Advances in Syndrome Coding based on Stochastic and Deterministic Matrices for Steganography

Dissertation

zur Erlangung des akademischen Grades Doktoringenieur (Dr.-Ing.)

vorgelegt an der
Technischen Universität Dresden
Fakultät Informatik

eingereicht von
Dipl.-Medieninformatikerin Antje Winkler
geboren am 10. November 1980 in Dresden

Gutachter:
Prof. Dr. rer. nat. Andreas Pfitzmann († 23.9.2010), Technische Universität Dresden
Prof. Dr. rer. nat. Hermann Härtig, Technische Universität Dresden
Prof. Dr. Ing. Jana Dittmann, Otto-von-Guericke Universität Magdeburg

Tag der Verteidigung: 1. November 2011

Dresden, den *20. Februar 2012*

Contents

List of Figures

List of Tables

1. Introduction

1.1. Motivation

The aim of communicating confidentially has historic roots which date back to ancient times. Generally, the scenario in steganography can be described as two parties - sender and receiver - who want to exchange a confidential message unobservably via a publicly observable channel.

Modern steganographic schemes are based on digital data. The sender hides the confidential message into inconspicuously looking digital cover material, e. g., text files or images. In contrast to cryptography, the existence of the message itself is hidden by using the cover material as a covert channel adding an additional level of confidentiality.

Today, advances in digital steganography are relevant for several reasons. First, the existence of steganography is a strong argument against crypto-regulation, a debate in Germany in the 1990s. This topic is still up to date considering countries, like Russia. Koops gives a good overview on countries regulating the application of crypto [72].

Furthermore, steganographic systems can be applied in military or economic context, whenever the content of the communication as well as the communication itself should remain confidential. Moreover, concepts and algorithms developed in steganography form also interesting ingredients for new methods in digital right protection systems as well as in digital forensic methods.

Generally, a steganographic scheme is considered broken, whenever the attacker discovers the confidential communication, i. e., whenever he is able to distinguish between cover and stego material with success better than random guessing. Note that it is not required to actually extract the message, since the goal of steganography is to hide the existence of the message itself.

Thus, the goal for the designer of a steganographic system is clear: cover and stego should be undistinguishable. Therefore, the characteristic of the cover has to be preserved during the embedding process, which equals the minimization of artifacts introduced during embedding.

However, since cover data is mostly high dimensional empirical data, such as digital images, modeling the covers' characteristic is not trivial. Thus, heuristic approaches for preserving the cover's characteristic are applied in practical schemes. Fridrich et al. distinguish between four heuristic principles reducing the distortion introduced during embedding: Model-Preserving Steganography,

Mimic Natural Processing, Steganalysis-Aware Steganography and Minimum-Embedding-Impact Steganography [36].

Generally, it is a hard task to preserve a statistical model during embedding, sometimes even additional artifacts are introduced (see, e. g., [30, 9]). Thus, this thesis focusses on several algorithms related to Minimum-Embedding-Impact Steganography, i. e., on embedding while minimizing the introduced distortion. The main goal for the design of steganographic algorithms of this class is to take into account the element-wise defined embedding impact. Thus, the sender tries

- to disturb the cover data as little as possible in order to prevent detectable changes in the statistical properties, and
- to modify the cover data only in inconspicuous parts.

Note that the application of concepts from channel coding in steganography is known to be advantageous in several scenarios. Considering, e. g., a transmission over a lossy channel, either due to an attacker or due to JPEG compression, channel codes are needed to retrieve the embedded message (e. g., [97, 10]). Furthermore, coding is required for the adjustment of distributions, e. g., in Mimic Functions (e. g. [32]).

This thesis focusses on the application of channel coding, more specifically syndrome coding, to reduce distortion introduced during embedding and to exclude parts of the cover from embedding.

1.2. Contribution of this Thesis

This work deals with syndrome coding as a well known concept in steganography for embedding a confidential message. In the past, several algorithms have been proposed in this field. A first solution for minimizing the introduced distortion was proposed by Crandall [16] and is well known as Matrix Embedding (ME). Wet Paper Codes as an approach for modifying only inconspicuous data parts, where the selection rule does not have to be shared with the receiver, was proposed first by Fridrich et al. in [44]. Furthermore, several approaches addressing both objects have been developed, e. g., based on BCH Codes as proposed by Schönfeld and Winkler [85, 86, 87].

The huge amount of algorithms for embedding based on syndrome coding proposed within the last years have never been comprehensively described. Thus, this work aims at describing state-of-the-art algorithms on a consistent basis in order to make them comparable to each other. Note that two of the algorithms presented within this thesis are developed at the chair of privacy and data security at the Technische Universität Dresden in collaboration of Dagmar Schönfeld and the author of this thesis. Furthermore, this thesis compares the different

approaches according to several properties in order to determine advantages and disadvantages for each class of algorithms.

The thesis is organized as follows: Part I gives an introduction into steganography motivating the application of syndrome-coding based steganography. After the basic description of steganographic schemes, fundamental design parameters of steganographic schemes such as security, complexity, capacity and embedding success are described.

Note that the algorithms discussed in this thesis are only a component of the embedding algorithm rather than a complete embedding scheme. Thus, we present an advanced sender model in order to clarify the function of the algorithms discussed in this thesis.

Furthermore, the basic concept for embedding based on syndrome coding is given. Generally, it is possible to separate the multitude of algorithms into several classes. In this thesis, we separate between approaches based on a deterministic parity-check matrix and approaches based on stochastic matrices. For both classes, we describe the basic algorithms for binary embedding. Note that applied schemes such as ZZW (Zhang Zhang Wang approach) [105] are not in the focus of this work.

Furthermore, we separate between concepts for a small code word length, described in Part II and concepts for a large code word length (Part III). For small code word lengths, approaches based on exhaustive search are applicable in order to determine the stego sequence with minimal distortion. This is not possible for large code word lengths and specific solutions are necessary.

After describing the algorithms on a consistent basis, we compare them according to:

- the security in terms of embedding efficiency,
- the capacity,
- the success of embedding,
- the complexity.

1.3. Notation

In this thesis, vectors and matrices are denoted in boldface with indices following the symbol in square brackets. The ith element of vector $\mathbf{x}$ is denoted as $\mathbf{x}[i]$. Furthermore, the ith row of matrix $\mathbf{X}$ is denoted as $\mathbf{X}[i,.]$ and the jth column as $\mathbf{X}[.,j]$. The transpose of a matrix is denoted as $\mathbf{X}^T$ and the transpose of vector $\mathbf{x}$ as $\mathbf{x}^T$. Note that sequences of vectors or matrices will be indexed with a subscript. Furthermore, the length of vector $\mathbf{x}$ is denoted as $|\mathbf{x}|$ and the concatenation of strings is denoted as $[\mathbf{x}\ \mathbf{y}]$.

Sets are denoted in this work in calligraphic font, with $|\mathcal{X}|$ denoting the cardinality of $\mathcal{X}$. Furthermore, a polynomial $p(x)$ is defined as $p(x) = u_k x^k + u_{k-1}x^{k-1} + ... + u_1 x + u_0$, where the sequence of its coefficients can be seen as a binary vector $\mathbf{p} = (u_k u_{k-1} ... u_1 u_0)$ and is denoted in boldface symbols.

In this thesis, operations are based on eXclusive OR operation on bits which we denote as $\oplus$. Furthermore, the binary representation of i is denoted as $[\mathbf{i}]_2$ and the denary representation as $[\mathbf{i}]_{10}$, respectively.

The Kronecker Product is denoted as $\otimes$. Furthermore, we reserve $\lfloor x \rfloor$, $\lceil x \rceil$ for rounding down and up, respectively. An overview on symbols and functions employed in this work is given at the end of this document.

1.4. Acknowledgements

A number of persons have supported me during the time in which I wrote and completed this thesis. I would like to give my thanks to all of them.

It is with respect and appreciation that I acknowledge Prof. Dr. Andreas Pfitzmann († 23.9.2010) for his support and encouragement. He gave me the opportunity to do my research and write this thesis.

I am especially indebted to Dr. Dagmar Schönfeld who drew my attention to the field of channel coding. It was under her supervision that I started to work on syndrome coding for steganography. I would like to thank her for the opportunity to work with her, for supporting my research and giving me insightful comments and inspiration on drafts of this manuscript.

I also want to express special thanks to all my dear colleagues at the Privacy and Security Group at the Dresden University of Technology (Technische Universität Dresden) for numerous hints and fruitful discussions.

And last but not least, I am very gratefully to my family and friends who supported me a lot - Thank you.

Part I.

Background and General Concept

2. General Concept of Steganography

Within this chapter, we describe the model of a steganographic scheme in general and how to parametrize this scheme in a scenario where the steganographic algorithm is publicly known. Furthermore, we give a short overview of parameters describing the properties of a steganographic scheme such as security. Based on these considerations, practical implications for the design of a steganographic scheme are then summarized.

Since this work deals with syndrome coding as a tool to improve steganography by minimizing the distortion introduced during embedding, the goal is not to design a whole steganographic scheme. We rather introduce and compare concepts so as to improve the security of a steganographic scheme by means of syndrome coding. Therefore, we present an advanced model for describing the steganographic scheme and describe the underlying concept of this work in detail.

2.1. A Brief Description of the Steganographic Scheme

Generally, steganography and cryptography are similar in that both are used for communicating a secret message. In the case of cryptography, the presence of the message itself is obvious; only the content is protected by a secret $\mathbf{k}$. The goal of *steganographic schemes* is different. The secret message is embedded into inconspicuously looking *cover* material, e. g., text files or images. Consequently, the existence of the message itself is hidden by using the cover material as a covert channel, which should be slightly modified in an imperceptible way. The resulting *stego* object is now transmitted over a public channel to the receiver.

Thus, steganography is considered broken whenever the attacker discovers the secret communication. Note the contrast to cryptography, where the attacker is successful if he breaks the cipher. Consequently, the most important feature of a steganographic scheme is undetectability, i. e., it should not be possible to decide whether steganography has been used better than random guessing.

2.1.1. Schematic Description of the Steganographic Scheme

The concept of steganography can be formalized using the *prisoners' problem* introduced by Simmons [90]. In this scheme, it is assumed that two prisoners, Alice and Bob, are imprisoned in separate cells and want to exchange an escape plan. Since they are separated from each other in different cells, they can communicate only by sending messages through the warden Wendy, i. e., their messages are observed by Wendy. Wendy can be seen as a *passive warden*, whose goal is to exchange each message, as long as she does not find it suspicious. Otherwise Wendy will not provide the message at all.

This scenario can be compared to the scenario of a steganographic scheme, where two parties would like to communicate confidentially as many bits as possible without introducing detectable artifacts. However, the channel they are using is publicly known and thus can also be observed by outside third parties.

Generally, there are three different approaches as to how to embed a secret message into a cover: By *cover selection*, the sender simply selects an appropriate cover that already contains the secret message, i. e., cover and stego are identical. Another, more artificial way of embedding a message is *cover synthesis*, where the sender has to create appropriate data containing the message. Beside these two rather theoretical approaches, *cover modification* is the most frequently used case in practical applications.

In this approach, the sender selects an arbitrary cover from a large source of covers and embeds the secret message by modifying the cover. The underlying scheme for this approach as it is used in this thesis follows [80] and is depicted in Figure 2.1.

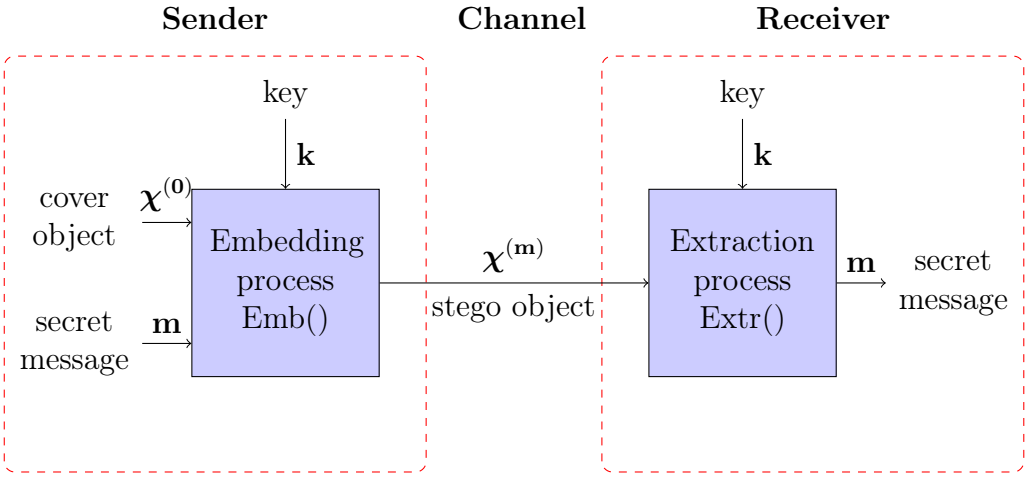

Figure 2.1.: Schematic Diagram of a Steganographic Scheme Including Sender, Receiver and the Communication Channel Following [80].

The scheme includes the two parties of sender and receiver and their trusted domains (depicted as dashed boxes), as well as the publicly known communication channel. Note that the channel is neither within the trusted domain of the sender nor in the trusted domain of the receiver. Therefore, both have to be aware of an attacker who is able to eavesdrop on their communications, just like the warden Wendy in Simmons' scenario.

In the scenario depicted in Figure 2.1, sender and receiver would like to communicate a secret message $\mathbf{m} \in \mathcal{M}$ via the public communication channel. Therefore, the sender hides the secret message $\mathbf{m}$ in inconspicuous-looking cover material $\boldsymbol{\chi}^{(\mathbf{0})} \in \mathcal{X}$ depending on a secret key $\mathbf{k} \in \mathcal{K}$, where $\mathcal{M}$ is the space of possible messages, $\mathcal{X}$ the space of possible cover objects and $\mathcal{K}$ the key space. Thus, the embedding process can be seen as a function $\text{Emb}() : \mathcal{X} \times \mathcal{M} \times \mathcal{K} \rightarrow \mathcal{X}$.

The output of the embedding process, called stego object $\boldsymbol{\chi}^{(\mathbf{m})} \in \mathcal{X}$, is then transmitted over the channel to the receiver, who must be able to extract the message from the stego data. Therefore, the receiver uses the secret key $\mathbf{k}$ to retrieve the secret message $\mathbf{m}$. Note that for him it is not necessary to reconstruct the cover data itself. Thus, the extraction step can be seen as a function $\text{Extr}() : \mathcal{X} \times \mathcal{K} \rightarrow \mathcal{M}$, such that $\text{Extr}(\text{Emb}(\boldsymbol{\chi}^{(\mathbf{0})}, \mathbf{m})) = \mathbf{m} \quad \forall \mathbf{m} \in \mathcal{M}, \ \forall \mathbf{k} \in \mathcal{K}$ and $\forall \boldsymbol{\chi} \in \mathcal{X}$. Consequently, a *steganographic algorithm* can be described as a 5-tuple $(\mathcal{X}, \mathcal{M}, \mathcal{K}, \text{Emb}(), \text{Extr}())$.

2.1.2. Parametrization of the Steganographic Scheme

We assume, just as in cryptography, that the steganographic algorithm applied by sender and receiver is publicly available. Therefore, the steganographic algorithm is generally also known to an attacker. This principle is known as *Kerckhoff's principle* [67].

Based on this assumption, it is necessary to parametrize the steganographic scheme in order to complicate breaking the scheme by an attacker.

2.1.2.1. Encrypting the Message

Usually, the secret message itself is encrypted before embedding. Therefore, in this work we assume the secret message $\mathbf{m}$ as a random bit string.

Generally, there are two different approaches. The first approach, called *Secret Key Steganography (SKS)*, is based on symmetric keys. Sender and receiver both use the same key to encrypt and decrypt the message. It is thus assumed that both sender and receiver share a secret key. In this case, the message can be recovered only by a person who possesses the secret key.

However, this implies that both must have had the possibility of communicating securely beforehand in order to exchange this key. This assumption implies the suspicion of a mandatory one-time confidential communication sometime. In

order to overcome this key distribution problem, *Public Key Steganography (PKS)* was invented [1].

In a PKS scheme, the receiver generates two keys, a public and a private one. He shares his public key with everyone. This means that the warden can obtain the key as well. The communication model can be described as follows. The sender takes the public key from the receiver, which can be provided by a phone book or something similar. He uses this key for encrypting the secret message before embedding it into the cover. After transmitting the stego data via the channel, the receiver extracts the encrypted message. Only the receiver is able to decrypt the secret message by using his private key.

However, if only the secret message is encrypted, an attacker still knows the steganographic algorithm. Therefore, he can extract the embedded message and mount an attack. Note that the steganographic scheme is considered insecure if an attacker is able to decide whether a secret message is hidden. Thus, even if he is not able to recover the message content itself (where he would have to break the underlying crypto scheme), the steganographic scheme is considered broken.

2.1.2.2. Parametrizing the Embedding Process

Thus, in order to keep the communication secret, we have to use a key to parametrize the steganographic algorithm, which is shared between sender and receiver and kept secret.

One possibility for parametrizing the steganographic algorithm is to build a steganographic scheme based on a *selection channel*, e. g., to select parts of the cover that should be used for embedding. Therefore, the key shared between sender and receiver can, e. g., be used to *generate a pseudo-random path* (**Path**). The message is embedded along this path. Thus, for an attacker who does not know the key to generate **Path**, extracting the correct message and by this the separation between steganographycally used and unused elements will be a hard task.

Generally, a pseudo-random path, as proposed by Crandall in [16], does not take into account the properties of the cover object. For an attacker it may be easier to discover embedding when homogeneous regions are used for embedding the secret message. Therefore, a method to improve the security of the steganographic scheme is *embedding in adaptively selected parts* of the cover, e. g., texture-rich or noisy image parts (e. g. [31, 32]).

Adaptivity in steganography means that the embedding process also considers the position where each embedding change will appear. Generally, the steganographic algorithm is designed so as to avoid changes in certain areas of the cover data and the selection channel is designed accordingly.

The main problem of an adaptive selection channel is being able to reconstruct the positions where message bits are embedded. The selection of the elements is influenced by some side information known to the sender, but not necessarily

known to the receiver. Therefore, the receiver does not always know from which positions he should extract the message.

It is generally in the interest of sender and receiver to reveal as little information as possible about the selection channel, as this knowledge can be explored by an attacker. If the knowledge of the selection channel is available to the sender but not to the receiver, this scheme is called *non-shared selection channel*.

Fridrich et al. explained non-shared selection channels using the metaphor *"writing on wet paper"* [44]: The cover image was exposed to rain and some pixels became wet. The sender can use only dry pixels to embed the secret message. However, during transmission the stego images dries out and the receiver has no information about the positions used for embedding. A solution to this problem is embedding based on syndrome coding, a principle of coding theory considered in this thesis. This approach is also referred to as Wet Paper Steganography in literature.

2.1.3. Properties of a Steganographic Scheme

A good overview on basic properties describing steganographic schemes, is given, e. g., by Böhme in [8]. There are three basic properties which can be used to describe steganographic schemes: their security, their capacity and their robustness. Note that the three dimensions are not independent. They should rather be considered as competing goals, which have to be balanced when designing a steganographic scheme.

Even if there is wide consensus regarding the properties, the metrics by which they are measured are not commonly defined. In this thesis, we will adhere to the definitions given by Böhme [8].

The purpose of steganographic communication is to hide the existence of a secret message. Therefore, the *security* of a steganographic scheme is judged on the basis of the impossibility of detecting the communication. There are different approaches to describe the security, e. g., in a general way, by means of information-theoretic or complexity-theoretic aspects.

Moreover, we can define steganographic security in a more heuristic way as the ability to resist steganalytical investigations. Generally, the steganalysis problem can be seen as a decision problem, e. g., finding an answer to the question as to whether observed data contains a hidden message or not. Further information on the definition of security is given in Section 2.2.

Furthermore, Böhme defines the *capacity* of a steganographic scheme as the maximum length of a secret message (in bit) divided by the number of bits required to store the stego object. Note that the capacity depends on the function chosen for embedding as well as on the properties of the cover. Moreover, the capacity of a steganographic scheme is bounded by the entropy of the cover [15].

Often, the measure Bit Per Pixel (bpp) is used, in order to express the *relative message length* $\alpha = \frac{|\mathbf{m}|}{N}$, i. e., the ratio between the maximum length of a secret

message (in bit) and the number of elements of the cover sequence[1]. In this thesis, we also refer to the relative message length.

Note that capacity and relative message length are related to the security of a steganographic scheme. Since embedding longer messages requires more changes due to embedding, it is statistically easier to detect. Filler and Ker also investigated the connection between relative message length and security [66, 26, 22]. They state that the cover size indeed plays an important role. Even if it is common to report the capacity as the rate of bpp, the rates are not compatible. Filler and Ker found that hiding in small covers is less detectable than data hiding by the same rate in larger covers. They state that the secure relative message length of practical stego systems grows only with the square root of the cover size, i. e., the secure capacity is proportional only to $\sqrt{N}$. This phenomenon is denoted as Square Root Law and has been experimentally confirmed.

Another property of steganographic schemes is its *robustness*, which refers to the difficulty of removing hidden information from a stego object. This property is important only when the communication channel is disordered by random errors or by systematic interference with the aim of preventing the use of steganography. Since this assumption is not true in most steganographic schemes, robustness has not received much attention so far in steganography.

Consequently, we take into account the properties of security and capacity in this thesis. Moreover, some additional measures such as the *embedding complexity* and the *success rate*, i. e., the probability that a given message can be embedded in a particular cover, are considered.

In this thesis, we consider time complexity as well as memory complexity. Whereas time complexity is expressed, e. g., by the number of XOR-operations needed for embedding, memory complexity is given as the number of bits that have to be stored.

To summarize, we find the following important parameters which should be considered when designing a steganographic algorithm:

- the security,
- the capacity,
- the success rate, and
- the embedding complexity.

Generally, the designer of a steganographic algorithm aims at maximizing the capacity as well as the security of his algorithm. Moreover, he tries to minimize the embedding complexity while maximizing the success rate. However, since these are competing goals, it is possible only to balance those four properties.

[1] Since N is defined as the number of discrete cover elements, this definition of α does not include embedding in multiple bit planes.

In the following, we will take a more detailed look at definitions of steganographic security.

2.2. A Security View on Steganography

Generally, the security of a steganographic scheme is judged by the impossibility of detecting the confidential communication. Therefore, the problem of an attacker can be seen as a decision problem, i. e., finding an answer to the question as to whether observed data contains a hidden message or not. Thus, decision-theoretic measures qualify as measures of steganographic security.

However, formalization of the Simmons model is not easy. The warden Wendy has access to everything except for the key, according to Kerckhoff's principle. Thus, she has complete knowledge of the steganographic algorithm and, therefore, also of the source of cover objects. However, since, e. g., digital images are high dimensional objects, it is not feasible to obtain even a rough approximation of its distribution [8].

Thus, only a set of statistical quantities derived from the cover can be used to describe the cover. The attacker can calculate these statistics and define significant deviations from the expected values as evidence for the presence of steganographic modifications. This quantitative view of security permits a mathematical formulation as well as statistical investigations.

Generally, the problem of steganography can be formulated as finding embedding and extraction algorithms for a given cover source that enable communication of reasonable large messages without introducing any embedding artifacts that could be detected by the warden. In other words, the goal is to embed secret messages in an undetectable manner.

Within this section, we first examine formal definitions of steganographic security. Afterwards, we give a short introduction to steganalytical approaches. An overview on this topic can also be found in [36].

2.2.1. Information-Theoretic Approach

Based on the definition of *information-theoretic security*, it should be impossible for an attacker to design a steganalytical method for a truly secure steganographic scheme that can distinguish between cover and stego objects.

Generally, the distribution of covers is denoted as $P_{\mathbf{c}}$ and the distribution of stego objects with $P_{\mathbf{s}}$. Consequently, the goal in steganography can be reformulated as constructing a steganographic scheme that assures $P_{\mathbf{s}}$ to be as close as possible to $P_{\mathbf{c}}$. Therefore, it would be hard for an attacker to decide whether the observed data was generated by the steganographic scheme or not. Note that according to Kerckhoff's principle, the attacker has access to the steganographic scheme.

To summarize, if an attacker is not able to determine whether observed data is drawn from $P_{\mathbf{c}}$ or drawn from $P_{\mathbf{s}}$, we will call the system information-theoretically secure or perfectly secure.

Therefore, the distance between $P_{\mathbf{c}}$ and $P_{\mathbf{s}}$ is related to the ability of an attacker to distinguish between cover and stego. Consequently, this distance can be taken as a measurement of steganographic security [13]. In this publication, the attacker's decision problem is described as a hypothesis testing problem.

2.2.1.1. Hypothesis

The basic idea of Cachin's work [13] can be formulated as follows: Based on an observed object $\boldsymbol{\chi}$, an attacker must decide between the two hypotheses:

- H_0: $\boldsymbol{\chi}$ is drawn from the distribution $P_{\mathbf{c}}$ and does not contain any hidden message.
- H_1: $\boldsymbol{\chi}$ is drawn from $P_{\mathbf{s}}$ and contains a hidden message.

Assuming the attacker can observe the communication channel (Kerckhoff's principle) and thus the distributions $P_{\mathbf{c}}$ and $P_{\mathbf{s}}$, the steganographic security can be expressed by means of their *Kullback-Leibler divergence (KL divergence)*, also called KL distance or relative entropy:

$$D_{KL}(P_{\mathbf{c}}||P_{\mathbf{s}}) = \sum_{\boldsymbol{\chi}\in\mathcal{X}} P_{\mathbf{c}}(\boldsymbol{\chi}) \operatorname{ld} \frac{P_{\mathbf{c}}(\boldsymbol{\chi})}{P_{\mathbf{s}}(\boldsymbol{\chi})}. \tag{2.1}$$

The Kullback-Leibler divergence can be seen as a measure of the difference between the two probability mass functions $P_{\mathbf{c}}, P_{\mathbf{s}}$. Whenever both distributions are equal, the KL divergence is zero. The more different both distributions are, the larger is their KL divergence, i. e., $D_{KL}(P_{\mathbf{c}}||P_{\mathbf{s}}) \geq 0$.

2.2.1.2. Definition of Security

In his work, Cachin defines a steganographic scheme as *perfectly secure*, if and only if $D_{KL}(P_{\mathbf{c}}||P_{\mathbf{s}}) = 0$ [13]. In this case, the stego and cover objects are identically distributed. Consequently, it is impossible for an attacker to distinguish between them.

Whenever we find $D_{KL}(P_{\mathbf{c}}||P_{\mathbf{s}}) \leq \epsilon$, the steganographic scheme is called ϵ-*secure*. Note that the smaller ϵ or the closer both distributions are, the lower the probability of being detected. Thus, by minimizing the KL divergence of cover and stego distribution, the security of the steganographic scheme can be increased.

2.2.1.3. Two Types of Error

As mentioned before, the attacker's problem can be seen as a decision problem, i. e., an attacker has to decide, whether a given object contains a secret message or not. Thus, his objective is to perform binary hypothesis testing and assume hypothesis H_0 if the observed data is a cover. Whenever the observed data seems to be suspicious, he assumes H_1.

Generally, decision-theoretic-based metrics can be used as a measure of steganographic security. Note that a measure of steganographic security can also be seen as a measure of steganalytic performance, i. e., the quality of an attack.

Given that the detector's response is binary, i. e., the output is a decision (cover or stego), two kinds of errors occur when applying hypothesis testing in practice (see Table 2.1). An *error of the first kind* occurs when the attacker decides for hypothesis H_1 when H_0 is true instead. This kind of error is also denoted as false positive or False Alarm (FA) since it occurs whenever the steganalyst misclassifies a plain cover as stego object.

The *second type of error* occurs whenever the steganalyst fails to detect a stego object, i. e., H_0 is accepted while H_1 is true instead. This kind of error is called Missed Detection (MD) or false negative. Note that the probability of false alarms (Type I error) is denoted as P_{FA} and the probability of missed detection (Type II error) is denoted as P_{MD}.

Table 2.1.: Two Types of Error.

Sent as \ Decided as	Cover	Stego
Cover		Type I Error: False Positive $\rightarrow P_{FA}$
Stego	Type II Error False Negative $\rightarrow P_{MD}$	

In general, the higher the error probabilities P_{FA} and P_{MD}, the better the security of a steganographic scheme, i. e., the worse the decisions made by a steganalyst. However, it is not possible to minimize both errors.

2.2.1.4. Problem Discussion

The adequacy of the information-theoretic model of steganographic security for real-world steganography depends on the assumption that there is a probabilistic model of the cover. However, since the cover data is high-dimensional and the attacker has access only to a limited number of observations, it is still not clear how to estimate and describe $P_{\mathbf{c}}$ and $P_{\mathbf{s}}$ in practice [8].

As mentioned previously, steganographic covers are high dimensional objects which can be modeled as random variables. Nevertheless, the sender, as well as the attacker, has access only to a limited set of statistical quantities derived from some cover or stego objects. Thus, the attacker has to deal with simplified models describing only some statistical properties of the observed data. Furthermore, because he is observing empirical data, it will never be possible to completely describe the cover source.

Note that the embedding distortion can be arbitrary considering this approach as long as $P_{\mathbf{s}}$ is as close as possible to $P_{\mathbf{c}}$. Consequently, perfectly secure steganographic schemes are based on the approach of cover synthesis and cover selection.

However, a cover is mainly selected from a source of covers and is afterwards modified to embed the secret message. Thus, the distortion introduced during embedding by cover modification has to be kept as small as possible in order to maintain $P_{\mathbf{s}}$ as close as possible to $P_{\mathbf{c}}$ and thus to get $D_{KL}(P_{\mathbf{c}}||P_{\mathbf{s}})$ to be as close as possible to zero.

Note that perfect security is only possible for artificial covers as in cover synthesis or cover selection [22]. For a perfectly secure stego scheme, we find $D_{KL}(P_{\mathbf{c}}||P_{\mathbf{s}}) = 0$. In this case, no detector does exist and the secure capacity is linear. For imperfect schemes, however, we find $D_{KL}(P_{\mathbf{c}}||P_{\mathbf{s}}) = \epsilon$ and a detector does exist. As mentioned before, the secure capacity of such a scheme is related to the root rate [66, 26, 22].

2.2.2. Complexity-Theoretic Approach

Generally, the *information-theoretic definition* of security [13] is based on two rather strong assumptions. Firstly, it is assumed that covers can be described by a probability distribution $P_{\mathbf{c}}$. However, as mentioned above, cover data in real scenarios is high dimensional. Therefore, descriptions can be based only on simplified models describing only some statistical properties of the observed data.

Secondly, the approach completely ignores complexity issues. It addresses only the possibility of constructing an attack rather than its practical realization. Since the security of a scheme depends on parameters such as key length, the designer of a steganographic scheme should make sure those parameters are large enough when choosing them. However, the computational complexity grows exponentially with the key length. Thus, an attacker with polynomial bounds in his resources will not be able to realize a theoretically possible attack.

The basic idea of defining steganographic security based on complexity-theoretic principles takes theses facts into account and defines a steganographic scheme as secure if the stego object is computationally indistinguishable from the cover object [63, 65]. Therefore, these models of steganographic security take into account the limited computational power of the attacker.

Within this section, we give a short overview of two basic approaches based on complexity-theoretic principles [63, 65]. Note that the complexity-theoretic definitions of steganographic security of Hopper et al. [63] and Katzenbeisser et al. [65] were independently proposed in 2002 and share some basic ideas.

In contrast to the information-theoretic approach, the assumption that an attacker knows the distribution $P_{\mathbf{c}}$ is replaced with the weaker assumption of the existence of an oracle O. This oracle simply samples from the set of covers according to their distribution.

Furthermore, instead of employing hypothesis testing, the security of the steganographic scheme is modeled as a *probabilistic game* between a judge and a warden. In preparation for the game, the warden draws samples of the oracle O. Furthermore, an embedding oracle is implemented as a black box seeded with an unknown stego key.

The game is formulated as follows: Based on the output of the two oracles (O and the embedding oracle), the warden has to distinguish between both of them. The advantage of the warden is defined as the probability of correct decision minus 0.5 (probability of a correct guess). The steganographic scheme is considered secure if the warden's advantage is negligible.

2.2.2.1. Complexity-Theoretic Approach According to Hopper et al. [63]

In their approach, Hopper et al. consider the communication channel in a rather general way. They describe it as the distribution over sequences of timestamped bits, "timestamped" because the model also considers steganographic methods that use timing between individual bits for message hiding.

In this model, the oracle O is allowed to make partial draws from the channel. Therefore, it samples the channel distribution based on the channel history. Consequently, the oracle $O_{h,o}$ is able to provide the next h timestamped bits based on the previous o bits.

Furthermore, Hopper et al. model the steganographic scheme as a pair of probabilistic algorithms $\mathrm{PEmb}(), \mathrm{PExtr}()$ rather than as mapping functions $\mathrm{Emb}()$ and $\mathrm{Extr}()$ as so far.

Based on the input of a stego key $\mathbf{k}$, a secret message $\mathbf{m}$, the history of h timestamped bits and the oracle $O_{h,o}$, the output of the embedding algorithm is a sequence of l blocks of o timestamped stego bits.

Furthermore, the warden has access to $O_{h,o}$. He is allowed to make as many queries as he wishes. Based on these assumptions, the security of the system is modeled on the following game:

A judge prepares two oracles. The first oracle, called embedding oracle, is seeded with a fixed stego key and is implemented as a black box PEmb($\mathbf{h}, \mathbf{k}, \mathbf{m}$). Furthermore, the judge prepares a second oracle, which also has two inputs defined as $O(\mathbf{h}, \mathbf{m})$.

Now the judge randomly selects one of the oracles and gives it to the attacker. The attacker has to make a decision as to whether he is observing the output of the embedding oracle or the cover oracle O. Hopper et al. define the advantage against a steganographic scheme and ϵ-insecurity on the basis of the attacker's ability to correctly identify the oracle using polynomial-complexity calculations.

2.2.2.2. Complexity-Theoretic Approach According to Katzenbeisser et al. [65]

In the approach of Katzenbeisser et al. [65], the judge prepares in a first step the so-called structure evaluation oracle PEmb($\boldsymbol{\chi}^{(\mathbf{0})}, \mathbf{k}, \mathbf{m}$) (embedding oracle), where $\mathbf{k}$ is a randomly chosen stego key. Note that the embedding oracle is implemented as a black box. As a result, the oracle returns a stego object containing the hidden message $\mathbf{m}$ in cover $\boldsymbol{\chi}^{(\mathbf{0})}$ using key $\mathbf{k}$ for embedding.

In a second step, the judge gives the attacker both the embedding oracle and the cover oracle O. The attacker is now allowed to query O an arbitrary but finite number of times and thus is allowed to obtain a number of covers. Based on these covers, he is able to describe the distribution of covers he has observed.

Furthermore, he can also query the embedding oracle with arbitrary messages and covers and thus obtain several stego objects. Generally, the attacker is not limited in the number of queries, but he performs only polynomial-complex calculations.

The probabilistic game itself is described as follows: The cover oracle O is queried twice, receiving $\boldsymbol{\chi}_1$ and $\boldsymbol{\chi}_2$. Now, the judge selects a random message $\mathbf{m}$ and determines PEmb($\boldsymbol{\chi}_2, \mathbf{k}, \mathbf{m}$). Afterwards, the judge flips a coin in order to decide whether to give the cover $\boldsymbol{\chi}_1$ or the stego object PEmb($\boldsymbol{\chi}_2, \mathbf{k}, \mathbf{m}$) to the attacker.

Now, the attacker performs a probabilistic test in order to decide whether he has observed a cover or a stego object. If the attacker has some systematic advantage in distinguishing between both objects, the steganographic scheme obviously leaks information. The advantage of the attacker can be expressed by the probability of a correct decision minus 0.5 (probability of a correct guess). According to Katzenbeisser et al., the steganographic scheme is considered secure for oracle O if the attacker's advantage is negligible.

2.2.3. A Brief Introduction to Steganalysis

In practical scenarios, such definitions of steganographic security play only a minor role. Generally, a steganographic scheme is considered broken whenever an attacker is able to distinguish between cover and stego with probability better than random guessing. Note that it is not necessary to gain access to the hidden content itself.

Consequently, steganalysis can be seen as the complement to steganography. Whereas the goal of steganography is to achieve a high embedding capacity while maintaining a high security, the goal of steganalysis is to detect the presence of the hidden message.

Therefore, in practice, the security of a steganographic scheme is defined by its power to defeat detection, i. e., to resist steganalytical investigations. An attack is considered successful if the steganalyst's decision problem can be solved with higher probability than random guessing.

Generally, there are *two types of steganalysis attacks*: targeted and blind attacks. Both types of attacks try to distinguish between cover and stego data based on some statistical quantities extracted from the observed data.

For *targeted steganalysis*, these quantities are usually designed by analyzing specific traces of embedding, i. e., they are chosen based on the knowledge of the embedding algorithm and are thus targeted at a specific embedding operation. Thus, in order to find appropriate features for the steganalytical investigation, an attacker has to find quantities that will be changed during embedding.

Examples for targeted attacks are, e. g., the histogram attack by Westfeld [98] or the Sample Pair Analysis by Fridrich et al. [39]. These attacks aim to detect LSB embedding-based schemes. Furthermore, targeted attacks based on calibration are known such as, e. g., the attack on Outguess [40] and F5 [41].

On the contrary, features for *blind steganalytical attacks* must be designed in a way so as to be able to detect every possible steganographic scheme, including future schemes. Therefore, these features have to be designed in a heuristic manner in order to be sensitive to typical steganographic changes. Examples for blind steganalyzers can be found, e. g., in [33, 68, 91].

Just as with the decision problem in Cachin's hypothesis testing (see Section 2.2.1), the attacker must be aware of the two kinds of errors (see Table 2.1), namely the misclassification of a cover as a stego object (also denoted as False Alarm or False Positive) and the missed classification, i. e., when he falsely classifies a stego object as cover (also denoted as False Negative).

Since we find in general that the higher the error probabilities, the better the security of a steganographic scheme, the performance of a steganalytical method can be described based on the probabilities of these two errors. A common way to visualize the characteristic relation between the two error rates is the so-called *Receiver Operating Characteristics (ROC) curve*.

It allows comparisons of the security of alternative steganographic schemes for

a fixed detector, or conversely, comparisons of detector performance for a fixed steganographic scheme. An example for an ROC curve is shown in Figure 2.2.

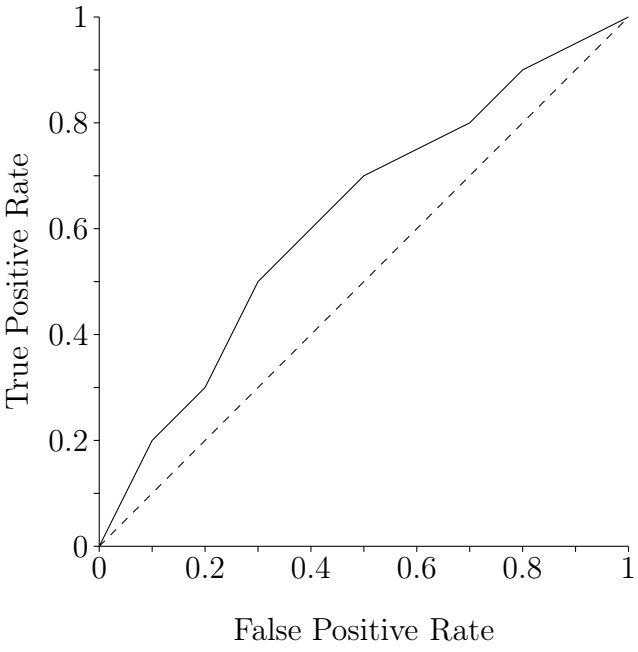

Figure 2.2.: Example of a ROC Curve.

Generally, a curve on the diagonal line would signal perfect security, i. e., the detector's results are no better than random guessing. In order to be able to compare different steganalytical methods, different metrics based on the ROC curve are calculated. Firstly, as the most frequently used metric, the *Area Under the Curve (AUC)* describes the detector's reliability as the area under the curve minus the triangle below the 45 degree line scaled to the interval $[0, 1]$. A value of 1 stands for a perfectly detectable steganographic scheme, whereas a value of 0 signifies the undetectability of the steganographic method under investigation. Note that this means only the undetectability of the scheme considering the investigated steganalytical method, not undetectability in general.

Other metrics for comparing different steganalytical approaches include, e. g., the *False Positive rate at 50 percent detection rate (FP50)* and the *Equal Error Rate (EER)*, i. e., the rate where $P_{MD} = P_{FA}$.

Among the list of ROC-based metrics, there is no unique best option [8]. Each metric suffers from specific weaknesses. For instance, AUC aggregates over practically irrelevant intervals of the curve, and EER and FP50 reflect the error rates for a single arbitrary value. For more details see, e. g., [8, 36].

2.2.4. Conclusion

To summarize the considerations about defining the security of steganographic schemes, we would like to recall that the information-theoretic approach is based on the assumption that the cover source distribution is known by an attacker. Since an attacker has access only to some statistical properties of a finite number of covers, it is not possible to comprehensively describe the cover source.

Generally, describing the cover using a simplified model, we obtain a concept of describing security with respect to a model. Thus, the steganographic scheme is considered secure within the model. This consideration also applies for practical steganalysis tools.

Consequently, the information-theoretic definition of security in steganography is the most widely accepted approach describing the security of steganographic schemes and is also the basis of this work. Note that in this approach the distortion introduced by cover modification during embedding has to be kept as small as possible in order to maintain $P_{\mathbf{s}}$ as close as possible to $P_{\mathbf{c}}$ and thus, achieving $D_{KL}(P_{\mathbf{c}}||P_{\mathbf{s}})$ as close to zero as possible.

2.3. Implications for the Practical Design of Steganographic Schemes

Based on the definition of steganographic security, the goal for constructing good steganographic schemes is clear - to preserve the statistical distribution of the cover. Since a digital image contains high dimensional empirical data, an accurate description based on statistical models is not possible. It is, however, possible to use the model of steganographic security based on the information-theoretic approach, to introduce a general approach for improving the security of known steganographic schemes.

According to Fridrich et al. [51], the security[2] is mainly influenced by several degrees of freedom that are interdependent but are mostly studied in an isolated manner:

- the *type of cover media*, whose properties are known to the attacker according to Kerckhoff's principle,
- the *method for selecting the positions* that might be modified,
- the *embedding operation* that is applied to the individual cover elements to embed a message, and
- the *number of embedding changes*.

[2]In practice, security is often understood as the inability to practically construct a reliable detector.

Even if all four aspects are implicitly a part of the steganographic scheme, in this thesis we focus on the last point, the number of embedding changes. This is motivated by results based on steganalytical investigations [50, 71]. These investigations show that it is advantageous to modify as little as possible and only in inconspicuous parts of the cover. It is assumed that the security of a steganographic scheme is proportional to $1/d_\rho(\mathbf{cover}, \mathbf{stego})$, where $d_\rho()$ is a measure to express the distortion introduced during embedding.

Consequently, the security of a steganographic scheme also depends on the relative message length α, i. e., the more bits have been embedded the more distortion is on average assumed. Thus, in order to increase the steganographic security, it seems reasonable to minimize the number of changed elements.

In the literature, we find different design principles or strategies for the design of steganographic schemes, e. g., [29, 8, 36]. In this thesis, we will adhere to the design principles of Fridrich, which shall be outlined in the following section.

2.3.1. Design Principles According to Fridrich

Within their work, Fridrich et al. propose four different heuristic principles that can be applied to decrease the KL divergence between cover and stego thus increasing steganographic security [36]: model-preserving steganography, making embedding mimic natural processing, steganalysis-aware steganography and Minimum-Embedding-Impact Steganography. Note that these principles are already mentioned in [71].

2.3.1.1. Model-Preserving Steganography

The basic idea of *model-preserving steganography* is to preserve the statistics of the cover. This principle follows directly from the definition of steganographic security. However, since cover objects are high dimensional, a simplified model of covers is formulated and the embedding process is designed to preserve this model. Steganographic systems preserving the simplified cover statistics are undetectable within the model. Nonetheless, such systems can be easily attacked by identifying a statistical quantity that is not preserved.

Within this approach, there are two main directions: *statistical restoration* and *model-based steganography*. A statistical restoration approach consists of a two pass procedure. Look, for example, at Outguess [81]. Within this approach, a message is embedded into a Joint Photographic Experts Group (JPEG) image by slightly modifying the quantized Discrete Cosine Transformation (DCT) coefficients. In order to preserve the histogram of all DCT coefficients, the steganographic algorithm is a two pass procedure. Embedding takes place in the first pass. In the second pass, corrections are made in order to match the histogram of the stego with the histogram of the cover.

Contrary to this two step approach, model-based steganography [84] fits a parametric model by means of the given sample data and tries to preserve this model during embedding. Thus, there is no need for a correction step.

One main disadvantage of this approach is that an attacker simply has to find a statistical quantity that is not preserved during embedding, i. e., one which is disturbed. Note that the additional changes of the restoration phase often cause artifacts to make steganalysis even simpler [9]. Then again, preserving a more complex model is in general not a solution to this problem.

2.3.1.2. Mimic Natural Processing

To overcome the problems of the approach mentioned above, *mimic natural processing* seems reasonable, i. e., to masquerade the embedding as a natural process, such as noise superposition during image acquisition. Therefore, the effect of embedding is indistinguishable from natural processing and the stego images should stay compatible with the distribution of cover images [38, 32].

An example of this class of steganographic algorithms is stochastic modulation [38], which tries to mimic the image acquisition process. The basic assumption of this approach is that the process of image acquisition is affected by multiple noise sources. The message is thus embedded by superimposing quantized independent and identically distributed (iid) noise with a given distribution.

A more precise model of an image acquisition process is proposed by Franz et al. in [32]. This approach describes the noise introduced during the scanning process and models the embedding process accordingly.

2.3.1.3. Steganalysis-Aware Steganography

A completely different approach is *steganalysis-aware steganography*, in which the designer of a stegosystem focuses on making the impact of embedding undetectable using existing steganalysis schemes. He is thus using known steganalysis tools as guidance for the design of a steganographic scheme.

Examples of this approach include, e. g., ± 1 embedding, also called LSB matching and was designed to overcome the weakness of LSB embedding [39, 17]. Another example is the steganographic algorithm F5 by Westfeld [96], which was originally designed to overcome the histogram attack [98]. This algorithm tries to preserve the shape of the histogram, which, after embedding, looks similar to the histogram of an image compressed using a lower quality factor.

2.3.1.4. Minimum-Embedding-Impact Steganography

The fourth class of steganographic algorithms, according to Fridrich [36], is *Minimum-Embedding-Impact Steganography*. The designer of such a steganographic scheme attempts to embed by minimizing the total embedding distortion. Therefore, the sender first assigns the cost of making an embedding change to

each element of the cover and then embeds the secret message while minimizing the total costs of embedding.

Contrary to the first three approaches where the sender aims at preserving a chosen model of the cover, the sender embeds by minimizing a heuristically defined embedding distortion within this approach. Whereas undetectability with respect to the chosen model can be assumed for the first three approaches, this is not possible for Minimum-Embedding-Impact Steganography.

Although the relationship between distortion and security is far from clear [25], embedding while minimizing a distortion function is easier to solve than embedding with constraints. Moreover, as stated in the literature, minimizing the distortion seems to lead to more secure stego systems [50, 71]. Note that the approaches become similar when minimizing is defined as norm between features as in [79].

Seeing that only selected parts of the cover were used for embedding, this approach can be seen as a non-shared selection channel. Thus, the receiver does not have access to the selection rule and has to find a way to extract the hidden message.

As a solution to this problem, channel coding techniques, namely syndrome coding, were introduced in the embedding process, e. g., [16, 44]. By means of these techniques, the sender is free to choose embedding positions arbitrarily without the need to share the selected positions with the recipient.

2.4. An Advanced Model of the Steganographic Scheme

This thesis focuses on the design principle of Minimum-Embedding-Impact Steganography [36]. The overall goal thereby is to reduce the detectability of embedding. Intuitively, we can see that the security of a steganographic scheme depends on the relative message length α. For a small relative message length α, we need less changes, hence less statistically detectable artifacts are introduced.

Therefore, concepts of coding theory, e. g., syndrome coding as a technique to improve steganographic security by minimizing the overall embedding distortion introduced during embedding are investigated in this thesis.

The goal, however, is not to design a whole steganographic scheme (see Figure 2.1), i. e., a complete embedding process. Instead, the specific part of the embedding algorithm will be more closely examined where the stego bit string $\mathbf{B}$ is chosen. Therefore, we introduce an advanced model of the steganographic scheme in order to clarify the basic concept of this thesis.

2.4.1. The Sender

Generally, the challenge for the sender is to modify the cover $\boldsymbol{\chi}^{(\mathbf{0})}$ in such a way that the receiver is able to extract the hidden message from the resulting stego data $\boldsymbol{\chi}^{(\mathbf{m})}$.

In this thesis, the embedding process Emb() is modeled in a more specific way compared to the general model of a steganographic scheme as given in Figure 2.1. In this model, following Günther [58], the embedding process is divided into 3 steps (see Figure 2.3): the pre-processing step Π, the embedding step Θ and the cover-modification step Γ:

- In the *pre-processing step* Π, a cover object $\boldsymbol{\chi}^{(\mathbf{0})}$ is processed to extract a bit sequence $\mathbf{A}$ of length N,
- In the *embedding step* Θ, this sequence $\mathbf{A}$ is modified to embed the message $\mathbf{m}$, resulting in sequence $\mathbf{B}$, and
- In the *cover-modification step* Γ, the modifications on the bit string are translated to the cover to obtain the stego object $\boldsymbol{\chi}^{(\mathbf{m})}$.

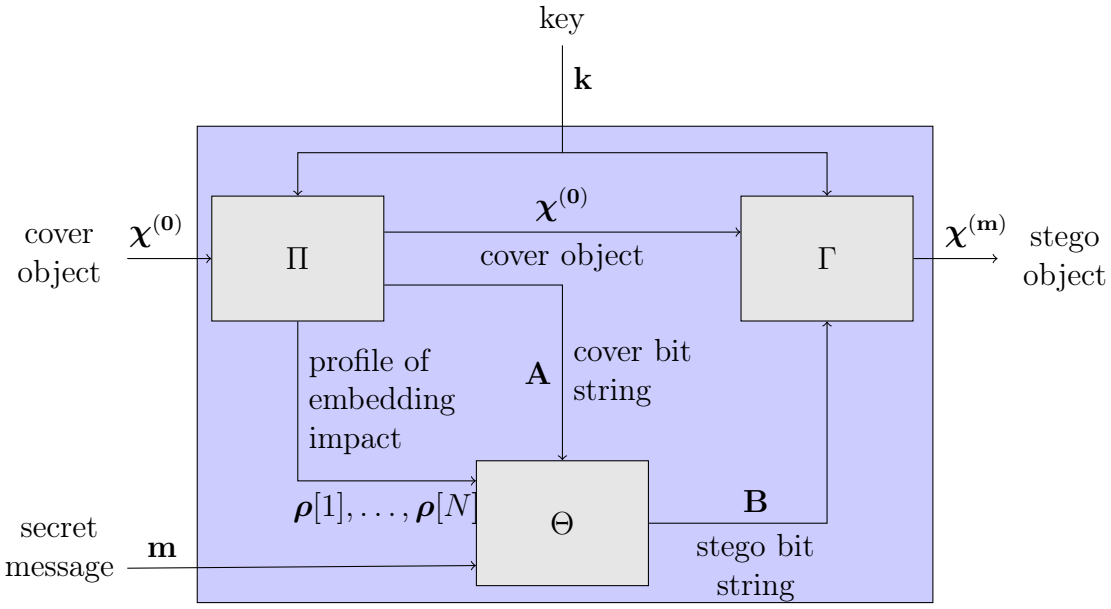

Figure 2.3.: Schematic Diagram of the Sender, Including the Pre-Processing Step Π, the Embedding Step Θ and the Cover-Modification Step Γ According to [58].

This thesis focuses on the second step, the embedding step Θ, i. e., how to reasonably extract an appropriate bit string $\mathbf{A}$ from the cover data is not considered. This is indeed an important point, the answers to which go far beyond the scope of this thesis. However, since the embedding step Θ is influenced by the output of the pre-processing step Π, a short description of this step will also be given. The last step, the cover-modification step Γ, does not form a part of this focus either. This step, of course, is closely connected to the pre-processing step. The goal of this last step of the embedding process is to apply the necessary modifications to the cover data in order to achieve the stego data which will be transmitted via the channel.

2.4.1.1. Pre-Processing Step Π

The cover object can be data of various formats, e. g., Bitmap Image (BMP), JPEG, MPEG-1 Audio Layer 3 (MP3) or Moving Picture Experts Group (MPEG), and can be seen as a sequence of N discrete symbols. In the pre-processing step Π, a bit string $\mathbf{A}$ is selected from these discrete symbols by means of a symbol assignment function. Generally, the pre-processing step can be seen as an interface between the cover object and the embedding step.

To give a naive example of such a *bit selection function*, it is assumed that the cover is given in a format that can be further sub-classified. For images such as BMP, pixels come to mind, or DCT coefficients for JPEG images. Audio files can be further sub-classified in samples. Consequently, each element of the cover (pixel, DCT coefficient, or audio sample) can be represented by t bits.

A popular approach for a bit selection function $\pi(x)$ is the selection of the Least Significant Bit (LSB)s of the cover $\boldsymbol{\chi}^{(\mathbf{0})}$. Consequently, the selected bit string consists of binary digits:

$$\mathbf{A}[i] = \boldsymbol{\chi}^{(\mathbf{0})}[i] \bmod 2. \tag{2.2}$$

Even if the approach of selecting the least significant bits of all cover elements is popular, it might be advantageous to incorporate knowledge of the statistical properties of the cover into the bit selection. Note that when speaking of cover and stego, this thesis refers to the bit strings $\mathbf{A} \in \mathbb{F}_2^N$ and $\mathbf{B} \in \mathbb{F}_2^N$ of length N, i. e., cover properties are not considered directly.

Generally, when embedding a message $\mathbf{m} \in \mathbb{F}_2^{|\mathbf{m}|}$, the sequence $\mathbf{A}$ is changed into a sequence $\mathbf{B}$. In order to determine which position should be changed and which should remain untouched, a function that determines the *embedding impact* is needed.

The impact of an embedding change in the i-th element of the cover (either pixel, DCT coefficient, or audio sample) is measured by a non-negative number $\boldsymbol{\rho}[i]$. This value should express how strong a modification of this particular position of the cover would influence the probability of an attacker detecting the embedding.

Consequently, the embedding impact should be designed to correlate with the statistical detectability of the embedding changes. Doing this for each element of the cover, the profile of embedding impact $\boldsymbol{\rho}[1], \ldots, \boldsymbol{\rho}[N]$ is achieved.

In general, the impact for each element may depend on different parameters. For example, in images, the local texture may be taken into account to reflect the fact that embedding changes in textured or noisy areas are more difficult to detect than changes in smooth regions of the cover. More information about the profile of embedding impact can be found in Section 2.5.

2.4.1.2. Embedding Step Θ

Within the embedding step Θ, the sender has to determine the stego bit string $\mathbf{B}$ containing the secret message $\mathbf{m}$ to be communicated.

Typically, embedding operations function on individual elements $\mathbf{A}[i]$ in the bit string $\mathbf{A}$. The simplest and probably the oldest embedding operation in digital steganography is the *LSB replacement* also called LSB embedding. This operation is based on the (incorrect) assumption that the least significant, i. e., the right-most bit in digitized signals is purely random. Thus, it can easily be replaced by the secret message:

$$\mathbf{B}[i] = \mathbf{m}[i]. \tag{2.3}$$

However, this embedding operation is weak, since LSBs are generally not purely random. Many steganalytical investigations using this effect have been proposed, e. g., the histogram attack of Westfeld [98].

Almost as simple as LSB replacement - though much more difficult to detect - is LSB matching. This embedding operation, also called ± 1 *embedding*, was introduced by Sharp [89]. In contrast to simply replacing the least significant bit, this embedding operation applies incrementing or decrementing with equal probability, whenever the message bit $\mathbf{m}[i]$ is not equal to $\mathbf{A}[i]$:

$$\mathbf{B}[i] = \begin{cases} \mathbf{A}[i] & \text{for } \mathbf{A}[i] = \mathbf{m}[i] \\ \mathbf{A}[i] \pm 1 & \text{otherwise.} \end{cases} \tag{2.4}$$

The random sign of the embedding change (incrementing and decrementing with equal probability) avoids structural dependencies. Many steganographic algorithms are based on this embedding operation.

This embedding operation, however, does not take into account the impact of embedding changes. Remember that by means of the profile of the embedding impact, the sender is able to determine the overall embedding impact when modifying sequence $\mathbf{A}$ into $\mathbf{B}$ in order to embed the message $\mathbf{m}$. Usually, there are different possibilities for choosing a sequence $\mathbf{B}$, including the secret message. Therefore, the sender has the possibility to choose one out of several possible

sequences in the embedding step Θ according to a given profile of the embedding impact $\boldsymbol{\rho}[1], \ldots, \boldsymbol{\rho}[N]$.

Generally, the goal within Minimum-Embedding-Impact Steganography is to minimize the *overall embedding impact* defined as:

$$d_{\rho}(\mathbf{A}, \mathbf{B}) = \sum_{i=1}^{N} \rho[i]|\mathbf{A}[i] - \mathbf{B}[i]|. \tag{2.5}$$

Note that this distortion measure is a strength measure of embedding changes. The definition of the overall embedding impact given here, implicitly assumes that the embedding impact of neighboring elements is independent. It is defined as the sum of measures at individual elements of the cover, i. e., no dependencies are considered. This is a simplification since the embedding modifications can usually interact among themselves. For example, making two changes to adjacent elements might be more detectable than making the same changes to two elements far apart from each other.

In general, $\boldsymbol{\rho}[i]$ may depend on the neighborhood of a cover element or some side-information, such as the quantization error. Alternatively, $\boldsymbol{\rho}[i]$ may be derived from theoretical considerations as $\boldsymbol{\rho} \propto -\ln p[i]$ [37], where $p[i]$ is the probability with which $\mathbf{A}[i]$ should be changed; $p[i]$ derived from a cover model that minimizes the root rate [22], see Section 2.1.3. Thus, $\boldsymbol{\rho}$ may be highly non-uniform [24]. A steganographic approach considering such a model is HUGO [79], working with an additive approximation and applying a model-correction step.

For the sake of simplification, in this thesis it is assumed that the density of embedding changes is low. In this case, the assumption of independence is plausible, because the distance between modified cover elements will usually be large. Consequently, the embedding changes will not interfere much.

2.4.1.3. Cover-Modification Step Γ

In the cover-modification step Γ, the last step of the embedding process, the cover-modification must be applied. Based on the necessary embedding changes determined in the embedding step, i. e., based on $\mathbf{B}$, the sender has to translate the modifications to the cover $\boldsymbol{\chi}^{(\mathbf{0})}$ in order to achieve $\boldsymbol{\chi}^{(\mathbf{m})}$ containing the secret message $\mathbf{m}$. Of course, this cover modification requires knowledge of the pre-processing step.

At this point, the simple bit selection function presented in Section 2.4.1.1, in which the LSBs of the cover $\boldsymbol{\chi}^{(\mathbf{0})}$ were selected to obtain the cover bit string $\mathbf{A}$: $\mathbf{A}[i] = \boldsymbol{\chi}^{(\mathbf{0})}[i] \bmod 2$ should be brought to mind. Based on this selection function, the sender has to apply the following operation within the cover-modification step:

$$\boldsymbol{\chi}^{(\mathbf{m})}[i] = \left[\left[2\left\lfloor\frac{\boldsymbol{\chi}^{(\mathbf{0})}[i]}{2}\right\rfloor\right]_2 \oplus \mathbf{B}[i]\right]_{10}. \tag{2.6}$$

2.4.2. The Receiver

The scheme for the receiver following Günther [58] can be seen in Figure 2.4. The challenge for the receiver is to extract the secret message from the stego data. Therefore, two processing steps are required: the pre-processing step Π and the extraction step Λ. Like the sender, the receiver also has to extract the bit string $\mathbf{B}$ from of the stego object $\boldsymbol{\chi}^{(\mathbf{m})}$. Afterwards, the extraction step Λ can be applied to obtain the embedded secret message. It is not necessary for the receiver to reconstruct the cover data itself.

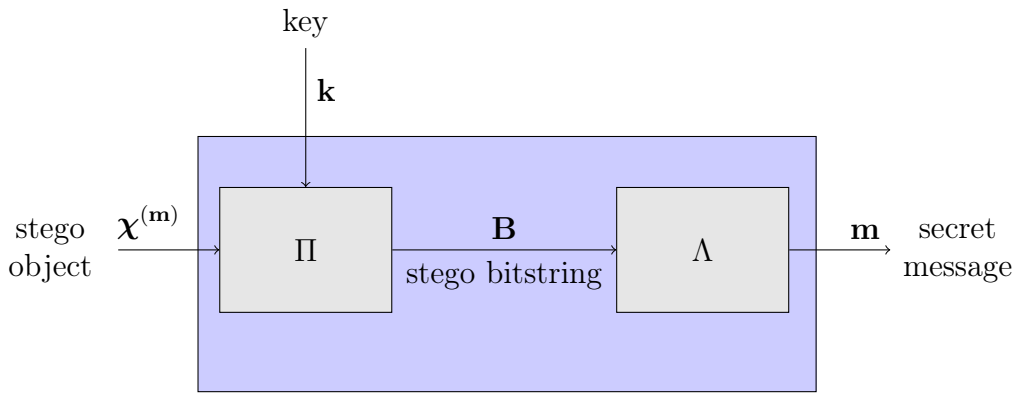

Figure 2.4.: Schematic Diagram of the Receiver, Including the Pre-Processing Step Π and the Extraction Step Λ According to [58].

2.5. Profile of Embedding Impact

Steganography by cover modification introduces some embedding changes into the cover which may have a different impact on the statistical detectability depending on, e. g., the local context - the properties of the cover element being modified.

For the most part, the impact of embedding is related to the detectability of an embedding change at that particular position of the cover, i.e., the embedding impact is influenced by to the costs of making an embedding change at this position. Assuming that there are no interactions between the costs of different cover elements, e. g., the costs are additive and independent, the overall embedding impact can be seen as a sum of costs for the different positions that have been changed during embedding (see Equation (2.5)).

It can be distinguished between two different profiles of embedding impact $\boldsymbol{\rho}[1], \ldots, \boldsymbol{\rho}[N]$: the uniform profile and the general profile. If a *uniform profile* is defined, i. e., $\boldsymbol{\rho}[i]$ is a constant, the embedding distortion can be measured as the number of changes necessary to embed the secret message.

In a more *general profile*, where $\boldsymbol{\rho}[i]$ is defined for each cover element independently, the severity of changes influences the embedding distortion. The terms uniform vs. general profile are based on Filler [19], who distinguishes between different profiles of embedding impact. The separation, however, can also be based on the different distortion measures resulting from different profiles of the embedding impact as introduced in [8]. Böhme uses the terms *number of changes* and *severity of changes* instead.

It is necessary to recall the principle of minimal embedding impact [36], where the security of the steganographic scheme should be increased by minimizing the Kullback-Leibler divergence between the distributions of cover and stego objects. Note that both profiles require different strategies to minimize the embedding distortion during embedding.

2.5.1. Uniform Profile

Whenever the profile of embedding impact is defined as $\boldsymbol{\rho}[i] = \text{const} \quad \forall i = 1, \ldots, N$, the embedding distortion linearly increases with the increasing number of embedding changes. Therefore, the more $\mathbf{B}$ differs from $\mathbf{A}$, the higher the embedding distortion. Consequently, the overall embedding distortion can be determined by simply counting the number of embedding changes:

$$\begin{aligned} d_\rho(\mathbf{A}, \mathbf{B}) &= \sum_{i=1}^{N} \boldsymbol{\rho}[i] \, |\mathbf{A}[i] - \mathbf{B}[i]| \\ &\propto \mathrm{w}(\mathbf{A} \oplus \mathbf{B}). \end{aligned} \tag{2.7}$$

The embedding distortion is thus simply defined as the Hamming distance between cover and stego. Therefore, minimizing the embedding impact means minimizing the number of embedding changes considering a uniform profile.

Note that considering typical embedding operations like LSB replacement or LSB matching [89] and $\boldsymbol{\rho}[i] = 1$, an average embedding distortion of:

$$d_\rho(\mathbf{A}, \mathbf{B}) = \frac{\alpha}{2} \times N = \frac{|\mathbf{m}|}{2}, \tag{2.8}$$

is obtained depending on the relative message length $\alpha = \frac{|\mathbf{m}|}{N}$. The average embedding distortion $d_\rho(\mathbf{A}, \mathbf{B})$ is calculated from all possible messages and covers for a given steganographic scheme.

2.5.2. General Profile

Besides a uniform profile, where only the number of embedding changes should be minimized during embedding, the severity of changes as a distortion measure, i. e., general profile, could also be considered.

Based on such a general profile, it is possible to apply an adaptive embedding rule, taking into account additional side information. Whenever $\boldsymbol{\rho}[i]$ is highly non-uniform, the sender restricts the changes to a selection channel formed by those elements with appropriate values $\boldsymbol{\rho}[i]$.

Within this approach, a fundamental problem occurs: Since the values $\boldsymbol{\rho}[i]$ are computed from the cover image or some side-information not available to the receiver, the receiver is generally unable to determine the same selection channel. This problem is known as a non-shared selection channel. A solution to this problem is embedding based on syndrome coding, a principle of coding theory considered in this thesis.

In general, the detectability measure should be compatible with results obtained by blind steganalyzers. Thus, the statistical impact of embedding at a given position depends on many factors [37], e. g.,

- the *character of the embedding modifications*,
- their *amplitude*, and
- the *local pixel neighborhood*.

Most frequently used measures are related to the distortion introduced during embedding, i. e., the distance between cover and stego. However, this may be a poor indicator for steganographic security. It is well known that LSB replacement is an operation with the smallest possible distortion[3]. It is still easily detectable. For most other embedding operations, however, the embedding distortion based on the distance between cover and stego is strongly positively correlated with the detectability of the scheme. This claim is supported by the results of $\pm k$ embedding in the spatial domain [53, 62] and attacks by blind steganalyzers [33, 68, 91].

2.5.3. Modeling the Profile of Embedding Impact

In practical steganographic schemes, modeling the profile of embedding impact is mostly done in a heuristic manner. For example, F5 applies syndrome coding based on Hamming Codes with $\boldsymbol{\rho}[i] = 1$ [96]. Moreover, Perturbed Quantization (PQ) use a method to minimize the distortion proportional to the quantization error [44]. Note that Fridrich et. al. state that even a simple distortion model counting the number of changes correlates well with the statistical detectability [50].

Now, an outline on how to model a profile of embedding impact based on [37] will follow. In accordance with the assumption in this thesis, Fridrich et al. state that a bit selection function is applied to select a binary bit string $\mathbf{A}$ from the

[3]Note that the difference between cover and stego is 1 at most.

cover $\boldsymbol{\chi}^{\mathbf{0}}$. Furthermore, it is assumed that each modification of $\mathbf{A}$ into $\mathbf{B}$ directly results in a modification of the associated position in the cover.

Fridrich et al. state that $\boldsymbol{\rho}[i]$ should depend on the local texture. Since changes in textured or noisy areas are less likely to introduce detectable artifacts, they propose to capture this by introducing weights $\boldsymbol{\omega}[i]$. The impact $\boldsymbol{\rho}[i]$ of such a modification is thus modeled as follows:

$$\boldsymbol{\rho}[i] = \boldsymbol{\omega}[i]\ |\mathbf{A}[i] - \mathbf{B}[i]|^{\nu}, \tag{2.9}$$

with a non-negative parameter ν and weight factors $\boldsymbol{\omega}[i] \geq 0$.

2.5.3.1. Modeling a Uniform Profile

Fridrich et al. propose in [37] two different strategies as how to model a uniform profile which is outlined here.

Number of Changes

It is possible to realize different distance measures by means of ν: Whenever $\boldsymbol{\omega}[i] = 1\ \ \forall i$ and $\nu = 1$, $|\mathbf{A}[i] - \mathbf{B}[i]|$ is found to be at most 1. Thus, the impact of embedding is directly proportional to the Hamming distance between $\mathbf{A}$ and $\mathbf{B}$, i. e., the higher the distance the higher the embedding impact.

Therefore, the distortion measure $d_{\rho}(\mathbf{A}, \mathbf{B})$ is equivalent to the total number of embedding changes. In this case the goal of the sender is to minimize the number of embedding changes.

Energy of Changes

Another example is $\boldsymbol{\omega}[i] = 1\ \ \forall i$ and $\nu = 2$, where $d_{\rho}(\mathbf{A}, \mathbf{B})$ is equivalent to the energy of the modifications in this case. Consequently, the goal for the sender is to keep the energy of the embedding modifications low.

2.5.3.2. Modeling a General Profile

In general, the weighting factors $\boldsymbol{\omega}[i]$ may depend on the local environment, reflecting the fact that embedding changes in noisy regions are more difficult to detect. Thus, the factors can be used to price the individual cover elements.

Therefore, Fridrich et al. introduced the concept of Wet Paper Steganography in [44], where each element of the cover is either wet or dry (Section 2.1.2.2), based on the severity of a change concerning the perceptibility or detectability. Within this scenario, it is found that $\boldsymbol{\omega}[i] = 1$ for $i \in |\text{Dry}|$ for some index set Dry and $\boldsymbol{\omega}[i] = 0$ otherwise[4]. The goal of the sender is to embed the message without altering any wet element.

[4]Note that it might be feasible to define $\boldsymbol{\omega}[i] \in [0, 1]$ instead, in order to distinguish between more than two classes of elements.

Usually, the embedding impact $\boldsymbol{\rho}[i]$ may also depend on some side-information about the i-th element available to the sender as in, e. g., Perturbed Quantization (PQ) [44, 48]. In this approach, information about rounding errors influence the embedding distortion $\boldsymbol{\rho}[i]$.

An example application for this approach is embedding while downsampling. The cover is, e. g., a 16 bit per channel TIFF image (48 bit per pixel) and the sender embeds while decreasing the color depth to 8 bit per channel. His goal is to minimize the combined quantization and embedding distortion.

For this scenario, Fridrich et al. derive a profile of embedding impact in [37]. The 16 bit color value is given with $\mathbf{z}[i]$ and $Q = 2^8$ denotes the quantization step for the color reduction. The quantization error at the i-th pixel can be determined as:

$$\mathbf{err}[i] = Q \left| \frac{\mathbf{z}[i]}{Q} - \left\lceil \frac{\mathbf{z}[i]}{Q} \right\rceil \right| \text{ with } 0 \leq \mathbf{err}[i] \leq \frac{Q}{2}. \tag{2.10}$$

In order to embed the message as the LSB of the pixel under investigation, the sender will, in half of the cases, need to round $\mathbf{z}[i]$ to the opposite direction, resulting in an increased quantization error $Q - \mathbf{err}[i]$.

The embedding impact can now be captured with:

$$\boldsymbol{\rho}[i] = Q - 2\,\mathbf{err}[i]. \tag{2.11}$$

Thus, $\boldsymbol{\rho}[i]$ expresses the difference between the two rounding errors, i. e., the difference between the increased quantization error and the quantization error.

Therefore, it would be advantageous for the sender to select those pixels with the smallest $\boldsymbol{\rho}[i]$ for embedding, i. e., the pixels whose values are close to the middle of the quantization intervals, i. e., $\mathbf{err}[i] \approx \frac{Q}{2}$.

2.5.4. Embedding Strategy

When designing a steganographic scheme, the question is, whether it is better to make fewer embedding changes with larger embedding impact or more changes with a smaller impact. The answer depends, according to Fridrich et al., on the profile of the embedding impact [34].

Each time the distortion sharply increases when more than $|\mathbf{m}|$ elements are used, the best strategy might be to embed the message only within the $|\mathbf{m}|$ bits with low $\boldsymbol{\rho}$, so, not using syndrome coding at all.

Whenever considering a uniform profile, the best approach is to use all elements thus applying syndrome coding in order to minimize the total number of embedding changes [34].

In case of Perturbed Quantization (PQ), it is always advantageous to use more elements for embedding than the message length, i. e., to apply syndrome coding.

It has been shown that using 25% more elements than message length is advantageous. In this case, the embedding impact can be decreased by one quarter compared to using only the best α elements [34].

Ideally, the sender should make use of all N pixels and select them with probabilities determined by their embedding impact. This would lead to the optimal embedding strategy [37]. These schemes, however, are most likely not able to reach the maximal possible embedding efficiency e, defined as number of embedded bits per introduced distortion.

Another interesting question is whether it is advantageous to allow fewer but more intense changes. Fridrich et al. therefore analyzed q-ary embedding in the spatial domain. As a result, they found that ternary embedding is the optimal choice for steganography if the goal is to minimize the embedding distortion [49]. The expected distortion per pixel was determined for a wide range of relative message lengths. It can be seen that the minimal distortion is obtained for $q = 3$.

Nonetheless, Filler stated that ternary coding is less effective for a smaller relative message length α as the differences between the embedding efficiency for ternary coding e_3 and binary coding e_2 decreases with decreasing α [20].

Moreover, note that larger values of q allow embedding using a fewer number of embedding changes at the expense of increasing their amplitude. These two trends are working against each other. A general result of Fridrich et al. can be formulated as follows: *It is not beneficial to increase the amplitude of embedding changes in exchange for their smaller number.*

2.6. Conclusion

As mentioned before, this thesis deals with steganographic schemes based on the design principle of Minimum-Embedding-Impact Steganography. Therefore, described and compared are different algorithms based on binary syndrome coding as a technique to improve steganographic security by minimizing the overall embedding distortion introduced during embedding[5]. The overall goal of these approaches is to reduce the detectability of embedding by minimizing the introduced distortion.

However, the goal is not to design a whole steganographic scheme, i. e., a complete embedding process including the pre-processing step Π, the embedding step Θ and the modification step Γ. Instead, the algorithms focus on the embedding step Θ. Thus, the cover bit string $\mathbf{A}$, the message $\mathbf{m}$ and a profile of embedding impact $\boldsymbol{\rho}[1], \ldots, \boldsymbol{\rho}[N]$ are considered as given. Note that the algorithms described in this work are mainly designed for a general profile of embedding impact based on the concept of Wet Paper Steganography. Within this scenario, wet elements

[5]Note that choosing embedding positions is possible for the sender only if the length of the secret message $|\mathbf{m}|$ is smaller than the number of elements of the cover sequence N. In this case, the sender is able to select the elements to be changed in order to embed $\mathbf{m}$.

are excluded from the embedding process, i. e., only dry elements are modified in order to embed the confidential message. Based on this assumption, the total number of embedding changes is suited as a measure for $d_\rho(\mathbf{A}, \mathbf{B})$.

Besides minimizing the introduced distortion, and thereby maximizing the security, the aim is to minimize the embedding complexity while maximizing the success rate and the embedding capacity. Since these are competing goals, however, there is an attempt to find a tradeoff between those four properties.

3. Basic Principle of Embedding

Within this chapter, we introduce the *principle of syndrome coding* which has its origin in Error Correcting Codes (ECC). We use the concept of syndrome coding as a tool to improve steganography by minimizing the distortion introduced during embedding.

In general, syndrome coding is based on a parity-check matrix $\mathbf{H}_{k\times n}$, whereas we introduce three different classes of matrices: *parity-check matrices built according to deterministic rules, parity-check matrices based on stochastic design rules* and *parity-check matrices based on stochastic design rules with constraints*. For the first class, i. e., matrices built on deterministic design rules, we furthermore outline the basics for the underlying codes.

Moreover, we introduce the concept of partitioning the vector space in cosets and define parameters describing the performance of a code suited for steganography, such as the covering radius R, the average number of embedding changes R_a and the embedding efficiency e. We summarize this section with the concept of this work.

3.1. Basic Principle of Embedding with Syndrome Coding

The *concept of syndrome coding* in general can be described as follows: In *coding theory* it is assumed that a codeword $\mathbf{c}$ is transformed during transmission due to a random error pattern $\mathbf{f}$ into a sequence $\mathbf{b} = \mathbf{c} \oplus \mathbf{f}$.

Note that $\mathbf{H}_{k\times n}\mathbf{c}^T = \mathbf{0} \;\; \forall \mathbf{c} \in \mathcal{C}$, where $\mathcal{C}$ is the code described by the parity-check matrix $\mathbf{H}_{k\times n}$, n describes the length of any codeword, and k gives the number of parity bits[6]. Thus, we can use the result of:

$$\mathbf{s} = \mathbf{H}_{k\times n}\mathbf{b}^T = \mathbf{H}_{k\times n}\mathbf{f}^T \tag{3.1}$$

for error detection, where $\mathbf{s}$ is the so-called syndrome. If $\mathbf{s}$ equals $\mathbf{0}$, we assume a correct transmission of the codeword $\mathbf{c}$, and in this way, $\mathbf{b} = \mathbf{c}$. Note that this result is also possible if the superposing error pattern $\mathbf{f}$ is a codeword. Because of linearity, in this case $\mathbf{b} = \mathbf{c} \oplus \mathbf{f}$ is also a codeword, but $\mathbf{b} \neq \mathbf{c}$. Such errors are not detectable at all.

[6]More information about this coherence can be found in Section 3.2.3.

Whenever $\mathbf{s} \neq \mathbf{0}$, the goal for ECC codes is to modify the received sequence $\mathbf{b}$ in a way that $\mathbf{H}_{k\times n}\mathbf{b}_{corr}^T = \mathbf{0}$. For more details on error correction algorithms see, e. g., [100, 11, 76, 70].

However, the goal in *steganography* is different. While for ECC the syndrome $\mathbf{s} = \mathbf{H}_{k\times n}\mathbf{b}^T$ of length k is used for error detection, in steganographic schemes, syndrome coding is used to embed a secret message of length k: The receiver should be able to extract the confidential message simply by calculating $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$.

Therefore, the cover sequence $\mathbf{A}$ is divided into parts $\mathbf{a}$ of length n and the message $\mathbf{m}$ into parts $\mathbf{emb}$ of length k. Usually, the syndrome derived by $\mathbf{H}_{k\times n}\mathbf{a}^T$ is not equal to the message we would like to embed, i. e., $\mathbf{s} \neq \mathbf{emb}$. Thus, we have to deliberately modify the cover part $\mathbf{a}$ in a way that the resulting stego part $\mathbf{b}$ fulfills $\mathbf{s} = \mathbf{emb}$:

$$\mathbf{s} = \mathbf{H}_{k\times n}(\mathbf{a} \oplus \mathbf{f})^T = \mathbf{H}_{k\times n}\mathbf{b}^T = \mathbf{emb}. \tag{3.2}$$

Consequently, when embedding a message part $\mathbf{emb}$ into the cover part $\mathbf{a}$, we have to make modifications. Therefore, embedding by means of syndrome coding forms part of the principle of embedding with cover modifications. However, the impact of modifications varies for each sequence $\mathbf{b}$, fulfilling Equation (3.2).

3.2. Determining Parity-Check Matrices

As described in the previous section, the concept of syndrome coding is based on a parity-check matrix $\mathbf{H}_{k\times n}$. Within this section, we would like to take a closer look at the parity-check matrix $\mathbf{H}_{k\times n}$. Generally, there are different possibilities for handling this matrix, which must be known to sender and receiver of the steganographic system. Generating $\mathbf{H}_{k\times n}$ adaptively, i. e., depending on the cover is complicated since the cover is modified during embedding.

Another, more promising approach is to use the matrix $\mathbf{H}_{k\times n}$ as a secret stego key, i. e., as a parameter to parametrize the steganographic system. However, we believe that using publicly known matrices for syndrome coding might be more suitable in practice. In this case, no matrices or generating algorithms have to be shared. The security of the system can be improved by using a key-based pseudo-random interleaver to pre-process the cover. In this case, only the sender and the receiver, who have knowledge of the key, are able to reproduce the pre-processing and thus, to reproduce the data used for embedding. Another advantage is that sender and receiver can agree on a publicly known and well investigated parity-check matrix $\mathbf{H}_{k\times n}$, whose properties are known. It is thus possible, when choosing the parity-check matrix $\mathbf{H}_{k\times n}$, to make it dependent on the channel used for communication as well as on the properties of the cover data used for embedding.

However, the question still remains as to how to design such a parity-check matrix $\mathbf{H}_{k\times n}$. Generally, there are different approaches: The design may depend on stochastic or on deterministic rules or on some mixture of both. In the following, we will examine in more detail the different approaches for the design of the parity-check matrix $\mathbf{H}_{k\times n}$. Note that the codes mentioned within this section differ only in their description, extracting the message is always done by Equation (3.2).

3.2.1. Parity-Check Matrices Based on Stochastic Design Rules

This section covers matrices built wholly according to stochastic rules. In the literature, these matrices are often called *random matrix*. One example of the usage of these matrices for syndrome-based embedding within a steganographic system can be found in [44]. Within this scenario, the sender and the receiver agree on a secret stego key that is used to generate a pseudo-random binary matrix $\mathbf{H}_{k\times n}$. The probability of 0 and 1 in $\mathbf{H}_{k\times n}$ is the same, i. e., $P(0) \approx P(1) \approx \frac{1}{2}$. An example of such a matrix is given below:

$$\mathbf{H}_{3\times 7} = \begin{pmatrix} 0\,0\,1\,0\,1\,0\,1 \\ 1\,1\,0\,1\,0\,1\,0 \\ 1\,0\,1\,0\,1\,0\,1 \end{pmatrix}.$$

The system of equations used for syndrome coding (see Equation (3.1)) has a solution for arbitrary messages **emb** of length k as long as $\text{rank}(\mathbf{H}_{k\times n}) = k$, i. e., as long as $\mathbf{H}_{k\times n}$ consists of k linearly independent rows. For stochastic matrices this is not always true. The probability for a stochastic binary matrix with equal probability for 0 and 1 being of rank s with $s \leq k$ can be expressed according to [12] by:

$$P_{k,n}(s) = 2^{s(k+n-s)-kn} \prod_{i=0}^{s-1} \frac{(1-2^{i-k})(1-2^{i-n})}{(1-2^{i-s})}. \tag{3.3}$$

Thus, the probability that a binary stochastic matrix $\mathbf{H}_{k\times n}$ can be used for embedding k bits is:

$$\begin{aligned} P_{k,n}(k) &= 2^{k(k+n-k)-kn} \prod_{i=0}^{k-1} \frac{(1-2^{i-k})(1-2^{i-n})}{(1-2^{i-k})} \\ &= \prod_{i=0}^{k-1} (1-2^{i-n}). \end{aligned} \tag{3.4}$$

3.2.2. Parity-Check Matrices Based on Constrained Stochastic Design Rules

Another class of matrices are stochastic matrices with constraints, i. e., these matrices are not completely random, they also fulfill some boundary conditions. One example of such matrices are sparse matrices, where $P(1) = \delta = 1 - P(0)$ with $\delta < 0.5$. Note that the Hamming weight of these matrices $\mathrm{w}(\mathbf{H}_{k\times n})$ grows linearly instead of exponentially depending on the codeword length.

Again, the system of equations used for syndrome coding (see Equation (3.1)) has a solution for arbitrary messages **emb** as long as $\mathrm{rank}(\mathbf{H}_{k\times n}) = k$. Otherwise it is not possible to embed k message bits. With decreasing density, the probability of the matrix $\mathbf{H}_{k\times n}$ being of rank k decreases. However, the rank remains approximately k until the density reaches a critical bound. Afterwards, it quickly falls to 0. According to Fridrich et al. [42], the critical density is close to $\mathrm{ld}k/k$.

An example of the usage of this class of matrices for syndrome-coding based embedding can be found in [43]. In this paper, Fridrich et al. used sparse matrices combined with Luby Transform (LT) Codes [77]. LT codes were introduced as an example of codes with low encoding and decoding complexity. They are based on graphs of logarithmic density.

Fridrich et al. exploit within their work the fact that the LT process provides a method for the fast solution of an over-determined system of equations $\mathbf{A}\mathbf{x}^T = \mathbf{y}$ as long as the weights of columns in $\mathbf{H}_{k\times n}$ follow the Robust Soliton Distribution (RSD). Thus, they use the encoding process of LT codes to fasten the solving of the system of linear equations (Equation (3.1)). This process is called Matrix LT Process. The matrix and, thereby, the graph are generated randomly in such a way that the degrees of encoding nodes follow the so-called RSD [43].

3.2.3. Parity-Check Matrices According to Deterministic Design Rules

This section gives a brief introduction to linear codes, the basis for matrices built according to deterministic design rules. Generally, a *linear code* can be defined as follows:

A q-ary linear[7] (n, l) code can be seen as a linear subspace $\mathcal{C} \subset \mathbb{F}_q^n$ including all possible codewords $\mathbf{c} \in \mathcal{C}$. The number of codewords of the code alphabet $\mathcal{C}$ depends on the parameter l, we find $|\mathcal{C}| = q^l$. In this thesis, we consider $q = 2$, i. e., binary codes.

Note that a code is generally characterized by several parameters: the *code rate* $\beta = \frac{l}{n}$ and its *performance*. The parameter n describes the length of any

[7] A code is denoted as linear if $\mathbb{F}_q$ is a finite field and $\mathcal{C}$ is a vector subspace of $\mathbb{F}_q^n$. A subspace is a subset closed with respect to algebraic operations such as addition by an element of the finite field, i. e., $\forall \mathbf{x}, \mathbf{y} \in \mathcal{C}$, $\mathbf{x} + \mathbf{y} \in \mathcal{C}$.

codeword, and l gives the dimension of the code. The parameter l also gives the number of information symbols, whereas $k = n - l$ gives the number of parity bits and is also denoted as co-dimension of the code.

Generally, the challenge in *coding theory* is to design codes that are able to correct the initial sequence corrupted during transmission. These linear codes, also called ECC, are based on the principle of adding redundancy to the information that has to be transmitted in order to enable the receiver to correct the corrupted sequence.

The *minimum Hamming distance* d_{min} is an important parameter to describe the error detection and error correction performance of a code. The minimum Hamming distance is defined as the minimum distance among all possible distinct pairs of codewords in $\mathcal{C}$:

$$d_{min} = \min_{\mathbf{x},\mathbf{y}\in\mathcal{C},\,\mathbf{x}\neq\mathbf{y}} d_H(\mathbf{x},\mathbf{y}), \tag{3.5}$$

where $d_H(\mathbf{x},\mathbf{y})$ is the Hamming distance. The Hamming distance between two vectors $\mathbf{x}$ and $\mathbf{y}$ of equal length is determined as the number of coordinates for which the corresponding symbols are different:

$$\begin{aligned} d_H(\mathbf{x},\mathbf{y}) &= |\{i : \mathbf{x}[i] \neq \mathbf{y}[i]\}| \\ &= \mathrm{w}(\mathbf{x} \oplus \mathbf{y}), \end{aligned} \tag{3.6}$$

where the Hamming weight $\mathrm{w}(\mathbf{x})$ of a binary vector $\mathbf{x}$ is defined as the total number of elements $\mathbf{x}[i] = 1$, i. e.,

$$\mathrm{w}(\mathbf{x}) = d_H(\mathbf{x},\mathbf{0}). \tag{3.7}$$

Considering only error correction, the *maximum number of correctable errors* can be calculated with $f_k = \lfloor \frac{d_{min}-1}{2} \rfloor$. Of course, a high performance f_k requires a high amount of parity bits k. Consequently, we find the distance d_{min} indicating the suitability of a code for error correction, since a preferably large distance between the codewords is required.

Because of the relevance of a transmission over noisy channels in practice, e. g., mobile phone techniques, most codes have been designed for error correction. In this case, the number of correctable errors is of primary interest.

According to the manner in which redundancy is added to the information, linear codes can be divided into two cases as depicted in Figure 3.1: *block codes* and *convolutional codes* [76]. In contrast to convolutional codes, block codes process the information on a block-by-block basis, treating each block of information bits independently from others. Thus, block coding is a memoryless operation.

In this thesis, we consider different kinds of linear codes, whereas each class of codes can be described by means of a matrix $\mathbf{H}_{k\times n}$. In this chapter, however, we give only a brief description of *group codes* and *cyclic codes* in order to

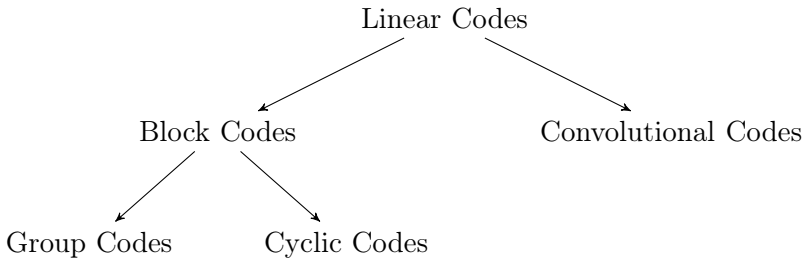

Figure 3.1.: Classification of Linear Codes.

demonstrate the underlying concepts of BCH Codes investigated by us. Note that convolutional codes also fulfill the definitions mentioned for block codes whenever we fix n to a constant number. For more details about codes, coding and decoding principles, we suggest [70, 11, 92, 78, 76, 14].

3.2.3.1. Group Codes

Group codes are a class of block codes fulfilling the algebraic properties of a group[8]. Generally, there are two different ways of describing the membership of any element of $\mathbb{F}_2^n$ to a group code.

All codewords $\mathbf{c} \in \mathcal{C}$ can be generated by a multiplication of elements $\mathbf{c}^* \in \mathbb{F}_2^l$, called source sequence or information word, with the generator matrix $\mathbf{G}_{l\times n} \in \mathbb{F}_2^{l\times n}$, whereas the matrix $\mathbf{G}_{l\times n} = \begin{pmatrix} \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_l \end{pmatrix}$ is formed using l linearly independent basis vectors $\mathbf{g}_i$ and generates or spans the (n, l) linear code, i. e., $\mathbf{G}_{l\times n}$ has to be of rank l.

Thus, a first approach to describe the set of codewords is *based on the generator matrix*:

$$\mathcal{C} = \{\mathbf{c} \in \mathbb{F}_2^n \,|\, \mathbf{c} = \mathbf{c}^*\mathbf{G}_{l\times n},\ \mathbf{c}^* \in \mathbb{F}_2^l\}. \tag{3.8}$$

Note that for any matrix $\mathbf{G}_{l\times n}$ there exist a matrix $\mathbf{H}_{k\times n}$, such that:

$$\mathbf{G}_{l\times n}{\mathbf{H}_{k\times n}}^T = (\mathbf{H}_{k\times n}{\mathbf{G}_{l\times n}}^T)^T = \mathbf{0}_{l\times k}. \tag{3.9}$$

Note that the dot product between an arbitrary row in $\mathbf{G}_{l\times n}$ and any row in $\mathbf{H}_{k\times n}$ is zero. Consequently, when multiplying a sequence $\mathbf{c} \in \mathcal{C}$ with the parity-check matrix $\mathbf{H}_{k\times n}$, we get the zero-vector of length k:

[8] A group is an algebraic structure, i. e., a subset closed with respect to an algebraic operation such as addition. Thus, combining any two of its elements forms a third element. Furthermore, the subset and the operation must satisfy the group axioms, namely closure, associativity, identity and invertibility.

$$\begin{aligned}\mathbf{H}_{k\times n}\mathbf{c}_{1\times n}^{T} &= \mathbf{H}_{k\times n}(\mathbf{c}_{1\times l}^{*}\mathbf{G}_{l\times n})^{T} \\ &= \mathbf{H}_{k\times n}{\mathbf{G}_{l\times n}}^{T}{\mathbf{c}_{1\times l}^{*}}^{T} \\ &= (\mathbf{c}_{1\times l}^{*}(\mathbf{H}_{k\times n}{\mathbf{G}_{l\times n}}^{T})^{T})^{T} \\ &= (\mathbf{c}_{1\times l}^{*}\mathbf{0}_{l\times k})^{T} \\ &= (\mathbf{0}_{1\times k})^{T} = \mathbf{0}_{k\times 1}.\end{aligned} \tag{3.10}$$

Hence, the second approach to describe the (n, l) group code $\mathcal{C}$ is *based on the parity-check matrix* $\mathbf{H}_{k\times n}$: An n-tuple $\mathbf{z}, \mathbf{z} \in \mathbb{F}_2^n$ is a codeword of the code $\mathcal{C}$ generated by $\mathbf{G}_{l\times n}$ if and only if $\mathbf{H}_{k\times n}\mathbf{z}^T = \mathbf{0}$:

$$\mathcal{C} = \{\mathbf{z} \in \mathbb{F}_2^n \,|\, \mathbf{H}_{k\times n}\mathbf{z}^T = \mathbf{0}\}. \tag{3.11}$$

To ensure that Equation (3.11) is not satisfied for elements $\mathbf{z} \in \mathbb{F}_2^n$ with $\mathbf{z} \notin \mathcal{C}$, $\mathbf{H}_{k\times n}$ has to be of rank $n-l$. Consequently, we assume $n = l+k$ in the following considerations.

Examples of this class of codes are, e. g., the Hamming Code (HC) and the Simplex Codes (SC).

Example: $(7, 4, f_k = 1)$ HC

The Hamming Code is a special linear code. For each integer $k \geq 2$, there exists a code with k parity bits and $n = 2^k - 1$.

The parity-check matrix $\mathbf{H}_{k\times n}$ of a Hamming Code is constructed by listing all columns of length k that are pairwise independent. In the binary case considered here, $\mathbf{H}_{k\times n}$ consists of the concatenation of column vectors giving the dual numbers from 1 to n.

The parity-check matrix of the $(7, 4, f_k = 1)$ HC is given with:

$$\mathbf{H}_{3\times 7} = \begin{pmatrix} 1\,1\,1\,1\,0\,0\,0 \\ 1\,1\,0\,0\,1\,1\,0 \\ 1\,0\,1\,0\,1\,0\,1 \end{pmatrix}.$$

■

Note that Hamming Codes are an example of perfect codes, i. e., codes that exactly match the following inequality:

$$2^k \geq \sum_{i=0}^{f_k} \binom{n}{i}. \tag{3.12}$$

For instance, for the $(7, 4, f_k = 1)$ HC, we obtain $2^3 = \binom{7}{0} + \binom{7}{1} = 8$. Every syndrome, excluding the syndrome with weight zero, is mapped to an error pattern $\mathbf{f}$ with $\mathrm{w}(\mathbf{f}) = f_k = 1$.

3.2.3.2. Cyclic Codes

An example of this class of codes fulfilling the properties of a finite field[9] are Bose-Chaudhuri-Hocquenghem (BCH) Codes. Since cyclic shifting of any codeword leads to another codeword, this family of codes is called cyclic. The application of these codes for syndrome coding in steganography was investigated, e. g., by Schönfeld and Winkler in [86, 87]. Note that Hamming Codes, known from Matrix Embedding [16], can also be treated as special BCH Codes.

Generally, a BCH Code, as with any cyclic code, is fully defined by its *generator polynomial* $g(x)$. With the help of $g(x)$, it is possible to describe a (n, l) BCH Code, where k can be determined with $k = \text{degree}(g(x))$.

In order to generate $g(x)$, we first have to find a modular polynomial $M(x)$. $M(x)$ has to be irreducible (or primitive), whereas the code parameter n is influenced by this property. Whenever $M(x)$ is primitive, we find n with $n = 2^{k_1} - 1$, where $k_1 = \text{degree}(M(x))$. A code constructed based on a primitive polynomial $M(x)$ is called *primitive BCH Code*. If $M(x)$ is only irreducible, we find n with $n|(2^{k_1} - 1)$. In this case, $M(x)$ can be used to construct a *non-primitive BCH Code*.

In general, it is possible to construct $g(x)$ for arbitrary values of f_k. Therefore, the polynomial $M(x)$ is used to build a finite extension field $GF(2^{k_1})$, related to $GF(2)$ (Galois Field (GF)). Moreover, for each element r^i of $GF(2^{k_1})$ there exists an irreducible polynomial $m_i(x)$, $(i = 0, 1, \ldots, (2^{k_1} - 2))$, with $m_1(x) = M(x)$.

The element r^i is the root of the polynomial $m_i(x)$. Moreover, we find, according to the fundamental theorem of algebra, t roots $r^i, r^{2i}, r^{2^2 i}, \ldots, r^{2^{t-1} i}$ of $m_i(x)$ with $t \leq k_1$ and thus,

$$m_i(x) = (x - r^i)(x - r^{2i})(x - r^{2^2 i}) \ldots (x - r^{2^{t-1} i}). \tag{3.13}$$

Since d_{min} is realized by means of a consecutive sequence of roots in $GF(2^{k_1})$, the generator polynomial $g(x)$ is a product of irreducible polynomials $m_i(x)$. The number of irreducible polynomials that have to be multiplied depends on the desired properties of the code:

$$g(x) = \; LCM \; \{m_\mu(x), m_{\mu+1}(x), \ldots, m_{\mu+d_{min}-2}(x)\}, \tag{3.14}$$

where μ is set either to 0 or to 1 in practice. Note that according to Equation (3.13), $m_i = m_{2i} = m_{2^2 i} = \ldots = m_{2^{t-1} i}$. For more details about mathematical coherences used for constructing $g(x)$, see [70, 92].

Beside the description based on $g(x)$, it is also possible to describe cyclic codes by means of the *parity-check matrix* $\mathbf{H}_{k \times n}$. Again, we find $\mathbf{H}_{k \times n} \mathbf{c}^T = \mathbf{0} \; \forall \mathbf{c} \in \mathcal{C}$.

The parity-check matrix $\mathbf{H}_{k \times n}$ is generated by means of $g(x)$, whereas there are several ways to calculate $\mathbf{H}_{k \times n}$. One approach is to calculate $\mathbf{H}_{k \times n}$ with x^i mod

[9] These codes $\mathcal{C}$ can be seen as a vector subspace of $\mathbb{F}_2^n$ closed with respect to addition and multiplication by an element of the finite field.

$g(x)$, $(i = 0, 1, \ldots, (n-1))$, as used in [85]. Another possibility is to determine $\mathbf{H}_{k\times n}$ by means of:

$$g(x)h(x) = f(x), \tag{3.15}$$

where $h(x)$ is the check polynomial and $f(x)$ the main polynomial, with $f(x) = x^n + 1$. The check polynomial $h(x)$ can be used to derive $\mathbf{H}_{k\times n}$, by simply shifting the coefficients of $h(x)$ k times and writing the results as a matrix.

Example: $(7, 4, f_k = 1)$ Cyclic HC

$M(x) = x^3 + x + 1$, $\mu = 1$

Since $M(x)$ is primitive, we calculate n with $n = 2^3 - 1 = 7$.

With $d_{min} = 3$, we find $g(x) =$ LCM $\{m_1(x), m_2(x)\} = m_1(x) = x^3 + x + 1$,

and determine $h(x) = \frac{f(x)}{g(x)} = \frac{x^7+1}{x^3+x+1} = x^4 + x^2 + x + 1$.

Cyclical shifting of the coefficients of $h(x)$ with $h_{1\times n} = (0010111)$ leads to

$$\mathbf{H}_{3\times 7} = \begin{pmatrix} 0\,0\,1\,0\,1\,1\,1 \\ 0\,1\,0\,1\,1\,1\,0 \\ 1\,0\,1\,1\,1\,0\,0 \end{pmatrix}.$$

■

3.3. Partitioning the Vector Space of Linear Codes in Cosets

Partitioning the vector space in cosets can be used to reduce the complexity of embedding. Therefore, we introduce the concept of cosets within this Section.

For any linear (n, l) code $\mathcal{C}$ with generator matrix $\mathbf{G}_{l\times n}$ and parity-check matrix $\mathbf{H}_{k\times n}$ we can define: Multiplying any vector $\mathbf{z} \in \mathbb{F}_2^n$ with the parity-check matrix $\mathbf{H}_{k\times n}$ leads to a vector $\mathbf{s} \in \mathbb{F}_2^k$, called the *syndrome* related to $\mathbf{z}$. Generally, each vector of $\mathbb{F}_2^n$ can be assigned to a set $\mathcal{C}(\mathbf{s}_i) \subset \mathbb{F}_2^n$, called *coset* of $\mathbf{s}_i$ with $i = 1, 2, \ldots, 2^k$. The coset $\mathcal{C}(\mathbf{s}_i)$ of the syndrome $\mathbf{s}_i$ denotes all vectors $\mathbf{z}$ leading to the syndrome $\mathbf{s}_i$ when multiplied with the parity-check matrix $\mathbf{H}_{k\times n}$:

$$\mathcal{C}(\mathbf{s}_i) = \{\mathbf{z} \mid \mathbf{s}_i = \mathbf{H}_{k\times n}\,\mathbf{z}^T, \mathbf{s}_i \in \mathbb{F}_2^k, \mathbf{z} \in \mathbb{F}_2^n\}. \tag{3.16}$$

Consequently, the code $\mathcal{C}$ can be seen as the coset of the syndrome $\mathbf{s}_i = \mathbf{0}$:

$$\mathcal{C}(\mathbf{0}) = \left\{\mathbf{c} \mid \mathbf{H}_{k\times n}\,\mathbf{c}^T = \mathbf{0}\right\} = \mathcal{C}. \tag{3.17}$$

The partitioning of the vector space in cosets is solely determined by the code. Generally, there are different parity-check matrices related to a code $\mathcal{C}$. Note that the partitioning of the code into cosets is not influenced by the selection of $\mathbf{H}_{k\times n}$, just the assignment of the cosets to the syndromes. Hence, the following properties can be formulated [58]:

1. Cosets are disjoint

$$\mathcal{C}(\mathbf{s}_i) \cap \mathcal{C}(\mathbf{s}_j) = \emptyset \quad \forall \mathbf{s}_i, \mathbf{s}_j \in \mathbb{F}_2^k,\ \mathbf{s}_i \neq \mathbf{s}_j.$$

2. The union of all cosets form the vector space $\mathbb{F}_2^n$

$$\bigcup_{\mathbf{s}_i \in \mathbb{F}_2^k} \mathcal{C}(\mathbf{s}_i) = \mathbb{F}_2^n.$$

Another important property concerns the summation of elements. Adding two elements $\mathbf{z}_i \in \mathcal{C}(\mathbf{s}_i)$ and $\mathbf{z}_j \in \mathcal{C}(\mathbf{s}_j)$ leads to an element of coset $\mathcal{C}(\mathbf{s}_i \oplus \mathbf{s}_j)$ since

$$\mathbf{H}_{k\times n}(\mathbf{z}_i \oplus \mathbf{z}_j)^T = \mathbf{H}_{k\times n}\mathbf{z}_i^T \oplus \mathbf{H}_{k\times n}\mathbf{z}_j^T = \mathbf{s}_i \oplus \mathbf{s}_j. \tag{3.18}$$

A special case of this occurs when adding an element $\mathbf{z}_j \in \mathcal{C}$. In this case, the resulting element is member of the same coset as $\mathbf{z}_i$. Thus, we find:

3. Each coset can be expressed as

$$\mathcal{C}(\mathbf{s}_i) = \mathbf{z}_i \oplus \mathcal{C}, \quad \mathbf{z}_i \in \mathcal{C}(\mathbf{s}_i).$$

Generally, there are 2^n possible sequences of length n. Furthermore, the message **emb** that should be embedded and, thereby, the related syndrome is of length k. Thus, there are 2^k different syndromes $\mathbf{s}$.

Since cosets associated with different syndromes are disjoint, we find for linear codes that there are $2^{n-l} = 2^k$ disjoint sets. Thus, the number of elements of each coset can be determined with:

$$\frac{2^n}{2^k} = \frac{2^{l+k}}{2^k} = \frac{2^l 2^k}{2^k} = 2^l, \tag{3.19}$$

i. e., each the vector space can be partitioned into 2^k cosets, each consisting of 2^l sequences. As coset leader $\mathbf{c}_L$ we denote the member of the coset $\mathcal{C}()$ with the smallest Hamming weight w.

In order to minimize the distortion introduced during embedding, it is advantageous to have as many sequences as possible within each coset. Thus, it is possible to find for each cover sequence $\mathbf{a}$ an appropriate sequence $\mathbf{f}$ that maps

a to a stego sequence **b**, where $d_\rho(\mathbf{a}, \mathbf{b})$ is minimal. Additionally, it must be possible to derive all 2^k possible syndromes, i. e., to embed all possible messages **emb**.

In error correction coding scenarios, this is not important at all[10]. However, in steganography based on syndrome coding this is an essential requirement since it has to be possible to embed all 2^k feasible syndromes **emb**. Therefore, the 2^n sequences of length n should be distinguishable into 2^k cosets, one for each syndrome, containing 2^l sequences each.

The question arises as to whether there are any codes that do not fulfill this requirement. First of all, codes built according to deterministic rules always fulfill this requirement and are therefore suited for steganographic systems.

However, there exits at least one code whose parity-check matrix is built according to stochastic rules that do not fulfill the partitioning requirement [86]: Gallager [56] describes a regular $(20, 5, d_{min} = 6)$ Low-Density-Parity-Check (LDPC) Code. Each row of the parity-check matrix $\mathbf{H}_{15\times 20}$ has a weight of 6, and each column a weight of 3. Based on this parity-check matrix it is not possible to achieve all feasible syndromes and thereby some combinations of **emb** could not be embedded.

Only $2^{k-2} = 2^{13}$ syndromes, out of $2^k = 2^{15}$ possible syndromes are covered by this code. Consequently, there are $\frac{2^n}{2^{k-2}} = 2^{l+2} = 2^7$ sequences, that lead to one and the same syndrome. Writing all possible syndromes as integer values, Table 3.1 gives the number of sequences leading to each syndrome:

int(**s**)	0	1	2	3	4	5	6	7	8	9	10	11	12	...
$\lvert\mathcal{C}(\mathbf{s})\rvert$	128	0	0	0	0	128	0	0	0	128	0	0	128	...

Table 3.1.: Number of Coset Members for Each Syndrome for the Regular $(20, 5, d_{min} = 6)$ LDPC Code in Extracts According to [86].

Because of the "missing" syndromes, this code is not suited for embedding at all. The question arises as to whether there are other codes that might not be used in steganography.

[10]For ECC, only error patterns **f** with weight $\mathrm{w}(\mathbf{f}) \leq f_k$ are analyzed. In most cases, this number is less than 2^k.

3.4. Performance of a Code Suited for Steganography

Generally, there are two main problems that can be solved using coding theory: *packing* and *covering*. While packing is fundamental in ECC, covering codes are used for, e. g., data compression [14].

The differences between the packing and the covering problem are clarified by [14]:

- Packing: Given n and R, what is the maximum number of non-intersecting spheres of radius R, that can be placed in the n-dimensional space?
- Covering: Given n and R, what is the smallest number of Hamming spheres of radius R that can be placed in such a way that every vector in the space is contained in at least one of them? Thus, we are looking for the smallest R where the spheres cover the whole space.

The parameters for describing the performance of a code suited for channel coding techniques, d_{min} or f_k are not suited for describing the performance of a code suited for steganography. While for ECC, the number of correctable errors $f_k = \lfloor \frac{d_{min}-1}{2} \rfloor$ is of primary interest, for the design of a steganographic scheme, the covering property R of a code is most important. Instead of a large distance between the codewords, good covering properties are required. Therefore, it is preferable that the codewords should cover the space $\mathbb{F}_2^n$ in a way that any sequence $\mathbf{z} \in \mathbb{F}_2^n$ is close to a codeword $\mathbf{c} \in \mathcal{C}$.

3.4.1. Covering Radius R

According to [14, Definition 2.1.3], the covering radius of a code $\mathcal{C} \subseteq \mathbb{F}_2^n$ is the smallest integer R such that every vector $\mathbf{z} \in \mathbb{F}_2^n$ is R-covered by at least one codeword of $\mathcal{C}$, i. e.,

$$\begin{aligned} R &= \max_{\mathbf{z} \in \mathbb{F}_2^n} d_H(\mathbf{z}, \mathcal{C}) \\ &= \max_{\mathbf{z} \in \mathbb{F}_2^n} \min_{\mathbf{c} \in \mathcal{C}} d_H(\mathbf{z}, \mathbf{c}). \end{aligned} \tag{3.20}$$

Thus, the covering radius is a measure of the distance between the code and the farthest-off vectors in space, i. e., the Hamming weight of any coset member is at most R.

The covering radius also gives the smallest integer R such that the union of the Hamming spheres of radius R centered at the codewords is the whole space. For a linear $(n, n-k)$ code, the covering radius is $R \leq \frac{n}{2}$.

3.4.2. Average Number of Embedding Changes R_a

Another important parameter describing the performance of a code suited for steganography is the average distance to code R_a, which expresses the expected value of $d_H(\mathbf{x}, \mathcal{C})$ over randomly uniform distributed $\mathbf{x} \in \mathbb{F}_2^n$, i. e.,

$$R_a = \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} d_H(\mathbf{x}, \mathcal{C}). \tag{3.21}$$

In [51] it is shown that the expected number of embedding changes is also equal to the average weight of the coset leaders with:

$$\begin{aligned} R_a &= \frac{1}{2^n} \sum_{\mathbf{x} \in \mathbb{F}_2^n} d_H(\mathbf{x}, \mathcal{C}) \\ &= \frac{1}{2^n} \sum_{\mathbf{s} \in \mathbb{F}_2^k} \sum_{\mathbf{x} \in \mathcal{C}(\mathbf{s})} d_H(\mathbf{x}, \mathcal{C}(\mathbf{s})) \\ &= \frac{1}{2^n} \sum_{i=1}^{2^k} 2^l \mathrm{w}(\mathbf{c}_L(\mathbf{s})) \\ &= \frac{1}{2^k} \sum_{i=1}^{2^k} \mathrm{w}(\mathbf{c}_L(\mathbf{s})). \end{aligned} \tag{3.22}$$

Note that given a coset $\mathcal{C}(\mathbf{s})$, for any $\mathbf{x} \in \mathcal{C}(\mathbf{s})$, the distance $d_H(\mathbf{x}, \mathcal{C})$ is equal to the weight of the coset leader $\mathrm{w}(\mathbf{c}_L(\mathbf{s}))$ [52].

Therefore, the expected number of embedding changes R_a for any syndrome-coding based embedding scheme is equal to the average weight of all coset leaders for randomly chosen messages and uniformly distributed covers. Note that the messages are drawn uniformly at random from $\mathbb{F}_2^k$, since they will be typically encrypted before embedding. If all cover sequences $\mathbf{a}$ and all message bit strings $\mathbf{emb}$ occur with equal probability, all possible syndromes also occur with equal probability.

Generally, R and R_a can be seen as important properties of a code concerning syndrome coding. While R_a gives the average number of embedding changes, the covering radius R is related to the worst case and can be seen as the maximum weight of all coset members [14, Theorem 2.1.11]. This means that R changes are necessary to embed in the worst case, i. e., for the most disadvantageous combination of $\mathbf{a}$ and $\mathbf{emb}$. With increasing length of the code and fixed α, we also find that $R_a \to R$ [47].

It is important to mention that the embedding distortion, i. e., the overall embedding impact (Equation 2.5) is low, if the average number of embedding changes R_a is low[11].

[11] When assuming a general profile, the algorithms discussed in this thesis exclude so-called wet

3.4.3. Embedding Efficiency e

The embedding efficiency e gives the number of message bits embedded depending on the distortion introduced during embedding. It can be determined with:

$$e = \frac{|\mathbf{m}|}{d_\rho(\mathbf{A}, \mathbf{B})} = \frac{k}{d_\rho(\mathbf{a}, \mathbf{b})}. \tag{3.23}$$

Note that the complexity of computing e depends on the steganographic scheme, on its inner mechanism and on the cover source. For simple embedding schemes, such as LSB or ± 1 embedding, the embedding efficiency can easily be derived analytically. However, whenever the distortion is rather complex to determine, the embedding efficiency e has to be determined experimentally. Note that e depends on the embedding impact d_ρ.

The embedding efficiency is often referred to as a measure of the quality of the embedding. A maximized embedding efficiency is related to a minimized embedding distortion, which seems to be preferable when considering an improved security of the steganographic scheme.

In fact, schemes with higher embedding efficiency, i. e., a higher number of message bits embedded per embedding change, are less likely to be successfully attacked than schemes with lower embedding efficiency [45].

3.4.3.1. Boundary on Embedding Efficiency

Assuming a uniform profile of embedding impact, or a general profile based on Wet Paper Steganography, the embedding distortion is measured by the number of embedding changes. Generally, the embedding distortion is low, if the average number of embedding changes R_a is low.

Since the impact of embedding is captured by a distortion metric taking into account the number of embedding changes, i. e., in terms of the Hamming distance between the corresponding cover and stego bit vectors, the embedding efficiency of a specific code can be calculated with:

$$e = \frac{k}{R_a}. \tag{3.24}$$

Embedding is therefore efficient if R_a is low. To derive an upper boundary on the embedding efficiency, Fridrich et al. state [47] that every syndrome can be generated by adding at most R columns of $\mathbf{H}_{k\times n}$. Thus, there exist

$$\sum_{i=0}^{R} \binom{n}{i} (q-1)^i \tag{3.25}$$

elements. Thus, even for this profile, we assume the average number of embedding changes R_a as a measure related to the embedding impact.

ways in which one can make at most R changes in n elements [51]. While this earlier publication considers only the binary case, i. e., for $q = 2$, a later version of this derivation is more general for arbitrary values of q (emphasized in blue [49]).

Consequently, the maximum number of bits that can be embedded in n bits is:

$$k = \mathrm{ld}\,|\mathcal{M}| \leq \mathrm{ld} \sum_{i=0}^{R} \binom{n}{i} (q-1)^i = \mathrm{ld}\, V_q(n, R), \tag{3.26}$$

where $V_q(n, R)$ is the volume of a ball of radius R in $\mathbb{F}_q^n$. For the binary case, we find:

$$V(n, R) = \sum_{i=0}^{R} \binom{n}{i}. \tag{3.27}$$

According to the sphere-covering boundary, a frequently used boundary in coding theory that can be found, e. g., in the book of Cohen et al. [14, Lemma 2.4.2], we find that

$$\mathrm{ld}\, V_q(n, R) \leq n H_q \left(\frac{R}{n}\right), \tag{3.28}$$

and thus, $k \leq n H_q(\frac{R}{n})$, where H_q is the entropy function defined as:

$$H_q = -x\mathrm{ld}x - (1-x)\mathrm{ld}(1-x) + x\mathrm{ld}(q-1). \tag{3.29}$$

By transposing the previous equation, we obtain:

$$H_q^{-1}(\alpha) \leq \frac{R}{n}, \tag{3.30}$$

where $\alpha = \frac{k}{n}$ is defined as the relative message length and $H^{-1}()$ is the inverse of the entropy function $H()$[12].

Based on this coherence, Fridrich et al. derive an upper boundary on the lower embedding efficiency $\underline{e}$ [51, 49], where $\underline{e}$ can be seen as the number of embedded bits per embedding change in the worst case:

$$\underline{e} = \frac{k}{R} = \alpha \frac{n}{R} \leq \frac{\alpha}{H_q^{-1}(\alpha)}. \tag{3.31}$$

Fridrich et al. also state that this is an asymptotic boundary on the embedding efficiency e:

$$e \lesssim \frac{\alpha}{H_q^{-1}(\alpha)}, \tag{3.32}$$

[12] Note that it is not possible to analytically determine the inverse to $H()$. The use of a Look-up Table is required.

which holds for almost all (n, l) codes, since the relative covering radius $\phi = \frac{R}{n}$ and the relative distance to code $\phi_a = \frac{R_a}{n}$ converge with $n \to \infty$. More precisely, Fridrich et al. state and prove [51, Theorem 2]: For any $0 < \alpha < 1$ and any $\epsilon > 0$, the fraction of all binary (n, l) codes for which $|\phi - \phi_a| \leq \epsilon$ tends to 1 as $n \to \infty$.

To summarize: the highest lower embedding efficiency $\underline{e}$ can be achieved using a (n, l) code with a small covering radius. Knowing the performance boundary, the problem faced by a steganographer is to find codes that could reach this theoretical boundary in practice with low computational complexity.

According to Fridrich et al., this boundary is tight and asymptotically achievable using linear codes [51]. Note that the relative codimension $(n-l)/n$ of almost all random (n, l) codes asymptotically achieves $H(R/n)$ for a fixed change rate $R/n < 1/2$ and $n \to \infty$ [14, Theorem 12.3.5]. Therefore, there exist embedding schemes based on linear codes whose embedding efficiency is asymptotically optimal.

3.5. Embedding with Syndrome Coding by Minimizing the Embedding Impact

The general concept of syndrome coding for steganography was introduced in Section 3.1. Within this section, we expand the basic principle in order to include approaches that minimize the embedding impact introduced during embedding and thus, increase the embedding efficiency. We describe the basic principles for finding the stego sequence $\mathbf{b}$ in order to make clear how the algorithms described in the following chapters work.

Recall the basic principle of syndrome coding, where the syndrome derived by multiplying the parity-check matrix $\mathbf{H}_{k\times n}$ and the stego sequence $\mathbf{b}$ should be equal to the confidential message part $\mathbf{emb}$ (see Equation (3.2)):

$$\mathbf{s} = \mathbf{H}_{k\times n}(\mathbf{a} \oplus \mathbf{f})^T = \mathbf{H}_{k\times n}\mathbf{b}^T = \mathbf{emb}. \tag{3.33}$$

In order to achieve this goal, the cover sequence $\mathbf{a}$ has to be modified. Whenever the overall goal is to minimize the embedding impact, a sequence $\mathbf{f}$, fulfilling Equation (3.33) has to be determined, that minimizes the introduced distortion:

$$\mathbf{f} = \underset{\mathbf{f}\in\mathbb{F}_2^n}{\operatorname{argmin}}\, d_\rho(\mathbf{a}, \mathbf{a} \oplus \mathbf{f}). \tag{3.34}$$

This approach however, requires an exhaustive search within the whole vector space. Based on the partitioning of the vector space in cosets as described in Section 3.3, it is possible to reduce the search space by searching only within the cosets.

Each coset $\mathcal{C}(\mathbf{s})$ is stored in a Look-up Table. For this approach, there are two different ways for finding an appropriate stego sequence.

First, the search for an appropriate sequence $\mathbf{b}$ can be carried out within $\mathcal{C}(\mathbf{emb})$. Note that all elements of coset $\mathcal{C}(\mathbf{emb})$ can be considered for embedding. However, to minimize the embedding impact, we look for sequence $\mathbf{b}$ according to:

$$\mathbf{b} = \underset{\mathbf{b}\in\mathcal{C}(\mathbf{emb})}{\operatorname{argmin}} d_\rho(\mathbf{a}, \mathbf{b}). \tag{3.35}$$

The second possibility for finding an appropriate stego sequence is to stick to the flipping pattern $\mathbf{f}$: Note that the cover sequence $\mathbf{a}$ is related to a syndrome $\mathbf{s} \in \mathcal{C}(\mathbf{H}_{k\times n}\mathbf{a}^T)$. The resulting stego sequence $\mathbf{b}$ after embedding, however, should lead to a syndrome $\mathbf{s} \in \mathcal{C}(\mathbf{emb})$. Thus, the sequence $\mathbf{f}$, which will be added to $\mathbf{a}$ in order to achieve $\mathbf{b}$ has to be a sequence of coset $\mathcal{C}(\mathbf{H}_{k\times n}\mathbf{a}^T \oplus \mathbf{emb})$ (according to Equation (3.18) in Section 3.3). In order to minimize the embedding impact, the sequence $\mathbf{f}$ out of all 2^l sequences in coset $\mathcal{C}(\mathbf{H}_{k\times n}\mathbf{a}^T \oplus \mathbf{emb})$ has to be chosen according to:

$$\mathbf{f} = \underset{\mathbf{f}\in\mathcal{C}(\mathbf{H}_{k\times n}\mathbf{a}^T\oplus\mathbf{emb})}{\operatorname{argmin}} d_\rho(\mathbf{a}, \mathbf{a}\oplus\mathbf{f}). \tag{3.36}$$

In this case, $\mathbf{f} = \mathbf{c}_L(\mathbf{H}_{k\times n}\mathbf{a}^T \oplus \mathbf{emb})$ is called a coset leader of coset $\mathcal{C}(\mathbf{H}_{k\times n}\mathbf{a}^T \oplus \mathbf{emb})$.

A coset leader of coset $\mathcal{C}(\mathbf{s})$ is defined as the optimal flipping pattern $\mathbf{f}$ to map the cover sequence $\mathbf{a}$ to a stego sequence $\mathbf{b}$. Consequently, a coset leader can be seen as the sequence $\mathbf{f}$ that minimizes the overall embedding impact $d_\rho(\mathbf{a}, \mathbf{a}\oplus\mathbf{f})$. Generally, it is possible to find different sequences $\mathbf{f} \in \mathcal{C}(\mathbf{s})$ as coset leaders, i. e., more than one sequence that minimizes the embedding impact.

For a distortion measure considering the number of changes, i. e., a uniform profile, the goal is to minimize the difference between $\mathbf{a}$ and $\mathbf{b}$. Thus, a coset leader $\mathbf{c}_L(\mathbf{s})$, added to $\mathbf{a}$ in order to obtain $\mathbf{b}$, is simply the sequence with minimum Hamming weight. Note that $\mathrm{w}(\mathbf{c}_L(\mathbf{s})) \leq R$.

Considering the severity of changes, i. e., a general profile instead, requires to find a sequence that minimizes $d_\rho(\mathbf{a}, \mathbf{a}\oplus\mathbf{f})$ based on the underlying profile of embedding impact $\boldsymbol{\rho}[1], ..., \boldsymbol{\rho}[n]$. A coset leader is therefore defined as a sequence that minimizes the embedding impact. When assuming a general profile, the algorithms discussed in this thesis are based on Wet Paper Steganography. Therefore, they exclude so-called wet elements from the embedding process. The dry elements are used for embedding, i. e., we assume $\boldsymbol{\rho}[\mathrm{Dry}] = const.$ Thus, even for this profile, we find the sequence with minimum Hamming weight as coset leader.

Based on Equation (3.36), Galand and Kabatiansky [54] specified a steganographic scheme: An (n, l) code $\mathcal{C}$ with covering radius R can be used to construct an embedding scheme which is able to communicate $n - l$ bits using at most R changes:

$$\begin{aligned}
\mathrm{Emb}(\mathbf{a}, \mathbf{emb}) &= \mathbf{a} \oplus \mathbf{c}_L(\mathbf{H}_{k\times n}\mathbf{a}^T \oplus \mathbf{emb}) = \mathbf{b}, \\
\mathrm{Extr}(\mathbf{b}) &= \mathbf{H}_{k\times n}\mathbf{b}^T, \quad \text{with} \\
\mathrm{Extr}(\mathrm{Emb}(\mathbf{a}, \mathbf{emb})) &= \mathbf{H}_{k\times n}\ \mathbf{b}^T \\
&= \mathbf{H}_{k\times n}\ \mathbf{a}^T \oplus \mathbf{H}_{k\times n}\ \mathbf{c}_L(\mathbf{H}_{k\times n}\ \mathbf{a}^T \oplus \mathbf{emb})^T \\
&= \mathbf{H}_{k\times n}\ \mathbf{a}^T \oplus \mathbf{H}_{k\times n}\ \mathbf{a}^T \oplus \mathbf{emb} = \mathbf{emb}.
\end{aligned} \tag{3.37}$$

Embedding by minimizing the embedding impact as described so far, includes an exhaustive search within cosets. In order to propose approaches to speed up the search for an appropriate flipping pattern $\mathbf{f}$ and thus, for an appropriate stego sequence $\mathbf{b}$, Fridrich et al. states that embedding can be seen as a binary quantization problem [37]. This is true since the search for an appropriate sequence $\mathbf{f}$ can also be reformulated as the search of the closest codeword. Note that the search for the closest codeword, i. e., binary quantization, is a well known problem for which several solutions have been proposed.

The embedding process can be reformulated as follows. Given an arbitrary coset member $\mathbf{f_m} \in \mathcal{C}(\mathbf{emb})$, it is possible to define the coset $\mathcal{C}(\mathbf{emb})$ based on the code $\mathcal{C}$:

$$\begin{aligned}
\mathcal{C}(\mathbf{emb}) &= \{\mathbf{f_m} \oplus \mathbf{c},\ \mathbf{c} \in \mathcal{C}\} \\
&= \mathbf{f_m} \oplus \mathcal{C}(\mathbf{0}).
\end{aligned} \tag{3.38}$$

The coset indeed includes all 2^l sequences leading to syndrome $\mathbf{emb}$, since adding 2^l different codewords to $\mathbf{f_m}$ will lead to 2^l different sequences associated with syndrome $\mathbf{s} = \mathbf{emb}$:

$$\mathbf{H}_{k\times n}\ (\mathbf{f_m} \oplus \mathbf{c})^T = \mathbf{H}_{k\times n}\ \mathbf{f_m}^T \oplus \mathbf{H}_{k\times n}\ \mathbf{c}^T = \mathbf{emb} \oplus \mathbf{0} = \mathbf{emb}. \tag{3.39}$$

This coherence can also be used to determine the optimal sequence for embedding. In order to minimize the embedding impact, a member $\mathbf{b}$ of coset $\mathcal{C}(\mathbf{emb})$ has to be chosen, that is closest to $\mathbf{a}$ in metric d_ρ:

$$\mathbf{b} = \underset{\mathbf{b}\in\mathcal{C}(\mathbf{emb})}{\operatorname{argmin}}\ d_\rho(\mathbf{a}, \mathbf{b}) \tag{3.40}$$

According to Equation (3.38), we can express $\mathbf{b}$ as $\mathbf{f_m} \oplus \mathbf{c}$ with $\mathbf{c} \in \mathcal{C}$.

Thus, we have to determine the closest codeword [19], i. e., the codeword $\mathbf{c_{emb}}$ that minimizes the distortion between $\mathbf{a} \oplus \mathbf{f_m}$ and $\mathcal{C}$:

$$\begin{aligned}
\mathbf{c_{emb}} &= \underset{\mathbf{c}\in\mathcal{C}}{\operatorname{argmin}}\, d_\rho(\mathbf{a}, \mathbf{f_m} \oplus \mathbf{c}) \\
&= \underset{\mathbf{c}\in\mathcal{C}}{\operatorname{argmin}}\, d_\rho(\mathbf{a} \oplus \mathbf{f_m}, \mathbf{c}).
\end{aligned} \tag{3.41}$$

The stego sequence $\mathbf{b}$ can now be determined with $\mathbf{b} = \mathbf{f_m} \oplus \mathbf{c_{emb}}$, where $\mathbf{f_m}$ is a coset member of $\mathcal{C}(\mathbf{emb})$ and $\mathbf{c_{emb}} \in \mathcal{C}$. This sequence is indeed a valid steganogram for $\mathbf{emb}$ since we find:

$$\mathbf{H}_{k\times n}(\mathbf{f_m} \oplus \mathbf{c_{emb}})^T = \mathbf{H}_{k\times n}\,\mathbf{f_m}^T \oplus \mathbf{H}_{k\times n}\,\mathbf{c_{emb}}^T = \mathbf{emb} \oplus \mathbf{0} = \mathbf{emb}. \tag{3.42}$$

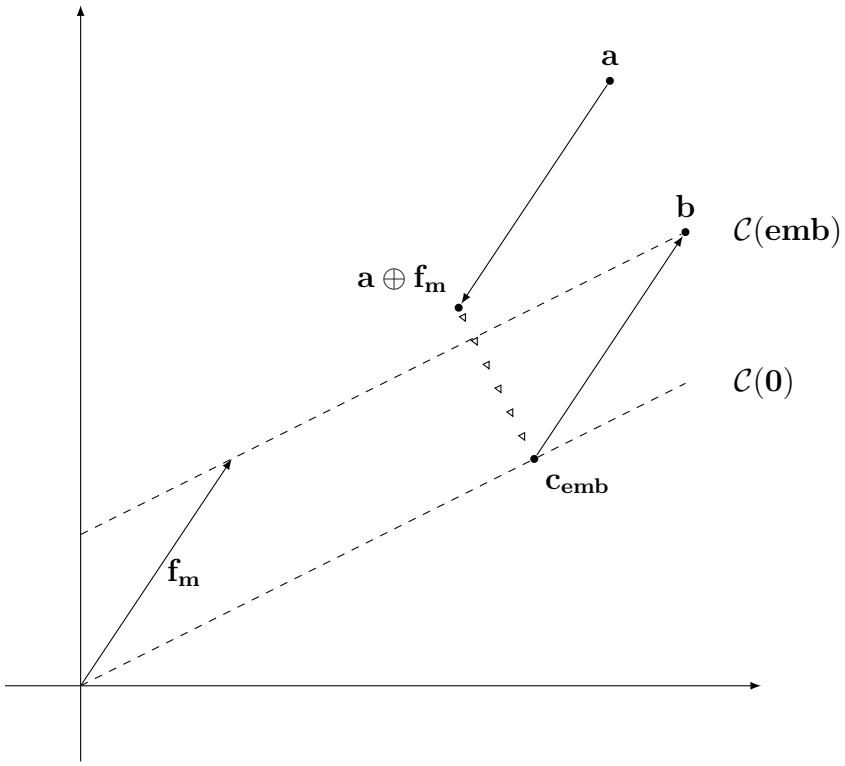

Figure 3.2.: Schematic Interpretation of the Embedding Process According to [19].

A schematic interpretation of the embedding process according to [19] can be found in Figure 3.2. The sender needs to find $\mathbf{c_{emb}}$ that is closest to $\mathbf{a} \oplus \mathbf{f_m}$. The embedding process can be interpreted as follows:

1. Shift the cover sequence $\mathbf{a}$ using an arbitrary coset member $\mathbf{f_m}$,
2. Find the closest codeword to $\mathbf{a} \oplus \mathbf{f_m}$ denoted as $\mathbf{c_{emb}}$,[13] and
3. Shift $\mathbf{c_{emb}}$ back into the coset $\mathcal{C}(\mathbf{emb})$.

[13] Note that in the binary case, i. e., in $GF(2)$, a subtraction is equivalent to an addition modulo 2 ($\oplus$).

The resulting stego sequence $\mathbf{b} \in \mathcal{C}(\mathbf{emb})$ is closest to the cover sequence $\mathbf{a}$. While the coset member can be easily determined by searching for a sequence $\mathbf{f_m} \in \mathcal{C}(\mathbf{emb})$, the search for the closest codeword is equivalent to finding the coset leader, i. e., an NP-complete problem [4].

3.6. Roadmap of This Thesis

Several approaches for Minimum-Embedding-Impact Steganography have been proposed in the past, like embedding based on Wet Paper Codes by Fridrich et al. [44] as well as embedding based on BCH Codes by Schönfeld and Winkler [86]. Note that these approaches share the basic idea of syndrome coding. Therefore, embedding should be done in a way which enables the receiver to simply calculate $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$ in order to read the confidential message. The sender however, has to find a solution to this system of linear equations in such a way that $d_\rho(\mathbf{a}, \mathbf{b})$ is minimized.

The first contribution of this work is a consistent description of algorithms based on binary syndrome coding. Note that two of the algorithms presented within this thesis, embedding based on BCH Codes [85, 86, 87] and based on LDGM Codes with Belief Propagation [58, 60, 59], are developed at the chair of privacy and data security at the Technische Universität Dresden in collaboration of Dagmar Schönfeld and the author of this thesis.

In order to make the algorithms comparable to each other, we propose a classification and distinguish between algorithms based on a deterministic parity-check matrix and algorithms based on stochastic matrices. Furthermore, we distinguish between concepts for small blocks and therefore a small codeword length and concepts for big blocks and therefore a large codeword length. While for a small codeword length it is possible to determine the optimal solution, i. e., the solution that minimizes d_ρ, by exhaustive search, a large codeword length often requires iterative search strategies such as Belief Propagation (BP)/Survey Propagation (SP).

We describe different approaches for embedding, whereas each approach is based on the concept of syndrome coding, i. e., based on a parity-check matrix $\mathbf{H}_{k\times n}$:

- Concepts for Embedding Based on a Small Code Word Length
  - Parity-check matrices based on stochastic design rules
    - Block Minimal Method (also denoted as Meet-the-Middle) [45, 47, 48]
    - Matrix Embedding for Large Payload [52, 51]
  - Parity-check matrix built according to deterministic rules
    - Hamming Codes [16]

    - BCH Codes [85, 86, 87, 3, 101, 83]
    - Simplex Codes and Augmented Simplex Codes [52, 51]

- Concepts for Embedding Based on a Large Codeword Length
  - Wet Paper Codes
    - Wet Paper Codes [44, 42, 43, 46, 45, 47]
    - Wet Paper Codes combined with Gaussian Elimination [44, 42]
    - Wet Paper Codes combined with Lanczos and Wiedemann solver [44]
    - Wet Paper Codes combined with Structured Gaussian Elimination [42]
    - Wet Paper Codes for sparse matrices combined with the Matrix LT Process [43, 46, 45, 47]
  - LDGM Codes
    - LDGM Codes combined with Survey Propagation [19, 37, 21]
    - LDGM Codes combined with Belief Propagation [58, 60, 59]
  - Convolutional Codes
    - Syndrom Trellis Codes [24, 25]

Based on this consistent description, this thesis provides a comparison of the algorithms according to the properties of a steganographic scheme: security, capacity, success rate and complexity (see Section 2.1.3). Considering only the embedding step Θ, we assume the embedding efficiency e as a feasible parameter describing the security of the algorithms. Recall that the embedding efficiency gives the number of embedded bits dependent on the introduced distortion.

Beside minimizing the introduced distortion, and thereby maximizing the security, the algorithms under investigation aim at minimizing the embedding complexity while maximizing the success rate and the embedding capacity. However, as these are competing goals, only a tradeoff between all four properties is possible. Generally, we would like to point out advantages of applying principles from channel coding, such as deterministic matrices and decoding principles.

Part II.

Approaches for a Small Code Word Length

In this part, we will give an overview on algorithms for embedding based on syndrome coding for a small code word length. The advantage of such matrices $\mathbf{H}_{k\times n}$ with small code parameters is the possibility to determine the coset leader, i. e., the optimal solution to $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$ that minimizes the distortion $d_\rho(\mathbf{a}, \mathbf{b})$, directly by exhaustive search.

Within the following chapters, we describe the functionality of the algorithms (Chapter 4 and 5) and evaluate the algorithms according to the properties of a steganographic scheme (Chapter 6). Note that we describe only the embedding step Θ, i. e., the input of the algorithms are a cover bit string $\mathbf{A}$, the message $\mathbf{m}$ and a profile of embedding impact $\boldsymbol{\rho}[1], \ldots, \boldsymbol{\rho}[N]$.

One general guideline principle for the design of a steganographic scheme is the principle of minimizing the embedding impact (Section 2.3.1.4). Thus, the embedding process should be designed in a way to embed only in inconspicuous parts of the cover and to reduce the overall embedding impact of the changes introduced during embedding. Algorithms from coding theory help to achieve both objects.

Whenever a message shorter than the cover data itself is embedded, the impact of changes required for embedding a confidential message into the cover can be reduced. Crandall has first introduced a solution to this problem [16]. He proposed the concept of syndrome coding in steganography, often called Matrix Embedding in literature, based on structured Hamming Codes. Later, this concept was independently re-discovered by Willems [93], Schönfeld [85] and Galand [54]. Moreover, this approach has been used in practical systems [95, 96].

In more theoretical investigations, Bierbrauer [5, 6] and Galand [54, 55], working independently from each other, discovered that the concept of embedding efficiency e is closely related to the covering radius R of codes: a linear code can be used to construct an embedding scheme whose embedding capacity is the code redundancy k, while the covering radius corresponds to the maximal number of embedding changes necessary for embedding.

The first concept for excluding conspicuous parts of the cover, i. e., for transmitting a message using a channel where access to its symbols is constrained, was proposed by Anderson and is called the Selection Channel [2]. This embedding scheme can be seen as parity coding based on a checksum over n elements of the cover. Moreover, it can be seen as embedding with an informed sender, as it does not require the sender to share any knowledge of the constraints with the recipient.

In Anderson's scheme, the sender is able to choose one element out of a block of n elements (parity block) in order to make the necessary embedding change. Thereby, the sender has the possibility to modify the most inconspicuous element of each block. However, a disadvantage of this approach is clearly that most of the elements are not used for embedding. Thus, embedding capacity decreases with increasing block length. Consequently, in practical cases only a fraction of the embedding capacity is used.

Later, Fridrich et al. propose the Wet Paper Codes [44] in order to deal with excluded parts of the cover. This approach is applied on the whole cover and does not even sacrifice embedding capacity - however, at the cost of increased embedding complexity.

Recent research has combined both aforementioned objectives, i. e., excluding conspicuous parts of the cover while minimizing the impact of the introduced changes. Therefore, linear codes such as the BCH Code were investigated by Schönfeld and Winkler [85, 86, 87]. Moreover the Golay Code [93, 27, 28], Simplex Codes [52] and random codes of small dimension [51] are investigated as further approaches considering small blocks.

4. Embedding Based on Stochastic Parity-Check Matrices

Within this chapter, we introduce state-of-the-art algorithms for syndrome coding based on stochastic parity-check matrices. We give a description as well as some specifics of the algorithms.

The concept of Wet Paper Codes, i. e., simple variable rate stochastic codes based on a stochastic parity-check matrix was proposed by Fridrich et al. in the context of Wet Paper Steganography (Section 2.1.2.2).

Since the sender would like to modify only the changeable elements, the system of linear equations (Equation (3.33)) has to be adapted as follows. The parity-check $\mathbf{H}_{k\times n}$ is separated into $\mathbf{H}_{k\times n} = \left[\mathbf{H}_{k\times|\text{Wet}|}\ \mathbf{H}_{k\times|\text{Dry}|}\right]$ with $n = |\text{Wet}| + |\text{Dry}|$ depending on the positions of excluded elements[14]. Thereby, $\mathbf{H}_{k\times|\text{Wet}|}$ is a submatrix of $\mathbf{H}_{k\times n}$ corresponding to the wet elements Wet, and $\mathbf{H}_{k\times|\text{Dry}|}$ is a submatrix of $\mathbf{H}_{k\times n}$ corresponding to the dry elements Dry.

Based on the positions of wet elements, we find the flipping pattern $\mathbf{f}$ with $\mathbf{f}[i] = 0$ for wet elements, i. e., for positions where $\mathbf{a}[i]$ should remain unchanged. Furthermore, we reformulate Equation (3.33) according to $\mathbf{b} = \mathbf{a} \oplus \mathbf{f}$:

$$\mathbf{H}_{k\times n}\,\mathbf{f}^T = \mathbf{emb} \oplus \mathbf{H}_{k\times n}\,\mathbf{a}^T. \tag{4.1}$$

The concept of embedding, i. e., for finding a solution to Equation (4.1) can be summarized with:

1. Eliminate positions with $\mathbf{f}[i] = 0$ from $\mathbf{f} \Rightarrow \mathbf{f}_{|\text{Dry}|}$
2. Eliminate the corresponding columns from $\mathbf{H}_{k\times n} \Rightarrow \mathbf{H}_{k\times|\text{Dry}|}$
3. Reformulate Equation (4.1) to

$$\mathbf{H}_{k\times|\text{Dry}|}\,(\mathbf{f}_{|\text{Dry}|})^T = \mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T \tag{4.2}$$

4. Determine a solution for $\mathbf{f}_{|\text{Dry}|}$

[14] Within the approach of Wet Paper Steganography, the sender denotes elements as wet whose embedding impact $\boldsymbol{\rho}[i]$ is above a chosen threshold. These elements are excluded from the embedding process.

5. Determine the flipping pattern $\mathbf{f}$ related to $\mathbf{f}_{|\mathrm{Dry}|}$

6. Determine the stego sequence $\mathbf{b} = \mathbf{a} \oplus \mathbf{f}$

Note that the approach described in [44] considers the whole cover at once, i. e., $n = N$. Thus, it is not feasible to determine the coset leader by means of exhaustive search. To realize stochastic matrices with maximized embedding efficiency, Fridrich et al. propose block-based approaches, which will be presented in this chapter in more detail.

Within a first approach, called Block Minimal Method, exhaustive search is applied in order to determine the coset leader, i. e., the sequence $\mathbf{f}$ that minimizes the introduced distortion. This approach is applied to small blocks instead of the whole cover. A related approach, called Matrix Embedding for Large Payloads, additionally impose some structure on $\mathbf{H}_{k\times n}$ making exhaustive search possible whenever the code dimension l is small.

4.1. Block Minimal Method

The basic assumption of the Block Minimal Method, first described by Fridrich et al. in [45] and also denoted as Meet-in-the-Middle Algorithm [47] is that there will be more than one solution for Equation (3.33) whenever $|\mathbf{m}|$ is smaller than $|\mathrm{Dry}|$. In this case, the sender is able to select the stego sequence $\mathbf{b}$ with minimal distance $d_\rho(\mathbf{a}, \mathbf{b})$. Thus, the embedding impact will be minimized and by this the embedding efficiency maximized.

The algorithm itself is applied to small blocks, since finding the coset leader is an NP complete problem and can be solved for a small block length by exhaustive search. Thus, in this block-based scheme, sender and receiver divide the cover bit string $\mathbf{A}$ into $n_B = |\mathbf{m}|/k$ disjoint pseudo-random blocks. Note that there are k message bits embedded in each block containing $n = N/n_B$ cover elements.

Embedding is done based on stochastic codes with a parity-check matrix $\mathbf{H}_{k\times n}$, where the probability of 0 and 1 are equal to 1/2. Note that the sub-matrix $\mathbf{H}_{k\times|\mathrm{Dry}|}$ can be different for each block. Generally, the sender aims at generating the matrix $\mathbf{H}_{k\times n}$ in a way that its columns are non-zero and mutually different in order to avoid duplicates and zero columns. However, this structure does not directly affect the sub-matrix $\mathbf{H}_{k\times|\mathrm{Dry}|}$.

Finding a sequence that minimizes $d_\rho(\mathbf{a}, \mathbf{b})$ in Step 4 of the embedding algorithm is carried out by searching within cosets as proposed in Section 3.5. Note that for stochastic matrices the cosets are not as regular as for deterministic matrices, i. e., the 2^k cosets does not contain 2^l sequences each.

Fridrich et al. propose an algorithm for finding the coset leader in Step 4 as follows [48]: In preparation, the sender forms the subsets $\mathcal{U}_i$, where $\mathcal{U}_i$ can be seen as the set of syndromes that can be obtained by adding i columns of $\mathbf{H}_{k\times|\mathrm{Dry}|}$.

Fridrich et al. define $\mathcal{U}_1 \subset \mathbb{F}_2^k$ as the set of all columns of $\mathbf{H}_{k\times|\mathrm{Dry}|}$. Furthermore, they define $\mathcal{U}_{i+1} = \mathcal{U}_1 + \mathcal{U}_i \backslash (\mathcal{U}_1 \cup \ldots \cup \mathcal{U}_i) \backslash \{\mathbf{0}\}$ for $i = 1, \ldots, k$ and $\mathcal{U}_i = \emptyset$ for $i > R$. Note that the sum of two sets is defined as $\mathcal{U}_i + \mathcal{U}_j = \{u_i \oplus u_j \mid u_i \in \mathcal{U}_i,\ u_j \in \mathcal{U}_j\}$.

The sender tries to find the coset leader by generating $\mathcal{U}_1, \mathcal{U}_2, \ldots$ and stop once $(\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T) \in \mathcal{U}_r$. Note that the cardinality of these sets increases exponentially.

Based on the coherence given in Equation (4.3):

$$\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T = \mathbf{H}_{k\times|\mathrm{Dry}|}[., j_1] \oplus \ldots \oplus \mathbf{H}_{k\times|\mathrm{Dry}|}[., j_r], \tag{4.3}$$

the algorithm substituting Step 4 can be summarized as follows according to [47]:

4.1 If $((\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T) \in \mathcal{U}_1)$,

(The solution is one of the original columns

$\mathbf{f}[j_1] = 1$ //because $(\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T) = \mathbf{H}_{k\times|\mathrm{Dry}|}[., j_1]$ for some j_1

for all $j \neq j_1$ set $\mathbf{f}[j] = 0$),

Otherwise set $t = 1, r = 1$

4.2 While $((\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T) + \mathcal{U}_t) \cap \mathcal{U}_r = \emptyset$,

(If $t = r$:

$r = r + 1$

If ($\mathcal{U}_r$ not generated) construct $\mathcal{U}_r$

If $t \neq r$:

$t = t + 1$

If ($\mathcal{U}_t$ not generated) construct $\mathcal{U}_t$)

4.3 Any element of the intersection $((\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T) + \mathcal{U}_t) \cap \mathcal{U}_r$ is a coset leader of weight $t + r$.

Note that Fridrich et al. propose k to be publicly known. Moreover, the receiver knows N as well. Thus, only the message length, i. e., $|\mathbf{m}|$ has to be communicated to the receiver.

In a practical scheme, it is also necessary to communicate to the receiver, which blocks failed to hold all k bits. An example of the algorithm is given below:

Example:

Given is a stochastic code with $n = 5$. The sender would like to embed the message $\mathbf{emb} = (110)^T$ into the cover sequence $\mathbf{a} = (10111)$, the elements $\mathbf{a}[2]$ and $\mathbf{a}[5]$ are wet. Thus, we find $\mathbf{f} = \mathbf{b} \oplus \mathbf{a} = (\mathbf{f}[1]0\mathbf{f}[3]\mathbf{f}[4]0)$ and $k = 3$ since $k = n - l$.

Within this example, the parity-check matrix is given with:

$$\mathbf{H}_{3\times 5} = \begin{pmatrix} 1\,0\,1\,1\,0 \\ 0\,1\,1\,0\,1 \\ 1\,0\,0\,0\,0 \end{pmatrix}.$$

The sender embeds the message according the following scheme:

2. and 3. Eliminate positions with $\mathbf{f}[i] = 0$ and the corresponding columns in $\mathbf{H}_{k\times n}$

4. $\mathbf{H}_{k\times|\mathrm{Dry}|}(\mathbf{f}_{|\mathrm{Dry}|})^T = \mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T$

$$\begin{pmatrix} 1\,1\,1 \\ 0\,1\,0 \\ 1\,0\,0 \end{pmatrix} \begin{pmatrix} \mathbf{f}[1] \\ \mathbf{f}[3] \\ \mathbf{f}[4] \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \oplus \begin{pmatrix} 1\,0\,1\,1\,0 \\ 0\,1\,1\,0\,1 \\ 1\,0\,0\,0\,0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$
$$= \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$$

4.1 Is $((\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T) \in \mathcal{U}_1)$? No

$\mathcal{U}_1$ is the set of all columns of $\mathbf{H}_{k\times|\mathrm{Dry}|}$

$\mathcal{U}_1 = \{(101), (110), (100)\}$

(011) not in this set: $t = 1, r = 1$

4.2 Is $((\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T) + \mathcal{U}_1) \cap \mathcal{U}_1 = \emptyset$? No

$\{(110), (101), (111)\} \cap \{(101), (110), (100)\} = \{(110), (101)\}$

4.3 Any element i of the intersection $((\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T) + \mathcal{U}_t) \cap \mathcal{U}_r$ is a coset leader of weight $t + r$

$i_1 = (110)$

Find $\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T = i_1 \oplus \mathbf{H}_{k\times|\mathrm{Dry}|}[., j]$ according to Equation (4.3)

$\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T = i_1 \oplus \mathbf{H}_{k\times|\mathrm{Dry}|}[., j]$

$$\begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \oplus \mathbf{H}_{k\times|\mathrm{Dry}|}[., j]$$

$\mathbf{H}_{k\times|\mathrm{Dry}|}[., j_1] = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$, corresponds to $\mathbf{H}_{k\times|\mathrm{Dry}|}[., 1]$

Furthermore, i_1 corresponds to $\mathbf{H}_{k\times|\mathrm{Dry}|}[., j_2] = \mathbf{H}_{k\times|\mathrm{Dry}|}[., 2]$

$$\mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T = \mathbf{H}_{k\times|\mathrm{Dry}|}[.,j_1] \oplus \mathbf{H}_{k\times|\mathrm{Dry}|}[.,j_2]$$
$$\mathbf{f}_{|\mathrm{Dry}|} = (110) \Rightarrow \mathbf{f} = (10100)$$

5. Change the first and the third element in order to embed the message

$\mathbf{b} = \mathbf{a} \oplus \mathbf{f} = (10111) \oplus (10100) = (00011)$.

The receiver extracts the message with $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$:

$$\begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1\,0\,1\,1\,0 \\ 0\,1\,1\,0\,1 \\ 1\,0\,0\,0\,0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}.$$

■

4.2. Matrix Embedding for Large Payloads

Matrix Embedding for Large Payloads also enables the sender to minimize the embedding impact introduced during embedding [52]. The basic idea of Matrix Embedding for Large Payloads is to use stochastic codes of small dimension for syndrome coding. An advantage is that for a large k, the dimension l is small enough to enable exhaustive search in order to find the coset leader.

Therefore, $\mathbf{H}_{k\times n}$ is generated randomly but in a systematic form. Fridrich et al. define $\mathbf{H}_{k\times n}$ with $\mathbf{H}_{k\times n} = [\mathbf{I}_k\ \mathbf{R}_{k\times l}]$, where $\mathbf{I}_k$ is a unity matrix and $\mathbf{R}_{k\times l}$ is a stochastic matrix with $P(0) \approx P(1)$.

The embedding algorithm itself operates on small blocks. Whenever the dimension of the code is small, a solution to Equation (3.33) can be quickly found by exhaustive search or using Look-up Tables (Section 3.5). The embedding algorithm can be summarized as follows according to [52].

1. Find an arbitrary coset member $\mathbf{f_m}$ of coset $\mathcal{C}(\mathbf{emb} \oplus \mathbf{H}_{k\times n}\,\mathbf{a}^T)$, i. e. find an arbitrary flipping pattern $\mathbf{f}$ fulfilling Equation (3.33)

2. Find the flipping pattern with minimum weight according to $\mathbf{f_{emb}} = \mathbf{f_m} \oplus \mathrm{argmin}_{\mathbf{c}\in\mathcal{C}}\,\mathrm{w}(\mathbf{f_m} \oplus \mathbf{c})$[15]

3. Determine the stego sequence $\mathbf{b} = \mathbf{a} \oplus \mathbf{f_{emb}}$.

[15] Recall the possibility to utilize the coherence denoted in Equation (3.38) for determining coset $\mathcal{C}(\mathbf{emb} \oplus \mathbf{H}_{k\times n}\,\mathbf{a}^T)$.

Since $\mathbf{H}_{k\times n}$ is systematic, finding $\mathbf{f_m}$ is easy. We find, e. g., $\mathbf{f_m} = [\mathbf{emb} \oplus \mathbf{H}_{k\times n}\,\mathbf{a}^T\ \mathbf{0}]$, where $\mathbf{0} \in \{0\}^l$. Thus, the most time consuming part is finding the flipping pattern with minimum weight $\mathbf{f_{emb}}$. Note that $\mathbf{l}_k$ can be publicly known. However, n has to be communicated.

An example of the algorithm is given below:

Example:

The $(6, 3)$ stochastic matrix is given by means of its parity-check matrix with

$$\mathbf{H}_{3\times 6} = \begin{pmatrix} 1\,0\,0\,1\,1\,0 \\ 0\,1\,0\,0\,1\,0 \\ 0\,0\,1\,1\,0\,1 \end{pmatrix} = [\mathbf{I}_3\,\mathbf{R}_{3\times 3}].$$

The sender would like to embed the confidential message $\mathbf{emb} = (010)$ into the cover sequence $\mathbf{a} = (100101) = [\mathbf{a}_k\ \mathbf{a}_l]$ with $l = 3, k = 3$. The embedding process is carried out as follows:

1. Find an arbitrary coset member $\mathbf{f_m}$[16]

   $$\begin{aligned}\mathbf{f_m} &= [\mathbf{emb} \oplus \mathbf{H}_{k\times n}\,\mathbf{a}^T\ \mathbf{0}_l] \\ &= [(010) \oplus (000)\ 000] \\ &= (010000)\end{aligned}$$

2. Find the flipping pattern with minimum weight according to $\mathbf{f_{emb}} = \mathbf{f_m} \oplus \operatorname{argmin}_{\mathbf{c}\in\mathcal{C}} \mathrm{w}(\mathbf{f_m} \oplus \mathbf{c})$

   $$\mathbf{f_{emb}} = (010000) \oplus (000000) = (010000)$$

3. Determine the stego sequence $\mathbf{b} = \mathbf{a} \oplus \mathbf{f_{emb}}$

   $$\mathbf{b} = (100101) \oplus (010000) = (110101).$$

The receiver extracts the message with $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$:

$$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1\,0\,0\,1\,1\,0 \\ 0\,1\,0\,0\,1\,0 \\ 0\,0\,1\,1\,0\,1 \end{pmatrix} \begin{pmatrix} 1 \\ 1 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}.$$

■

[16] Note that it is also possible to chose $\mathbf{f_m}$ as $[\mathbf{emb}\ \mathbf{0}_l]$ as described in [58].

5. Embedding Based on Deterministic Parity-Check Matrices

Within this chapter, we give a description of algorithms for syndrome coding based on deterministic parity-check matrices. We describe the basic algorithms for embedding based on a uniform profile of embedding impact. Moreover, we outline the concept of embedding based on a general profile, i. e., suited for Wet Paper Steganography, exemplarily for BCH Codes.

5.1. Embedding Based on Hamming Codes

This approach for minimizing the number of embedding changes has been addressed first by Crandall [16], and has become well known as Matrix Embedding. Within his paper, Crandall proposes different techniques for lower rate embedding. As an example, he proposes the use of a $(3, 1, f_k = 1)$ Hamming Code [61] in order to embed two message bits within three cover bits by making at most one embedding change [16].

Generally, Hamming Codes (HC) can be described for a couple of parameters. By means of applying Hamming Codes for syndrome coding, the sender embed k bits in $2^k - 1$ cover bits by at most one embedding change, where $k > 1$. Therefore, sender and receiver share a $k \times (2^k - 1)$ binary matrix $\mathbf{H}_{k\times n}$ that contains all non-zero binary vectors of length k as its columns.

Note that codes like Hamming Codes and the non-primitive Golay Code are called perfect codes. They fulfill the Hamming bound $2^k \geq \sum_{i=0}^{f_k} \binom{n}{i}$ with equality. For instance for the $(7, 4, f_k = 1)$ Hamming Code, we obtain $2^3 = \binom{7}{0} + \binom{7}{1} = 8$. Every syndrome, excluding the syndrome with weight zero, is mapped to an error pattern $\mathbf{f}$ with $\mathrm{w}(\mathbf{f}) = f_k = 1$.

For the application in steganography, this property ensures the possibility to embed by making at most one embedding change. For Hamming Codes it is possible to simply analyze the syndrome derived from $\mathbf{s} = \mathbf{H}_{k\times n}\mathbf{a}^T \oplus \mathbf{emb}$, where $\mathbf{s}$ is related to a position in $\mathbf{H}_{k\times n}$. This position has to be flipped in $\mathbf{a}$ in order to embed. In this case, the weight of sequence $\mathbf{f}$ is at most 1. The algorithm for embedding is given below:

1. Determine $\mathbf{s} = \mathbf{H}_{k\times n}\mathbf{a}^T$
2. Whenever $\mathbf{s} \neq \mathbf{emb}$, flip the element corresponding to column $\mathbf{s} \oplus \mathbf{emb}$ in $\mathbf{a}$.

Generally, Hamming Codes are easy to implement with low computational complexity. Thus, these codes have been used in the steganographic community [96]. An example for the Hamming Code is given below.

Example:
The parity-check matrix of the $(7, 4, f_k = 1)$ HC is given with:

$$\mathbf{H}_{3\times 7} = \begin{pmatrix} 1\,1\,1\,1\,0\,0\,0 \\ 1\,1\,0\,0\,1\,1\,0 \\ 1\,0\,1\,0\,1\,0\,1 \end{pmatrix}.$$

The sender would like to embed the confidential message $\mathbf{emb} = (100)$ into the cover sequence $\mathbf{a} = (0101111)$. The embedding process is carried out as follows:

1. Determine $\mathbf{s} = \mathbf{H}_{k\times n}\mathbf{a}^T$

$$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} 1\,1\,1\,1\,0\,0\,0 \\ 1\,1\,0\,0\,1\,1\,0 \\ 1\,0\,1\,0\,1\,0\,1 \end{pmatrix} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

2. Determine $\mathbf{s} \oplus \mathbf{emb}$

$$\begin{pmatrix} 0 \\ 1 \\ 0 \end{pmatrix} \oplus \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1 \\ 1 \\ 0 \end{pmatrix} \text{ and}$$

flip the element of the corresponding column in $\mathbf{a}$:

$\mathbf{b} = \mathbf{a} \oplus \mathbf{f} = (0101111) \oplus (0100000) = (0001111)$.

The receiver extracts the message with $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$:

$$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1\,1\,1\,1\,0\,0\,0 \\ 1\,1\,0\,0\,1\,1\,0 \\ 1\,0\,1\,0\,1\,0\,1 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 1 \\ 1 \end{pmatrix}.$$

■

5.2. Embedding Based on BCH Codes

Within this section, we present different approaches for syndrome coding based on BCH Codes following the description given in Section 3.1. First, we give some approaches considering a uniform profile of embedding impact in Section 5.2.1. Afterwards, we describe how to adapt these approaches to a general profile in Section 5.2.2.

5.2.1. Embedding Based on a Uniform Profile of Embedding Impact

In this section, we describe algorithms based on a uniform profile of embedding impact (see Section 2.5.1). In order to minimize the impact of embedding and by so doing maximize the embedding efficiency, the number of embedding changes has to be reduced. Note that no elements of the cover are excluded from the embedding process.

Generally, there are two different approaches for calculating a syndrome: one based on the parity-check matrix $\mathbf{H}_{k\times n}$ and another based on the generator polynomial $g(x)$. Within this section, we focus on algorithms based on the parity-check matrix $\mathbf{H}_{k\times n}$. Therefore, we describe several approaches proposed by Schönfeld and Winkler [85, 86, 87] as well as the approach by Zhang et al. for Fast BCH Syndrome Coding for $f_k = 2$ [101]. For more information about approaches based on $g(x)$, we refer to Appendix A and [86, 87].

5.2.1.1. Classical Approach

As already mentioned in Section 3.5, finding a coset leader is an NP complete problem that requires exhaustive search. By means of Look-up Tables, it is possible to speed up the process of finding the optimal solution. For every coset (see Section 3.3), the 2^l sequences leading to one sequence **emb** are stored. This will reduce the complexity, since we have to consider only 2^l sequences for exhaustive search.

For the Classical Approach, proposed by Schönfeld and Winkler in [86], the goal is to achieve $\mathbf{s} = \mathbf{H}_{k\times n}\,(\mathbf{a} \oplus \mathbf{f})^T = \mathbf{H}_{k\times n}\,\mathbf{b}^T = \mathbf{emb}$ with $d_\rho(\mathbf{a}, \mathbf{b})$ being minimal. Embedding is done by means of exhaustive search according to Equation (3.36):

1. Determine $\mathbf{f} = \operatorname{argmin}_{\mathbf{f}\in\mathcal{C}(\mathbf{H}_{k\times n}\mathbf{a}^T\oplus\mathbf{emb})} d_\rho(\mathbf{a}, \mathbf{a}\oplus\mathbf{f})$
2. Determine $\mathbf{b} = \mathbf{a} \oplus \mathbf{f}$.

An example is given below.

Example:

The $(15, 7, f_k = 2)$ BCH Code is given with its generator polynomial $g(x) = x^8 + x^7 + x^6 + x^4 + 1$. We find $h(x) = \frac{f(x)}{g(x)} = \frac{x^{15}+1}{x^8+x^7+x^6+x^4+1} = x^7 + x^6 + x^4 + 1$.

Cyclical shifting of the coefficients of $h(x)$ with $h = (000000011010001)$ leads to

$$H_{8\times 15} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0 \\ 0\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0 \\ 1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0 \end{pmatrix}.$$

The sender would like to embed the confidential message $\mathbf{emb} = (01011010)$ into the cover sequence $\mathbf{a} = (001101010001010)$.

Therefore, he determines $\mathbf{f} = \text{argmin}_{\mathbf{f}\in\mathcal{C}(\mathbf{H}_{k\times n}\mathbf{a}^T\oplus\mathbf{emb})}\, d_\rho(\mathbf{a}, \mathbf{a}\oplus\mathbf{f})$, i. e., he searches for the flipping pattern with minimum weight within coset $\mathcal{C}((10111110) \oplus (01011010)) = \mathcal{C}(11100100)$ according to:

1. Determine the flipping pattern $\mathbf{f}$: Based on an exhaustive search within coset $\mathcal{C}(11100100)$, we find a coset leader $\mathbf{f} = (000000000100101)$, i. e., a flipping pattern with minimum weight (see Appendix B). Note that there are $2^k = 2^8$ different cosets, whereas each coset consists of $2^l = 2^7$ elements (Section 3.3).

2. Determine the stego sequence

   $\mathbf{b} = \mathbf{a}\oplus\mathbf{f} = (001101010001010)\oplus(000000000100101) = (001101010101111)$.

The receiver extracts the message with $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$:

$$\begin{pmatrix} 0\\1\\0\\1\\1\\0\\1\\0 \end{pmatrix} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0 \\ 0\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0 \\ 1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0 \end{pmatrix} \begin{pmatrix} 0\\0\\1\\1\\0\\1\\0\\1\\0\\1\\0\\1\\1\\1\\1 \end{pmatrix}.$$

■

Of course, the approach based on Look-up Tables is limited by storage since 2^l sequences have to be stored within 2^k tables. Consequently, embedding based on the classic approach is really complex and therefore time consuming. Thus, we would like to outline more efficient embedding strategies as presented in [85, 86, 87, 101].

5.2.1.2. Fast BCH Syndrome Coding for $f_k = 2$

This approach, proposed by Zhang et al., is based on the $(n, l, f_k = 2)$ BCH Code and can hide the same amount of data as the algorithms described in the previous section with less computational time [101]. The authors describe a method for easily finding the multiple solutions for data hiding based on an advanced way to find roots over Galois Fields.

Within their approach, only applicable for $f_k = 2$, the parity-check matrix is described as:

$$\mathbf{H} = \begin{pmatrix} 1 & r & r^2 & \dots & r^{n-1} \\ 1 & (r^3) & (r^3)r^2 & \dots & (r^3)^{n-1} \end{pmatrix}, \tag{5.1}$$

where r is the primitive element in $GF(2^m)$.

Moreover, the cover sequence $\mathbf{a}$ as well as the stego sequence $\mathbf{b}$, are represented as polynomials $a(x) = a_0 + a_1x + a_2x^2 + a_3x^3 + \ldots + a_{n-1}x^{n-1}$ and $b(x) = b_0 + b_1x + b_2x^2 + b_3x^3 + \ldots + b_{n-1}x^{n-1}$, respectively.

The algorithm itself is again based on syndrome coding. Thus, the sender again has to determine a solution to:

$$\mathbf{emb} = \mathbf{H}\,\mathbf{b}^T, \tag{5.2}$$

in order to embed.

As described in Section 3.5, the goal is to minimize the distortion introduced during embedding. Within this approach, the difference $\mathbf{f} = \mathbf{a} \oplus \mathbf{b}$ is represented by the polynomial $f(x) = x^{u_1} + x^{u_2} + x^{u_3} + \ldots + x^{u_l}$, where u gives the positions of the elements in $\mathbf{a}$ that have to be flipped in order to obtain $\mathbf{b}$.

Based on this coherence and Equation (5.2), we find:

$$\mathbf{emb} = \mathbf{H}\,(\mathbf{a} \oplus \mathbf{f})^T \tag{5.3}$$

$$\mathbf{H}\,\mathbf{f}^T = \mathbf{emb} \oplus \mathbf{H}\,\mathbf{a}^T, \tag{5.4}$$

and formulate a system of linear equations:

$$\mathbf{s} = [\mathbf{s}_1\ \mathbf{s}_2]^T = \mathbf{H}\,\mathbf{f}^T. \tag{5.5}$$

The goal from the steganographic point of view is to find the minimal number of flips for $f(x)$ satisfying Equation (5.5). As already mentioned, Equation (5.5) has 2^l possible solutions which form a coset, where the solution with the smallest number of flips is called coset leader $\mathbf{c}_L$.

The proposed algorithm is based on finding the multiple solutions easily based on an advanced way to find roots r over Galois Fields. Therefore, the system of linear equations is written as:

$$\mathbf{s}_1 = r^{u_1} + r^{u_2} + r^{u_3} + \ldots + r^{u_l} \tag{5.6}$$

$$\mathbf{s}_2 = (r^3)^{u_1} + (r^3)^{u_2} + (r^3)^{u_3} + \ldots + (r^3)^{u_l}, \tag{5.7}$$

where $r^{u_1}, \ldots, r^{u_l}$ are unknown values. Note that the powers of the roots refer to the indexes of the coefficients to be modified in vector $f(x)$.

The authors propose to utilize the method of Zhao et al. based on fast Look-up Tables for finding roots for quadratic and cubic polynomials [108]. Since the roots are calculated before embedding and are stored in Look-up Tables, the method does not require exhaustive search to find roots.

A brief description on the coherences used to build the tables $\mathbf{q}$ and $\mathbf{c}$, containing the roots of the quadratic and cubic polynomial, respectively, can be found in Appendix C. Furthermore, a table $\mathbf{tab}$ denotes entries that have 3 roots. For more information about the mathematical background, we refer to [101].

The proposed data hiding scheme for $\mathbf{s} \neq \mathbf{0}$ can be summarized as follows:

1. One flip is required: $\mathbf{s}_2 + \mathbf{s}_1^3 = 0$
   - Root is $\beta_1 = \mathbf{s}_1$, the flip location is $u_1 = log(\beta_1)$, flip pattern is $f(x) = x^{u_1}$

2. More flips: $\mathbf{s}_2 + \mathbf{s}_1^3 \neq 0$
   - Determine $o = \frac{\mathbf{s}_2 + \mathbf{s}_1^3}{\mathbf{s}_1^3}$ as index of the quadratic look-up table $\mathbf{q}$
   - Basic root $y_1 = \mathbf{q}(o)$ is obtained from $\mathbf{q}$ in row o
   - Two flips: $y_1 \neq -1$
     - Roots of the flip location polynomial are $\beta_1 = \mathbf{s}_1 y_1$ and $\beta_2 = \mathbf{s}_1 y_1 + \mathbf{s}_1$
     - Flip locations are $u_1 = log(\beta_1)$ and $u_1 = log(\beta_2)$
     - Flip pattern is $f(x) = x^{u_1} + x^{u_2}$
   - Three flips: $y_1 = -1$
     - There are at most D flip patterns $f(x)$
     - For each D_i with $i = 1, 2, \ldots, D$, find $o = \mathbf{tab}(d)$, where o is the index of the Look-up Table **cubic**
     - Get 3 basic roots: $y_1 = \mathbf{cubic}(o, 1),\ y_2 = \mathbf{cubic}(o, 2)$ and $y_3 = \mathbf{cubic}(o, 3)$
     - Roots are $\beta_1 = py_1 + \mathbf{s}_1,\ \beta_2 = py_2 + \mathbf{s}_1$ and $\beta_3 = py_3 + \mathbf{s}_1$ with $p = (\frac{\mathbf{s}_1^3 + \mathbf{s}_2}{o})^{1/3}$
     - Flip locations are $u_i = log(\beta_i)$ with $i = 1, 2, 3$
     - Flip pattern is $f(x) = x^{u_1} + x^{u_2} + x^{u_3}$
3. Determine the stego sequence $b(x) = a(x) + f(x)$

A practical embedding algorithm applying this approach can be found in [83].

5.2.1.3. Systematic Parity-Check Matrix

Note that embedding by finding a coset leader is extremely complex if the code parameters increase. Moreover, the approach presented in Section 5.2.1.2 is applicable only for $f_k = 2$.

For more powerful BCH Codes, we can speed up the process of finding a solution using a parity-check matrix in systematic form, similar to the approach in Section 4.2. Thus, we are able to embed faster and with lower complexity.

An approach proposed by Schönfeld in [85] is based on a parity-check matrix $\mathbf{H}_{k \times n}$, whose columns are determined with $\mathrm{mod}(x^i, g(x)),\ (i = 0, 1, ..., (n-1))$. In this case, the parity-check matrix has a special structure. We find $\mathbf{H}_{k \times n} = [\mathbf{C}^* \, \mathbf{I}_k] = [\mathbf{R}_{k \times l} \, \mathbf{I}_k]$, whereas $\mathbf{C}^*$ is the basis of the code. Thus, $\mathbf{H}_{k \times n}$ corresponds to a systematic code, where $\mathbf{R}_{k \times l}$ covers the l information bits and $\mathbf{I}_k$ is an identity matrix covering the k parity bits[17].

[17] Note that within paper [85] also the approach based on $\mathbf{G} = [\mathbf{I}_l \, \mathbf{C}^*]$ and thus, $\mathbf{H} = \left[\mathbf{C}^{*T} \, \mathbf{I}_k\right]$ was investigated.

Within [85], Schönfeld describes different approaches for embedding based on a systematic parity-check matrix. An interesting approach can be described as follows: In order to accelerate the process of finding a solution to Equation (3.36), Schönfeld considers a pre-flipping approach. For that purpose, $\mathbf{a}$ is divided according to the systematic structure of $\mathbf{H}_{k\times n}$ in $\mathbf{a} = [\mathbf{a}_l\, \mathbf{a}_k]$,[18].

Within the first step of the algorithm, Schönfeld propose to pre-flip within the l information bits. The goal is to find a sequence $\mathbf{f}_l$ of length l (see Section 3.5) with:

$$\mathbf{f}_l = \underset{\mathbf{f}_l \in \mathbb{F}_2^l}{\operatorname{argmin}}(\mathrm{w}(\mathbf{f}_l) + \mathrm{w}(\underbrace{\mathbf{H}_{k\times n}([\mathbf{a}_l \oplus \mathbf{f}_l \;\; \mathbf{a}_k])^T}_{\mathbf{f}_k} \oplus \mathbf{emb})). \tag{5.8}$$

In a second step, the combination of $\mathbf{s}$ and $\mathbf{emb}$ gives the positions of the additional bits that have to be flipped within the remaining k bits $\mathbf{a}_k$ related to the identity matrix $\mathbf{I}_k$, i. e., $\mathbf{f}_k$. As a result, $\mathbf{b}$ fulfills $\mathbf{emb} = \mathbf{H}_{k\times n}\,\mathbf{b}^T$. Note that for this approach, the search area is reduced to 2^l, i. e., the complexity is reduced to the task of finding an optimal sequence $\mathbf{f}_l$. For more details, we refer to [85].

The algorithm based on a systematic parity-check matrix $\mathbf{H}_{k\times n} = [\mathbf{R}_{k\times l}\, \mathbf{I}_k]$ can be summarized as follows:

1. Divide $\mathbf{a}$ in $[\mathbf{a}_l\, \mathbf{a}_k]$
2. Find a sequence $\mathbf{f}_l = \operatorname{argmin}_{\mathbf{f}_l \in \mathbb{F}_2^l}(\mathrm{w}(\mathbf{f}_l) + \mathrm{w}(\mathbf{H}_{k\times n}([\mathbf{a}_l \oplus \mathbf{f}_l \;\; \mathbf{a}_k])^T \oplus \mathbf{emb}))$
3. Combine $\mathbf{s}$ and $\mathbf{emb}$ to get the positions of the additional bits that have to be flipped within the remaining k bits
4. Determine the stego sequence $\mathbf{b} = [\mathbf{a}_l \oplus \mathbf{f}_l \;\; \mathbf{a}_k \oplus \underbrace{(\mathbf{s}^T \oplus \mathbf{emb})}_{\mathbf{f}_k}]$

Note that it is not necessary to investigate all 2^l pre-flipping pattern $\mathbf{f}_l$ within this approach: Starting with flipping pattern with weight 1, it is feasible to stop whenever finding a sequence $\mathbf{f}_l$ fulfilling Equation (5.8). More details are given in [85].

An example of the algorithm is given below:

Example:

The $(7, 4, 1)$ BCH Code is given by means of its parity-check matrix with $\mathrm{mod}(x^i, g(x))$, $(i = 0, 1, ..., (n-1))$ and $g(x) = x^3 + x + 1 = (1011)$

$$\mathbf{H}_{3\times 7} = \begin{pmatrix} 1\,1\,1\,0\,1\,0\,0 \\ 1\,1\,0\,1\,0\,1\,0 \\ 1\,0\,1\,1\,0\,0\,1 \end{pmatrix} = [\mathbf{R}_{k\times l}\, \mathbf{I}_k].$$

[18] We translate the structure of $\mathbf{H}_{k\times n}$ into the structure of $\mathbf{a}$, i. e., the first l positions in $\mathbf{a}$ are information bits, the remaining k positions are parity bits.

The sender would like to embed the confidential message $\mathbf{emb} = (001)$ into the cover sequence $\mathbf{a} = (1011101) = [\mathbf{a}_l\ \mathbf{a}_k]$ with $l = 4, k = 3$. The embedding process is carried out as follows:

2. Find a sequence $\mathbf{f}_l = \text{argmin}_{\mathbf{f}_l \in \mathbb{F}_2^l}(\text{w}(\mathbf{f}_l) + \text{w}(\mathbf{H}_{k\times n}([\mathbf{a}_l \oplus \mathbf{f}_l\ \ \mathbf{a}_k])^T \oplus \mathbf{emb}))$

   $\mathbf{f}_l = (0010)$

3. Determine the additional positions that have to be flipped

   $\mathbf{s}^T \oplus \mathbf{emb} = (001) \oplus (001) = (000)$

4. Determine the stego sequence $\mathbf{b} = [\mathbf{a}_l \oplus \mathbf{f}\ \mathbf{a}_k \oplus (\mathbf{s}^T \oplus \mathbf{emb})]$

$$\begin{aligned}\mathbf{b} &= [\mathbf{a}_l \oplus \mathbf{f}_l\ \ \mathbf{a}_k \oplus (\mathbf{s}^T \oplus \mathbf{emb})] \\ &= [(1011) \oplus (0010)\ \ (101) \oplus (000)] \\ &= (1001101)\end{aligned}$$

The receiver extracts the message with $\mathbf{emb} = \mathbf{H}_{k\times n}\,\mathbf{b}^T$:

$$\begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1\,1\,1\,0\,1\,0\,0 \\ 1\,1\,0\,1\,0\,1\,0 \\ 1\,0\,1\,1\,0\,0\,1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \\ 1 \end{pmatrix}.$$

■

5.2.1.4. Working with $\mathcal{C}(\mathbf{0})$

Since embedding by means of the classic approach as described in Section 5.2.1.1 requires a lot of storage to store the Look-up Tables, we describe the adaption of the embedding algorithm according to Section 3.5.

This approach, proposed by Schönfeld and Winkler in [87], utilizes the coherences described in Equation (3.38) where each coset can easily be determined by means of $\mathcal{C}(\mathbf{0})$. Thus, only one Look-up Table containing all codewords, i. e., $\mathcal{C}(\mathbf{0}) = \mathcal{C}$ is required for embedding.

The embedding process, can be summarized as follows:

1. Find an arbitrary coset member $\mathbf{f_m}$ of coset $\mathcal{C}(\mathbf{emb} \oplus \mathbf{H}_{k\times n}\,\mathbf{a}^T)$, i. e. find an arbitrary flipping pattern $\mathbf{f}$ fulfilling Equation (3.33)

2. Find the flipping pattern with minimum weight according to $\mathbf{f_{emb}} = \mathbf{f_m} \oplus \text{argmin}_{\mathbf{c}\in\mathcal{C}}\, \text{w}(\mathbf{f_m} \oplus \mathbf{c})$

3. Determine the stego sequence $\mathbf{b} = \mathbf{a} \oplus \mathbf{f_{emb}}$.

Note that the embedding process is similar to those described in Section 4.2. However, the parity-check matrix does not have to be systematic. Consequently, the search for an arbitrary coset member $\mathbf{f_m}$ is not as simple, i. e., the coset member has to be determined by a search within the 2^n sequences of length n.

5.2.1.5. Working with the reduced $\mathcal{C}(\mathbf{0})_{red}$

Working with $\mathcal{C}(\mathbf{0})$, as described in the previous section, reduces the memory requirements considerably. However, for large code parameters, especially a high number of information bits l, not only a lot of time for the exhaustive search in $\mathcal{C}(\mathbf{0})$ but also a lot of storage is required for a Look-up Table containing all 2^l codewords. Because of this, an approach to further reduce time and memory complexity by reducing $\mathcal{C}(\mathbf{0})$ was discussed by Schönfeld and Winkler in [87].

Our first investigations were motivated by the symmetrical distribution of the weight of the codewords of a linear code. For example the code alphabet of the $(15, 7, 2)$ BCH Code contains $|\mathcal{C}| = 2^l = 128$ codewords distributed as follows: $\text{w}(\mathcal{C}) = (1, 0, 0, 0, 0, 18, 30, 15, 15, 30, 18, 0, 0, 0, 0, 1)$, where we find, e. g., one codeword with weight 0 and 18 codewords with weight 5.

Thus, the question is how does a reduction of $\mathcal{C}$ influence the result of the embedding process. Is it sufficient to store only 2^{l-1} codewords with weight $\text{w}(\mathbf{c}) \leq \lfloor \frac{n}{2} \rfloor$ or maybe even less in the Look-up Table?

Investigations have shown that reducing the searching area to 2^{l-1} sequences with:

$$\mathcal{C}(\mathbf{0})_{red} = \left\{ \mathbf{c} \mid \text{w}(\mathbf{c}) \leq \left\lfloor \frac{n}{2} \right\rfloor \right\} \tag{5.9}$$

will result in a slightly higher average number of embedding changes R_a compared to the classical approach ($R_{a,classic}$).

To give some examples [87]:

- For the $(15, 7, 2)$ BCH Code this approach achieves $R_a = 2.508$, while $R_{a,classic} = 2.461$, and
- For the $(23, 12, 3)$ Golay Code $R_a = 2.864$ can be achieved, while $R_{a,classic} = 2.853$.

Nevertheless, the embedding complexity in terms of time and memory can be reduced considerably. Note that the deviation of R_a is caused by the reduced $\mathcal{C}(\mathbf{0})_{red}$. Each coset member $\mathbf{f_m}$ leads to different reduced cosets:

$$\mathcal{C}(\mathbf{emb})_{red} = \mathbf{f_m} \oplus \mathcal{C}(\mathbf{0})_{red}. \tag{5.10}$$

Of course, each of these reduced cosets is only part of $\mathcal{C}(\mathbf{emb}) = \mathbf{f_m} \oplus \mathcal{C}(\mathbf{0})$. As a result, it is not always possible to find the coset leader.

In order to minimize the differences between $R_{a,classic}$ and R_a, it seems reasonable to find different sequences $\mathbf{f_m}$ and thus to evaluate different reduced cosets. Of course, this approach is more time consuming since the time complexity increases by the number num of evaluated sequences $\mathbf{f_m}$ and thus by the number of different reduced cosets. However, this additional time complexity is negligible, as long as $num2^l \ll 2^n$ and $num \ll 2^k$ respectively.

Table 5.1.: Influence of num Sequences $\mathbf{f_m}$ when Working with the Reduced Coset $\mathcal{C}(\mathbf{0})_{red}$ According to [87].

(n, l, f_k)	$R_{a,classic}$	$\lvert\mathcal{C}_{red}\rvert$	$R_{a,1\mathbf{f_m}}$	$R_{a,5\mathbf{f_m}}$	$R_{a,10\mathbf{f_m}}$
(15,7,2)	2.461	2^6	2.508	2.461	2.461
(15,11,1)	0.938	2^{10}	0.938	0.938	0.938
(23,12,3)	2.852	2^{11}	2.864	2.853	2.853

Table 5.1 summarizes some of our results. The results confirm that evaluating different sequences $\mathbf{f_m}$, and thereby different reduced cosets $\mathcal{C}(\mathbf{emb})_{red}$, indeed results in an improved R_a. As can be seen, e. g., for the $(15, 7, 2)$ BCH Code, considering 5 different sequences $\mathbf{f_m}$ in the embedding process leads to $R_{a,classic}$.

We also investigated a further reduction of the alphabet. Therefore, it is indeed important to use only those codewords with the lowest weight. Table 5.2 summarizes the results of our investigations [87].

Table 5.2.: Influence of num Evaluated Sequences $\mathbf{f_m}$ when Further Reducing $\mathcal{C}(\mathbf{0})_{red}$ According to [87].

(n, l, f_k)	$R_{a,classic}$	$\lvert\mathcal{C}_{red}\rvert$	$R_{a,1\mathbf{f_m}}$	$R_{a,5\mathbf{f_m}}$	$R_{a,10\mathbf{f_m}}$	$R_{a,15\mathbf{f_m}}$
(31,21,2)	2.482	30443, $\mathrm{w}(\mathbf{c}) < k$	2.490	2.482	2.482	2.482
(31,21,2)	2.482	11533, $\mathrm{w}(\mathbf{c}) < k-1$	2.531	2.482	2.482	2.482
(31,21,2)	2.482	3628, $\mathrm{w}(\mathbf{c}) < k-2$	2.757	2.486	2.483	2.482

In this example, the alphabet $\mathcal{C} = \mathcal{C}(\mathbf{0})$ of the $(31, 21, 2)$ BCH Code was reduced from 2^{21} down to $\approx 2^{12}$ sequences. Nevertheless, we are able to achieve $R_{a,classic}$ evaluating up to 15 different reduced cosets.

The embedding process working with a reduced coset is carried out as described in the previous section.

5.2.1.6. Generalizing the Approach

Even if we discussed this approach for linear codes like BCH Codes, it is not limited to this class of codes. It is also possible to generalize the approach of working with $\mathcal{C}(\mathbf{0})$ as well as the approach working with $\mathcal{C}(\mathbf{0})_{red}$ [87].

Whenever it is possible to bring $\mathbf{H}_{k\times n}$ (e. g., through permuting columns) into a systematic form according to $\mathbf{H}_{k\times n} = [\mathbf{H}_{k\times l}\, \mathbf{H}_{k\times k}]$, where $\mathbf{H}_{k\times k}$ is non-singular, all codewords $\mathbf{c} \in \mathcal{C}$ can be easily determined since they are arranged systematically. Based on the 2^l source sequences $\mathbf{c}^*$ of length l, the $\mathbf{c} \in \mathcal{C}$ is determined by:

$$\mathbf{c} = [\mathbf{c}_l\ \mathbf{c}_k] \tag{5.11}$$

$$= [\mathbf{c}^*\ \ \mathbf{H}_{k\times k}^{-1}\mathbf{H}_{k\times l}\mathbf{c}^{*T}]. \tag{5.12}$$

The resulting codewords $\mathbf{c} \in \mathcal{C}$ are stored in a Look-up Table. Of course a reduction of this Look-up Table is also possible.

5.2.2. Embedding Based on a General Profile of Embedding Impact

Since most practical steganographic applications have to deal with so-called wet elements distributed randomly over the cover, it is necessary to adapt the algorithms described so far for syndrome coding based on deterministic parity-check matrices. Note that we simply exclude the wet elements from embedding, the dry elements have a uniform profile of embedding impact. Consequently, the average number of embedding changes R_a is still an adequate measure for the embedding impacts introduced during embedding.

5.2.2.1. Description of the Basic Approach

In this section, we will describe the adaption of syndrome coding based on BCH Codes to a scenario where conspicuous parts of the cover are excluded as proposed by Schönfeld and Winkler in [86]. Since the sender would like to modify only the changeable elements, the system of linear equations has to be adapted.

In a first step, we divide $\mathbf{H}_{k\times n}$ into $\mathbf{H}_{k\times n} = \left[\mathbf{H}_{k\times|\mathrm{Wet}|}\, \mathbf{H}_{k\times|\mathrm{Dry}|}\right]$ with $n = |\mathrm{Wet}| + |\mathrm{Dry}|$ depending on the positions of wet elements. Thereby, $\mathbf{H}_{k\times|\mathrm{Wet}|}$ is a submatrix of $\mathbf{H}_{k\times n}$ corresponding to wet elements, and $\mathbf{H}_{k\times|\mathrm{Dry}|}$ is a submatrix of $\mathbf{H}_{k\times n}$ corresponding to dry elements.

Moreover, we separate the cover sequence in two parts $\mathbf{a} = [\mathbf{a}_{|\mathrm{Wet}|}\ \mathbf{a}_{|\mathrm{Dry}|}]$. Thus, to embed a confidential message, a sequence $\mathbf{f}_{|\mathrm{Dry}|}$ has to be determined in a way that fulfills:

$$\mathbf{s} = \underbrace{\mathbf{H}_{k\times|\mathrm{Wet}|}\ \mathbf{a}^T_{|\mathrm{Wet}|}}_{\mathbf{s}_{|\mathrm{Wet}|}} \oplus \underbrace{\mathbf{H}_{k\times|\mathrm{Dry}|}\ (\mathbf{a}_{|\mathrm{Dry}|} \oplus \mathbf{f}_{|\mathrm{Dry}|})^T}_{\mathbf{s}_{|\mathrm{Dry}|}} = \mathbf{emb}. \tag{5.13}$$

Since $\mathbf{s}_{|\mathrm{Dry}|} = (\mathbf{s}_{|\mathrm{Wet}|} \oplus \mathbf{emb})$ is invariant, the simplification of the embedding process is similar to the concept of embedding with Wet Paper Codes (Section 4).[19] The embedding algorithm itself can be described as follows:

1. Define the parity-check matrix as $\mathbf{H}_{k\times n} = \left[\mathbf{H}_{k\times|\mathrm{Wet}|}\ \mathbf{H}_{k\times|\mathrm{Dry}|}\right]$
2. Define the cover sequence as $\mathbf{a} = [\mathbf{a}_{|\mathrm{Wet}|}\ \mathbf{a}_{|\mathrm{Dry}|}]$
3. Reformulate Equation (3.33) to

$$\underbrace{\mathbf{H}_{k\times|\mathrm{Wet}|}\ \mathbf{a}^T_{|\mathrm{Wet}|}}_{\mathbf{s}_{|\mathrm{Wet}|}} \oplus \underbrace{\mathbf{H}_{k\times|\mathrm{Dry}|}\ (\mathbf{a}_{|\mathrm{Dry}|} \oplus \mathbf{f}_{|\mathrm{Dry}|})^T}_{\mathbf{s}_{|\mathrm{Dry}|}} = \mathbf{emb}$$

4. Determine the flipping pattern $\mathbf{f}_{|\mathrm{Dry}|}$ according to

$$\mathbf{f}_{|\mathrm{Dry}|} = \underset{\mathbf{f}_{|\mathrm{Dry}|}\in\mathbb{F}_2^{|\mathrm{Dry}|}}{\operatorname{argmin}} (\mathbf{H}_{n\times|\mathrm{Dry}|}(\mathbf{a}_{|\mathrm{Dry}|} \oplus \mathbf{f}_{|\mathrm{Dry}|})^T = \mathbf{s}_{|\mathrm{Dry}|}) \tag{5.14}$$

5. Determine the stego sequence $\mathbf{b} = [\mathbf{a}_{|\mathrm{Wet}|}\ \mathbf{a}_{|\mathrm{Dry}|} \oplus \mathbf{f}_{|\mathrm{Dry}|}]$

Note that the positions of wet elements are distributed randomly, i. e., the number of wet elements and their positions varies from block to block.

Investigations have shown that the cardinality of the cosets for linear codes determined with $\frac{2^n}{2^k} = 2^l$ (see Equation (5.13)) is also true if wet positions are considered, of course reduced by the number of wet elements |Wet| [86]:

$$\frac{2^{n-|\mathrm{Wet}|}}{2^k} \approx 2^{l-|\mathrm{Wet}|} \qquad (|\mathrm{Wet}| = 0, 1, ..., l). \tag{5.15}$$

As it can be seen in Equation (5.15), the number of wet elements is limited by l. However, this equation holds up only to a certain bound with equality. With increasing number of wet elements, there exist no longer exactly $2^{l-|\mathrm{Wet}|}$ sequences $\mathbf{f}_{|\mathrm{Dry}|}$ for each syndrome. Discarding the message length $|\mathbf{emb}|$ would solve the problem.

[19] Note that both concepts are proposed independently. Similar to the concept of embedding with Wet Paper Codes, we propose to separate the parity-check matrix into $\mathbf{H}_{k\times n} = \left[\mathbf{H}_{k\times|\mathrm{Wet}|}\ \mathbf{H}_{k\times|\mathrm{Dry}|}\right]$. However, we also propose to separate the cover sequence in two parts $\mathbf{a} = [\mathbf{a}_{|\mathrm{Wet}|}\ \mathbf{a}_{|\mathrm{Dry}|}]$ and calculate partial syndromes $\mathbf{s}_{|\mathrm{Dry}|}$ and $\mathbf{s}_{|\mathrm{Wet}|}$.

Furthermore, the search for the flipping pattern $\mathbf{f}_{|\mathrm{Dry}|}$ is realized in a different way. Due to the evenly distributed coset members considering a deterministic matrix (2^l elements within each coset), the Look-up Tables can be determined beforehand.

5.2.2.2. Description of the Improved Approach

The basic algorithm described above, can also be improved in terms of stepping up the process of solving Equation (5.14). Again, it is possible to work with a Look-up Table containing all codewords as described by Schönfeld and Winkler in [87]. However, it is necessary to consider the fact that the positions of wet elements are distributed randomly, i. e., the number of wet elements and their positions varies from block to block.

Thus, in a first step, it is necessary to reduce the sequences within the Look-up Table by the wet positions for each block $\mathbf{a}$ of length n: $\mathcal{C}(\mathbf{0})_{|\mathrm{Dry}|} = \mathcal{C}_{|\mathrm{Dry}|}$.

The embedding process can be described as follows:

1. Find an arbitrary coset member $\mathbf{f}_{\mathbf{m},|\mathrm{Dry}|}$ fulfilling (5.13)
2. Find the flipping pattern with minimum weight based on $\mathcal{C}(\mathbf{0})_{|\mathrm{Dry}|}$ with $\mathbf{f}_{\mathbf{emb}} \in \mathcal{C}(\mathbf{emb})_{|\mathrm{Dry}|}$

$$\mathbf{f}_{\mathbf{emb}} = \mathbf{f}_{\mathbf{m},|\mathrm{Dry}|} \oplus \underset{\mathbf{c}_{|\mathrm{Dry}|} \in \mathcal{C}_{|\mathrm{Dry}|}}{\operatorname{argmin}} \mathrm{w}(\mathbf{f}_{\mathbf{m},|\mathrm{Dry}|} \oplus \mathbf{c}_{|\mathrm{Dry}|}) \tag{5.16}$$

3. Determine the stego sequence $\mathbf{b} = \left[\mathbf{a}_{|\mathrm{Dry}|} \oplus \mathbf{f}_{\mathbf{emb}} \;\; \mathbf{a}_{|\mathrm{Wet}|}\right]$

Again, it is possible to reduce the alphabet $\mathcal{C}$ and therefore to reduce the number of sequences that have to be considered by exhaustive search. This reduction will also result in different reduced cosets for different sequences $\mathbf{f}_{\mathbf{m},|\mathrm{Dry}|}$. Thus, it is again advantageous to evaluate several sequences $\mathbf{f}_{\mathbf{m},|\mathrm{Dry}|}$ and consequently different reduced cosets in order to achieve a maximized embedding efficiency.

5.3. Embedding Based on Simplex Codes

Another class of codes investigated as basis for syndrome coding are Simplex Codes [11]. The application of these matrices was proposed by Fridrich et al. in [52].

Generally, $(2^k - 1, k)$ Simplex Codes are a class of structured codes. More exactly, they are the dual code of the Hamming Code, i. e., their generator matrix is equal to the parity-check matrix of the Hamming Code. Since all non-zero codewords have the same weight 2^{k-1}, Simplex Codes are called constant-weight codes. This weight is also the distance between any two codewords.

The proposed algorithm for syndrome coding using Simplex Codes is given as follows [52]:

1. Find an arbitrary coset member $\mathbf{f}_{\mathbf{m}}$ of coset $\mathcal{C}(\mathbf{emb} \oplus \mathbf{H}_{k\times n}\,\mathbf{a}^T)$, i. e., find an arbitrary flipping pattern $\mathbf{f}$ fulfilling Equation (3.33)

2. Find the flipping pattern with minimum weight according to $\mathbf{f_{emb}} = \mathbf{f_m} \oplus \mathbf{c_{emb}}$

   $\mathbf{c_{emb}} = \text{argmin}_{\mathbf{c}\in\mathcal{C}}\, \text{w}(\mathbf{f_m} \oplus \mathbf{c})$ can be solved using the Fast Hadamard Transform[20].

3. Determine the stego sequence with $\mathbf{b} = \mathbf{a} \oplus \mathbf{f_{emb}}$

Note that Fridrich et al. propose to use the Fast Hadamard Transform to find the closest codeword [52]. Thus, this approach is based on quantization, i. e., finding the quantized sequences related to $\mathbf{f_m}$. Based on this quantized vector, the closest codeword $\mathbf{c_{emb}}$ is determined. A short overview on the Hadamard Transformation is given with:

- Form $\hat{\mathbf{f}_\mathbf{m}} = (0, \mathbf{f_m}[1], \ldots, \mathbf{f_m}[2^k - 1])^T$
- Determine $\mathbf{F} = (\mathbf{F}[1], \ldots, \mathbf{F}[2^k]) = \mathbf{F}^{(k)}$

  $\mathbf{F}^{(0)} = (\mathbf{1} - \mathbf{2}\hat{\mathbf{f}_\mathbf{m}})^T$

  $\mathbf{F}^{(i)} = \mathbf{F}^{(i-1)}\mathbf{M}_{2^k}^{(i)}$ for $i = (1, \ldots, k)$
- Find the largest number $\mathbf{F}_{i_0}$ among $\mathbf{F}[1], \ldots, \mathbf{F}[2^k]$
- $\mathbf{u} = [i_0 - 1]_2$
- Determine $\mathbf{c_{emb}} = \sum_{i=1}^{k} \mathbf{u}[i]\mathbf{G}[i, .]^T$

Note that $\mathbf{H}_{2^k}$ is defined as $\mathbf{H}_{2^k} = \mathbf{M}_{\mathbf{2^k}}^{(\mathbf{1})}\mathbf{M}_{\mathbf{2^k}}^{(\mathbf{2})}\ldots\mathbf{M}_{\mathbf{2^k}}^{(\mathbf{k})}$, where $\mathbf{M}_{\mathbf{2^k}}^{(\mathbf{i})} = \mathbf{I}_{2^{k-i}} \otimes \mathbf{H}_2 \otimes \mathbf{I}_{2^{i-1}}$ with $\mathbf{H}_2 = \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix}$. Furthermore, $\otimes$ stands for the Kronecker product.

An example for embedding is given below.

Example: The generator matrix of the $(7, 3, d_{min} = 4)$ Simplex Code is given with:

$$\mathbf{G}_{3\times 7} = \begin{pmatrix} 1\,1\,1\,1\,0\,0\,0 \\ 1\,1\,0\,0\,1\,1\,0 \\ 1\,0\,1\,0\,1\,0\,1 \end{pmatrix}.$$

Note that this generator matrix is equivalent to the parity-check matrix of the $(7, 4, f_k = 1)$ Hamming Code.

[20] We give only a short overview on the functionality of the Fast Hadamard Transform. For more information, we refer to [52].

The related parity-check matrix can be obtained according to the following coherence: $\mathbf{G}_{l\times n}\,\mathbf{H}_{k\times n}{}^{T} = \mathbf{0}$. This can be easily done in the case of a systematic form of $\mathbf{G}_{l\times n}$ with $\mathbf{G}_{l\times n}{}^{`} = [\mathbf{I}_l\ \mathbf{R}_{l\times k}]$. In this case, we find $\mathbf{H}_{k\times n} = [\mathbf{R}_{l\times k}^T\ \mathbf{I}_k]$. Thus, we find:

$$\mathbf{G}_{3\times 7}^{`} = \begin{pmatrix} 1\,0\,0\,1\,0\,1\,1 \\ 0\,1\,0\,1\,1\,0\,1 \\ 0\,0\,1\,1\,1\,1\,0 \end{pmatrix} = [\mathbf{I}_3\ \mathbf{R}_{3\times 4}]$$

$$\mathbf{H}_{4\times 7} = \begin{pmatrix} 1\,1\,1\,1\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0 \\ 1\,0\,1\,0\,0\,1\,0 \\ 1\,1\,0\,0\,0\,0\,1 \end{pmatrix} = [\mathbf{R}_{3\times 4}^T\ \mathbf{I}_4]. \tag{5.17}$$

The sender would like to embed the confidential message $\mathbf{emb} = (1100)$ into the cover sequence $\mathbf{a} = (1011101)$. The embedding process is carried out as follows:

1. Find an arbitrary coset member $\mathbf{f_m}$ of coset $\mathcal{C}(\mathbf{emb} \oplus \mathbf{H}_{k\times n}\,\mathbf{a}^T)$

   Find $\mathbf{f_m}$, e. g., with $\mathbf{f_m} = [000\ \ \mathbf{emb} \oplus \mathbf{H}_{k\times n}\,\mathbf{a}^T]$

   $\mathbf{emb} \oplus \mathbf{H}_{k\times n}\,\mathbf{a}^T = (1100) \oplus (1000) = (0100)$

   $\mathbf{f_m} = (0000100)$

2. Find the flipping pattern with minimum weight according to $\mathbf{f_{emb}} = \mathbf{f_m} \oplus \mathbf{c_{emb}}$

   $\mathbf{c_{emb}} = \mathrm{argmin}_{\mathbf{c}\in\mathcal{C}}\,\mathrm{w}(\mathbf{f_m} \oplus \mathbf{c}) = (0000000)$

   $\mathbf{f_{emb}} = \mathbf{f_m} \oplus \mathbf{c_{emb}} = (0000100)$

3. Determine the stego sequence with $\mathbf{b} = \mathbf{a} \oplus \mathbf{f_{emb}}$

   $\mathbf{b} = (1011101) \oplus (0000100) = (1011001)$.

The receiver extracts the message with $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$:

$$\begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1\,1\,1\,1\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0 \\ 1\,0\,1\,0\,0\,1\,0 \\ 1\,1\,0\,0\,0\,0\,1 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \end{pmatrix}.$$

■

5.4. Embedding Based on Augmented Simplex Codes

Furthermore, Fridrich et al. propose to apply Augmented Simplex Codes to syndrome coding based on a parity-check matrix [52]. Augmenting a code means adding a codeword to the generator matrix. Fridrich et al. propose to augment with an all 1 vector. The resulting $(2^k-1, k+1)$ code coincides with the punctured first-order Reed-Muller Code [11].

This approach also gives a good performance and can be decoded using a simple modification of the decoding algorithm for Simplex Codes (Section 5.3). It is necessary to modify the 2nd step as follows:

- Run Step 2 with $\hat{\mathbf{f}}_{\mathbf{m}0}$ and $\hat{\mathbf{f}}_{\mathbf{m}1}$, i. e., prepending both a zero and a one
- Obtain two vectors $\mathbf{c}_{\mathbf{emb}0}$ and $\mathbf{c}_{\mathbf{emb}1}$ as result of the
- Choose the vector closer to $\mathbf{f}_{\mathbf{m}}$ as output of the Fast Hadamard Transform.

6. Evaluation of the Algorithms for a Small Code Word Length

Beside a description of the algorithms on a consistent basis, this thesis aims at making the state-of-the-art algorithms comparable to each other. Therefore, we would like to compare the approaches for syndrome coding described in the previous chapters. Recall the following important parameters which should be considered when designing a steganographic algorithm (see Section 2.1.3):

- the security,
- the capacity,
- the success of embedding, and
- the embedding complexity.

Generally, these different parameters should be optimized. The designer of a steganographic algorithm aims at maximizing the capacity as well as the security of the algorithm. Moreover, he tries to minimize the embedding complexity while maximizing the success rate. However, since these are competing goals, it is only possible to balance those four properties dependent on the application.

Note that we do not search for a scalar metric aggregating all these properties. Instead, we evaluate and compare the algorithms described in the previous chapters according to the parameters given above:

- Embedding Based on Stochastic Parity-Check Matrices
  - Block Minimal Method
  - Matrix Embedding for Large Payloads
- Embedding Based on Deterministic Parity-Check Matrices
  - Hamming Codes
  - BCH Codes
  - Simplex Codes
  - Augmented Simplex Codes.

6.1. Evaluation Concerning the Security

Note that steganographic security is mostly influenced by the type of cover media, the method for the selection of places that might be modified, the type of embedding operation and the number of embedding changes. If a scheme fixes the first three influences, the scheme that introduces fewer embedding changes will be less detectable [51].

Thus, we measure the security of a steganographic scheme by means of its achieved embedding efficiency $e = k/d_\rho(\mathbf{a}, \mathbf{b})$ in this thesis. As also mentioned in [51], a higher embedding efficiency translates into better steganographic security. Since we consider either a uniform profile or a general profile, where parts of the cover are denoted as wet and skipped during embedding, $d_\rho(\mathbf{a}, \mathbf{b})$ can be easily measured by means of the average number of embedding changes R_a.

Note that all approaches presented for a small codeword length fulfill the desired property of minimizing the introduced distortion. Thus, they are able to achieve a maximized embedding efficiency. The absolute value of e depends on the code's properties. In the following, we'll give some detailed results of the algorithms presented in the previous chapters as well as a comparison between them (see Figure 6.3).

6.1.1. Block Minimal Method

This approach, proposed by Fridrich et al. and described in Section 4.1, is a block-based scheme embedding small message parts in each block. Therefore, random codes of codimension k are used. The separation into blocks depend on k and the total message length $|\mathbf{m}|$ since the cover is divided into $n_B = |\mathbf{m}|/k$ pseudo-random blocks of length n.

The embedding efficiency was investigated by Fridrich et al. for different values k with $k = 4, \ldots, 18$ and $N = 10^6$. The results were achieved for $5 \cdot 10^4$ dry elements by averaging 100 trials for embedding random messages into the same cover image [45].

Note that Fridrich defines the inverse relative message length α^{-1} as $\alpha^{-1}_{\text{Dry}} = |\text{Dry}|/|\mathbf{m}|$ within this approach, i. e., as the number of dry elements related to the number of embedded bits. Several values for $\alpha_{\text{Dry}^{-1}}$ were considered, e. g., $\alpha_{\text{Dry}^{-1}} = 1.5, 2, \ldots, 12$ [45].

Given $N = 10^6$ and $|\text{Dry}| = 5 \cdot 10^4$, we find for $\alpha^{-1}_{\text{Dry}} = 2$ a message length of $|\mathbf{m}| = 25000$ and thus, for $k = 18$ the number of blocks with $n_B = |\mathbf{m}|/k = 1388$. As a result, the embedding process is based on a stochastic matrix with parameters $(n, l) = (720, 702)$. Generally, we find for this example, $n = 20\frac{k}{\alpha}$. Table 6.1 summarizes some typical parameters for n for different parameters k. Furthermore, we give a link between α^{-1}_{Dry} and α^{-1}.

Table 6.1.: Block Length n for the Block Minimal Method Considering $N = 10^6$ and $|\text{Dry}| = 50000$ for Different Values k.

α_{Dry}^{-1}	α	α^{-1}	$k = 4$	$k = 10$	$k = 18$
1.5	0.033	30	120	300	540
1.75	0.029	35	140	350	630
2	0.025	40	160	400	720
2.5	0.02	50	200	500	900
3	0.017	60	240	600	1080
3.5	0.014	70	280	700	1260
4	0.013	80	320	800	1440

Generally, the embedding efficiency of this approach depends on the code's properties. The results for $k = 18$ are presented in Figure 6.1 according to [45], depicting the embedding efficiency e based on the inverse relative message length α_{Dry}^{-1}. Within this figure, also the upper boundary for the embedding efficiency is given. Note that the more bits of the cover are used for embedding, the smaller the inverse relative message length. Furthermore, a high embedding efficiency is related to a higher security of the steganographic scheme.

The results presented in Figure 6.1 confirm that e increases with growing inverse relative message length α_{Dry}^{-1}. However, the results for $k = 18$ are still far away from the upper boundary. Once the number of dry elements in each block exceeds 2^k, e starts saturating at $k/(1-2^{-k})$ [36]. Note that it is possible to design codes for arbitrary values n and l, denoted with the dotted line.

Moreover, Fridrich et al. report that for a fixed α_{Dry}^{-1} that e increases in an interesting non-monotone manner (see e. g. [47], Figure 3) with increasing k.

6.1.2. Matrix Embedding for Large Payloads

Within this approach, described in Section 4.2, Fridrich et al. utilize the fact that for a low inverse relative message length $\alpha^{-1} = n/k \to 1$, the co-dimension k of the code will be close to the codeword length n. Thus the dimension will be small enough, enabling a fast determining of the coset leader. Generally, for a fixed code dimension l, the inverse relative message length α^{-1} reaches 1 as $n \to \infty$.

Fridrich et al. investigated 200 different randomly generated codes for a fixed codeword length $n \leq 165$ and $l = 14, 10$. They observed that for a low inverse relative message length α^{-1}, the curve is closer to the upper bound [36]. Note that this approach is limited to low inverse relative message lengths.

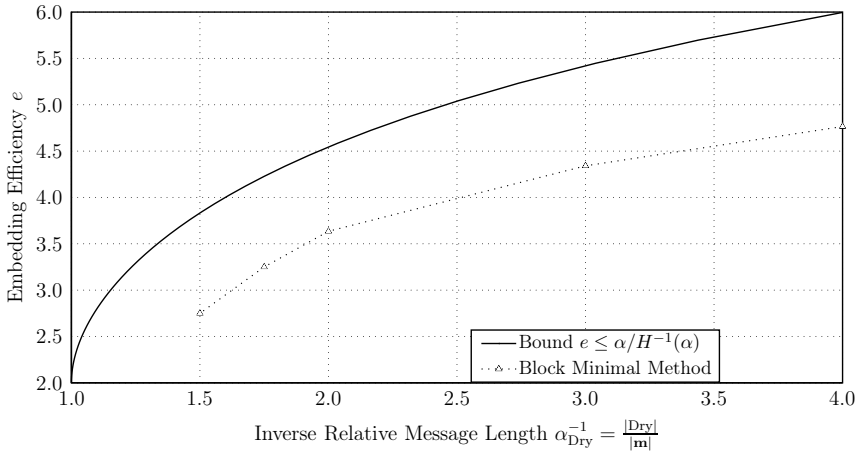

Figure 6.1.: Embedding Efficiency e Dependent on the Inverse Relative Message Length for Embedding Based on the Block Minimal Method ($N = 10^6, |\text{Dry}| = 50000, k = 18$) According to [36].

6.1.3. Hamming Codes

The embedding efficiency for Hamming Codes, as described in Section 5.1, depending on the parameter k is presented in Table 6.2 (see, e. g.,[36]). Note that for Hamming Codes, we find $e = k/(1 - 2^{-k})$.

Table 6.2.: Embedding Efficiency for Hamming Codes.

k	Inverse Relative Message Length $\alpha^{-1} = n/k$	Embedding Efficiency e
2	1.5	2.667
3	2.333	3.429
4	3.75	4.267
5	6.2	5.161
6	10.5	6.093
7	18.143	7.055
8	31.875	8.031
9	56.778	9.018

As it can be found in Table 6.2, Hamming Codes do not improve the embedding efficiency for messages whose inverse relative message length is above 1.5. We find for $k = 1$ the embedding efficiency with $e = 2$ equivalent to embedding without syndrome coding.

However, it is possible to use Hamming Codes for a higher message length using a construction called direct sum. Within this approach, the message is divided

into two or more segments, which are embedded into disjoint parts of the cover using Hamming Codes with different parameters. More information regarding this approach can be found in [14, 36].

6.1.4. BCH Codes

Embedding based on BCH Codes as described in Section 5.2, was investigated by Schönfeld and Winkler in [85, 86, 87, 3]. Schönfeld and Winkler investigated several BCH Codes covering a wide range of parameters.

Furthermore, several embedding approaches for embedding based on BCH Codes were proposed, such as the Classic Approach (Section 5.2.1.1), embedding based on a systematic parity-check matrix (Section 5.2.1.3), embedding based on Coset $\mathcal{C}(\mathbf{0})$ (Section 5.2.1.4) as well as embedding based on a reduced coset (Section 5.2.1.5). Furthermore, Zhang et al. proposed a method for fast embedding based on BCH Codes with $f_k = 2$ (Section 5.2.1.2).

Figure 6.2 compares the results of e for several BCH Codes dependent on the inverse relative message length α^{-1}. Again, the more bits of the cover are used for embedding, the smaller the inverse relative message length. Note that all investigated codes achieve the same results for e, independently of the applied embedding approach[21].

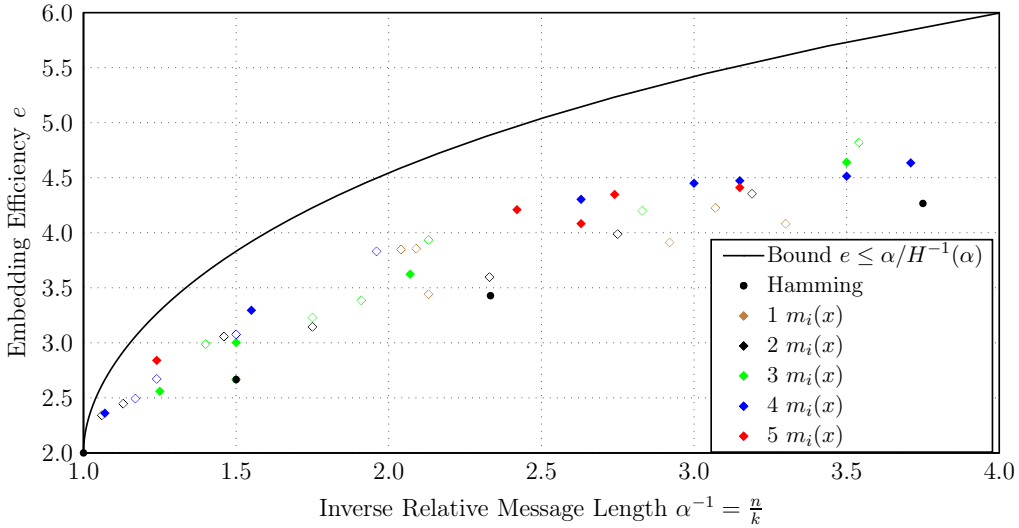

Figure 6.2.: Embedding Efficiency e Dependent on the Inverse Relative Message Length For Several BCH Codes According to [3].

Additionally, Figure 6.2 shows the influence of the generator polynomial - used

[21] Note that a reduced embedding efficiency is possible for the approach based on a reduced coset (Section 5.2.1.5) whenever to few sequences $\mathbf{f_m}$ are evaluated.

for constructing the parity-check matrix $\mathbf{H}_{k\times n}$ - on the achievable embedding efficiency according to Bellmann [3]. In her thesis, Bellmann investigated several strategies for constructing $g(x)$ depending on the choice of the minimal polynomials $m_i(x)$ (see Section 3.2.3.2). Examples of the codes investigated within [3] are given in Appendix D.

Within Figure 6.2, solid symbols stand for primitive BCH Codes with $n = n_{max}$. Non-primitive codes with $n < n_{max}$ are denoted with non-solid symbols. The different colors refer to the different number of minimal polynomials $m_i(x)$ used for the construction of $g(x)$ as basis of the parity-check matrix $\mathbf{H}_{k\times n}$. Note that an increasing number of minimal polynomials $m_i(x)$ results in a bigger block length n as well as in a higher embedding complexity.

Within her thesis, Bellmann found, e. g., that when choosing 2 minimal polynomials for the construction of $g(x)$, it is always advantageous to determine $g(x)$ as $g(x) = m_1(x)m_3(x)$. Moreover, for the construction of $g(x)$ considering 3 minimal polynomials, it is never advantageous to chose $m_1(x), m_3(x)$ and $m_5(x)$ as it is common for channel coding.

As can be seen in Figure 6.2, choosing 4 or 5 minimal polynomials for the construction of $g(x)$ is advantageous considering the embedding efficiency. The results for the investigated BCH Codes are noticeably better than those for the Hamming Code. Note that the codes with $g(x) = m_1(x)$ in case of a primitive BCH Code are equivalent to those of the Hamming Code. Furthermore, the results for Fast BCH are equivalent to those for 2 $m_i(x)$, like $g(x) = m_1(x)m_3(x)$.

Even if it is possible to cover a wide range of parameters, the approach based on deterministic matrices itself is somehow limited. Remember that it is possible to construct BCH Codes for arbitrary values of f_k. However, it is not possible to achieve arbitrary combinations of n and k. One solution to this problem is to reduce the code parameters[22], whereas the results for the embedding efficiency are even worse.

6.1.5. Simplex Codes and Augmented Simplex Codes

Within this approach, described in Section 5.3, k bits are embedded in $2^{k-1} - 1$ elements, leading to an inverse relative message length $\alpha^{-1} = \frac{2^{k-1}-1}{k}$. Fridrich et al. investigated Simplex Codes for $k = 3, \ldots, 11$.

Evaluating the embedding efficiency, Fridrich et al. state that the performance is not as good as for stochastic codes since Simplex Codes do not cover the range of relative message length as densely as random codes. However, they easily reach into the range of $\alpha^{-1} < 1.25$ with a low computational complexity. Thus, they can be seen as examples for structured codes suited to embed large relative message lengths [52, 51].

[22] In coding theory, it is possible to reduce the number of information bits l, and thereby the codeword of length n. The parameter k has to be constant.

Generally, Augmented Simplex Codes, as described in Section 5.4, achieve a better performance than Simplex Codes, but again not as good as for the random linear codes. However, they populate the range for α^{-1} more sparsely.

6.1.6. Comparing the Approaches

The results comparing the algorithms in terms of embedding efficiency are presented in Figure 6.3. We depict the embedding efficiency depending on the inverse relative message length $\alpha^{-1} = \frac{n}{k}$. The upper theoretical bound on embedding efficiency is given as well (see Section 3.4.3.1). Note that the more bits of the cover are used for embedding, the smaller the inverse relative message length. We find, e. g., $\alpha^{-1} = 1$ for LSB embedding where all cover elements are used for embedding. Furthermore, note that schemes with a high embedding efficiency e are less likely to be detected.

Note that all investigated codes achieve better results than the Hamming Codes. However, since the complexity for embedding based on Hamming Codes is low, these codes are often used in practical steganographic schemes (e. g. [96]).

As visualized in Figure 6.3, we find the best results for Matrix Embedding for Large Payloads, Simplex Codes and Augmented Simplex Codes considering a low inverse relative message length α^{-1}. These approaches achieve results close to the bound for a inverse relative message length ≈ 1. Whenever the inverse relative message length gets higher, the results for Simplex Codes and Augmented Simplex Codes, as examples for deterministic matrices, get worse compared to those of the approach Matrix Embedding for Large Payloads.

Note that Matrix Embedding for Large Payloads as well as Simplex Codes are applicable only for a small inverse relative message length α^{-1}. For a inverse relative message length of 1.5 and higher, BCH Codes achieve better results in terms of the embedding efficiency. Thus, Matrix Embedding for Large Payloads and Simplex Codes are only reasonable for $\alpha^{-1} < 1.5$.

Generally, the investigated BCH Codes achieve good results in terms of the embedding efficiency[23]. We can use the algorithm proposed by Schönfeld and Winkler in [86] and described in Appendix E in order to find appropriate code parameters suitable to achieve a high embedding efficiency. Note that it is not possible to design codes based on deterministic matrices for arbitrary values n and l.

The results reported for the Block Minimal Method are not directly comparable to the other approaches at a first impression. Note that the reported results for e are related to $\alpha_{\text{Dry}}^{-1} = \frac{|\text{Dry}|}{|\mathbf{m}|}$ instead of $\alpha^{-1} = \frac{n}{k}$.

[23] Note that the results visualized within Figure 6.3 do not consider wet elements, i. e., elements that should be excluded during embedding. In this case, the results for the embedding efficiency are slightly worse due to an increased average number of embedding changes R_a (see also Figure 6.6).

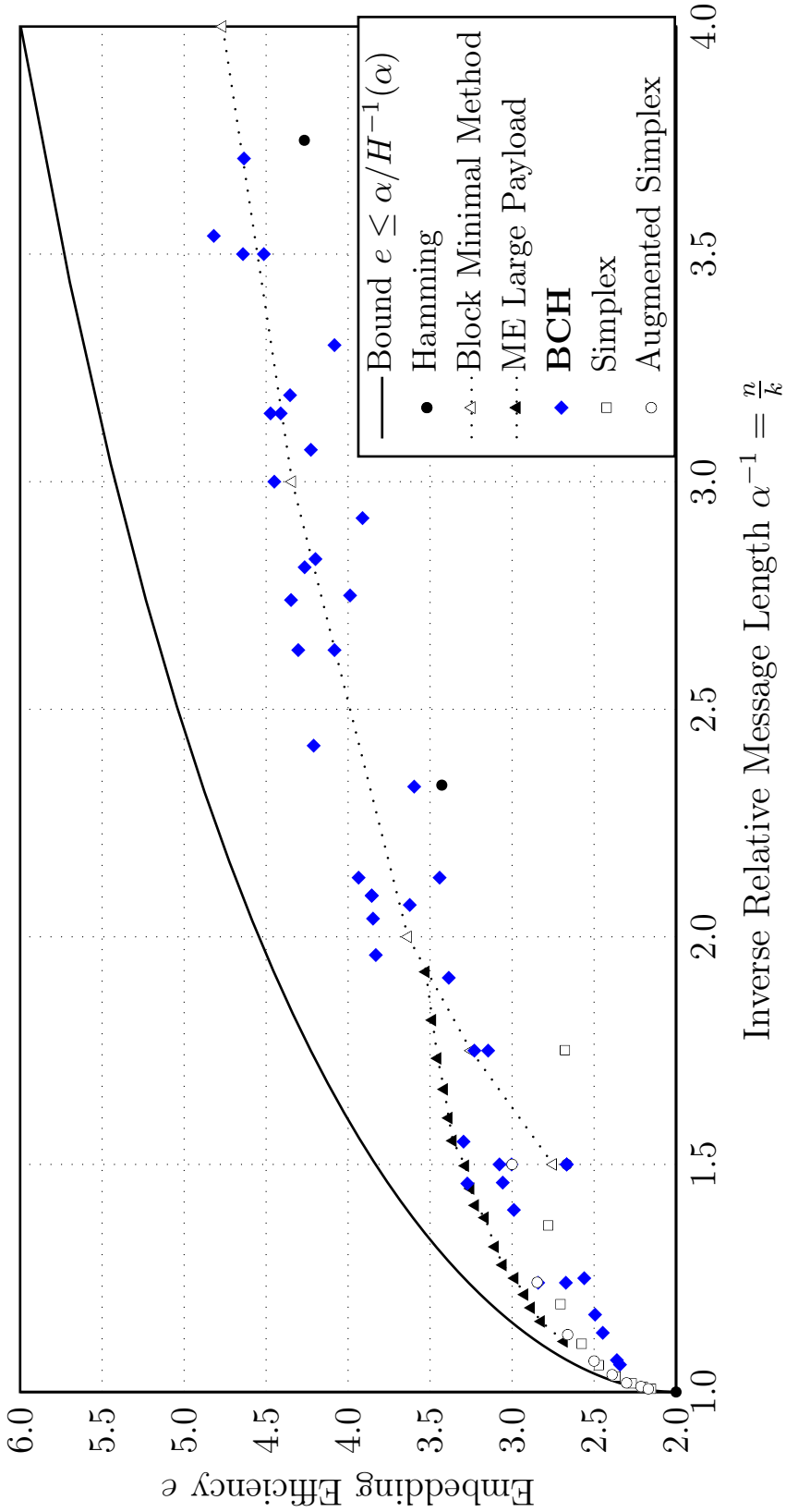

Figure 6.3.: Embedding Efficiency e Dependent on the Inverse Relative Message Length - A Comparison.

However, considering the Block Minimal Method for embedding, the matrix is divided into two sub-matrices, where $\mathbf{H}_{k\times|\mathrm{Dry}|}$ is used for embedding. While $\mathbf{H}_{k\times n}$ is a matrix of dimension $k \times n$, $\mathbf{H}_{k\times|\mathrm{Dry}|}$ is a matrix containing k rows and on average k/α columns [47]. Thus, the dimension of $\mathbf{H}_{k\times|\mathrm{Dry}|}$ is independent of the actual number of wet elements. Based on the parameters presented in Table 6.1, we summarize the related code parameters of $\mathbf{H}_{k\times|\mathrm{Dry}|}$ in Table 6.3.

Table 6.3.: Number of Columns of $\mathbf{H}_{k\times|\mathrm{Dry}|}$ for Different Values k.

$\alpha_{\mathrm{Dry}}^{-1}$	$k = 4$	$k = 10$	$k = 18$
1.5	6	15	27
1.75	7	18	32
2	8	20	36
2.5	10	25	45
3	12	30	54
3.5	14	35	63
4	16	40	72

Based on this fact, we actually can compare the results reported for the Block Minimal Method directly to the other approaches, whenever $|\mathrm{Dry}| = n$. In this case, visualized in Figure 6.3, we find $\alpha^{-1} = \alpha_{\mathrm{Dry}}^{-1}$. Whenever $|\mathrm{Dry}| < n$, i. e., whenever wet elements are considered, the results concerning the embedding efficiency are getting worse similar to those for embedding based on BCH Codes[24].

Generally, we find that the Block Minimal Method is comparable to the approach of embedding based on BCH Codes since both approaches cover a wide range of inverse relative message length α^{-1}. Note that the results considering the embedding efficiency are not as good as the results achievable for BCH Codes. However, it is possible to design codes for arbitrary values n and l. Thus, codes related to the Block Minimal Method are able to cover the range of α^{-1} more densely.

To summarize, it seems reasonable to apply schemes with a higher inverse relative message length α^{-1} whenever the security of the scheme is of interest. Thus, the application of embedding based on BCH Codes as well as based on the Block Minimal Method seems advantageous. However, none of the algorithms investigated so far is able to reach the upper boundary of embedding efficiency.

[24] Contrary to the BCH Codes, where the curve of the embedding efficiency will be on a lower level of e (a downside shift), the curve of the embedding efficiency for the Block Minimal Method will be shifted to the right hand side.

6.2. Evaluation Concerning the Capacity

The capacity of a steganographic scheme is defined as the maximum length of a secret message (in bit) divided by the number of bits required to store the stego object (see Section 2.1.3). For approaches working on a block-based manner, we find the inverse relative message length $\alpha^{-1} = \frac{n}{k}$. Thus, the maximum length of the embeddable message depends on the parameter k and the block length n.

However, for embedding based on a stochastic parity-check matrix, the maximum achievable capacity is $|\mathbf{emb}| \leq k$ (see Table 6.4). Due to the non-negligible probability of a stochastic matrix of not being of rank k, the capacity for approaches based on a stochastic parity-check matrix is reduced. First, these approaches require the sender to store the actual achievable message length requiring a segment of the cover not usable for embedding. Second, a coding loss has to be considered since the message length is reduced within each embedding trial in case of non-solvability.

Table 6.4.: Achievable Message Length for the Different Approaches.

Algorithm	Message Length
Block Minimal Method	$\|\mathbf{emb}\| \leq k$
Matrix Embedding for Large Payloads	$\|\mathbf{emb}\| \leq k$
Hamming Code	$\|\mathbf{emb}\| = k$
BCH Codes	$\|\mathbf{emb}\| = k$
Simplex Codes	$\|\mathbf{emb}\| = k$
Augmented Simplex Codes	$\|\mathbf{emb}\| = k$

The decrease of the embedding capacity due to non-solvability is given in Table 6.5 for the Block Minimal Method according to [45]. Beside $\alpha_{\text{Dry}}^{-1} = \frac{|\text{Dry}|}{N\alpha}$, i. e., the maximal inverse relative message length, Table 6.5 gives the actual reachable inverse relative message length $\alpha_{\text{Dry}}^{\prime -1}$. Furthermore, we give the block length n according to $n = 20\frac{k}{\alpha}$ similar to Table 6.1.

Table 6.5.: Capacity Loss for $N = 10^6, |\text{Dry}| = 50000, k = 18$ According to [45].

n	1200	900	720	600	514	450	400
α_{Dry}	0.3	0.4	0.5	0.6	0.7	0.8	0.9
α_{Dry}^{-1}	3.333	2.5	2	1.667	1.429	1.25	1.111
$\alpha_{\text{Dry}}^{\prime -1}$	3.333	2.5	2	1.692	1.515	1.437	1.433

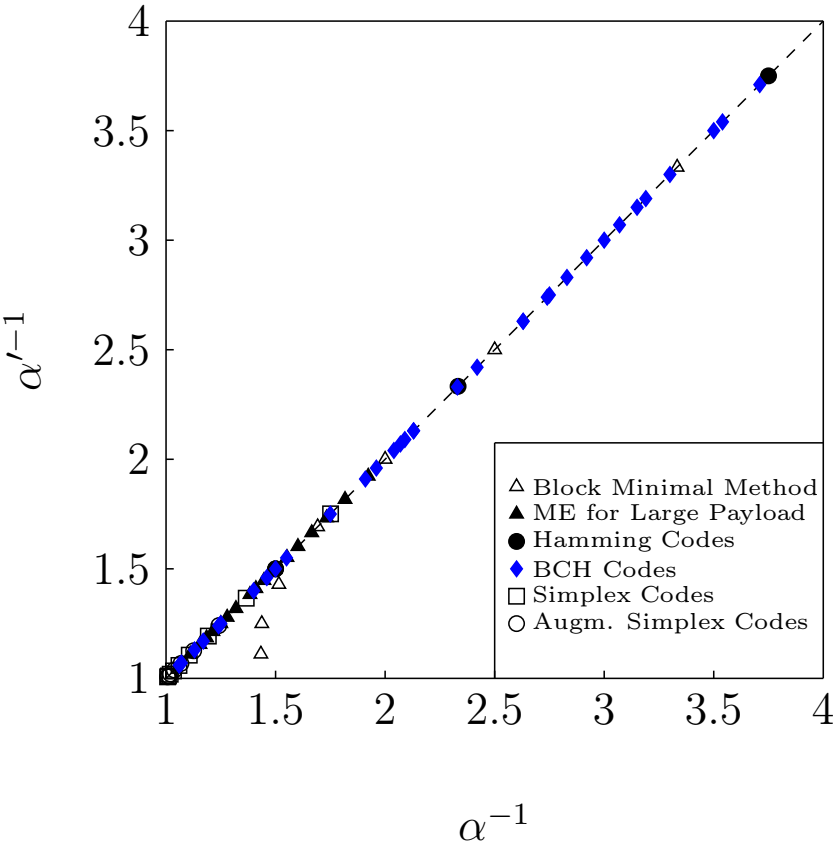

Figure 6.4.: Achievable Inverse Relative Message Length α'^{-1} Dependent on the Maximal Inverse Relative Message Length α^{-1}.

Note that the loss becomes negligible for $\alpha_{\text{Dry}}^{-1} > 1.667$, i. e., in the case where less than 60% of the dry elements are used for embedding the confidential message ($\alpha < 0.6$). For a message length larger than $\approx 70\%$ of the cover, it is not feasible to apply this embedding approach.

In order to visualize the coding loss, we give the achievable inverse relative message length α'^{-1} depending on the maximal inverse relative message length α^{-1} in Figure 6.4. Additionally, Figure 6.4 visualizes how dense the codes cover the range of the relative message length. Note that a high inverse relative message length is linked to a higher security of the scheme.

As it can be seen in Figure 6.4, there is indeed a coding loss for the Block Minimal Method considering an inverse relative message length $\alpha^{-1} < 1.667$. Note that even if Hamming Codes do not improve the embedding efficiency for messages whose inverse relative message length is below 1.5, they cover the range for a high inverse relative message length ($\alpha^{-1} > 5$) more densely. Moreover, as

stated in [52], Simplex Codes can easily reach into the range of $\alpha^{-1} \approx 1$. Note that Augmented Simplex Codes populate the range for α^{-1} more sparsely. Also Matrix Embedding for Large Payloads is applicable only for a low inverse relative message length.

Generally, the Block Minimal Method as well as BCH Codes cover the range of α^{-1}. While the Block Minimal Method is able to cover continuously, the BCH Codes are not able to achieve matrices for arbitrary values of α^{-1}. However, they do not cause coding loss. The property of coding loss, i. e., the success of embedding will be discussed in more detail in the next section.

6.3. Evaluation Concerning the Success Rate

Within this section, we describe the success of embedding within a block of length n as well as within the whole cover considering a general profile, i. e., embedding while excluding conspicuous cover parts (Wet Paper Steganography).

Note that considering a deterministic parity-check matrix and a uniform profile of embedding impact, it is always possible to embed arbitrary messages, i. e., the success rate is 100%. For embedding based on a stochastic parity-check matrix instead, a coding loss has to be accepted since the probability of finding a solution to the system of linear equations is less than 100%. This coding loss results in a reduced capacity as well as in an increased complexity since in case of non-solvability the sender has to start the embedding process again with a reduced parameter k. This is true even for a uniform profile of embedding impact.

Note that considering the exclusion of cover parts (Wet Paper Steganography), also codes based on a deterministic parity-check matrix have to deal with non-solvability. This will be considered in more detail within the next section.

6.3.1. Success of Embedding Considering Single Blocks

Within this section, we describe the success of embedding within a block of length n considering a general profile. In the following, the **so**lvability is referred to as $p_{\mathbf{so}}$. The embedding process is successful, i. e. $p_{\mathbf{so}} = 1$, whenever $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$ has a solution for arbitrary messages **emb**. Thus, $\operatorname{rank}(\mathbf{H}_{k\times n})$ has to be k.

As an example for embedding based on stochastic parity-check matrices, we discuss properties of the Block Minimal Method. Embedding based on BCH Codes should be investigated as an example for syndrome coding with deterministic matrices.

6.3.1.1. Block Minimal Method

The probability of a stochastic binary matrix with equal probability for 0 and 1 being of rank s with $s \leq k$ is given in Equation (3.3) with:

$$P_{k,n}(s) = 2^{s(k+n-s)-kn} \prod_{i=0}^{s-1} \frac{(1-2^{i-k})(1-2^{i-n})}{(1-2^{i-s})}. \tag{6.1}$$

Thus, the probability of a stochastic binary matrix being of full rank and by this the solvability can be determined with:

$$p_{\mathbf{so}} = \prod_{i=0}^{k-1} (1-2^{i-n}). \tag{6.2}$$

As already stated by Fridrich, $p_{\mathbf{so}}$ quickly approaches 1 with decreasing message length $|\mathbf{m}|$ or increasing k for a fixed number of changeable elements and a fixed message length [48]. However, for any $|\text{Dry}| \approx |\mathbf{m}|$, i. e., a low inverse relative message length α^{-1}, the probability of $\text{rank}(\mathbf{H}_{k\times n}) < k$ may become large enough to encounter a failure to embed all k bits in some blocks. Thus, we have to denote a capacity loss [47].

The fact that the number of columns in $\mathbf{H}_{k\times|\text{Dry}|}$ varies from block to block also contributes to failures. Fridrich et al. propose that failures be dealt with by communicating to the receiver which blocks failed.

In Figure 6.5, we depict the solvability $p_{\mathbf{so}}$ depending on the inverse relative message length α^{-1}_{Dry}, i. e., depending on the percentage of dry elements used for embedding. Note that $p_{\mathbf{so}}$ was determined according to Equation (6.2) and based on Table 6.1.

Within this figure, we give the general properties of the Block Minimal Method dependent on k and the inverse relative message length, independently of the actual number of wet cover elements. Note that a high inverse relative message length is related to a higher security of the scheme.

As it can be seen in Figure 6.5, $p_{\mathbf{so}}$ depends on both, the inverse relative message length as well as the parameter k. While for a small parameter k with $k = 4$ the solvability quickly decreases for $\alpha^{-1}_{\text{Dry}} < 3$, the results for larger parameters k are better. However, even for $k = 18$, we find a decrease in $p_{\mathbf{so}}$ for $\alpha^{-1}_{\text{Dry}} < 1.5$.

To summarize, we find that the fewer bits are embedded within a block, the better the solvability. Consequently, the application of the Block Minimal Method is only feasible, whenever the percentage of dry elements used for embedding is low.

The results presented in Figure 6.5 confirm the results presented by Fridrich et al. in [48]. Moreover, we state that these results are true, independently of the actual number of dry elements since the dimension of the sub-matrix $\mathbf{H}_{k\times|\text{Dry}|}$ used for embedding depends only on k and α^{-1}_{Dry} (see Table 6.3).

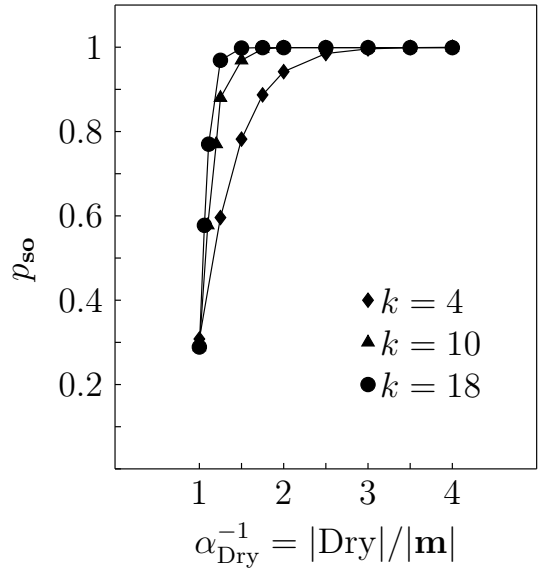

Figure 6.5.: Solvability Dependent on the Inverse Relative Message Length $\alpha_{\mathrm{Dry}}^{-1}$ for the Block Minimal Method.

6.3.1.2. BCH Codes

Based on experimental investigations of different BCH Codes by Schönfeld and Winkler [86, 87], we derived a general bound for the maximum number of arbitrarily distributed wet elements within a block of length n.

We achieve $p_{\mathbf{so}} = 1.0$ for

$$|\mathrm{Wet}| \leq \left\lceil \frac{l}{2} \right\rceil, \tag{6.3}$$

and $p_{\mathbf{so}} \geq 0.9$ for

$$\left\lceil \frac{l}{2} \right\rceil < |\mathrm{Wet}| \leq (l-1). \tag{6.4}$$

Thus, even if there are up to $\lceil \frac{l}{2} \rceil$ arbitrarily distributed wet elements within a block of length n, embedding based on BCH Codes always has a solution for arbitrary messages **emb**. Whenever we accept a little lower probability $p_{\mathbf{so}}$, we can exclude up to $(l-1)$ arbitrary positions within **a**. The results presented in Figure 6.6 according to [86] illustrate these bounds.

The diagrams on the left-hand side illustrate the probability of finding a solution depending on the number of wet elements for the $(17, 9, 2)$ non-primitive BCH Code and also for the $(15, 11, 1)$ Hamming Code and the $(23, 12, 3)$ Golay Code. Again, all results for $p_{\mathbf{so}}$ with $|\mathrm{Wet}| = (0, 1, ..., l)$ were determined experimentally. For given sequences **a**, we evaluated the results for all possible

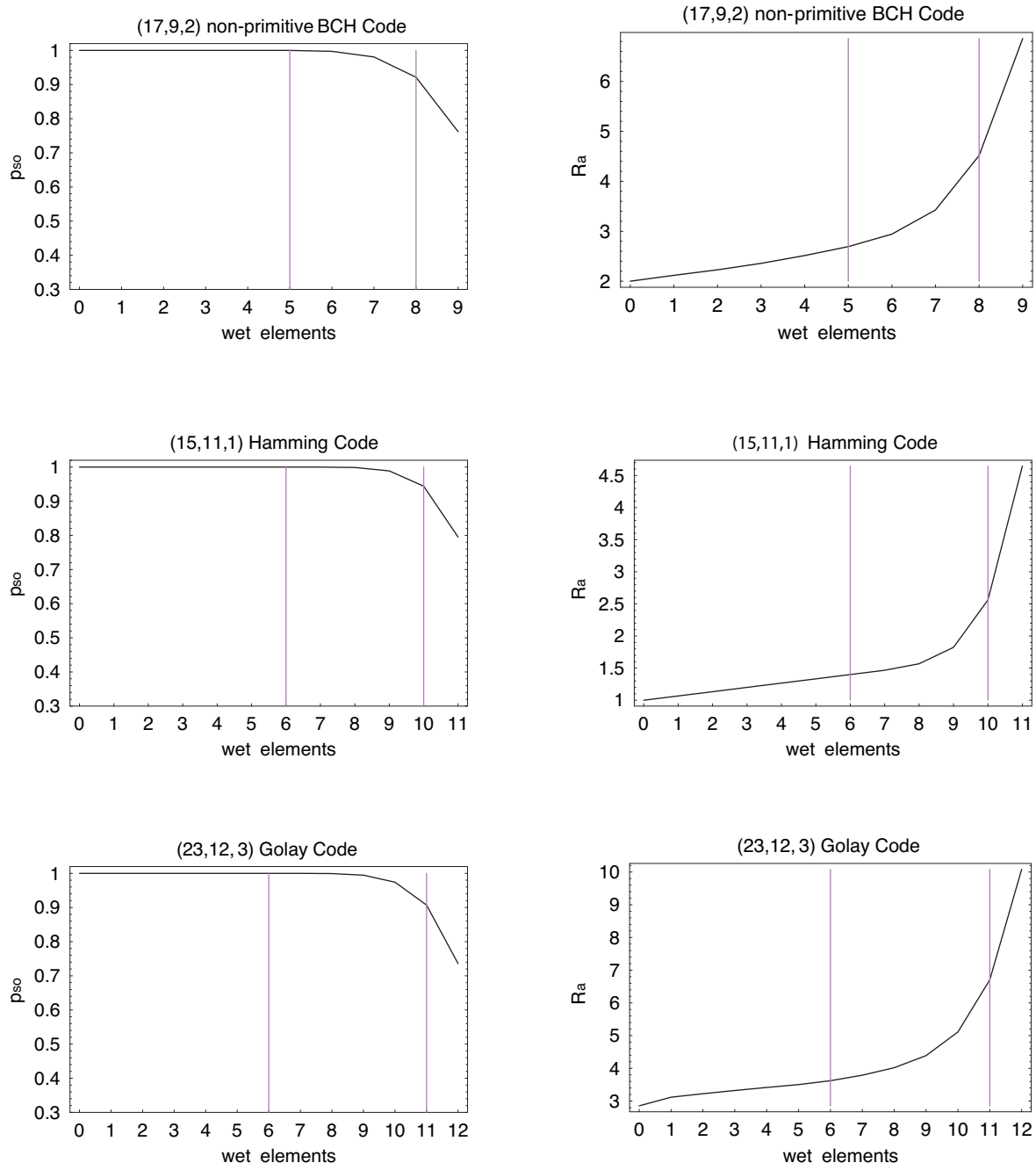

Figure 6.6.: R_a and Solvability Dependent on the Number of Wet Elements for Several BCH Codes [86]. The Bounds According to Equation (6.3) and (6.4) are High-lighted.

sequences **emb**. Within the diagrams, the bounds according to (6.3) and (6.4) are highlighted.

The diagrams on the right-hand side of Figure 6.6 show the average number of embedding changes R_a. All results for R_a were determined experimentally by means of exhaustive search. As expected, the average number of embedding changes R_a for one block increases with a rising number of wet elements.

6.3.1.3. Choosing an Appropriate BCH Code

Whenever choosing a code out of different possible codes, we should have a look at its performance concerning the number of wet elements and the number of elements k that can be embedded within a block of length n. Of course, we cannot maximize both.

If the solvability for arbitrary blocks of length n should be $p_{\mathbf{so}} = 1.0$, we find for BCH Codes, e. g.,

$$p_{\text{Wet}} = \frac{\lceil \frac{l}{2} \rceil}{n}, \tag{6.5}$$

where p_{Wet} describes the percentage of wet elements.

If we choose $p_{\mathbf{so}} \geq 0.9$, we find:

$$p_{\text{Wet}} = \frac{l-1}{n}. \tag{6.6}$$

Considering a $(17, 9, 2)$ non-primitive BCH Code, we find: whenever the solvability for arbitrary blocks of length $n = 17$ should be $p_{\mathbf{so}} = 1.0$, we determine $p_{\text{Wet}} = 0.294$, i. e., 29.4% out of n elements in each block can be excluded arbitrarily while we are able to embed arbitrary sequences **emb**. If we accept $p_{\mathbf{so}} = 0.9$, we can exclude up to 47.1% of the n elements of each block arbitrarily.

Generally, codes with a good performance f_k and with it a small code rate always have a relatively low number of information bits l, resulting in a relatively low maximum number of realizable wet elements $|\text{Wet}|$. However, they provide a low complexity since the code alphabet containing 2^l elements is small. Moreover, they are suited for a low inverse relative message length, since the number of embeddable bits per block k is high.

In practice, we have to find a compromise between l and k, depending on the actual cover and its percentage of randomly distributed wet elements on the one hand and the inverse relative message length on the other hand.

6.3.1.4. Comparing Stochastic and Deterministic Matrices

Within this section, we would like to compare the success of embedding within a block for the two approaches, i. e., a matrix based on deterministic design rules and a stochastic parity-check matrix. Therefore, we compare BCH Codes to the Block Minimal Method.

Considering, e. g., the $(17, 9, 2)$ non-primitive BCH Code. Applying this code for embedding, we are able to exclude up to $\left\lceil \frac{l}{2} \right\rceil = 5$ arbitrarily distributed elements while $p_{\mathbf{so}} = 1$ (Equation (6.3)). Thus, 5/17, i. e., 29.4% of the cover can be wet. Note that we find $k = 8$ for this code and thus, $|\mathbf{m}| = 470584$ for a cover of length $N = 10^6$ containing of $n_B = 58823$ blocks.

For a comparable code based on a stochastic matrix, we want to embed again $|\mathbf{m}| = 470584$ elements within a cover of length $N = 10^6$. We find $n_B = |\mathbf{m}|/k = 58823$ and thus, $k = 8$. The number of dry elements can be determined as 70.6% of the cover elements, i. e. $|\mathrm{Dry}| = 706000$. Thus, we find $\alpha_{\mathrm{Dry}}^{-1} = |\mathrm{Dry}|/|\mathbf{m}| = 1.5$.

Consequently, the Block Minimal Method is applied with a pseudo-random parity-check matrix $\mathbf{H}_{8\times 17}$. However, as mentioned before, the dimension of $\mathbf{H}_{k\times|\mathrm{Dry}|}$ is dependent only on $\alpha_{\mathrm{Dry}}^{-1}$. We find $\mathbf{H}_{k\times|\mathrm{Dry}|}$ as a matrix of dimension 8×12 since 5 out of 17 elements are considered wet. The probability of this matrix being of full rank is $P(8) = 0.939$ (see Equation (3.3)).

In this example, the considered BCH Code is clearly better than a comparable stochastic matrix. We find a difference for the solvability of 0.061 between both approaches.

Another example is the $(63, 39, 4)$ BCH Code, which enables to exclude 20 elements while embedding $k = 24$ elements into each block of length 63. Thus, we find that 31.746% of the cover can be wet. For a cover of length $N = 10^6$, we find $n_B = 15873$ and thus, $|\mathbf{m}| = 380952$.

Embedding $|\mathbf{m}| = 380952$ elements within the same cover based on a stochastic parity-check matrix, we find $|\mathrm{Dry}| = 682540$ and thus, $\alpha_{\mathrm{Dry}}^{-1} = |\mathrm{Dry}|/|\mathbf{m}| = 1.792$. Consequently, embedding will be based on a pseudo-random parity-check matrix $\mathbf{H}$ of dimension 24×63. Considering $\alpha_{\mathrm{Dry}}^{-1}$, we find $\mathbf{H}_{k\times|\mathrm{Dry}|}$ as a sub-matrix of dimension 24×43 since 20 out of 63 elements are considered wet. The probability of this matrix being of full rank is $P(24) = 0.999998$.

For this example, we find that the Block Minimal Method is almost as powerful as a comparable BCH Code. This is true, whenever we have either a high $\alpha_{\mathrm{Dry}}^{-1}$ or a large k. In this case, the solvability of the Block Minimal Method is not that crucial. However, there is still a small possibility of failure, resulting in a reduction of the message length as well as in an increased complexity due to the additional embedding trial. In this case, less than k bits can be embedded within a block of length n. This is not an issue at all for BCH Codes.

Of course, these considerations behold only the probability of finding a solution for an individual block of length n, i. e., the performance of a code.

6.3.2. Success of Embedding Considering the Whole Cover

So far, the considerations only take into account the (n, l) code itself, i. e., we studied the performance of individual code blocks. Within this section, we generalize the analysis to an entire cover consisting of $\left\lfloor \frac{N}{n} \right\rfloor$ blocks.

As an example for embedding based on stochastic parity-check matrices, we discuss properties of the Block Minimal Method. Embedding based on BCH Codes should be investigated as an example for syndrome coding with deterministic matrices.

In order to calculate the probability of being able to embed a message within a cover, we also have to take into account the percentage of wet elements within the entire cover. Therefore, we have to combine the probability of finding a solution within an individual block and the number of wet elements within this block. Of course, this consideration influences the choice of an appropriate code and thereby the possible message length.

Depending on the percentage of wet elements, denoted as p_{Wet}, the actual number of wet elements $|\text{Wet}|$ within an arbitrary block of length n follows the binomial distribution $P(\text{Wet})$:

$$P(x = \text{Wet}) = \binom{n}{x} p_{\text{Wet}}^{x} (1 - p_{\text{Wet}})^{n-x} \qquad (\text{Wet} = 1, 2, ..., n). \tag{6.7}$$

Consequently, the success of embedding depends not only on the solvability $p_{\mathbf{so}}$ within each block but also on the distribution of wet elements. We find the probability for **su**ccess of embedding $p_{\mathbf{su}}$ within an arbitrary block of length n as a combination of $p_{\mathbf{so}}$ and $P(|\text{Wet}|)$:

$$p_{\mathbf{su}} = \sum_{\text{Wet}=1}^{l} P(\text{Wet})\, p_{\mathbf{so}}. \tag{6.8}$$

6.3.2.1. Block Minimal Method

In order to determine the success of embedding within the entire cover, we first have to determine the probability for success of embedding $p_{\mathbf{su}}$ within an arbitrary block of length n.

For the first example given in the previous section, we find $N = 10^6$ and thus, $n_B = 58823$ blocks of length 17 (see Section B). Furthermore, we find $\mathbf{H}_{k\times|\text{Dry}|}$ as a matrix of dimension 8×12. The probability of this matrix being of full rank is $P(8) = 0.939$ (see Equation (3.3)). Note that it is possible to exclude up to 5 arbitrarily distributed elements within each block.

Considering $p_{\text{Wet}} = 0.294$, we find $\sum_{\text{Wet}=1}^{5} P(|\text{Wet}|) = 0.615$, i. e., only 61.5% of the blocks contain 5 or less wet elements. For the remaining blocks, we have to exclude more elements.

Whenever we exclude 6 elements, we find $\mathbf{H}_{k\times|\text{Dry}|}$ as matrix of dimension 8×11. The parameters of the matrix, the corresponding $p_{\mathbf{so}}$ (see Equation (3.3)) and $P(|\text{Wet}|)$ (see Equation (6.7)) are given in Table 6.6.

Note that $\sum_{\text{Wet}=1}^{9} P(|\text{Wet}|) = 0.986$, i. e., only 98.6% of the blocks contain 9 or less wet elements. The remaining blocks cannot be used for embedding at all.

Table 6.6.: Dimension of $\mathbf{H}_{k\times|\mathrm{Dry}|}$, $p_{\mathbf{so}}$ and $P(|\mathrm{Wet}|)$ for the Block Minimal Method and $\mathbf{H}_{8\times 17}$.

$\lvert\mathrm{Wet}\rvert$	Dimension of $\mathbf{H}_{k\times\lvert\mathrm{Dry}\rvert}$	$p_{\mathbf{so}}$	$P(\lvert\mathrm{Wet}\rvert)$
6	8×11	0.881	0.174
7	8×10	0.771	0.114
8	8×9	0.579	0.0591
9	8×8	0.290	0.025

Based on the results given in Table 6.6, we find $s_{\mathbf{su}}$ as

$$\begin{aligned} p_{\mathbf{su}} &= \sum_{\mathrm{Wet}=1}^{l} P(\mathrm{Wet})\, p_{\mathbf{so}} \\ &= 0.615\cdot 0.939 + 0.174\cdot 0.881 + 0.114\cdot 0.771 + 0.059\cdot 0.579 + 0.025\cdot 0.290 \\ &= 0.860. \end{aligned}$$

Thus, for the solvability within an arbitrary block of length n containing wet elements distributed according to $P(\mathrm{Wet})$, we find $p_{\mathbf{su}} = 0.860$.

In order to determine the success of embedding within the whole cover, we have to calculate the product of $p_{\mathbf{su}}$, effective for one block, over all blocks. Considering $n_B = 58823$ blocks within the whole cover, the success of embedding within the whole cover will be $p_{\mathbf{su}} = 0.860^{58823} = 0.00$, i. e., the success of embedding the complete message of length $|\mathbf{m}| = 470584$ will be less than 1%.

For the second example, considering $N = 10^6$, $n_B = 15873$, $n = 63$ and $|\mathbf{m}| = 380952$, we find $\mathbf{H}_{k\times|\mathrm{Dry}|}$ as a matrix of dimension 24×43 since 20 out of 63 elements are considered wet. The probability of matrix $\mathbf{H}_{k\times|\mathrm{Dry}|}$ being of full rank is $P(24) = 0.999998$. In this example, 31.746% of the cover can be wet, i. e., $p_{\mathrm{Wet}} = 0.317$.

For this code, we find $\sum_{\mathrm{Wet}=1}^{20} P(|\mathrm{Wet}|) = 0.563$, i. e., only 56% of the blocks contain 20 or less wet elements. For the remaining blocks, we have to exclude more elements.

Whenever we exclude 21 elements, we find $\mathbf{H}_{k\times|\mathrm{Dry}|}$ as matrix of dimension 24×42. The parameters of the matrix, the corresponding $p_{\mathbf{so}}$ and $P(|\mathrm{Wet}|)$ are given in Table 6.7.

Note that $\sum_{\mathrm{Wet}=1}^{39} P(|\mathrm{Wet}|) = 0.9999998$. The remaining blocks contain 40 wet elements and cannot be used for embedding. Based on the results given in Table 6.7, we find $p_{\mathbf{su}}$ according to

Table 6.7.: Dimension of $\mathbf{H}_{k\times|\mathrm{Dry}|}$, $p_{\mathbf{so}}$ and $P(|\mathrm{Wet}|)$ for the Block Minimal Method and $\mathbf{H}_{24\times 63}$.

$\lvert\mathrm{Wet}\rvert$	Dimension of $\mathbf{H}_{k\times\lvert\mathrm{Dry}\rvert}$	$p_{\mathbf{so}}$	$P(\lvert\mathrm{Wet}\rvert)$
21	24×42	0.99999	$10.212 \cdot 10^{-2}$
22	24×41	0.99999	$9.049 \cdot 10^{-2}$
23	24×40	0.9999	$7.487 \cdot 10^{-2}$
24	24×39	0.9999	$5.791 \cdot 10^{-2}$
25	24×38	0.9999	$4.193 \cdot 10^{-2}$
26	24×37	0.999	$2.844 \cdot 10^{-2}$
27	24×36	0.999	$1.809 \cdot 10^{-2}$
28	24×35	0.999	$1.079 \cdot 10^{-2}$
29	24×34	0.999	$0.605 \cdot 10^{-2}$
30	24×33	0.998	$0.318 \cdot 10^{-2}$
31	24×32	0.996	$0.157 \cdot 10^{-2}$
32	24×31	0.992	$0.073 \cdot 10^{-2}$
33	24×30	0.984	$0.032 \cdot 10^{-2}$
34	24×29	0.969	$0.013 \cdot 10^{-2}$
35	24×28	0.939	$5.009 \cdot 10^{-5}$
36	24×27	0.880	$1.808 \cdot 10^{-5}$
37	24×26	0.770	$6.124 \cdot 10^{-6}$
38	24×25	0.578	$1.945 \cdot 10^{-6}$
39	24×24	0.289	$5.786 \cdot 10^{-7}$

$$
\begin{aligned}
p_{\mathbf{su}} &= \sum_{\text{Wet}=1}^{l} P(\text{Wet})\, p_{\mathbf{so}} \\
&= 0.563 \cdot 0.999998 + 10.212 \cdot 10^{-2} \cdot 0.99999 + 9.049 \cdot 10^{-2} \cdot 0.99999 \\
&+ 7.487 \cdot 10^{-2} \cdot 0.9999 + 5.791 \cdot 10^{-2} \cdot 0.9999 + 4.193 \cdot 10^{-2} \cdot 0.9999 \\
&+ 2.844 \cdot 10^{-2} \cdot 0.999 + 1.809 \cdot 10^{-2} \cdot 0.999 + 1.0796 \cdot 10^{-2} \cdot 0.999 \\
&+ 0.605 \cdot 10^{-2} \cdot 0.999 + 0.318 \cdot 10^{-2} \cdot 0.998 + 0.157 \cdot 10^{-2} \cdot 0.996 \\
&+ 0.073 \cdot 10^{-2} \cdot 0.992 + 0.032 \cdot 10^{-2} \cdot 0.984 + 0.013 \cdot 10^{-2} \cdot 0.969 \\
&+ 5.009 \cdot 10^{-5} \cdot 0.939 + 1.808 \cdot 10^{-5} \cdot 0.880 + 6.124 \cdot 10^{-6} \cdot 0.770 \\
&+ 1.945 \cdot 10^{-6} \cdot 0.578 + 5.786 \cdot 10^{-7} \cdot 0.289 \\
&= 0.9995834.
\end{aligned}
$$

In order to determine the success of embedding within the whole cover, we have to calculate the product of $p_{\mathbf{su}}$, effective for one block, over all blocks. Considering $n_B = 15873$ blocks within the whole cover, the success of embedding within the whole cover will be $p_{\mathbf{su}} = 0.9995834^{15873} = 0.001$, i. e., the success of embedding the complete message will be less than 1%.

6.3.2.2. BCH Codes

For the examples given in Section 6.3.1.4, e. g. the $(17, 9, 2)$ BCH Code, we find $p_{\mathbf{so}} = 1$, considering $|\text{Wet}| \leq 5$ (Equation (6.3)).

However, considering the distribution of wet elements related to $p_{\text{Wet}} = 0.294$, we find $\sum_{\text{Wet}=1}^{5} P(|\text{Wet}|) = 0.615$ (Equation (6.7)). Thus, only 61.5% of all blocks contain 5 or less wet elements. For the remaining blocks, we are not able to achieve $p_{\mathbf{so}} = 1$ since we have to exclude additional elements. Note that the distribution of $P(\text{Wet})$ is similar to those given in Table 6.6. Within Figure 6.7, we depict $p_{\mathbf{so}}$ depending on the number of wet elements within one block of length n. These results are determined experimentally by Schönfeld and Winkler in [86].

Again, we find $\sum_{\text{Wet}=1}^{9} P(|\text{Wet}|) = 0.986$, i. e., only 98.6% of the blocks contain 9 or less wet elements. The remaining blocks cannot be used for embedding.

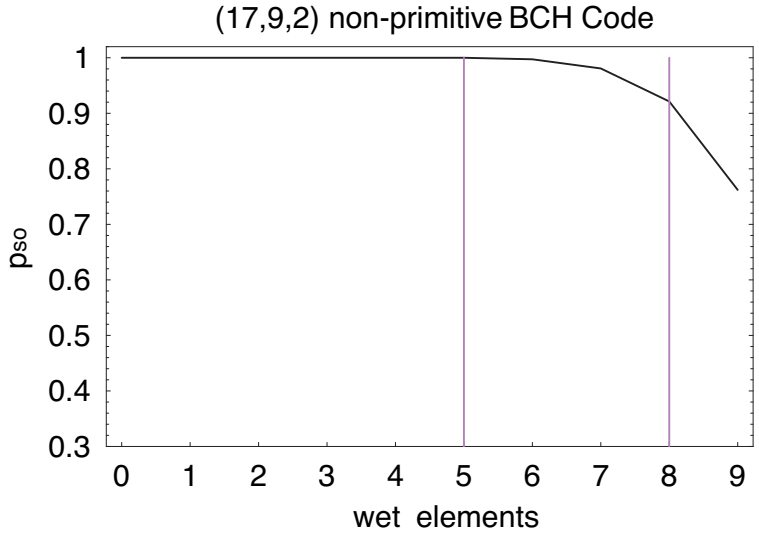

Figure 6.7.: Solvability Dependent on the Number of Wet Elements for the $(17, 9, 2)$ Non-Primitive BCH Code According to [86].

For this example, we find $s_{\mathbf{su}}$ as

$$\begin{aligned} p_{\mathbf{su}} &= \sum_{\text{Wet}=1}^{l} P(\text{Wet})\, p_{\mathbf{so}} \\ &= 0.615 \cdot 1 + 0.174 \cdot 0.999 + 0.114 \cdot 0.990 + 0.059 \cdot 0.910 + 0.025 \cdot 0.730 \\ &= 0.974. \end{aligned}$$

Thus, for the solvability within an arbitrary block of length n containing wet elements distributed according to $P(\text{Wet})$, we find $p_{\mathbf{su}} = 0.974$.

In order to determine the success of embedding within the whole cover, we have to calculate the product of $p_{\mathbf{su}}$, effective for one block, over all blocks. Considering $n_B = 58823$ blocks within the whole cover, the success of embedding within the whole cover will be $p_{\mathbf{su}} = 0.974^{58823} = 0.00$, i. e., the success of embedding the complete message of length $|\mathbf{m}| = 470584$ will be less than 1%. This result is of course not acceptable, the $(17, 9, 2)$ non-primitive BCH Code is not suited in this case.

The results confirm that neither the approach based on the stochastic matrix as applied in the Block Minimal Method nor embedding based on the deterministic matrix applied for BCH Codes are able to embed the whole message. However, the BCH Code is able to achieve a higher solvability per block $p_{\mathbf{so}}$.

6.3.2.3. Choosing an Appropriate BCH Code

In order to find an appropriate code suited for steganographic systems, we have to find code parameters for which $p_{\mathbf{su}} = 1.0$. Note that in theory, $p_{\mathbf{su}} = 1.0$ is only true if there are only $\lceil \frac{l}{2} \rceil$ wet elements within the entire cover. Otherwise, there is a small chance that additional wet elements occur in one block.

Therefore, the goal is to determine a lower bound for the code parameter l for a given block length n and a given percentage of wet elements p_{Wet} within the cover. Whenever we are not able to find an appropriate code according to l, we have to choose a larger block length n.

We illustrate this process for deterministic matrices based on BCH Codes [86]. As realized in the previous section, the distribution of wet elements within the cover blocks follow the binomial distribution $P(\text{Wet})$.

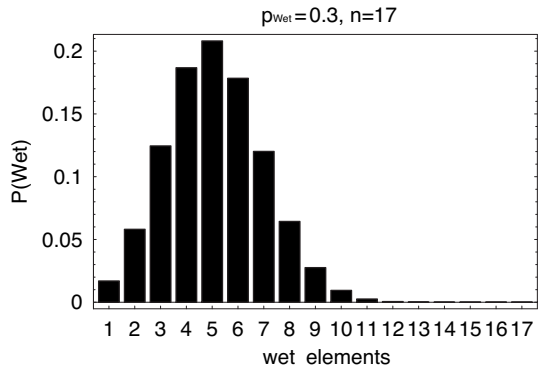

Figure 6.8.: Distribution of Wet Elements for $n = 17$ and $p_{\text{Wet}} = 0.3$ [86].

In Figure 6.8, we visualized the distribution of wet elements for $n = 17$ and $p_{\text{Wet}} = 0.3$. As mentioned in the previous section, the $(17, 9, 2)$ non-primitive BCH Code is not suited for embedding since we are able to exclude only up to 5 elements considering this code (Equation (6.3)).

A solution will be the application of codes with a larger codeword length n. In Figure 6.9 we find for $p = 0.3$ and $n = 31$ that l has to be at least 21 for $p_{\mathbf{so}} = 0.9$. Thus, it is possible to choose, e. g., the $(31, 26, 1)$ or the $(31, 21, 2)$ primitive BCH Code. For $n = 63$, l has to be at least 35, which allows us to use more powerful codes like the $(63, 39, 4)$ primitive BCH Code [86].

However, even for these codes the probability of failure is not zero. The $(31, 26, 1)$ BCH Code, e. g., is able to exclude only 13 arbitrarily distributed elements within a block of length 31 (Equation (6.3)). As Figure 6.9 shows, there are several blocks containing a higher number of wet elements and thus, a reduced $p_{\mathbf{so}}$.

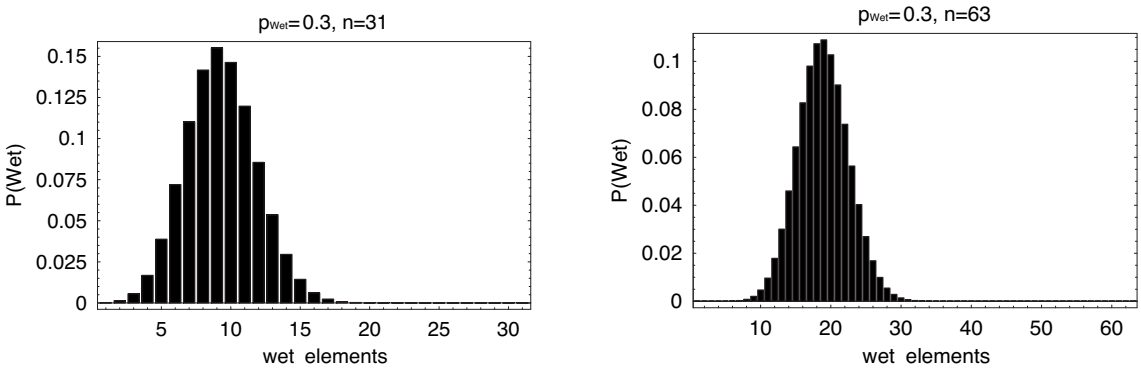

Figure 6.9.: Distribution of Wet Elements for $n = 31$ and $n = 63$ and $p_{\text{Wet}} = 0.3$ [86].

A solution to this problem would be the reduction of k. Whenever the embedding process is not able to embed k bits within a block of length n, we have to reduce k and thereby the dimension of $\mathbf{H}_{k\times n}$ and try again. This approach is similar to the one applied for the Block Minimal Method. However, the solvability per block $p_{\mathbf{so}}$ is higher for BCH Codes resulting in less cases of failure. Consequently, it is possible to embed more bits on average with a lower complexity.

Generally, we find that whenever the percentage of wet elements p_{Wet} is high, codes with a higher codimension l are more suited, i. e., the number of embeddable message bits is reduced. Thus, the smaller the given percentage p_{Wet}, the more can be embedded within the cover, resulting in a low inverse relative message length. Nevertheless, we also have to keep in mind the embedding efficiency, whereas our goal has to be to find an appropriate tradeoff.

6.4. Evaluation Concerning the Embedding Complexity

Within this section, we investigate the complexity of embedding for the algorithms presented and discussed in this work. Complexity is considered in terms of time and memory complexity.

6.4.1. Time Complexity

Note that a requirement for a steganographic algorithm is the possibility to embed in an acceptable time. Which time is acceptable depends on the application. While it seems relatively uncritical for offline approaches, steganography in live-time environments is more crucial.

6.4.1.1. Block Minimal Method

This block-based approach determines the coset leader in order to maximize the embedding efficiency (see Section 4.1). To avoid the search within $2^{|\mathrm{Dry}|}$ sequences, Fridrich et al. propose a step-wise calculation of the cosets.

The most time-consuming part of the embedding algorithm is the computation of the sets $\mathcal{U}_i$. Note that it is necessary to generate $\mathcal{U}_1 + \mathcal{U}_i$ for $i = 1, \ldots, \lceil R/2 \rceil$ in the worst case, where the cardinality of $\mathcal{U}_i$ increases exponentially [36]. According to Fridrich et al., the computational complexity for one embedding trial is bounded by $O(k\alpha_{\mathrm{Dry}}^{-1} \times R/2 \times \binom{k\alpha_{\mathrm{Dry}}^{-1}}{R/2})$, considering $\alpha_{\mathrm{Dry}}^{-1} = \frac{|\mathrm{Dry}|}{|\mathbf{m}|}$. This term increases exponentially with increasing k [36]. Thus, Fridrich et al. find $k \approx 18$ realistic for embedding.

Fridrich et al. investigated the embedding time in seconds for $N = 10^6$ and 50000 changeable elements for different parameters $k = 16, \ldots, 20$ [45]. They found that with increasing message length, the embedding time increases too. Furthermore, the embedding time increases with increasing k [47]. Fridrich et al. found for $k = 17$, e. g., an embedding time for finding the coset leader in the range of 0.73 seconds up to 2.24 seconds, dependent on the relative message length ($\alpha = 0.1, \ldots, 0.5$). For $k = 20$, they found an embedding time in the range of 15.74 seconds up to 18.68 seconds. For these investigations, a PC equipped with a 3.4MHz Intel Pentium IV processor was used. Note that for stochastic matrices, we have to take into account several trials for embedding, whenever the matrix has no full rank. This will result in an increased complexity.

6.4.1.2. Matrix Embedding for Large Payloads

Fridrich et al. investigated the embedding time for several parameters l with $l = 10, 12, 14$ for embedding into a 1280×1024 cover image divided in blocks of length $n = 100$ [52]. They found for $l = 10$ an embedding time of 0.82 seconds, 2.42 seconds for $l = 12$ and 8.65 seconds for $l = 14$.

Thus, the complexity of embedding is $O(n2^{n(1-\alpha)})$ for one embedding trial. Again, we have to consider several trials for embedding since this approach is based on stochastic matrices.

6.4.1.3. BCH Code

As described in Section 5.2, we investigated several approaches for embedding based on BCH Codes. Table 6.8 gives the number of XOR operations needed to embed the message part **emb** of length k within n bits according to Schönfeld and Winkler [87].

The classic approach is based on evaluating 2^n sequences in order to determine the sequence that minimizes the introduced distortion. Thus, it is the most

Table 6.8.: Complexity of Embedding for Several Approaches of Embedding Based on BCH Codes According to [87].

Approach	Complexity	Example: $(15, 7, 2)$
Classic Approach	$O(((kn)\, n + k)\, 2^n)$	59244544
$\mathbf{H}_{k\times n} = [\mathbf{R}\mathbf{I}_k]$	$O(((kl)\, l + k)\, 2^l)$	51200
$\mathcal{C}(\mathbf{0})$	$O(n2^l)$	1920

complex approach. Note that applying a systematic matrix can considerably reduce the complexity.

However, the complexity of embedding can further be reduced by syndrome coding based on $\mathbf{H}_{k\times n}$ combined with Look-up Tables. Embedding based on coset $\mathcal{C}(\mathbf{0})$ reduces time complexity dramatically compared to the classic approach based on exhaustive search.

After determining a coset member $\mathbf{f_m}$, the exhaustive search can be reduced to coset $\mathcal{C}(\mathbf{0})$. Thus, instead of 2^n only 2^l sequences have to be considered. The time needed for finding a coset member $\mathbf{f_m}$ is negligible. Note that it is still possible to determine the coset leader.

Note that the time complexity for Fast BCH is almost negligible and constant for any n [101]. The method has to access the Look-up Table 3 times and makes simple mathematical operations. Consequently, it is easy to extend this method to large n, while embedding based on coset $\mathcal{C}(\mathbf{0})$ is still too complex to be used in real-time applications when n is large.

However, the approach Fast BCH is not applicable for codes with $f_k > 2$. Thus, the embedding efficiency is rather low as visualized in Figure 6.2 (see results for 2 $m_i(x)$). We find for $f_k = 2$ an embedding efficiency worse that those for the Block Minimal Method.

6.4.1.4. Simplex Codes

Fridrich et al. determine the complexity of embedding in terms of the code length with $O(k2^k)$ since evaluating the Hadamard Transform takes $O(k2^k)$ operations. Note that this is better than a direct implementation with $O(2^{2k})$ [52, 51].

6.4.1.5. Comparing the Algorithms

While there are fast solutions for calculating the stego sequence that minimizes the introduced distortion for Hamming Codes, Simplex Codes, Augmented Simplex Codes and BCH Codes with $f_k = 2$, no such strategy is known for approaches based on stochastic matrices or BCH Codes with $f_k > 2$. Thus, this is a clear advantage of codes based on deterministic parity-check matrices. Furthermore, we have to consider additional embedding trials for embedding based on stochastic

matrices due to non-solvability. This also results in a higher embedding complexity.

Note that Matrix Embedding for Large Payloads is less complex than the Block Minimal Method. Due to the small co-dimension l with $l = 10, 14$, the exhaustive search for the coset leader has to take into account only a small set of codewords. However, this class of codes is limited to a low inverse relative message length.

For the BCH Codes with $f_k > 2$, an approach for embedding with lower complexity is based on a reduced coset $\mathcal{C}(\mathbf{0})$. By means of this approach, it is possible to embed with a reduced complexity.

6.4.2. Memory Complexity

Within this Section, we compare the approaches concerning their memory requirements. Note that the memory is limited by the physical properties of the computer used for determining the stego sequence.

6.4.2.1. Block Minimal Method

As already mentioned, the most time-consuming part of the embedding algorithm is the computation of the sets $\mathcal{U}_i$. Note that it is necessary to generate $\mathcal{U}_1 + \mathcal{U}_i$ for $i = 1, \ldots, \lceil R/2 \rceil$ in the worst case, where the cardinality of $\mathcal{U}_i$ increases exponentially [36]. According to Fridrich, we find $|\mathcal{U}_i| \leq (\frac{k\alpha^{-1}}{i})$ on average, the total memory requirement is $O(R/2 \times (\binom{k\alpha^{-1}}{R/2}))$ for this approach.

6.4.2.2. Matrix Embedding for Large Payloads

For embedding based on this approach, it is necessary to keep in memory all 2^l codewords which requires $n2^l$ bits. Note that finding the coset leader requires $O(n2^l)$ operations. Consequently, the code dimension l should be small, i. e., $l \leq 14$. For these codes with low inverse relative message length $\alpha^{-1} = \frac{n}{k} \approx 1$, there is a small number of codewords. As a result, coset leaders can be found efficiently using Look-up Tables.

6.4.2.3. BCH Codes

The proposed strategy for embedding based on coset $\mathcal{C}(\mathbf{0})$ dramatically reduces the storage requirements. In this case, only 2^l sequences have to be stored, which requires $n2^l$ bits. However, when dealing with large code parameters, especially large parameters l, this storage space might be unacceptable. Therefore, approaches further reducing the cardinality of $\mathcal{C}(\mathbf{0})$ were investigated (see Section 5.2.1.5).

Thus, the storage space can be reduced dramatically, i. e., for the $(31, 21, 2)$ BCH Code, e. g., only 0.0002% (2^{12} instead of 2^{31} sequences) of the original

search area has to be considered in our approach. However, this result does not affect e.

Note that for the approach Fast BCH Codes the storage complexity is linear while that of the other methods is exponential with n. The Look-up Table size for the quadratic polynomial is $n \times 1$ and the Look-up Table size for cubic polynomial is $n \times 3$. However, this approach is limited to codes with $f_k = 2$.

6.4.2.4. Comparing the Algorithms

For the approaches based on Hamming Codes and Simplex Codes, no memory is required. However, when applying Simplex Codes with the Fast Hadamard Transform, matrices have to be stored in order to speed up the calculation.

Note that the matrices needed to be stored for the Fast Hadamard Transform are sparse with only two non-zero elements in each row and column. This drastically reduced the memory requirements. Thus, we need only a space of $(O(k2^k))$ for storing all matrices.

The approaches based on Matrix Embedding for Large Payloads and based on BCH Codes both require a storage of $n2^l$ bits. We found for Matrix Embedding for Large Payloads $n = 100$ and $l = 14$ and thus, 1638400 bits of storage space required.

For the $(31, 21, 2)$ BCH Code, we find a storage requirement of 65011712 bits. However, when applying BCH Codes based on a reduced coset, the number of sequences that have to be stored can be reduced to 3628 (see Section 5.2.1.5). Thus, the total storage requirement for this approach is 112468 bits, which is considerably less than for the approach Matrix Embedding for Large Payloads.

6.5. Summary

The evaluation of the algorithms for syndrome coding based on matrices with a small code dimension was carried out according to the parameters security, capacity, success of embedding and complexity.

Whenever choosing a syndrome coding based embedding scheme for a steganographic algorithm, we have to address the goal of the scheme in a first step. First of all, we can achieve either a high security by means of a high inverse relative message length $\alpha^{-1} = n/k$ or a high capacity (low inverse relative message length). In a second step, we can evaluate appropriate algorithms concerning the success of embedding and the complexity of embedding. The results are summarized in Table 6.9 (see also Figure 6.3). Note that the security of the algorithms is considered in terms of the embedding efficiency e

- If the goal is to achieve a *high capacity* and by this a low inverse relative message length α^{-1}, the best results are achieved for Matrix Embedding

Table 6.9.: Evaluation of Embedding Algorithms Based on a Small Code Word Length - A Comparison.

	security	capacity	success of embedding	complexity
Block Minimal Method	+	−	−	−
ME for Large Payload	−	+	−	−
Hamming Codes	−	−	+	+
BCH Codes	+	−	+	−
Simplex Codes	−	+	+	+

for Large Payloads, Simplex Codes and Augmented Simplex Codes.

Whenever the inverse relative message length gets higher, the results of e for Simplex Codes and Augmented Simplex Codes, as examples for deterministic matrices, get worse compared to those of the approach Matrix Embedding for Large Payloads.

- Even if the *security of these schemes is rather low*, they enable to embed in selected parts of the cover without the need of sharing the selection rule with the receiver, in contrast to LSB embedding.

  Note that the positive impact of syndrome coding on security is bigger for a lower inverse relative message lengths since a high inverse relative message length is less detectable anyway [36]. However, considering a low inverse relative message length, we find $|\mathbf{m}| \approx N$. Consequently, these schemes achieve an embedding efficiency of $e \approx 2$. Thus, the improvements compared to LSB embedding are only marginal. This is visualized by the upper boundary of embedding efficiency. Moreover, we find that - according to the square root law of capacity [66] - embedding a low inverse relative message length is rather insecure. Thus, for a practical scheme a lower capacity is preferable.

- Note that for Simplex Codes and Augmented Simplex Codes, the sender is able to calculate the solution while for Matrix Embedding for Large Payloads he has to apply exhaustive search resulting in higher complexity. Moreover, a major advantage for embedding based on codes described by means of a deterministic parity-check matrix is a high solvability even in a scenario with wet elements. A high success of embedding is indeed important considering applications with a fixed message length or applications with real-time requirements. In this case, several embedding trials are unacceptable.

- According to the square root law of capacity [66], the application of a higher inverse relative message length α^{-1} is preferable whenever the *security* of the embedding schemes is of prior interest.

- As algorithms covering a wide range of α, the BCH Codes as well as the Block Minimal Method achieve good results in terms of the embedding efficiency. We found both approaches to be comparable. Nevertheless, since it is not possible to determine BCH Codes for arbitrary code parameters, the Block Minimal Method covers the range of α^{-1} more densely.

- However, as the investigations confirmed, matrices built according to stochastic processes have to deal with a coding loss. Contrary to the BCH Codes built according to deterministic rules, they are not able to embed k bits within each block. This is true due to the non-zero probability of matrix $\mathbf{H}_{k\times n}$ of being of rank $< k$.

  This non-solvability results in a reduced capacity as well as in an increased complexity. In case of failure, the sender has to start the embedding process again with a reduced parameter k. Due to this fact, we find that the Block Minimal Method is not feasible for embedding whenever the relative message length α_{Dry} is higher than 70%.

  Considering a uniform profile, we find that deterministic matrices always provide a solvability $p_{\mathbf{so}} = 1$. However, considering a general profile, i. e., a scenario where wet elements are excluded during the embedding process, we find for both approaches cases of failure. This is true, whenever blocks contain a higher number of wet elements than the code can process. However, since we find a better solvability $p_{\mathbf{so}}$ within a block for deterministic matrices, the success of embedding within the whole cover is also better. Moreover, we find a lower complexity for embedding due to the lower percentage of failure.

- Note that considering the security in terms of the embedding efficiency of the algorithms, we find that the Hamming Codes achieve the worst results. However, since these codes are easy to implement with low complexity, they are often used in practical systems (e. g., [96]).

- Generally, we find that codes with a good performance f_k always have a relatively low number of information bits l, resulting in a relatively low maximum number of wet elements $|\mathrm{Wet}|$. However, they are suited for a low inverse relative message length, since the number of embeddable bits per block k is high. In practice, we have to find a compromise between l and k, dependent on the actual cover and its percentage of randomly distributed wet elements p_{Wet} on the one hand and the relative message length on the other hand.

6.6. Problem Discussion

Note that the codes investigated so far are not able to achieve results close to the upper boundary of embedding efficiency. However, it is known from Cohen et al. [14, Theorem 12.3.5], that the upper boundary on embedding efficiency is asymptotically achievable using (n, l) linear codes with $n \to \infty$. Thus, for codes with a large codeword length, the upper boundary is theoretically reachable.

However, for this kind of codes it is hard to determine the coset leader, i. e., it is hard to minimize the embedding impact. Generally, finding the coset leader, i. e., the sequence that minimizes the embedding distortion, is an NP-complete problem [4] since the size of a coset increases exponentially with l.

Thus, an exhaustive search is technically feasible as long as the parameter l remains small. However, with increasing codeword length n and constant l, the inverse relative message length α^{-1} decreases. Consequently, the embedding efficiency decreases. In the worst case, i. e., for $n \to \infty$ and constant l, $\alpha^{-1} \to 1$. Thus, the embedding efficiency is limited to $e = 2$.

Fridrich et al. investigate in their approach Matrix Embedding for Large Payloads this effect for $l = 10, 12, 14$ and different values for the inverse relative message length α^{-1}. However, even for $l = 10$ and $\alpha^{-1} = 1.11$, the maximum achievable block length is $n = 100$ [52]. Thus, moderate embedding rates are not feasible for codes with a large codeword length. However, some codes built according to deterministic rules, can be seen as quantizers. For this class of codes, no exhaustive search is required, the coset leader can be calculated directly.

For Hamming Codes (HC) with $f_k = 1$, for example, it is possible to simply analyze the syndrome derived from $\mathbf{s} = \mathbf{H}_{k\times n}\mathbf{a}^T \oplus \mathbf{emb}$, where $\mathbf{s}$ is related to a position in $\mathbf{H}_{k\times n}$. This position has to be flipped in $\mathbf{a}$, in order to embed while minimizing the embedding impact. Every syndrome, excluding the syndrome with weight zero, is mapped to a sequence $\mathbf{f}$ with $\mathrm{w}(\mathbf{f}) = 1$, i. e., the weight of each coset leader is at maximum 1.

However, comparing the embedding efficiency for Hamming Codes with different parameters k with the upper boundary on embedding efficiency, we find that there is indeed still a huge gap.

Another code with practical and fast decoders that are quantizers, are Simplex Codes, the dual codes to Hamming Codes. However, also this class of codes, investigated by Fridrich et al. in [52] is not close to the upper boundary. Note that exhaustive search is necessary for structured codes (aside from Hamming Codes and Simplex Codes), since these codes do not allow us to directly calculate the coset leader.

Part III.

Approaches for a Large Code Word Length

In this part, we introduce algorithms for syndrome coding based on constrained stochastic parity-check matrices designed for a large codeword length with $n = N$. We give a description and some specifics of state-of-the-art algorithms. Recall that we focus on the embedding step Θ.

As a measure for the performance of an embedding algorithm, the embedding efficiency e plays an important role. It represents the average number of embeddable bits per embedding change and is limited by a theoretically determined upper bound (Section 3.4.3.1). In practical investigations, the reachable embedding efficiency depends on both, the code used for embedding as well as the embedding algorithm itself. Note that we have to minimize the distortion between cover and stego sequence introduced during embedding in order to maximize e.

The codes investigated so far are limited to a relatively small codeword length. However, as [14, Theorem 12.3.5] indicates, the use of large codeword length is advantageous to reach the upper bound for e.

The first approach proposed for a large codeword length are the so-called Wet Paper Codes, i. e., simple variable rate stochastic codes [44]. However, these codes are not designed to maximize the embedding efficiency since they simply determine one solution to the system of linear equations (Equation (3.33)). This solution does not necessarily have to minimize the distance between the cover sequence $\mathbf{a}$ and the stego sequence $\mathbf{b}$ and thus not necessarily maximize the embedding efficiency e.

Generally, it seems to be reasonable to investigate sparse matrices in order to handle a large codeword length. Lately, independent investigations in Low-Density-Generator-Matrix (LDGM) Codes combined with SP by Fridrich et al. and BP by Günther et al. have reached a remarkable embedding efficiency and thus seem to be promising [37, 60].

Another approach, based on convolutional codes was proposed by Filler et al. as Syndrome Trellis Codes (STC) [24]. This approach is able to minimize the embedding impact and also to handle arbitrary profiles of embedding impact.

7. Embedding Based on Wet Paper Codes

Generally, Embedding considering Wet Paper Codes (WPC) is realized with syndrome coding based on a pseudo-random parity-check matrix $\mathbf{H}_{k\times n}$. The probability of 0 and 1 in $\mathbf{H}_{k\times n}$ is the same. Thus, the weight of the matrix is $\mathrm{w}(\mathbf{H}_{k\times n}) \approx \frac{kn}{2}$.

Generally, Wet Paper Codes were introduced in the context of the concept "writing on wet paper", introduced in Section 2.1.2.2, which can be seen as a non-shared selection channel [44]. Since the selection rule is known only to the sender, the receiver has no information about the positions used for embedding.

This problem is known in information theory as writing in memory with defective cells [73]: A computer memory contains n cells out of which $n-k$ cells are permanently stuck. The task is now to write as many bits as possible in the memory considering the fact that only the writing device knows the location and status of the stuck cells. Of course, the reading device should be able to correctly read the data.

It follows from the Gel´fand-Pinsker theorem for channels with random parameters known to the sender [57], that it is possible to write all k bits into the memory as $n \to \infty$. Thus, the capacity of this channel is k.

Fridrich et al. pointed out that the problem of writing in memory with defective cells is similar to the problem of writing on wet paper, where l bits are wet and should not be modified, while the remaining $k = n - l$ bits can be used for embedding[25]. Again, the receiver does not know the dry set.

However, there are three main differences between coding for defective memory and Wet Paper Steganography [45]: First, the number of wet pixels can be quite large in natural covers, e. g., $n = N = 10^6$ and $l = 10^4$. Second, the number of wet pixels varies from cover to cover, i. e., depending on the selection channel, different amounts of cover elements are considered wet. Beside these two disadvantages of Wet Paper Steganography, the last difference can be seen as an advantage: The steganographic approach is not that time critical since often no real time performance is required.

[25] Thus, this approach can be applied for a general profile of embedding impact. However, since the approach excludes wet elements and all dry elements can be used for embedding, the average number of embedding changes R_a is again an appropriate measure for the introduced distortion.

Generally, Wet Paper Steganography can also be seen as a generalization of the selection channel of Anderson [2], where the message bit is encoded as parity of a group of elements. Note that neither the Selection Channel [2] nor Wet Paper Steganography [44] require the sender to share any knowledge of the constraints with the recipient, whereas the latter approach does not even sacrifice embedding capacity – however at the cost of increased embedding complexity.

7.1. Description of the Basic Approach

Since the receiver will extract the message by solving $\mathbf{emb} = \mathbf{H}_{k\times n}\ \mathbf{b}^T$, the sender needs to solve a system of linear equations. The algorithm is equivalent to the one described in Section 4 and can be summarized as follows [44]:

1. Eliminate positions with $\mathbf{f}[i] = 0$ from $\mathbf{f} \Rightarrow \mathbf{f}_{|\mathrm{Dry}|}$
2. Eliminate the corresponding columns from $\mathbf{H}_{k\times n} \Rightarrow \mathbf{H}_{k\times|\mathrm{Dry}|}$
3. Recall Equation (4.2)

$$\mathbf{H}_{k\times|\mathrm{Dry}|}\left(\mathbf{f}_{|\mathrm{Dry}|}\right)^T = \mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T$$

4. Determine a solution for $\mathbf{f}_{|\mathrm{Dry}|}$
5. Determine the flipping pattern $\mathbf{f}$ related to $\mathbf{f}_{|\mathrm{Dry}|}$
6. Determine the stego sequence $\mathbf{b} = \mathbf{a} \oplus \mathbf{f}$

Contrary to the approaches described in Section 4, the algorithm is applied to the whole cover instead to small blocks as for the Block Minimal Method. Furthermore, there is no search for a coset leader, i. e., a flipping pattern that minimizes the introduced distortion in Step 4. Instead, simply the system of linear equations is solved.

As mentioned before, $\mathbf{emb} = \mathbf{H}_{k\times n}\ \mathbf{b}^T$ has a solution as long as $\mathbf{H}_{k\times n}$ has full rank. Thus, ideally $\mathrm{rank}(\mathbf{H}_{k\times n}) = k$ bits can be embedded by means of syndrome coding, i. e. $|\mathbf{emb}| = k$. However, given that stochastic matrices will have no full rank with high probability (see Section 3.2.1), the sender is able to communicate only q bits with $q <= k$ with this scheme, i. e. $|\mathbf{emb}| <= k$.

Whenever it is impossible to solve the system of linear equations, i. e., the matrix' rank is $< k$, the sender needs either to discard the cover or to shorten the message, i. e., to reduce k and try again to find a solution. Of course this iterative procedure is time consuming.

Because of the variable message length $q <= k$, the sender needs to communicate this parameter to the receiver. Therefore, he encodes the message length q within a fixed section of the cover. By doing so, the receiver is able to extract

the message length q and build the pseudo-random binary $q \times n$ matrix $\mathbf{H}$ needed for extracting the message.

A simple example for the Wet Paper Code is given below.

Example:

A stochastic code with block length $n = N = 5$ is given. The sender would like to embed the message $\mathbf{emb} = (100)^T$ into the cover sequence $\mathbf{a} = (10111)$, the elements $\mathbf{a}[2]$ and $\mathbf{a}[5]$ are wet.

In this example, the parity-check matrix is given with:

$$\mathbf{H}_{3\times 5} = \begin{pmatrix} 1\,0\,1\,1\,0 \\ 0\,1\,1\,0\,1 \\ 1\,0\,0\,0\,0 \end{pmatrix}.$$

After defining $\mathbf{f} = \mathbf{b} \oplus \mathbf{a} = (\mathbf{f}[1]0\mathbf{f}[3]\mathbf{f}[4]0)$, the sender embeds the message according the following scheme:

1. Eliminate positions with $\mathbf{f}[i] = 0$ from $\mathbf{f} \Rightarrow \mathbf{f}_{|\mathrm{Dry}|} = (\mathbf{f}[1]\mathbf{f}[3]\mathbf{f}[4])$
2. Eliminate the corresponding columns from $\mathbf{H}_{k\times n} \Rightarrow \mathbf{H}_{k\times|\mathrm{Dry}|} = \begin{pmatrix} 1\,1\,1 \\ 0\,1\,0 \\ 1\,0\,0 \end{pmatrix}$
3. Recall Equation (4.2) with $\mathbf{H}_{k\times|\mathrm{Dry}|}(\mathbf{f}_{|\mathrm{Dry}|})^T = \mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T$
4. Determine a solution to $\mathbf{f}_{|\mathrm{Dry}|}$

$$\begin{pmatrix} 1\,1\,1 \\ 0\,1\,0 \\ 1\,0\,0 \end{pmatrix} \begin{pmatrix} \mathbf{f}[1] \\ \mathbf{f}[3] \\ \mathbf{f}[4] \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \oplus \begin{pmatrix} 1\,0\,1\,1\,0 \\ 0\,1\,1\,0\,1 \\ 1\,0\,0\,0\,0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{f}[1] \oplus \mathbf{f}[3] \oplus \mathbf{f}[4] \\ \mathbf{f}[3] \\ \mathbf{f}[1] \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} \oplus \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{f}[1] \oplus \mathbf{f}[3] \oplus \mathbf{f}[4] \\ \mathbf{f}[3] \\ \mathbf{f}[1] \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$$\begin{pmatrix} \mathbf{f}[1] \\ \mathbf{f}[3] \\ \mathbf{f}[4] \end{pmatrix} = \begin{pmatrix} 1 \\ 0 \\ 1 \end{pmatrix}$$

5. Determine the flipping pattern $\mathbf{f}$ related to $\mathbf{f}_{|\mathrm{Dry}|}$

$\mathbf{f}_{|\mathrm{Dry}|} = (101) \Rightarrow \mathbf{f} = (10010)$

6. Determine the stego sequence
   $\mathbf{b} = \mathbf{a} \oplus \mathbf{f} = (10111) \oplus (10010) = (00101)$

The receiver extracts the message with: $\mathbf{emb} = \mathbf{Hb}^T$

$$\begin{pmatrix} 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 1\,0\,1\,1\,0 \\ 0\,1\,1\,0\,1 \\ 1\,0\,0\,0\,0 \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 1 \end{pmatrix}.$$

■

Note that the complexity of the communication lies on the sender's side, because he has to solve $q <= k$ linear equations with q unknown parameters. Contrary to the approaches described in Section 4, the algorithm is applied to the whole cover. Furthermore, the system of linear equations is solved and no search for the coset leader is applied. Consequently, the embedding process based on this approach is not able to minimize the introduced embedding distortion.

In order to reduce the complexity of embedding, several approaches have been proposed, e. g., the application of Structured Gaussian Elimination or the application of sparse matrices. These approaches for finding a faster solution, are summarized in the next sections.

7.2. Wet Paper Codes Combined with Structured Gaussian Elimination

Since solving the system of linear equations (Equation (3.33)) is quite complex applying the approach presented in Section 7.1, several approaches were proposed reducing the computational complexity.

One approach to accelerate the process of finding a solution to the system of linear equations was proposed in [44, 42]. In order to speed up the process of solving (Equation (3.33)), the algorithm described in Section 7.1 is applied to small blocks. Note that this algorithm can also be considered as an approach for a small codeword length. However, similar to the approach described in Section 7.1, this approach is not able to minimize the distortion introduced during embedding. Instead of searching the coset leader, simply a solution to the system of linear equations is determined. Thus, we consider this approach as a possibility to fasten the process of solving the system of linear equations.

The algorithm itself can be summarized as follows: The cover is divided into $n_B = |\mathbf{m}|/250$ pseudo-random disjoint subsets with $n = N/n_B$. For each block $\approx$

250 changeable elements are required in order to keep the complexity controllable. Note that the actual number of changeable elements varies from subset to subset following a hyper-geometrical distribution with mean k/n_B [42]. Note also that the subsets should be of the same size and sender and receiver need to obtain the same blocks.

After dividing the cover into blocks, Structured Gaussian Elimination is applied on each subset. Dividing the cover into blocks will reduce the complexity of Gaussian Elimination dramatically. While the complexity is reduced by factor n_B^3, the number of solvings increases n_B-times. Thus, the total performance is increased by factor n_B^2.

A disadvantage of this approach is the need to communicate the message length embedded in each subset, resulting in a slight loss in capacity. Since the sender tries to embed as many bits as possible in each subset, the message length cannot be encoded in the header at the beginning of each block. As a solution, Fridrich et al. proposed to encode the message length for each block within the last block. Consequently, the receiver has to read this block first.

Note that this algorithm is still not able to maximize the embedding efficiency by minimizing the introduced distortion. However, we believe that this approach was the first step forward to the Block Minimal Method described in Section 4.1.

7.3. Wet Paper Codes Based on Sparse Matrices

Instead of a block-wise application of the algorithm described in Section 7.1, different approaches try to make the process of solving Equation (3.33) faster by substituting the process of solving the system of linear equations in Step 4.

Therefore, the basic idea of Wet Paper Codes based on sparse matrices is to increase the velocity of the process of solving by introducing some structure into the pseudo-random parity-check matrix $\mathbf{H}_{k\times n}$ and thereby into the sub-matrix $\mathbf{H}_{k\times|\mathrm{Dry}|}$ used for embedding.

Contrary to Wet Paper Codes, where matrices of weight $\approx \frac{nk}{2}$ where used, Fridrich et al. propose the usage of sparse matrices to reduce the complexity [42]. Therefore, the parity-check matrix is based on a constrained stochastic process (see Section 3.2.2). Thus, we find $P(1) = \delta$ with $\delta < 0,5$. Consequently, $\mathrm{w}(\mathbf{H}_{k\times n})$ grows only linearly with increasing block length.

Remember that $\mathbf{H}_{k\times n}$ is a matrix independent of the cover. However, since the positions of wet elements vary from block to block, also the sub-matrix $\mathbf{H}_{k\times|\mathrm{Dry}|}$ derived from $\mathbf{H}_{k\times n}$ varies from block to block. Thus, the sender has no option to control or influence the sub-matrix $\mathbf{H}_{k\times|\mathrm{Dry}|}$ directly. However, imposing some structure to the columns of $\mathbf{H}_{k\times n}$ will inherit the columns of $\mathbf{H}_{k\times|\mathrm{Dry}|}$ too, because it is a sub-matrix obtained by removing some columns.

Note that the sparser the matrices, the faster Gaussian Elimination in Step 4 can be carried out. However, with decreasing density $\mathrm{w}(\mathbf{H}_{k\times n})$, the probability

of the matrix of being of rank k decreases. The rank remains approximately k until the density reaches the critical density $\frac{\mathrm{ld}k}{k}$. In order to be able to embed, the matrix must guarantee that in each subset the density does not fall below the critical density [42].

Beside Gaussian Elimination, solving the system of linear equations in Step 4 can be carried out using significantly faster methods such as the probabilistic algorithms of Lanczos [74] and Wiedemann [99] for sparse matrices.

However, since in this application, the matrix $\mathbf{H}_{k\times n}$ is rectangular and may be singular, the application of both methods is complicated, i. e., the design of the matrix slows down the solver. Consequently, Fridrich et al. found in a comparison between Gaussian Elimination and the algorithms of Wiedemann and Lanczos that Gaussian Elimination is best for $k = \{250, 500, 1000, 2000, 5000\}$ and Wiedemann is best only for a large message length such as $k = \{10000, 20000\}$ [42]. Furthermore, Fridrich et al. mention within their work that imposing some structure on $\mathbf{H}_{k\times|\mathrm{Dry}|}$ may lead to codes with suboptimal performance.

7.4. Wet Paper Codes Combined with the Matrix LT Process

Another approach for the simplification of Step 4 (see Section 7.1) is the application of principles from Luby Transform (LT) Codes in order to speed up the process of solving Equation (3.33).

Luby Transform Codes (LT Codes) are universal erasure codes with low encoding and decoding complexity [77]. They provide a method for a fast solution of an overdetermined system of equations as long as the number of ones in each row follows the robust soliton distribution (RSD). This distribution ensures at least one row with weight 1. For this class of matrices, fast methods for solving the system of linear equations based on the bipartite graph exist.

However, LT Codes cannot be applied directly to Equation (3.33) because this system of linear equations is under-determined. Instead, LT can be used to bring $\mathbf{H}_{k\times n}$ quickly in the upper triangular form by swapping of rows and columns.

The basic idea described in [43, 46, 45, 47], can be formulated as follows: make $\mathbf{H}_{k\times n}$ and thus $\mathbf{H}_{k\times|\mathrm{Dry}|}$ sparse. In this case, it can be put into upper-diagonal form with high probability simply by permuting its rows and columns. Consequently, Equation (3.33) can be efficiently solved in Step 4 using the standard back-substitution as in Gaussian Elimination.

This permutation procedure is called Matrix LT Process because it was originally invented for LT Codes. The algorithm substituting Step 4 can be described as follows:

4.1 Find a column in $\mathbf{H}_{k\times|\mathrm{Dry}|}$ that has exactly one 1 in the i_1th row
- Swap this column with the first column

- Swap the first row and the i_1th row
- 1 is in the upper left corner, i. e., $\mathbf{H}_{k\times|\text{Dry}|}[1,1] = 1$ and $\mathbf{H}_{k\times|\text{Dry}|}[i,1] = 0$ for $i > 1$

4.2 Apply the same step ignoring the first column and row

- Column with one 1 in the i_2th row
- Swap column with second column of $\mathbf{H}_{k\times|\text{Dry}|}$
- Swap second and i_2th row
- $\mathbf{H}_{k\times|\text{Dry}|}[1,1] = 1, \mathbf{H}_{k\times|\text{Dry}|}[i,1] = 0$ for $i > 1$
- $\mathbf{H}_{k\times|\text{Dry}|}[2,2] = 1, \mathbf{H}_{k\times|\text{Dry}|}[i,2] = 0$ for $i > 2$

4.3 Continue this process ignoring the first two columns and rows

4.4 Find $\mathbf{H}_{k\times|\text{Dry}|}$ in the upper triangular form with

$$\mathbf{H}_{k\times|\text{Dry}|}[j,j] = 1, j = (1,...,k) \text{ and } \mathbf{H}_{k\times|\text{Dry}|}[i,j] = 0 \text{ for } i > j$$

4.5 Use back-substitution to determine the solution to $\mathbf{f}'_{|\text{Dry}|}$

4.6 Apply the transpositions to $\mathbf{f}'_{|\text{Dry}|}$ in reverse order to achieve $\mathbf{f}_{|\text{Dry}|}$

The resulting sequence $\mathbf{f}_{|\text{Dry}|}$ is the solution to the system $\mathbf{H}_{k\times|\text{Dry}|}(\mathbf{f}_{|\text{Dry}|})^T = \mathbf{emb} \oplus \mathbf{H}_{k\times n}\mathbf{a}^T$. Afterwards, Step 5 and Step 6 (see Section 7.1) have to be applied.

Note that the sender applies the transpositions in reverse order in the last step. Thus, only the sender needs to rearrange the matrix $\mathbf{H}_{k\times|\text{Dry}|}$, the receiver does not have to care about this. He extracts by simply calculating $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$.

However, whenever the permutation process is not able to find a column with exactly one 1, the Matrix LT Process fails. Since this algorithm is based on stochastic matrices with constraints, i. e., the weight of columns of $\mathbf{H}_{k\times n}$ follows the RSD, this failure cannot be excluded.

The procedure in the case of failure can be described as follows: Since discarding rows as proposed so far is not a solution in this case,[26] Fridrich et al. propose to make $\mathbf{H}_{k\times n}$ dependent on the message length $k = |\mathbf{emb}|$, e. g., as input for the Pseudo Random Number Generator (PRNG) [36]. In the case of a failure, Fridrich et al. propose to append a dummy bit to the message and try again with seed $k + 1$.

Note that in practice, the receiver has to know the message length $|\mathbf{emb}|$. Since the RSD depends on the message length, it must be communicated to the receiver.

[26] Discarding rows has no influence on the distribution of ones per row. Thus, only discarding columns would be a solution (but, a rather impractical one).

7.5. Summary

Wet Paper Codes were proposed as an approach to communicate in a non-shared selection channel scenario. The goal was to communicate without the necessity of informing the receiver about the selection rule, i. e., about the parts of the cover used for embedding the confidential message. Therefore, Fridrich et al. proposed simple variable rate stochastic codes [44].

However, these codes suffer from two problems. First, since they consider the cover at once, finding a solution to the system of linear equations (Equation (3.33)) is really complex. Second, these codes are not designed to maximize the embedding efficiency since they simply determine one solution to the system of linear equations (Equation (3.33)). This solution does not necessarily have to minimize the distance between the cover sequence $\mathbf{a}$ and the stego sequence $\mathbf{b}$ and thus not necessarily maximize the embedding efficiency e.

To overcome the first problem concerning complexity, several approaches have been proposed in literature in order to speed up the process of solving Equation (3.33). Within the previous sections, we summarized the basic ideas of state-of-the-art methods based on sparse matrices as well as on small blocks.

However, in order to maximize the embedding efficiency, algorithms based on small blocks were introduced in literature such as embedding based on stochastic matrices (Block Minimal Method, see Section 4.1) or based on deterministic matrices (BCH Codes, see Section 5.2).

Yet, due to the inability of these codes to reach the upper bound on embedding efficiency (Section 6.6), further investigations were based on different approaches for large block length. Within the next sections, we will introduce embedding based on Low-Density-Generator-Matrix (LDGM) Codes [37, 60] and also based on Syndrome Trellis Codes [24].

8. Embedding Based on LDGM Codes

So far, the approaches based on random matrices were not able to minimize the distortion introduced during embedding. The approaches described in the previous section determine only one solution - to the system of linear equations (Equation (3.33)) - which does not have to minimize the introduced distortion.

Instead of a stochastic parity-check matrix, it is also possible to generate the parity-check matrix used for embedding based on a linear code. However, it is not feasible to search for a coset leader by means of exhaustive search in the case of a large block length since the complexity increases exponentially with increasing n (see Section 6.5).

Instead, stochastic matrices with constrained rules were investigated, more precisely, sparse matrices. For this class of matrices, the weight of the matrix increases linearly instead of quadratically with increasing block length n. In this case, iterative search strategies for the coset leader become applicable.

Consequently, the application of codes based on sparse matrices, i. e., Low-Density-Codes, reduce the complexity of embedding and thus, make the application of a large codeword length in Minimum-Embedding-Impact Steganography feasible.

Based on this assumption, it seemed to be reasonable to investigate Low-Density-Parity-Check (LDPC) Codes combined with the iterative message passing algorithm Belief Propagation (BP). In this case, embedding is based on a low density matrix $\mathbf{H}_{k\times n}$. Furthermore, existing decoding algorithms for LDPC Codes based on BP can be applied in order to determine the closest codeword.

However, as our investigations have shown, it is not possible to apply the iterative decoding of LDPC Codes in steganography when the confidential message **emb** and the cover bit string **a** are selected randomly [58]. The decoder fails, since the average Hamming distance between the cover sequence **a** and the nearest codeword is mostly large. This is not a problem in channel coding where the distance to a codeword is usually small.

Instead, Low-Density-Generator-Matrix (LDGM) Codes were considered, the dual codes of LDPC Codes, i. e., an approach based on a low density generator matrix $\mathbf{G}_{l\times n}$. For LDGM Codes, the decoder, and thus embedding, does not fail even for larger distances to the next codeword. This is true since this approach is based on compression. The cover bit string **a** is compressed to $\mathbf{c}^*$. The desired

codeword $\mathbf{c}$ is then achieved through a multiplication of $\mathbf{c}^*$ with the generator matrix $\mathbf{G}$.

In this section, we would like to give a description of the approach of applying LDGM Codes to syndrome-based embedding in steganography. The application of such LDGM Codes in steganography was independently proposed by Fridrich and Filler based on Survey Propagation (SP) [19, 37] and Günther, Schönfeld and Winkler based on Belief Propagation (BP) [58, 59, 60].

Note that the BP algorithm is known as a considerably less complex special case of the SP algorithm introduced in [94]. Furthermore, additional approximations are known in the context of LDPC Codes, to further reduce the decoding complexity of BP. Thus, in this thesis, we adapted principles from LDPC decoding such as the BP algorithm combined with approximations resulting in a reduced embedding complexity compared to [37].

Note that the approach based on LDGM Codes presented in this section tries to maximize the embedding efficiency. However, since the algorithm is based on an iterative approximation of the coset leader, the maximum possible embedding efficiency cannot be achieved in all cases. However, the iterative approach enables the application of a large codeword length and thus more powerful codes.

8.1. Basic Considerations

LDGM Codes are linear block codes with a low-density generator matrix $\mathbf{G}_{l\times n}$, i. e., $\mathrm{w}(\mathbf{G}_{l\times n})$ grows only linearly with growing n. Generally, linear codes can be described based on a parity-check matrix $\mathbf{H}_{k\times n}$ or on a generator matrix $\mathbf{G}_{l\times n}$, where l stands for the number of information bits with $n = l + k$ (see Section 3.2.3). Note that it is possible to determine the parity-check matrix $\mathbf{H}_{k\times n}$ related to $\mathbf{G}_{l\times n}$ according to the following coherence: $\mathbf{G}_{l\times n}\,{\mathbf{H}_{k\times n}}^T = \mathbf{0}$.

In this approach, the extraction of the secret message at the receiver's side is again based on a parity-check matrix $\mathbf{H}_{k\times n}$ which has to be derived from $\mathbf{G}_{l\times n}$. The confidential message can be found with $\mathbf{emb} = \mathbf{H}_{k\times n}\,\mathbf{b}^T$.

The special characteristic of LDGM Codes is the iterative application of the decoding algorithm with linear complexity in n in order to find the closest codeword. Thus, the embedding process itself is based on the concept of reformulating the search for an appropriate flipping pattern $\mathbf{f}$ as the search of the closest codeword (see Section 3.5):

1. Find a coset member $\mathbf{f_m} \in \mathcal{C}(\mathbf{emb})$
2. Determine the closest codeword using an iterative message passing algorithm according to Equation (3.41)

$$\mathbf{c_{emb}} = \underset{\mathbf{c}\in\mathcal{C}}{\operatorname{argmin}}\, d_\rho(\mathbf{a} \oplus \mathbf{f_m},\ \mathbf{c}) \tag{8.1}$$

3. Determine the flipping pattern $\mathbf{f} = \mathbf{a} \oplus \mathbf{f_m} \oplus \mathbf{c_{emb}}$

4. Determine the stego sequence $\mathbf{b} = \mathbf{f_m} \oplus \mathbf{c_{emb}} = \mathbf{a} \oplus \mathbf{f}$.

Note that it is always possible to derive a systematic parity-check matrix $\mathbf{H}_{k\times n}$ of the form $\mathbf{H}_{k\times n} = \begin{bmatrix}\mathbf{I}_k & \mathbf{R}_{k\times l}\end{bmatrix}$, e. g., by Gaussian elimination as described in [58]. In this case, an arbitrary member of $\mathcal{C}(\mathbf{emb})$ can easily be found with $\mathbf{f_m} = \begin{bmatrix}\mathbf{emb} & \mathbf{0}_l\end{bmatrix} \in \mathcal{C}(\mathbf{emb})$[27].

Considering an LDGM Code, iterative approximations with linear complexity in n can be used for the search in Equation (8.1). This enables us to choose codes with a large codeword length, implying a high probability to achieve an embedding efficiency close to the upper bound $e \leq \alpha / H^{-1}(\alpha)$ (see Section 3.4.3.1).

Note that the message passing part of the iterative algorithm can be carried out according to several approaches: Survey Propagation as described by Fridrich and Filler [94, 37] and Belief Propagation as described by Günther, Schönfeld and Winkler in [60] can be applied. Note that Belief Propagation can also be carried out in the logarithmic domain resulting in a considerably low complexity [60].

In the following, we describe our approach for finding the closest codeword based on Belief Propagation. More details are given in the next section, where $\mathbf{a} \oplus \mathbf{f_m}$ is denoted as $\mathbf{y}$.

8.2. Embedding with Belief Propagation

In order to find $\mathbf{c_{emb}}$ according to Equation (8.1), we apply the BP algorithm to iteratively converge to the closest codeword. Therefore, we first have to determine the vector $\mathbf{c}^* \in \mathbb{F}_2^l$, which minimizes the number of unsatisfied equations in the overdetermined equation system:

$$\mathbf{c}^* \, \mathbf{G}_{l\times n} = \mathbf{y}^T. \tag{8.2}$$

To solve Equation (8.2) with the BP algorithm, we need to describe the system as an undirected bipartite Graph $G = (V, E)$ with nodes V and edges E, denoted as Tanner Graph (see Figure 8.1). Therefore, matrix $\mathbf{G}_{l\times n}$ is interpreted as the incidence matrix of the graph. Every variable in $\mathbf{c}^*$ and $\mathbf{y}$ is represented by a variable node.

Furthermore, the graph contains n functional nodes $\{f_1, \ldots, f_n\}$, one for each equation $\mathbf{c}^* \, \mathbf{G}_{l\times n}[., j] = \mathbf{y}[., j]$, with $\mathbf{G}_{l\times n}[., j]$ denoting the j^{th} column of $\mathbf{G}_{l\times n}$. The functional node f_j is connected to the variable node c_i^* by an edge $e \in E$, if c_i^* occurs in the equation. Thus, the graph contains a total of $(l + 2n)$ nodes $V = \{c_1^*, \ldots, c_l^*, y_1, \ldots, y_n, f_1, \ldots, f_n\}$. The notation $E(\text{node})$ denotes the set of edges connected to a node, where e_t is the t^{th} element. Figure 8.1 visualizes the translation of the matrix $\mathbf{G}_{l\times n}$ into a Tanner Graph.

[27] Note that it is also possible to chose $\mathbf{f_m}$ as $[\mathbf{emb} \oplus \mathbf{H}_{k\times n} \, \mathbf{a}^T \; \mathbf{0}_l]$ as described in Section 4.2.

$$\mathbf{G}_{l\times n} = \begin{array}{|ccccccc|c|} \hline f_1 & f_2 & f_3 & f_4 & f_5 & f_6 & f_7 & \\ \hline 1 & 1 & 0 & 1 & 0 & 1 & 0 & \mathbf{c}_1^* \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & \mathbf{c}_2^* \\ 1 & 0 & 1 & 1 & 0 & 1 & 1 & \mathbf{c}_3^* \\ 0 & 0 & 0 & 1 & 1 & 0 & 1 & \mathbf{c}_4^* \\ \hline \end{array} \Longrightarrow$$

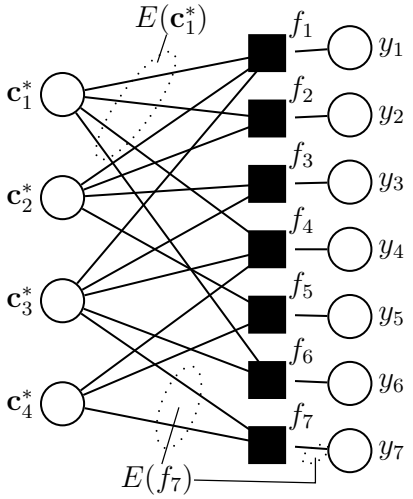

Figure 8.1.: Translation of the Matrix $\mathbf{G}_{l\times n}$ with $n = 7$ and $l = 4$ into a Tanner Graph According to [60].

The iterative algorithm for finding the closest codeword itself is divided into two parts: message passing and variable decimation.

Every iteration starts with the update of the variable nodes and the delivery of messages from these nodes to their functional neighbors. In the second step, the functional nodes are updated. Based on the messages they received, new messages are computed and delivered to their variable neighbors. Since $\mathbf{G}_{l\times n}$ is sparse, the number of neighbors and thus the required effort for updating is quite low.

After several iterations T, the algorithm provides evidence for each element in $\mathbf{c}^*$ which value to assign. Now, elements with high evidence are fixed and BP is restarted with the reduced system [94]. This iterative procedure as applied by Günther, Schönfeld and Winkler in [60] is visualized in Figure 8.2. Note that $\mathbf{c}$ can be achieved according to $\mathbf{c}^* \mathbf{G}_{l\times n}$. As it can be seen in this figure, this iterative approach is not always able to find the optimum, i. e., the coset leader that minimizes the introduced distortion.

In the next section, we describe the individual steps of the BP algorithm in order to give a survey according to [60]. For more details, we refer to [58].

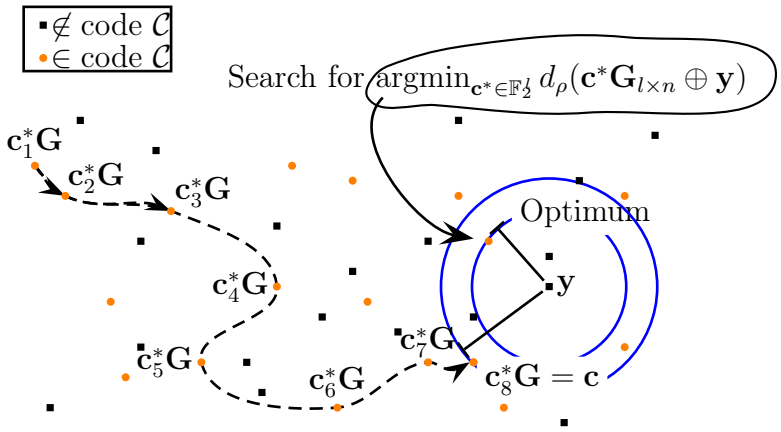

Figure 8.2.: Visualization of the Iterative Search for the Closest Codeword According to [60].

8.2.1. Updating Variable Nodes

In order to solve Equation (8.2), we have to find an assignment for each $c_i^* \in \mathbf{c}^*$. Our main concern is to decide for every variable c_i^* in $\mathbf{c}^*$ which value from $\mathbb{F}_2 = \{0, 1\}$ to assign to it. With $P_i(0)$ we denote an estimation or "belief" for the probability that all equations involving c_i^* are satisfied for $c_i^* = 0$. The same applies for $P_i(1)$. In order to decide whether $c_i^* = 0$ or $c_i^* = 1$, we try to maximize the probability of satisfying all equations that are related to c_i^*:

$$c_i^* = \begin{cases} 0 & \text{for } P_i(0) \geq P_i(1) \\ 1 & \text{for } P_i(0) < P_i(1). \end{cases} \tag{8.3}$$

In each step, the node c_i^* receives two messages $P_{t \to i}(0)$ and $P_{t \to i}(1)$ from each of the t connected functional nodes.

Note that in the case of Survey Propagation, the matrix is also interpreted as Tanner Graph. However, the code is additionally extended to generalized codewords: Each variable in $\mathbb{F}_2$ can additionally be set to $*$, i. e, can be assigned to a value not related to 0 nor 1. Instead of two messages as for BP, in the case of SP 5 messages have to be exchanged between the nodes.

The two messages in case of BP are estimates for the probabilities of the assignments $c_i^* = 0$ and $c_i^* = 1$ respectively, satisfying the equation connected to the edge e_t. The probability that all equations connected to c_i^* are satisfied equals the product of the probabilities for satisfying each equation independently:

$$P_i(0) = \frac{\prod_{e_t \in E(c_i^*)} P_{t\to i}(0)}{\prod_{e_t \in E(c_i^*)} P_{t\to i}(0) + \prod_{e_t \in E(c_i^*)} P_{t\to i}(1)} \tag{8.4}$$
$$P_i(1) = 1 - P_i(0).$$

The product is normalized in order to fulfill the constraint $P_i(0) + P_i(1) = 1$.

This information needs to be propagated to the t connected functional nodes with the messages $P_{i\to t}(0)$ and $P_{i\to t}(1)$. One important principle of the BP algorithm is that only extrinsic information is forwarded, i. e., information previously received from node t needs to be eliminated from Equation (8.4):

$$P_{i\to t}(0) = \frac{P_i(0)/P_{t\to i}(0)}{P_i(0)/P_{t\to i}(0) + P_i(1)/P_{t\to i}(1)}$$
$$P_{i\to t}(1) = 1 - P_{i\to t}(0).$$

The variable nodes $y_1, \ldots, y_n$ do not need to be updated, since they send the same messages in every iteration:

$$P_{j\to t}1(0) = \frac{(1-y_j)e^{\gamma_j} + y_j e^{-\gamma_j}}{e^{\gamma_j} + e^{-\gamma_j}}$$
$$P_{j\to t}1(1) = 1 - P_{j\to t}1(0).$$

The parameter γ_j reflects how strongly the algorithm should try to preserve the value of y_j. We found appropriate values based on simulations (see Section 10.1.2).

8.2.2. Updating Functional Nodes

Each functional node f_j receives two messages $P_{t\to j}(0)$ and $P_{t\to j}(1)$ from each of the t connected variable node c_i^*. These messages are used to calculate the messages $P_{j\to t}(0)$ and $P_{j\to t}(1)$, which are sent back to every connected variable node:

$$P_{j\to t}(0) = \frac{1}{2}\left(1 + \prod_{e_{t'} \in E(f_j)\setminus\{e_t\}} (1 - 2P_{t'\to j}(1))\right) \tag{8.5}$$
$$P_{j\to t}(1) = 1 - P_{j\to t}(0). \tag{8.6}$$

The messages $P_{j\to t}(0)$ and $P_{j\to t}(1)$ represent the "belief" that the equation $\mathbf{c}^* \mathbf{G}_{l\times n}[., j] = \mathbf{y}[., j]$ is satisfied, if the variable connected to e_t equals 0 and 1, respectively.

8.2.3. Survey Inspired Decimation (SID)

After several iterations T, the BP algorithm provides evidence for each element in $\mathbf{c}^*$ which value to assign.

In [37] and [94], it was proposed to select a percentage of at least r_{min} and at most r_{max} variables for assignment and decimation. In order to choose the variables for decimation, we have to calculate β_i for each variable as:

$$\beta_i = |P_i(0) - P_i(1)| .$$

For decimation, the $r_{max}l$ variables with the highest values β_i are selected. Based on this selection, all variables fulfilling $\beta_i > \beta_{th}$ but at least $r_{min}l$ variables are assigned according to Equation (8.3). The threshold $\beta_{th} \in (0, 1)$ needs to be adjusted experimentally to r_{max} in order to obtain good results (see [58]).

Due to the decimation of c_i^*, the Tanner graph needs to be reduced and the constraints $\mathbf{y}$ have to be adapted according to $\mathbf{y}^T = \mathbf{y}^T \oplus c_i^* \, \mathbf{G}[i, .]$. The alternating execution of BP and decimation terminates, when all variables in $\mathbf{c}^*$ have been assigned.

8.2.4. Log Likelihood Relations and Approximations

The BP algorithm described so far has one major drawback: it involves multiplications. As it is commonly known, multiplications are more complex than, e. g., an addition, a comparison, or a shift-operation. To avoid this problem, Günther, Schönfeld and Winkler proposed the application of Log Likelihood Ratios (LLR) [58]. Within this approach, the two messages passed through each edge and the probabilities $P_i(0)$ and $P_i(1)$ are replaced by the logarithm of their ratios.

Applying this adaption to the variable node update results in a simple summation over the incoming messages. By transforming Equation (8.5) into an LLR, a functional node update can be simplified to a recursion [64]. Therewith the calculation of the LLR of two independent random variable becomes a central element:

$$\begin{aligned}
\mathrm{LLR}(x_1 \oplus x_2) &= \ln \frac{P(x_1 \oplus x_2 = 0)}{P(x_1 \oplus x_2 = 1)} \\
&= \ln \frac{e^{\mathrm{LLR}(x_1)+\mathrm{LLR}(x_2)} + 1}{e^{\mathrm{LLR}(x_1)} + e^{\mathrm{LLR}(x_2)}} \\
&= \mathrm{signLLR}(x_1)\mathrm{signLLR}(x_2) \min(|\mathrm{LLR}(x_1)|, |\mathrm{LLR}(x_2)|) \qquad (8.7) \\
&\quad + \ln\left(1 + e^{-|\mathrm{LLR}(x_1)+\mathrm{LLR}(x_2)|}\right) - \ln\left(1 + e^{-|\mathrm{LLR}(x_1)-\mathrm{LLR}(x_2)|}\right). \qquad (8.8)
\end{aligned}$$

Since the exact evaluation of the last term is still quite complex, different possibilities for an approximation are used in channel coding for LDPC Codes combined with BP.

Sign-Min: The simplest approach, denoted as Sign-Min approximation, is to ignore the Term (8.8).

Log-Offset: A more advanced approach, denoted as Log-Offset [64], replaces Term (8.8) with a constant offset:

$$\ln\left(1+e^{-|\mathrm{LLR}(x_1)+\mathrm{LLR}(x_2)|}\right)-\ln\left(1+e^{-|\mathrm{LLR}(x_1)-\mathrm{LLR}(x_2)|}\right)$$
$$\approx\begin{cases} o & \text{for } |\mathrm{LLR}(x_1)+\mathrm{LLR}(x_2)|<2 \;\&\; |\mathrm{LLR}(x_1)-\mathrm{LLR}(x_2)|>2\,|\mathrm{LLR}(x_1)+\mathrm{LLR}(x_2)| \\ -o & \text{for } |\mathrm{LLR}(x_1)-\mathrm{LLR}(x_2)|<2 \;\&\; |\mathrm{LLR}(x_1)+\mathrm{LLR}(x_2)|>2\,|\mathrm{LLR}(x_1)-\mathrm{LLR}(x_2)| \\ 0 & \text{otherwise,}\end{cases}$$

where the constant o should be predetermined in a way that the resulting approximation error is minimal. The optimal value depends on the distribution of $\mathrm{LLR}(x_1)$ and $\mathrm{LLR}(x_2)$, in channel coding we often find $o = 0.5$. We used this value for our simulations.

Log-Look-Up: Another, also quite simple implementation of Term (8.8) can be done by means of a step function as in our approach Log-Look-Up. Therefore, [64] proposes a Look-up Table with eight entries (Table 8.1). This enables us to obtain a fairly close approximation.

Table 8.1.: Implementation of $\ln(1+e^{-|\mathrm{LLR}(x_1)\pm\mathrm{LLR}(x_2)|})$ with a Look-up Table (Log-Look-Up).

			$\ln\left(1+e^{-\lvert\mathrm{LLR}(x_1)\pm\mathrm{LLR}(x_2)\rvert}\right)$
0	$\leq \lvert\mathrm{LLR}(x_1)\pm\mathrm{LLR}(x_2)\rvert <$	0.196	0.65
0.196	$\leq \lvert\mathrm{LLR}(x_1)\pm\mathrm{LLR}(x_2)\rvert <$	0.433	0.55
0.433	$\leq \lvert\mathrm{LLR}(x_1)\pm\mathrm{LLR}(x_2)\rvert <$	0.71	0.45
0.71	$\leq \lvert\mathrm{LLR}(x_1)\pm\mathrm{LLR}(x_2)\rvert <$	1.05	0.35
1.05	$\leq \lvert\mathrm{LLR}(x_1)\pm\mathrm{LLR}(x_2)\rvert <$	1.508	0.25
1.508	$\leq \lvert\mathrm{LLR}(x_1)\pm\mathrm{LLR}(x_2)\rvert <$	2.252	0.15
2.252	$\leq \lvert\mathrm{LLR}(x_1)\pm\mathrm{LLR}(x_2)\rvert <$	4.5	0.05
4.5	$\leq \lvert\mathrm{LLR}(x_1)\pm\mathrm{LLR}(x_2)\rvert <$	$+\infty$	0.00

Log-Linear: We also investigated a more precise approximation than Log-Offset but less complex than the Look-up Table, denoted as Log-Linear [82], where Term

(8.8) is replaced by a function:

$$\ln(1 + e^{-|\mathrm{LLR}(x_1) \pm \mathrm{LLR}(x_2)|}) \approx \begin{cases} u - v\,|\mathrm{LLR}(x_1) \pm \mathrm{LLR}(x_2)| & \text{for } |\mathrm{LLR}(x_1) \pm \mathrm{LLR}(x_2)| < \frac{u}{v} \\ 0 & \text{otherwise.} \end{cases}$$

The optimum value of the constants u and v again depends on the distribution of the LLR. A good setup seems to be $u = 0.6, v = 0.24$ [82]. Günther et al. propose to use $u = 0.6$ and $v = 0.25$, since the multiplication with v evolves into a shift operation in this case [58].

Note that applying LLR to functional node updates results in a higher numerical stability, i. e., no multiplications are required. All approximations investigated in this thesis, are visualized in Figure 8.3.

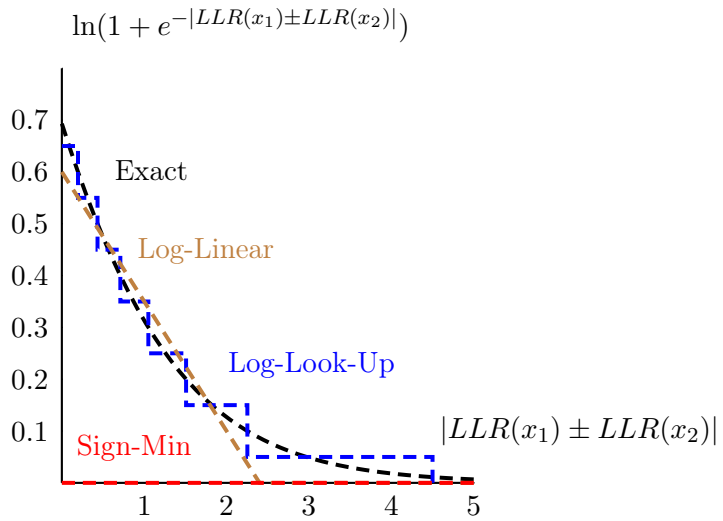

Figure 8.3.: Approximations of the Log Likelihood Ratios (LLR) for Embedding with LDGM Codes Based on BP According to [58].

9. Embedding Based on a Code Trellis

Our intention in this chapter is to sketch the basic idea of the Syndrome Trellis Codes (STC) as proposed by Filler et al. in [24]. This approach for minimizing the embedding impact based on a code trellis is able to handle arbitrary profiles of embedding impact.

Within the following section, we give some basic considerations on convolutional codes as a basis for the code trellis used within Syndrome Trellis Codes for embedding.

9.1. Basic Considerations

Note that binary convolutional codes, first introduced by Elias [18], are perhaps the most popular form of binary ECC. They process information serially or continuously and have numerous applications, e. g. in wireless communication.

Generally, the encoder has memory, i. e., the output symbols do not only depend on the input symbols but also on previous inputs. In other words, the encoder can be described as a sequential circuit or finite state machine, whereas the state is defined as the contents of the memory. In theory, the produced sequences have infinite end. In practice however, the state of the convolutional code is periodically forced after $l + h$ steps to state $\mathbf{0}$. Thus, code sequences are produced in a block-wise manner. The following description of the underlying basics of convolutional codes is based on the book of Klimant et al. [70].

Generally, a convolutional coder can be described by means of a *shift register* with linear interconnection. In Figure 9.1, an example for a convolutional coder is given according to [70]. The shift register has one input and m outputs. Furthermore, it is characterized by the memory constraint length h.

It is possible to formalize a shift register as a *deterministic automat* (dA), where the source code sequence $\mathbf{c}^* = (..., u(\tau), u(\tau{+}1), u(\tau{+}2)...)$ is transformed into the codeword $\mathbf{c} = (..., (v_1(\tau), v_2(\tau), ..., v_m(\tau)), (v_1(\tau+1), v_2(\tau+1), ..., v_m(\tau+1)), ...)$.

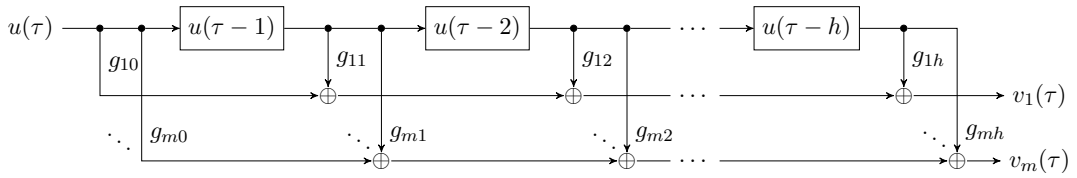

Figure 9.1.: Schematic Diagram of a Convolutional Coder Visualized as Shift Register According to [70].

Furthermore, the encoder can also be described by means of a *generator matrix* of dimension $m \times (h+1)$, whereas the shift register is a realization of $\mathbf{G}$:

$$\mathbf{G}_{m\times(h+1)} = \begin{pmatrix} g_{10} & g_{11} & g_{12} & \cdots & g_{1h} \\ g_{20} & g_{21} & g_{22} & \cdots & g_{2h} \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ g_{m0} & g_{m1} & g_{m2} & \cdots & g_{mh} \end{pmatrix}.$$

Note that the generator matrix is often denoted as m-tuple, where each row is given as an octal number[28]. By means of this generator, it is possible to calculate the code sequence $\mathbf{v}(\tau) = \{0,1\}^m$ at the moment τ:

$$\mathbf{v}(\tau) = \begin{pmatrix} \mathbf{v}_1(\tau) \\ \mathbf{v}_2(\tau) \\ \vdots \\ \mathbf{v}_m(\tau) \end{pmatrix} = \mathbf{G} \begin{pmatrix} \mathbf{u}(\tau) \\ \mathbf{u}(\tau-1) \\ \vdots \\ \mathbf{u}(\tau-h) \end{pmatrix}. \tag{9.1}$$

Generally, the behavior of such a convolutional coder can be described by means of a *encoder state diagram*. Note that a memory constraint length of h connotes 2^h states of the encoder. Within the encoder state diagram, also the allowed transitions including the associated input and output are visualized. An example is given below according to [70].

Example: Given the convolutional coder shown in Figure 9.2 with $h = 2$ states and $m = 2$ outputs, we find the associated generator matrix with $G_{2\times3} = \left(\begin{smallmatrix} 1 & 0 & 1 \\ 1 & 1 & 1 \end{smallmatrix}\right) = (5_8, 7_8)$.

The related encoder state diagram is given in Figure 9.3.

Based on this encoder state diagram, it is possible to describe the encoder by means of a *code trellis*, which contains the information of the state diagram, but

[28] The generator matrix $\left(\begin{smallmatrix} 0 & 0 & 1 \\ 1 & 0 & 1 \end{smallmatrix}\right)$, e. g., is given by $(1_8, 5_8)$.

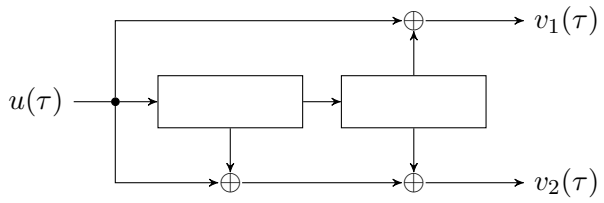

Figure 9.2.: Coder Example for Convolutional Codes According to [70].

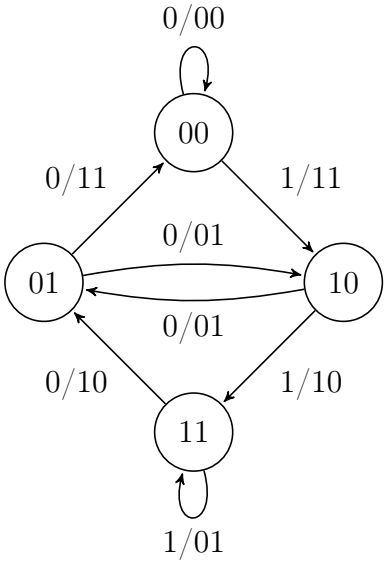

Figure 9.3.: Encoder State Diagram for Convolutional Codes According to [70].

also uses time as a horizontal axis to show the state transition paths. The code trellis for the encoder given in the example above is shown in Figure 9.4.

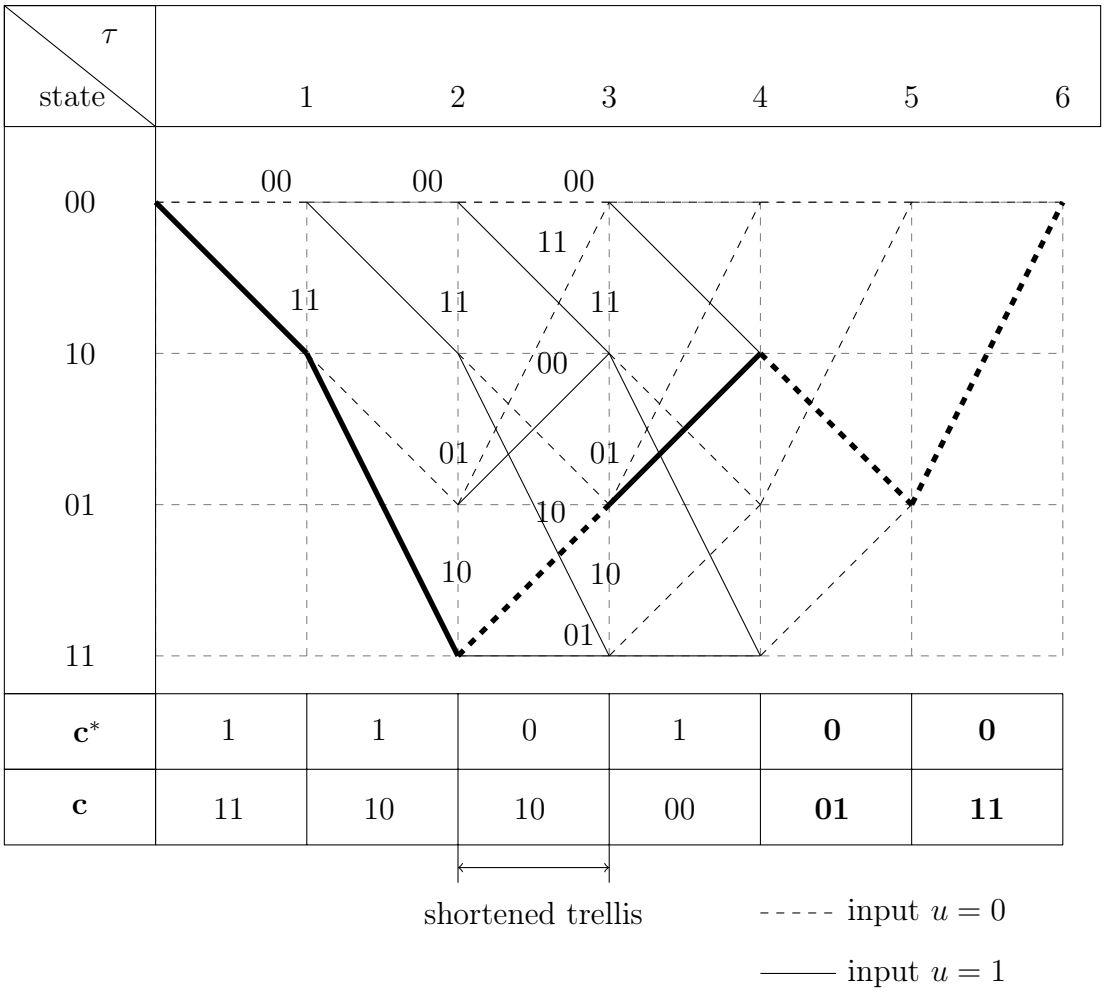

Figure 9.4.: Code Trellis According to [70].

The encoder states are given on the left hand side and the transitions are denoted as lines between the states[29]. Note that the transition paths specify the possible code words $\mathbf{c} \in \mathcal{C}$. Thus, using the code trellis, it is possible to achieve the related codeword $\mathbf{c}$ to arbitrary source codewords $\mathbf{c}^*$. Note that the *Viterbi algorithm* can be used for decoding only. However, based on the code trellis, it is possible to describe the coding as well as the decoding process.

An example for encoding is given in Figure 9.4. The labels related to each

[29]Note that the code trellis has to be read from left to right.

transition are determined by the output of the encoder. The code trellis recurs after $\tau = l$ clockings. In order to be able to describe the code for coding and decoding purposes, the shortened code trellis is sufficient.

As it is shown in Figure 9.4, the source codeword of length l is extended by h bit in order to get the shift register back to its initial state. This procedure is advantageous for decoding, since the decoder has to consider only paths stating and ending in the zero state.

For decoding by means of the Viterbi algorithm, the receiver compares the received sequence $\mathbf{b}$ with all possible paths through the Trellis, i. e. with all codewords. The sequence $\mathbf{c}$ with minimum distance to $\mathbf{b}$ is considered as corrected sequence $\mathbf{b}_{corr}$. For more information about the underlying basics of convolutional codes, coding and decoding principles, we refer to [70].

9.2. Syndrome Trellis Codes - Embedding Algorithm

A practical approach for syndrome coding to minimize the embedding impact based on a code trellis was presented by Filler et al. in [24]. The main advantage of this algorithm is its linear complexity and memory requirements w.r.t the number of cover elements. Thus, the application of large codeword length becomes feasible.

Note that this approach is carried out in the dual domain, i. e., the code is represented by its parity-check matrix $\mathbf{H}$ which is represented by a code trellis[30]. In this case, it is possible to determine a solution to Equation (3.33) in an elegant way[31].

As described in Section 3.5, the sender's goal is to minimize the distortion introduced during embedding. Therefore, a stego sequence $\mathbf{b}$ has to be determined according to (Equation (3.35)):

$$\mathbf{b} = \underset{\mathbf{b}\in\mathcal{C}(\mathbf{emb})}{\operatorname{argmin}}\, d_\rho(\mathbf{a}, \mathbf{b}). \tag{9.2}$$

Note that the parity-check matrix $\mathbf{H}_{k\times n}$ has a special form, allowing to represent every solution of $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$ as a path through the trellis. Filler et al. propose to determine $\mathbf{H}_{k\times n}$ by placing a small sub-matrix $\hat{\mathbf{H}}$ of size $h \times w$ next to each other and shifted down by one row. As a result, a sparse, banded matrix $\mathbf{H}_{k\times n}$ is achieved.

Note that the constraint length h affects the algorithm's speed and efficiency. Consequently, Filler et al. investigated $6 \leq h \leq 12$. The width of $\hat{\mathbf{H}}$ depends on the inverse relative message length $\alpha^{-1} = w$.

[30] In case of convolutional codes, the generator matrix $\mathbf{G}$ is represented by a code trellis instead.

[31] Note that the receiver determines the embedded message again by calculating the syndrome $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$.

An example of such a matrix construction according to [24] is given below with $k = 4$ and $n = 8$:

$$\hat{\mathbf{H}}_{h\times w} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \quad \mathbf{H}_{k\times n} = \begin{pmatrix} 1 & 0 & & & & & & \\ 1 & 1 & 1 & 0 & & & & \\ & & 1 & 1 & 1 & 0 & & \\ & & & & 1 & 1 & 1 & 0 \end{pmatrix}$$

Filler et al. also investigated how to design $\hat{\mathbf{H}}$. They found that a matrix $\hat{\mathbf{H}}$ optimized for one profile of embedding impact seems to be good for other profiles as well. Moreover, they investigated several matrices found through exhaustive search with some simple design rules. Filler et al. state, that good matrices consist of no identical columns and ones in the first and last rows. The remaining bits are assigned at random.

In order to embed, the sequence $\mathbf{a}$ has to be modified in sequence $\mathbf{b}$. In order to find the sequence $\mathbf{b}$ that minimizes the introduced distortion, the *Viterbi algorithm* is applied. Note that due to the form of the matrix, only the first w bits of $\mathbf{b}$ can effect the first bit of the message. Consequently, the stego sequence $\mathbf{b}$ within our example has to be chosen according to:

$$\begin{aligned} \mathbf{emb}[1] &= (\mathbf{H}[1,1]\,\mathbf{H}[1,2])(\mathbf{b}[1]\,\mathbf{b}[2])^T \\ \mathbf{emb}[2] &= (\mathbf{H}[2,1]\,\mathbf{H}[2,2]\,\mathbf{H}[2,3]\,\mathbf{H}[2,4]))(\mathbf{b}[1]\,\mathbf{b}[2]\,\mathbf{b}[3]\,\mathbf{b}[4])^T \\ \mathbf{emb}[3] &= (\mathbf{H}[3,3]\,\mathbf{H}[3,4]\,\mathbf{H}[3,5]\,\mathbf{H}[3,6]))(\mathbf{b}[3]\,\mathbf{b}[4]\,\mathbf{b}[5]\,\mathbf{b}[6])^T \\ \mathbf{emb}[4] &= (\mathbf{H}[4,5]\,\mathbf{H}[4,6]\,\mathbf{H}[4,7]\,\mathbf{H}[4,8]))(\mathbf{b}[5]\,\mathbf{b}[6]\,\mathbf{b}[7]\,\mathbf{b}[8])^T. \end{aligned} \tag{9.3}$$

Thus, it is possible to determine a solution to Equation (3.35) with low complexity based on a trellis since each $\mathbf{b}$ satisfying $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$ is represented as a path through the trellis. Consequently, the number of paths is exponential in n. The process of embedding consists of two steps: a forward and a backward part:

1. Construct the trellis dependent on $\mathbf{H}_{k\times n}$, $\mathbf{a}$ and $\mathbf{emb}$, and
2. Determine the closest codeword by means of the backward step.

Recall that each path in the trellis starts in the leftmost all-zero state. The edges represent adding ($\mathbf{b}[i] = 1$) or not adding ($\mathbf{b}[i] = 0$) the ith column of $\mathbf{H}_{k\times n}$ to the current partial syndrome, the states correspond to the partial syndromes.

A state in the trellis is reachable whenever there is a path connecting this state with the leftmost all-zero state. At the end of each block, all paths, for which the first bit of the partial syndrome does not match $\mathbf{emb}[i]$ are terminated.

In order to find the closest $\mathbf{b}$, weights are assigned to all trellis edges. Thus, the problem of finding the closest stego sequence is transformed to the problem of

finding the path with minimum weight through the trellis. During the backward path, it is easily possible to track back this path.

An example for the Syndrome Trellis Code is given below.

Example:

In this example, the sender wants to embed the confidential message $\mathbf{emb} = (1001)$ into the cover sequence $\mathbf{a} = (10011011)$, while a uniform profile is assumed.

In a first step, the parity-check matrix $\mathbf{H}_{k\times n}$ is designed for $k = 4$ and $n = 8$ with:

$$\hat{\mathbf{H}}_{h\times w} = \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix} \quad \mathbf{H}_{k\times n} = \begin{pmatrix} 1 & 0 & & & & & & \\ 1 & 1 & 1 & 0 & & & & \\ & & 1 & 1 & 1 & 0 & & \\ & & & & 1 & 1 & 1 & 0 \end{pmatrix}.$$

In order to determine the sequence $\mathbf{b}$ that minimizes $d_\rho(\mathbf{a}, \mathbf{b})$, we construct the code trellis in a first step. Within this example, we find a coder with 4 states: $s_{00}, s_{01}, s_{10}s_{11}$. Note that the Viterbi Algorithm is realized including breaks $p_1, \ldots, p_k$ for embedding the confidential message $\mathbf{emb}$ of length k.

1. Construct the trellis dependent on $\mathbf{H}_{k\times n}$, $\mathbf{a}$ and $\mathbf{emb}$ (see Figure 9.5)
   - Start with the all-zero syndrome, i. e., with state s_{00} and p_0
   - Two edges are drawn from $p_0 s_{00}$
     - The first dashed edge connects $p_0 s_{00}$ with $\mathbf{H}[., 1]s_{00}$ and corresponds to not adding the first column of $\mathbf{H}_{k\times n}$ to the state s_{00}
     - Assign a weight of 1 to this edge since it is related to $\mathbf{b}[1] = 0$ and $\mathbf{a}[1] = 1$, thus the introduced distance between cover sequence $\mathbf{a}$ and stego sequence $\mathbf{b}$ would be $d_\rho(\mathbf{a}, \mathbf{b}) = 1$[32]
     - The second solid edge connects $p_0 s_{00}$ with $\mathbf{H}[., 1]s_{11}$ and corresponds to adding the first column of $\mathbf{H}_{k\times n}$ to the state s_{00}
     - Assign a weight of 0 to this edge since it is related to $\mathbf{b}[1] = 1$ and $\mathbf{a}[1] = 1$, i. e., no distortion would be introduced and $d_\rho(\mathbf{a}, \mathbf{b}) = 0$
   - Connect the states related to column $\mathbf{H}[., 1]$ and $\mathbf{H}[., 2]$
     - Two edges are drawn from $\mathbf{H}[., 1]s_{00}$ to $\mathbf{H}[., 2]s_{00}$ and $\mathbf{H}[., 2]s_{10}$, weighted with 1 and 2 respectively
     - Two edges are drawn from state $\mathbf{H}[., 1]s_{11}$ to $\mathbf{H}[., 2]s_{01}$ and $\mathbf{H}[., 2]s_{11}$, weighted with 1 and 0 respectively

[32] Note that the assignment of 1 is true considering a uniform profile of embedding impact. In case of a general profile, i. e., in case wet elements should be excluded during the embedding process, we have to assign the embedding impact ρ whenever $\mathbf{a} \neq \mathbf{b}$.

- Set the first message bit $\mathbf{emb}[1] = 1$
  - Discard the states s_{00} and s_{10} since their least significant bit is 0
  - No edges come out of these states
- Shift the trellis from s_{01} to s_{00} and from s_{11} to s_{01}
  - Dotted edge between $\mathbf{H}[.,2]s_{01}$ and $p_1 s_{00}$
  - Dotted edge between $\mathbf{H}[.,2]s_{11}$ and $p_1 s_{01}$
- Construct column $\mathbf{H}[.,3]$ identically to column $\mathbf{H}[.,1]$ and $\mathbf{H}[.,2]$
- Construct column $\mathbf{H}[.,4]$
  - Find two incoming edges into each state
  - Eliminate the edge with higher weight, ties can be resolved arbitrarily
  - The surviving edge is denoted in bold
- Construct the remaining columns
- In the last portion of the trellis, there are only two states in each column due to cropping $\mathbf{H}_{k\times n}$ in the last $h-1$ sections[33]

2. Determine the path with minimum weight using the backward step
   - Go back from the right-most state
   - Use the surviving edges and construct $\mathbf{b}$ (denoted red in Figure 9.5)
   - The sender finds $\mathbf{b} = (10110011)$ with $d_\rho(\mathbf{a}, \mathbf{b}) = 2$.

The receiver extracts the message with: $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$

$$\begin{pmatrix} 1 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} 1 & 0 & & & & & & \\ 1 & 1 & 1 & 0 & & & & \\ & & 1 & 1 & 1 & 0 & & \\ & & & & 1 & 1 & 1 & 0 \end{pmatrix} \begin{pmatrix} 1 \\ 0 \\ 1 \\ 1 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}.$$

■

[33] The edges from $\mathbf{H}[.,7]$ to $\mathbf{H}[.,8]$ are solid, they correspond to adding the zero-column to the syndrome.

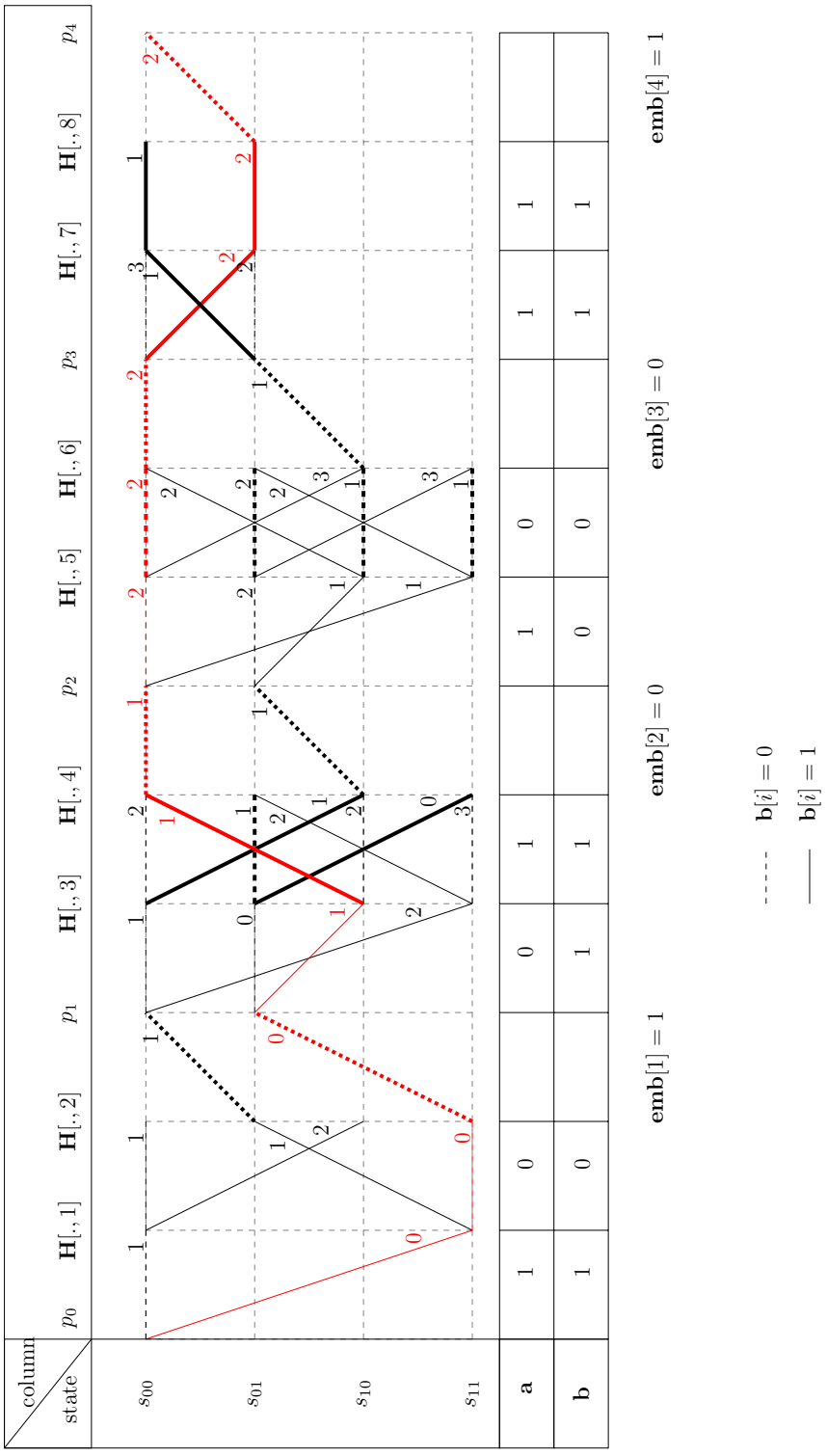

Figure 9.5.: Example for Syndrome Trellis Code Based Embedding.

10. Evaluation of the Algorithms for a Large Codeword Length

As we did in Chapter 6, we will compare the different approaches for a large codeword length presented in this thesis:

- Embedding Based on Wet Paper Codes,
- Embedding Based on LDGM Codes, and
- Embedding Based on Syndrome Trellis Codes.

Therefore, we consider the following properties of a steganographic scheme (Section 2.1.3):

- the security,
- the capacity,
- the success of embedding, and
- the embedding complexity.

Recall that these are competing goals. It is not possible to maximizing the capacity as well as the security of an algorithm. Moreover, the goal is to minimize the embedding complexity while maximizing the success rate.

Since the weighting of the properties depends on the individual application, we do not search for a scalar metric aggregating all these properties. Instead, we evaluate the presented algorithms according to the properties mentioned above in an isolated manner.

10.1. Evaluation Concerning the Security

In this thesis, we measure the security of a steganographic scheme by means of its achieved embedding efficiency $e = k/d_\rho(\mathbf{a}, \mathbf{b})$. Since we consider either a uniform profile or a general profile, where parts of the cover are denoted as wet and excluded during embedding (Wet Paper Steganography), $d_\rho(\mathbf{a}, \mathbf{b})$ can be easily measured using the average number of embedding changes R_a.

In the following, we give some detailed results for the algorithms presented in the previous chapters as well as a comparison between them.

10.1.1. Wet Paper Codes

Within this section, we take a closer look at the algorithms described in Chapter 7, namely Wet Paper Codes, Wet Paper Codes combined with Gaussian elimination or combined with the Matrix LT Process. These algorithms, proposed by Fridrich et al., have in common the fact that they do not search for a solution to $\mathbf{emb} = \mathbf{H}_{k\times n}\,\mathbf{b}^T$ (Equation (3.33)) that minimizes the introduced distortion $d_\rho(\mathbf{a}, \mathbf{b})$. Instead, they simply solve the system of linear equations. Consequently, they are not able to achieve a maximized embedding efficiency.

According to Fridrich et al., we find the embedding rate always close to 2 bits per embedding change for embedding with Wet Paper Codes combined with Gaussian Elimination [44]. Thus, we find the embedding efficiency with $e \approx 2$, which is comparable to LSB embedding.

Furthermore, Fridrich et al., state that the solution obtained by solving the system of linear equations with the Matrix LT Process will have on average 50% of ones for arbitrary messages. This results in 50% embedding changes and thus, the message will be also embedded with $e \approx 2$.

Notwithstanding the dissatisfying results in terms of the embedding efficiency, Böhme pointed out another security related pitfall in his Rump Session Talk at the Information Hiding Workshop [7]. Böhme found that for Wet Paper Codes with Matrix LT Process, 25% of the samples where never altered during embedding considering a fixed sub-matrix $\mathbf{H}_{k\times|\mathrm{Dry}|}$ and matrices following the robust soliton distribution. Thus, the positions of changes are predictable whenever no wet elements are within the cover.

Considering wet elements, Böhme found a bimodal characteristic in the probability distribution. Note that neither permuting rows or column nor different matrix parameters affect this distribution. Thus, he summarized that it has to be the process of solving the system of linear equations itself that causes this distribution. Consequently, it seems reasonable to consider a pseudo-random path through the cover whenever embedding is based on Wet Paper Codes with Matrix LT Process. Note that this problem can not be observed for Embedding based on LDGM Codes.

10.1.2. LDGM Codes

The application of LDGM Codes in steganography was independently proposed by Fridrich and Filler based on Survey Propagation (SP) [19, 37] and Günther, Schönfeld and Winkler based on Belief Propagation (BP) [58, 60].

As stated in Chapter 8, embedding based on BP offers several possibilities to reduce the embedding complexity. Generally, the BP algorithm is known as considerably less complex special case of the SP algorithm introduced in [94]. Additional approximations are considered within our thesis.

In our simulations [58, 60], every data point is obtained by averaging over

100 trials. We set the parameters described in Section 8.2.1 to $r_{max} = 1\%$, $r_{min} = 0.1\%$, and $\beta_{th} = 0.8$.

Since the optimal value for γ_j depends on the embedding rate α, we experimentally determined the following appropriate values [60]: $\gamma_j^{1/\alpha=1.59} = 1.0$, $\gamma_j^{1/\alpha=2.0} = 1.13$, $\gamma_j^{1/\alpha=2.86} = 1.4$, $\gamma_j^{1/\alpha=3.45} = 1.54$, and $\gamma_j^{1/\alpha=4.0} = 1.65$; $j = 1, \ldots, N$[34].

The results of our simulations in terms of embedding efficiency can be seen in Figure 10.1 for $n = N = 10^4$ and $T = 60$. Within this figure, the embedding efficiency for the different approaches is given for several inverse embedding rates α^{-1}. The higher the inverse relative message length, the lower the percentage of cover elements used for embedding. Again, a high embedding efficiency is linked to a more secure embedding scheme.

Within Figure 10.1, we compare the different approximations for embedding based on Belief Propagation (BP) as presented in Section 8.2.4 as well as the approach for embedding based on Survey Propagation (SP) as presented in [37].

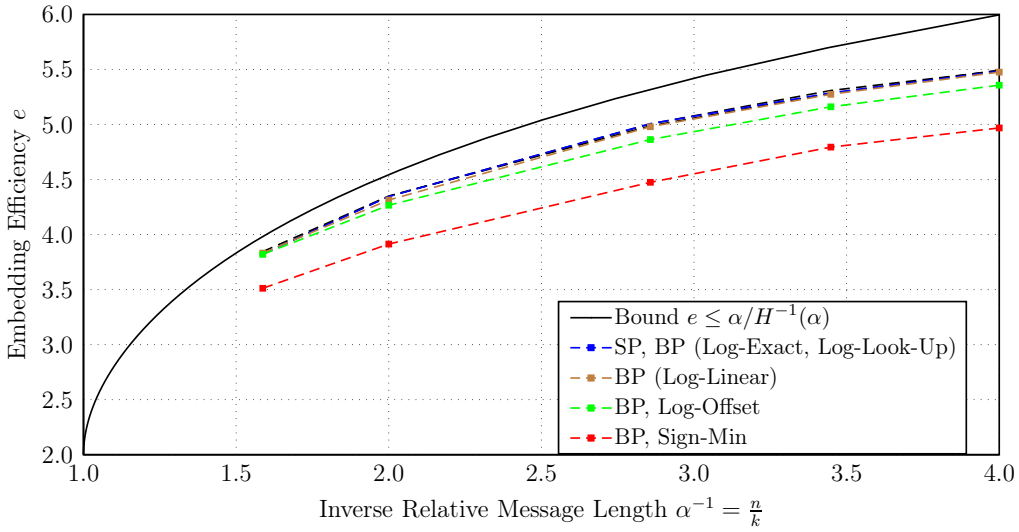

Figure 10.1.: Embedding Efficiency e for LDGM Codes with BP, $n = N = 10^4$ and $T = 60$ According to [58].

Regarding Figure 10.1, the approaches for BP based on an exact evaluation of Term (8.8) (Log-Exact), based on a Look-up Table (Log-Look-Up), and based on a linear approximation (Log-Linear) are very similar to those of SP [37]. Note that by means of LDGM Codes, we achieve results relative close to the upper bound for a wide range of α^{-1} for these approaches.

Whenever we consider the approach Log-Offset, we find a small derivation for the embedding efficiency e compared to the other approaches. However, with this

[34]Note that the parameter γ_j reflects how strongly the algorithm should try to preserve the value of y_j (Section 8.2.1).

approach, the evaluation of Term (8.8) is trivial and thus, it is possible to embed with a low complexity. Whenever we apply the Sig-Min approach, i. e., ignoring Term (8.8), we still find good results in terms of embedding efficiency.

Note that embedding based on LDGM Codes is based on an iterative approximation of the coset leader. Thus, the algorithm may stop, whenever a solution close to the coset leader is determined (see Figure 8.2). Consequently, the maximum possible embedding efficiency can only be achieved whenever the algorithm determines the coset leader.

Within the work of Günther, Schönfeld and Winkler [58, 59], the influence of the number of iterations on the embedding efficiency is also analyzed exemplarily for $\alpha^{-1} = 2$. The results according to [58, 59] are given in Figure 10.2.

We investigate a reinitialization of the messages on the one hand, i. e., the variable nodes $c_1^*, \ldots, c_l^*$ were initialized with random values at the beginning of the embedding process and after every decimation step.

On the other hand, the variable nodes were not reinitialized after the decimation step. Instead, the remaining variable nodes were updated based on the messages they received from the functional nodes in the last iteration before decimation. Note that reusing the message in the next iteration enables the sender to reach the maximum achievable embedding efficiency already with $T = 30$ iterations between two decimation rounds. Otherwise, ≈ 60 iterations are required.

Additionally, Günther et al. investigated the influence of the cover length N [58, 59]. They found that with increasing cover length, the codes are able to converge to the upper boundary on embedding efficiency without reaching it.

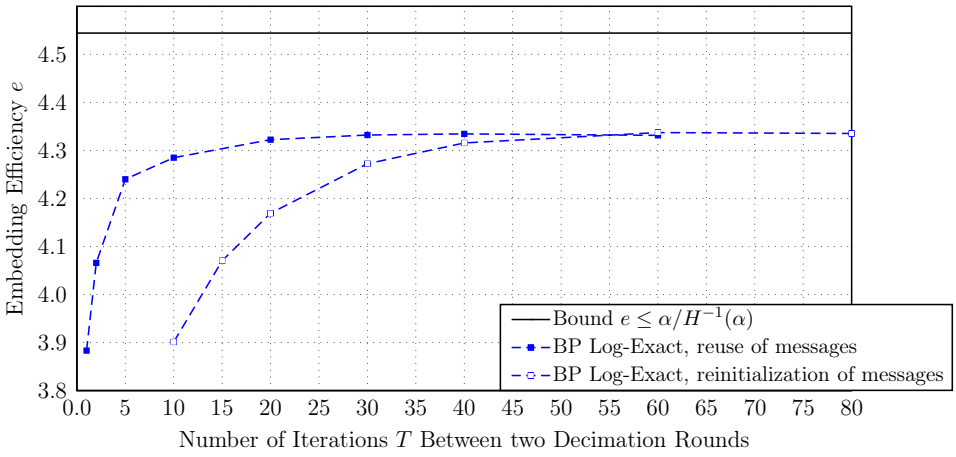

Figure 10.2.: Dependency of the Embedding Efficiency on the Number of Performed Iterations between two Decimation Rounds for LDGM Codes with BP ($\alpha = 0.5$, $n = N = 10^4$) According to [59].

10.1.3. Syndrome Trellis Codes

Embedding based on Syndrome Trellis Codes (STC) as proposed by Filler et al. and described in Chapter 9, can be seen as an efficient algorithm for embedding with an inverse relative message length of $\alpha^{-1} \geq 2$. Note that according to the square root law, the inverse relative message length must increase with increasing size of the cover in order to maintain the same level of security [26]. Thus, the secure inverse relative message length in steganographic schemes should be always above 2.

The width of $\hat{\mathbf{H}}_{h\times w}$ depends on the inverse relative message length $\alpha^{-1} = w$ while the constraint height h affects the algorithm's speed and efficiency. As stated by Filler et al., it is also possible to realize a higher relative message length.

Filler et al. investigated Syndrome Trellis Codes for several values for the constraint height h with $h = 7, \ldots, 12$ and several inverse relative message length α^{-1}. The results are illustrated in Figure 10.3 according to [25].

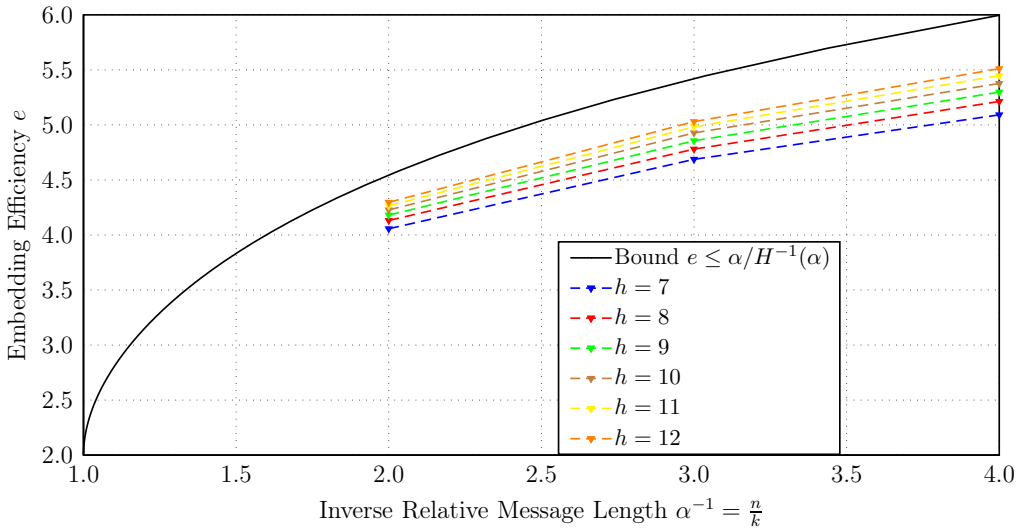

Figure 10.3.: Embedding Efficiency e for STC Codes for Several Constraint Heights h According to [25].

As it can be seen in Figure 10.3, the embedding efficiency increases with increasing constraint height h. For $h = 12$, results close to the upper boundary on embedding efficiency can be achieved.

Filler et al. state that the bound can be theoretically achieved by increasing the constraint height. However, the embedding complexity increases exponentially since the encoder can have 2^h states. Consequently, the constraint height is limited by the computational power of the sender.

10.1.4. Comparing the Approaches

In this section, we will give a short comparison in terms of embedding efficiency for the following two approaches: embedding based on LDGM Codes and embedding based on Syndrome Trellis Codes. Note that the approach embedding based on Wet Paper Codes is not comparable since its goal is not to maximize the embedding efficiency (see Section 10.1.1).

As visualized in Figure 10.1, the sender is able to achieve results close to the upper boundary on embedding efficiency by means of embedding based on LDGM Codes. Note that embedding based on LDGM Codes combined with Belief Propagation achieve the same results in terms of embedding efficiency as LDGM Codes combined with Survey Propagation. However, the former approach is less complex.

Even if embedding based on Syndrome Trellis Codes is less complex, the results presented in Section 10.1.3 are slightly worse than those of the LDGM Codes. A comparison between both approaches is visualized in Figure 10.4.

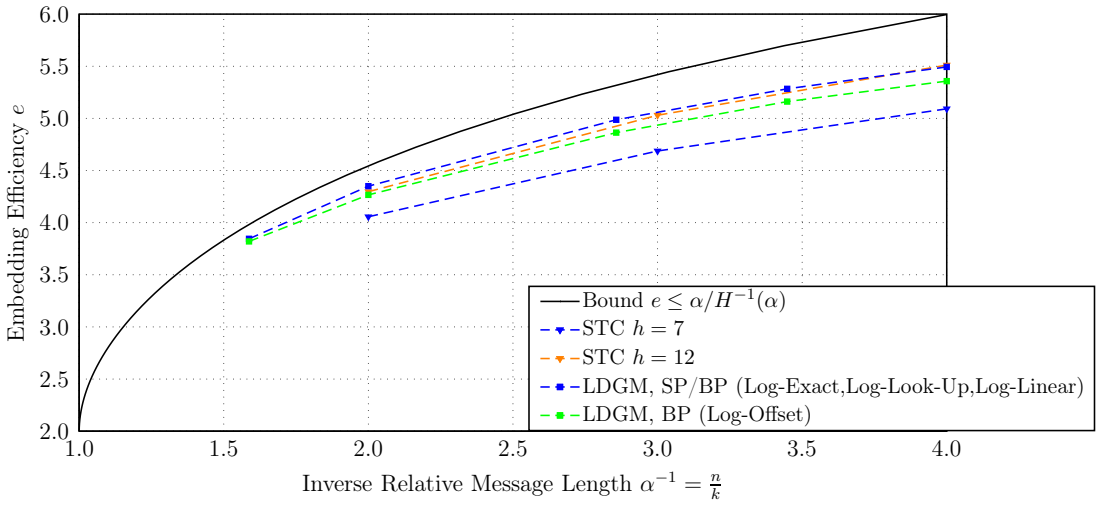

Figure 10.4.: Embedding Efficiency e - A Comparison.

As we can see, all approaches are able to cover a wide range of α^{-1} and achieve results close to the bound. However, only STC with $h = 12$ achieve comparable results to those of embedding based on LDGM Codes.

10.2. Evaluation Concerning the Capacity

Within this section, we would like to compare the approaches considering their capacity. In this thesis, we express the capacity as relative message length $\alpha = \frac{|\mathbf{m}|}{N} = \frac{k}{n}$, i. e., the ratio between the maximum length of a secret message (in bit) and the number of elements of the cover sequence.

For the approaches for a large codeword length, we find $n = N$. Note that the maximum length of the embeddable message depends on the parameter k and the block length n with $k = n - l$.

However, since the approaches based on Wet Paper Codes (Chapter 7) and on Syndrome Trellis Codes (Chapter 9) are centered on stochastic matrices or stochastic matrices with constraints, a non-negligible probability of failure exists. This results in a decrease of the capacity due to the need to store the actual message length in a segment of the cover not usable for embedding.

Second, a coding loss has to be considered since the message length is reduced within each embedding trial in case of non-solvability. Within the following section, we will have a deeper look at this property.

Note that for embedding based on LDGM Codes (Chapter 8) we do not have to deal with a reduced capacity even if the approach is also based on stochastic matrices with constraints. Since this approach is based on compression, we are always able to find a solution. Thus, we do not have to deal with non-solvability. Based on this, we do not have to deal with either a capacity reduction to store the actual message length or a coding loss due to several embedding trials.

10.3. Evaluation Concerning the Success Rate

Within this Section, we evaluate the probability of success for embedding an arbitrary message of length k within the cover of length $n = N$.

Recall the considerations concerning the success of embedding within one block $p_{\mathbf{so}}$ but also depending on the distribution of wet elements for approaches considering small blocks (see Section 6.3). For approaches applicable for a large codeword length, i. e., in case $n = N$, we have to consider only the solvability $p_{\mathbf{so}}$.

Note that the success of embedding $p_{\mathbf{so}}$ depends on the property of the code and affects the complexity of the whole embedding step whenever several embedding trials are required.

Furthermore, as stated in Section 10.2, the success for embedding an arbitrary message of length k within the cover of length $n = N$ is always 1 considering embedding based on LDGM Codes since this approach is based on compression.

Thus, in this section, we discuss the results concerning the success of embedding for the approaches based on Wet Paper Codes as well as based on Syndrome Trellis Codes.

10.3.1. Wet Paper Codes

In order to evaluate the success of embedding, we need to determine the probability with which the sender is able to solve the system of linear equations $\mathbf{emb} = \mathbf{H}_{k\times n}\mathbf{b}^T$. Note that for embedding based on Wet Paper Codes, the columns related to wet elements are removed before solving (see Step 2, Section 7.1). Consequently, to determine $p_{\mathbf{so}}$, we need to evaluate the probability that matrix $\mathbf{H}_{k\times|\mathrm{Dry}|}$ has full rank, i. e., that $\mathbf{H}_{k\times|\mathrm{Dry}|}$ consists of k linearly independent rows.

Based on Equation (3.4), we find the success of embedding $p_{\mathbf{so}}$ with:

$$p_{\mathbf{so}} = \prod_{i=0}^{k-1}(1 - 2^{i-|\mathrm{Dry}|}). \tag{10.1}$$

Generally, it is possible to assume $k = |\mathrm{Dry}|$ for Wet Paper Codes. Thus, we have to consider Equation (10.1) with $k = |\mathrm{Dry}|$ in order to determine $p_{\mathbf{so}}$ in this case. We visualized $p_{\mathbf{so}}$, i. e., the probability that a randomly generated matrix has full rank, for several values of k with $k = 1, \ldots, 50$ in Figure 10.5.

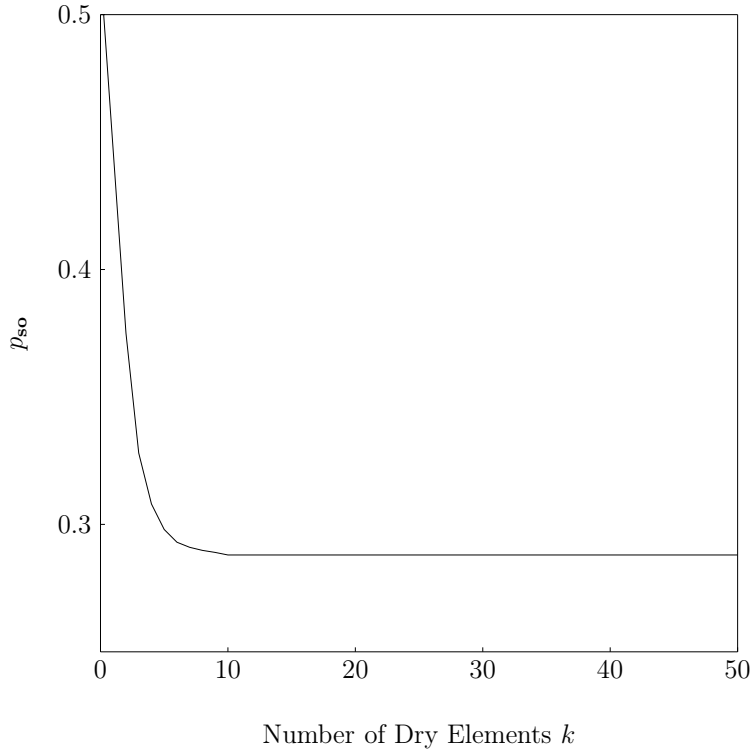

Figure 10.5.: Solvability $p_{\mathbf{so}}$ of Wet Paper Codes Dependent on k for $k = |\mathrm{Dry}|$.

As it can be seen in Figure 10.5, the solvability for large stochastic matrices $\mathbf{H}_{k\times|\mathrm{Dry}|}$ is low whenever we want to embed a message of length $k = |\mathrm{Dry}|$. We find $p_{\mathbf{so}}$ with 0.5 for a binary 1×1 matrix. With increasing k, the solvability $p_{\mathbf{so}}$ is reduced considerably and stays constant at 0.288.

Thus, due to the non-negligible probability of a stochastic matrix $\mathbf{H}_{k\times|\mathrm{Dry}|}$ of not being of rank k (see Section 3.2.1), the approaches based on stochastic matrices are only able to communicate $|\mathbf{emb}| \leq k$ message bits. Thus, there exists the necessity of reducing the actual message length to $q \leq k$ due to non-solvability.

In this case, $\mathbf{H}_{k\times|\mathrm{Dry}|}$ becomes a $\mathbf{H}_{q\times|\mathrm{Dry}|}$ matrix with $q \leq k$ and thus, $q \leq |\mathrm{Dry}|$. The probability of this matrix being of rank q has to be determined based on Equation (3.4).

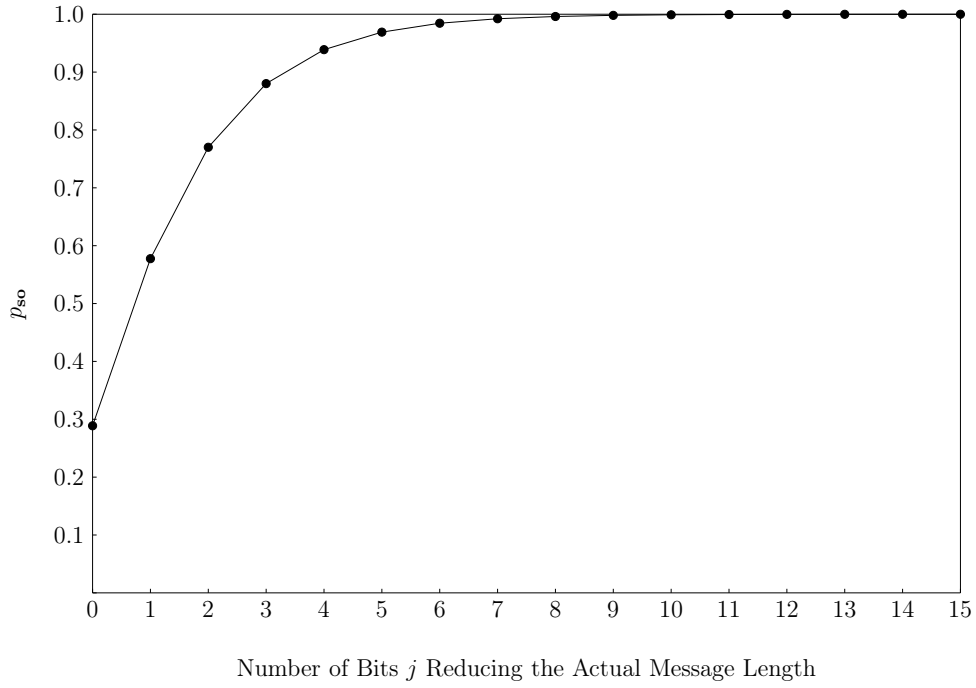

Figure 10.6.: Solvability $p_{\mathbf{so}}$ of Wet Paper Codes with $|\mathrm{Dry}| = k = 10^4$, $q = k - j$ and $j = 1, \ldots, 15$.

An example for $|\mathrm{Dry}| = 10^4$ is visualized in Figure 10.6. We give the solvability $p_{\mathbf{so}}$ for different numbers of bits reducing the actual message length. Figure 10.6 visualizes $|\mathrm{Dry}| = k = 10^4$, $q = k - j$ and $j = 1, \ldots, 15$.

As it can be seen in Figure 10.6, reducing the actual message length to $q \leq k$ significantly increases the solvability $p_{\mathbf{so}}$. In this example, we find a reduction to $q = k - 10$ sufficient to increase $p_{\mathbf{so}}$ to 0.999.

Note that Equation (10.1) is only valid for stochastic matrices. Thus, in the case of Wet Paper Codes combined with the Matrix LT Process, different considerations apply. Note that the Matrix LT Process will fail if at some point the algorithm fails to find a row with only one 1 [43]. Thus, we have to deal with a slight capacity loss applying this faster process of solving the system of linear equations.

According to Fridrich et al., the loss can be made small depending on the choice of the parameters used for generating $\mathbf{H}_{k\times n}$, i. e., the parameters initializing the RSD distribution. Note that the capacity loss decreases with increasing k. In their paper, Fridrich et al. report the capacity loss according to Table 10.1. Thus, considering a message length of 10% of the cover size ($k = 10^4$, $n = N = 10^5$), we find a capacity loss of 6.2% [46].

Table 10.1.: Capacity Loss for Wet Paper Codes based on the Matrix LT Process According to [46].

k	1000	10000	30000	100000
Percentage of a Successful Path	43	75	82	90
Capacity Loss	0.098	0.062	0.047	0.033

10.3.2. Syndrome Trellis Codes

Filler et al. investigated several parameters for the constraint height h with $h = 6, \ldots, 12$ for several inverse relative message length $\alpha^{-1} = w$ with $w = 2, \ldots, 20$ [24, 25]. A set of codes related to these parameters is given in [24, Table 1].

Note that the non-negligible probability of not being able to embed depends on the inverse relative message length and the properties of the code. Figure 10.7 visualizes the coding loss due to non-solvability depending on the inverse relative message length according to [25].

As can be seen in Figure 10.7, the coding loss rises with increasing inverse relative message length α^{-1}. For example, for $\alpha^{-1} = 2$ and $h = 8$ the coding loss is about 6%. For $h = 8$ and $\alpha^{-1} = 4$, we find a loss of about 10%.

Furthermore, a decreasing coding loss is reported for an increasing constraint length h. We find, e. g., for $\alpha^{-1} = 2$ and $h = 8$ a coding loss of about 6% and for $h = 12$ a coding loss of about 4%. Filler et al. state that by increasing the constraint length, we can theoretically make the coding loss approach 0. However, an increased constraint height directly results in an increased complexity as the number of states in the trellis is defined as 2^h. Consequently, Filler et al. find $h = 12$ to be reasonable in practice.

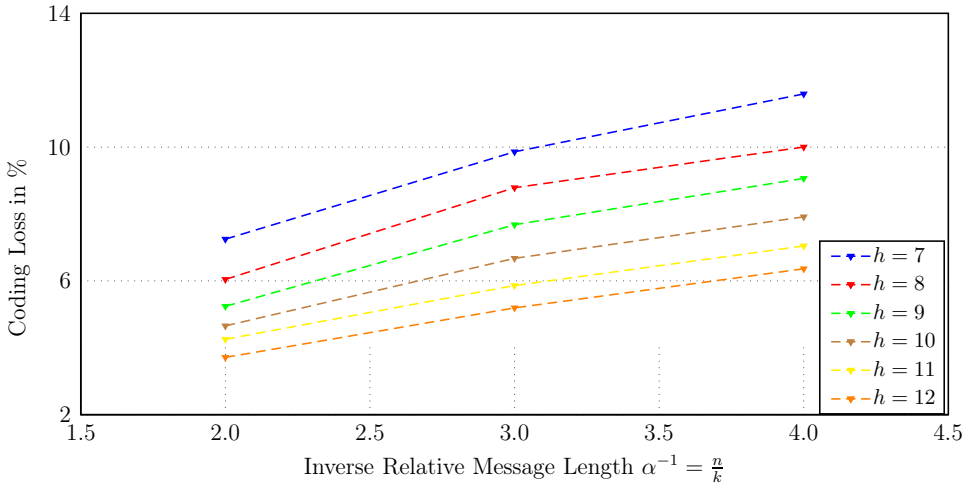

Figure 10.7.: Coding Loss for Syndrome Trellis Codes According to [25].

To minimize the coding loss, Filler et al. propose to use a pseudo-random path through the cover. The homogeneous distribution of wet elements within the cover makes the Viterbi algorithm less likely to fail [24]. The message size can be used as input for a Pseudo Random Number Generator (PRNG). Whenever the embedding process fails, k is reduced by one. Note that embedding $k-1$ bits leads to a different permutation.

10.4. Evaluation Concerning the Embedding Complexity

Within this section, we investigate the complexity of embedding for the algorithms based on a large codeword length presented and discussed in this thesis. As we did in Section 6.4, we consider complexity in terms of time complexity and in terms of memory complexity.

Note that a requirement for a steganographic algorithm is the possibility to embed in an acceptable time depending on the application. While it seems relatively uncritical for offline approaches, steganography in live-time environments is more crucial. Furthermore, the memory is limited by the physical properties of the computer used for determining the stego sequence.

10.4.1. Wet Paper Codes

Considering embedding based on Wet Paper Codes, the sender needs to solve a system of linear equations for k unknowns in binary arithmetic.

For Wet Paper Codes combined with Gaussian Elimination (Section 7.1), Fridrich et al. report a complexity of $O(k^3)$ [43]. This embedding complexity is of course impractical for a low inverse relative message length α^{-1}, i. e., for $k > 10^5$.

The complexity of embedding lies on the sender's side because he has to solve $q <= k$ linear equations with q unknown parameters. As a solution to reduce the complexity of embedding, several approaches have been proposed by Fridrich et al., e. g., the application of Structured Gaussian Elimination or the application of sparse matrices.

For Wet Paper Codes combined with Structured Gaussian Elimination (Section 7.2), the complexity of embedding is linear in the number of cover elements, however, with a large multiplicative constant. While the complexity is reduced by factor n_B^3, the number of solvings increases n_B-times, where n_B gives the number of pseudo-random disjoint subsets. Thus, the total performance is increased by factor n_B^2. Fridrich et al. find it advantageous to choose |Dry| as 200-500 elements in each subset to achieve a good performance [42].

Furthermore, Fridrich et al. considered several solvers such as those of Lanczos [74] and Wiedemann [99] for sparse matrices (Section 7.3). However, since in this application the matrix $\mathbf{H}_{k\times n}$ is rectangular and may be singular, the application of both methods is complicated, i. e., the design of the matrix slows down the solver.

Applying Wiedemann or Lanczos, the complexity of embedding is proportional to $k(k + \mathrm{w}k)(\log k)^c$, where w is the average number of ones in each row of $\mathbf{H}_{k\times n}$ and c is a small positive constant.

Fridrich et al. report in their paper the average running time for Wiedemann, Lanczos and Gauss for $k = 250, 500, 1000, 2000, 5000, 10000, 20000, 30000$ [42]. They found that Gaussian Elimination is best for $k = \{250 - 5000\}$. Wiedemann is best only for large payloads such as $k = \{10000, 20000\}$ [42].

Note that for embedding based on Wet Paper Codes, no storage space is required. However, making use of the Lanczos solver, tables of size $4 \times 8 \times 2^r$ need to be stored considering $GF(2^r)$. The best results were achieved for $r = 14$, resulting in a cache size of 2^9kByte.

Another approach to speed up the process of solving Equation (3.33) is the application of principles from Luby Transform Codes (LT Codes). Fridrich et al. propose to speed up the process of solving by making the matrix $\mathbf{H}_{k\times n}$ sparse and apply the Matrix LT Process (Section 7.4).

Since the density of ones in $\mathbf{H}_{k\times n}$ is $O(\ln(k/\delta)/k)$, the average number of operations required to complete the LT process is $O(n\ln(k/\delta) + (k\ln(k/\delta)) = O(n\ln(k/\delta))$ assuming the maximal-length message is sent ($q \approx k$) [43]. The first term stands for calculating $\mathbf{H}_{k\times n}\mathbf{a}^T$, while the second term gives the complexity

of the Matrix LT Process.

Note that the Matrix LT Process enables to solve Equation (3.33) as a whole at once which greatly simplifies the implementation and decreases the computational complexity. Thus, embedding based on Wet Paper Codes combined with the Matrix LT Process is significantly faster than Gaussian Elimination.

Fridrich et al. report a runtime of 0.023 seconds for $k = 1000$ for Gaussian Elimination and a runtime of 0.008 seconds for the Matrix LT Process. For $k = 100000$, they found a runtime of 9320 seconds for Gaussian Elimination and a runtime of 3.1 seconds for the Matrix LT Process [46].

10.4.2. LDGM Codes

In this section, we compare the different approximations presented in Section 8.2.4 to each other and to the approach presented in [37].

For a constant inverse relative message length α^{-1}, the number of nodes grows linearly with increasing block length n. Thus, the BP algorithm works with linear complexity $O(n)$. The effort for updating one node depends on the number of its neighbors. The more neighbors it possesses, the more messages need to be calculated in every iteration.

Generally, the number of neighbors of the functional nodes $f_1, \ldots, f_n$ and the variable nodes $c_1^*, \ldots, c_l^*$ depend on the row and column weights of $\mathbf{G}_{l \times n}$.

LDGM Codes can be classified according to their row and column weight distributions. If row and column weights are uniform, we speak about a regular LDGM Code, otherwise it is called irregular. Similar to [37], we use distributions from [75] that are optimized for the Binary Symmetric Channel (BSC) with an irregular functional node distribution and a variable node distribution as uniform as possible.

In Figure 10.8, we compare the update complexity for the different approximations in terms of the number of operations per node update. We assume that additions, comparisons and shifts all require the same computational effort, the evaluation of sign and absolute value are neglected.

According to Figure 10.8, unlike SP, our approach based on BP with LLR does not need any multiplications or divisions to update nodes. Notwithstanding the approximation, for updating the variable nodes the effort is at least 100 times smaller than for SP. Furthermore, the update complexity of our approach does not grow with increasing irregularity. That makes the LLR based algorithm especially interesting for embedding.

A comparison between SP and the linear approximation shows that the number of operations at the functional nodes can be reduced by 81% without loss of performance.

If the Term (8.8) is completely neglected as with the Sign-Min approach, updating the functional nodes becomes even faster. Compared to Log-Linear, only

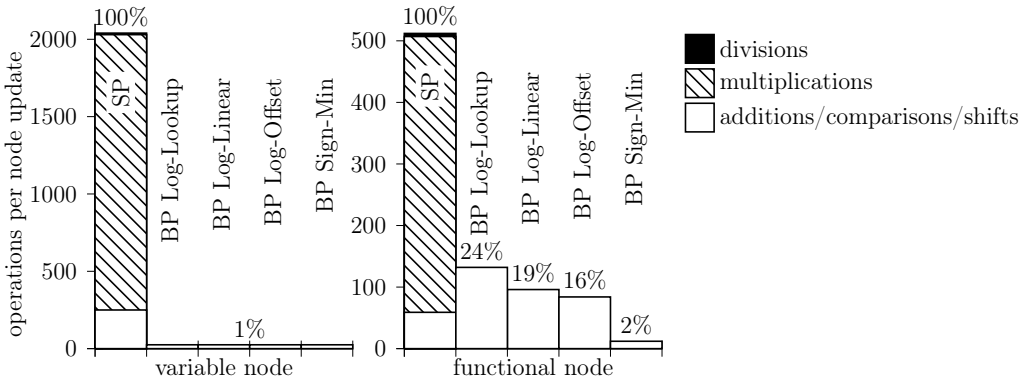

Figure 10.8.: Required Average Number of Operations (Multiplication, Division, Addition, Comparison, Shift) for the Update of One Variable Node c_i^* and One Functional Node for an LDGM Code, $\alpha^{-1} = 2$ According to [60].

12 instead of 96 operations are required. Concurrently, the speed-up comes along with a loss in embedding efficiency of around 0.43 at $\alpha^{-1} = 2$. The approximation with a simple constant (Log-Offset) can improve the performance considerably compared to Sign-Min. But it still performs worse than Log-Linear at nearly the same computational effort.

Note that the number of iterations $\mathbf{T}$ between two decimation rounds is a parameter with linear impact on the embedding speed.

10.4.3. Syndrome Trellis Codes

As already mentioned in Section 9, the transformation of the problem in the dual domain is preferable for a small relative message length since usually the Viterbi algorithm grows exponentially with decreasing $\alpha^{-1} = w$.

Note that the Viterbi algorithm has a time and space complexity of $O(2^h n)$, whereas the forward part requires $n + b$ steps [24]. Thus, we find the complexity linear depending on the block length $n = N$ but exponential depending on the constraint height h. Considering for example $n = N = 10^6$ and $h = 10$, the required memory complexity is $2^{10}10^6/8bytes \approx 122MB$.

Consequently, it seems to be reasonable to apply codes with a small constraint height h. However, these codes are not able to achieve good results in terms of the embedding efficiency (see Section 10.1.3). Generally, we have to find a trade-off between time complexity and performance in terms of embedding efficiency.

In order to speed up the search for the coset leader, Filler et al. propose to vectorize the calculations. In the forward step, we need to store only one bit corresponding to the label of the incoming edge in order to be able to reconstruct the path in the backward run [24]. Thus, a runtime of $1-5$ seconds for processing

$n = N = 10^6$ elements is reported for $h = 6, \ldots, 12$ and $\alpha^{-1} = w, w = 2, \ldots, 20$ [24].

10.5. Summary

The evaluation of the algorithms for syndrome coding based on matrices with a large code dimension was carried out according to the parameters security, capacity, success of embedding and complexity. The results are summarized in Table 10.2.

Table 10.2.: Evaluation of Embedding Algorithms Based on a Large Code Word Length - A Comparison.

	security	capacity	success of embedding	complexity
Wet Paper Codes	−	−	−	+
LDGM Codes	+	+	+	−
STC	+	−	−	+

- Considering the *security*, i. e., the embedding efficiency of the algorithms, we found that the approach embedding based on Wet Paper Codes is not able to maximize the embedding efficiency.

  Furthermore, Böhme found a bimodal characteristic in the probability distribution considering embedding based on the Matrix LT Process [7]. Consequently, it seems reasonable to consider a pseudo-random path through the cover.

- For embedding based on LDGM Codes, results close to the upper bound of embedding efficiency were achieved for a wide range of α^{-1}. Note that this approach is based on an iterative approximation of the coset leader.

- As the results achieved so far confirm, matrix embedding based on LDGM Codes combined with BP is indeed advantageous in terms of embedding complexity. No multiplications or divisions are required and the number of additions can be reduced considerably.

  While the embedding efficiency of the BP approach with some of the investigated approximations is equal to those of SP, there are also approximations that achieve slightly worse results. However, using approximations seems

to be promising since they offer a tradeoff between complexity and performance. We found that the approximation Log-Offset achieves the best tradeoff between performance and complexity.

- Even if embedding based on Syndrome Trellis Codes is even less complex, the results for the embedding efficiency presented Section 10.1.3 are slightly worse than those of the LDGM Codes.

- All approaches based on LDGM Codes and STC are able to cover a wide range of α^{-1} and achieve results close to the bound, whereas only STC with $h = 12$ achieve comparable results to those of embedding based on LDGM Codes. However, an increased constraint height directly results in an increased complexity as the number of states in the trellis is defined as 2^h. Thus, Filler et al. find $h = 12$ is reasonable in practice.

- Furthermore, we find for Wet Paper Codes and Syndrome Trellis Codes a non-negligible probability of failure. This non-solvability results in a coding loss, i. e., in a reduced capacity as well as in an increased complexity. In case of failure, the sender has to start the embedding process again with a reduced parameter k. Note that this issue is not valid for embedding based on LDGM Codes. This approach is based on compression and is always able to find a solution to Equation (3.33).

  To minimize the coding loss, Filler et al. propose to use a pseudo-random path through the cover. The more homogeneous distribution of wet elements within the cover makes the Viterbi algorithm less likely to fail [24].

Part IV.

Synthesis

11. Summary and Outlook

11.1. Results of this Thesis

In practice, a steganographic system is considered insecure, whenever an attacker is able to distinguish between cover and stego objects with more success than random guessing. Thus, the goal for the designer of a steganographic system is clear: cover and stego should be undistinguishable. Therefore, the characteristic of the cover has to be preserved during the embedding process, i. e., the introduction of embedding artifacts has to be minimized.

This thesis focused on the ability of syndrome coding as a concept of channel coding to reduce the distortion introduced during embedding and to enable a selection of the positions of changes.

In a first step, we proposed a classification of algorithms related to the concept of embedding based on syndrome coding. We separated the multitude of algorithms into two classes: approaches based on a deterministic parity-check matrix and approaches based on stochastic matrices. Furthermore, we differentiated between concepts for a small code word length and concepts for a large code word length.

In this thesis, we focused on basic algorithms related to binary embedding and described them on a consistent basis in a second step. Two of the state-of-the-art algorithms presented within this thesis - embedding based on BCH Codes [85, 86, 87] and based on LDGM Codes with Belief Propagation [58, 60, 59] - are developed in collaboration of Dagmar Schönfeld and the author of this thesis. Note that the algorithms discussed in this thesis are only a component of the embedding algorithm rather than a complete embedding scheme.

In a third step, the different approaches are compared to each other in order to determine advantages and disadvantages for each class of algorithms according to the following important design principles:

- the security in terms of embedding efficiency,
- the capacity,
- the success of embedding,
- the complexity.

The main results are summarized within the following sections.

11.1.1. Small Codeword Length

In this thesis, we considered the following approaches for syndrome coding based on matrices with a small code dimension:

- Embedding Based on Stochastic Parity-Check Matrices
  - Block Minimal Method
  - Matrix Embedding for Large Payloads
- Embedding Based on Deterministic Parity-Check Matrices
  - Hamming Codes
  - BCH Codes
  - Simplex Codes
  - Augmented Simplex Codes.

The evaluation of these algorithms was carried out according to the parameters security, capacity, success rate and complexity. Recall that when choosing a syndrome coding based embedding scheme, we have to address the goal of the scheme in a first step. It is not possible to maximize both, the security of the scheme (related to a high inverse relative message length $\alpha^{-1} = n/k$) as well as the capacity of the scheme (related to a low inverse relative message length).

Considering the *security*, i. e., the embedding efficiency of the algorithms, we find that all investigated codes achieve better results than the Hamming Codes. However, since Hamming Codes are easy to implement with low complexity, they are often used in practice (e. g., [96]).

Moreover, we found that Matrix Embedding for Large Payloads, Simplex Codes and Augmented Simplex Codes are only applicable for an inverse relative message length $\alpha^{-1} \approx 1$. However, according to the square root law of capacity [66], embedding a low inverse relative message length is rather insecure. Thus, for a practical scheme, a lower capacity is preferable.

As algorithms covering a wide range of the inverse relative message length α^{-1}, the BCH Codes as well as the Block Minimal Method achieve good results in terms of the embedding efficiency. We found both approaches as comparable. Even if the results considering the embedding efficiency for the Block Minimal Method are not as good as the results achievable for BCH Codes, the Block Minimal Method covers the range of α^{-1} more densely.

However, for embedding based on a stochastic parity-check matrix, we have to consider a non-negligible probability of the matrix of not being of full rank. Consequently, the *capacity* for approaches based on a stochastic parity-check matrix such as the Block Minimal Method is reduced. First, these approaches require the sender to store the actual achievable message length. Second, several

embedding trials have to be considered resulting in a further reduction of the capacity as well as an increased complexity.

Generally, we find that codes with a good performance f_k always have a relatively low number of information bits l, resulting in a relatively low maximum number of wet elements |Wet| excluded in the embedding process. However, they are suited for a large relative message length, since the number of embeddable bits per block k is high.

While there are fast solutions for calculating the stego sequence that minimizes the introduced distortion for Hamming Codes, Simplex Codes, Augmented Simplex Codes and BCH Codes with $f_k = 2$, no such strategy is known for approaches based on stochastic matrices. Thus, this is another clear advantage of codes based on deterministic parity-check matrices.

11.1.2. Large Codeword Length

In this thesis, we considered the following approaches for syndrome coding based on matrices with a large code dimension:

- Embedding Based on Wet Paper Codes,
- Embedding Based on LDGM Codes, and
- Embedding Based on Syndrome Trellis Codes.

Again, the evaluation of these algorithms was carried out according to the parameters security, capacity, success rate and complexity.

Note that the approaches based on Wet Paper Codes are not directly comparable to the other approaches since they do not minimize the introduced distortion $d_\rho(\mathbf{a}, \mathbf{b})$.

As the results presented in this thesis confirm, matrix embedding based on LDGM Codes combined with Belief Propagation is indeed advantageous in terms of *security*, i. e., embedding complexity. Note that embedding based on LDGM Codes combined with Belief Propagation achieve the same results in terms of embedding efficiency as LDGM Codes combined with Survey Propagation. However, the former approach is less complex. No multiplications or divisions are required and the number of additions can be reduced considerably. We achieve results close to the upper bound of embedding efficiency for a wide range of the inverse relative message length α^{-1}.

Even if embedding based on Syndrome Trellis Codes is less complex, the results in terms of embedding efficiency are slightly worse compared to those of the LDGM Codes. Only STC with a constraint height $h = 12$ achieve comparable results to those of embedding based on LDGM Codes. However, an increased constraint height h directly results in an increased complexity.

Furthermore, we find for Wet Paper Codes and Syndrome Trellis Codes a non-negligible probability of failure. This non-solvability results in a coding loss, i. e., in a reduced *capacity* as well as in an increased complexity. In case of failure, the sender has to start the embedding process again with a reduced message length. Note that this issue is not valid for embedding based on LDGM Codes. This approach is based on compression and is always able to find a solution.

11.2. Conclusion

As the results based on steganalytical investigations in [50, 71] confirm, it is indeed advantageous to modify as little as possible and only in inconspicuous parts of the cover. Thus, the application of syndrome coding in practical embedding schemes is preferable.

When choosing a code within a practical embedding scheme, we have to make the *main goal of the scheme* clear in a first step. As already mentioned, it is not possible to achieve optimal results for all four properties, namely security, capacity, success rate and complexity.

The results concerning the embedding efficiency dependent on the inverse relative message length α^{-1} are summarized in Figure 11.1. The dashed lines display the possibility of continuously covering the range of α^{-1} for codes built according to stochastic design rules.

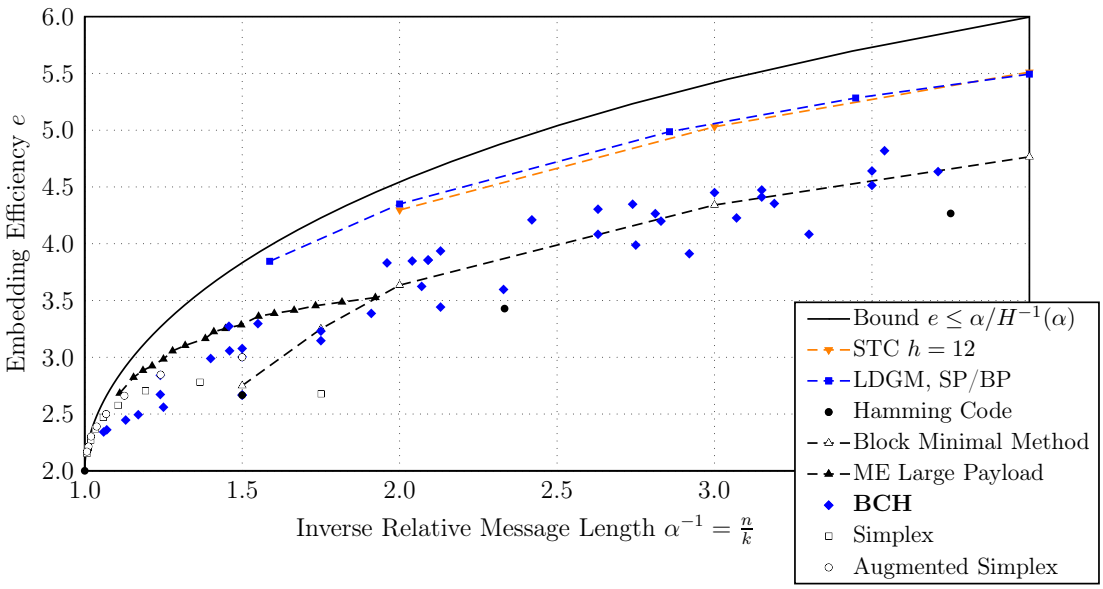

Figure 11.1.: Embedding Efficiency e Dependent on the Inverse Relative Message Length.

Whenever the goal is to achieve a *high capacity*, we have to choose a code with an inverse relative message length $\alpha^{-1} \approx 1$. We find Matrix Embedding for Large Payloads, Simplex Codes, Augmented Simplex Codes and Wet Paper Codes as suitable codes.

Due to the high capacity, we find $|\mathbf{m}| \approx N$. Consequently, these schemes achieve an embedding efficiency of $e \approx 2$. This result is comparable to embedding based on LSB or ± 1-embedding, i. e., embedding without syndrome coding. Even if the *security of these schemes is rather low* due to the low embedding efficiency, they enable to embed in selected parts of the cover without the need of sharing the selection rule with the receiver, in contrast to LSB embedding.

Note that there are rather small differences in the results concerning the capacity and the security considering Matrix Embedding for Large Payloads, Simplex Codes, Augmented Simplex Codes or Wet Paper Codes. However, for Wet Paper Codes, Simplex Codes, and Augmented Simplex Codes the sender is able to calculate the solution while for Matrix Embedding for Large Payloads he has to apply exhaustive search.

A major advantage for embedding based on codes described by means of a deterministic parity-check matrix is a high solvability even in a scenario with wet elements. While for codes based on stochastic parity-check matrices, the sender has to deal with non-solvability resulting in a reduced capacity as well as an *increased complexity*, this issue is not as crucial for codes based on deterministic matrices. Thus, the *success of embedding* for this class of codes is higher.

Generally, we find that codes with a good performance f_k always have a relatively low number of information bits l, resulting in a relatively low maximum number of wet elements $|\text{Wet}|$ excluded during the embedding process. However, they are suited for a large relative message length, since the number of embeddable bits per block k is high.

A high success of embedding is indeed important considering applications with a fixed message length or applications with real-time requirements. In this case, several embedding trials are unacceptable.

According to the square root law of capacity [66], the application of a higher inverse relative message length α^{-1} is preferable whenever the *security* of the embedding schemes is of prior interest. A good performance in terms of the embedding efficiency for such a high inverse relative message length can be achieved by means of LDGM Codes and Syndrome Trellis Codes.

These codes are based on stochastic matrices with constraints. Note that the *complexity of embedding* is reduced due to the application of decoding schemes known from channel coding such as message passing algorithms including Belief Propagation and Survey Propagation or the Viterbi algorithm.

While we have to deal with a reduced capacity and an increased complexity when applying the approach based on Syndrome Trellis Codes, this is not true for embedding based on LDGM Codes. Since this iterative embedding process is based on compression, we always find a solution.

However, some steganographic applications require an embedding scheme based on small blocks. In this case, we can apply either BCH Codes or the Block Minimal Method. While the results for BCH Codes in terms of the embedding efficiency are better, the range of α^{-1} can be covered more densely through the Block Minimal Method. However, we propose the application of BCH Codes due to their higher solvability.

11.3. Outlook

In this thesis, we described the basic algorithms for embedding based on binary codes built according to stochastic parity-check matrices as well as deterministic parity-check matrices. We focussed on the embedding step Θ, i. e., the input of the algorithms are a cover bit string $\mathbf{A}$, the message $\mathbf{m}$ and a profile of embedding impact $\boldsymbol{\rho}[1], \ldots, \boldsymbol{\rho}[N]$.

Thus, the first approach for further research will be the improvement of the embedding step, i. e., the task of finding codes or combinations of codes that are able to *achieve a higher embedding efficiency*.

Most embedding techniques are not limited to binary cases. Thus, some generalizations to arbitrary finite fields exist. For example, Galand et al. investigated embedding based on q-ary Reed Solomon (RS) Codes [28]. RS Codes enable the exclusion of up to l elements while it is possible to embed $(n-l)$ elements. However, we found the embedding efficiency for codes with a small l and an arbitrary q equal to those of the Hamming Codes. Considering an increased l, i. e. a higher number of wet elements, the results concerning the embedding efficiency become even worse. Consequently, these codes achieve an embedding efficiency far away from the upper bound.

Furthermore, a useful tool for constructing non-binary coverings, called blockwise direct sum, was proposed by Bierbrauer et al. in [5, 6]. This simple and efficient construction requires as input two factorizations of equal dimension and outputs a factorization of larger length. It allows controlling the covering radius of the output factorization.

As mentioned in Section 2.5.4, ternary embedding is the optimal choice for steganography if we wish to minimize the embedding distortion [49]. However, Filler stated that ternary coding is less effective for a higher inverse relative message length [20]. A general result of Fridrich et al. can be formulated as follows: it is not beneficial to increase the amplitude of embedding changes in exchange for their smaller number.

Thus, we focussed on binary embedding schemes. However, as the results in Figure 11.1 confirm, there is still a gap between the results in terms of embedding efficiency achieved so far and the upper boundary on embedding efficiency. Consequently, it remains an open question how to reach more closely to the bound with low computational complexity.

The second approach for further research will be the *design of a complete embedding algorithm*. An extension to syndrome coding based on Hamming Codes, called Modified Matrix Embedding, was presented by Kim et al. in [69, 103]. This simple and practical, even though suboptimal, steganographic algorithm for JPEG images is a combination of syndrome coding using binary $(2^k - 1, l)$ Hamming Codes and Perturbed Quantization (PQ). This scheme allows up to three embedding changes. Thus, the sender can select out of multiple possibilities the solution that introduces the smallest distortion. Even if this approach was presented for Hamming Codes, it is applicable also for other codes.

A different approach might be the concatenation of codes as applied for Turbo Codes used for error-correction. Within this approach, two codes are combined either serially or in parallel, increasing the performance considerably. Even though first investigations are not that promising [88], this might be a solution to reach the upper boundary on embedding efficiency.

Another extension, achieving a higher embedding efficiency, are so-called multilayer approaches such as ZZW or Paper Folding [107, 105, 104, 106, 35, 23, 102]. Within these approaches the cover is divided into several layers. In each layer, a part of the message is embedded, whereas only the elements in the lowest layer, i. e., in the least significant bitplane are modified. Therefore, the dependencies between the different layers have to be modeled. Codes based on LDGM matrices in combination with the ZZW construction [104, 35] achieve an embedding efficiency e very close to the bound for arbitrarily small relative message length.

A third approach for further research is the *definition of reasonable profiles of embedding impact*. A first approach is HUGO, where the profile of embedding impact is related to results from steganalyzers [79]. However, this is still a research field in its infantile stage. It seems to be reasonable to combine knowledge of the image acquisition process as well as knowledge gained from steganalytical investigations. However, the question remains whether it is possible to design reasonable cost functions on the fly.

A. Syndrome Coding Based on BCH Codes - Embedding Based on $g(x)$

As mentioned in Section 3.2.3.2, there are two possible ways to describe BCH Codes: one based on the parity-check matrix $\mathbf{H}_{k\times n}$ and another based on the generator polynomial $g(x)$. While the classic syndrome coding approach is based on the former description, we give heuristic approaches for the later one in this section according to [86].

Basic Approach. Using the generator polynomial $g(x)$ in coding theory, a syndrome $\mathbf{s}$ is calculated by dividing the received sequence $\mathbf{b}$ (or the corresponding polynomial $b(x)$) by the generator polynomial $g(x)$ and analyze the remainder. Whenever the received sequence $\mathbf{b}$ is a codeword, it is divisible without remainder. Otherwise an error is detected.

In steganographic systems, the remainder can be used to embed the secret message. Therefore, the cover sequence $\mathbf{a}$ is defined as $[\mathbf{a}_l\, \mathbf{a}_k]$, according to the parameters l and k.

The embedding process can be described as follows: in a first step, the cover sequence $\mathbf{a}$ is divided by $\mathbf{g}$ to obtain the syndrome $\mathbf{s}$ of length k:

$$\mathbf{s} = \mathrm{mod}(\mathbf{a}, \mathbf{g}). \tag{A.1}$$

Second, to achieve the positions which have to be flipped in order to embed $\mathbf{emb}$ into $\mathbf{a}$, the syndrome $\mathbf{s}$ has to be combined with $\mathbf{emb}$. As a result, the positions of ones are related to the positions that have to be flipped within $\mathbf{a}_k$. Using this approach, we flip only within the k parity bits of a block of length n[35].

In a third step, the stego sequence $\mathbf{b}$ leading to the confidential message $\mathbf{emb}$ and related to $\mathbf{a}$ can be achieved according to:

$$\mathbf{b} = [\mathbf{a}_l \;\; (\mathbf{a}_k \oplus (\mathbf{s} \oplus \mathbf{emb}))]\,. \tag{A.2}$$

The resulting stego sequence certainly fulfills the desired property:

$$\mathbf{s} = \mathrm{mod}(\mathbf{b}, \mathbf{g}) = \mathbf{emb}. \tag{A.3}$$

[35] It is recommended to use an interleaver to pre-process the image before embedding in order to reduce the impact of possible attacks.

An example for this embedding scheme is given below:

Example:
The sender would like to embed the confidential message $\mathbf{emb} = (001)$ within the cover bitstring $\mathbf{a} = (1011101) = [\mathbf{a}_l\ \mathbf{a}_k]$ with $l = 4, k = 3$. The code is given by means of its generator polynomial with $g(x) = x^3 + x + 1 = (1011)$. The embedding process is carried out as follows:

- Goal: $\mathbf{s} = \mathrm{mod}(\mathbf{b}, \mathbf{g}) = \mathbf{emb}$
- Determine $\mathbf{s} = \mathrm{mod}(\mathbf{b}, \mathbf{g}) = (101)$
- Determine the stego sequence with:

  $\mathbf{b} = [\mathbf{a}_l\ \mathbf{a}_k \oplus (\mathbf{s} \oplus \mathbf{emb})]$

  $\mathbf{b} = (1011001)$

■

Pre-flipping of $\mathbf{a}_l$. Note that it is not possible to achieve the maximum possible embedding efficiency with syndrome coding based on $g(x)$ as described above. However, we can improve embedding efficiency by heuristic approaches as described in this section.

First investigations for an improved algorithm [86] considered only one additional bit within l information bits. The essential idea of this heuristic approach is to pre-flip one out of all l possible information bits, embed the confidential message and evaluate the results.

The embedding process can be described as follows: in each step $i, (i = 1, 2, ..., l)$, one out of l information bits in $\mathbf{a}_l$ is pre-flipped. Afterward, $\mathbf{s}$ denoted as $\mathbf{s}_i$ is determined according to Equation (A.1). For each syndrome $\mathbf{s}_i$, the weight of $(\mathbf{s}_i \oplus \mathbf{emb})$ is determined within the evaluation step, since the goal is to find the coset leader or at least the combination with minimal Hamming weight.

The total number of bits, which have to be flipped within the cover sequence $\mathbf{a}$ in order to embed the confidential message part $\mathbf{emb}$, is now
$\mathrm{w}_1 = \min_{i=1}^{l}(\mathrm{w}(\mathbf{s}_i \oplus \mathbf{emb}) + 1)$, including the pre-flipped bit.

In order to determine whether the application of the improved approach is advantageous in comparison to the basic algorithm, w_2 is calculated as well, with the basic unmodified $\mathbf{a}_l$, as $\mathrm{w}_2 = \mathrm{w}(\mathbf{s} \oplus \mathbf{emb})$.

Whenever w_1 is smaller than w_2, pre-flipping one bit is indeed advantageous in comparison to the basic method. In this case, the w_1 bits are flipped. Otherwise the basic method is applied and w_2 bits are flipped.

An example for this embedding scheme is given below:

Example:
The sender would like to embed the confidential message $\mathbf{emb} = (001)$ within the cover bitstring $\mathbf{a} = (1011101) = [\mathbf{a}_l\ \mathbf{a}_k]$ with $l = 4, k = 3$. The code is given by means of its generator polynomial with $g(x) = x^3 + x + 1 = (1011)$. The embedding process is carried out as follows:

- Goal: $\mathbf{s} = \mathrm{mod}(\mathbf{b}, \mathbf{g}) = \mathbf{emb}$
- Approach needs l pre-processing steps (flipping patterns f_i with $\mathrm{w}(f_i) = 1$)
- Pre-flip one bit in $\mathbf{a}_l$ in each step and determine $\mathbf{s}$ as $\mathbf{s}_i = \mathrm{mod}(\mathbf{a}_{\mathbf{f}_i}, \mathbf{g})$
- Calculate $\mathbf{s}_i = \mathrm{mod}(\mathbf{a}_{\mathbf{f}_i}, \mathbf{g})$ with $\mathrm{w}(\mathbf{s}_i \oplus \mathbf{emb})$-> min

  $\mathbf{a}_{\mathbf{f}_3} = (1001101)$

  $\mathbf{s}_3 = (011)$, $\mathbf{s}_3 \oplus \mathbf{emb} = (010)$
- If $\mathrm{w}(\mathbf{s}_i \oplus \mathbf{emb}) + 1 < \mathrm{w}(\mathbf{s} \oplus \mathbf{emb})$

  $\mathbf{b} = [\mathbf{a}_l \oplus \mathbf{f}_i\ \mathbf{a}_k \oplus (\mathbf{s}_i \oplus \mathbf{emb})]$

  $\mathbf{b} = (1001111)$

■

By the same token, in the optimal case up to f_k pre-flipped bits are included. Experimental investigations have shown that it is sufficient to consider up to f_k instead of l pre-flipped bits. Using this approach, i.e., considering all possible flipping patterns $\binom{l}{i}$, $(i = 1, 2, ..., f_k)$, it is possible to reach the minimal possible R_a, i. e., the same embedding efficiency as with the approach described in Section 5.2.1.1.

B. $(15, 7, f_k = 2)$ BCH Code - Look-up Table

The $(15, 7, f_k = 2)$ BCH Code is given with its generator polynomial $g(x) = x^8 + x^7 + x^6 + x^4 + 1$. We find $h(x) = \frac{f(x)}{g(x)} = \frac{x^{15}+1}{x^8+x^7+x^6+x^4+1} = x^7 + x^6 + x^4 + 1$.

Cyclical shifting of the coefficients of $h(x)$ with $h = (000000011010001)$ leads to

$$H_{8\times 15} = \begin{pmatrix} 0\,0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1 \\ 0\,0\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0 \\ 0\,0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0 \\ 0\,0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0 \\ 0\,0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0 \\ 0\,0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0 \\ 0\,1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0 \\ 1\,1\,0\,1\,0\,0\,0\,1\,0\,0\,0\,0\,0\,0\,0 \end{pmatrix}.$$

Within the following table, all members of coset $\mathcal{C}(11100100)$ are given.

Table B.1.: Coset Member of coset $\mathcal{C}(11100100)$.

coset	coset member
(11100100)	(000000000100101),(000000100110010),(000001000001011) (000001100011100),(000010001111001),(000010101101110) (000011001010111),(000011101000000),(000100010011101) (000100110001010),(000101010110011),(000101110100100) (000110011000001),(000110111010110),(000111011101111) (000111111111000),(001000001000010),(001000101010101) (001001001101100),(001001101111011),(001010000011110)

coset	coset member
	(001010100001001),(001011000110000),(001011100100111)
	(001100011111010),(001100111101101),(001101011010100)
	(001101111000011),(001110010100110),(001110110110001)
	(001111010001000),(001111110011111),(010000011101011)
	(010000111111100),(010001011000101),(010001111010010)
	(010010010110111),(010010110100000),(010011010011001)
	(010011110001110),(010100001010011),(010100101000100)
	(010101001111101),(010101101101010),(010110000001111)
	(010110100011000),(010111000100001),(010111100110110)
	(011000010001100),(011000110011011),(011001010100010)
	(011001110110101),(011010011010000),(011010111000111)
	(011011011111110),(011011111101001),(011100000110100)
	(011100100100011),(011101000011010),(011101100001101)
	(011110001101000),(011110101111111),(011111001000110)
	(011111101010001),(100000010101110),(100000110111001)
	(100001010000000),(100001110010111),(100010011110010)
	(100010111100101),(100011011011100),(100011111001011)
	(100100000010110),(100100100000001),(100101000111000)
	(100101100101111),(100110001001010),(100110101011101)
	(100111001100100),(100111101110011),(101000011001001)
	(101000111011110),(101001011100111),(101001111110000)
	(101010010010101),(101010110000010),(101011010111011)
	(101011110101100),(101100001110001),(101100101100110)
	(101101001011111),(101101101001000),(101110000101101)
	(101110100111010),(101111000000011),(101111100010100)
	(110000001100000),(110000101110111),(110001001001110)
	(110001101011001),(110010000111100),(110010100101011)
	(110011000010010),(110011100000101),(110100011011000)
	(110100111001111),(110101011110110),(110101111100001)
	(110110010000100),(110110110010011),(110111010101010)
	(110111110111101),(111000000000111),(111000100010000)
	(111001000101001),(111001100111110),(111010001011011)
	(111010101001100),(111011001110101),(111011101100010)
	(111100010111111),(111100110101000),(111101010010001)
	(111101110000110),(111110011100011),(111110111110100)
	(111111011001101),(111111111011010)

C. Determining Look-up Tables for Fast Embedding Based on BCH Codes with $f_k = 2$

In this section, we give a brief overview on determining the Look-up Tables required for the algorithm of embedding by syndrome coding by means of BCH Codes with f_k as described in Section 5.2 according to [101]. For more information, we refer to [108] which is available only in Chinese.

In order to determine r^{u_l} from the syndrome components (Equation (5.6)), we assume $\beta_l = r^{u_l}$:

$$\mathbf{s}_1 = \beta_1 + \beta_2 + \beta_3 + \ldots + \beta_l \tag{C.1}$$
$$\mathbf{s}_2 = \beta_1^3 + \beta_2^3 + \beta_3^3 + \ldots + \beta_l^3. \tag{C.2}$$

Furthermore, we define the flip location polynomial as: $\sigma(x) = (x - \beta_1)(x - \beta_2)(x - \beta_3)\ldots(x - \beta_l)$ or $\sigma(x) = x^l + \sigma_1 x^{l-1} + \sigma_2 x^{l-2} + \ldots + \sigma_l$.

Note that the roots β can be determined after getting coefficients of the polynomial $\sigma(x)$. The relationship between σ and β is derived as follows:

$$\sigma_1 = \beta_1 + \beta_2 + \ldots + \beta_l \tag{C.3}$$
$$\sigma_2 = \beta_1\beta_2 + \beta_2\beta_3 + \ldots + \beta_{l-1}\beta_l \tag{C.4}$$
$$\sigma_3 = \beta_1\beta_2\beta_3 + \beta_3\beta_4\beta_5 + \ldots + \beta_{l-2}\beta_{l-1}\beta_l \tag{C.5}$$
$$. \tag{C.6}$$

The coefficients $\sigma_1, \sigma_2, \sigma_3$ relay to syndrome components $[\mathbf{s}_1\mathbf{s}_2]$ according to Newton's identities:

$$\mathbf{s}_1 + \sigma_1 = 0 \tag{C.7}$$
$$\mathbf{s}_2 + \sigma_1\mathbf{s}_1^2 + \sigma_2\mathbf{s}_1 + \sigma_3 = 0. \tag{C.8}$$

Again, this system of equations may have different solutions because it is underdetermined. The objective of steganography is to find in a first step a solution for $\sigma(x)$ with minimal degree.

Once $\sigma(x)$ is determined, in a second step is to find the roots β. The roots are calculated before embedding and are stored in Look-up Tables. Thus, the method does not require exhaustive search to find the roots.

The authors propose to utilize the method of Zhao et al. based on fast Look-up Tables for finding roots for quadratic and cubic polynomials $\sigma(x)$ [108]. A brief description of the coherences used to build the tables **quadratic** and **cubic**, containing the roots of the quadratic and cubic polynomial, respectively, can be found in this section. Furthermore, a table **tab** denotes entries that have 3 roots.

Determining a Look-up Table quadratic for Quadratic Polynomial

The approaches exploits the relationship between roots of the pair of quadratic polynomials in $GF(2^m)$ $f(x) = x^2 + \sigma_1 x + \sigma_2$ and $f(y) = y^2 + y + \frac{\sigma_2}{\sigma_1^2}$:

- If y_1 is the root of the polynomial $f(y)$, then $y_1 + 1$ is another root of $f(y)$
- also, $x_1 = \sigma_1 y_1$ and $x_2 = \sigma_1 y_2 + \sigma_1$ are roots of the polynomial $f(x)$.

The Look-up Table **q** keeps the first root of the polynomial $f_i(y) = y^2 + y + i$, where $i \in [1; 2^m - 1]$ in $GF(2^m)$. Thus, the size of the Look-up Table is $(2^m - 1) \times 1$. The roots of the polynomial $f(x)$ can be computed using the root y_0 from the Look-up Table in position $\frac{\sigma_2}{\sigma_1^2}$, empty placed in the table are marked with -1.

Determining a Look-up Table cubic for Cubic Polynomial

First, two parameters h and j are defined for a cubic polynomial $f(x) = x^3 + \sigma_1 x^2 + \sigma_2 x + \sigma_3$ such as $h = \sigma_1^2 + \sigma_2$ and $j = \sigma_1 \sigma_2 + \sigma_3$. Note that the polynomial $f(y) = y^3 + y + \frac{j}{h^{3/2}}$ is a pair to polynomial $f(x)$.

- If y_1, y_2, y_3 are the root of the polynomial $f(y)$ for a certain value $\frac{j}{h^{3/2}}$, then $x_s = h^{1/2} y_s + \sigma_1$ with $(s = 1, 2, 3)$ is the set of three roots of the polynomial $f(x)$

The Look-up Table **cubic** keeps roots y for any polynomial $f_i(y) = y^3 + y + i$, where $i \in [1; 2^m - 1]$ in $GF(2^m)$. Consequently, the size of the Look-up Table is $(2^m - 1) \times 3$. Note that polynomial which do not have 3 roots are marked as -1.

The roots of the polynomial $f(x) = x^3 + \sigma_1 x^2 + \sigma_2 x + \sigma_3$ can be computed using roots y in the Look-up Table **cubic** in row $\frac{j}{h^{3/2}}$. For further calculations, the indexes of the rows which have 3 roots have to be stored in a special table **tab** of size D, where D is the number of rows which have 3 roots.

D. BCH Code - Example Code Parameters

Within this section, example code parameters are given for several BCH Codes according to Bellmann [3]. While Table D.1 gives examples for primitive codes, Table D.2 lists examples for non-primitive BCH Codes. Note that the codes are separated according to the number of minimal polynomials $m_i(x)$ used for the construction of $g(x)$, as basis of the parity-check matrix $\mathbf{H}_{k\times n}$.

Table D.1.: Examples for Primitive BCH Codes.

(n, l)	e
3 $m_i(x)$	
(15,10)	3.003
(15,12)	2.560
(31,15)	3.624
(63,18)	4.641
4 $m_i(x)$	
(15,14)	2.361
(31,20)	3.296
(63,17)	4.635
(63,18)	4.514
(63,20)	4.473
(63,21)	4.450
(63,24)	4.304
5 $m_i(x)$	
(31,25)	2.841
(63,20)	4.411
(63,23)	4.348
(63,24)	4.083
(63,26)	4.210

Table D.2.: Examples for Non-Primitive BCH Code.

(n, l)	e
1 $m_i(x)$	
(9,6)	2.667
(17,8)	3.442
(23,11)	3.856
(33,10)	4.083
(35,12)	3.912
(43,14)	4.227
(47,23)	3.848
2 $m_i(x)$	
(9,8)	2.447
(17,16)	2.342
(21,9)	3.597
(21,12)	3.146
(33,12)	3.988
(35,24)	3.057
(51,16)	4.354
3 $m_i(x)$	
(21,11)	3.386
(21,12)	3.230
(21,14)	2.667
(21,15)	2.989
(51,18)	4.199
(51,24)	3.936
(85,24)	4.819
4 $m_i(x)$	
(21,14)	3.077
(21,17)	2.672
(21,18)	2.494
(51,26)	3.831

E. Algorithm for Approximating the Embedding Efficiency for BCH Codes

Given that the estimation of R_a and thereby that of e so far requires extensive simulations, we describe an approximation of the embedding efficiency within this section [86]. We developed an algorithm to approximate the embedding efficiency e directly from the code parameters k and n. Of course this algorithm can be used the other way around, in order to find suitable code parameters.

Unfortunately, only perfect codes such as Hamming Codes ($f_k = 1$) and the non-primitive Golay Code ($f_k = 3$) are able to achieve an average number of embedding changes R_a even smaller than f_k. The reason is that the weights of the coset leaders are distributed according to the Hamming bound with:

$$2^k = \sum_{i=0}^{f_k} \binom{n}{i}, \tag{E.1}$$

i. e., there are exactly $\binom{n}{i}$ coset leader with weight i. According to this property, the average number of embedding changes can be calculated with:

$$R_a = \frac{1}{2^k} \sum_{i=0}^{f_k} \binom{n}{i} i. \tag{E.2}$$

For non-perfect codes, R_a will be always greater than f_k since the Hamming bound is not fulfilled with equality: there are more possible syndromes than needed to realize the performance f_k for the code applied in coding theory. When using these codes in steganographic systems, the remaining syndromes have to be realized using sequences $\mathbf{f}$ with $\mathrm{w}(\mathbf{f}) > f_k$.

For every quasi-perfect code ($f_k = 2$), we can determine the weight of the remaining $\left(2^k - \sum_{i=0}^{f_k} \binom{n}{i}\right)$ coset leaders exactly with $(f_k + 1)$. Since the equation to calculate R_a can be adapted, we calculate R_a with:

$$R_a = \frac{1}{2^k} \left(\sum_{i=0}^{f_k} \binom{n}{i} i + \left(2^k - \sum_{i=0}^{f_k} \binom{n}{i}\right) (f_k + 1) \right). \tag{E.3}$$

Input: k; n	
$s := 0$; $i := -1$	
$s < 2^k$	
$i := i + 1$	
$s := s + \binom{n}{i}$	summation of all syndromes
$\mathsf{w}[i] := \binom{n}{i}$	number of coset leader with weight i
$\mathsf{w}[i] := 2^k - \sum_{j=0}^{i-1} \mathsf{w}[j]$	
$R := i$	maximum number of embedding changes
$R_a := \frac{1}{2^k} \sum_{j=0}^{R} \mathsf{w}[j] \cdot j$	average number of embedding changes
$e := \frac{k}{R_a}$; $\alpha^{-1} := \frac{n}{k}$	
Output: e; α^{-1}	

Figure E.1.: Algorithm to Estimate the Embedding Efficiency e According to [86].

However, for shortened codes and for codes with better performance, i. e., $f_k > 2$, the remaining coset leaders do not have only weight $(f_k + 1)$. As our investigations have confirmed, they also contain coset leaders with weight $\mathrm{w}(\mathbf{f}) > (f_k + 1)$.

In order to estimate R_a for these codes, we assume Equation (E.3), although this is only a lower bound since this equation is true for $f_k = 2$. The actual average number of embedding changes R_a will be higher for shortened codes and codes with $f_k > 2$ depending on the number of remaining coset leaders. With increasing codeword length, the number of coset leaders not covered by the Hamming bound is up to 90% of all 2^k possible coset leaders [86], i. e., these coset leaders have a weight of $\mathrm{w}(\mathbf{f}) \geq (f_k + 1)$.

Approximating the maximum and the average number of embedding changes (R, R_a) enables us to estimate the embedding efficiency e and the ratio $\alpha^{-1} = n/k$ (Figure E.1).

Table C.1 contains examples for the accuracy of our approximation of R_a. The vector $\mathbf{w}$ gives the distribution of the coset leaders among the weights i, i. e., $\mathbf{w}[1]$ for example gives the number of coset leader with weight 1. Additionally, the ratio between the calculated and the estimated value of R_a provides a measure

Table E.1.: Quality of the Approximation Algorithm According to [86].

(n, l, f_k)	Distribution of the Coset Leaders among the Weights $\mathbf{w}[i]$, $(i = 0, 1, ..., n)$	R_a	$\frac{R_{a,\mathbf{c}}}{R_{a,\mathbf{e}}}$
$(15, 5, 3)$, primitive	**c**alculated: $(1, 15, 105, 455, 420, 28, 0, ..., 0)$ **e**stimated: $(1, 15, 105, 455, 448, 0, ..., 0)$	3.330 3.303	99.2%
$(18, 7, 3)$, shortened, non-primitive	**c**alculated: $(1, 18, 153, 816, 955, 100, 5, 0, ..., 0)$ **e**stimated: $(1, 18, 153, 816, 1060, 0, ..., 0)$	3.478 3.213	92.4%
$(31, 11, 5)$, primitive	**c**alculated: $(1, 31, 465, 4495, 31465, 169911, 522009, 320199, 0, \ldots, 0)$ **e**stimated: $(1, 31, 465, 4495, 31465, 169911, 736281, 105927, 0, \ldots, 0)$	6.068 5.864	96.6%

of quality for our estimation.

As already mentioned, the approximation gives always a lower bound for codes with $f_k > 2$. However, there are only minor differences between the calculated and the estimated average number of embedding changes R_a (Table C.1).

Hence, we used the algorithm described in Figure E.1 the other way around in order to find appropriate code parameters [86]. Therefore, we varied the code parameters n, k and evaluated the results concerning e and α^{-1}. The most promising results are presented in Table E.2.

Table E.2.: Results for Promising Code Parameters According to [86].

n	k	e	α^{-1}
37	19	3.904	1.947
51	18	4.603	2.833
99	34	4.864	2.912

These parameters are substantially better than the results presented in Section 6.1. However, it is still a challenge to derive an appropriate code based on the estimated parameters. For example, for $n = 37$ and $k = 19$, we find the embedding efficiency $e = 3.904$ and the inverse relative message length $\alpha^{-1} = 1.947$ as promising parameters [86].

According to these code parameters, we can use shortened primitive or shortened non-primitive BCH Codes. The question is which code will be able to achieve the calculated embedding efficiency. The results determined by Schönfeld and Winkler in [86] are summarized in Table E.3. For further information, we refer to this paper. Nevertheless, the question is whether it is possible to find $\mathbf{H}_{k\times n}$ according to the parameters n and k easier with random codes, since they densely cover the range of α^{-1}.

Table E.3.: Results for Finding an Appropriate Code According to [86].

(n, l, f_k), shortened primitive	e	α^{-1}
$(37, 19, 3)$	2.934	2.056
$(37, 19, 2)$	3.001	2.056
$(37, 18, 1)$	2.673	1.947

(n, l, f_k), non-primitive	e	α^{-1}
$(35, 23, 1)$	3.092	2.917
$(35, 11, 2)$	2.588	1.458

Glossary

±1 steganography	Steganographic embedding operation which applies incrementing or decrementing with equal probability, whenever the message bit is not equal to the cover bit.
Adaptive Selection Channel	Embedding Principle where the selection rule depends on the cover.
Adaptive Steganography	Steganographic scheme, where the embedding rule depends on the cover content.
Attack	Algorithm whose goal is to detect the existence of steganography.
AUC	Area Under the Curve. A metric based on the ROC curve.
BCH Code	BCH Code, named after its discoverers Bose, Chaudhuri and Hocquenghem. Algorithm for embedding with syndrome coding based on deterministic parity-check matrices.
Block Minimal Method	Algorithm for embedding with syndrome coding based on stochastic parity-check matrices.
BMP	Bitmap Image. A widely-used image format.
BP	Belief Propagation. Iterative message-passing algorithm applied for embedding based on LDGM Codes.
bpp	Bit Per Pixel. Measure to express the relative message length.
Capacity	Maximum number of message bits that can be embedded in a cover object.

Changeable Element	Cover element that might be changed during embedding. Also denoted as dry element.
Code	A vector subspace of $\mathbb{F}_q^n$.
Code Co-Dimension	The co-dimension k of a linear code gives the number of parity bits.
Code Dimension	Dimension l of a linear code. The code dimension determines the cardinality of the code alphabet with 2^l elements.
Codeword	An element of the code.
Codeword Length	Number of elements in each codeword given as n.
Coding Theory	The science of the properties of codes and their practicability for specific applications.
Confidential Message	Message the sender wants to embed.
Coset	Set of all codewords leading to the same syndrome.
Coset Leader	Member of the coset with the smallest Hamming weight w.
Cover	Original unmodified object before embedding.
Cover Element	Individual element of the cover, e. g., a pixel or a DCT coefficient.
Cover Modification	Steganographic method where the cover is modified in order to embed the confidential message.
Cover Selection	Steganographic method where the cover is selected out of a source of several covers. The chosen cover contains the confidential message.
Cover Synthesis	Steganographic method where the cover containing the confidential message is generated in a computer-based way.
Covering	One of the two main problems that can be solved using coding theory.
Covering Radius	Measure of the distance between the code and the farthest-off vectors in space.

DCT	Discrete Cosine Transformation.
Deterministic Parity-Check Matrix	Parity-check matrix built according to deterministic design rules, i. e., related to a linear code.
Dry Element	Cover element allowed to be changed during embedding.
ECC	Error Correcting Codes.
EER	Equal Error Rate. A metric based on the ROC curve.
Embedding Algorithm	Algorithm that embeds a confidential message into a cover.
Embedding Distortion	Distortion introduced to the cover during embedding.
Embedding Efficiency	Number of embeddable bits dependent on the introduced distortion.
Embedding Impact	Expresses how strongly a modification of a particular position of the cover would influence the probability of an attacker detecting the embedding.
Embedding Operation	Procedure modifying the individual cover element in order to embed.
Extraction Algorithm	Algorithm to extract the confidential message from the stego object.
False Alarm	Misclassification of a cover object as a stego object.
FP50	False Positive rate at 50 percent detection rate. A metric based on the ROC curve.
Generator Matrix	Matrix whose rows form the basis of a linear code.
GF	Galois Field.
Golay Code	Beside the Hamming Code the only perfect code. Algorithm for embedding with syndrome coding based on deterministic parity-check matrices.

Hamming Distance	Number of elements in which two words differ.
Hamming Weight	Number of non-zero elements in each codeword.
HC	Hamming Code: Perfect linear code. Parity-check matrix contains all non-zero elements of length k as rows. Algorithm for embedding with syndrome coding based on deterministic parity-check matrices.
JPEG	Joint Photographic Experts Group. A widely-used image format.
KL Divergence	Kullback-Leibler divergence: Concept from information theory. Gives a measure of the distance between two random variables.
LDGM	Low-Density-Generator-Matrix Codes. Algorithm for embedding with syndrome coding based on stochastic parity-check matrices.
LDPC	Low-Density-Parity-Check Matrix Codes.
Linear Code	Linear subspace $\subset \mathbb{F}_q^n$ including all possible codewords.
LLR	Log Likelihood Ratio.
LSB	Least Significant Bit.
LSB Embedding	Steganographic method. Embedding by replacing the least significant bit with the message bit.
LSB Matching	Steganographic method, ± 1-embedding.
LT Code	Luby Transform Code. Sparse linear code.
Matrix LT Process	Algorithm based on LT Codes which allows to bring a matrix in the upper triangular form.

ME	Matrix Embedding: Synonym for Embedding based on Syndrome Coding.
ME for Large Payloads	Algorithm for embedding with syndrome coding based on stochastic parity-check matrices.
Meet-in-the-Middle	Algorithm based on stochastic parity-check matrices. Synonym for Block Minimal Method.
Missed Detection	Misclassification of a stego object as cover object.
MP3	MPEG-1 Audio Layer 3. A widely-used audio format.
MPEG	Moving Picture Experts Group. A widely-used video format.
Number of Embedding Changes	Average weight of the coset leaders.
Packing	One of the two main problems that can be solved using coding theory.
Parity-Check Matrix	Matrix used for error-correction in coding theory and for embedding and extraction in steganography.
PKS	Public Key Steganography.
PQ	Perturbed Quantization: Embedding Principle in which embedding occurs during an information-reducing process.
Random Parity-Check Matrix	A parity-check matrix built according to stochastic design rules.
Robustness	Property describing a steganographic scheme. Describes the difficulty of removing hidden information from a stego object.
ROC	Receiver Operating Characteristics: A common way to visualize the characteristic relation between the two errors a steganalyzer can make.

RSD	Robust Soliton Distribution.
SC	Simplex Code. Algorithm for embedding with syndrome coding based on deterministic parity-check matrices.
Security	Property describing a steganographic scheme. Judged by the impossibility of detecting the communication.
Selection Channel	Selects parts of the cover that should be used for embedding.
SID	Survey Inspired Decimation.
SKS	Secret Key Steganography.
SP	Survey Propagation. Iterative message-passing algorithm applied for embedding based on LDGM Codes.
Sparse Matrix	For this class of matrices, the weight of the matrix increases linearly instead of quadratically with increasing block length n.
Square Root Law	Thesis claiming that the secure capacity of a stego system is proportional only to the square root of the number of cover elements.
STC	Syndrome Trellis Codes. Algorithm for embedding with syndrome coding based on stochastic parity-check matrices.
Steganalysis	The counterpart to steganography.
Steganographic Scheme	A covert communication system consisting of the cover source, the stego key source, the message source and the communication channel.
Steganography	The art of communicating messages in a covert manner.
Stego	Raw, unmodified data before embedding.
Success of Embedding	Probability that a given message can be embedded in a particular cover.
Syndrome	Result when multiplying a sequence of length n with the parity-check matrix.

Syndrome Coding	Method for communicating a secret message as the syndrome of a linear code.
Systematic Form	Special arrangement of a matrix. One part of the matrix is the identity matrix.
Ternary Code	A code built over a ternary alphabet, i.e., an alphabet consisting of three symbols.
Undetectability	Property describing a steganographic scheme. Describes the inability of the attacker to prove the existence of the confidential message.
Wet Element	Cover element not used for embedding, remains unchanged during the embedding process.
WPC	Wet Paper Codes. Algorithm for embedding with syndrome coding based on stochastic parity-check matrices.

Notation and Symbols

$\boldsymbol{\rho}[i]$	Embedding impact at pixel i
Γ	Modification step
Λ	Extraction step
Π	Pre-processing step
Θ	Embedding step
$\mathbf{A}$	Cover bit string
$\mathbf{a}$	Cover sequence
α^{-1}	Inverse relative message length $\alpha^{-1} = \frac{n}{k}$
α_{Dry}^{-1}	Inverse relative message length related to the dry cover elements $\alpha_{\text{Dry}}^{-1} = \lvert\text{Dry}\rvert/\lvert\mathbf{m}\rvert$
α	Relative message length $\alpha = \frac{k}{n}$
$\mathbf{B}$	Stego bit string
$\mathbf{b}$	Stego sequence
$\mathcal{C}$	Code
$\mathbf{c}$	Codeword
$\mathcal{C}(\mathbf{s})$	Coset corresponding to syndrome $\mathbf{s}$
$\mathbf{c}_L(\mathbf{s})$	Coset leader of the coset corresponding to syndrome $\mathbf{s}$
$d_H(\mathbf{x}, \mathbf{y})$	Hamming distance between $\mathbf{x}$ and $\mathbf{y}$
$D_{KL}(P\|\|Q)$	Kullback-Leibler (KL) divergence between the probability distributions P and Q
d_{min}	Minimum Hamming distance among all possible distinct pairs in $\mathcal{C}$
$d_\rho(\mathbf{x}, \mathbf{y})$	Impact introduced during embedding $d_\rho(\mathbf{A}, \mathbf{B}) = \sum_{i=1}^{n} \rho[i]\lvert\mathbf{A}[i] - \mathbf{B}[i]\rvert$
Dry	Set of changeable (dry) elements
e	Embedding efficiency

$\underline{e}$	Lower embedding efficiency
Emb	Embedding function
emb	Message part used for embedding
Extr	Extraction function
$f(x)$	Main polynomial
$\mathbf{f}$	Error pattern, flipping pattern
$\mathbb{F}_2$	Finite field containing two elements
$\mathbf{f}_{\|\text{Dry}\|}$	Flipping pattern related to the dry elements
f_k	Number of correctable errors $f_k = \lfloor \frac{d_{min}-1}{2} \rfloor$
$\mathbf{f_m}$	Coset member
$\mathbb{F}_q$	Finite field containing q elements
$\mathbf{G}_{l\times n}$	Generator matrix
$g(x)$	Generator polynomial
GF	Galois field
$H(x)$	Binary entropy function
$\mathbf{H}_{k\times\|\text{Dry}\|}$	Sub-matrix of $\mathbf{H}_{k\times n}$ corresponding to the dry elements
$\mathbf{H}_{k\times\|\text{Wet}\|}$	Sub-matrix of $\mathbf{H}_{k\times n}$ corresponding to the wet elements
$\mathbf{H}_{k\times n}$	Parity-check matrix
$h(x)$	Check polynomial
H_0	Null hypothesis
H_1	Alternative hypothesis
$H^{-1}(x)$	Inverse binary entropy function
$\mathbf{I}_k$	Identity matrix
$\mathcal{K}$	Set of possible stego keys
$\mathbf{k}$	Secret stego key from the set of possible stego keys, $\mathbf{k} \in \mathcal{K}$
k	Co-dimension of the code, number of parity bits

l	Dimension of the code, number of information bits
LCM	Least common multiplier
ld	Logarithm dualis, logarithm to the base 2
ln	Logarithm naturalis, logarithm to the base e
log	Logarithm, logarithm to the base 10
$M(x)$	Modular polynomial
$\mathcal{M}$	Set of possible messages
$\mathbf{m}$	Message from the set of possible messages, $\mathbf{m} \in \mathcal{M}$
$m_i(x)$	Minimal polynomial
N	Number of cover elements
n	Codeword length
n_B	Number of blocks per cover
Path	Random path trough the image
$P_{\mathbf{c}}$	Distribution of cover objects
$\pi(x)$	Symbol-assignment function
$P_{\mathbf{s}}$	Distribution of stego objects
q	Basis of the q-ary alphabet
R	Covering radius
R_a	Average distance to code
rank	Rank of a matrix
$\mathbf{s}$	Syndrome
w	Hamming weight of $\mathbf{x} \in \{0,1\}^n$
Wet	Set of unchangeable (wet) elements
$\lvert\mathcal{X}\rvert$	Cardinality of set $\mathcal{X}$
$\mathcal{X}$	Set of possible covers
$[\mathbf{i}]_2$	Binary representation of $\mathbf{i}$
$\lvert\mathbf{x}\rvert$	Length of vector $\mathbf{x}$

$[\mathbf{i}]_{10}$	Denary representation of $\mathbf{i}$
$\boldsymbol{\chi}$	Cover
$\mathbf{X}[i,.]$	ith row of matrix $\mathbf{X}$
$\mathbf{x}[i]$	ith element of vector $\mathbf{x}$
$\mathbf{X}[.,j]$	jth column of matrix $\mathbf{X}$
$\oplus$	XOR (eXclusive OR)
$\mathbf{X}^T$	Transpose of matrix $\mathbf{X}$
$\mathbf{x}^T$	Transpose of vector $\mathbf{x}$
$\otimes$	Kronecker Product

Bibliography

[1] R. Anderson. Stretching the Limits of Steganography. In R. Anderson, editor, *Proc. of Information Hiding, 1st International Workshop*, volume LNCS 1174, pages 39–48, Cambridge, UK, May 30-June 1 1996.

[2] R. Anderson and F. A. P. Petitcolas. On the Limits of Steganography. *IEEE Journal on Selected Areas in Communication*, 16(4):474–481, 1998.

[3] U. Bellmann. *Einbettung mit Syndromauswertung*. Belegarbeit, Technische Universität Dresden, Oktober 2007. Supervisor: D. Schönfeld.

[4] E. Berlekamp, R. McEliece, and H. van Tilborg. On the Inherent Intractability of Certain Coding Problems. *IEEE Transactions on Information Theory*, 24(3):384–386, 1978.

[5] J. Bierbrauer. Crandall's Problem. www.ws.binghamton.edu/fridrich/covcodes.pdf, August 27, 1998 and November 13, 2001.

[6] J. Bierbrauer and J. Fridrich. Constructing Good Covering Codes for Applications in Steganography. *Transactions on Data Hiding and Multimedia Security III*, LNCS 4920:1–22, 2008.

[7] R. Böhme. Wet Paper Codes for Public Key Steganography? Rump Session Talk at the 7th International Workshop on Information Hiding, June 2005.

[8] R. Böhme. *Improved Statistical Steganalysis using Models of Heterogeneous Cover Signals*. PhD thesis, Technische Universität Dresden, 2008. Supervisor: A. Pfitzmann.

[9] R. Böhme and A. Westfeld. Exploiting Preserved Statistics for Steganalysis. In J. Fridrich, editor, *Proc. of Information Hiding, 6th International Workshop*, volume LNCS 3200, pages 82–96, Toronto, Canada, May 23-25 2004.

[10] K. Borowka. *Integration von Fehlerkorrektur bei JPEG-Kompression des Steganogramms*. Belegarbeit, Technische Universität Dresden, Januar 2007. Supervisor: A. Winkler.

[11] M. Bossert. *Channel Coding for Telecommunications*. John Wiley & Sons, New York, NY, USA, 1st edition, 1999.

[12] R. P. Brent, S. Gao, and A. G. B. Lauder. Random Krylov Spaces over Finite Fields. *SIAM Journal on Discrete Mathematics*, 16(2):276–287, 2003.

[13] C. Cachin. An Information-Theoretic Model for Steganography. In D. Aucsmith, editor, *Proc. of Information Hiding, 2nd International Workshop*, volume LNCS 1525, pages 306–318, Portland, Oregon, April 14-17 1998.

[14] G. Cohen, I. Honkala, S. Litsyn, and A. Lobstein. *Covering Codes*. Elsevier Science B.V., 1997.

[15] P. Comesana and F. Pérez-González. On the Capacity of Stegosystems. In *Proc. of ACM Multimedia and Security (MM & Sec)*, pages 15–24, Dallas, TX, September 20-21 2007.

[16] R. Crandall. Some Notes on Steganography. http://os.inf.tu-dresden.de/~westfeld/crandall.pdf, 1998.

[17] S. Dumitrescu, X. Wu, and N. D. Memon. On Steganalysis of Random LSB Embedding in Continous-Tone Images. In *Proc. of IEEE International Conference on Image Processing*, pages 641–644, Rochester, NY, September 22-25 2002.

[18] P. Elias. Coding for Noisy Channels. *IRE National Convention Record*, 4:37–46, 1955.

[19] T. Filler. Minimizing Embedding Impact in Steganography using Low Density Codes. Master's thesis, School of Nuclear Science and Physical Engineering, Czech Technical University, May 2007. Supervisor: J. Fridrich.

[20] T. Filler. How Good is Ternary Coding For Small Payloads? Rump Session Talk at the 11th International Workshop on Information Hiding, June 2009.

[21] T. Filler and J. Fridrich. Binary Quantization using Belief Propagation with Decimation over Factor Graphs of LDGM Codes. In *Proc. of 45th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, September 26-28 2007.

[22] T. Filler and J. Fridrich. Fisher Information Determines Capacity of epsilon-secure Steganography. In S. Katzenbeisser and A.-R. Sadeghi, editors, *Proc. of Information Hiding, 11th International Workshop*, volume LNCS 5806, pages 31–47, Darmstadt, Germany, June 7-10 2009.

[23] T. Filler and J. Fridrich. Wet ZZW Construction for Steganography. In *Proc. of 1st IEEE Workshop on Information Forensic and Security (WIFS)*, pages 131–135, London, UK, December 6-9 2009.

[24] T. Filler, J. Judas, and J. Fridrich. Minimizing Embedding Impact in Steganography Using Trellis-Coded Quantization. In N. D. Memon, J. Dittmann, A. M. Alattar, and E. J. Delp, editors, *Proc. of IS&T/SPIE Electronic Imaging, Media Forensics and Security XII*, volume 7541, pages 754105 01–754105 14, San Jose, CA, January 18-20 2010.

[25] T. Filler, J. Judas, and J. Fridrich. Minimizing Additive Distortion in Steganography using Syndrome-Trellis Codes. *to appear in IEEE Transactions on Information Forensics and Security*, 2011.

[26] T. Filler, A. Ker, and J. Fridrich. The Square Root Law of Steganographic Capacity for Markov Covers. In E. J. Delp, J. Dittmann, N. Memon, and P. W. Wong, editors, *Proc. of IS&T/SPIE Electronic Imaging, Media Forensics and Security XI*, volume 7254, pages 725408 1–725408 11, San Jose, CA, January 19-21 2009.

[27] C. Fontaine and F. Galand. How Can Reed-Solomon Codes Improve Steganographic Schemes? In T. Furon, F. Cayre, G. Doërr, and P. Bas, editors, *Proc. of Information Hiding, 9th International Workshop*, volume LNCS 4567, pages 130–144, Saint Malo, France, June 11-13 2007.

[28] C. Fontaine and F. Galand. How Reed-Solomon Codes Can Improve Steganographic Schemes. *EURASIP Journal on Information Security*, page Article ID 274845, 2009.

[29] E. Franz. *Steganographie durch Nachbildung plausibler Änderungen.* PhD thesis, Technische Universität Dresden, 2002. Supervisor: A. Pfitzmann.

[30] E. Franz. Steganography Preserving Statistical Properties. In F. A. P. Petitcolas, editor, *Proc. of Information Hiding, 5th International Workshop*, volume LNCS 2578, pages 278–294, Noordwijkerhout, The Netherlands, October 7-9 2002.

[31] E. Franz and A. Schneidewind. Adaptive Steganography Based on Dithering. In J. Dittmann and J. Fridrich, editors, *Proc. of ACM Multimedia and Security (MM & Sec)*, pages 56–62, Magdeburg, Germany, September 20-21 2004.

[32] E. Franz and A. Schneidewind. Pre-Processing for Adding Noise Steganography. In M. Barni, J. Herrera-Joancomartí, S. Katzenbeisser, and F. Pérez-González, editors, *Proc. of Information Hiding, 7th International Workshop*, volume LNCS 3727, pages 189–203, Barcelona, Spain, June 6-8 2005.

[33] J. Fridrich. Feature-Based Steganalysis for JPEG Images and its Implications for Future Design of Steganographic Schemes. In J. Fridrich, editor,

Proc. of Information Hiding, 6th International Workshop, volume LNCS 3200, pages 67–81, Toronto, Canada, May 23-25 2004.

[34] J. Fridrich. Minimizing the Embedding Impact in Steganography. In *Proc. of ACM Multimedia and Security (MM & Sec)*, pages 2–10, Geneva, Switzerland, September 26-27 2006.

[35] J. Fridrich. Asymptotic Behavior of the ZZW Embedding Construction. *IEEE Transactions on Information Forensics and Security*, 4(1):151–154, March 2009.

[36] J. Fridrich. *Steganography in Digital Media - Principles, Algorithms, and Applications*. Cambridge University Press, 2010.

[37] J. Fridrich and T. Filler. Practical Methods for Minimizing Embedding Impact in Steganography. In E. J. Delp and P. W. Wong, editors, *Proc. of SPIE Security and Watermarking of Multimedia Contents IX*, volume 6505, page 650502, San Jose, CA, January 29 - February 1 2007.

[38] J. Fridrich and M. Goljan. Digital Image Steganography Using Stochastic Modulation. In E. J. Delp and P. W. Wong, editors, *Proc. of SPIE Security and Watermarking of Multimedia Contents V*, volume 5020, pages 191–202, Santa Clara, CA, January 21-23 2003.

[39] J. Fridrich, M. Goljan, and R. Du. Detecting LSB Steganography in Color and Gray-Scale Images. *IEEE Multimedia*, 8(4):22–28, October-December 2001.

[40] J. Fridrich, M. Goljan, and D. Hogea. Attacking the OutGuess. In J. Dittmann and J. Fridrich, editors, *Proc. of ACM Multimedia and Security (MM & Sec)*, Juan-les-Pins, France, December 6 2002.

[41] J. Fridrich, M. Goljan, and D. Hogea. Steganalysis of JPEG Images: Breaking the F5 Algorithm. In F. A. P. Petitcolas, editor, *Proc. of Information Hiding, 5th International Workshop*, volume LNCS 2578, pages 310–323, Noordwijkerhout, The Netherlands, October 7-9 2002.

[42] J. Fridrich, M. Goljan, P. Lisoněk, and D. Soukal. Writing on Wet Paper. *IEEE Transactions on Signal Processing*, 53(10):3923–3935, 2005.

[43] J. Fridrich, M. Goljan, P. Lisoněk, and D. Soukal. Writing on Wet Paper. In E. J. Delp and P. W. Wong, editors, *Proc. of SPIE Security and Watermarking of Multimedia Contents VII*, volume 5681, pages 328–340, San Jose, CA, January 17-20 2005.

[44] J. Fridrich, M. Goljan, and D. Soukal. Perturbed Quantization Steganography with Wet Paper Codes. In J. Dittmann and J. Fridrich, editors, *Proc. of ACM Multimedia and Security (MM & Sec)*, pages 4–15, Magdeburg, Germany, September 20-21 2004.

[45] J. Fridrich, M. Goljan, and D. Soukal. Efficient Wet Paper Codes. In M. Barni, J. Herrera-Joancomartí, S. Katzenbeisser, and F. Pérez-González, editors, *Proc. of Information Hiding, 7th International Workshop*, volume LNCS 3727, pages 204–218, Barcelona, Spain, June 6-8 2005.

[46] J. Fridrich, M. Goljan, and D. Soukal. Perturbed Quantization Steganography with Wet Paper Codes. *Multimedia Systems*, 11(2):98–107, 2005.

[47] J. Fridrich, M. Goljan, and D. Soukal. Steganography via Codes for Memory with Defective Cells. In G. Dellurud, editor, *Proc. of 43th Annual Allerton Conference on Communication, Control, and Computing*, Monticello, IL, September 28-30 2005.

[48] J. Fridrich, M. Goljan, and D. Soukal. Wet Paper Codes with Improved Embedding Efficiency. *IEEE Transactions on Information Forensics and Security*, 1(1):102–110, March 2006.

[49] J. Fridrich, P. Lisoněk, and D. Soukal. On Steganographic Embedding Efficiency. In J. Camenisch, C. Collberg, N. F. Johnson, and P. Sallee, editors, *Proc. of Information Hiding, 8th International Workshop*, volume LNCS 4437, pages 282–296, Alexandria, VA, July 10-12 2006.

[50] J. Fridrich, T. Pevný, and J. Kodovský. Statistically Undetectable JPEG Steganography: Dead Ends, Challenges, and Opportunities. In *Proc. of ACM Multimedia and Security (MM&Sec)*, pages 3–14, Dallas, TX, September 20-21 2007.

[51] J. Fridrich and D. Soukal. Matrix Embedding for Large Payloads. *IEEE Transactions on Information Forensics and Security*, 1(3):390–395, September 2006.

[52] J. Fridrich and D. Soukal. Matrix Embedding for Large Payloads. In E. J. Delp and P. W. Wong, editors, *Proc. of SPIE Security and Watermarking of Multimedia Contents VIII*, volume 6072, page 60721W, San Jose, CA, January 23-27 2006.

[53] J. Fridrich, D. Soukal, and M. Goljan. Maximum Likelihood Estimation of Secret Message Length Embedded Using PMK Steganography in Spatial Domain. In E. J. Delp and P. W. Wong, editors, *Proc. of SPIE Security and Watermarking of Multimedia Contents VII*, volume 5681, pages 595–606, San Jose, CA, January 17-20 2005.

[54] F. Galand and G. Kabatiansky. Information Hiding by Coverings. In *Proc. of IEEE Information Theory Workshop*, pages 151–154, March 31 - April 4 2003.

[55] F. Galand and G. Kabatiansky. Steganography via Covering Codes. In *Proc. of IEEE International Symposium on Information Theory*, page 192, Yokohama, Japan, June 29 - July 4 2003.

[56] R. G. Gallager. Low-Density Parity-Check Codes. *IRE Transactions on Information Theory*, 8(1):21–28, January 1962.

[57] S. I. Gel'fand and M. S. Pinsker. Coding for Channel With Random Parameters. *Problems of Control Theory*, 9(1):19–31, 1980.

[58] P. Günther. Einbettung mittels Message-Passing Algorithmen. Master's thesis, Technische Universität Dresden, Juni 2007. Supervisor: D. Schönfeld and A. Winkler.

[59] P. Günther, D. Schönfeld, and A. Winkler. Efficient Implementation of Belief Propagation for Lossy Source Compression with LDGM Codes. In *Proc. of 7th International ITG Conference on Source and Channel Coding (SCC'08)*, Ulm, Germany, January 14-16 2008. VDE Verlag.

[60] P. Günther, D. Schönfeld, and A. Winkler. Reduced Embedding Complexity using BP Message Passing for LDGM Codes. In E. J. Delp, P. W. Wong, J. Dittmann, and N. D. Memon, editors, *Proc. of SPIE Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*, volume 6819, page 681919, San Jose, USA, January 28 2008.

[61] R. W. Hamming. Error Detecting and Error Correcting Codes. *The Bell System Technical Journal*, 26(2):147–160, April 1950.

[62] T. Holotyak, J. Fridrich, and D. Soukal. Stochastic Approach to Secret Message Length Estimation in PMK Embedding Steganography. In E. J. Delp and P. W. Wong, editors, *Proc. of SPIE Security and Watermarking of Multimedia Contents VII*, volume 5681, pages 673–684, San Jose, CA, January 17-20 2005.

[63] N. J. Hopper, J. Langford, and L. von Ahn. Provably Secure Steganography. In M. Yung, editor, *Advances in Cryptology: CRYPTO 2002*, volume LNCS 2442, pages 77–92, Santa Barbara, CA, August 18-22 2002. Springer.

[64] X.-Y. Hu, A. E. Eleftheriou, and A. Dholakia. Efficient Implementations of the Sum-Product Algorithm for Decoding LDPC Codes. In *Proc. of IEEE Global Telecommunications Conference (GLOBECOM)*, volume 2, pages 1036–1036E, San Antonio, TX, November 25-29 2001.

[65] S. Katzenbeisser and F. A. P. Petitcolas. Defining Security in Steganographic Systems. In E. J. Delp and P. W. Wong, editors, *Proc. of SPIE Security and Watermarking of Multimedia Contents IV*, volume 4675, pages 50–56, San Jose, CA, January 2002.

[66] A. Ker, T. Pevný, J. Kodovský, and J. Fridrich. The Square Root Law of Steganographic Capacity. In *Proc. of ACM Multimedia and Security (MM & Sec)*, pages 107–116, Oxford, UK, September 22-23 2008.

[67] A. Kerckhoffs. La Cryptographie Militaire. *Journal des sciences militaires*, 9:5–38,161–191, January and February 1883.

[68] M. Kharrazi, H. T. Sencar, and N. D. Memon. Benchmarking Steganographic and Steganalysis Techniques. In E. J. Delp and P. W. Wong, editors, *Proc. of SPIE Security, Steganography, and Watermarking of Multimedia Contents VII*, volume 5681, pages 252–263, San Jose, CA, January 17-20 2005.

[69] Y. Kim, Z. Duric, and D. Richards. Modified Matrix Encoding Technique for Minimal Distortion Steganography. In J. L. Camenisch, C. S. Collberg, N. F. Johnson, and P. Sallee, editors, *Proc. of Information Hiding, 8th International Workshop*, volume LNCS 4437, pages 314–327, Alexandria, VA, July 10-12 2006.

[70] H. Klimant, R. Piotraschke, and D. Schönfeld. *Informations- und Kodierungstheorie*. Teubner Verlag Wiesbaden, 3rd edition, 2006.

[71] J. Kodovský and J. Fridrich. Influence of Embedding Strategies on Security of Steganographic Methods in the JPEG Domain. In *Proc. of SPIE Security, Forensics, Steganography, and Watermarking of Multimedia Contents X*, volume 6819, page 681902, San Jose, CA, January 2008.

[72] B. J. Koops. Summary of International Crypto Controls. http://rechten.uvt.nl/koops/cryptolaw/cls-sum.htm.

[73] A. V. Kuznetsov and B. S. Tsybakov. Coding in a Memory with Defective Cells. *Problems of Information Transmission*, 10(2):132–138, 1974.

[74] B. A. LaMacchia and A. M. Odlyzko. Solving Large Sparse Linear Systems over Finite Fields. In A. J. Menezes and S. A. Vanstone, editors, *Proc. of Advances in Cryptology-CRYPT0' 90*, volume LNCS 537, pages 109–133. Springer, 1991.

[75] E. Lausanne. LDPC Opt Online Degree Distribution Optimizer. http://lthcwww.epfl.ch/research/ldpcopt/, 2007.

[76] S. Lin and D. J. Costello. *Error Control Coding*. Person Education, 2nd edition, 2004.

[77] M. Luby. LT Codes. In *Proc. of the 43rd Symposium on Foundations of Computer Science*, pages 271–282. IEEE, 2002.

[78] R. H. Morelos-Zaragoza. *The Art of Error Correcting Coding*. John Wiley & Sons, 2005.

[79] T. Pevný, T. Filler, and P. Bas. Using High-Dimensional Image Models to Perform Highly Undetectable Steganography. In R. Böhme, P. W. L. Fong, and R. Safavi-Naini, editors, *Proc. of Information Hiding, 12th International Workshop*, volume LNCS 6387, pages 161–177, Calgary, Canada, June 28-30 2010.

[80] B. Pfitzmann. Information Hiding Terminology. In R. Anderson, editor, *Proc. of Information Hiding, 1st International Workshop*, volume LNCS 1174, pages 347–350, Cambridge, UK, May 30-June 1 1996.

[81] N. Provos. Defending Against Statistical Steganalysis. In *Proc. of 10th USENIX Security Symposium*, volume 10, pages 24–24, 2001.

[82] G. Richter, G. Schmidt, M. Bossert, and E. Costa. Optimization of a Reduced-Complexity Decoding Algorithm for LDPC Codes by Density Evolution. In *Proc. of IEEE International Conference on Communications*, volume 1, pages 642–646, May 16-20 2005.

[83] V. Sachnev, H. J. Kim, and R. Zhang. Less Detectable JPEG Steganography Method Based on Heuristic Optimization and BCH Syndrome Coding. In *Proc. of ACM Multimedia and Security (MM & Sec)*, pages 131–139, Princeton, NJ, September 7-8 2009.

[84] P. Sallee. Model-Based Steganography. In T. Kalker, I. J. Cox, and Y. M. Ro, editors, *Proc. of Digital Watermarking IWDW*, volume LNCS 2939, pages 254–260, Seoul, Korea, October 20-22 2003.

[85] D. Schönfeld. Einbetten mit minimaler Werksänderung. In *Datenschutz und Datensicherheit (DuD)*, pages 666–671. Vieweg Verlag, 2001.

[86] D. Schönfeld and A. Winkler. Embedding with Syndrome Coding Based on BCH Codes. In *Proc. of ACM Multimedia and Security (MM & Sec)*, pages 214–223, Geneva, Switzerland, September 26-27 2006.

[87] D. Schönfeld and A. Winkler. Reducing the Complexity of Syndrome Coding for Embedding. In T. Furon, F. Cayre, G. Doërr, and P. Bas, editors, *Proc. of Information Hiding, 9th International Workshop*, volume LNCS 4567, pages 145–158, Saint Malo, France, June 11-13 2007.

[88] S. Schulze. Einbettung mit (verketteten) LDGM Kodes. Master's thesis, Technische Universität Dresden, July 2009. Supervisor: D. Schönfeld.

[89] T. Sharp. An Implementation of Key-Based Digital Signal Steganography. In I. S. Moskowitz, editor, *Proc. of Information Hiding, 4th International Workshop*, volume LNCS 2137, pages 13–26, Pittsburgh, PA, April 25-27 2001.

[90] G. J. Simmons. The Prisoner's Problem and the Subliminal Channel. In D. Chaum, editor, *Proc. of Advances in Cryptology: CRYPTO' 83*, pages 51–67, 1983.

[91] K. Sullivan, U. Madhow, B. S. Manjunath, and S. Chandrasekaran. Steganalysis for Markov Cover Data with Applications to Images. *IEEE Transactions on Information Forensics and Security*, 1(2):275–287, June 2006.

[92] P. Sweeney. *Error Control Coding - From Theory to Practice*. John Wiley & Sons, 2002.

[93] M. van Dijk and F. Willems. Embedding Information in Grayscale Images. In *Proc. of 22nd Symposium on Information and Communication Theory*, pages 147–154, Enschede, The Netherlands, May 15-16 2001.

[94] M. J. Wainwright and E. Maneva. Lossy Source Encoding via Message-Passing and Decimation over Generalized Codewords of LDGM Codes. In *Proc. of International Symposium on Information Theory*, pages 1493–1497, Adelaide, SA, September 4-9 2005.

[95] A. Westfeld. *Prinzipien sicherer Steganographie*. PhD thesis, Technische Universität Dresden, 2000. Supervisor: A. Pfitzmann.

[96] A. Westfeld. F5 - A Steganographic Algorithm - High Capacity Despite Better Steganalysis. In I. S. Moskowitz, editor, *Proc. of Information Hiding, 4th International Workshop*, volume LNCS 2137, pages 289–302, Pittsburgh, PA, April 25-27 2001.

[97] A. Westfeld. Steganography for Radio Amateurs–A DSSS Based Approach for Slow Scan Television. In J. Camenisch, C. Collberg, N. F. Johnson, and P. Sallee, editors, *Proc. of Information Hiding, 8th International Workshop*, volume LNCS 4437, pages 201–215, Alexandria, VA, July 10-12 2006.

[98] A. Westfeld and A. Pfitzmann. Attacks on Steganographic Systems - Breaking the Steganographic Utilities EzStego, Jsteg, Steganos, and S-Tools - and Some Lessons Learned. In A. Pfitzmann, editor, *Proc. of Information Hiding, 3rd International Workshop*, volume LNCS 1768, pages 61–76, Dresden, Germany, September 29 - October 1 2000.

[99] D. Wiedemann. Solving Sparse Linear Equations over Finite Fields. *IEEE Transactions on Information Theory*, 32(1):54–62, 1986.

[100] F. J. M. Williams and N. J. Sloane. *The Theory of Error-Correcting Codes*. North-Holland, Amsterdam, 1977.

[101] R. Zhang, V. Sachnev, and H. J. Kim. Fast BCH Syndrome Coding for Steganography. In S. Katzenbeisser and A.-R. Sadeghi, editors, *Proc. of Information Hiding, 11th International Workshop*, volume LNCS 5806, pages 48–58, Darmstadt, Germany, June 8-10 2009.

[102] W. Zhang, J. Liu, X. Wang, and N. Yu. Generalization and Analysis of the Paper Folding Method for Steganography. *IEEE Transactions on Information Forensics and Security*, 5(4):694–704, December 2010.

[103] W. Zhang, S. Wang, and X. Zhang. Improving Embedding Efficiency of Covering Codes for Applications in Steganography. *IEEE Communications Letters*, 11(8):680–682, August 2007.

[104] W. Zhang and X. Wang. Generalization of the ZZW Embedding Construction for Steganography. *IEEE Transactions on Information Forensics and Security*, 4(3):564–569, September 2009.

[105] W. Zhang, X. Zhang, and S. Wang. Maximizing Steganographic Embedding Efficiency by Combining Hamming Codes and Wet Paper Codes. In K. Solanki, K. Sullivan, and U. Madhow, editors, *Proc. of Information Hiding, 10th International Workshop*, volume LNCS 5284, pages 60–71, Santa Barbara, CA, May 19-21 2008.

[106] W. Zhang and X. Zhu. Improving the Embedding Efficiency of Wet Paper Codes by Paper Folding. *IEEE Signal Processing Letters*, 16(9):794–797, September 2009.

[107] X. Zhang, W. Zhang, and S. Wang. Efficient Double-Layered Steganograpic Embedding. *Electronic Letters*, 43(8):482–483, April 2007.

[108] Z. Zhao, F. Wu, S. Yu, and J. Zhou. A Lookup Table Based Fast Algorithm for Finding Roots of Quadratic or Cubic Polynomials in the GF(2^m). *Journal of Huazhong University of Science and Technology (Nature Science Edition)*, 33(2):70, April 2005.