text{if } L(c_i|\boldsymbol{y}) < 0 \\ 0 & \text{if } L(c_i|\boldsymbol{y}) > 0 \end{cases} . \tag{1.17}$$

Due to the usually long LDPC codes optimal decoding is however not feasible. Therefore, sub-optimal decoding algorithms may be used and is also what Gallager proposed in his 1963 paper. The algorithms proposed was based on the structure of the Tanner graph representation and characterised by the exchange of message between the nodes, known as message passing. By considering the interpretation of the LDPC code as a network of connected repetition and SPC codes the local decoding of each connected component can be done optimally using MAP decoding. Soft information in the form of LLR:s is then iterated between the nodes through message passing until convergence. As the messages are past between the nodes, the problem with the short cycles arise. In short, the length of the cycles in the graph affects the bit error rate (BER) by influencing the level of the error floor.

1.1.6 Belief propagation in LDPC decoding

The message passage algorithm that Gallager introduced is today referred to as *belief propagation*, *sum-product algorithm* or *turbo decoding*, which is also the principle in another coding scheme called *Turbo codes* [2]. The message passing is illustrated in Figure 1.3 as an exchange of LLR:s. In the figure the message from VN j to CN i is denoted by $L_{j\to i}$. Furthermore, all CN:s connected to to VN j is referred to its neighbourhood $N(j)$. Lastly, the information from the channel is denoted L_j.

As the message that is passed between the nodes are LLR:s the respective nodes must perform a MAP decoding before the message is passed, which will be different for VN:s and CN:s. As previously stated a VN can be seen as a repetition code whilst a CN is a SPC code. The MAP decoding for these two cases are therefore defined. In both cases a memoryless channel is assumed.


$\boldsymbol{L_j}$
v_j
$\boldsymbol{L_{N(j)/\{i\}\to j}}$
$\boldsymbol{L_{j\to i}}$
c_i
v_j
$\boldsymbol{L_{i\to j}}$
$\boldsymbol{L_{N(i)/\{j\}\to i}}$
c_i

(a) Message passing from VN to CN

(b) Message passing from CN to VN

Figure 1.3: Illustration of the message passing from VN:s (a) and CN (b). $N(l)/\{k\}$ denotes the neighbourhood of node l except node k.

1.1.6.1 MAP decoding for repetition code

Consider a repetition code with the binary code symbol $c \in \{0,1\}$ that is transmitted over the channel n times. By eq. (1.16), the LLR, considering the received vector $\boldsymbol{y}$, can be written as

$$L(c|\boldsymbol{y}) = \ln \frac{P(c=0|\boldsymbol{y})}{P(c=1|\boldsymbol{y})}. \tag{1.18}$$

Assuming equiprobable symbols and applying Bayes' law this is equal to

$$\begin{aligned} L(c|\boldsymbol{y}) &= \ln \frac{P(\boldsymbol{y}|c=0)}{P(\boldsymbol{y}|c=1)} \\ &= \ln \frac{\prod_{l=1}^{n} P(y_l|c=0)}{\prod_{l=1}^{n} P(y_l|c=1)} \\ &= \sum_{l=1}^{n} \ln \frac{P(y_l|c=0)}{P(y_l|c=1)} \\ &= \sum_{l=1}^{n} L(y_l|c). \end{aligned} \tag{1.19}$$

To compute the marginal, i.e the final prediction at VN j, all the incoming messages is summed

$$L_j^{\text{tot}} = L_j(c|y_j) + \sum_{l=1}^{n} L(y_l|c) \tag{1.20}$$

1.1.6.2 MAP decision for SPC codes

To derive the MAP decoder for a SPC code we first need to establish an important result from Gallager, namely the probability that a vector of d independent binary random variables $\boldsymbol{x} = [x_0,\ x_1,\ ...,\ x_{d-1}]$ contains an even number of 1's. Assume that $P(x_l = 1) = p_1^{(l)}$ and $P(x_l = 0) = p_0^{(l)}$, then the probability is given as

$$P(w_H = \text{"even"}) = \frac{1}{2} + \frac{1}{2} \prod_{l=0}^{d-1} \left(1 - 2p_1^{(l)}\right). \tag{1.21}$$

The probability for a 0 at a CN i, considering that the node is a SPC code, with the received vector $\boldsymbol{y}$ can be written as

$$P(c_i = 0|\boldsymbol{y}) = \frac{1}{2} + \frac{1}{2} \prod_{l \in N(i)} \left(1 - 2P(c_l = 1|y_l)\right). \tag{1.22}$$

By using that $P(c_i = 0|\boldsymbol{y}) = 1 - P(c_i = 1|\boldsymbol{y})$, eq. (1.22) can be rearranged as

$$1 - 2P(c_i = 1|\boldsymbol{y}) = \prod_{l \in N(i)} \left(1 - 2P(c_l = 1|y_l)\right). \tag{1.23}$$

It is desirable to express this result as an LLR. By using the equality derived in the appendix A.1 the following result can be acquired:

$$\tanh\left(\frac{1}{2}\left(\frac{P(c_i = 0|\boldsymbol{y}}{P(c_i = 1|\boldsymbol{y}}\right)\right) = \prod_{l \in N(i)} \tanh\left(\frac{1}{2}\left(\frac{P(c_l = 0|y_l}{P(c_l = 1|y_l}\right)\right) \tag{1.24}$$

or similarly

$$\tanh\left(\frac{1}{2}L(c_i|\boldsymbol{y})\right) = \prod_{l\in N(i)} \tanh\left(\frac{1}{2}L(c_l|y_l)\right). \tag{1.25}$$

Finally, by rearranging this result we get the LLR for CN i:

$$L(c_i|\boldsymbol{y}) = 2\tanh^{-1}\prod_{l\in N(i)} \tanh\left(\frac{1}{2}L(c_l|y_l)\right). \tag{1.26}$$

1.1.6.3 Message passing algorithm

As the sum-product algorithm run (see Alg. 1), the message from VN j to CN i is computed as the sum of all incoming messages, except from CN i, i.e

$$L_{j\to i} = L_j(c_j|y_j) + \sum_{i'\in N(j)/\{i\}} L_{i'\to j}. \tag{1.27}$$

Similarly, the message from CN i to VN j is computed as

$$L_{i\to j} = 2\tanh^{-1}\left(\prod_{j'\in N(i)/\{j\}} \tanh\left(\frac{1}{2}L_{j'\to i}\right)\right). \tag{1.28}$$

This information is known as the *extrinsic* information, where the total information is defined as the sum of the *intrinsic* and extrinsic information. Intuitively, information should not be sent to the same neighbour that the information previously was received from since as it would be redundant. Therefore, each VN and CN only passes the extrinsic (or new) information to each connected neighbour.

This algorithm provides exact LLR:s under the assumption of independent messages, which is true if the graph contains no cycles. If the graph contains cycles with girth γ the independence assumption is only valid up to the $\gamma/2$:th iteration after which messages starts to loop back in cycles.

1.2 Project Aim and Objectives

In the previous section the basic concepts of LDPC decoding was discussed. It was shown how the decoding of each bit relied on calculating LLR:s that then are fed into the message passing process between variable nodes and check nodes. This thesis will focus on the LLR-calculation and how it can be extended when the AWGN channel model is extended to also incorporate phase noise (PN). Introducing additional sources of noise to the channel further distorts the received signal and degrades performance. However, by utilising information about the distribution of all the noise sources it may be possible to improve the decoding scheme and increase the performance. From a theoretical standpoint it would be interesting to pursue the approach of joint decoding and phase estimation [7], where information about the phase noise is incorporated in the entire decoding scheme, including the message passing. Nevertheless, from an implementation point of view this is unpractical since large parts of existing implementations would need altering that potentially increase hardware complexity. Instead, it would be

Algorithm 1 Gallager's sum-product algorithm

```
Set: maxit
Initialise $L_{j \to i}$ with LLR:s from channel $L_j$
while (it < maxit) do
    for all check-nodes do                                        ▷ CN update
        Compute extrinsic info $L_{i \to j}$ to each VN in neighbourhood using eq. 1.28
    end for
    for all variable-nodes do                                     ▷ VN update
        Compute extrinsic info $L_{j \to i}$ to each CN in neighbourhood using eq. 1.27
    end for
    Compute final prediction $L_j^{\text{tot}}$ according to eq. (1.20)
    Calculate codeword $\hat{\boldsymbol{c}}$ according to eq. (1.17)
    if $\hat{\boldsymbol{c}}\boldsymbol{H}^\top = \emptyset$ then              ▷ stopping criteria
        Break
    end if
end while
Quantise converged LLR:s and map back to info bits $\hat{\boldsymbol{u}}$
```

interesting to develop a small standalone block that easily can be incorporated into existing solutions and that is able to compensate for the additional phase noise. This would enable extensions on current systems at low cost and complexity.

The purpose of this thesis project is to explore the state-of-the-art LDPC codes used for wireless back-haul communication systems and extend the current understanding of how information about phase noise can be used to improve the coding gain. The project aims to develop an extension of the conventional LLR calculation by considering phase noise in addition to AWGN. Furthermore to ensure practical relevance, the developed scheme will be adjusted for hardware implementation in an attempt to provide indications about potential gains and cost. The project is structured around the following objectives:

- How can information about phase noise be incorporated into the LLR calculation to improve the coding gain of LDPC codes?
- What approximations and adjustments must be considered to ensure practical relevance?
- What is the cost vs. gain relation of extending the conventional LLR scheme?

1.3 Project Scope

Real communication systems are complex and consist of a long series of subsequent operations to offer certain functionalities and to handle various sources of disturbance. In Figure 1.1 a simplified channel model is presented, which then can be extended to include a variety of additional sources of noise that are found in the real world. This thesis will be limited to a channel model composed solely of AWGN and phase noise. Furthermore, this thesis will focus on the coding aspect of a wireless communication systems. As a result, some parts of the communication system, such as e.g. modulation or equalisation, will not be covered in depth. There exists many different decoding schemes, however this thesis is limited to LDPC codes. When extending the conventional LLR-scheme, a standalone block is sought after and therefore further investigation on e.g. joint decoding schemes will be excluded. As mentioned in the previous section, this thesis focus on high-capacity wireless backhaul links, which operate with high data rates. Hence the scope will be limited to testing the developed scheme for larger QAM modulation formats such as 1024-, 4096-, and 16k-QAM.

2

Theory

This chapter will lay the theoretical foundation needed to later derive the extended LLR calculation. The chapter begins by establishing a method for constructing parity-check matrices through a protograph which is then followed by a review of a LDPC decoding algorithm used in practice. After that the phase noise channel is introduced together with the notion of residual phase noise, which forms the basis from which the extended LLR calculation scheme can be derived.

2.1 Protograph based codes

A LDPC code is defined as linear code whose parity matrix $\boldsymbol{H}$ of size $m \times n$ has the property that $w_{\mathsf{r}} \ll m$ and $w_{\mathsf{c}} \ll n$, i.e it should be a sparse matrix. Although only a cycle-free $\boldsymbol{H}$ can be decoded optimally, Gallager showed that for a binary symmetric channel the probability of a decoding error decreases exponentially with the block length [3]. Thus, the loss of sub-optimal decoding of LDPC codes is suppressed by the fast decoding enabled by the sparsity of the parity-check matrix. A convenient way of constructing a sparse parity-check matrix is through a protograph. A protograph based code is a code constructed from a smaller matrix, often called base matrix, by a combination of copy and permutation [8]. One typical type of protographs used in LDPC codes, which is also used in today's 5G systems, are quasi-cyclic codes. Quasi-cyclic codes are constructed by shifting an identity matrix with operations defined by the base matrix $\boldsymbol{P}$ and an expansion factor $\boldsymbol{Q}$. Consider the case of the base matrix

$$\boldsymbol{B} = \begin{bmatrix} 2 & 0 & 1 \\ 1 & -1 & 2 \end{bmatrix} \tag{2.1}$$

and expansion factor $\boldsymbol{Q} = 3$, which defines the size of the identity matrix. Each element in the base matrix then describes the right shift of each corresponding individual identity matrix. An element with -1 indicates an all-zero matrix. The full parity-check matrix expanded by $\boldsymbol{B}$ and $\boldsymbol{Q}$ will then be

$$\boldsymbol{H}_{\mathsf{qc}} = \begin{bmatrix} 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix}, \tag{2.2}$$

which is a combination of shifted identity matrices. In Figure 2.1 an even bigger protograph is visualised, with black pixels representing ones. The diagonal of ones in the right part of the graph enables fast encoding. Furthermore, by creating a parity-check matrix in a protograph manor, the decoding can be done efficiently based on the base matrix and utilising shift operations. This kind of decoding scheme is named *layered* decoding and will be discussed in detail in later sections.

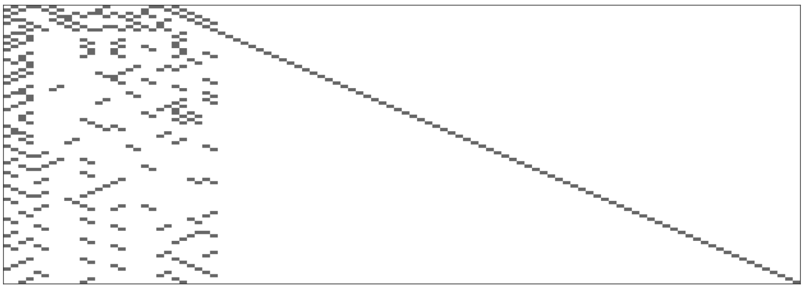


Figure 2.1: An example of protograph with a 42 × 52 base graph and expansion factor 2.

2.2 Decoding Techniques

In section 1.1.6 we previously touched on LDPC decoding, where Gallager's Sum-product algorithm was presented (see alg. 1) to illustrate the concept. However, when it comes to applications and realisation of LDPC decoders the algorithm have several implementation issues, which make room for further optimisation. One of these flaws is the computational complexity of the product and the tanh and $\tanh^{-1}$ operation in the CN update step in eq. 1.28. Next, the *Min-Sum* algorithm will be presented, which through approximation circumvents this issue.

2.2.1 Min-Sum approximation

In many applications, the Min-Sum decoding algorithm with various modifications is a common approximation of the Sum-Product algorithm that reduces computation complexity [9]. Remember from eq. 1.28 how the CN update was given. By the following definitions

$$\begin{aligned} L_{i\to j} &= \alpha_{ji} \cdot \beta_{ji}, \\ \alpha_{ji} &= \operatorname{sign}(L_{i\to j}), \\ \beta_{ji} &= |L_{i\to j}|, \end{aligned} \tag{2.3}$$

the CN update can be rewritten as

$$\tanh\left(\frac{1}{2}L_{j\to i}\right) = \prod_{j'\in N(i)/\{j\}} \alpha_{j'i} \cdot \prod_{j'\in N(i)/\{j\}} \tanh\left(\frac{1}{2}\beta_{j'i}\right). \tag{2.4}$$

It can be shown that by using eq. 2.4 the CN update $L_{i\to j}$ can be written as

$$L_{i\to j} = \prod_{j'\in N(i)/\{j\}} \alpha_{j'i} \cdot \phi\left(\sum_{j'\in N(i)/\{j\}} \phi(\beta_{j'i})\right), \tag{2.5}$$

where ϕ is

$$\phi(x) = -\ln[\tanh(x/2)] = \ln\left(\frac{e^x+1}{e^x-1}\right) \tag{2.6}$$

and where $\phi(x) = \phi^{-1}(x)$ when $x > 0$. It's clear that $\phi(x)$ is a exponentially decaying function which means that it will obtain it largest value for the smallest input argument $x > 0$. Due to the rapid decay of ϕ, one can assume that the largest term will dominate when summing over ϕ for a range of input arguments. Hence, looking at eq. 2.5, one can assume that the largest term in the sum, corresponding to the smallest β_{ji}, will dominate the other terms which enable the approximation

$$\phi\left(\sum_{j'\in N(i)/\{j\}} \phi(\beta_{j'i})\right) \simeq \phi\left(\phi\left(\min_{j'\in N(i)/\{j\}} \beta_{j'i}\right)\right) = \min_{j'\in N(j)/\{i\}} \beta_{j'i} \tag{2.7}$$

without much loss of accuracy. This finally yields the Min-Sum approximation where the CN update can be written as

$$L_{i\to j'} = \prod_{j'\in N(i)/\{j\}} \alpha_{j'i} \cdot \min_{j'\in N(i)/\{j\}} \beta_{j'i}. \tag{2.8}$$

Compared to eq. 1.28, this expression is less complex as it removes the tanh and $\tanh^{-1}$, which is costly to implement in hardware.

2.2.2 Scaled and offset Min-Sum

The approximation of the extrinsic information $L_{i\to j}$ passed from CN:s to VN in the a Min-Sum decoder in the first iteration have been shown to on average be larger in magnitude than the true extrinsic information that would have been passed in the Sum-Product decoder [2]. In other words, the Min-Sum decoder is too optimistic in the initial passing probabilities in the tanner graph. A natural approach to compensate this behaviour is therefore to attenuate the extrinsic information being sent from CN:s by a scale factor c_{att}. This yields an updated expression for the extrinsic information being passed from CN:s to VN:s

$$L_{i\to j} = c_{\text{att}} \cdot \prod_{j'\in N(i)/\{j\}} \alpha_{j'i} \cdot \min_{j\in N(i)/\{j\}} \beta_{j'i}, \tag{2.9}$$

where $0 \leq c_{\text{att}} \leq 1$. This modified decoder is named *Scaled Min-Sum.* Another approach for compensation of the overly optimistic extrinsic information is to subtract a constant offset c_{offset} from the message magnitude β_{ji}, while satisfying the condition that $\beta_{ji} - c_{\text{offset}} \geq 0$. This yields the *Offset Min-Sum* decoder, where the CN update is given by

$$L_{i\to j} = \prod_{j'\in N(i)/\{j\}} \alpha_{j'i} \cdot \max\left\{\left(\min_{j'\in N(i)/\{j\}} \beta_{j'i}\right) - c_{\text{offset}},\ 0\right\}. \tag{2.10}$$

2.2.3 Min-Sum decoding algorithm

In this chapter, three different type of Min-Sum decoders have been presented, namely: *standard Min-Sum*, *scaled Min-Sum*, and *offset Min-Sum*. When implemented they all follow the same algorithm, with the only difference being how they calculate the extrinsic information that is being passed from CN:s to VN:s. The algorithm for the Min-Sum decoder can be implemented according to Sum-Product algorithm, see Alg. 1, but with a different calculation of the extrinsic information $L_{i \to j}$. This way of decoding the channel output y is referred to as *flooding*.

When using flooding in the Min-Sum decoding, the algorithm starts by assigning the LLR:s L_j, based on the channel output, to their corresponding VN. In the second step, all VN:s marginalise by summing their L_j with all incoming messages over connected edges from CN:s. In the first iteration these messages are all zero. The extrinsic information is calculated for each neighbouring CN$_i$, by subtracting the previous message from CN$_i$ from the sum of all messages (see eq. 1.27), and then passed to all CN:s. In the third step, all CN:s receive extrinsic information from all neighbouring VN:s and computes the new extrinsic info, according to eq. 2.8, to be passed back to neighbouring VN:s. The algorithm then iterates until the LLR:s converge to a codeword. What distinguishes flooding is that all CN:s are updated simultaneously in each iteration, without single VN being updated in between.

An alternative decoding approach for LDPC codes is what is called *layering*, which utilises the protographic structure of the parity-check matrix when running the decoding. By applying layering, the decoding algorithm can converge to a decoding decision using fewer iterations between CN:s and VN:s. This increases the speed of the decoding. The main difference between the layering and flooding approach is that layering, in each iteration, updates the CN:s corresponding to each *layer* $r_p^{(l)}$ (or row of protograph) and their neighbouring VN separately. This means that first the Q CN:s corresponding to the first layer are updated and their extrinsic information is passed to their neighbouring VN:s, which then updates their LLR:s. Note that Q is the expansion factor of the protograph. Now that the LLR:s have been updated for the first layer, the algorithm continues and repeat the process for the remaining layers until all CN:s have updated. At this point, one iteration is complete, the layering algorithm then goes back to the first layer and continues to iterate until LLR:s converge to a code word. The algorithm is shown in detail in Alg. 2.

2.3 Phase noise channel

In the channel model introduced in section 1.1 the noise is modelled as AWGN. This is a valid assumption considering the scope of a high capacity wireless backhaul link that usually has a slow fading channel. However, in a physical setup there will be impairments in the hardware, namely that the oscillators are neither perfect nor in sync, which will cause phase noise. The high capacity systems in the wireless backhaul also rely on higher order constellations, which, as shown later, are more sensitive to phase noise. Modern systems apply methods such as phase tracking [10] for synchronisation between independent oscillators. Nevertheless, the synchronisation is seldom perfect and there often remains some residual error that will affect the received symbols. In this section, the modelling of phase noise for single-input single-output (SISO) and multiple-input multiple-output (MIMO) channels will be presented.

Algorithm 2 Layered Min-Sum algorithm

```
Set: maxit
Initialise $L_{j\to i}$ with LLR:s from channel $L_j$
while (it < maxit) do
    for all rows $r_p^{(l)}$ in protograph do
        for all check-nodes corresponding to the columns in $r_p^{(l)}$ do      ▷ CN update
            Compute extrinsic info $L_{i\to j}$ to each VN in neighbourhood using eq. 1.28
        end for
        for all variable-nodes corresponding to the rows $r_p^{(l)}$ do         ▷ VN update
            Compute extrinsic info $L_{j\to i}$ to each CN in neighbourhood using eq. 1.27
        end for
    end for
    Compute final prediction $L_j^{\text{tot}}$ according to eq. (1.20)
    Calculate codeword $\hat{\boldsymbol{c}}$ according to eq. (1.17)
    if $\hat{\boldsymbol{c}}\boldsymbol{H}^\top = \emptyset$ then                 ▷ stopping criteria
        Break
    end if
end while
Quantise converged LLR:s and map back to info bits $\hat{\boldsymbol{u}}$
```

2.3.1 SISO system

In a SISO channel the system consists of a single transmitter transmitting to a single receiver. Phase noise is often modelled as a multiplicative noise, which yields the phase noise SISO channel model

$$y_i = x_i\, e^{j\phi_i} + n_i, \tag{2.11}$$

where ϕ_i is the phase noise, n_i the Gaussian noise and i the index. The received symbol $\hat{x}_i$ can then simply be estimated as

$$\hat{x}_i = y_i\, e^{-j\hat{\phi}_i}, \tag{2.12}$$

where $\hat{\phi}_i$ is the estimated phase noise. Since we have oscillators at both the transmitter and receiver, the phase noise can be seen as a combination of the contribution from the receiver and transmitter antennas, denoted by superscripts t and r, as

$$\phi_i = \phi_i^t + \phi_i^r. \tag{2.13}$$

A common approach is to model the transmitter and receiver radios' phase noise as a Wiener process [11], or random walk, as

$$\begin{aligned} \phi_{i+1}^t &= \phi_i^t + \Delta_i^t \\ \phi_{i+1}^r &= \phi_i^r + \Delta_i^r \end{aligned} \tag{2.14}$$

with $\Delta^t \in \mathcal{N}(0, \sigma_t^2)$ and $\Delta^r \in \mathcal{N}(0, \sigma_r^2)$. The phase noise variances σ_t^2 and σ_r^2 is usually small and can be calculated by using oscillator measurements [12]. Considering that both phase noise variances Δ^t and Δ^r are i.i.d and using eq. (2.13), we can write

$$\phi_{i+1} = \phi_i + \Delta_i, \tag{2.15}$$

where $\Delta \in \mathcal{N}(0, \sigma_L^2)$ and $\sigma_L^2 = \sigma_t^2 + \sigma_r^2$. Under these assumptions it follows that

$$p(\phi_i|\,\phi_i, \phi_{i-1}, ..., \phi_0) = p(\phi_{i+1}|\phi_i), \tag{2.16}$$

which shows that the phase noise is a Markov process.

2.3.2 MIMO system

Compared to the SISO case, a MIMO system operates several transmitter and receiver antennas and may be equipped with separate radios with independent oscillators for each antenna, a setup referred to as a *non-synchronous* operation. On the other hand, the *synchronous* setup occurs when all antennas, in transmitter and receiver respectively, are connected to the same oscillator [12]. In this thesis the case of non-synchronous operation will be considered.

Consider a 2 × 2 MIMO as pictured in Figure 2.2 with each transmitter antenna transmitting a different symbol and the antennas placed at a distance so that the received signals have a phase offset of $-\pi/2$ rad. The received signal, for some time index, can then be written as

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} e^{j\phi_{r_1}} & 0 \\ 0 & e^{j\phi_{r_2}} \end{bmatrix} \begin{bmatrix} 1 & -j \\ -j & 1 \end{bmatrix} \begin{bmatrix} e^{j\phi_{t_1}} & 0 \\ 0 & e^{j\phi_{t_2}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}, \tag{2.17}$$

where $\phi_{r_1}, \phi_{r_2}, \phi_{t_1}$ and ϕ_{t_2} are the independent phase noise processes as in eq. (2.15), and n_1 and n_2 are independent Gaussian noise at the receiver and transmitter respectively [13]. As in the SISO case, each channel can be seen as a combination of the phase noise processes from the transmitter and receiver, allowing for the mapping $\boldsymbol{y} = \boldsymbol{P}\boldsymbol{x} + \boldsymbol{n}$, where

$$\boldsymbol{P} = \begin{bmatrix} e^{j(\phi_{t_1}+\phi_{r_1})} & -je^{j(\phi_{t_2}+\phi_{r_1})} \\ -je^{j(\phi_{t_1}+\phi_{r_2})} & e^{j(\phi_{t_2}+\phi_{r_2})} \end{bmatrix} = \begin{bmatrix} e^{j\phi_{11}} & -je^{j\phi_{21}} \\ -je^{j\phi_{12}} & e^{j\phi_{22}} \end{bmatrix} \tag{2.18}$$

and each ϕ_{tr}, with subscript t and r denoting the respective receiver and transmitter, has a phase noise rate σ_L. The received signal at each antenna will then be a superposition of signals from transmitter antenna 1 and 2, where the antenna 1 will have a phase shift of $-\pi$ due to the longer path it has to travel. The received symbols can then be estimated by computing

$$\hat{\boldsymbol{x}} = (\hat{\boldsymbol{P}}\hat{\boldsymbol{P}}^{\mathsf{H}})^{-1}\hat{\boldsymbol{P}}^{\mathsf{H}}\boldsymbol{y} \tag{2.19}$$

where H denotes the Hermitian transpose and $\hat{\boldsymbol{P}}$ is the channel with the estimated phase noise.

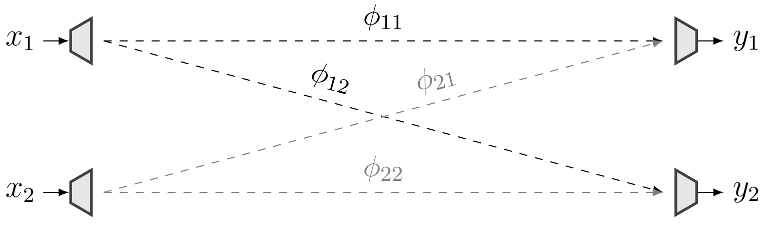


Figure 2.2: The 2 × 2 MIMO system considered.

2.4 Residual phase noise

The main topic of this report is not to find methods of phase noise estimation, but rather to investigate the LDPC decoder under the influence of residual phase noise (RPN), e.g. the small error that follows from the imperfect phase estimation. Especially, the case when the phase noise has been estimated by P-periodic pilot symbols will be considered. The known phase at each pilot, ϕ_i^P, will then be used to estimate the phase at the symbols in between through interpolation. The interpolation techniques considered in this report will be strictly linear, in particular zero- and first-order-hold (ZOH and FOH respectively) methods will be applied. The two linear methods result in the following expressions for the estimated phase in the phase tracking

$$\hat{\phi}_{i+m} = \begin{cases} \phi_i^P & \text{if } m \leq P/2 \\ \phi_{i+P}^P & \text{if } m > P/2, \end{cases} \tag{2.20}$$

$$\hat{\phi}_{i+m} = \frac{\phi_{i+P}^P - \phi_i^P}{P} m + \phi_i^P \tag{2.21}$$

where m is the distance from the previous pilot and eq. (2.20) correspond to ZOH and eq. (2.21) to FOH. From the previous section, we remember the expression in eq. (2.15) for the added phase noise ϕ over a SISO channel. From this expression, the RPN θ from the phase tracking, using either ZOH or FOH, can be modelled as the difference between eq. (2.21) and (2.15), namely

$$\theta_{i+m} = \phi_{i+m} - \hat{\phi}_{i+m}. \tag{2.22}$$

The RPN is a cyclo-stationary random process and can therefore be approximated as a Gaussian random process [14]. The phase noise ϕ_i^P at any pilot can be expressed as a sum of two independent sources of noise, one caused by AWGN, ϕ^W, and the other by the Wiener process, ϕ^{PN}, as in eq. (2.15). It can be shown that both ϕ^W and ϕ^{PN} are in fact zero-mean Gaussian random variables, thus the RPN θ is a linear combination of zero-mean Gaussian random variables, which itself is a zero-mean Gaussian random variable with non-constant variance σ_θ^2. The variance σ_θ^2 for a given time-sample is dependent on the distance in time to the nearest pilot and is symmetric around the centre of the pilot section, i.e., symmetric around the data sample at equal distance from two subsequent pilots [14]. In addition, depending on the interpolation method of choice, e.g. ZOH or FOH, the time-varying variance of RPN σ_θ^2 will have different characteristics. In Appendix A.2.1 it is shown that for ZOH σ_θ^2 can be expressed as

$$\sigma_\theta^2(m) = \begin{cases} \sigma_n^2 + \sigma_L^2 \cdot m & \text{if } m \leq P/2 \\ \sigma_n^2 + \sigma_L^2 \cdot (P - m) & \text{if } m > P/2, \end{cases} \tag{2.23}$$

which shows that the variance is linear function of the position m in the pilot section, where $m \in (1, ..., P)$, and σ^2 is the variance of the AWGN noise. Similarly for FOH, as shown in Appendix A.2.2, σ_θ^2 can be expressed as

$$\sigma_\theta^2(m) = m\left(1 - \frac{m}{P}\right)\sigma_L^2 + \left(1 - 2\frac{m}{P} + 2\frac{m^2}{P^2}\right)\sigma_n^2. \tag{2.24}$$

These insights about the variance of RPN will later prove to be useful in the derivation of the PN-LLR scheme.

3

Methods

This chapter details the main procedures and techniques used, and serves as a blueprint for the steps that was conducted over the course of the project. The chapter begins by presenting the system model for various channels, followed by a derivation of the conventional LLR calculation for higher order modulation schemes. Thereafter, the chapter extends the conventional LLR calculation to include information on phase noise and provides full derivations for different channels. Finally, the chapter discusses adjustments for hardware implementation and methods for hardware synthesis to yield information about cost-benefit trade-offs.

3.1 System Model

The targeted model in this thesis is that of high-capacity line-of-sight wireless backhaul links, which are usually defined by a quasi-static and frequency flat channel [15]. Furthermore, since the purpose of this thesis is coding, the channel is assumed to be known to the receiver and that perfect equalisation and timing synchronisation is performed. For the analysis, both a SIMO and a MIMO system has been considered and the block diagram of the system is shown in Figure 3.1.

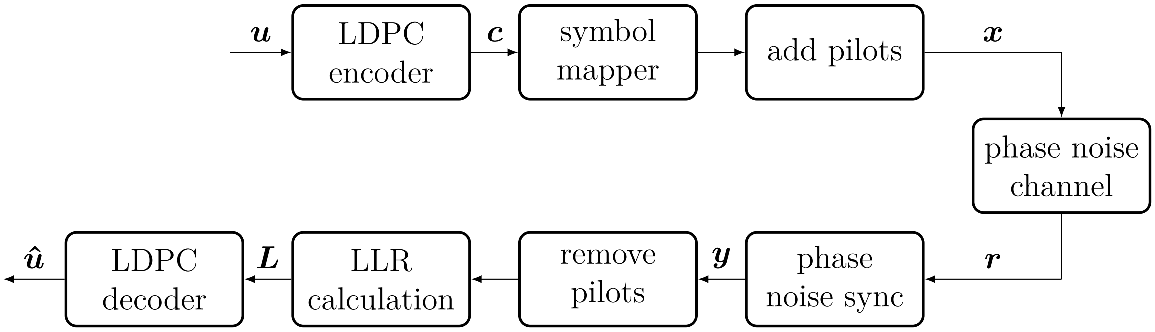


Figure 3.1: System model.

First, random information bits $\boldsymbol{u}$ are generated and sent into the LDPC encoder. The encoded bits are then mapped into symbols, framed with pilots and sent through the phase noise channel that adds AWGN and a multiplicative phase noise in eq. (2.11) and (2.17) for SISO and MIMO respectively. The noisy samples received in the vector $\boldsymbol{r}$ is passed into the phase synchronisation block, which estimates and corrects for the phase rotation. Within the block the samples of the estimated phase is passed through a Wiener filter to reduce the impact of AWGN noise.

In Figure 3.2 the resulting effect of the RPN on the received symbols $\boldsymbol{y}$ is shown for a 64-QAM constellation without any AWGN noise. The RPN creates a slight rotation across the constellation point that is more apparent as the radius increases. After the synchronisation the pilots are removed and the LLR:s $\boldsymbol{L}$ are calculated and sent to the LDPC decoder that estimates the received bit $\hat{\boldsymbol{u}}$.

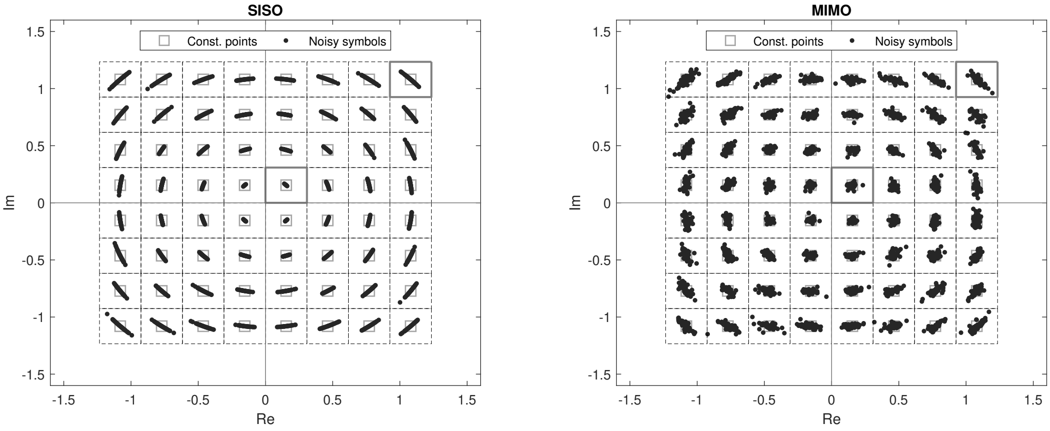


Figure 3.2: Constellation diagram for 64-QAM with RPN. The dotted lines indicate the decision regions. Symbols further away from the centre has higher amplitude, hence is rotating closer to decision borders.

3.1.1 SISO

To calculate the added RPN, a pilot based estimation was implemented with a pilot placed at regular distances in the data. For SISO, one single pilot is used in each frame to estimate the phase noise. To reduce noise, the samples of the phase noise estimate is passed through a Wiener filter. The samples between the pilots are then interpolated by either ZOH or FOH interpolation and finally removed from the signal. After estimating the phase noise the received signal is

$$y_i = x_i \, e^{j\theta_i} + n_i, \tag{3.1}$$

where $\theta_i = \phi_i - \hat{\phi}_i$ is the RPN.

3.1.2 MIMO

For MIMO a 2 × 2 system is considered, as depicted in Figure 2.2, resulting in the received signal

$$\begin{bmatrix} r_1 \\ r_2 \end{bmatrix} = \begin{bmatrix} e^{j\phi_{11}} & -je^{j\phi_{21}} \\ -je^{j\phi_{12}} & e^{j\phi_{22}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}, \tag{3.2}$$

that follows from eq. (2.18). As the received signal has four independent unknown phase noise processes, a phase estimation scheme utilising two consecutive pilots in each frame was applied for both transmitters [16], as illustrated in Figure 3.3. This simple but effective scheme starts with sending a pilot at one antenna while the other one is silent. In the second pilot the other antenna transmits while the first one is silent. By this method the phase noise at the pilots can be properly estimated. As for SISO the phase estimations are then passed through a Wiener filter and interpolated. After estimating the phase noise from the pilots the 2 × 2 combined phase noise channel matrix $\hat{\boldsymbol{P}}$ is constructed. As shown in Appendix A.3, performing $\boldsymbol{y} = \hat{\boldsymbol{P}}^{-1}\boldsymbol{r}$ yields the final expression of the received phase compensated signal as

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \frac{x_1(e^{j(\hat{\phi}_{11}+\hat{\phi}_{22}+\theta_{11})}+e^{j(\hat{\phi}_{12}+\hat{\phi}_{21}+\theta_{12})})+jx_2(e^{j(\hat{\phi}_{22}+\hat{\phi}_{21}+\theta_{22})}-e^{j(\hat{\phi}_{21}+\hat{\phi}_{22}+\theta_{21})})}{e^{j(\hat{\phi}_{11}+\hat{\phi}_{22})}+e^{j(\hat{\phi}_{21}+\hat{\phi}_{12})}} \\ \frac{jx_1(e^{j(\hat{\phi}_{11}+\hat{\phi}_{12}+\theta_{11})}-e^{j(\hat{\phi}_{12}+\hat{\phi}_{11}+\theta_{12})})+x_2(e^{j(\hat{\phi}_{22}+\hat{\phi}_{11}+\theta_{22})}+e^{j(\hat{\phi}_{21}+\hat{\phi}_{12}+\theta_{21})}}{e^{j(\hat{\phi}_{11}+\hat{\phi}_{22})}+e^{j(\hat{\phi}_{21}+\hat{\phi}_{12})}} \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}, \tag{3.3}$$

where θ_{tr} is the RPN for the channel between transmitter t and receiver r. In comparison with the SISO case, the MIMO system introduces a leakage from the neighbouring antenna due to imperfect phase estimation that is the RPN. As can be seen in eq. (3.3), setting $\theta_{tr} = 0$ results in the AWGN case $\boldsymbol{y} = \boldsymbol{x} + \boldsymbol{n}$.

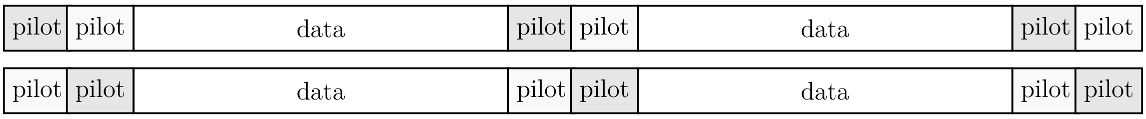


Figure 3.3: The double pilot scheme used in the MIMO channel. The darker grey pilot symbols indicate when the transmitter is active and the lighter grey indicate when the transmitter is silent.

3.2 LLR calculation schemes

In the introduction to this report, the concept of LLR:s was briefly introduced for the simple case of BPSK over an AWGN channel. However, when designing a real communication system today the requirements of higher order modulations has surpassed BPSK to achieve higher data rates. Furthermore, due to the nature of LDPC decoding algorithms, where LLR:s are used to pass extrinsic information, finding methods of calculating LLR:s accurately is an important part for improving decoding performance. As a start, the LLR calculations for the AWGN channel from the introduction will in this section be extended for higher order modulations of type M-QAM. This will then be followed by a further extensions where RPN will be considered in the LLR calculations, both for SISO and MIMO channels.

3.2.1 LLR for M-QAM constellations over the AWGN channel

The LLR for the i:th code bit c_i, $i = 1, ..., \log_2(M)$, for the received symbol y is given as

$$L(c_i|y) \triangleq \ln \frac{P(c_i = 0|y)}{P(c_i = 1|y)}. \tag{3.4}$$

The LLR outputs information about if c_i is more likely to be a 0 or 1. In the case of BPSK and AWGN, this expression simplifies to

$$L(c_i|y) = \frac{2y}{\sigma_n^2}. \tag{3.5}$$

However, when the modulation order M is increased, e.g. $M \geq 4$, the expression in eq. (3.5) does no longer hold since each symbol is mapping more than 1 code bits. Now, a more general expression is needed which can be derived as follows. By assuming equiprobable symbols and applying Bayes theorem the LLR of eq. (3.4) can be formulated as

$$L(c_i|y) = \ln \left(\frac{\sum\limits_{x \in \mathcal{X}_0} P(y|x)}{\sum\limits_{x \in \mathcal{X}_1} P(y|x)} \right), \tag{3.6}$$

where $\mathcal{X}_{0,i}$ and $\mathcal{X}_{1,i}$ denotes the two sets of constellation points where the i:th bit c_i is 0 respectively 1. For a M-QAM it is easy to show that $\mathcal{X}_{0,i}$ and $\mathcal{X}_{1,i}$ are equal-sized if a conventional mapping is used [14]. For the AWGN channel the likelihood function $P(Y|X)$ in eq. (3.6) is given as

$$P(y|x) = \frac{1}{2\pi\sigma_n^2} \exp\left(-\frac{|y-x|^2}{2\sigma_n^2}\right). \tag{3.7}$$

Combining eq. (3.6) and (3.7) yields

$$L(c_i|y) = \ln\left(\frac{\sum\limits_{x\in\mathcal{X}_{0,i}} \frac{1}{2\pi\sigma_n^2}\exp\left(-\frac{|y-x|^2}{2\sigma_n^2}\right)}{\sum\limits_{x\in\mathcal{X}_{0,i}} \frac{1}{2\pi\sigma_n^2}\exp\left(-\frac{|y-x|^2}{2\sigma_n^2}\right)}\right). \tag{3.8}$$

It is well-known in communication theory that the term in the sums that correspond to the closest constellation point will dominate the other terms, thus only considering the closest symbol in $\mathcal{X}_{0,i}$ and $\mathcal{X}_{1,i}$ is a solid approximation, i.e.

$$L(c_i|y) \approx \ln\left(\frac{\max\limits_{x\in\mathcal{X}_{0,i}}\left\{\frac{1}{2\pi\sigma_n^2}\exp\left(-\frac{|y-x|^2}{2\sigma_n^2}\right)\right\}}{\max\limits_{x\in\mathcal{X}_{0,i}}\left\{\frac{1}{2\pi\sigma_n^2}\exp\left(-\frac{|y-x|^2}{2\sigma_n^2}\right)\right\}}\right). \tag{3.9}$$

By further simplification the final expression of the LLR for a M-QAM constellation is

$$L(c_i|y) \approx \frac{1}{2\sigma_n^2}\left\{\min_{x\in\mathcal{X}_{0,i}}|y-x|^2 - \min_{x\in\mathcal{X}_{1,i}}|y-x|^2\right\}. \tag{3.10}$$

3.2.2 LLR for RPN over SISO channel

The LLR calculations for higher order modulations can be extend by also considering information about phase noise over the channel. For the LLR calculation in the presence of RPN over a SISO channel the approach of [14] is followed. By using the well-known inequality $e^x \geq 1+x$, which follows from a first-order Taylor expansion, eq. (3.1) can be written as

$$y \approx x(1+j\theta) + n \tag{3.11}$$

where the indices have been dropped for simplicity. This is a valid approximation since the RPN θ is quite small. Furthermore, θ is defined as in eq. (2.23) or (2.24) depending on the choice of ZOH or FOH interpolation. As θ and n are i.i.d zero-mean Gaussian random variables with variance σ_θ and σ_n, the real and imaginary components of y, denoted y_i and y_q as the in-phase and quadrature components respectively, are Gaussian:

$$y_i = \Re\{y\} = x_i - x_q\theta + n_i \quad \sim \mathcal{N}(x_i, x_q^2\sigma_\theta^2 + \sigma_n^2) \tag{3.12}$$

$$y_q = \Im\{y\} = x_q + x_i\theta + n_q \quad \sim \mathcal{N}(x_q, x_i^2\sigma_\theta^2 + \sigma_n^2) \tag{3.13}$$

Similar to the case for the AWGN channel the LLR can be expressed as in eq. (3.6) but now with the likelihood function in eq. (3.7) as a bi-variate normal distribution dependent on both the white noise and the RPN. The new likelihood function can therefore be expressed as

$$P(y|x) = \frac{1}{2\pi\sqrt{\det(\Sigma)}}\exp\left(-\frac{1}{2}(y-x)^T\Sigma^{-1}(y-x)\right), \tag{3.14}$$

where Σ is the covariance matrix defined as

$$\Sigma = \begin{bmatrix} x_q^2\sigma_\theta^2 + \sigma_n^2 & -x_i x_q \sigma_\theta^2 \\ -x_i x_q \sigma_\theta^2 & x_i^2\sigma_\theta^2 + \sigma_n^2 \end{bmatrix} = \begin{bmatrix} \sigma_i^2 & \sigma_{cov}^2 \\ \sigma_{cov}^2 & \sigma_q^2 \end{bmatrix} \tag{3.15}$$

with determinant $\det(\Sigma) = [\sigma_i^2\sigma_q^2 - \sigma_{cov}^4]$. By inserting eq. (3.15) into eq. (3.14) the likelihood function can be rewritten as

$$P(y_i, y_q|x) = \frac{1}{2\pi\sqrt{\det(\Sigma)}} \cdot \exp(\Psi(x)), \tag{3.16}$$

where

$$\Psi(x) = -\frac{1}{2\det(\Sigma)} \left(\sigma_q^2(y_i - x_i)^2 - 2\sigma_{cov}^2(y_i - x_i)(y_q - x_q) + \sigma_i^2(y_q - x_q)^2\right). \tag{3.17}$$

By inspection of the likelihood function the only added part compared to the AWGN case is the consideration of the effect of the RPN. Consequently, when $\sigma_\theta^2 = 0$ eq. (3.16) is equal to the original function in eq. (3.7). By substituting the new likelihood function into eq. (3.6) the final expression for the PN-LLR can be expressed as

$$L(c_i|y) \approx \left[\max_{x\in\mathcal{X}_{0,i}} \left\{\Psi(x) + \ln\sqrt{\det(\Sigma)}\right\}\right] - \left[\max_{x\in\mathcal{X}_{1,i}} \left\{\Psi(x) + \ln\sqrt{\det(\Sigma)}\right\}\right]. \tag{3.18}$$

3.2.3 LLR for RPN over MIMO channel

As described in the system model in section 3.1, introducing RPN in the MIMO system adds a leakage term from the neighbouring antenna. Without loss of generality this calculation will for simplicity only be shown for one channel. Starting from equation eq. (3.3), the y_1 channel can be written as

$$\begin{aligned} y_1 &= \frac{x_1\big(e^{j(\hat\phi_{11}+\hat\phi_{22}+\theta_{11})} + e^{j(\hat\phi_{12}+\hat\phi_{21}+\theta_{12})}\big) + jx_2\big(e^{j(\hat\phi_{22}+\hat\phi_{21}+\theta_{22})} - e^{j(\hat\phi_{21}+\hat\phi_{22}+\theta_{21})}\big)}{e^{j(\hat\phi_{11}+\hat\phi_{22})} + e^{j(\hat\phi_{21}+\hat\phi_{12})}} + n_1 \\ &= \frac{x_1\big(e^{j\theta_{11}}e^{j(\hat\phi_{11}+\hat\phi_{22})} + e^{j\theta_{12}}e^{j(\hat\phi_{12}+\hat\phi_{21})}\big) + jx_2\big(e^{j\theta_{22}}e^{j(\hat\phi_{22}+\hat\phi_{21})} - e^{j\theta_{21}}e^{j(\hat\phi_{21}+\hat\phi_{22})}\big)}{e^{j(\hat\phi_{11}+\hat\phi_{22})} + e^{j(\hat\phi_{21}+\hat\phi_{12})}} + n_1. \end{aligned}$$

The derivation of the RPN-LLR for MIMO will take the same approach as for SISO by utilising the inequality $e^x \geq 1 + x$ for the exponents including the θ_{tr} terms. Accordingly, eq. (3.3) can be approximated as

$$\begin{aligned} &\approx \frac{x_1\left((1+j\theta_{11})e^{j(\hat\phi_{11}+\hat\phi_{22})} + (1+j\theta_{12})e^{j(\hat\phi_{12}+\hat\phi_{21})}\right)}{e^{j(\hat\phi_{11}+\hat\phi_{22})} + e^{j(\hat\phi_{21}+\hat\phi_{12})}} \\ &\quad + \frac{jx_2\left((1+j\theta_{22})e^{j(\hat\phi_{21}+\hat\phi_{22})} - (1+j\theta_{21})e^{j(\hat\phi_{21}+\hat\phi_{22})}\right)}{e^{j(\hat\phi_{11}+\hat\phi_{22})} + e^{j(\hat\phi_{21}+\hat\phi_{12})}} + n_1 \\ &= x_1 + \frac{jx_1(\theta_{11}e^{j(\hat\phi_{11}+\hat\phi_{22})} + \theta_{12}e^{j(\hat\phi_{12}+\hat\phi_{21})}) + x_2(\theta_{21}e^{j(\hat\phi_{21}+\hat\phi_{22})} - \theta_{22}e^{j(\hat\phi_{21}+\hat\phi_{22})})}{e^{j(\hat\phi_{11}+\hat\phi_{22})} + e^{j(\hat\phi_{21}+\hat\phi_{12})}} + n_1 \\ &= x_1 + \frac{jx_1(\theta_{11}e^{j(\hat\phi_{11}+\hat\phi_{22})} + \theta_{12}e^{j(\hat\phi_{12}+\hat\phi_{21})}) + x_2e^{j(\hat\phi_{21}+\hat\phi_{22})}(\theta_{21} - \theta_{22})}{e^{j(\hat\phi_{11}+\hat\phi_{22})} + e^{j(\hat\phi_{21}+\hat\phi_{12})}} + n_1. \end{aligned} \tag{3.19}$$

Using this result to solve the distributions for the real and imaginary part as in SISO is undesirable since the estimated phases is not accessible in the LLR block in the targeted solution. A rough approximation is assuming that the exponential terms is equal to one as the phases are Wiener processes with zero mean. This yields a more tractable expression of

$$y_1 = x_1 + \frac{jx_1(\theta_{11}+\theta_{12})}{2} + \frac{x_2(\theta_{21}-\theta_{22})}{2} + n_1. \tag{3.20}$$

Now the real and imaginary parts of y_1 can be expressed as

$$y_{1,i} = x_{1,i} - \frac{1}{2}\left(x_{1,q}(\theta_{11}+\theta_{12}) + x_{2,i}((\theta_{21}-\theta_{22})\right) + n_{1,i} \tag{3.21}$$

$$y_{1,q} = x_{1,q} + \frac{1}{2}\left(x_{1,i}(\theta_{11}+\theta_{12}) + x_{2,q}(\theta_{21}-\theta_{22})\right) + n_{1,q}. \tag{3.22}$$

In the above term, all the θ_{tr}:s are assumed to be i.i.d random variables distributed as $\theta_{tr} \sim \mathcal{N}(0,\sigma_\theta^2)$ where σ_θ^2 is defined in eq. (2.23) or eq. (2.24). By using the result that the variance of the linear combination of N i.i.d random variables $\{X_1,...,X_N\}$ is the sum of their variances, e.g. $\mathrm{Var}(\sum_{i=1}^N a_i^2 X_i) = \sum_{i=1}^N a_i^2 \mathrm{Var}(X_i)$, where a_i is a constant, their respective distribution of $y_{1,i}$ and $y_{1,q}$ can be expressed as

$$y_{1,i} \sim \mathcal{N}\left(x_{1,i}, \frac{1}{2}\left(x_{1,q}^2\sigma_\theta^2 + x_{2,i}^2\sigma_\theta^2 + \sigma_n^2\right)\right) \tag{3.23}$$

$$y_{1,q} \sim \mathcal{N}\left(x_{1,q}, \frac{1}{2}\left(x_{1,i}^2\sigma_\theta^2 + x_{2,q}^2\sigma_\theta^2 + \sigma_n^2\right)\right). \tag{3.24}$$

Computing the covariance matrix gives

$$\Sigma = \frac{1}{2}\begin{bmatrix} x_{1,q}^2\sigma_\theta^2 + x_{2,i}^2\sigma_\theta^2 + \sigma_n^2 & \sigma_\theta^2(-x_{1,i}x_{1,q} + x_{2,i}x_{2,q}) \\ \sigma_\theta^2(-x_{1,i}x_{1,q} + x_{2,i}x_{2,q}) & x_{1,i}^2\sigma_\theta^2 + x_{2,q}^2\sigma_\theta^2 + \sigma_n^2 \end{bmatrix} = \begin{bmatrix} \sigma_i^2 & \sigma_{cov}^2 \\ \sigma_{cov}^2 & \sigma_q^2 \end{bmatrix}, \tag{3.25}$$

with the determinant $\det(\Sigma) = [\sigma_i^2\sigma_q^2 - \sigma_{cov}^4]$. Again inserting eq. (3.15) into eq. (3.14) the likelihood function can be rewritten as

$$P(y_{1_i}, y_{1_q}|x) = \frac{1}{2\pi\sqrt{\det(\Sigma)}} \cdot \exp(\Psi(x_1)), \tag{3.26}$$

where

$$\Psi(x_1) = -\frac{1}{2\det(\Sigma)}\left(\sigma_q^2(y_{1,i}-x_{1,i})^2 - 2\sigma_{cov}^2(y_{1,i}-x_{1,i})(y_{1,q}-x_{1,q}) + \sigma_i^2(y_{1,q}-x_{1,q})^2\right). \tag{3.27}$$

The final LLR expression is obtained by inserting the likelihood expression in eq. (3.26) into (3.18). To calculate the LLR for x_2, the values of x_1 and x_2 are simply swapped. Thus, the LLR for MIMO is the same as for SIMO but with a different covariance matrix, which takes into account the leakage from the other antenna.

3.3 Hardware implementation

Adding PN into the equation of LLR introduces some extra calculations rather than just considering the minimum distance, which is the case for conventional AWGN-LLR. In theory, this is not a problem if it improves the performance. However, to be useful in a practical system the coding system should be able to be realised in hardware, which raises the question of cost versus benefit. In a real system there may also be limitations in memory and computation time that requires the LLR values to be limited to a certain amount of bits, hence degrading the resolution. A comparison will therefore be made between using the double-precision floating point format of 64 bits and limiting the LLR to a fixed-point value with six bits. The six bits consist of one sign-bit, three integer-bits, and two decimal-bits.

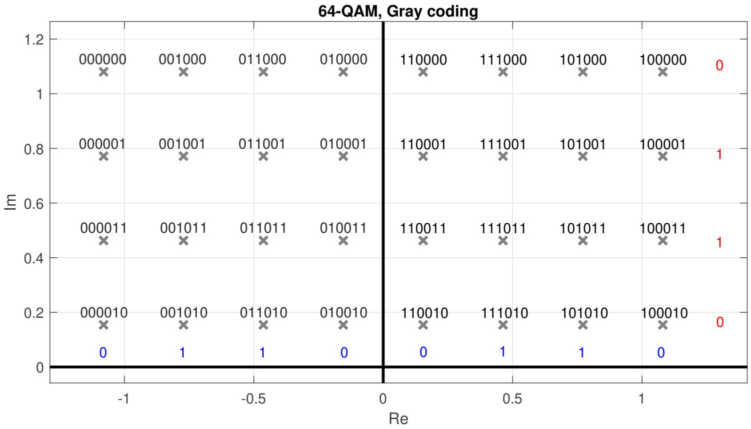


Figure 3.4: An example of a Gray coding for a 64-QAM constellation displayed in two quadrants.

Depending on how many symbols are mapped into bits, different symbols may be affected differently by noise. A common approach is to use Gray coding to ensure that neighbouring symbols only differs by one bit. In this context, the neighbourhood consists of the symbols at minimum distance from the symbol of interest. In Figure 3.4 a Gray coded 64-QAM constellation's upper quadrants are shown. As can be noticed, only the left half of the bits change along the real axis. Likewise, the right half of the bits only change along the imaginary axis. Within these two subsets the rightmost bits, that can be referred to as the least significant bits (LSB) of the real and imaginary part respectively, change every second bit, illustrated in Figure 3.4 by the numbers in blue and red. The other two bits of each respective part changes every fourth bit. Arguably, the LSB:s of the real and imaginary part of the symbols is affected by noise to a greater extent than the other more significant bits since they change at a higher frequency.

It could be advantageous in terms of redundancy to only code these bits. To still be able to correct errors in the other bits, another coding scheme with higher code rate could be applied on that subset, overall increasing the code rate. This concept is known as multilevel coding [17], where different coding schemes is adopted on different bits. Following that the least significant bits is most affected by the phase noise, the PN extension should make the largest difference for those bits as the more significant bits have larger decision regions. Therefore, challenges concerning scaling of LLR:s arise when coding all bits and quantising the output. Henceforth, the hardware implementation assume that only the least significant bits of each dimension are coded. This will facilitate the quantisation and improve the accuracy since all LLR:s will be within the same range as a result of having equally sized decision regions.

3.4 Hardware synthesis

To realise the PN-LLR calculation in hardware, the Vivado High-Level Synthesis (HLS) tool provided by Xilinx [18] was used. High-level synthesis takes an abstract structure, in this case a high-level programming language such as C++, and realises it in the register-transfer level (RTL), e.g. as hardware registers and logical operators. The resulting RTL can then be synthesised to a gate-level netlist, which then can be used in a field-programmable gate array (FPGA). An FPGA is an integrated circuit that can be programmed to realise interconnected logic blocks and thus perform computations at high speed. By synthesising to a gate-level netlist, we can estimate the amount of resources that are needed to perform the computations. The basic structure for the computations in a FPGA comprises of look-up tables (LUT), flip-flops (FF) and digital signal processing (DSP) blocks, and will be used in performance evaluation in this report. The purpose of the three blocks' respective application is shown in Table 3.1.

The FPGA enables parallel computing as all computations are instantiated on independent sets of LUT:s, compared to the structure in a processor where computations are shared on a single arithmetic logical unit. As a consequence, all the computations happens at the same clock cycle. To speed up computations it is therefore necessary to limit dependencies along the variables such that the data flows with every clock cycle. Vivado HLS solves most of these problems by using pipelining and dividing dependencies into independent stages. Additionally, while the previous simulations were done in Matlab, Vivado HLS uses C or C++ code to compile in the high-level synthesis process. The implementation was consequently converted to C++ code and the data type used in the calculations was 18 bit fixed-point values. The final LLR values was outputted as 6 bit fixed-point values. Due to lack of time, hardware implementation was only done for the SISO case.

Table 3.1: The basic blocks in a FPGA used to evaluate the resource usage.

Block	Task
LUT	Implements logical functions of Boolean variables
FF	Storage unit. Assists LUT with logic pipelining and data storage
DSP	An arithmetic logical unit

4

Results

In this chapter, the results from applying the extended PN-LLR scheme is presented. Most of the results are presented as bit error rate (BER) plots that show the rate of which error occurs for different E_b/N_0 levels. The chapter is divided into three sections, starting with the simulation results without the adjustment for a hardware implementation. This is followed by results of the quantised values and the requirements for an hardware implementation. All results are simulated using a layered scaled min-sum decoder with $\alpha = 0.75$. The parity-check matrix used is an irregular quasi-cyclic protograph based matrix with expansion factor $\boldsymbol{Q} = 128$ and base matrix according to the 5G standard with a code rate of 0.3235. Maximum number of iterations in the decoding is set to 25.

4.1 Simulation results

In previous chapters, it has been shown how the presence of RPN impacts communication systems as a multiplicative term. The new LLR calculation scheme derived in this thesis extends the conventional LLR-calculation by adjusting LLR:s based information about RPN. In Figure 4.1 the AWGN-LLR:s and PN-LLR:s of the least significant bit are compared for three different E_b/N_0 levels. When comparing PN-LLR with AWGN-LLR it is evident that the former have a larger spread of values for most given amplitudes. The least spread is closest to

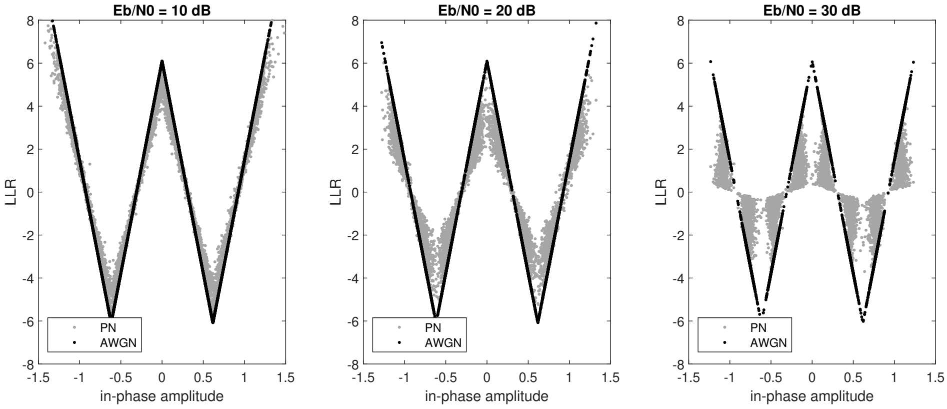


Figure 4.1: Comparison of LSB LLR values from the AWGN-LLR and PN-LLR schemes over a 64-QAM MIMO channel with $\sigma_L^2 = 10^{-4}$. X-axis represent the real part of the received symbol.

the constellation points. This is explained mainly by two factors: the impact of the amplitude of the received symbol and the distance to the closest pilot. The extended PN-LLR tunes the LLR values depending on the amplitude of the received symbol in the orthogonal dimension, i.e. when decoding the real bits the impact from phase noise will mainly depend on the amplitude of the imaginary part of the received symbol from the main channel and the real part of the second leakage channel. This follows from that the impact of phase noise vary between different parts of the constellation, as seen in Figure 3.2. The impact of varying pilot distance adds additional spread along the LLR values since two identical received symbols with different pilot distance will have been scaled slightly differently. Furthermore, it is evident how the impact of using the PN-LLR become more prominent for larger E_b/N_0 levels, i.e., when the PN variance will dominate over the AWGN variance.

In Figure 4.2 the BER-curves is shown for PN-LLR and the conventional AWGN-LLR to illustrate the performance gain of using the introduced PN-LLR scheme. The results are show for two channels: SISO and MIMO. In this first section of the results chapter, all information bits encoded using the same LDPC code. The figure shows that incorporating information about RPN in the LLR calculation yields additional coding across all both channel types compared to using the conventional assumption of a pure AWGN-channel. For the given system setup, the added coding gain for MIMO is around 0.2 dB compared to 0.1 dB for SISO.

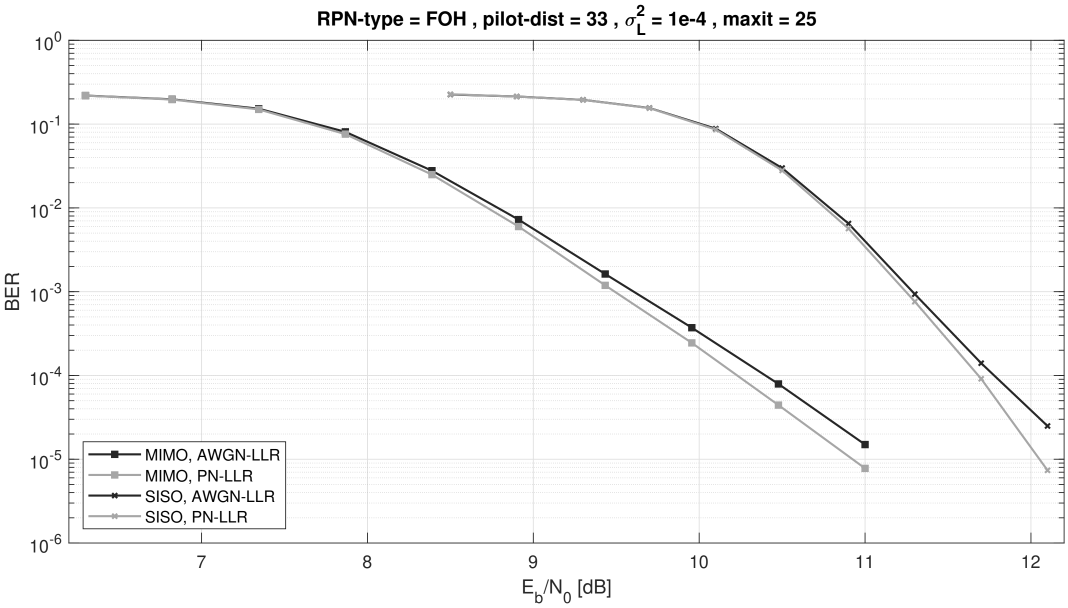
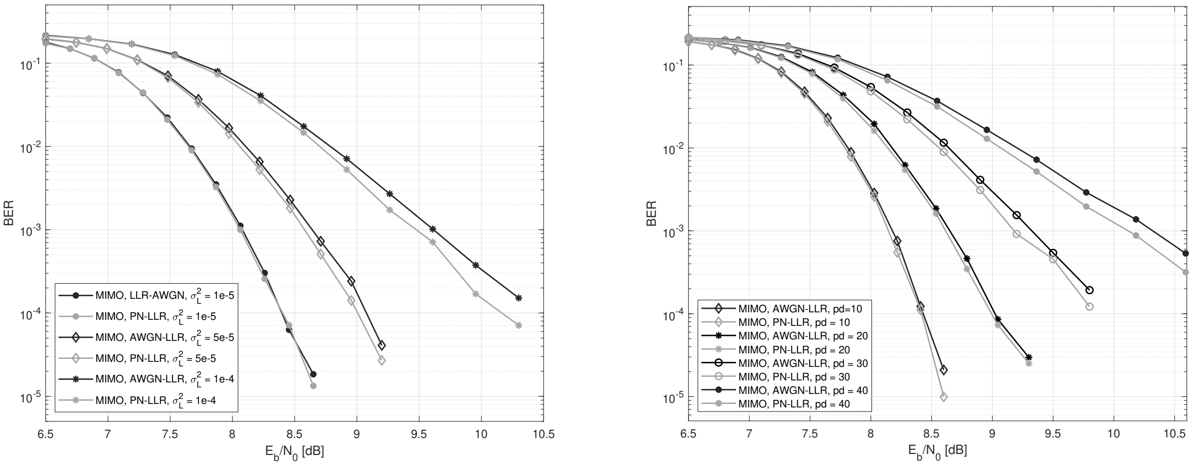


Figure 4.2: BER performance for a LDPC coded system using AWGN-LLR and PN-LLR over 4096-QAM SISO and MIMO channels with $\sigma_L = 10^{-4}$.

In addition, it is of interest to determine how the performance improvement depend on different parameters, e.g. the pilot distance in the phase tracking and the variance of the phase noise over the channel. Figure 4.3 shows the system performance for different levels of phase noise variances and pilot distances for MIMO. In Figure 4.3a it is evident how the overall performance is degraded with increasing phase noise. However, the relative improvement in performance from the RPN-extension remains. As E_b/N_0 increases, the BER improves quickly and the size of the additional gain increase slightly as the error rate become lower. This follows from a changing ratio between the variance of the AWGN och RPN. Similarly, as shown in Figure 4.3b, when the pilot distance increase the overall performance is degraded. This follows from the fact that a larger distance between pilots result in an inadequate approximation of the phase noise, hence introducing larger residual error in the interpolation step. Consequently, a larger variance of the RPN has a similar effect as to simply increasing the variance of the phase noise. Similarly to when varying the phase noise variance, the performance improvement remains when changing the pilot distance and grows slightly in size as the BER reach lower levels.



(a) MIMO with varying phase noise variance.

(b) MIMO with varying pilot distance (pd).

Figure 4.3: BER performance for a LDPC coded system using AWGN-LLR and PN-LLR over 4096-QAM SISO and MIMO channels with $\sigma_L = 10^{-4}$.

To ensure relevance for practical implementation it is important that the extended LLR calculation also support a range of constellation sizes, especially the larger modulation schemes which are more sensitive to RPN. The additional coding gain induced by the extended LLR calculation is shown for a set of constellations in Figure 4.4 below. The results show that the additional gain from the PN-LLR increases for larger constellation sizes, which is explained by decreasing minimum distance between constellation points and hence a higher sensitivity to RPN. Furthermore, in Figure 4.5 the performance with ZOH or FOH interpolation in the phase tracking step is compared for MIMO. The results indicate that the performance is slightly better for ZOH, however the additional gain for ZOH is too small to definitely conclude a performance advantage.

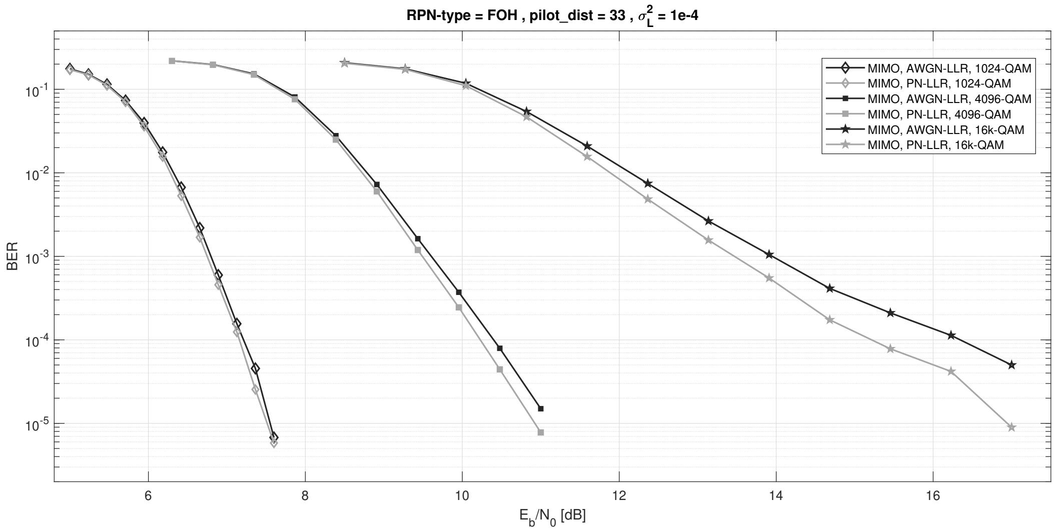


Figure 4.4: BER performance for different modulations in a LDPC coded system using AWGN-LLR and PN-LLR over SISO and MIMO channels with $\sigma_L = 10^{-4}$.

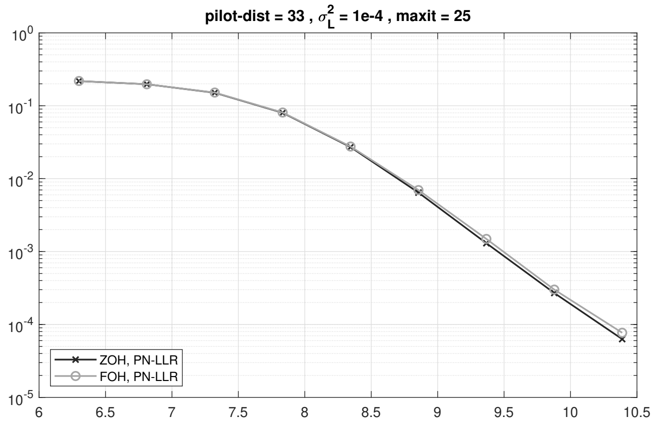


Figure 4.5: ZOH vs FOH interpolation for a MIMO system.

4.2 Quantised results adjusted for hardware implementation

In the end of the previous chapter adjustments for hardware implementation was discussed. To illustrate the practical relevance of the presented PN-LLR scheme for LDPC codes, simulations with coding only LSB:s was done to emulate the role of the LDPC code in multi-level coded systems. The LLR values was quantised to fixed-point values with six bits to simulate limitations in resolution that practical applications impose, which introduce quantisation errors. A 6-bit fixed-point value support values in the range [-7.75, 7,75] with a resolution of 0.25. In Figure 4.6 the resulting quantised PN-LLR:s of the LSB is compared with conventional AWGN-LLR:s over a MIMO channel using 64-QAM. It is evident that the reduction in resolution had little impact on the resulting PN-LLR:s if compared with the non-quantised LLR:s presented in Figure 4.1, which indicates that it is feasible to implement in practice. As have been discussed in

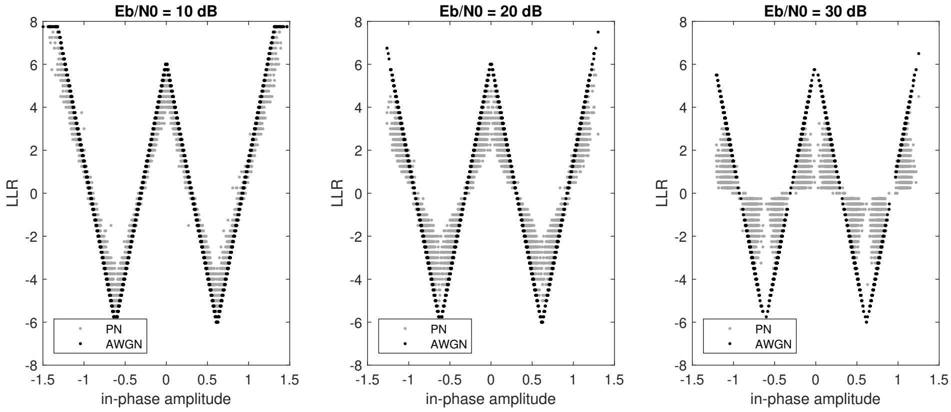


Figure 4.6: Comparison of quantised LSB LLR values from the AWGN-LLR and PN-LLR schemes over a 64-QAM MIMO channel with $\sigma_L^2 = 10^{-4}$. X-axis represent the real part of the received symbol.

previous sections, the LSB is more affected by RPN than other bits due to its relative proximity to decision boundaries. Hence the additional gain from the extended PN-LLR scheme is larger for the LSB:s than compared to the more significant bits. In Figure 4.7 are BER-curves for the LSB shown for both SISO and MIMO, which at a BER of 10^{-4} indicates a coding gain of around 0.16 dB for MIMO and around 0.05 dB for SISO.

When comparing the relative performance improvement it should also be noted that the results in Figure 4.7 are simulated for lower PN variance compared to that in Figure 4.2, $\sigma_{PN} = 5e-6$ instead of $1e-4$, which illustrates the larger impact on LSB:s. In addition, there is also a large difference in performance improvement between MIMO and SISO. The difference follows from that MIMO experience differential RPN due to impairments of channel leakage which effectively increase the variance of the experienced phase noise of the channel. As discussed in section 4.1, the gain from the PN-LLR scheme increase with larger variance of the phase noise, hence MIMO will gain more from the new scheme compared to SISO.

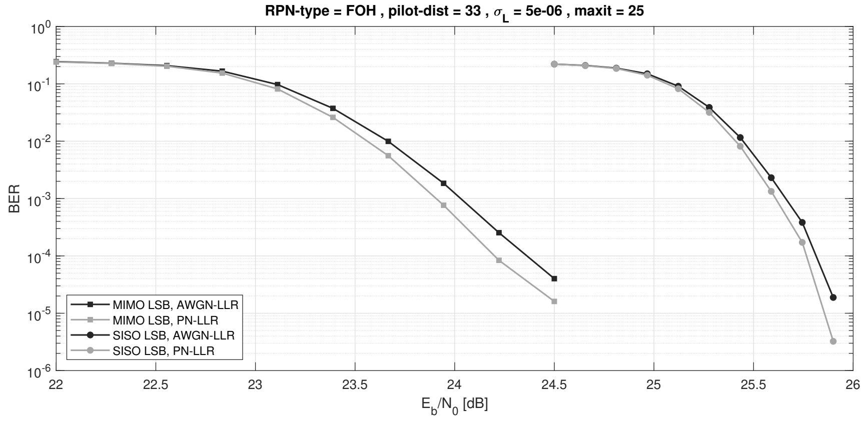


Figure 4.7: BER performance for LDPC coded system for LSB using quantised AWGN-LLR and PN-LLR over 4096-QAM SISO and MIMO channels with $\sigma_L = 5 \cdot 10^{-6}$.

4.3 HLS results of hardware implementation

In the previous section, it was shown that implementing the PN-LLR scheme gave improved performance, even after quantisation to support the realisation in hardware. Next, it is of interest to understand the added cost and resource requirements that follow from extending the conventional LLR calculation scheme. In Table 4.1 resource metrics from Vivado HLS, which was introduced in the method chapter, is presented. The resource metrics are defined in Table 3.1 and PN-LLR* represent PN-LLR without divisions. The results show that the improved performance come at a large cost in terms hardware resource usage. However, a large majority of the added cost is attributed to the division of the determinant in eq. (3.27). By removing the divisions the required resources drastically decrease and brings the resource allocations of the PN-LLR scheme closer to the original AWGN-LLR scheme. In addition, Table 4.1 also compares the resource usage depending on the choice of interpolation scheme. The results suggest that the resource usage can be slightly decreased by using ZOH instead of FOH. However, the reduced resource allocation is traded for less accurate phase estimation.

Table 4.1: Resource usage of different LLR calculations acquired from Vivado HLS.

Type	**LUT**	**FF**	**DSP**
AWGN-LLR	841	860	6
PN-LLR (ZOH)	51,491	36,738	103
PN-LLR (FOH)	51,754	37,025	116
PN-LLR* (FOH)	2,499	3,782	85

5

Discussion

It has been shown that the implemented PN-LLR improved the performance, measured in terms of BER, of the LDPC decoder for both SISO and MIMO. Furthermore, the improvement remained when the LLR values was quantised down to a resolution of 6 bits for the LSBs. However, considering the cost-benefit aspect, the hardware implementation of such a system was shown to be more complicated than the regular AWGN-LLR. This section will discuss some of the choices made and what further developments that could continue the work.

5.1 System performance evaluation

It is evident that the implemented PN-LLR scheme outperforms the conventional LLR scheme for the given system model in terms of BER. The size of the improvement is dependent on the variance of the phase noise in relation to the variance of the AWGN over the channel. In Figure 4.3a it was shown how the overall performance is degraded as the PN variance increases, but that the relative improvement of the PN-LLR scheme increases. This suggests that with larger PN variance, there is a bigger gain to reap. In addition, in Figure 4.1 it was shown how the the impact of the PN-LLR scheme becomes more prominent as E_b/N_0 increase, i.e. as the PN to AWGN ratio becomes large enough and PN starts dominating. When increasing the pilot spacing, the system performance behaves similar to when increasing the PN variance. More scattered pilots will degrade the accuracy of the interpolation step in the phase tracking, causing greater RPN, which effectively is equivalent to increasing the PN variance.

The system model used in this project intend to model a line-of-sight wireless backhaul link, which is characterised by high SNR and large constellations to facilitate large data rates. Figure 4.4 showed how the overall performance decreased for larger constellation sizes as a result of a decreased minimum distance, hence less resilience to noise. However, the size of the relative improvement from the PN-LLR scheme increased as the constellation grow. Once again, the conclusion is that the additional gain from the PN-LLR scheme increase with increased impact from RPN on received symbols. Hence, the introduced scheme is particularly suitable for systems using large constellations, having solid SNR conditions, using larger pilot spacing, and with significant PN present.

The performance was also evaluated for the two channels: SISO and MIMO. Both channels saw a performance improvement with the PN-LLR scheme, however the improvement was greater for MIMO as is shown in Figure 4.2. This can be accounted to the differential phase noise caused by impairments of channel leakage from the cancellation process at the receiver. The resulting RPN for each channel is depicted in Figure 3.2. Clearly, MIMO experiences a larger impact from PN compared to SISO, hence the performance improvement is greater.

Finally, the system was simulated using two different interpolation schemes in the phase tracking: ZOH and FOH. The choice of interpolation scheme slightly impacts the RPN, where ZOH introduce more noise since the interpolations is sub-optimal compared to the linear interpolation of FOH. However, using ZOH interpolation reduces complexity when realised in hardware, as seen in Table 4.1. Hence, it is interesting to evaluate whether the choice of interpolation technique impacts the performance of the PN-LLR scheme. Figure 4.5 showed almost identical performance for ZOH and FOH, with indications of a slightly better performance for ZOH. However, it is hard to draw any definite conclusion since the curves are close together.

5.2 Quantisation errors and LSB coding

When developing communication systems, it's crucial to minimise redundancy and resource usage to achieve satisfactory performance at a low cost. Higher-resolution arithmetic increases the hardware cost, and it is therefore important to balance the resolution of the calculations and the system performance. The implemented PN-LLR scheme was quantised by replacing double precision data types with 6-bit fixed-point variables. However, the quantisation proved to not have any significant impact on the result. This indicates that the contribution from adding information about RPN in LLR calculations is significant enough to not be discarded when lowering the resolution,at least down to 6 bits for LSBs. The impact of varying the resolution has not been further investigated, and could be subject to future studies.

From the results it was evident that the PN-LLR scheme improved the performance, both for when all bits were coded and for when only considering LSBs. However, the additional coding gain seen in Figure 4.2, where all bits are encoded, is quite small given the level of phase noise variance at $\sigma_L^2 = 10^{-4}$. From previous discussion we remember that the size of the performance improvement scales with the variance of the phase noise σ_L^2. Hence, in practice the gain from using the PN-LLR scheme would be rather small, especially for larger constellations where only a fraction of coded bits will experience a substantial gain. However, when looking only at LSBs, the performance improvement is significant, which is seen in Figure 4.7. This suggest that the PN-LLR scheme is most relevant for LSBs when using larger constellations, since that is where it provides a significant gain in return for the added cost and complexity. In addition, the results in both Figure 4.2 and 4.7 show that the additional gain from the PN-LLR scheme is larger for MIMO compared to SISO, hence a practical implementation might be more attractive for MIMO, compared to SISO, from a cost-benefit perspective.

As discussed in the background of this thesis, the fundamental idea of coding is to add redundancy to increase the systems resilience to errors, hence trading lower coding rate for improved performance. It is desirable to maximise the coding rate for any given level of performance. We have previously seen that different bits are affected differently by noise, mainly due to varying proximity to decision boundaries. This difference tends to be large for larger constellations since each symbol represents more bits, e.g. for 4096-QAM each symbol represent 12 bits.

As a result, multi-level coding can be used to optimise the trade-off between coding rate and performance by using different coding schemes for bits of different significance. Due to the near-capacity characteristics of LDPC codes, they are a good choice for encoding the LSBs. In the result chapter it is evident that the improved performance from the PN-LLR scheme is greater for LSBs, compared to when coding all bits, hence offering a more attractive cost-benefit trade-off. This suggests applicability in multi-level coding schemes where maximised benefit can be gained for the LSB:s, while using other coding schemes with larger coding rate for more significant bits needing less protection. In section 4.2, the impact of coding only the least significant bit of each dimension was shown. However, it should be noted that the second least significant bit can also be coded together with the LSB and achieve equal performance if Gray coding is applied separately for the two least significant bits of each dimension, in contrast to Gray coding all bits as in Figure 3.4.

5.3 Hardware resource requirements

In the result of the hardware synthesis it was seen that the implementation of the PN-LLR required more hardware to be realised than for AWGN-LLR. The division of the determinant was especially expensive. For further work, a approximation of the division is necessary to make it attractive for an implementation in a real system. A famous approximation is the so called *fast inverse square root* [19] created for the 1999 game Quake III Arena. However, this approximation works with floating point numbers and not fixed point numbers as in this report. The algorithm is based on that type casting a float to an integer is a good approximation of the logarithm. The inverse square root is then computed as

$$\log_2\left(\frac{1}{\sqrt{x}}\right) = -0.5\log_2(x). \tag{5.1}$$

In [20] an algorithm for a fast base-2 logarithm is suggested using a priority encoder, barrel shifter and a look up table. This algorithm could then be combined with eq. (5.1), without the square root.

Additionally, considering Figure 4.1 an approximation of the PN-LLR could perhaps be found using the received symbol and the RPN variance σ_θ^2 to further decrease the complexity.

5.4 Suggestions on further research

All the simulations done were simulated down to a BER of around 10^{-5}. It would be interesting to further simulate to lower BER-levels, especially down to the error floor region, to investigate if the PN extension provides better performance in these regions. However, this requires time consuming simulations and may not be feasible. There exists techniques for estimating error floor regions [21] and also specific codes with high error floor, such as Margulis codes [22]. The estimation techniques are usually based on the AWGN assumption and would therefore have to be extended for the PN case.

The scope of this project has included improving the LLR calculations in the presence of phase noise and has been detached from the specific LDPC decoding techniques. There has been work proposing joint decoding and phase compensation algorithms, such in [15] and [7], that combines the phase estimation and decoding. Such algorithms is based around belief propagation on the factor graph representing the joint á posterior probability mass function of information bits and the channel, including the phase noise.

6

Conclusion

We have developed an extension of the conventional LLR-calculation by considering phase noise in addition to AWGN, ultimately increasing the performance of the LDPC system. The extension is posing as a standalone block to facilitate practical incorporation into existing solutions. Furthermore, the thesis aimed at identifying necessary adjustments of the theoretical model to enable hardware implementation. The results have shown that the coding gain can be improved by implementing the proposed PN-LLR scheme, which in addition to the distance to nearest constellation point also considers its location in the constellation diagram and the distance to nearest pilot. The additional gain scale with the phase noise and consequently increase for large constellations, large pilot spacing, and increasing phase noise variance. In addition, the proposed scheme was shown to be more effective for MIMO compared to SISO. The results also showed that the performance improvement was larger for less significant bits, compared to when looking at all bits. Mainly due to LSBs being more sensitive to phase noise. This suggests that the PN-LLR scheme could be suitable for multi-level coding schemes with large constellation sizes, which often is used in line-of-sight wireless backhaul links. Hardware synthesis simulations showed that the proposed PN-LLR scheme significantly increased the hardware resource usage compared to the conventional AWGN-LLR scheme. However, the absolute majority of the increase is attributed to the high cost of divisions in hardware. In conclusion, this thesis has shown that it is possible to extend the conventional LLR calculation to consider phase noise in addition to AWGN and consequently improve the coding gain in the presence of residual phase noise.

References

[1] C. E. Shannon, “A mathematical theory of communication,” *The Bell System Technical Journal*, vol. 27, no. 3, pp. 379–423, 1948. DOI: 10.1002/j.1538-7305.1948.tb01338.x.

[2] W. Ryan and S. Lin, *Channel Codes: Classical and Modern.* Cambridge University Press, 2009. DOI: 10.1017/CBO9780511803253.

[3] R. Gallager, “Low-density parity-check codes,” *IRE Transactions on Information Theory*, vol. 8, no. 1, pp. 21–28, 1962. DOI: 10.1109/TIT.1962.1057683.

[4] R. Tanner, “A recursive approach to low complexity codes,” *IEEE Transactions on Information Theory*, vol. 27, no. 5, pp. 533–547, 1981. DOI: 10.1109/TIT.1981.1056404.

[5] D. MacKay, “Good error-correcting codes based on very sparse matrices,” in *Proceedings of IEEE International Symposium on Information Theory*, 1997, pp. 113–. DOI: 10.1109/ISIT.1997.613028.

[6] J. H. Bae, A. Abotabl, H.-P. Lin, K.-B. Song, and J. Lee, “An overview of channel coding for 5g nr cellular communications,” *APSIPA Transactions on Signal and Information Processing*, vol. 8, e17, 2019. DOI: 10.1017/ATSIP.2019.10.

[7] R. Krishnan, G. Colavolpe, A. Graell i Amat, and T. Eriksson, “Algorithms for joint phase estimation and decoding for mimo systems in the presence of phase noise and quasi-static fading channels,” *IEEE Transactions on Signal Processing*, vol. 63, no. 13, pp. 3360–3375, 2015. DOI: 10.1109/TSP.2015.2420533.

[8] J. Thorpe, “Low-Density Parity-Check (LDPC) Codes Constructed from Protographs,” *Interplanetary Network Progress Report*, vol. 42-154, pp. 1–7, Aug. 2003.

[9] M. Franceschini, G. Ferrari, and R. Raheli, *LDPC Coded Modulations.* Springer Berlin, Heidelberg, 2009. DOI: 10.1007/978-3-540-69457-1.

[10] J. R. Barry, E. A. Lee, and D. G. Messerschmitt, *Digital communication.* Kluwer Academic, 2004, ISBN: 0792375483.

[11] L. Tomba, “On the effect of wiener phase noise in ofdm systems,” *IEEE Transactions on Communications*, vol. 46, no. 5, pp. 580–583, 1998. DOI: 10.1109/26.668721.

[12] H. Mehrpouyan, A. A. Nasir, S. D. Blostein, T. Eriksson, G. K. Karagiannidis, and T. Svensson, “Joint estimation of channel and oscillator phase noise in mimo systems,” *IEEE Transactions on Signal Processing*, vol. 60, no. 9, pp. 4790–4807, 2012. DOI: 10.1109/TSP.2012.2202652.

[13] C. Camarda, "Analysis and design of line of sight mimo transmission systems," Ph.D. dissertation, Politecnico di Torino, Torino, Italy, 2014. DOI: `10.6092/polito/porto/2538729`.

[14] P. Neshaastegaran and A. H. Banihashemi, "Log-likelihood ratio calculation for pilot symbol assisted coded modulation schemes with residual phase noise," *IEEE Transactions on Communications*, vol. 67, no. 5, pp. 3782–3790, 2019. DOI: `10.1109/TCOMM.2019.2896190`.

[15] G. Colavolpe, A. Barbieri, and G. Caire, "Algorithms for iterative decoding in the presence of strong phase noise," *IEEE Journal on Selected Areas in Communications*, vol. 23, no. 9, pp. 1748–1757, 2005. DOI: `10.1109/JSAC.2005.853813`.

[16] M. Sjodin, P. Ligander, L. Bao, and J. Hansryd, "A 40.2 bps/hz single polarization 4x4 line-of-sight mimo link with unsynchronized oscillators," in *2019 IEEE Radio and Wireless Symposium (RWS)*, 2019, pp. 1–3. DOI: `10.1109/RWS.2019.8714464`.

[17] U. Wachsmann, R. Fischer, and J. Huber, "Multilevel codes: Theoretical concepts and practical design rules," *IEEE Transactions on Information Theory*, vol. 45, no. 5, pp. 1361–1391, 1999. DOI: `10.1109/18.771140`.

[18] Xilinx. "Introduction to fpga design with vivado high-level synthesis." (2019), [Online]. Available: `https://docs.xilinx.com/v/u/en-US/ug998-vivado-intro-fpga-design-hls` (visited on 05/12/2023).

[19] C. Lomont, "Fast inverse square root," 2003. [Online]. Available: `http://www.matrix67.com/data/InvSqrt.pdf` (visited on 05/12/2023).

[20] M. Dunn. "Fast base-2 logarithms." (2011), [Online]. Available: `http://www.cantares.on.ca/extras.html` (visited on 05/12/2023).

[21] A. Farsiabi and A. H. Banihashemi, "Error floor estimation of ldpc decoders — a code independent approach to measuring the harmfulness of trapping sets," *IEEE Transactions on Communications*, vol. 68, no. 5, pp. 2667–2679, 2020. DOI: `10.1109/TCOMM.2020.2975195`.

[22] T. J. Richardson, "Error floors of ldpc codes," Urbana-Champaign, USA, 2003.

A

Appendix 1

A.1 Derivation of LLR relation for SPC codes

Below is the full derivation of the equality in equation (1.24) presented.

$$
\begin{aligned}
\tanh\left(\frac{1}{2}\log\left(\frac{p_0}{p_1}\right)\right) &= \frac{e^{\left(\frac{1}{2}\log\left(\frac{p_0}{p_1}\right)\right)} - e^{-\left(\frac{1}{2}\log\left(\frac{p_0}{p_1}\right)\right)}}{e^{\left(\frac{1}{2}\log\left(\frac{p_0}{p_1}\right)\right)} + e^{-\left(\frac{1}{2}\log\left(\frac{p_0}{p_1}\right)\right)}} = \frac{\sqrt{\frac{p_0}{p_1}} - \sqrt{\frac{p_1}{p_0}}}{\sqrt{\frac{p_0}{p_1}} + \sqrt{\frac{p_1}{p_0}}} \qquad \text{(A.1)}\\
&= \frac{\left(\frac{p_0}{p_1}\right) - 2\sqrt{\frac{p_0 p_1}{p_1 p_0}} + \left(\frac{p_1}{p_0}\right)}{\left(\frac{p_0}{p_1}\right) - \left(\frac{p_1}{p_0}\right)} = \frac{\left(\frac{p_0}{p_1}\right) + \left(\frac{p_1}{p_0}\right) - 2}{\left(\frac{p_0}{p_1}\right) - \left(\frac{p_1}{p_0}\right)}\\
&= \{p_0 = 1 - p_1\} = \frac{\left(\frac{1-p_1}{p_1}\right) + \left(\frac{p_1}{1-p_1}\right) - 2}{\left(\frac{1-p_1}{p_1}\right) - \left(\frac{p_1}{1-p_1}\right)}\\
&= \frac{\frac{1-2p_1+2p_1^2}{p_1(1-p_1)} - \frac{2p_1-2p_1^2}{p_1(1-p_1)}}{\frac{1-2p_1}{p_1(1-p_1)}} = \frac{1-4p_1+4p_1^2}{1-2p_1}\\
&= \frac{(1-2p_1)^2}{1-2p_1} = 1 - 2p_1
\end{aligned}
$$

A.2 Derivation of RPN variance

Below is the full derivation of the variance of the residual phase noise (RPN) when using ZOH and FOH as interpolation methods. The derivation of FOH is derived with support from [14].

A.2.1 ZOH interpolation

For ZOH the variance σ_θ^2 is calculated as follows. First we introduce the notion $\mathbb{1}_1 = 1$, if $m \leq P/2$ and $\mathbb{1}_2 = 1$, if $m > P/2$. By combining eq. (2.22) and eq. (2.20), the RPN θ can be expressed as

$$
\begin{aligned}
\theta_{i+m} &= \phi_{i+m} - \hat{\phi}_{i+m} = \phi_{i+m} - (\phi_i^P \mathbb{1}_1 + \phi_{i+P}^P \mathbb{1}_2) = \qquad \text{(A.2)}\\
&= \phi_{i+m} - (\phi_i^W \mathbb{1}_1 + \phi_{i+p}^W \mathbb{1}_2) - (\phi_i^{PN} \mathbb{1}_1 + \phi_{i+p}^{PN} \mathbb{1}_2) =
\end{aligned}
$$

$$\begin{aligned} &= -\,(\phi_i^W \mathbb{1}_1 + \phi_{i+p}^W \mathbb{1}_2) + (\phi_{i+m} - (\phi_i^{PN} \mathbb{1}_1 + \phi_{i+p}^{PN} \mathbb{1}_2)) = \\ &= -\,\theta_{i+m}^W + \theta_{i+m}^{PN}. \end{aligned}$$

As seen above, θ_{i+m}^W and θ_{i+m}^{PN} are both linear combinations of zero-mean Gaussian processes, which means that their variances, $\sigma^2_{\theta,\,\mathrm{W}}$ and $\sigma^2_{\theta,\,\mathrm{PN}}$, are sums of the variance of the corresponding Gaussian processes. As a result, the variance of the two components θ_{i+m}^W and θ_{i+m}^{PN} for ZOH can be expressed as

$$\begin{aligned} \sigma^2_{\theta,\,\mathrm{W}} &= \mathrm{Var}(\phi_i^W \mathbb{1}_1 + \phi_{i+P}^W \mathbb{1}_2) = \mathrm{Var}(\phi_i^W)(\mathbb{1}_1 + \mathbb{1}_2) \\ &= \mathrm{Var}(\phi_i^W) = \sigma_n^2, \end{aligned} \tag{A.3}$$

and similarly

$$\begin{aligned} \sigma^2_{\theta,\,\mathrm{PN}} &= \mathrm{Var}(\phi_{i+m} - (\phi_i^{PN} \mathbb{1}_1 + \phi_{i+P}^{PN} \mathbb{1}_2)) = \\ &= \mathrm{Var}(\phi_{i+m} - \phi_i^{PN})\mathbb{1}_1 + \mathrm{Var}(\phi_{i+m} - \phi_{i+P}^{PN})\mathbb{1}_2 = \\ &= \sigma_L^2 \cdot m \cdot \mathbb{1}_1 + \sigma_L^2 \cdot (P-m) \cdot \mathbb{1}_2 = \\ &= \begin{cases} \sigma_L^2 \cdot m & \text{if } m \leq P/2 \\ \sigma_L^2 \cdot (P-m) & \text{if } m > P/2. \end{cases} \end{aligned} \tag{A.4}$$

Combining eq. (A.3) and (A.4), the expression for the variance of the RPN for ZOH is

$$\sigma_\theta^2 = \begin{cases} \sigma_n^2 + \sigma_L^2 \cdot m & \text{if } m \leq P/2 \\ \sigma_n^2 + \sigma_L^2 \cdot (P-m) & \text{if } m > P/2. \end{cases} \tag{A.5}$$

A.2.2 FOH interpolation

A similar approach is used for FOH to calculate the variance σ_θ^2. By combining eq. (2.21) and eq. (2.22), the RPN θ for FOH can be expressed as

$$\begin{aligned} \theta_{i+m} = \phi_{i+m} - \hat{\phi}_{i+m} &= \phi_{i+m} - \left(\frac{\phi_{i+P}^P - \phi_i^P}{P} m + \phi_i^P \right) = \\ &= \phi_{i+m} - \left(\frac{(\phi_{i+P}^W + \phi_{i+P}^{PN}) - (\phi_i^W + \phi_i^{PN})}{P} m + (\phi_i^W + \phi_i^{PN}) \right) = \\ &= \left(\frac{\phi_i^W - \phi_{i+P}^W}{P} m - \phi_i^W \right) + \left(\phi_{i+m} - \phi_i^{PN} - \frac{\phi_{i+P}^{PN} - \phi_i^{PN}}{P} m \right) = \\ &= \theta_{i+m}^W + \theta_{i+m}^{PN} \end{aligned} \tag{A.6}$$

where θ_{i+m}^W and θ_{i+m}^{PN} are independent. This implies that the variance of θ_{i+m} equals the sum of the variance of the two components accordingly

$$\sigma_\theta^2(i+m) = \sigma_{\theta^W}^2(i+m) + \sigma_{\theta^{PN}}^2(i+m). \tag{A.7}$$

To calculate $\sigma_{\theta^W}^2(i+m)$ it is useful to first rewrite θ_{i+m}^W as

$$\theta_{i+m}^W = \left(\frac{m}{P} - 1 \right) \phi_i^W - \frac{m}{P} \phi_{i+P}^W \tag{A.8}$$

where ϕ_i^W and ϕ_{i+P}^W are zero-mean Gaussian random variables, both with variance σ^2. From here the variance of θ_{i+m}^W can be expressed as

$$\sigma^2_{\theta^W}(i+m) = \left(\frac{m}{P} - 1\right)^2 \sigma_n^2 + \frac{m^2}{P^2}\sigma_n^2 = \left(1 - 2\frac{m}{P} - 2\frac{m^2}{P^2}\right)\sigma_n^2. \tag{A.9}$$

To calculate the second term $\sigma^2_{\theta^{PN}}(i+m)$ in eq. (A.7) it is similarly useful to rewrite θ_{i+m}^{PN} as

$$\theta_{i+m}^{PN} = \left(\frac{m}{P} - 1\right)(\phi_{i+m}^{PN} - \phi_i^{PN}) - \frac{m}{P}(\phi_{i+P}^{PN} - \phi_{i+m}^{PN}) \tag{A.10}$$

where $(\phi_{i+m}^{PN} - \phi_i^{PN})$ and $(\phi_{i+P}^{PN} - \phi_{i+m}^{PN})$ both are increments of a Wiener process over two separate non-overlapping intervals and are as a consequence independent. Based on eq. (2.15) it is clear that

$$\begin{aligned}(\phi_{i+m}^{PN} - \phi_i^{PN}) &\sim \mathcal{N}(0,\ \sigma_L^2 m)\\ (\phi_{i+P}^{PN} - \phi_{i+m}^{PN}) &\sim \mathcal{N}(0,\ \sigma_L^2 (P-m)).\end{aligned} \tag{A.11}$$

From this $\sigma^2_{\theta^{PN}}(i+m)$ can be expressed as

$$\sigma^2_{\theta^{PN}}(i+m) = \left(1 - \frac{m}{P}\right)^2 \sigma_L^2 m + \frac{m^2}{P^2}\sigma_L^2(P-m) = m\left(1 - \frac{m}{P}\right)\sigma_L^2. \tag{A.12}$$

Finally, by inserting eq. (A.9) and (A.12) into eq. (A.7) one get the final expression of σ_θ^2

$$\sigma_\theta^2(m) = m\left(1 - \frac{m}{P}\right)\sigma_L^2 + \left(1 - 2\frac{m}{P} + 2\frac{m^2}{P^2}\right)\sigma_n^2. \tag{A.13}$$

as presented in eq. (2.24).

A.3 Derivation of pilot-based phase estimation for MIMO

Below is the derivation of the pilot-based phase estimation in the MIMO system. Starting from eq. (3.2), we want to compute $\boldsymbol{y} = \hat{\boldsymbol{P}}^{-1}\boldsymbol{r}$:

$$\begin{aligned}
\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} &= \begin{bmatrix} e^{j\hat{\phi}_{11}} & -je^{j\hat{\phi}_{21}} \\ -je^{j\hat{\phi}_{12}} & e^{j\hat{\phi}_{22}} \end{bmatrix}^{-1} \begin{bmatrix} r_1 \\ r_2 \end{bmatrix}\\
&= \frac{1}{\det(\hat{\boldsymbol{P}})} \begin{bmatrix} e^{j\hat{\phi}_{22}} & je^{j\hat{\phi}_{21}} \\ je^{j\hat{\phi}_{12}} & e^{j\hat{\phi}_{11}} \end{bmatrix} \begin{bmatrix} e^{j\phi_{11}} & -je^{j\phi_{21}} \\ -je^{j\phi_{12}} & e^{j\phi_{22}} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}\\
&= \frac{1}{\det(\hat{\boldsymbol{P}})} \begin{bmatrix} e^{j\hat{\phi}_{22}} & je^{j\hat{\phi}_{21}} \\ je^{j\hat{\phi}_{12}} & e^{j\hat{\phi}_{11}} \end{bmatrix} \begin{bmatrix} x_1 e^{j\phi_{11}} + jx_2 e^{j\phi_{21}} \\ -jx_1 e^{j\phi_{12}} + x_2 e^{j\phi_{22}} \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}\\
&= \frac{1}{\det(\hat{\boldsymbol{P}})} \begin{bmatrix} x_1 e^{j(\phi_{11}+\hat{\phi}_{22})} - jx_2 e^{j(\phi_{21}+\hat{\phi}_{22})} + x_1 e^{j(\phi_{12}+\hat{\phi}_{21})} + jx_2 e^{j(\phi_{22}+\hat{\phi}_{21})} \\ jx_1 e^{j(\phi_{11}+\hat{\phi}_{12})} + x_2 e^{j(\phi_{21}+\hat{\phi}_{12})} - jx_1 e^{j(\phi_{12}+\hat{\phi}_{11})} + x_2 e^{j(\phi_{22}+\hat{\phi}_{11})} \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}\\
&= \frac{1}{\det(\hat{\boldsymbol{P}})} \begin{bmatrix} x_1(e^{j(\phi_{11}+\hat{\phi}_{22})} + e^{j(\phi_{12}+\hat{\phi}_{21})}) + jx_2(e^{j(\phi_{22}+\hat{\phi}_{21})} - e^{j(\phi_{21}+\hat{\phi}_{22})}) \\ jx_1(e^{j(\phi_{11}+\hat{\phi}_{12})} - e^{j(\phi_{12}+\hat{\phi}_{11})}) + x_2(e^{j(\phi_{22}+\hat{\phi}_{11})} + e^{j(\phi_{21}+\hat{\phi}_{12})} \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}.
\end{aligned} \tag{A.14}$$

By using the definition for the residual phase noise $\theta_{tr} = \phi_{tr} - \widehat{\phi_{tr}}$ for transmitter t and receiver r and computing the determinant $\det(\hat{\boldsymbol{P}}) = e^{j(\hat{\phi}_{11}+\hat{\phi}_{22})} + e^{j(\hat{\phi}_{21}+\hat{\phi}_{12})}$ the final expression for the received phase compensated signal is

$$\begin{bmatrix} y_1 \\ y_2 \end{bmatrix} = \begin{bmatrix} \frac{x_1(e^{j(\hat{\phi}_{11}+\hat{\phi}_{22}+\theta_{11})}+e^{j(\hat{\phi}_{12}+\hat{\phi}_{21}+\theta_{12})})+jx_2(e^{j(\hat{\phi}_{22}+\hat{\phi}_{21}+\theta_{22})}-e^{j(\hat{\phi}_{21}+\hat{\phi}_{22}+\theta_{21})})}{e^{j(\hat{\phi}_{11}+\hat{\phi}_{22})}+e^{j(\hat{\phi}_{21}+\hat{\phi}_{12})}} \\ \frac{jx_1(e^{j(\hat{\phi}_{11}+\hat{\phi}_{12}+\theta_{11})}-e^{j(\hat{\phi}_{12}+\hat{\phi}_{11}+\theta_{12})})+x_2(e^{j(\hat{\phi}_{22}+\hat{\phi}_{11}+\theta_{22})}+e^{j(\hat{\phi}_{21}+\hat{\phi}_{12}+\theta_{21})}}{e^{j(\hat{\phi}_{11}+\hat{\phi}_{22})}+e^{j(\hat{\phi}_{21}+\hat{\phi}_{12})}} \end{bmatrix} + \begin{bmatrix} n_1 \\ n_2 \end{bmatrix}. \tag{A.15}$$

DEPARTMENT OF ELECTRICAL ENGINEERING
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden
www.chalmers.se

CHALMERS
UNIVERSITY OF TECHNOLOGY