Department of Informatics

Master Thesis

Secure, reliable, and efficient communication over the wiretap channel

Joakim Algrøy
Supervisor: Øyvind Ytrehus

May 29, 2022

Abstract

Secure wireless communication between devices is essential for modern communication systems. Physical-layer security over the wiretap channel may provide an additional level of secrecy beyond the current cryptographic approaches. Given a sender Alice, a legitimate receiver Bob, and a malicious eavesdropper Eve, the wiretap channel occurs when Eve experiences a worse signal-to-noise ratio than Bob. Previous study of the wiretap channel has tended to make assumptions that ignore the reality of wireless communication. This thesis presents a study of short block length codes with the aim of both reliability for Bob and confusion for Eve. The standard approach to wiretap coding is shown to be very inefficient for reliability. Quantifying Eve's confusion in terms of entropy is not solved in many cases, though it is possible for codes with a moderate complexity trellis representation. Using error rate arguments, error correcting codes with steep performance curves turn out to be desirable both for reliability and confusion.

Acknowledgements

First, I would like to thank my supervisor, Øyvind Ytrehus, for being a great mentor and teacher. I am very grateful for the opportunity to work with you, and for the positive encouragement which has made this thesis an exciting project instead of an obligation.

I also want to thank my friends and fellow students for making this a fun year, and for the many big and small conversations over lunch. Even on days I felt completely stuck, I always had 11:30 to look forward to.

Finally, I would like to thank everyone at Simula UiB for all the opportunities you have given me, and for providing a great work environment.

Contents

Nomenclature

x	Scalar value
$\mathbf{x}$	Vector
X	Random variable
$\mathbf{X}$	Matrix
$\mathcal{X}$	Alphabet
$\mathcal{X}^n$	Set of sequences of length n with elements from $\mathcal{X}$
$\{0,1\}^n$	Set of binary sequences of length n
$\mathbb{F}_2$	The finite field of two elements
$H(X)$	Entropy of X
FER	Frame error rate
BER	Bit error rate
Γ	Security gap

Chapter 1

Introduction

1.1 Motivation

Modern communication is characterized by an abundance of digital devices which communicate wirelessly. In addition to personal devices such as phones, computers, and wearables, the internet of things is connecting a massive number of wireless devices in industry, infrastructure, offices, and homes. In almost all such applications, there is information being communicated that is worth keeping secret, whether it relates to personal privacy, passwords, financial details, trade secrets, military intelligence, government secrets, etc.

Wireless communication is by nature a broadcast medium. We do not have complete control over where a wireless signal we transmit ends up. Consequently, eavesdropping is a big potential security risk for wireless communication. Eavesdropping is when a malicious actor listens in on communication between legitimate parties. Traditionally, eavesdropping has been dealt with through cryptography. Cryptographic systems do not prevent an eavesdropper from listening to the transmitted data, but instead modifies the data such that only the legitimate receiver, who possesses a secret key, is able to understand what is being communicated.

Let us present an analogy with three people; Alice, Bob, and Eve. Alice wants to tell Bob a secret. However, she does not want Eve to know the secret. Cryptography may be viewed as Alice and Bob agreeing on a secret language, and talking so that Eve can hear them, but is unable to understand. In this thesis we will study an alternative approach, which is more akin to Alice whispering to Bob, with a low enough volume that Eve is unable to discern what is being said from the surrounding noise.

This situation where Eve has a "signal" of lower quality than Bob, is called the *wiretap channel*. While it can be seen as an alternative to cryptography, we do not consider it as a replacement. Instead, wiretap communication may for instance be implemented at the physical layer, as a supplement to the cryptography used in software. It may also be used specifically for the initial key exchange of a cryptographic protocol, as it is generally the stage most vulnerable to attack. In theory, there are stronger guarantees about Eve's information available for the wiretap channel than for standard cryptographic systems [1].

As an example, government agencies are known to require secrecy for many decades [2]. A dedicated enemy state may store encrypted data observed through eavesdropping today, only to attack it decades later, when quantum computers, more powerful supercomputers, and other unknown tools for attacking cryptography may be available. Under certain assumptions across the wiretap channel, this is impossible, as the transmitted information was not received in the first place.

In addition to the arguments above, cryptography may be too computationally expensive for certain low power applications. We also have no need for the parties to exchange a key securely before starting communication.

1.2 Goal

Our goal with this thesis is to evaluate methods and approaches to coding for the wiretap channel, with wireless communication in mind. We will consider alternatives to the standard approach from existing literature, specifically by studying the effect of using codes designed for communication to achieve secrecy over the wiretap channel. Additionally, we will evaluate the various metrics that may be used for quantifying the information gained by Eve. Our discussion will be backed up by both new and reproduced results from implementations of various simulations and calculations.

1.3 Overview

The remainder of this thesis is organized as follows. In Chapter 2, we introduce and discuss the prerequisite tools and knowledge needed to follow our work. In Chapter 3, we discuss relevant previous work, and how it relates to our thesis. Chapter 4 describes our methods and motivates our choices of tools such as models, implementations, codes, performance measures, etc. These tools are used to answer research questions, the results of which are presented and discussed in Chapter 5. In Chapter 6 we summarize our findings and their implications, and we discuss what directions of future research we believe may yield the most useful or interesting results.

Chapter 2

Background

2.1 Information theory

In the process of communicating information from A to B, there are sometimes inaccuracies introduced in the process. In day-to-day speech this can happen when someone is misheard, misinterpreted or otherwise misunderstood. In analog radio communication, inaccuracies in the received audio may come from cosmic background radiation, atmospheric phenomenon such as lightning, and from the electronic circuits in the radio device [3]. These inaccuracies are generally referred to as *noise*. In digital communication noise can also occur, and will manifest itself in the form of bit flips; either a transmitted "1" is received as a "0", or vice versa.

In his 1948 paper *A Mathematical Theory of Communication* [4], Claude Shannon introduced a set of new tools to study noisy communication channels and created the field of information theory. The main focus of this field has traditionally been on achieving reliable communication *despite* communicating over noisy channels. However, in this thesis we will focus on using the noise to our advantage for secure communication. More on that in Section 2.5. In this section, we will introduce some of the important concepts and results in information theory, in order to use them in our following work.

2.1.1 Shannon entropy

In [4], *entropy* is defined as a measure of uncertainty, or equivalently, information. When Shannon was deciding what to name his new measure of information, the famous mathematician John von Neumann reportedly told him [5]

> You should call it entropy, for two reasons. In the first place your uncertainty function has been used in statistical mechanics under that name, so it already has a name. In the second place, and more important, no one knows what entropy really is, so in a debate you will always have the advantage.

Intuitively, entropy represents our uncertainty about some random event. As an example we imagine a coin flip. The outcome of the coin flip is either heads or tails, with equal probability. We only have two possible outcomes, which is analogous to 1 bit which has the value 0 or 1. Given that heads and tails are equally probable, we have no information about the outcome. We can therefore say that we have 1 bit of uncertainty, or equivalently, 1 bit of entropy.

Now, imagine that we have some magical coin, where instead of being equally probable, heads has a 99% chance of occurring, and tails only has a 1% chance. We know that we have less than 1 bit of uncertainty, as we will correctly predict the outcome in most cases, but we also know that we still have *some* uncertainty.

Shannon's definition of entropy allows us to quantify the amount of uncertainty in a given situation. Entropy is typically denoted H, and is a function of some stochastic random variable X taking values from $\mathcal{X}$ with probability mass function $p(x) = P(X = x)$. The Shannon entropy of X is defined as

$$H(X) \triangleq -\sum_{x\in\mathcal{X}} p(x)\log_2 p(x). \tag{2.1}$$

In the special case where X is a Bernoulli distribution with only two outcomes of probabilities p and $1-p$, we use the term *binary entropy*, and the notation $H(X) = h_b(p)$, where

$$h_b(p) \triangleq -p\log_2 p - (1-p)\log_2(1-p). \tag{2.2}$$

For reference and intuition, a plot of the binary entropy function is shown in Fig. 2.1.

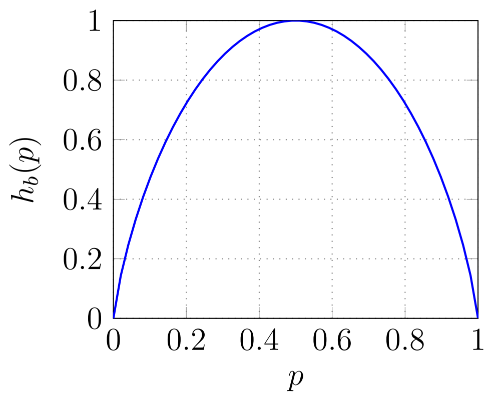

Figure 2.1: Binary entropy function $h_b(p)$

Given two random variables X and Y, Shannon defined the *conditional entropy* of X given Y as

$$H(X|Y) \triangleq -\sum_{x,y\in\mathcal{X},\mathcal{Y}} p(x,y)\log_2 p(x|y). \tag{2.3}$$

The *mutual information* $I(X;Y)$ of X and Y is defined as

$$I(X;Y) \triangleq H(X) - H(X|Y) = H(Y) - H(Y|X). \tag{2.4}$$

The conditional entropy and mutual information are useful measures in a communication scenario. Given some transmitted message X and a received message Y, $H(X|Y)$ represents the receiver's remaining uncertainty about the transmitted message after having observed the noisy received version Y. Similarly, $I(X;Y)$ represents how much information is shared between the sender and the receiver.

2.1.2 Rényi entropy

Rényi generalized the concept of entropy with the following definition [6]

$$H_\alpha(X) = \frac{1}{1-\alpha}\log\left(\sum_{i=1}^{n} p_i^\alpha\right). \tag{2.5}$$

Where α is called the *order*, and $\lim_{\alpha\to 1} H_\alpha(X)$ corresponds to Shannon entropy. When $\alpha \to \infty$, $H_\alpha(X)$ entropy converges to

$$H_\infty(X) = -\log \max_i p_i. \tag{2.6}$$

This is called the *min-entropy.* Note the relationship between min-entropy and Shannon entropy

$$H_\infty(X) \leq H(X). \tag{2.7}$$

Unless otherwise specified, the term *Rényi entropy* on it's own is sometimes used to refer to $H_2(X)$, and is sometimes denoted $R(X)$ [7, 8].

As opposed to Shannon entropy, there is no standard way of defining the conditional Réniy entropy $H_\alpha(X|Y)$ [9]. However, it is argued in [9] that Arimoto's definition [10] is the most appropriate, as it satisfies more desirable properties than other proposed definitions. For the case of conditional min-entropy, Arimoto's definition gives

$$H_\infty(X|Y) \triangleq -\log \sum_{y\in\mathcal{Y}} p(y) \max_{x\in\mathcal{X}} p(x|y). \tag{2.8}$$

2.1.3 Channel models

In order to study how noise affects communication, we need some mathematical models that approximate real-world noisy communication. These models are described as channels, and choosing a channel to study can often be a trade-off between simplicity and realism. Simple channels may be easier to study, but may not be very accurate descriptions of real-world communication.

A channel model is described by a set of input symbols that may be transmitted over the channel, and a set of possible outputs. A channel typically has some parameter that describes the channel quality, from which a probability distribution on the output can be calculated. "Good" channels have low probabilities of error in the output, and conversely "bad" channels have a higher probability of error.

We will look at two types of channel models, *discrete* and *continuous*. Discrete channels have outputs from a finite set of values, while continuous channels have outputs on a continuous spectrum. The channels we will look at all have the characteristic that errors occur independently in each transmitted symbol, these are known as *memoryless* channels.

Binary Symmetric Channel

The binary symmetric channel (BSC), is a simple discrete memoryless channel with binary input and output. The channel quality is characterized by a transition probability p. Each input bit will retain its value with probability $1-p$ and will flip with probability p. The BSC is illustrated in Fig. 2.2 with input on the left and output on the right.

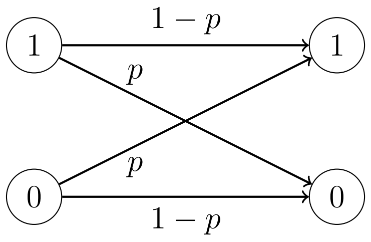

Figure 2.2: The binary symmetric channel with transition probability p

Binary Erasure Channel

The binary erasure channel (BEC) is a discrete memoryless channel with binary input. The channel either outputs the input symbol, or it outputs an *erasure* denoted "?", which gives no information about the transmitted symbol. The probability of erasure is denoted ϵ, and is the same for both input symbols.

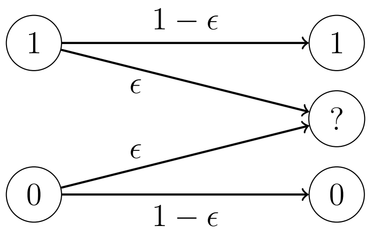

Figure 2.3: The binary erasure channel with erasure probability ϵ

Additive White Gaussian Noise

An Additive White Gaussian Noise (AWGN) channel, is a gaussian channel with discrete inputs and continuous outputs. The output follows a gaussian distribution with the input symbol as mean, and some given variance. A basic illustration of a binary input AWGN is shown in Fig. 2.4. The conditional probability density function $f(y|x)$ for the output y given an input symbol x is given as [11]

$$f(y|x) = \frac{1}{\sqrt{2\pi N}} e^{-\frac{(y-x)^2}{2N}} \tag{2.9}$$

where N is the noise variance. N is the parameter that defines the quality of the channel.

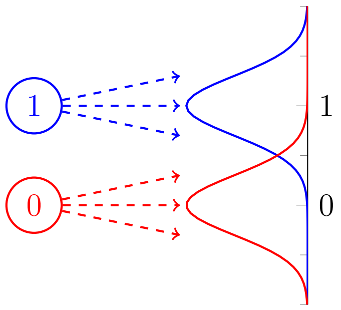

Figure 2.4: Illustration of an additive white gaussian noise channel with binary input and continuous output

2.1.4 Channel capacity and channel coding theorem

When describing communication across a channel, we have some set of messages $\mathcal{W} = \{0,1\}^k$, where $\{0,1\}^k$ is the set of all binary sequences of length k. Each message is mapped to a codeword in $\mathcal{X}^n$, where $\mathcal{X}$ is the input alphabet of the channel. An (n,k) *code*, is defined by a set of codewords in $\mathcal{X}^n$ and a mapping $f : \mathcal{W} \to \mathcal{X}^n$. The *rate* R of an (n,k) code is defined as

$$R = \frac{k}{n} \text{ bits per channel use.} \tag{2.10}$$

The *channel capacity* of a channel is defined as [4]

$$C \triangleq \sup_{p(x)} I(X;Y) \tag{2.11}$$

where X and Y are random variables representing the input and output of the channel, and $p(x)$ is the probability distribution of X.

Shannon proved the *channel coding theorem* [4], which states the following.

Theorem 2.1.1 (Channel coding theorem) *Given a channel of capacity C, there exists a code of rate R which achieves arbitrarily low error probability if and only if $R \leq C$.*

The channel capacity is therefore a fundamental limit that codes can be compared against. The capacities of the binary symmetric channel and the binary erasure channel can be shown to be

$$C_{\text{BSC}} = 1 - h_b(p) \tag{2.12}$$

and

$$C_{\text{BEC}} = 1 - \epsilon, \tag{2.13}$$

respectively [11].

2.1.5 Information-theoretic security

Information-theoretic security refers to security with information theoretical guarantees. We are specifically interested in secrecy, i.e. concealing the contents of a message from an eavesdropper. This situation can be described with the random variable X^n representing the encoded message, taking on values from $\{0,1\}^n$. And the random variable Z^n taking on values from $\mathcal{Z}^n$ where $\mathcal{Z}$ is the output alphabet of the channel. Z^n represents what is observed by the eavesdropper. We can describe the amount of information gained by the eavesdropper using the mutual information between the message and the observation by the eavesdropper. *Perfect secrecy* is said to be achieved when no information is gained, meaning

$$I(X^n; Z^n) = 0, \tag{2.14}$$

which by Eq. (2.4) implies

$$H(X^n|Z^n) = H(X^n). \tag{2.15}$$

Perfect secrecy is a very strong requirement. Shannon showed that it can be achieved using a one-time pad [4]. Using a one-time pad is not practical in most circumstances, as it requires a pre-shared key at least as long as the message, and the key cannot be reused. Wyner introduced a weaker requirement [1], referred to as *weak secrecy*, which can be formulated as [12]

$$\frac{1}{n} I(X^n; Z^n) \leq \epsilon, \text{ for some suitably small } \epsilon > 0. \tag{2.16}$$

2.1.6 Fano's inequality

Let X and Y be random variables, both taking on values from the same alphabet $\mathcal{X}$. If we let Y represent an estimate of X, we can define the error probability P_e as

$$P_e = P(X \neq Y). \tag{2.17}$$

Fano's inequality is then given as [11]

$$H(X|Y) \leq h_b(P_e) + P_e \log\big(|\mathcal{X}| - 1\big). \tag{2.18}$$

In addition to giving an upper bound on $H(X|Y)$ given P_e, it also provides us a lower bound on P_e, given the conditional entropy $H(X|Y)$.

2.2 Finite fields

Here, we provide a very brief definition of finite fields and $\mathbb{F}_2$ specifically, which originate from abstract algebra. Finite fields are required to describe the construction of some error correcting codes. An introduction to abstract algebra in the context of codes is given in [13]. For a more general introduction, see for example [14].

A finite field is a finite set of elements that can be added, subtracted, multiplied, and divided such that the result is always in the same finite set. The mathematical operations in a field $\mathbb{F}$ do not necessarily have the same definition as in everyday use, however they must meet the following requirements for all $a, b, c \in \mathbb{F}$ [14]:

1. If $a \in \mathbb{F}$ and $b \in \mathbb{F}$, then $a + b \in \mathbb{F}$.
2. (Associativity) $a + (b + c) = (a + b) + c$.
3. (Commutativity) $a + b = b + a$.
4. There exists an element $0_{\mathbb{F}} \in \mathbb{F}$ such that $a + 0_{\mathbb{F}} = a = 0_{\mathbb{F}} + a$. From now on denoted simply as 0.
5. The equation $a + x = 0$ has a solution in $\mathbb{F}$.
6. If $a, b \in \mathbb{F}$, then $ab \in \mathbb{F}$.
7. $a(bc) = (ab)c$.
8. $a(b + c) = ab + ac$ and $(a + b)c = ac + bc$.
9. $ab = ba$.
10. There exists an element $1_{\mathbb{F}} \in \mathbb{F}$ such that $a \cdot 1_{\mathbb{F}} = a = 1_{\mathbb{F}} \cdot a$ for all $a \in \mathbb{F}$. From now on denoted simply as 1.
11. Whenever $a, b \in \mathbb{F}$ and $ab = 0$, then $a = 0$ or $b = 0$.
12. For each $a \neq 0$ in $\mathbb{F}$, the equation $ax = 1$ has a solution in $\mathbb{F}$.

A field of q elements is typically denoted $\mathbb{F}_q$. In this work, we will use the field $\mathbb{F}_2$ consisting of the elements 0 and 1, because it conveniently corresponds to binary. Addition in $\mathbb{F}_2$ is done modulo 2, meaning that $1 + 1 = 0$. This is equivalent to the *exclusive or* (XOR) operation.

2.2.1 Vector spaces

A vector space is defined by a finite set of elements $\mathbb{V}$ as well as a finite field $\mathbb{F}$. We call the elements of $\mathbb{V}$ *vectors*, and we say that $\mathbb{V}$ forms a vector space over $\mathbb{F}$. Addition must be defined between vectors in $\mathbb{V}$, and must follow rules 1 through 5 from above. In addition, multiplication of any element in $\mathbb{F}$ with any vector in $\mathbb{V}$ must be defined. This is called *scalar multiplication*, and must meet the following requirements for all $a, a_1, a_2 \in \mathbb{F}$ and $\mathbf{v}, \mathbf{v}_1, \mathbf{v}_2 \in \mathbb{V}$ [14]:

1. $a(\mathbf{v}_1 + \mathbf{v}_2) = a\mathbf{v}_1 + a\mathbf{v}_2$.
2. $(a_1 + a_2)\mathbf{v} = a_1\mathbf{v} + a_2\mathbf{v}$.
3. $a_1(a_2\mathbf{v}) = (a_1\mathbf{v}a_2)\mathbf{v}$.
4. $1_{\mathbb{F}}\mathbf{v} = \mathbf{v}$.

Just as the motivation for defining $\mathbb{F}_2$ was the relation to binary, we define vector spaces here because they can represent sets of binary sequences. Specifically, let $\mathbb{V}$ be the set of ordered sequences of length n of elements in $\mathbb{F}_2$. An element $\mathbf{v} \in \mathbb{V}$ can be expressed as

$$\mathbf{v} = (a_0, a_1, ..., a_{n-2}, a_{n-1}), \text{ where } a_0, a_1, ..., a_{n-2}, a_{n-1} \in \mathbb{F}_2 \tag{2.19}$$

Addition is performed pairwise. If we define two vectors

$$\mathbf{v}_1 = (a_{1,0},\ a_{1,1},\ ...,\ a_{1,n-1}) \text{ and } \mathbf{v}_2 = (a_{2,0},\ a_{2,1},\ ...,\ a_{2,n-1}), \tag{2.20}$$

then

$$\mathbf{v}_1 + \mathbf{v}_2 = (a_{1,0} + a_{2,0},\ a_{1,1} + a_{2,1},\ ...,\ a_{1,n-1} + a_{2,n-1}). \tag{2.21}$$

And scalar multiplication is defined as

$$b\mathbf{v} = (ba_0, ba_1, ..., ba_{n-1}), \text{ where } b \in \mathbb{F}_2 \tag{2.22}$$

Ordered sequences of length n are sometimes called n-tuples. From the above, we can see that the set of binary n-tuples, denoted $\{0,1\}^n$, form a vector space over $\mathbb{F}_2$ with XOR (pairwise addition modulo 2) as its addition operator.

Another concept we will use is a subspace. Given a vector space $\mathbb{V}$ over $\mathbb{F}$, if some subset $\mathbb{S} \subset \mathbb{V}$ also forms a vector space over $\mathbb{F}$, we call $\mathbb{S}$ a subspace of $\mathbb{V}$.

2.3 Error correcting codes

In Section 2.1, we introduced the concept of using a *code* to map messages into codewords before transmitting them across a channel. The idea behind doing so is to introduce redundancies that allow the receiver to reconstruct the message despite the channel noise.

The receiver uses a *decoder* to produce an estimate $\hat{\mathbf{w}}$ of the message $\mathbf{w}$. Ideally, we want the decoder to find an estimate $\hat{W}$ which minimizes the probability of incorrect decoding, i.e.

$$\hat{\mathbf{w}} = \arg\min_{\mathbf{v}\in\mathcal{W}} P(\mathbf{w} \neq \mathbf{v}|\mathbf{r}) \tag{2.23}$$

where $\mathbf{r}$ is the received channel output. If we have a bijective mapping between messages and codewords, this is equivalent to the same process for an estimate $\hat{\mathbf{x}}$ of the codeword $\mathbf{x}$

$$\hat{\mathbf{x}} = \arg\min_{\mathbf{v}\in\mathcal{X}} P(\mathbf{x} \neq \mathbf{v}|\mathbf{r}) \tag{2.24}$$

This is further equivalent to maximizing the conditional probability of the codeword [13], giving us

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} P(\mathbf{x}|\mathbf{r}) = \arg\max_{\mathbf{x}} \frac{P(\mathbf{x})P(\mathbf{r}|\mathbf{x})}{P(\mathbf{r})} \tag{2.25}$$

A decoder that evaluates Eq. (2.25) is called a *maximum a posteriori* (MAP) decoder. A subtly different approach is to find $\hat{\mathbf{x}}$ such that

$$\hat{\mathbf{x}} = \arg\max_{\mathbf{x}} P(\mathbf{r}|\mathbf{x}) \tag{2.26}$$

This is called *maximum likelihood* (ML) decoding, and is equivalent to MAP decoding when $P(\mathbf{x})$ is constant, i.e. all codewords are equally probable. MAP and ML decoding of linear codes in general is NP-hard [15], however some codes do have ML and MAP decoding algorithms available.

We also defined the channel capacity and referenced the channel coding theorem, which states that it is possible to achieve error free communication across noisy channels using codes. However, when Shannon [4] proved this theorem, he did not provide practical ways to construct such codes. Instead, arguments consisting of random codewords as well as letting $n \to \infty$ were used. Random codes are difficult to decode because of their lack of structure, and very large or infinite block lengths are highly impractical.

In the years following the publication of [4], several more practical codes and accompanying decoding algorithms were constructed including Hamming codes [16], Reed-Muller codes [17, 18], cyclic codes [18–24], and convolutional codes [25–27]. As opposed to random codes, all these examples have some sort of *structure*. The structure of a code is some set of mathematical properties that allow us to reason about the code, and is often used to build efficient encoding and decoding procedures. These codes make it possible to detect and correct errors in the received channel output. However, the code rates needed to achieve small error probabilities don't come particularly close the channel capacity [13].

More recently efforts have been made to construct so-called "capacity achieving" codes, such as Turbo codes [28], Low Density Parity Check (LDPC) codes [29–32], and Polar codes [33].

So far we have given relatively generalized definitions of codes, however from this point forward we will limit ourselves to *binary* codes. For binary codes, messages have some length k and are represented as binary k-tuples, i.e. from the set $\{0,1\}^k$. Codewords of length n are represented as binary n-tuples.

2.3.1 Linear block codes

In order to define the specific codes we use in this work, it helps to first define the more general concept of binary *linear block codes.* A block code is a code with constant length messages and codewords. In an (n,k) block code, a message or *information vector* with k symbols is encoded into a codeword of n symbols. We have 2^k possible messages and corresponding codewords.

One structure we can give a code is *linearity*, which can be defined as follows [13].

Definition 2.3.1 *A block code of length n and 2^k codewords is called a linear (n,k) code if and only if its 2^k codewords form a k-dimensional subspace of the vector space of all the n-tuples over the field $\mathbb{F}_2$.*

It can also be shown that a binary block code is linear if and only if the sum of any two codewords is also a codeword, i.e.

$$(\mathbf{c}_1 + \mathbf{c}_2) \in \mathcal{C}, \text{ for all } \mathbf{c}_1, \mathbf{c}_2 \in \mathcal{C}, \tag{2.27}$$

where $\mathcal{C}$ is the set of codewords, sometimes called the codebook or simply the code.

Linear block codes can be defined by their generator matrix $\mathbf{G}$, or by their parity check matrix $\mathbf{H}$. The generator matrix $\mathbf{G}$ is a $k \times n$ binary matrix, where the rows are k linearly independent codewords. All codewords in the code may be generated by linear combinations of the rows of $\mathbf{G}$. Because we have 2^k possible linear combinations of the rows, there are 2^k codewords. The generator matrix can also be used as an encoder. Given a generator matrix

$$\mathbf{G} = \begin{bmatrix} g_{00} & g_{01} & \cdots & g_{0,n-1} \\ g_{10} & g_{11} & \cdots & g_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1} \end{bmatrix}, \tag{2.28}$$

and some information vector $\mathbf{u} = (u_0, u_1, ..., u_{k-1})$, we can encode the information vector into a codeword $\mathbf{c}$ by calculating

$$\mathbf{c} = \mathbf{uG} = (u_0, u_1, ..., u_{k-1}) \begin{bmatrix} g_{00} & g_{01} & \cdots & g_{0,n-1} \\ g_{10} & g_{11} & \cdots & g_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ g_{k-1,0} & g_{k-1,1} & \cdots & g_{k-1,n-1} \end{bmatrix}. \tag{2.29}$$

The parity check matrix $\mathbf{H}$ is a $(n-k) \times n$ matrix

$$\mathbf{H} = \begin{bmatrix} h_{00} & h_{01} & \dots & h_{0,n-1} \\ h_{10} & h_{11} & \dots & h_{1,n-1} \\ \vdots & \vdots & \ddots & \vdots \\ h_{n-k-1,0} & h_{n-k-1,1} & \dots & h_{n-k-1,n-1} \end{bmatrix}, \tag{2.30}$$

constructed such that for any codeword $\mathbf{c} \in \mathcal{C}$, we have $\mathbf{cH}^T = \mathbf{0}$. For any received vector $\mathbf{r}$ of length n, we can compute $\mathbf{rH}^T = \mathbf{s}$, where $\mathbf{s}$ is called the *syndrome*. Calculating the syndrome allows us to check for errors in the received vector.

2.3.2 Cyclic codes

A cyclic shift is an operation where all elements of a vector is moved between 0 and $n-1$ steps to the right, placing any overflowing elements back at the beginning. We denote an i-step cyclic shift of a vector $\mathbf{v}$ as $\mathbf{v}^{(i)}$. As an illustrative example, consider the vector $\mathbf{v} = (v_1, v_2, v_3, v_4)$ and its cyclic shifts

$$\begin{aligned} \mathbf{v}^{(0)} &= (v_1, v_2, v_3, v_4) \\ \mathbf{v}^{(1)} &= (v_4, v_1, v_2, v_3) \\ \mathbf{v}^{(2)} &= (v_3, v_4, v_1, v_2) \\ \mathbf{v}^{(3)} &= (v_2, v_3, v_4, v_1) \end{aligned} \tag{2.31}$$

A *cyclic code* is a linear block code where all cyclic shifts of codewords are also codewords. In order to show some of the properties of cyclic codes, it is useful to view codewords and messages as polynomials. Given the codeword n-tuple $\mathbf{c} = (c_0, c_1, ..., c_{n-2}, c_{n-1})$, we have the corresponding polynomial

$$c(X) = c_0 + c_1X + c_2X^2 + ... + c_{n-2}X^{n-2} + c_{n-1}X^{n-1}. \tag{2.32}$$

Of all codewords in an (n, k) cyclic code, one and only one codeword polynomial will have degree $n-k$ [13]. We call this polynomial the *generator polynomial*, and denote it $g(X)$. It can be shown that all codeword polynomials are multiples of $g(X)$ [13], and therefore, encoding an information polynomial $u(X)$ is done by multiplying it with the generator, i.e.

$$c(X) = u(X)g(X). \tag{2.33}$$

Where the coefficients $u_0, u_1, ..., u_{k-1}$ of $u(X)$ are the message bits.

Perhaps the most well-known class of cyclic codes are Bose, Chaudhuri, and Hocquenghem (BCH) codes, which were discovered independently in [20] and [21]. BCH codes can be constructed from the integer paramters $m \geq 3$ and $t < 2^{m-1}$. They have a block length of $n = 2^m - 1$, and k must be chosen such that $n - k \leq mt$. Such a code is called a *t-error correcting BCH code*, because there exists a decoder that is guaranteed to detect and correct any error pattern with up to t errors [13].

2.3.3 Convolutional codes

So far we have discussed block codes exclusively, however there exists classes of codes that do not necessarily have constant length messages and codewords. One of these classes is *convolutional codes*. Convolutional codes have some rate k/n, and the encoder will continuously produce n output symbols for every k input symbols. As opposed to block codes, the n output symbols depend not only on the k input symbols, but also on the state of the encoder. In our work we only utilize convolutional codes with $k = 1$, however other values for k are also common to get higher code rates. In the rest of this section it can be assumed that $k = 1$.

A convolutional encoder has m state bits, which means it has 2^m possible states. At a given time, each of the n output bits are determined by some linear function which takes the message bit as well as the m state bits as inputs. As the current state is determined by the m previous inputs, the state may be considered *memory*.

Consider a rate $1/n$ convolutional code with m state bits. The n output bits are determined by n binary *generator sequences* $\mathbf{g}^{(i)} = (g_0^{(i)}, g_1^{(i)}, ..., g_m^{(i)}), 0 \leq i < n$. Given an input bit u and m state bits $s_1, ..., s_m$, the n-bit output sequence $\mathbf{c}$ is then determined by

$$\mathbf{c} = (c^{(0)}, c^{(1)}, ..., c^{(n-1)}), \tag{2.34}$$

where

$$c^{(i)} = ug_0^{(i)} + s_1 g_1^{(i)} + ... + s_m g_m^{(i)}. \tag{2.35}$$

A convolutional code may be represented by its encoder diagram. See Fig. 2.5 for the diagram representing the rate 1/2 convolutional code with generator sequences $\mathbf{g}^{(0)} = (1, 1, 1)$ and $\mathbf{g}^{(1)} = (1, 0, 1)$.

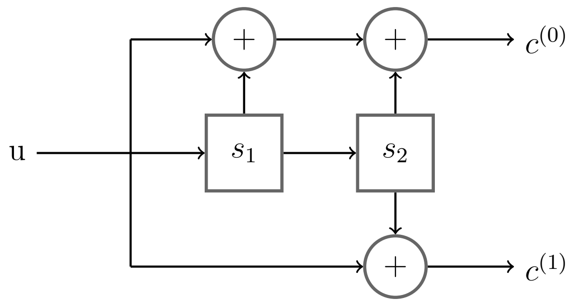

Figure 2.5: Convolutional encoder.

Another way to describe a convolutional code is using a *trellis*. A trellis is a directed graph where the nodes represent the encoder state, and the edges represent some input. Encoding a sequence of bits can be done by traversing the trellis and choosing which edge to follow based on the input bit. Edges also have an associated output label, so the output sequence can be read from the traversed path. The trellis is important because its structure is exploited by the ML and MAP decoding algorithms known as the

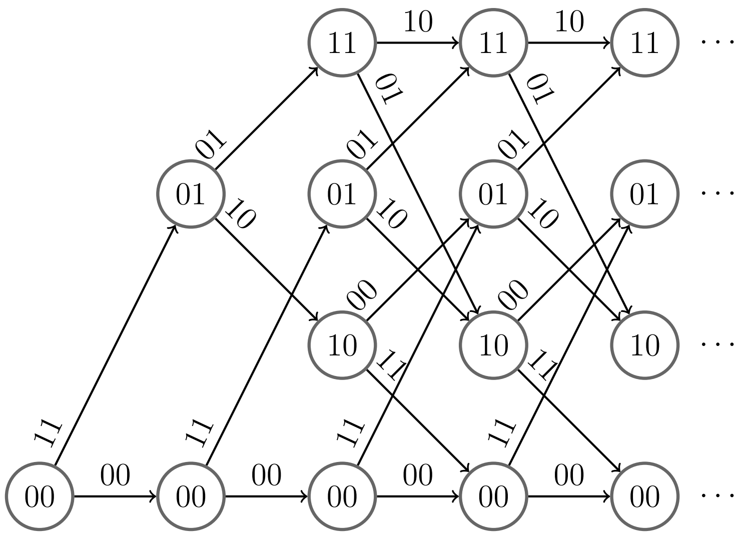

Figure 2.6: Trellis for a rate 1/2 convolutional code with $m = 2$.

Viterbi algorithm [26] and the BCJR algorithm [27], respectively. The complexity of these algorithms depend on the number of states (2^m) as well as the message length.

While we have presented convolutional codes as continuous codes which encode one bit at a time, in practice we tend to limit the length of the sequences we encode, because practical decoding depends on it. One way to have a finite block length for convolutional codes is to append m zero bits to the end of the message, which ensures that the encoder returns to the zero state. This does slightly reduce the code rate from $\frac{1}{n}$ to $\frac{k'}{(k'+m)n}$, where k' is the message length. A technique to eliminate this reduction in the code rate is *tail-biting* convolutional codes. See [13] for more about tail-biting codes, as we do not discuss them further.

Using a finite block length turns a convolutional code into a linear block code [13]. It is also possible to construct generator and parity check matrices for a finite convolutional code.

2.3.4 Turbo codes

Invented in 1993 [28], *turbo codes* are designed to achieve small error rates while maintaining a code rate close to Shannon's channel capacity. Turbo encoders consist of two or more convolutional encoders and an *interleaver*. The convolutional codes that are part of the turbo code are called *constituent codes*, and their encoders may be arranged in serial, in parallel, or in some hybrid arrangement. The interleaver performs a pseudo-random permutation on the input bits before it is passed to the encoders, except for the first encoder. An illustration of a simple turbo encoder can be seen in Fig. 2.7, where u is the input bit, π represents the interleaver, and $p^{(1)}$ and $p^{(2)}$ are the parity bits from each constituent encoder. For each input bit there are 3 output bits, meaning the rate of the encoder is 1/3.

The main innovation of turbo codes is the iterative decoding process. As we have

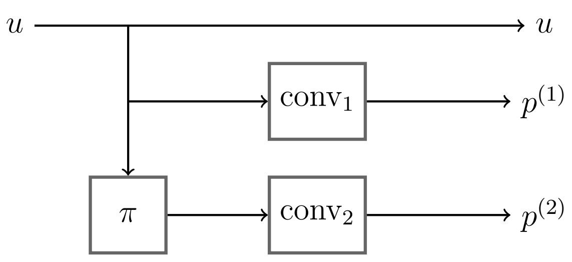

Figure 2.7: Rate 1/3 turbo encoder.

discussed, the individual convolutional codes have a MAP decoding algorithm available. In the case of a turbo code with two constituent codes, this algorithm is used in alternation between the two convolutional codes along with a technique called *belief propagation*. Without describing the process in detail here, the main concept is that probability estimates for each bit is passed from one decoder to the other, with the output of the two eventually converging. If they do not converge, decoding error is very likely. The number of iterations is a parameter of the decoder, but can typically fall in the range of 10-20 [13].

2.4 Wireless communication

2.4.1 Modulation

Wireless communication between digital devices normally happens through the transmission of radio waves. Different wireless protocols have specific designated *carrier* frequencies, and some bandwidth on each side of this center frequency. Carriers in the 2400-2500 MHz frequency range are, for example, common. In order to send digital information using this carrier, we must convert bit patterns (abstracted as "symbols") into waveforms through the process of *modulation*. There are several modulation techniques that encode symbols as waves by changing some property of a waveform. Frequency shift keying (FSK) encodes symbols into different frequencies and amplitude shift keying (ASK) encodes symbols as different amplitudes. Phase shift keying (PSK) changes the phase of a waveform. See Fig. 2.8 for an illustration of these techniques.

A common and simple modulation we will utilize in our work is Binary PSK (BPSK). As the name suggests, we have two symbols: 0 and 1. 1 is encoded with a 0° phase, and 0 is encoded with a 180° phase.

Modern communication systems will often combine ASK and PSK into Quadrature Amplitude Modulation (QAM). QAM allows us to represent symbols as points on a two-dimensional plane, where the angle from the origin represents the phase, and the distance from the origin represents the amplitude. These modulation techniques are typically named n-QAM, where n is the number of symbols. 2-QAM corresponds to BPSK, with the points $(-1, 0)$ and $(1, 0)$ representing 0 and 1, respectively.

A wireless receiver will typically measure the phase and the amplitude of the signal. Demodulating in an n-QAM system is done by choosing the symbol with the smallest

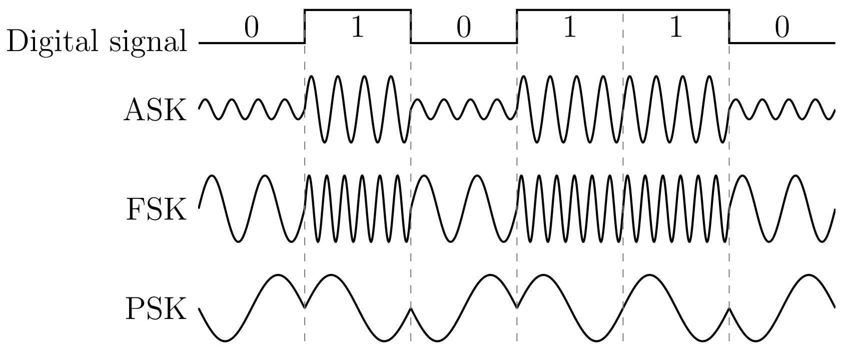

Figure 2.8: Modulation techniques.

euclidean distance to the received value.

2.4.2 Quantifying signal quality

The quality of a wireless signal is typically denoted by a signal-to-noise ratio (SNR).

$$\text{SNR} = \frac{\text{Signal power}}{\text{Noise power}} \tag{2.36}$$

The SNR is often expressed in decibels (dB).

$$\text{SNR}_{\text{dB}} = 10 \log_{10}(\text{SNR}) \ \text{dB} \tag{2.37}$$

Decibels are a logarithmic scale, which may require some adjustment of intuition. A doubling of the SNR constitutes approximately a 3 dB increase in SNR_{dB}, and an SNR of 1 is equal to an SNR_{dB} of 0. A negative SNR_{dB} represents an SNR between 0 and 1.

2.4.3 Signal strength over distance

In an idealized model of wireless radio communication, we have a transmitting antenna at a point in an infinite empty space in a vacuum. Using an omnidirectional antenna, the transmitted signal will propagate in a sphere centered on the antenna. The signal strength of the receiver can then be determined by what portion of the sphere is covered by the receiver's antenna.

Because the surface area of a sphere increases with the square of the radius ($4\pi r^2$), the signal strength S of the receiver is proportional to the inverse square of the distance d:

$$S \propto \frac{1}{d^2} \tag{2.38}$$

Given a signal strength S_1 at distance d_1, we have the signal strength S_2 at distance d_2 given by

$$S_2 = S_1 \left(\frac{d_1}{d_2} \right)^2 . \tag{2.39}$$

In Eq. (2.38) and Eq. (2.39) the idealized scenario described above is assumed. In reality, there are several factors that may affect the propagation of radio waves, perhaps most importantly the physical geography and buildings in the surrounding area. Such scenarios can be modeled with a modified version of the inverse square law [34]

$$S \propto \frac{1}{d^\gamma} \tag{2.40}$$

where γ is known as the *path loss exponent.* In a vacuum surrounded by infinite empty space, we have $\gamma = 2$. In other scenarios, reflections may amplify the signal leading to $\gamma < 2$, or the signal may be blocked or attenuated leading to $\gamma > 2$.

Given the SNR expressed in decibel at two distances d_1 and d_2, we then have [34]

$$\mathrm{SNR}_2 = \mathrm{SNR}_1 - 10\gamma \log_{10} \left(\frac{d_2}{d_1} \right) . \tag{2.41}$$

In [34], a table of typical values for the path loss exponent γ is given, based on measurements in the field. We list these values in Table 2.1.

Environment	γ
Free space	2
Urban area cellular radio	2.7 to 3.5
Shadowed urban cellular radio	3 to 5
In building line-of-sight	1.6 to 1.8
Obstructed in building	4 to 6
Obstructed in factories	2 to 3

Table 2.1: Path loss exponent γ in various environments [34].

2.5 The wiretap channel

Introduced in 1975 by Wyner [1], the wiretap channel is a channel model consisting of a sender we will call Alice, a recipient Bob, and a wiretapper Eve. Bob receives data through a noisy channel we refer to as the main channel. Eve receives data through the main channel as well as a second noisy channel (see Fig. 2.9). The real-world analog to this model could for example be case of wireless communication where Eve is further away from Alice than Bob is.

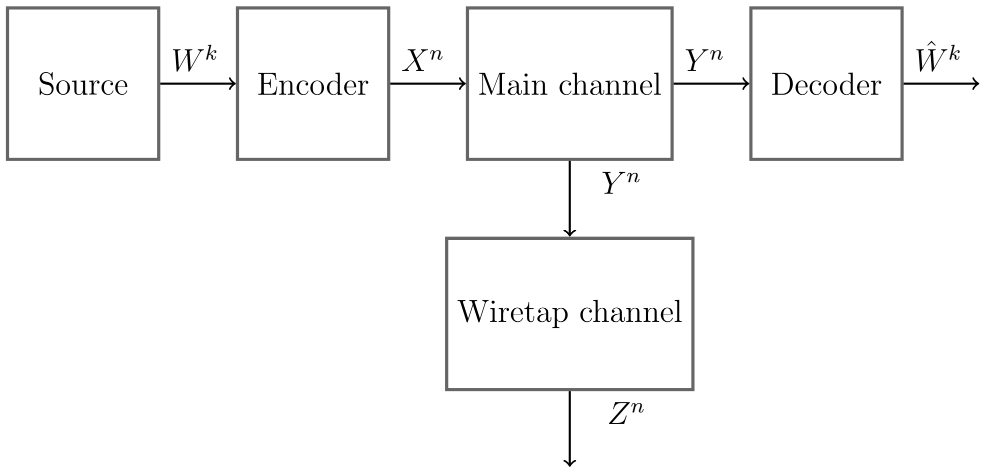

Figure 2.9: Wyner's general case wiretap channel

Wyner defines the equivocation rate Δ of Eve as her uncertainty about the source symbols given the received symbols.

$$\Delta \triangleq \frac{1}{k} H(W^k | Z^n) \tag{2.42}$$

To achieve secure and reliable communication in this model, we want to maximize the equivocation rate of Eve, while minimizing the probability of decoding error Bob as well as maximizing the transmission rate k/n. Perfect secrecy is achieved if the equivocation Δ is equal to the unconditional source entropy $H(W^k)$. The secrecy capacity C_s of a wiretap channel is defined as the maximum achievable transmission rate R such that $\Delta = H(W^k)$.

Wyner shows that if $C_m > C_{mw}$ where C_m and C_{mw} are the capacities of the main channel and the wiretap channel, respectively, there exists a secrecy capacity C_s which satisfies

$$0 < C_m - C_{mw} \leq C_s \leq C_m \tag{2.43}$$

2.6 Coding for the wiretap channel

If our goal is to achieve secrecy across a wiretap channel, we want to use a code that increases the confusion of Eve. More precisely, given a message W encoded as a codeword X^n, and an observed channel output Z^n by Eve, we want to minimize the mutual information $I(W; Z^n)$. We have that [12]

$$\begin{aligned} I(W; Z^n) &= I(X^n; Z^n) - I(X^n; Z^n | W) \\ &= I(X^n; Z^n) + H(X^n | Z^n, W) - H(X^n | W). \end{aligned} \tag{2.44}$$

From this it is clear that if $H(X^n|W)$ is zero, i.e. there is a bijective mapping between messages and codewords, we get $I(W|Z^n) = I(X^n; Z^n)$, which grows linearly with n [12].

There is now a motivation to make $H(X^n|W)$ non-zero, so we can control the growth of $I(W; Z^n)$. A non-zero $H(X^n|W)$ can be achieved by mapping a message randomly into one of multiple valid codewords, so that encoding is no longer deterministic. This is the primary motivation for *syndrome coding.*

2.6.1 Syndrome coding

In an (n, k) linear code with an $(n-k) \times n$ parity check matrix $\mathbf{H}$, the *syndrome* $\mathbf{s}$ of a length-n vector $\mathbf{r}$ is defined as [13]

$$\mathbf{s} = \mathbf{r} \cdot \mathbf{H}^T. \tag{2.45}$$

Syndrome coding, sometimes referred to as coset coding, was originally described by Wyner in [1]. The idea of syndrome coding is to transmit a channel vector whose syndrome is the information vector. An (n, k) linear code $\mathcal{C}$ will have 2^{n-k} cosets $\mathcal{C}_0 = \mathcal{C}, \mathcal{C}_1, \mathcal{C}_2, ..., \mathcal{C}_{n-k-1}$. We can define a coset $\mathcal{C}_i$ as

$$\mathcal{C}_i = \{\mathbf{v} \mid \mathbf{v}\mathbf{H}^T = \mathbf{b}_i,\ \mathbf{v} \in \{0,1\}^n\}, \tag{2.46}$$

where $\mathbf{b}_i$ is the n-bit binary representation of i. As is clear from this definition, all members of a coset will produce the same syndrome. For example, for all $\mathbf{x} \in \mathcal{C}_0$, $\mathbf{x}\mathbf{H}^T = \mathbf{0}$. Thus, encoding information using coset coding can be described as choosing a random member of the coset which produces the syndrome equal to the desired information vector. The decoding process is simply calculating the syndrome of the received vector $\mathbf{r}$.

In practice, encoding boils down to choosing a solution to Eq. (2.45), where $\mathbf{s}$ is the information vector and $\mathbf{r}$ is unknown, and calculating the codeword $\mathbf{x}$ as follows

$$\mathbf{x} = \mathbf{c} + \mathbf{r}, \tag{2.47}$$

where $\mathbf{c}$ is chosen uniformly at random from the code $\mathcal{C}$.

Chapter 3

Previous work

3.1 Privacy amplification

Privacy amplification was introduced in [7] as the specific scenario where Alice and Bob is exchanging a cryptographic key over a wiretap channel. A cryptographic key is in this context is simply a bit vector of a set length. This differs from a traditional communication scenario in that Alice and Bob do not care what the exact content of the key is, only that Eve has as little information about the key as possible.

[8] provides a discussion of syndrome coding for privacy amplification, in the situation where Bob has a noiseless channel. They argue that the min-entropy is a more appropriate measure than Shannon entropy for quantifying Eve's information, and furthermore provides a probabilistic lower bound on the min-entropy of Eve given a randomly chosen syndrome function.

The argument for using min-entropy is based on certain probability distributions where the Shannon entropy $H(X)$ and Rényi entropy $H_2(X)$ may be high, while there still being a relatively high probability for some single value x. To illustrate this, take the eavesdroppers Eve's probability distribution of a 128 bit message $P(W)$. Let us assume some message $\mathbf{a}$ has probability $P(W = \mathbf{a}) = 1/1000$, and that $P(W = \mathbf{w}) = (1 - 1/1000)/(2^{128} - 1)$ for $\mathbf{w} \neq \mathbf{a}$. In this situation we have the Shannon entropy $H(W) \approx 127.88$, which implies that Eve has only about 0.1 bits of information about the message. However, if Eve tries to guess the message they will be correct with probability 1/1000, which may be unacceptable for the system designer. On the other hand, we have the min-entropy $H_\infty(W) \approx 9.97$, which more accurately captures Eve's capabilities in this situation.

We see that the min-entropy is a stronger requirement on security than Shannon entropy. The authors also prove the following probabilistic bound on the min-entropy [8].

> Let X be a random binary vector of length n with a fixed probability distribution, with min-entropy $H_\infty(X) = r$. Let $\mathbf{H}$ be a uniformly randomly chosen $r \times n$ binary matrix, and let σ be the associated syndrome function. The probability, over the choice of $\mathbf{H}$, that $H_\infty(\sigma(X)) < r - \log_2(1 + 2^m)$ is not more than $2^{-3m^2/4+m+r}$.

Given that the min-entropy of the codeword, or equivalently the channel noise, is equal to the message length, this allows us to give a lower bound on the min-entropy of the message with a desired level of probability by adjusting m.

3.2 Calculating equivocation

In [35], an algorithm for designing syndrome codes for the wiretap channel is presented. The focus is on constructing codes which give a large equivocation for Eve over the binary symmetric channel. As part of this algorithm, they present a method for calculating the equivocation of a given syndrome code over the BSC with a given error probability. This method involves evaluating a sum over 2^m elements, where $m = n - k$ is the number of message bits. 2^m grows very fast with m. Consequently, the method is limited to small values of m, and is used to construct syndrome codes with $m = 15$.

In [36], Pfister et. al. discuss the equivocation of Eve over a binary erasure wiretap channel. They show that the specific equivocation $H(W|Z^n = \mathbf{z}^n)$ can be expressed as

$$H(W|Z^n = \mathbf{z}^n) = k - \mu + \text{rank}(\mathbf{G}_\mu), \tag{3.1}$$

where W is the message, $\mathbf{z}^n$ is Eve's observed vector, μ is the number of erasures, and $\mathbf{G}_\mu$ is the generator matrix of the underlying code with the columns at the erasure positions removed. This could be used for direct calculation of the equivocation with a weighted average over all possible received vectors $\mathbf{z}^n$. This has a complexity of 3^n due to the alphabet being $\{0, 1, ?\}$ with ? representing an erasure. This complexity is even more limiting than the method for the BSC in [35]. However, because they are able to calculate the specific equivocation for a received vector, they can use a Monte Carlo simulation to estimate the equivocation for longer block lengths. We will describe Monte Carlo estimation of equivocation more in detail in Chapter 4.

3.3 Convolutional encoding for the wiretap channel

Verriest and Hellman [37] proposed the use of a rate 1 convolutional code for the wiretap channel where Bob has a noiseless channel, and Eve has a binary symmetric channel. They show that when Eve has a small error probability, the number of memory bits alone dictates her entropy as n tends to infinity.

This approach, while somewhat novel compared to other wiretap coding schemes, has limited practical utility on its own due to Bob's noiseless channel and n tending to infinity. We mention it regardless due to our own use of convolutional codes on the wiretap channel.

3.4 Alternatives to syndrome coding

There has been some limited work done on coding for the wiretap channel with other methods than syndrome coding. In [38], Klinc, et. al. proposes a coding scheme based on LDPC codes, where the message bits are punctured in the encoding process, leaving only the parity bits in the codeword. A code where the information bits are not present in the codeword is called a *nonsystematic* code. Due to the difficulty in measuring equivocation, they use the bit error rate as an analog. This means they make no claim of achieving information-theoretical security, but instead use the term *physical-layer security.* Physical-layer security is said to be achieved when the BER of Eve is above a given confusion threshold, and the BER of Bob is below a given reliability threshold. The performance of a code is measured by the *security gap*, which is the difference in SNR required between Bob and Eve to achieve the given thresholds. They also conjecture that nonsystematic codes of other classes than LDPC may exhibit similar BER performance on the wiretap channel.

3.5 Real-world wireless wiretap communication

In [39], a method for constructing a secure wireless communications channel using wiretap codes is presented. The authors use measurements of the SNR of a wireless signal at varying locations within an office building as a basis for their work. The intended receiver Bob is assumed to be in a specific office, and the eavesdropper Eve is assumed to be in one of several other offices. Using SNR measurements from all these locations along with a channel model, they choose wiretap codes that maximize throughput to the Bob while minimizing the information gained by Eve. They model the channel as an erasure channel, specifically as follows:

$$y = \begin{cases} x, & \text{if SNR} \geq \tau \\ ?, & \text{if SNR} < \tau, \end{cases} \tag{3.2}$$

where y is the received symbol, x is the transmitted symbol, and ? represents an erasure. They consider a symbol to be an erasure if the signal-to-noise ratio at the receiver, SNR, is below a threshold τ. This model is justified by pointing out that many practical communications systems behave very similar to an erasure channel, where we either observe a very accurate representation of the transmitted data, or the transmitted data is discarded entirely.

The authors do not conduct any experiments with coded communication over the wireless channel, but instead use measurements of the SNR at each location along with the model presented in Eq. (3.2) to calculate secrecy capacity. They then find the Reed-Muller (RM) code that maximizes throughput with zero mutual information between the transmitter and the wiretapper, in their specific scenario.

Changing the value of τ in the model is the equivalent of changing the transmission power an equal negative amount. I.e. increasing τ by 1 dB has the same effect as reducing the transmission power by 1 dB. The value for τ is chosen to maximize the secure throughput.

Their erasure model may reflect typical communication scenarios, but it is important to note that Eve might not behave like a typical receiver. Even if decoding does not yield the correct result, Eve might gain some information that could, for example, help reduce the search space of a brute-force approach.

The process of measuring SNR at physical locations will be more accurate than a path loss model as presented in Section 2.4, however it is very resource intensive, and would have to be performed for each location the transmitter may be positioned. Such measurements could also be utilized to inform the choice of the parameter γ of a path loss model.

Chapter 4

Methodology

When designing a communication system for the wiretap channel, we want to achieve both secrecy *and* reliability. Secrecy may be defined by some threshold on Eve's equivocation, frame error rate, or bit error rate. We will discuss the merits of each of these measures. Reliability can be defined by some threshold on Bob's frame error rate. Intuitively, improving reliability will weaken secrecy, and improving secrecy will weaken reliability. We want to explore which codes provide a good trade-off between secrecy and reliability.

In this section, we present the set of codes we evaluate, as well as our methods for evaluation. We will especially focus on the merits of syndrome coding versus more traditional codes used for reliable communication.

4.1 Channel model

We use a slightly modified version of Wyner's model in Fig. 2.9. Instead of the wiretap channel working on the output of the main channel, it is a completely independent channel taking the encoder output as its input. See Fig. 4.1 for an illustration.

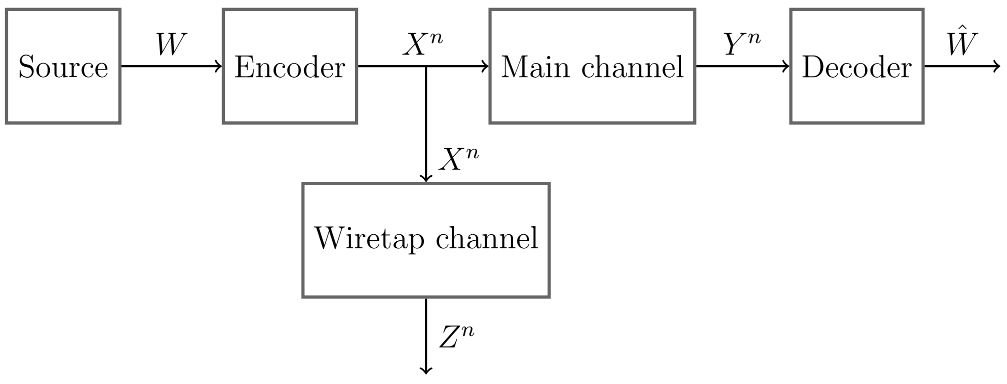

Figure 4.1: A modified wiretap channel.

We always use the same channel type for the main and wiretap channel, only with differing signal quality. We use the binary symmetric channel due to its simplicity, and the AWGN channel due to its better approximation of wireless communication.

4.2 Simulation procedure

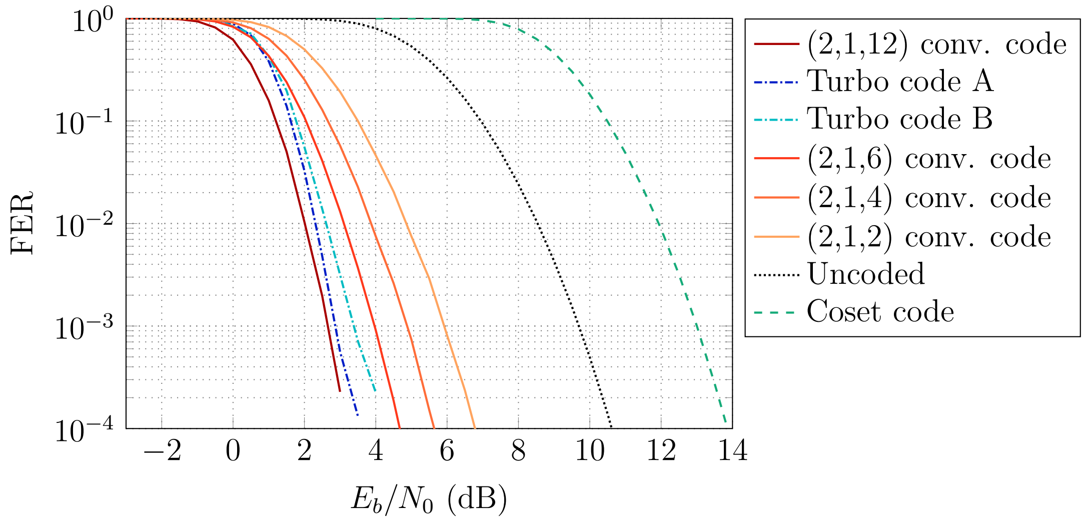

Figure 4.2: Frame error rate of codes with message length 128 over the AWGN channel.

In order to measure the performance of the above codes, we run simulations of encoding, channel transmission, and decoding across the AWGN and BSC channels, with varying noise levels. The source code for our implementations and simulations is available at [40].

From our simulations, there are two primary measures we are interested in, the *frame error rate* (FER) and the *bit error rate* (BER). They are defined as follows

$$\mathrm{FER} \triangleq \frac{\text{Incorrectly decoded codewords}}{\text{Total simulations}}, \tag{4.1}$$

$$\mathrm{BER} \triangleq \frac{\text{Incorrectly decoded bits}}{k(\text{Total simulations})}, \tag{4.2}$$

where k is the message length in bits. From a reliability standpoint, we are most interested in the FER, however for an eavesdropper, a frame error where only a single bit is incorrect is very different from a frame error where around half the bits are incorrect. Therefore the bit error rate can also be a useful measure because it gives an answer to how many bits are incorrect on average. However, bit errors in a decoded message are not independent, meaning that the average number of bit errors may not be a typical number.

We simulate at a given noise level until we have observed at least 100 frame errors as well as at least 100 correctly decoded messages, with the exception of BER simulations for poor channels (FER > 0.999), in which case the contribution of the correctly decoded messages would be insignificant to the BER. This requirement is in addition to a minimum number of transmissions, usually set at 100 000.

The frame error rate performance of the codes with message length 128 over the AWGN channel is shown in Fig. 4.2. In Chapter 5, we will show and discuss these and related results in different contexts relating to the wiretap channel.

4.3 Wiretap codes

In Section 2.6, we motivate the use of syndrome coding by showing that it can increase the equivocation of Eve. However, syndrome coding on its own provides no improvement in reliability for Bob and requires an almost noiseless channel to be usable. This motivates looking at other codes which do provide reliability. In addition, the steep performance curves of some codes may be useful in a wiretap scenario, as a smaller difference in signal quality between very poor performance and high reliability will translate into a smaller physical distance required between Bob and Eve in a wireless system.

In order to further illustrate this concept, let us define two thresholds T^E_{FER} and T^B_{FER}. These thresholds represent the lowest acceptable FER for Eve and the highest acceptable FER for Bob, respectively. Given two such thresholds as well as a given code, we may define the *security gap* Γ as [38]

$$\Gamma \triangleq \frac{\text{SNR}_{Bob}}{\text{SNR}_{Eve}}, \tag{4.3}$$

where SNR_{Bob} and SNR_{Eve} are the SNRs where the FER is equal to T^B_{FER} and T^E_{FER}, respectively. When the SNR is expressed in decibels, Γ becomes the difference between the two SNRs. We may also use the security gap with other measures than the FER. In order to give a preliminary intuition for how the code performance relates to the security gap, we show the security gap of four codes with $T^B_{\text{FER}} = 10^{-3}$ and $T^E_{\text{FER}} = 0.9$ in Fig. 4.3. Note that these are quite relaxed requirements. In Chapter 5, rather than setting T^E to a fixed value, we will show Eve's confusion as a function of Γ.

In the rest of this section, we will describe the codes we have used, and motivate these choices. To be able to compare codes directly, all codes have a rate of roughly 1/2. For simplicity, all message lengths are powers of two.

4.3.1 Syndrome coding

Syndrome coding is the standard method of coding for the wiretap channel [12], and is therefore the benchmark we compare our other codes against.

In our simulations we use syndrome codes based on BCH codes. This means we use a BCH code to create a parity check matrix, which is then in turn used to encode information in the syndrome of a vector, and to compute the syndrome of a received vector. This technique was more thoroughly described in Chapter 2. While [35] showed that there are syndrome codes that achieve higher equivocation than the best codes for communication, their technique is not viable for longer block lengths. It can also be argued

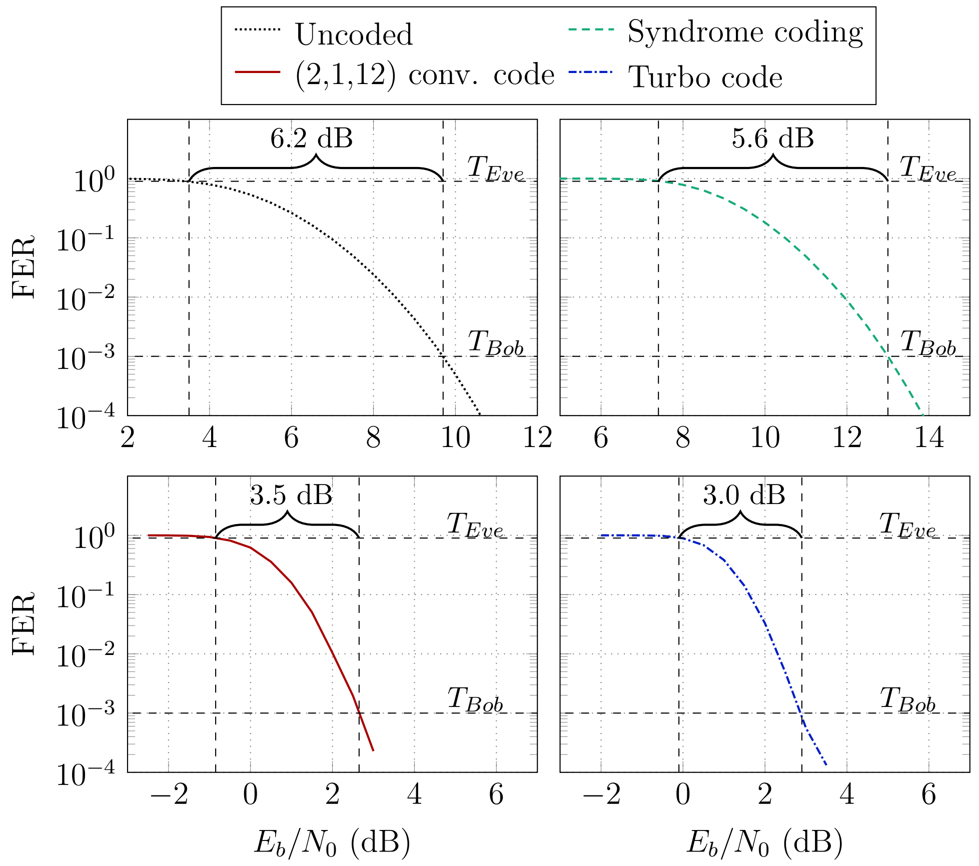

Figure 4.3: Security gap for different codes with $T^E_{\text{FER}} = 0.9$ and $T^B_{\text{FER}} = 10^{-3}$.

that due to the major difference in behavior between syndrome codes and other codes, it is not critical that the codes we use are completely optimal for comparison purposes.

The BCH codes we use are listed in Table 4.1, with the generator polynomial presented using octal digits. The binary form of each octal digit represents three coefficients in the polynomial, and the digits are listed with the highest order coefficients on the left. As discussed, we want our codes to have a rate of 1/2 and a message length that is a power of two. To get BCH codes on this (2^i, 2^{i-1}) form, the following constructions from [41] were used. Both BCH codes were extended by adding an all-one column to the generator matrix. For the resulting (256,131) code, we removed rows from the generator matrix to get a (256,128) subcode.

n	k	t	Generator polynomial
31	16	3	107657
255	131	18	2157133314715101512612502774421420241654771[†]

Table 4.1: BCH codes used [13].

4.3.2 Convolutional Codes

Convolutional codes are normally used for error correction, which as opposed to syndrome codes, will help us communicate reliably between Alice and Bob. We want to study how using convolutional codes affects Eve. A useful property of convolutional code is that a maximum likelihood (ML) decoder is available in the Viterbi algorithm [26]. This not only ensures optimal decoding for Bob, but it means we know how the best possible decoder for Eve performs. If we did not have an ML decoder and could not ensure that neither did Eve, it would be more difficult to reason about Eve's decoding performance.

Additionally, we are able to calculate the equivocation of Eve, given a specific received vector. We will discuss this in detail in Section 4.4.

The convolutional codes we use are listed in Table 4.2 with generator polynomials in octal form. We use terminated convolutional codes as block codes in our simulations and calculations. Messages of length k' are padded with m zeros before encoding, and the codeword length n' is given by $n' = 2(k' + m)$. The rate r is then given by

$$r = \frac{k'}{2(k' + m)}. \tag{4.4}$$

Consequently, our convolutional codes are not of rate exactly 1/2. The rates of our convolutional-based block codes are listed in Table 4.3, rounded to four decimal places. This difference in rate is inconsequential at larger block lengths, however it can be significant for shorter block lengths.

n	k	m	$\mathbf{g}^{(0)}$	$\mathbf{g}^{(1)}$
2	1	2	5	7
2	1	4	27	31
2	1	6	117	155
2	1	12	10627	16765

Table 4.2: Convolutional codes used [13].

4.3.3 Turbo codes

We are interested in Turbo codes for the wiretap channel because of their steep performance curves. A steep curve translates to a smaller security gap, as shown in Fig. 4.3.

[†]The generator polynomial of the (255,131) code is listed with a misprint in [13], missing the two octal digits "71" at the end of the polynomial. The polynomial is listed correctly in [42].

n'	k'	m	Rate
260	128	2	0.4923
264	128	4	0.4848
268	128	6	0.4776
280	128	12	0.4571
36	16	2	0.4444
40	16	4	0.4
44	16	6	0.3636

Table 4.3: Rate of convolutional-based block codes.

We therefore expect them to perform better than convolutional codes, however because of their more complex structure, it is harder to reason about the equivocation when using turbo codes. As opposed to for convolutional codes, there is not a practical ML decoder available. In theory, this means Eve could do better than the decoder we have used. In practice however, we perceive it as unlikely that Eve has access to a meaningfully better decoder. Due to the iterative nature of turbo decoding, an obvious approach for Eve would be to increase the number of iterations. While this can improve decoding performance, it is a process of diminishing returns. The more iterations you perform, the smaller the gain in performance will be, and eventually there is nothing to be gained from added iterations.

In our simulations, we use an implementation of turbo codes from the Coded Modulation Library (CML) from Iterative Solutions [43]. We use a turbo code with two constituent codes, which are both identical $(2, 1, 4)$ systematic feedback convolutional codes. We also puncture the code to get a rate closer to $1/2$. The generators and puncturing patterns of the constituent codes are listed in Table 4.4. Because the two constituent codes are identical, we only list their generators once. Puncturing patterns are represented with a 0 indicating a puncture, and are listed with 3 rows representing the information sequence and the two parity sequences. Each pattern is repeated over the length of the corresponding sequence.

The interleavers of our turbo codes are semi-random or *s-random* interleavers with $s = 4$. An s-random interleaver is simply a pseudorandom permutation that is generated with the constraint that neighbouring symbols in the input must be at least s places apart in the output.

Turbo code A is a relatively standard turbo code designed for communication. It is based on a rate $1/2$ code from the CCSDS standard [44] with a modified block length and interleaver. However, codes good for communication are not necessarily the best for the wiretap scenario. We will observe in Chapter 5 that when using the BER as a measure of Eve's confusion, turbo code A's BER does not grow as fast as we would like for poor-quality channels.

Inspired by [38], we look at a code that is not completely systematic, i.e. the full information sequence is not part of the codeword. In turbo code B, we puncture most of the information bits and instead retain most parity bits. This is known as a *partially*

Name	n	k	Rate	m	$\mathbf{g}^{(0)}$	$\mathbf{g}^{(1)}$	$\mathbf{P}$
A	264	128	0.4848	4	23	33	$\begin{matrix} 1 & 1 \\ 1 & 0 \\ 0 & 1 \end{matrix}$
B	260	128	0.4923	2	5	7	$\begin{matrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \end{matrix}$

Table 4.4: Turbo codes used.

systematic turbo code (PSTC), and was first described in [45]. Turbo code B is the result of a very rudimentary and manual search to find a code that has a higher BER than turbo code A for poor channels, while still retaining good enough performance for Bob.

4.4 Equivocation

While the frame- and bit error rates give an indication of what Eve is capable of in practice, the equivocation is a more fundamental and powerful measure that can ensure information-theoretical security. It can be difficult to calculate the equivocation in general. As noted in Chapter 2, maximum a posteriori decoding is known to be NP-hard [15]. MAP decoding involves finding the most probable transmitted codeword given a received vector. Intuitively, this should be easier than calculating equivocation, as the equivocation depends on the entire probability distribution of codewords given a received vector. Consequently, we conjecture that calculating the equivocation of a code over a given channel is NP-hard. We don't attempt to prove this statement here, and no proof has been found in the existing literature.

While calculating equivocation may be difficult in general, there are techniques we can use to estimate it for specific codes over certain channels. It is also possible to find bounds on the equivocation. In this section we will explore these options.

4.4.1 Direct calculation

In the uncoded case, bit errors are independent of each other, which means the total equivocation is simply a sum of the equivocation of each bit, which in turn is a function of the bit error probability. This gives us

$$H(X|Y) = k \cdot h_b(P_{be}), \tag{4.5}$$

where P_{be} is the bit error probability and h_b is the binary entropy function.

In [35], a method for directly calculating the equivocation of a syndrome code on the BSC is presented. The method involves computing a sum with 2^m terms, where m is the length of a message or syndrome, i.e. $m = n - k$ of the underlying code. Thus, we can only use this method for small values of m.

4.4.2 Estimation with Monte-Carlo method

Given a received vector $\mathbf{y}$, we may in certain cases be able to calculate the specific equivocation $H(X|Y=\mathbf{y})$. We have that

$$H(X|Y) = \sum_{\mathbf{y}\in\mathcal{Y}} P(Y=\mathbf{y})H(X|Y=\mathbf{y}). \tag{4.6}$$

A sum over all possible received vectors is computationally infeasible for all but the smallest block lengths on simple channels. However, by encoding and transmitting random messages over a simulated channel, a given vector $\mathbf{y}$ will naturally occur with probability $P(Y=\mathbf{y})$. Using this fact, an estimate of $H(X|Y)$ can be obtained by taking the average of the specific equivocation $H(X|Y=\mathbf{y})$ over a large number of simulations. This is called the *Monte-Carlo* method.

In [46], we present an algorithm for calculating the specific equivocation $H(X|Y=\mathbf{y})$ for codes that can be represented by a trellis, e.g. convolutional codes. The main idea of the algorithm is that whenever two paths merge into a single state s in the trellis, we have a decision with some entropy $H(s)$. Given $H(s)$ for all states, $H(X|Y=\mathbf{y})$ is a weighted average of $H(s)$. It is weighted by the probability π_s of a state being in the codeword path given the received channel output $\mathbf{y}$. π_s can be calculated through a forward and backward pass through the trellis. For a more detailed description of the algorithm, see the paper itself [46].

4.4.3 Upper bound from Fano's inequality

In Section 2.1.6 we describe Fano's inequality, which can give an upper bound on $H(X|Y)$ if X and Y take values from the same alphabet $\mathcal{X}$.

$$H(X|Y) \leq h_b(P_e) + P_e \log\big(|\mathcal{X}|-1\big) \tag{2.18 revisited}$$

We let $\mathcal{X} = \{0,1\}^k$, and let P_e be the frame error rate from our simulations. Thus, we can use simulation results to upper bound the equivocation.

4.4.4 Min-equivocation

As discussed in Section 3.1, the conditional min-entropy, or min-equivocation, $H_\infty(X|Y)$ is a stronger measure of Eve's confusion. Recall the definition of the min-equivocation

$$H_\infty(X|Y) = -\log \sum_{\mathbf{y}\in\mathcal{Y}} p(\mathbf{y}) \max_{\mathbf{x}\in\mathcal{X}} p(\mathbf{x}|\mathbf{y}). \tag{2.8 revisited}$$

As with Shannon equivocation, we have an average overall $\mathbf{y}$ weighted by $p(\mathbf{y})$. This lends itself to Monte-Carlo estimation, as $\mathcal{Y}$ may be a very large or even infinite set. Calculating $\max_{\mathbf{x}\in\mathcal{X}} p(\mathbf{x}|\mathbf{y})$ is not necessarily trivial for all codes. ML and MAP decoding algorithms will maximize similar probabilities, but they typically do not compute the actual value of those probabilities.

In the case where all codewords are equally likely, an ML decoding algorithm will maximize $p(\mathbf{x}|\mathbf{y})$. Raghavan and Baum [47] present the Reliability Output Viterbi Algorithm (ROVA). Using a trellis representation of a code, this algorithm will output $p(\mathbf{x}|\mathbf{y})$ where $\mathbf{y}$ is a received vector and $\mathbf{x}$ is the codeword chosen by the Viterbi algorithm. Given that all codewords are equally likely at the source, this will correspond to $\max_{\mathbf{x}\in\mathcal{X}} p(\mathbf{x}|\mathbf{y})$. The ROVA combined with Monte-Carlo simulations will thus allow us to estimate the min-equivocation of convolutional codes.

For uncoded transmission, we have independence between bits which allows us to compute the min-entropy of a single bit and multiply it by the frame length.

Chapter 5

Results

5.1 Energy use

In this chapter, we will mostly look at the relative difference in SNR between Bob and Eve. While this is a useful view for the wiretap channel, it hides the differences in absolute SNR between codes. Recall that the SNR is the energy per information bit divided by the noise. Assuming noise is constant, the difference in SNR between codes tells us how much more or less energy is required to achieve the same performance. Table 5.1 lists the energy required for a frame error rate of 10^{-3} relative to uncoded communication for codes with information length 128. This information can also be read from Fig. 4.2, however the decibel representation may be unintuitive for reasoning about the actual energy requirements.

Code	Relative Energy
Coset code	2.14
Uncoded	1.00
(2,1,2) conv. code	0.42
(2,1,4) conv. code	0.33
(2,1,6) conv. code	0.27
Turbo code B	0.23
Turbo code A	0.21
(2,1,12) conv. code	0.20

Table 5.1: Energy needed for FER $= 10^{-3}$, relative to uncoded. All codes have information length 128.

From Table 5.1, we see that coset coding requires more than twice the energy as uncoded, and approximately ten times the energy of the highest complexity convolutional code. This illustrates that coset coding on its own is extremely inefficient in practice. Viewing the wiretap channel in isolation without considering efficient and reliable communication for Bob will result in impractical conclusions.

5.2 Equivocation

In Fig. 5.1, we see the equivocation for a syndrome code, uncoded, and three convolutional codes with a message length of 16 bits. The syndrome code equivocation is calculated using the method from [35], while the convolutional equivocation is a Monte-Carlo estimate as described in Section 4.4.2 using the algorithm from [46].

This illustrates how in syndrome codes, the redundancy is used to increase equivocation, whereas in communication codes the redundancy reduces equivocation. It is clear from this plot that in the case where Bob has a noiseless channel, syndrome codes are the better choice to increase the equivocation for Eve.

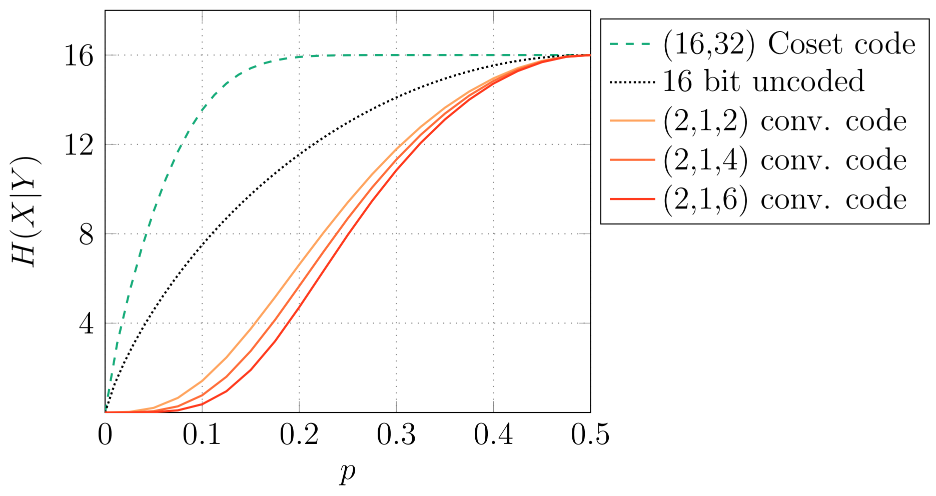

Figure 5.1: Equivocation over the binary symmetric channel with transition probability p.

As discussed in Chapter 4, we are largely unable to measure the equivocation for larger block lengths and on the AWGN channel, except for convolutional codes. We present some results on the equivocation of convolutional codes here, however we are unable to make direct comparisons with other codes except the uncoded case. In Fig. 5.2, we show the equivocation of convolutional codes and uncoded transmission for the 128-bit message length. We observe that convolutional coding gives a significant reduction in the required signal energy to achieve a small equivocation compared to uncoded transmission.

If Bob does not have a noiseless channel, we want to set a threshold T^B_{FER} which is an upper limit on Bob's frame error rate. We can then observe Eve's equivocation as a function of the difference in channel quality between her and Bob. In Fig. 5.3, we set $T^B_{\text{FER}} = 10^{-3}$, and show Eve's equivocation as a function of the difference in SNR between Bob and Eve. We see that coding improves the situation for Eve, meaning it worsens our security. This illustrates the issue faced when Bob does *not* have a noiseless channel; *helping Bob will also help the eavesdropper Eve.*

However, this dynamic is not necessarily as strong if we move away from information-theoretical security to a more practical view. In addition to her equivocation, let us consider Eve's frame error rate. We assume she uses a maximum likelihood decoder,

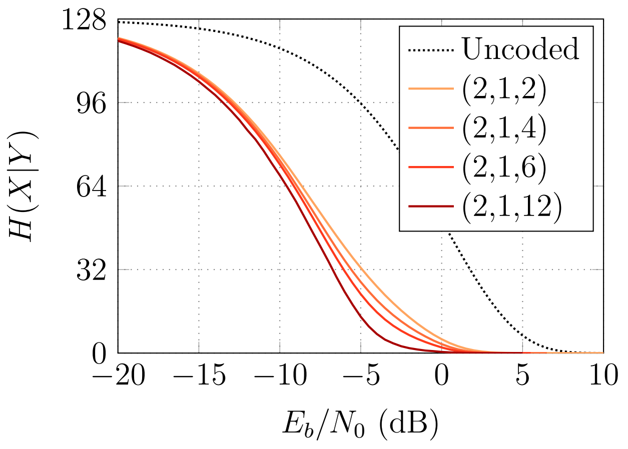

Figure 5.2: Equivocation $H(X|Y)$ of convolutional codes, compared to uncoded transmission.

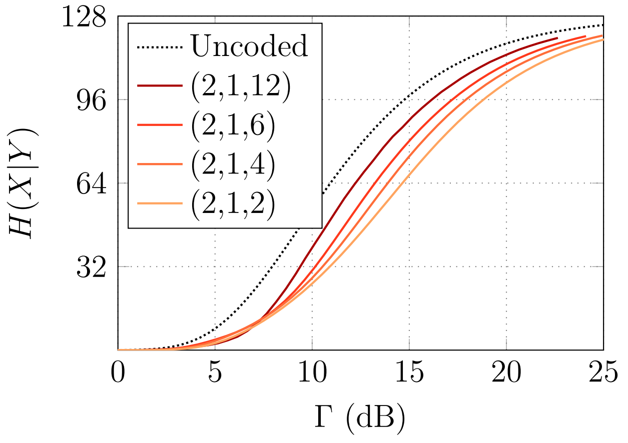

Figure 5.3: Equivocation of Eve when $T^B_{\text{FER}} = 10^{-3}$.

and only considers the single most probable codeword. In Fig. 5.4, we show both the equivocation and FER of Eve with a (2,1,6) convolutional code with a message length of 128 bits. We see that in order for Eve to have a FER ≈ 1, a *much* smaller difference in SNR is required compared to an equivocation close to 128 bits.

5.2.1 Min-equivocation

In Fig. 5.5, we show the conditional min-entropy of the convolutional codes as well as uncoded with 128 bit messages. Fig. 5.6 shows the same results as a function of the security gap when $T^B_{\text{FER}} = 10^{-3}$. Comparing these to equivocation (Fig. 5.2 and Fig. 5.3), it is clear that a larger security gap is needed to ensure a high min-equivocation for Eve.

For both equivocation and min-equivocation, we observe that it can be very difficult to get close to perfect secrecy, i.e. $H(X|Y) \approx 128$ or $H_\infty(X|Y) \approx 128$. However, in the context of privacy amplification where Alice and Bob are exchanging a key, we may

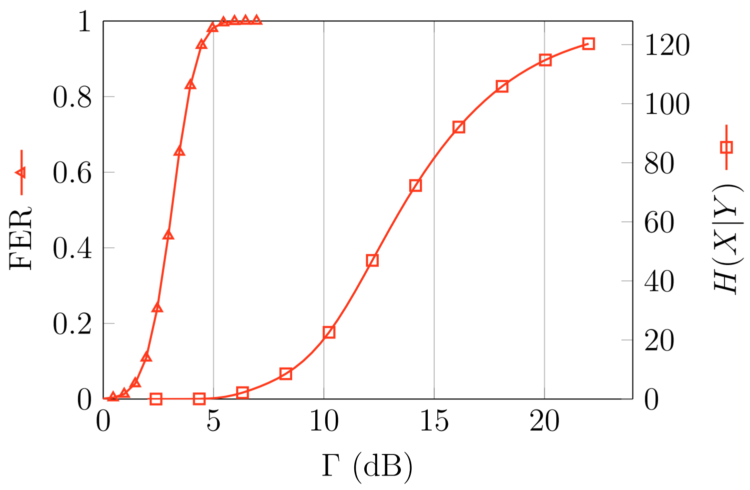

Figure 5.4: Equivocation and FER of Eve when $T_{\text{FER}}^{B} = 10^{-3}$. (2,1,6) covolutional code with 128 bit messages.

consider sending a larger message than necessary, and computing a one-way function or hash function with the message as input, where the output is smaller than the equivocation or min-equivocation. In such a situation it is not necessary to reach perfect secrecy over the full message. Note that security in this situation would depend on the properties of the hash function, but discussing that in detail is beyond the scope of this work.

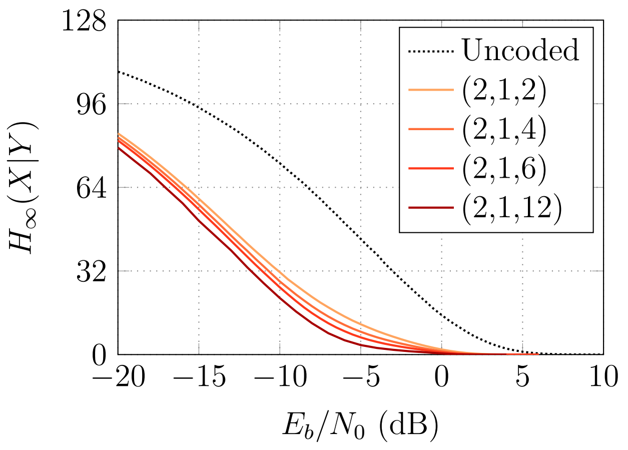

Figure 5.5: Min-equivocation for 128-bit messages over the AWGN channel.

5.2.2 Fano bound

As discussed in Chapter 4, Fano's inequality can give an upper bound on the equivocation $H(X|Y)$ based on the FER. This allows us to at least in a small sense reason about the entropy of other codes than convolutional codes. In Fig. 5.7, we show an upper bound on the equivocation of Eve as a function of the security gap when $T_{\text{FER}}^{B} = 10^{-3}$. We also

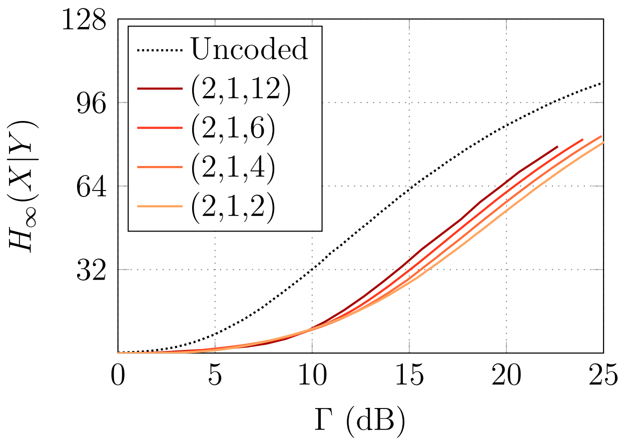

Figure 5.6: Min-equivocation of Eve when $T^B_{\text{FER}} = 10^{-3}$. 128 bit messages.

reproduce the values from Fig. 5.3, which is the equivocation for convolutional codes and uncoded in this situation. Comparing the bound with the actual value, we see that the Fano bound preserves the order of the convolutional codes, but does not give an accurate comparison between coded and uncoded. Additionally, the bounds are quite far from the true values (it is not a *tight* bound). These two factors diminish the value of Fano's inequality in this context. It is hard for us to make definitive statements about which codes give the highest equivocation based solely on these bounds.

For the future, newer, tighter bounds such as the one presented in [48] may provide a more useful picture.

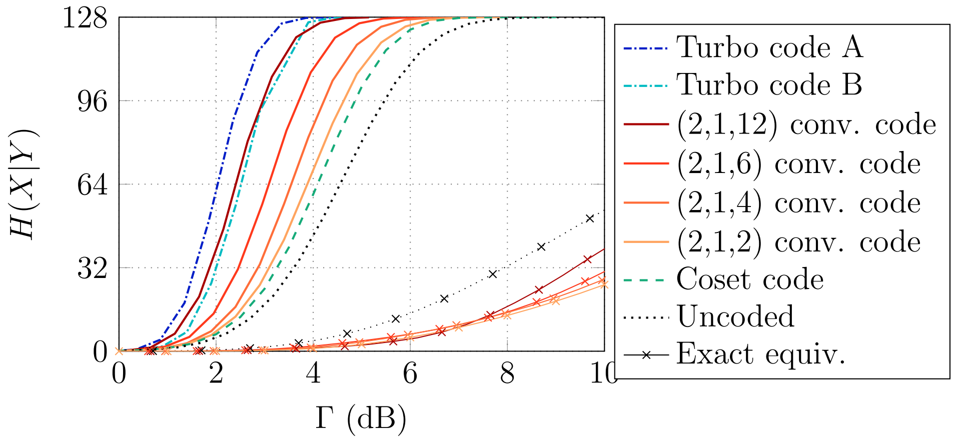

Figure 5.7: Upper bounds on $H(X|Y)$ from Fano's inequality, compared with the true value of $H(X|Y)$ for convolutional and uncoded.

5.3 Error rates

In this section, we will look at how the error rate performance curves of different codes affect Eve. However, keep in mind that error rates alone can not ensure information-theoretic security, and that there may be a rather large gap from a high error rate to a high equivocation, as seen in Fig. 5.4.

Using the frame- and bit error rate as a measure of Eve's confusion allows us to compare codes where it's not trivial to compute the equivocation. As seen in Fig. 4.3, the steeper performance curves of communication codes can give a smaller security gap under certain conditions. In Fig. 5.8, we show the FER of Eve, given that $T^B_{\text{FER}} = 10^{-3}$. We observe that Eve's FER is higher for the turbo and convolutional code than for the coset code. If we set $T^E_{\text{FER}} = 0.9999 = 1 - 10^{-4}$, the security gap of the turbo, convolutional and coset codes are approximately 4.5 dB, 6.3 dB, and 7.9 dB, respectively. Note that we omit the $(2, 1, 12)$ convolutional code here, due to the computational complexity of simulating it for such high frame error rates.

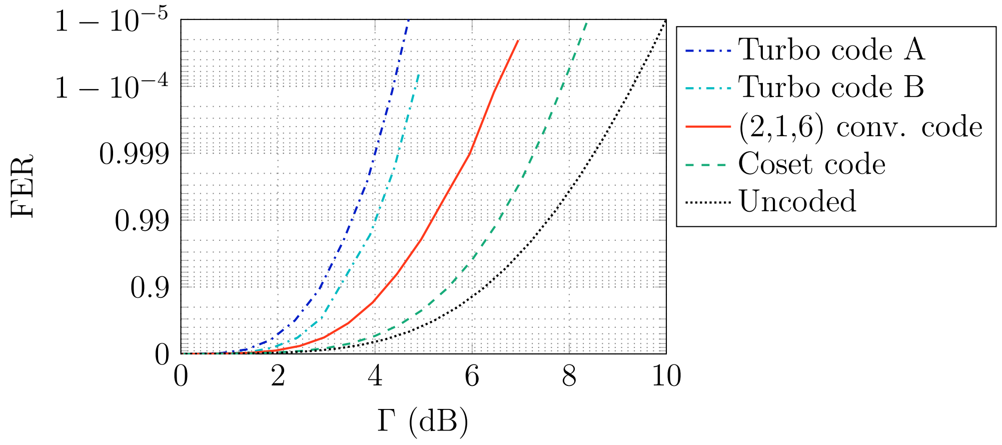

Figure 5.8: FER of Eve when $T^B_{\text{FER}} = 10^{-3}$. 128 bit messages.

In addition to the FER, we can look at Eve's bit error rate (BER). Ideally, we want Eve to have a bit error rate as close to 0.5 as possible. We will see that this is more difficult to achieve than a FER close to 1. In Fig. 5.9, we show Eve's bit error rate when $T^B_{\text{FER}} = 10^{-3}$.

For syndrome coding and convolutional codes, it is important to note that these simulations were performed using maximum likelihood (ML) decoding, which minimizes the frame error probability. Using a bitwise maximum a-posteriori (MAP) decoding may improve the BER performance. For the turbo codes, the complete decoder is neither ML nor MAP, however the decoder for each of the constituent codes are bitwise MAP decoders. The apparent "flattening" of the turbo code curves may be a result of the MAP decoders.

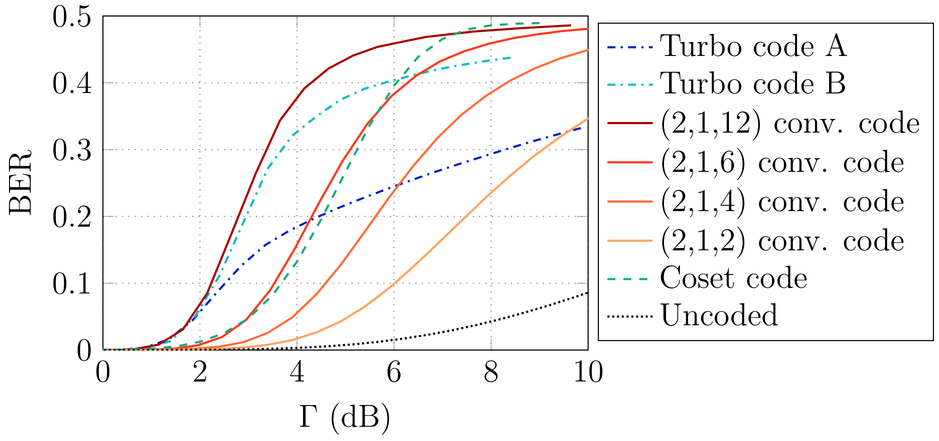

Figure 5.9: BER of Eve when $T^B_{\text{FER}} = 10^{-3}$. 128 bit messages.

5.4 Physical interpretation

As previously discussed, a real-world situation where the wiretap channel can arise is one where Alice is communicating wirelessly with Bob, and Eve is at some position further away than Bob. In this section we will illustrate the performance of the tested codes in such a situation.

In addition to placing Bob at a given SNR based on T^B_{FER}, we now also place him at a given physical distance from Eve. As described in Chapter 2, we can model wireless communication with a simple path loss model. The path loss exponent γ quantifies path loss, i.e. how fast the signal strength drops off over distance. In an idealized scenario we have $\gamma = 2$, but it may be higher or lower depending on the physical environment. Recall the following equation

$$\text{SNR}_2 = \text{SNR}_1 - 10\gamma \log_{10}\left(\frac{d_2}{d_1}\right). \tag{2.41 revisited}$$

In Fig. 5.10, we place Bob 10 meters from Alice and show Eve's FER as a function of her distance to Alice. In practice, this situation could be achieved by tuning the transmission power of Alice such that Bob has a FER of 10^{-3} at 10 meters. We see that a higher γ leads to a smaller security gap, but does not affect the comparison of the various codes. This illustrates that the codes may be compared on a channel model, however for a real-world application of wiretap codes, measuring γ or equivalent parameters of other models is necessary to guarantee security. In fact, bypassing the path loss model and instead measuring the SNR itself at various points may be a valid solution, as is done in [39].

By introducing a physical model we also make some new assumptions about Eve's capabilities. The SNR of a receiver will depend on several factors beyond the physical location. The size of a receiver's antenna may affect the SNR, as a larger antenna can pick up a larger share of the total signal energy. Additionally, the quality of the components of the receiver and its design may reduce electrical noise, which will increase the SNR. For our purposes we assume that Bob and Eve have the same receivers, however for a real-world deployment, one may want to quantify this possible increase in receiver quality, for example by comparing with the best available off-the-shelf devices, or by adding a security margin by multiplying Eve's estimated SNR by some constant.

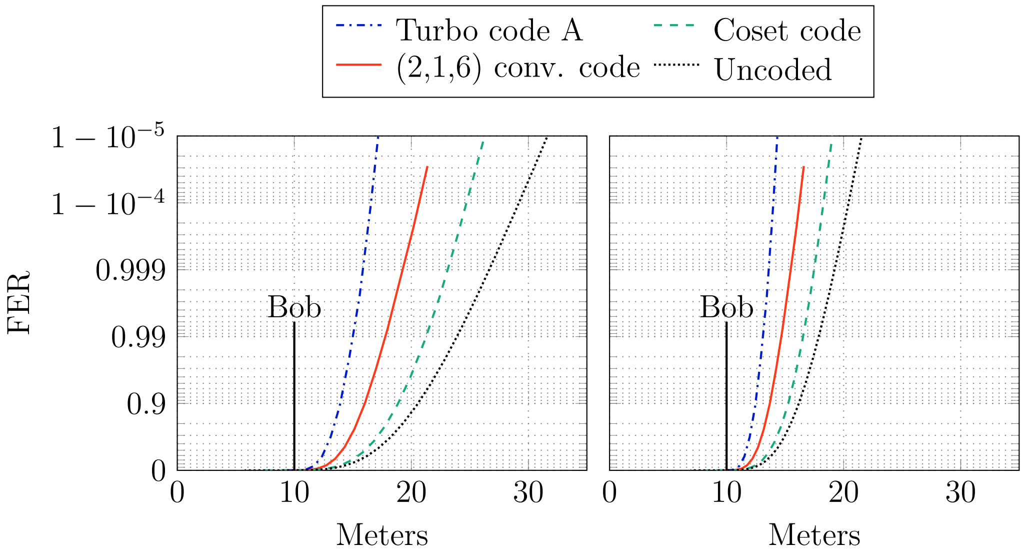

Figure 5.10: FER over distance, using the path loss model with $\gamma = 2$ (left) and $\gamma = 3$ (right). Bob has a FER of 10^{-3} at 10 meters.

Chapter 6

Conclusions

In this chapter we will summarize our findings, discuss the implications of those findings, and discuss which directions of research have not been fully explored in the scope of this thesis.

6.1 Summary

We have discussed how a lot of work on the wiretap channel assumes that Bob has a noiseless channel, and that analysis of Eve's confusion is done as the block length n tends to infinity. Neither of these assumptions apply to the case of real-world wireless communication. We therefore choose to explore alternative options to the ones presented previously, including the use of codes designed for communication. We also study the standard wiretap coding, syndrome coding, when the assumption about a noiseless channel is dropped. This reveals the major inefficiency of using syndrome coding for communication over a noisy channel, compared to other codes. The increase in the necessary signal energy makes it in our view infeasible to use syndrome coding on its own for wireless communication.

We have looked at several ways to measure Eve's confusion in this work. Specifically the equivocation, the min-equivocation, the frame error rate, and the bit error rate. For information-theoretical security as Shannon [4] and Wyner [1] defined it, the equivocation is the measure used. As min-equivocation is a lower bound on the equivocation, it too can ensure information-theoretical security, and is additionally a stronger measure of security as shown by Cohen and Zémor [8].

We have however seen that the equivocation and min-equivocation can be difficult to calculate, and that in the cases where we can compute them, only syndrome coding provides an improvement in security over uncoded communication. This presents a problem, because as we have discussed above, syndrome coding is impractical for noisy channels.

From this, we move to discuss another form of security, not based on information theory, but instead on observations of the bit error rate and frame error rate. This comes with the caveat that Eve may employ more advanced techniques than standard decoding, such as brute force through, for example, list decoding.

Our results on the frame error rate are very clear; to maximize Eve's frame error rate we should employ codes with steep performance curves, such as turbo codes or convolutional codes of high trellis complexity. The bit error rate gives a slightly less clear picture, but indicates that searching for codes that are not necessarily optimal for communication may give better security over the wiretap channel. Specifically, we find a partially systematic turbo code (PSTC) that requires only ~ 0.5 dB higher SNR for Bob than a standard systematic turbo code, but gives a significantly higher bit error rate for Eve. This is consistent with the suggestions of [38], where longer block length LDPC codes are considered.

We also note that convolutional codes of high trellis complexity give a higher BER for Eve than those of lower complexity. This may be counter-intuitive as high complexity convolutional codes tend to give a lower BER for good channels, but there is a crossover point where the higher complexity codes start performing worse.

6.2 Unexplored avenues

In this time-limited thesis project there are naturally interesting areas of study we were unable to explore. In this section we will discuss some of those areas.

A central question of the thesis is when we are able to calculate the equivocation of Eve. We know that the equivocation of syndrome codes is possible to estimate over the erasure channel using Monte-Carlo simulation [36], and using similar arguments as [46] we should be able to do the same for convolutional codes. This would provide a comparison of equivocation for longer block lengths than we were able to over the BSC.

In our discussion of turbo codes we indicate that there may be codes that are not optimal for communication, but provide a good trade-off when it comes to the reliability for Bob vs. the confusion of Eve. A more thorough search for such codes may yield better results.

Finally, we have seen that syndrome coding has good information-theoretical properties for security, but is impractical on its own. We suspect that a concatenated code with an outer communication code and an inner syndrome code may provide some of the information-theoretical properties of syndrome coding while still being usable for reliability. No calculations or simulations were made to support this, so this is purely a suggestion for future investigation. In this case, the rates of the two codes may be adjusted to spend more of the redundancy on confusion and less on reliability or vice versa.

Bibliography

[1] A. D. Wyner, “The wire-tap channel,” *The Bell System Technical Journal*, vol. 54, no. 8, pp. 1355–1387, 1975. DOI: 10.1002/j.1538-7305.1975.tb02040.x.

[2] United States, Executive Office of the President, *Executive order no. 13526, classified national security information*, https://www.archives.gov/isoo/policy-documents/cnsi-eo.html, 2009.

[3] International Telecommunications Union, *Recommendation ITU-R P.372-15 (radio noise)*, https://www.itu.int/rec/R-REC-P.372/en, Geneva, 2021.

[4] C. E. Shannon, “A mathematical theory of communication,” *The Bell System Technical Journal*, vol. 27, no. 4, pp. 623–656, 1948. DOI: 10.1002/j.1538-7305.1948.tb00917.x.

[5] M. Tribus and E. C. McIrvine, “Energy and information,” *Scientific American*, vol. 225, no. 3, pp. 179–190, 1971, ISSN: 00368733, 19467087. [Online]. Available: http://www.jstor.org/stable/24923125.

[6] A. Rényi, “On measures of entropy and information,” in *Proceedings of the Fourth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Contributions to the Theory of Statistics*, University of California Press, vol. 4, 1961, pp. 547–562.

[7] C. Bennett, G. Brassard, C. Crepeau, and U. Maurer, “Generalized privacy amplification,” *IEEE Transactions on Information Theory*, vol. 41, no. 6, pp. 1915–1923, 1995. DOI: 10.1109/18.476316.

[8] G. Cohen and G. Zemor, “Syndrome-coding for the wiretap channel revisited,” in *2006 IEEE Information Theory Workshop - ITW ’06 Chengdu*, 2006, pp. 33–36. DOI: 10.1109/ITW2.2006.323748.

[9] S. Fehr and S. Berens, “On the conditional rényi entropy,” *IEEE Transactions on Information Theory*, vol. 60, no. 11, pp. 6801–6810, 2014. DOI: 10.1109/TIT.2014.2357799.

[10] S. Arimoto, “Information measures and capacity of order α for discrete memoryless channels,” *Topics in information theory*, 1977.

[11] R. Yeung, *Information Theory and Network Coding*, ser. Information Technology: Transmission, Processing and Storage. Springer US, 2008, ISBN: 9780387792347.

[12] M. Bloch, O. Günlü, A. Yener, F. Oggier, H. V. Poor, L. Sankar, and R. F. Schaefer, "An overview of information-theoretic security and privacy: Metrics, limits and applications," *IEEE Journal on Selected Areas in Information Theory*, vol. 2, no. 1, pp. 5–22, 2021. DOI: `10.1109/JSAIT.2021.3062755`.

[13] S. Lin and J. Daniel J. Costello, *Error Control Coding*, Second. Upper Saddle River, NJ: Person Education, Inc., 2004.

[14] T. W. Hungerford, *Abstract algebra : An introduction*, eng, Pacific Grove, Calif., 2014.

[15] E. Berlekamp, R. McEliece, and H. van Tilborg, "On the inherent intractability of certain coding problems (corresp.)," eng, *IEEE transactions on information theory*, vol. 24, no. 3, pp. 384–386, 1978, ISSN: 0018-9448.

[16] R. W. Hamming, "Error detecting and error correcting codes," eng, *Bell System Technical Journal*, vol. 29, no. 2, pp. 147–160, 1950, ISSN: 0005-8580.

[17] D. E. Muller, "Application of boolean algebra to switching circuit design and to error detection," eng, *Transactions of the I.R.E. Professional Group on Electronic Computers*, vol. EC-3, no. 3, pp. 6–12, 1954, ISSN: 2168-1740.

[18] I. Reed, "A class of multiple-error-correcting codes and the decoding scheme," eng, *Transactions of the I.R.E. Professional Group on Information Theory*, vol. 4, no. 4, pp. 38–49, 1954, ISSN: 2168-2690.

[19] E. Prange, *Cyclic error-correcting codes in two symbols.* Air force Cambridge research center, 1957.

[20] A. Hocquenghem, "Codes correcteurs d'erreurs," *Chiffers*, vol. 2, pp. 147–156, 1959.

[21] R. C. Bose and D. K. Ray-Chaudhuri, "On a class of error correcting binary group codes," *Information and control*, vol. 3, no. 1, pp. 68–79, 1960.

[22] W. Peterson, "Encoding and error-correction procedures for the bose-chaudhuri codes," *IRE Transactions on information theory*, vol. 6, no. 4, pp. 459–470, 1960.

[23] E. R. Berlekamp, *Algebraic coding theory.* New York: McGraw-Hill, 1968.

[24] J. Massey, "Shift-register synthesis and bch decoding," *IEEE transactions on Information Theory*, vol. 15, no. 1, pp. 122–127, 1969.

[25] P. Elias, "Coding for noisy channels," *IRE Conv. Rec.*, vol. 3, pp. 37–46, 1955.

[26] A. Viterbi, "Error bounds for convolutional codes and an asymptotically optimum decoding algorithm," *IEEE transactions on Information Theory*, vol. 13, no. 2, pp. 260–269, 1967.

[27] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate (corresp.)," *IEEE Transactions on information theory*, vol. 20, no. 2, pp. 284–287, 1974.

[28] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near shannon limit error-correcting coding and decoding: Turbo-codes. 1," in *Proceedings of ICC '93 - IEEE International Conference on Communications*, vol. 2, 1993, 1064–1070 vol.2. DOI: `10.1109/ICC.1993.397441`.

[29] R. Gallager, "Low-density parity-check codes," eng, *I.R.E. transactions on information theory*, vol. 8, no. 1, pp. 21–28, 1962, ISSN: 0096-1000.

[30] R. G. Gallager, *Low-density parity-check codes*, eng, Cambridge, 1963.

[31] R. Tanner, "A recursive approach to low complexity codes," eng, *IEEE transactions on information theory*, vol. 27, no. 5, pp. 533–547, 1981, ISSN: 0018-9448.

[32] D. MacKay and R. Neal, "Near shannon limit performance of low density parity check codes," eng, *Electronics letters*, vol. 32, no. 18, pp. 1645–1646, 1996, ISSN: 0013-5194.

[33] E. Arikan, "Channel polarization: A method for constructing capacity-achieving codes for symmetric binary-input memoryless channels," *IEEE Transactions on Information Theory*, vol. 55, no. 7, pp. 3051–3073, 2009. DOI: `10.1109/TIT.2009.2021379`.

[34] T. S. Rappaport, *Wireless communications : principles and practice*, eng, 2nd ed., ser. Prentice Hall communications engineering and emerging technologies series. Upper Saddle River, N.J: Prentice Hall PTR, 2002, ISBN: 0130422320.

[35] S. Al-Hassan, M. Z. Ahmed, and M. Tomlinson, "New best equivocation codes for syndrome coding," in *2014 International Conference on Information and Communication Technology Convergence (ICTC)*, 2014, pp. 669–674. DOI: `10.1109/ICTC.2014.6983251`.

[36] J. Pfister, M. A. C. Gomes, J. P. Vilela, and W. K. Harrison, "Quantifying equivocation for finite blocklength wiretap codes," in *2017 IEEE International Conference on Communications (ICC)*, 2017, pp. 1–6. DOI: `10.1109/ICC.2017.7996925`.

[37] E. Verriest and M. Hellman, "Convolutional encoding for wyner's wiretap channel (corresp.)," *IEEE Transactions on Information Theory*, vol. 25, no. 2, pp. 234–236, 1979. DOI: `10.1109/TIT.1979.1056012`.

[38] D. Klinc, J. Ha, S. W. McLaughlin, J. Barros, and B.-J. Kwak, "Ldpc codes for the gaussian wiretap channel," *IEEE Transactions on Information Forensics and Security*, vol. 6, no. 3, pp. 532–540, 2011. DOI: `10.1109/TIFS.2011.2134093`.

[39] B. Jensen, B. Clark, D. Flanary, K. Norman, M. Rice, and W. K. Harrison, "Physical-layer security: Does it work in a real environment?" In *ICC 2019 - 2019 IEEE International Conference on Communications (ICC)*, 2019, pp. 1–7. DOI: `10.1109/ICC.2019.8761418`.

[40] J. Algrøy, *Master thesis (github)*, `https://github.com/jalgroy/master-thesis`, 2022.

[41] M. Grassl, *Bounds on the minimum distance of linear codes and quantum codes*, Online available at `http://www.codetables.de`, Accessed on 2021-12-14, 2007.

[42] G. C. Clark, *Error-correction coding for digital communications*, eng, ser. Applications of communications theory. New York: Plenum Press, 1981, ISBN: 0306406152.

[43] Iterative Solutions. (2009). "Coded modulation library," [Online]. Available: `http://iterativesolutions.com/Matlab.htm` (visited on 04/13/2022).

[44] G. P. Calzolari, E. Vassallo, and S. Habinc, "Ccsds telemetry channel coding: The turbo coding option," in *5th CCSDS Workshop New Technologies, New Standards*, 1998. [Online]. Available: `http://microelectronics.esa.int/vhdl/doc/TurboCCSDS.pdf`.

[45] I. Land and P. Hoeher, "Partially systematic rate 1/2 turbo codes," in *Proc. Int. Symp. Turbo Codes*, 2000, pp. 287–290.

[46] J. Algrøy, A. I. Barbero, and Ø. Ytrehus, "Determining the equivocation in coded transmission over a noisy channel," in *2022 IEEE International Symposium on Information Theory (ISIT) (ISIT 2022)*, Espoo, Finland, Jun. 2022.

[47] A. R. Raghavan and C. W. Baum, "A reliability output viterbi algorithm with applications to hybrid arq," *IEEE Transactions on Information Theory*, vol. 44, no. 3, pp. 1214–1216, 1998.

[48] M. Hledík, T. R. Sokolowski, and G. Tkačik, "A tight upper bound on mutual information," in *2019 IEEE Information Theory Workshop (ITW)*, 2019, pp. 1–5. DOI: `10.1109/ITW44776.2019.8989292`.