Finite-size effects in low-density parity-check codes applied to cryptography

FRÉDÉRIC LACOMBE

MSc by Research in Pattern Analysis and Neural Networks

ASTON UNIVERSITY

September 2001

ASTON UNIVERSITY

Finite-size effects in low-density parity-check codes applied to cryptography

FRÉDÉRIC LACOMBE

MSc by Research in Pattern Analysis and Neural Networks, 2001

Thesis Summary

Low density parity check error-correcting codes can be applied as an efficient public key cryptosystem. The scheme, based on the computational difficulty in factorising dense matrices, has been studied so far in the limit of infinite system size. In this work we study numerically the finite-size effects in such a system, such that may have a significant impact on the reliability of the cryptosystem, and we derive the corresponding practical limitations in using these algorithms. We also describe the difficulties we encountered in using a similar approach to devise an electronic signature cryptosystem.

Acknowledgements

I would like to thank particularly Professor David Saad and Doctor Jort van Mourik for their helpful supervision, all the time they have taken for discussions, and specifically David Saad for the large amount of advice he gave me to achieve this thesis.

I am grateful to all the people in the Neural Computing Research Group for their friendly and dynamic contacts. Particularly, I would like to thank my lecturers Professor D. Lowe, Professor D. Saad, Doctor M. Opper, Doctor I. Nabney and Doctor D. Barber, for their so clear approach to neural nets.

I would like to cite also the MSc students in PANN, Pierre, Stéphane and Bert' as well as the PhD students from the NCRG, Lehel and Wei Lee for their friendship for all this year in Birmingham.

I am grateful to all my friends for their support, Argiro, Jacinto, Bhupi, Peyman, Rosana, Joaquín and Claire to share my life in Aston, Lilian and Fabrice from Manchester for their visits, Thierry and Stéphanie for the correction throughout this paper, and last but not least Virginie, Vanessa, Mickael and Fleur to support me for so many years.

Finally, I want to thank my parents for their love and their financial support to make it possible. I want to dedicate this work to my young niece Eva who came to life in the same time as this thesis.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Secure data transmission

The increase in digital communication over the last thirty years was followed by the development of sophisticated methods of monitoring communication, falsifying identities and corrupting messages. Therefore, integrity, authenticity and confidentiality became dominant in modern data transmission and give rise to rapid development in various aspects of cryptography.

Nowadays, it is common to use secure media, or supposedly secure media, for many daily activities such as credit card information transfer, mobile communication, online banking or e-commerce to name but a few. Data encryption and digital signatures provide an efficient solution to the security problem, since the development of public key cryptosystems.

Many different algorithms have been developed for public key cryptography, applying specific mathematical theories ranging from Galois fields to prime numbers factorisation, and information theory approaches. After the recent rediscovery of Gallager error-correcting codes [13, 14], they have been applied as a public key cryptosystem with encouraging results [7]. These codes, based on the difficulty of factorising a product of Boolean matrices can also be analysed by using the methods of statistical mechanics; the analysis supports conjectures about their expected performance.

1.2 Outline

In the following chapter, low-density parity-check codes will be described, especially the construction used in this work based on random sparse matrices. Furthermore, some information theory background will be discussed as it is required for the forthcoming chapters. The belief propagation algorithm will also be presented within the Bayesian framework and linked to optimal estimators and decoding/decrypting error robabilities.

In chapter 3, we introduce the use of low-density parity-check codes as a public key cryptosystem. Emphasis will be given to secure data transfer, describing the main encryption/decryption scheme in the specific system examined here. A discussion about the limitation of the cryptosystem will naturally lead to the study of finite-size effects.

In chapter 4, we will study the role of finite-size effects and their relevance to secure data transmission. The study will be based on a numerical approach and will concentrate on the practical limits of the system as it emerges from the analysis. Then, the agreement between theory and the numerical results will be pointed out.

Chapter 5 introduces the problem of digital signature, explaining the main possible attacks, followed by a discussion of the different possible solutions. We will explain the different approaches suggested, and their weaknesses or drawbacks.

Finally, the main insight gained from the thesis will be summarised in chapter 6, and future research directions will be indicated.

Chapter 2

Low-density parity-check codes

In this chapter, I will present the basic definitions, coming from information theory, which are relevant to the area of error-correcting codes, as well as a description of the probability propagation algorithm. Specific code structures coming from a statistical mechanics analysis will then be discussed and motivated, forming the basis of the public key cryptosystem as introduced in the following chapter.

2.1 Information theory

First defined for communication purposes, information theory provides the main mathematical tool and formulation of error-correcting codes [12]. As the suggested cryptosystem is derived from error-correcting codes, I will provide some useful definitions and the main framework to be used in the following chapters. For convenience, all descriptions will use Boolean algebra $\{0, 1, +\}$, as we can transform any message to a binary version $\{1, -1, .\}$.

2.1.1 Entropy

A message is a sequence of symbols $(a_i)_{i\in[1,m]}$ from an alphabet $\mathcal{A}$. A probability is associated with each symbol of the alphabet such that $\sum_{a_i\in\mathcal{A}} p(a_i) = 1$. The entropy, which measures the effective amount of information in a message is defined as follows:

Definition 1 (Entropy) *Let a message be of length m*

$$A = \begin{pmatrix} a_1 & \cdots & a_i & \cdots & a_m \\ p(a_1) & \cdots & p(a_i) & \cdots & p(a_m) \end{pmatrix}$$

The entropy of the message is defined as:

$$H_2(A) = -\sum_{i=1}^{m} p(a_i)\log_2(p(a_i)). \tag{2.1}$$

In a binary representation, the alphabet is reduced to $\mathcal{A} = \{0, 1\}$ and the binary entropy to $H_2(A) = H_2(p) = -p\log_2(p) - (1-p)\log_2(1-p)$ where p is the probability of the symbol being 1.

2.1.2 Transmission through a channel

In error-correcting codes, our main interest is transmission through a noisy channel such that we will be able to retrieve the original message after corruption. The noisy channel can be interpreted as a conditional probability $P(\boldsymbol{J}|\boldsymbol{J}^0)$, where $\boldsymbol{J}$ is the received message and $\boldsymbol{J}^0$ the transmitted one. One can define the conditional entropy for $\mathcal{T}$ and $\mathcal{R}$, the two alphabets of the transmitted and received vectors respectively:

$$H_2(\boldsymbol{J}^0|\boldsymbol{J}) = -\sum_{J_i\in\mathcal{R}}\sum_{J_j^0\in\mathcal{T}} P(J_i)P(J_j^0|J_i)\log(P(J_j^0|J_i)) \tag{2.2}$$

The mutual information represents the information on the original signal $\boldsymbol{J}^0$ conveyed by the received signal $\boldsymbol{J}$, and is defined by:

$$I(\boldsymbol{J}^0, \boldsymbol{J}) = H_2(\boldsymbol{J}^0) - H_2(\boldsymbol{J}^0|\boldsymbol{J}) \tag{2.3}$$

Therefore, the channel capacity is introduced as the maximal information that a channel can retain.

Definition 2 (Channel capacity)

$$C_{channel} = \max_{P(J_j^0)} I(\boldsymbol{J}^0, \boldsymbol{J})$$

where $I(\boldsymbol{J}^0, \boldsymbol{J})$ is defined as a functional of the probability of transmitted bits $P(J_j^0)$ and the corruption process.

In this thesis, we only consider binary symmetric channels (BSC). Both alphabets are binary, the conditional probability corresponds to a flipping rate: $P(J_i = 0 \mid J_j^0 = 1) = p$ or $P(J_i = 1 \mid J_j^0 = 0) = p$. For the BSC, the channel capacity is equal to $C_{\text{BSC}} = 1 - H_2(p)$.

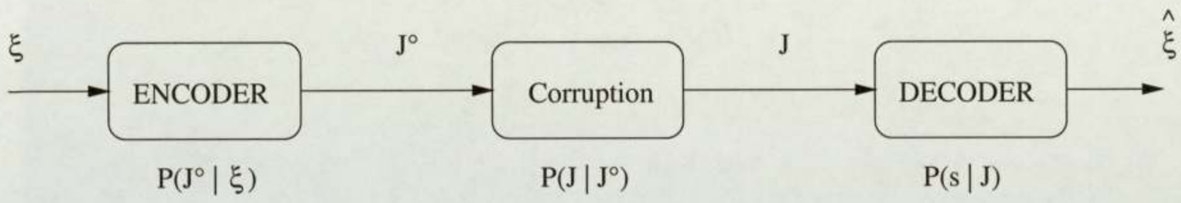

Figure 2.1: Main scheme of message transmission. The message $\boldsymbol{\xi}$ is encoded into a codeword $\boldsymbol{J}^0$. The transmission through a noisy channel gives rise to the received corrupted message $\boldsymbol{J}$. Finally, the decoding process extracts from $\boldsymbol{J}$ the most likely original message $\hat{\boldsymbol{\xi}}$. $\boldsymbol{s}$ represents the dynamic variable used in the previous process.

2.1.3 Encoding and redundancy

Successful decoding after corruption is achievable due to the addition of some redundancy to the original message $\boldsymbol{\xi}$, creating the codeword $\boldsymbol{J}^0$ to be transmitted. Defining the lengths of the vectors $\boldsymbol{\xi}$ and $\boldsymbol{J}^0$ as N and M respectively, the definition of the code rate is thus $R = N/M$ in the case of unbiased messages. Therefore, a code represents the mapping of 2^N words of N bit sequences onto M bit codewords, denoted $(2^N, M)$. Encoding consists of applying a function e such that $e(\boldsymbol{\xi}) = \boldsymbol{J}^0$, decoding results from the inverse process with a function d such that $d(\boldsymbol{J}) = \boldsymbol{J}^0$. One may consider the probability of block error, p_B, defined as the probability for any symbol to be decoded incorrectly,

$$P_B = P(d(\boldsymbol{J}) \neq \boldsymbol{J}^0) \tag{2.4}$$

Shannon's channel coding theorem can be expressed in the following way:

Theorem 1 (Channel coding) *For every rate R below the channel capacity $C_{channel}$, there exists a mapping $(2^N, M)$ with $p_B \to 0$ when $M \to \infty$. Conversely, for any set of code words mapping $(2^N, M)$ with $p_B \to 0$ when $M \to \infty$, R is below the channel capacity.*

From this fundamental theorem, a theoretical limit is defined for all data transmission. For a given rate, codes can only be applied up to a determined level of corruption. Shannon's limit has been obtained in the infinite system size, practical systems are more restricted due to finite-size effects.

2.2 Low-density parity-check codes

2.2.1 Linear codes

The encoding process uses an $M \times N$ binary matrix G, called generator matrix, such that $\boldsymbol{J}^0 = G\boldsymbol{\xi}$. Hence all $\boldsymbol{J}_i^0$ are a linear binary combinations of the $\boldsymbol{\xi}_i$, considering (mod 2) operations:

$$J_i^0 = \sum_{j \in \mathcal{K}(i)} \xi_j \tag{2.5}$$

where $\mathcal{K}(i)$ represents the set of non-zero elements in i-th row of G.

After corruption by a vector $\boldsymbol{\zeta}$, we receive the vector $\boldsymbol{J} = \boldsymbol{J}^0 + \boldsymbol{\zeta}$. For Gallager algorithm decoding process, we first generate the syndrome vector $\boldsymbol{z}$ such that $\boldsymbol{z} = H\boldsymbol{J}$, H is usually called the parity-check matrix and has the property $HG = 0$. Then we get $H\boldsymbol{J} = H\boldsymbol{\zeta} = \boldsymbol{z}$. Various low-density parity-check codes have been studied starting from Gallager original code [4]. The variation of low-density parity-check codes that we focus on here is the MN code in which we use two binary random sparse matrices A and B such that:

$$G = B^{-1}A \tag{2.6}$$

Thus we choose $H = [A|B]$ to find back the same scheme as Gallager codes: the syndrome vector is given by $\boldsymbol{z} = B\boldsymbol{J} = A\boldsymbol{\xi} + B\boldsymbol{\zeta} = H\boldsymbol{x}$, where $\boldsymbol{x}^T = [\boldsymbol{\xi}|\boldsymbol{\zeta}]^T$. The decoding results thus are obtained by solving a linear system [13], looking for the most probable signal $\boldsymbol{s}$ and noise $\boldsymbol{\tau}$ vectors which obey $\boldsymbol{z} = A\boldsymbol{s} + B\boldsymbol{\tau}$.

The two matrices A and B are defined by their structures and their sparseness: A has K non-zero unit elements per row and $C(= K/R)$ per column while B has L non-zero unit elements per row and per column.

2.2.2 Decoding efficiency

The decoding process is efficient due to the sparseness of the matrices A and B. As we choose the matrices to be regular, i.e. with a fixed number of non-zero elements per row and per column, we can categorise the performance of the various constructions with respect to these parameters. These codes have been analysed using methods adapted from spin glasses and mean field theory [8, 10, 15], which results in three distinct classes in the case of unbiased messages:

- For $K \geqslant 3$, one could ideally decode successfully up to Shannon's limit. However, practically one cannot use these constructions as the basin of attraction around the perfect solution is very narrow, resulting in a decoding failure when a practical decoding algorithm is used.

- The case $K = 2$ is marked by the appearance of a point below which only the perfect solution exists and above which other solutions emerge, determining the practical limitation. Decoding to the perfect solution is feasible up to this point, above it the convergence is possible only if exhaustive search algorithms are used. Typically, practical decoding methods will converge to sub-optimal solutions which may obey the parity checks but will differ from the initial vector.

- Finally, for $K = 1$, the picture is similar to that of $K = 2$ but the codes' performance is generally lower. In some cases these codes perform better than $K = 2$ codes, mainly in high code rates. For more details, see [15].

Using irregular codes, i.e. defining the sparse matrices as block matrices with different distributions of non-zero bits per row and per column, the codes efficiency can be improved [21]. Advantages from each of the previous regular cases can be exploited to improve the performance. Ideal constructions could be obtained by extremizing the noise value below which only the optimal solution exists, pushing it as close as possible to Shannon's limit.

2.3 Statistical decoding

2.3.1 The Bayesian approach

After encoding the data by adding redundancy into the vector $\boldsymbol{J}^0$, the latter is then corrupted by a noise. The received vector, $\boldsymbol{J}$, is therefore given by $\boldsymbol{J} = \boldsymbol{J}^0 + \boldsymbol{\zeta}$ where $\boldsymbol{\zeta}$ represents the noise added to the encoded data.

The decoding scheme relies on finding the most likely estimate $\hat{\boldsymbol{\xi}}$ assuming a model for the corruption $P(\boldsymbol{J}|\boldsymbol{J}^0)$ and a prior for the original data $P(\boldsymbol{\xi})$. This estimator should minimise on average a loss function related to the posterior distribution $P(\boldsymbol{\xi}|\boldsymbol{J})$.

1. One commonly used measure is the percentage of perfectly retrieved messages which correspond to the minimisation of the block error probability. The loss function rewards any single bit failure as an error in retrieval:

$$L(\boldsymbol{\xi}, \hat{\boldsymbol{\xi}}) = 1 - \prod_{i=1}^{N} \delta(\xi_i, \hat{\xi}_i) \tag{2.7}$$

   where δ refers to the Kronecker tensor.

   The posterior probability is obtained by applying Bayes theorem and noticing that $P(\boldsymbol{J}|\boldsymbol{\xi}) = P(\boldsymbol{J}|\boldsymbol{J}^0)\delta(\boldsymbol{J}^0, e(\boldsymbol{\xi}))$:

$$P(\boldsymbol{\xi}|\boldsymbol{J}) = \frac{P(\boldsymbol{J}|\boldsymbol{J}^0)\delta(\boldsymbol{J}^0, e(\boldsymbol{\xi}))P(\boldsymbol{\xi})}{\sum_{\xi} P(\boldsymbol{J}|\boldsymbol{J}^0)\delta(\boldsymbol{J}^0, e(\boldsymbol{\xi}))P(\boldsymbol{\xi})} \tag{2.8}$$

where $\delta(\boldsymbol{x}, \boldsymbol{y}) = \prod_{i=1}^{M} \delta(x_i, y_i)$.

Therefore, the optimal estimator is defined by the maximal a posteriori estimator (MAP): $\hat{\boldsymbol{\xi}} = \mathrm{argmax}_{\boldsymbol{\xi}} P(\boldsymbol{\xi}|\boldsymbol{J})$. This estimator chooses the global maximum of the posterior over a space of 2^N solutions. This is in general a hard computational problem, but it can be solved in some cases by using the belief propagation/revision algorithms.

2. Another measure is given by the overlap between the original message and the decoded vector, defining the bit error probability as the percentage of well-retrieved bits in the message. This gives rise to a different loss function, using a binary alphabet $\xi_i \in \{-1, 1\}$:

$$L(\boldsymbol{\xi}, \hat{\boldsymbol{\xi}}) = -\sum_{i=1}^{N} \xi_i \hat{\xi}_i \tag{2.9}$$

The optimal estimator for this loss function is the marginal posterior maximiser (MPM), defined by $\hat{\xi}_j = \mathrm{argmax}_{\xi_j} P(\xi_j|\boldsymbol{J})$, with $P(\xi_j|\boldsymbol{J}) = \sum_{\{\xi_i, i\neq j\}} P(\boldsymbol{\xi}|\boldsymbol{J})$ [6].

2.3.2 Belief propagation/revision algorithm

To evaluate the a posteriori estimator or the marginal posterior, one should obtain the probabilistic dependencies between variables. This can be carried out by mapping the system onto a bipartite graph, one layer corresponding to the M received bits z_μ, and the other to the N estimated message bits s_j. The connections between the two layers are specified by the matrices A and B. The computational improvement using this algorithm is significant, as the process is based on passing probabilities forward from the source layer to the receiving layer and then backward iteratively until convergence, which takes $\mathcal{O}(N)$ operations in comparison to the $\mathcal{O}(2^N)$ operations generally required for obtaining the exact solution.

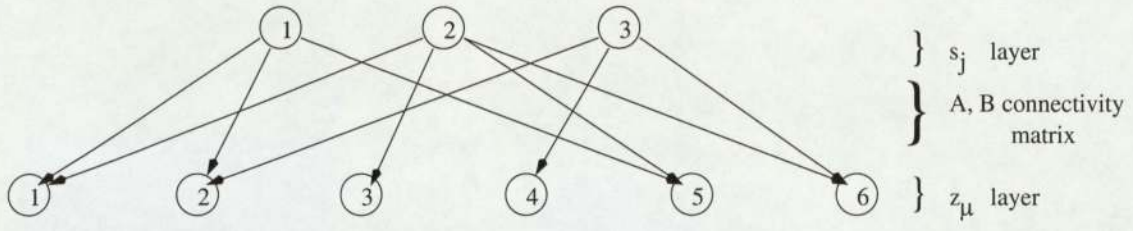

Figure 2.2: Bayesian network with a rate $R = 1/2$. Each z_μ represents a syndrome bit and each s_j stands for a bit of the original message. A and B are defined as the adjacency matrices or connectivity matrices between the nodes of the network.

These algorithms can be proven to give an exact solution only in case of a tree-like graph (graph without any loops). However, they should converge to a good approximation in general graphs even in the presence of small loops [22].

Now, we will explain how the algorithm works, for the example of figure 2.2. The joint probability is given by:

$$P(s_1, ..., s_3, z_1, ..., z_6) = P(z_6|s_3, s_2)P(z_5|s_2, s_1)P(z_4|s_3)P(z_3|s_2)P(z_2|s_3, s_1)P(z_1|s_2, s_1)$$

and the connectivity matrix is defined as follows:

$$G = \begin{pmatrix} 1 & 1 & 0 \\ 1 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 1 & 1 \end{pmatrix}$$

Given the received information z_μ, one has to recover all the $P(s_j)$ in order to estimate the initial message. In this algorithm, we do not have to compute all terms as we exploit the explicit dependencies between nodes. To compute the pseudo posterior marginal we arrange the system as a bipartite graph, iterating conditional probability of single variables with respect to others back and forth until a consistent solution is obtained. The main scheme is described as follows:

- First, we call s_j and z_μ layers the estimate and syndrome layers respectively.

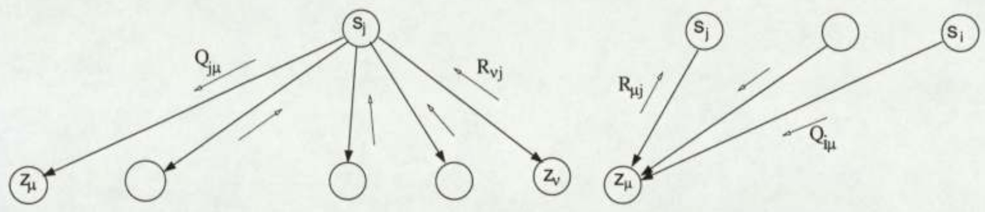

Figure 2.3: Left part: forward message Q, from the estimate layer to the syndrome layer. Right part: backward message R, from the syndrome layer to the estimate layer.

- At each step, each estimate node s_j sends forward the message $Q_{j\mu}$ to the syndrome z_μ to approximate the node's belief given the messages received.
- In the same way, each syndrome node z_μ sends backward the message $R_{j\mu}$ to each estimate s_j.
- The process is iterated until convergence.

Initially, each $Q_{j\mu}$ is set to the corruption probability p_j. The update rules are given by:

$$R^a_{j\mu} = \sum_{s_i | i \in \mathcal{N}(\mu)\setminus j} P(z_\mu | s_j = a, s_i i \in \mathcal{N}(\mu) \setminus j) \prod_{i \in \mathcal{N}(\mu)\setminus j} Q^{s_i}_{i\mu} \tag{2.10}$$

where $\mathcal{N}(\mu) \setminus j$ refers to the set of indices of the nodes in the estimate layer that are linked to the node z_μ with exception of the $j - th$ node, and $a \in \{0, 1\}$.

$$\begin{aligned} Q^a_{j\mu} &= P(s_j = a | \{z_\nu\}_{\nu \in \mathcal{M}(j)\setminus\mu}) \\ &= \frac{P(s_j = a) P(\{z_\nu\}_{\nu \in \mathcal{M}(j)\setminus\mu} | s_j = a)}{P(\{z_\nu\}_{\nu \in \mathcal{M}(j)\setminus\mu})} \\ &= \alpha_{j\mu} P(s_j = a) \prod_{\nu \in \mathcal{M}(j)\setminus\mu} R_{\nu j} \end{aligned} \tag{2.11}$$

where $\mathcal{M}(j) \setminus \mu$ stands for the set of indices of the nodes in the syndrome layer that are linked to the node s_j with exception of the node μ and $\alpha_{j\mu}$ is a normalising pre-factor [1]. Iterating equations (2.11) and (2.10) until convergence, we reach an approximation for the marginal posterior $P(s_j = a | \boldsymbol{z})$.

The belief propagation algorithm reduces considerably the computational time required for calculating marginal posteriors. Usually, the belief propagation algorithm is used for directed graphs to compute the MPM estimator while the belief revision procedure is used in undirected graphs to compute the MAP estimator (Markov networks) [22]. Although we are mainly interested in the MAP estimator, we decided to use belief propagation algorithm to employ the MPM estimator as it is easier to use and has better convergence rates. In addition, using MPM, we have a measure for the distance between sub-optimal solutions and the original one.

Chapter 3

Application to cryptography

After describing the low-density parity-check matrices and their use in error-correcting codes, we show how they can be used as a public key cryptosystem. Advantages and drawbacks of the cryptosystem presented, are discussed, as well as practical considerations to improve the reliability of the process.

3.1 Conversion to cryptography

In the previous chapter, an algorithm based on low-density parity-check codes was presented to recover data after corruption due to a transmission through a noisy channel. We will now present a method for using a similar algorithm as a public key cryptosystem. This cryptosystem is described in relation to the scheme of figure 2.1 where $\boldsymbol{\xi}$, representing the original message (plain-text), is encoded into a cipher-text $\boldsymbol{t}$ by generating a codeword using the parity-check matrix G (part of the public key), and corrupting the codeword vector by adding a corruption vector with flipping probability p. Both the probability p and the matrix G define the public key.

The analogy with the previously described error-correcting codes is obvious. On the one hand, the encryption process is similar to the encoding process and the corruption of the initial data. On the other hand, the decryption process is similar to the decoding process when full knowledge of the process is available. The secret key is accessible only to the authorised user and contains the decomposition of the matrix G as well as the flipping probability. In equation (2.6), which represents the encoding/decoding process, the matrix G is defined by two random sparse matrices A and B. However, such a matrix can be decomposed in feasible time scales, due to the sparseness of A and B. Therefore, a dense matrix D is added in the definition of G, as will be explained later on.

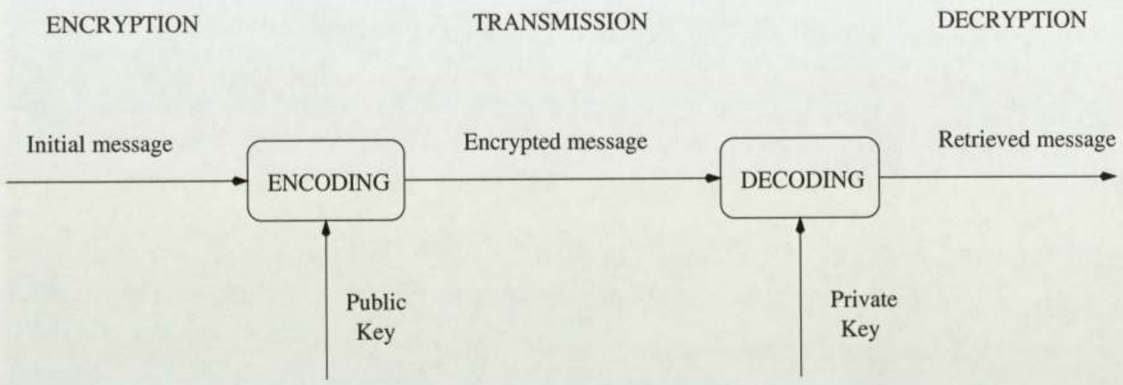

Figure 3.1: In cryptography, the public key is used for encryption. The private key includes extra information, enabling its owner to decrypt the cipher text in practical time scales. The performance of the algorithms relies on the encoding and decoding methods used.

It is worth mentioning that in error-correcting codes, we are typically interested in optimising the overlap between the initial vector and the decoded vector, while in a cryptosystem we would typically be interested in the probability of block error as any single error becomes critical. Computationally, the belief propagation algorithm should be converted to a belief revision procedure as the former corresponds to the marginal probability maximiser (MPM) and the latter to the maximum a posteriori estimator (MAP). These two algorithms are based on similar principles and the adaptation from one to the other is straightforward [22]. Practically, we will be using belief propagation in view of its higher convergence rate and ease of use.

3.2 The public key framework

In a public key cryptosystem, the encryption process as well as the extra information used to encrypt the message, are accessible to all users. Decryption in practical time scales should be impossible for those who do not possess the private key. In some of the commonly used methods, the public key is composed of the elements used in the private key: prime number factorisation in RSA schemes [18], product of matrices for low-density parity-check codes [7, 14].

3.2.1 Encryption

The encryption scheme is based on generating a codeword from the original message using a low-density parity-check matrix G and adding a corruption vector $\boldsymbol{\zeta}$. The corruption vector is defined with a flip rate p, that must stay below p_s, the threshold above which the cryptosystem will not be guaranteed to retrieve the exact solution. Considering the initial message $\boldsymbol{\xi}$, the encrypted message $\boldsymbol{J}$ is obtained using equation (3.1).

$$G = B^{-1}A \tag{3.1}$$

$$\boldsymbol{J} = G\boldsymbol{\xi} + \boldsymbol{\zeta} \tag{3.2}$$

Both the matrix G and the corruption level p define the public key. A and B are two random sparse matrices.

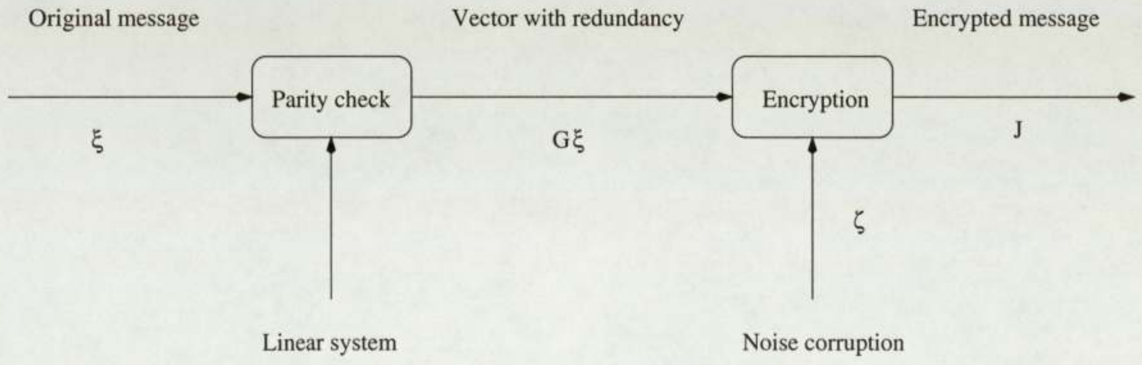

Figure 3.2: In a first step, the initial message is related to a codeword of length M by multiplying $\boldsymbol{\xi}$ by the low-density parity-check matrix G. A corruption vector $\boldsymbol{\zeta}$ is then added to make the decryption difficult for non authorised users. The corruption vector has a flip probability $p < p_s$ and represents a part of the public key.

3.2.2 Decryption

The decryption process uses the belief propagation algorithm to recover the most probable original vector. The process is feasible only due to the sparseness of the matrices. The first step consists in creating a *syndrome* vector z by multiplying the received vector $\boldsymbol{J}$ by the sparse matrix B.

$$\begin{aligned} z = B\boldsymbol{J} &= A\boldsymbol{s} + B\boldsymbol{\tau} \\ &= A\boldsymbol{\xi} + B\boldsymbol{\zeta} \end{aligned} \tag{3.3}$$

We then try to estimate the most likely dynamic variables s and τ by employing the belief propagation iterative technique which will converge to the optimal solution in the infinite system case, which at $p < p_s$ is the only existing solution.

3.2.3 Security

If one defines the public key as $G = B^{-1}A$, the decomposition of the matrix G is computationally achievable in practical time scales. The attacker will be able to find two matrices A' and B' in computational effort which scales as $\mathcal{O}(M^2)$ satisfying $G = (B')^{-1}A'$. Therefore, our opponent will be able to use his home-made key in the same way as the original private key to decrypt the encrypted message, generating the new syndrome vector:

$$\begin{aligned} \boldsymbol{z}' &= B'\boldsymbol{J} \\ &= A'\boldsymbol{\xi} + B'\boldsymbol{\zeta} \end{aligned} \tag{3.4}$$

To remove this weakness, the use of a dense matrix D is necessary to avoid the easy decomposition of the private key. Hence, the definition of the matrix G becomes $G = B^{-1}AD$. Then, the decomposition of G into acceptable components is unfeasible in practical time scales.

As the algorithm we used is based on sparse matrices, we need to point out that this dense matrix D does not affect our decryption process. The public key remains the

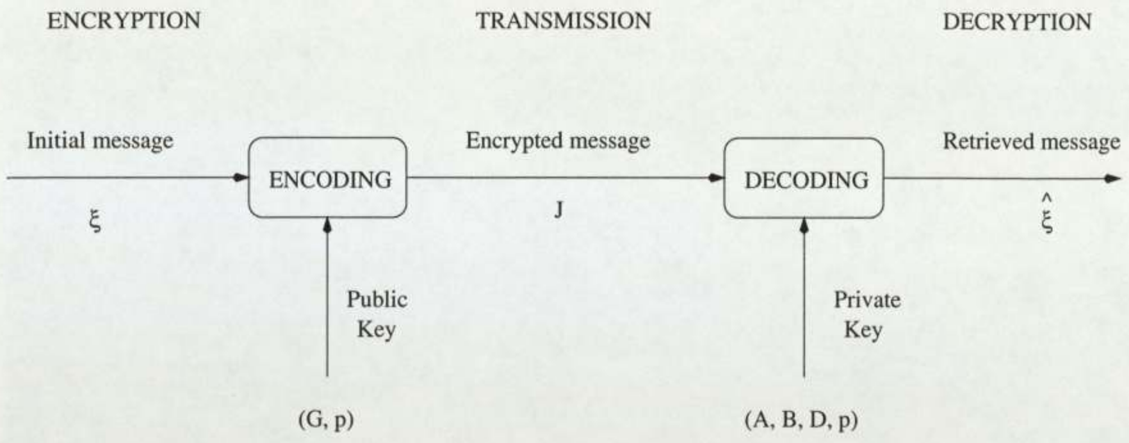

Figure 3.3: This figure describes the suggested cryptosystem. The encryption of the original vector $\boldsymbol{\xi}$ in a vector J is carried out using the public key (composed of the parity-check matrix G and the corruption probability p). The decryption procedure by an authorised user makes use of the secret key (A, B, D, p) where A and B are random sparse matrices and where D is a dense matrix such that $G = B^{-1}AD$. The result of the decryption is the estimated vector $\hat{\boldsymbol{\xi}}$ guaranteed to have a perfect overlap with the initial message in the long system limit.

same, (G,p) where $G = B^{-1}AD$. The length of this key is not affected as B^{-1} is already a dense matrix (the inverse of a sparse matrix is dense) and the encryption algorithm is unchanged. The private key becomes (A, B, D, p). Therefore, the decryption process is applied in the similar way. We receive a message $\boldsymbol{J} = G\boldsymbol{\xi} + \boldsymbol{\zeta}$, our syndrome vector is defined as $\boldsymbol{z} = B\boldsymbol{J} = A(D\boldsymbol{\xi}) + B\boldsymbol{\zeta}$. Thus, it is clear that the decryption algorithm is performed exactly as previously to retrieve a pseudo-decrypted message $\boldsymbol{\xi}' = D\boldsymbol{\xi}$, to achieve the message retreival, we conclude multiplying $\boldsymbol{\xi}'$ by D^{-1}.

3.3 Characteristics of the cryptosystem

3.3.1 Decoding complexity

Considering the decryption, the complexity of the belief propagation algorithm is $\mathcal{O}(N)$ times the number of iterations. Practically, this number was usually less than 100 except close to the critical point. However, one should also multiply the decrypted value by the inverse of D, what should be done by $\mathcal{O}(N^2)$ operations in general and $\mathcal{O}(N\log(N))$ in special cases. One should compare it to RSA, which takes $\mathcal{O}(N^3)$

operations for the decryption procedure. For encryption, a matrix multiplication is necessary and will require $\mathcal{O}(N^2)$ operations. The use of particular matrices as bi-diagonal matrices can reduce the inversion to $\mathcal{O}(N)$, but such simple matrices may slightly compromise the security of the codes. Note that the algorithm may be improved using parallelisation that will lead to a decryption process of complexity $\mathcal{O}(1)$.

3.3.2 Advantages and drawbacks

In order to evaluate the performance of the cryptosystem, we will compare it to the most commonly used cryptosystem, RSA. We notice as an advantage that the decryption is time $\mathcal{O}(N \log N)$ for our cryptosystem, while for RSA it is $\mathcal{O}(N^3)$. Furthermore, our cryptosystem contains an error-correcting mechanism that is implicitly embedded, while RSA needs an additional error-correcting algorithm to compensate for transmission errors.

One of the obvious drawbacks is the low information transmission rate; the code rate R is less than 1 as we introduce redundancy, while this rate is 1 for RSA which is based on a one-to-one mapping. Hence, the length of the cipher text is longer than that of the plain text, increasing the transmission time. On the other hand, RSA is sensitive to any alteration of the transmitted message, while our system embeds error-correction due to the presence of redundancy [7]. Another drawback is the length of the public key that is much more longer than in RSA, but this fact is not of great importance as it is sent only once. Finally, the decryption process relies on a statistical decryption mechanism; this raises the question of reliability, especially when the system size is finite and decryption errors may be higher. The estimation of the reliability of the system is of great significance for its practical implementation. This leads us directly to the next chapter in which we study the finite-size effects in our cryptosystem.

Chapter 4

Finite-size effects

Theoretically, the cryptosystem leads to the correct solution in a large range of noise values when the system is infinite. However, in finite systems, the algorithm still converges but not always to the optimal solution, especially close to the critical point, where sub-optimal solutions emerge. The convergence to sub-optimal solutions is due to finite-size effects and may put the reliability of the cryptosystem in question. The main goal of this chapter is to provide a numerical analysis of finite-size effects. We will describe the methods applied and the results obtained.

4.1 Theoretical and ideal behaviour

4.1.1 Critical points

Below a critical corruption level p_s, it is theoretically possible to find the optimal solution which corresponds to the state with lowest the free energy of the system. As described in [17], Gallager-type code is similar to a diluted many-body Mattis model in an external field, known in statistical mechanics. The decryption process is represented by equation (3.3):

$$\begin{aligned} z = B\boldsymbol{J} &= A\boldsymbol{s} + B\boldsymbol{\tau} \\ &= A\boldsymbol{\xi} + B\boldsymbol{\zeta} \end{aligned} \tag{4.1}$$

This equation defines an Ising spin representation from a statistical physics point of view, corresponding to the formula:

$$\prod_{i\in\mathcal{L}(\mu)} s_i \prod_{j\in\mathcal{M}(\mu)} \tau_j = \prod_{i\in\mathcal{L}(\mu)} \xi_i \prod_{j\in\mathcal{M}(\mu)} \zeta_j = J_\mu \tag{4.2}$$

where $\mathcal{L}(\mu)$ stands for a set of indices of non-zero elements in the μ-th row of A and $\mathcal{L}(\mu)$ stands for a set of indices of non-zero elements in the μ-th row of B. In equation (4.2), $\boldsymbol{s}$, $\boldsymbol{\tau}$, $\boldsymbol{\xi}$ and $\boldsymbol{\zeta}$ are Ising variables, i.e. they are vectors of $\{-1,+1\}$ components. The Hamiltonian reflects this constraint as well as the bias due to the corruption with probability p.

$$\begin{aligned}\mathcal{H} \;=\; & \sum_{\langle i_1,\ldots,i_K;j_1,\ldots,j_L\rangle} \mathcal{A}_{\langle i_1,\ldots,i_K;j_1,\ldots,j_L\rangle} \\ & \times\delta[-1;\mathcal{J}_{\langle i_1,\ldots,i_K;j_1,\ldots,j_L\rangle}\cdot s_{i_1}\cdots s_{i_K}\tau_{j_1}\cdots\tau_{j_L}] \\ & -\frac{F_s}{\beta}\sum_{i=1}^{N} s_i - \frac{F_\tau}{\beta}\sum_{j=1}^{M}\tau_j\end{aligned} \tag{4.3}$$

In equation (4.3), the term F_s is the bias of the original message which is zero as it is the product of a vector with a dense matrix ; $F_\tau = \frac{1}{2}(\log(1-p) - \log p)$ comes from the corruption rate p. The tensor $\mathcal{A}_{\langle i_1,\ldots,i_K;j_1,\ldots,j_L\rangle}$ is equivalent to $A\boldsymbol{\xi}+B\boldsymbol{\zeta}$. The δ function identifies the product of sites $s_{i_1}\cdots S_{i_K}\tau_{j_1}\cdots\tau_{j_L}$ that is in disagreement with $\mathcal{J}_{\langle i_1,\ldots,i_K;j_1,\ldots,j_L\rangle}$. As this first term is not frustrated, it will vanish at low temperature ($\beta \to +\infty$). Therefore, Decoding is carried out at Nishimori's temperature, $F_\tau = \frac{1}{2}(\ln(1-p) - \ln p)$ and $F_s = 0$ for unbiased messages which is the same as giving the correct prior within the Bayesian framework. From the Hamiltonian, we get the partition function, $\mathcal{Z} = \mathrm{Tr}_{\{\boldsymbol{s},\boldsymbol{\tau}\}}e^{(-\beta\mathcal{H})}$. Looking for the minima of the free energy $\mathcal{F} = -\langle \ln \mathcal{Z}\rangle_{\boldsymbol{\xi},\boldsymbol{\zeta},\mathcal{D}}$, we first need to consider the configurational average in order to obtain an expression independant of noise and the randomness of the matrices [17], then we carry out the saddle point equations with the replica method. The latter is based on the following identity:

$$\langle \ln \mathcal{Z} \rangle = \lim_{n \to 0} \frac{\langle \mathcal{Z}^n \rangle - 1}{n} \tag{4.4}$$

Thus we can define the thermodynamically dominant state evaluating the free energy of the different solutions.

Figure 4.1 provides a schematic view of the free energy of the system [7, 15]. Increasing the value of p changes the free energy landscape: the perfect retrieval of the original message can theoretically be obtained up to a corruption level $p_3 < p_c$ (p_c is Shannon's limit), below which the perfect solution is dominant over other solutions, but practical decoding can be achieved only below the spinodal point p_s, that corresponds to the noise level below which only the perfect overlap solution exists. Table 4.1 gives a comparison between p_s and the theoretical value of p_c for different code rates [15].

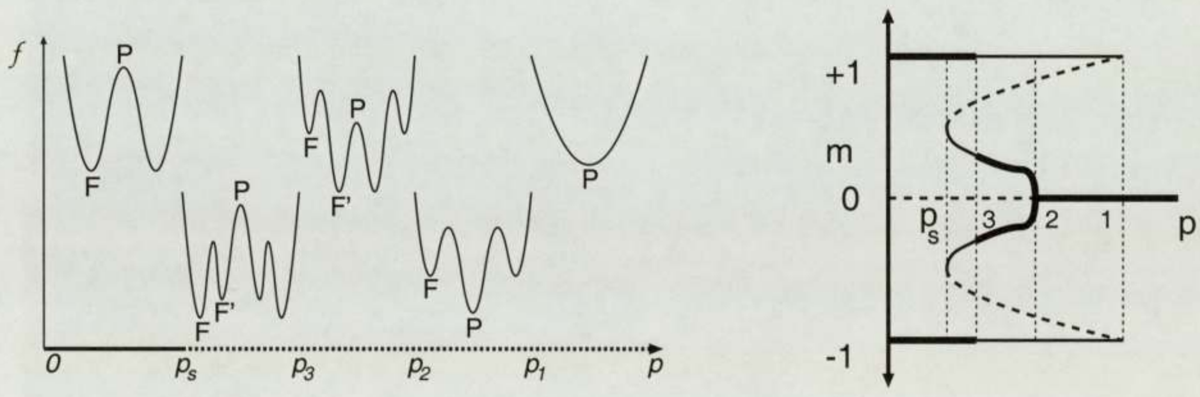

Figure 4.1: On the left : the free energy plotted as a function of the noise level p for a system with $K = 2$ and $L = 2$. The symbol F refers to the perfect overlap state, F' to the sub-optimal decoding solution, and P to the non-informative solution. Below p_s, the spinodal point, the only stable solution is the optimal one (or the mirror solution), so the algorithm is guaranteed to converge to the right solution. Up to p_3, the perfect overlap solution is still dominant but sub-optimal solutions appear. Above p_3, the sub-optimal solutions become dominant even if the right solution is still existing and other solutions emerge later on. On the right: the overlap m between the original message and the optimally decoded message. The thick line denotes the dominant solution.

4.1.2 The system's performance

In the limit of infinite message length, $M \to \infty$, practical decryption should result in an optimal estimate up to p_s, as no other solution exists, and should typically fail

Rate R	2/3	1/2	2/5	1/3
p_s	0.0528	0.0930	0.1206	0.1439
p_c	0.0615	0.1099	0.1461	0.1739

Table 4.1: Noise values at the spinodal point and at Shannon's limit as they vary with the rate of the code. These results are given for the code construction $K = 2$ and $L = 2$.

above this point.

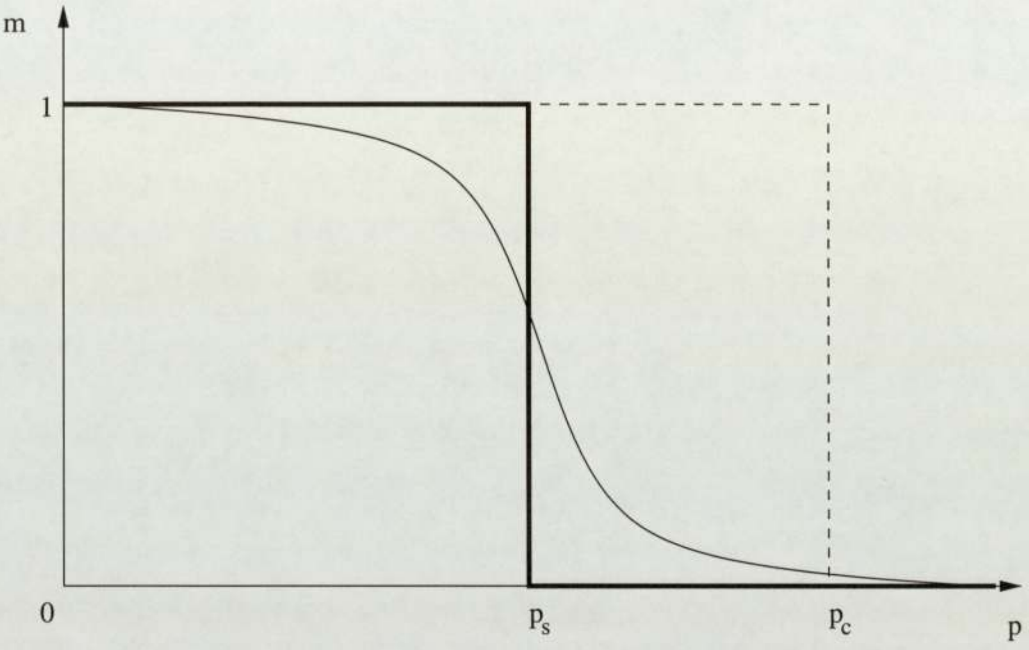

Figure 4.2: The performance of the suggested cryptosystem where m measures the overlap between the estimated and original messages. The dashed line corresponds to an ideal cryptosystem which decrypts the message successfully up to the transition point where the perfect overlap solution ceases to be dominant. The thick line represents the practical limitation for the infinite system size; as the perfect overlap state is the only one existing below the spinodal point, the algorithm will always converge to it. Above this point, convergence to sub-optimal solutions is likely to occur. Finally, the thin line represents the finite-size effects that appear in practical cases for finite M values.

In practice, finite-size effects are quite important since sub-optimal solution are obtained even below the spinodal point. Figure 4.2 demonstrates the expected behaviour of the average overlap between the decoded/decrypted message and the original message as a function of the noise level p, both practically and theoretically. Therefore, the dependence of the block error probability P_e (equation (2.4)) with respect to the length of the message M and the corruption level p is of great interest. From theoretical results obtained in the field of information theory, we expect an exponential behaviour for the block error probability:

$$P_e = A(M,p)e^{ME(p)} \tag{4.5}$$

where $E(p)$ is the so-called reliability function and $A(M,p)$ a pre-factor. The pre-factor is expected to be almost constant and it will be considered to be so for all the exponential regressions we will apply on the data.

This theoretical result can be obtained by considering upper and lower bounds for the block error probability using Tchebychev and Chernov inequalities (see [5] for details). On the one hand, the *random coding upper bound*, which corresponds to a random mapping of messages to codewords, provides:

$$E_r(R) = \begin{cases} 1 - R - \log_2(1 + \sqrt{4p(1-p)}) & \text{for } R \in [0, R_c] \\ T_p(D) - H_2(D) & \text{for } R \in [R_c, 1 - H_2(p)] \end{cases} \tag{4.6}$$

where $T_p(D) = -D\log_2(p) - (1-D)\log_2(1-p)$, $R_c = 1 - H_2(p - \sqrt{p(1-p)})$ and D is here defined as $R = 1 - H_2(D)$, i.e. $D = p_c$.

On the other hand, the *sphere packing lower bound* defined as the optimal packing of noise spheres around codeword vectors in the M dimensional space provides:

$$E_{sp}(R) = T_p(D) - H_2(D) \text{ for } R \in [0, R_c] \tag{4.7}$$

Therefore, we have the relation:

$$\frac{p}{(1-p)\sqrt{8(M+1)}}e^{-ME_{sp}(R)} \leqslant P_B(M,R) \leqslant e^{-ME_r(R)} \tag{4.8}$$

With these bounds [14], we can consider the finite-size effects by assuming that they are of an exponential nature, and analyse the practical decryption results. Notice that the upper and lower bounds represent an optimal encoding/decoding scheme and we expect any practical algorithm to give worse results. We will also mainly focus on corruption values below p_s where the sub-optimality of our decoding algorithm is likely to have a smaller effect on the decay pattern.

4.2 Experiments and results

4.2.1 Modus operandi

We have decided to apply the belief propagation algorithm as it is easier to use, has better convergence properties, and because the bit error probability gives additional information on the distance between the retrieved and the original message (Cf. page 18). We focus on random matrices with $K = L = 2$ and a code rate of $1/2$.

The halting criterion for the algorithm is the convergence of both message and corruption variables, i.e. both vectors remain stable, for 15 steps. When convergence is reached, the retrieved message is compared with the original one to determine the overlap between them. If the overlap m is different from 1, then sub-optimal convergence is recorded as well as an error. If no convergence is reached after 500 iterations, the algorithm is stopped and the case is consider as an error.

We consider messages of length M from 50 to 5000, and a flip bit probability p ranging from 0.05 to 0.12 (the spinodal point is around $p_s = 0.09$, and Shannon's limit is for $p_c = 0.11$). For each couple (M, p), we monitor $10,000$ runs of the algorithm where the matrices A and B, the message and the corruption vector were generated randomly, for each run. All messages are taken to be unbiased.

For all the numerical tables, we have applied an exponential regression with the assumption that the block error probability follows the law $A(p)\exp(-ME(p))$.

4.2.2 Error probability measures

Two measures for the block error probability are introduced. The first one is referred to P_{ec} and considers only those runs where the iterative decoding process has converged, the second one is referred to P_e and also includes runs where the decoding process has not converged (which are counted as errors). If we note N_{sub} the number of runs which converged to a sub-optimal solution, N_{total} the total number of converging runs and N_{runs} the total number of runs of our experiment, we get:

$$\begin{aligned} P_{ec} &= \frac{N_{sub}}{N_{total}} \\ P_e &= 1 - \frac{N_{total} - N_{sub}}{N_{runs}} \end{aligned}$$

The block error probability denotes any single bit error. In runs where the decoding process has not converged within the time limits, we cannot determine whether an error occurred. Therefore, we measure the probability P_{ec} omitting ambiguous cases. On the other hand, considering that the decryption/decoding is generally fast, we also decided to count the non-converging cases as an error as they can therefore be considered as case where the algorithm enters a cycle. This second approach leads to the P_e probability. The two error measures are quite similar, except for codes where a large fraction of runs do not achieve convergence in the time limits.

Finally, one should notice that as M increases, the total number of converging cases increases, approaching N_{runs}, leading to the equivalence of the two measures when $M \to \infty$. Moreover, we have the relation:

$$\frac{1 - P_e}{1 - P_{ec}} = \frac{N_{total}}{N_{runs}}$$

4.2.3 Results

From the results, we notice that the block error probability was following the expected behaviour and is approaching a step function with increasing values of M. This behaviour is shown in on figure 4.3, where we measure the equivalence between vectors with respect to both signal and noise as described further on.

It is desirable to compare the results with the theoretical bounds. In the specific range of code rates studied, both lower and upper bounds are identical, so we expect the results to be close to the theoretical limits. One should point out that we actually expect the performance to be worse that these bounds which correspond to optimal decoding. One notices from figure 4.4 that numerical results have the same shape as the theoretical limit but do not lie on top of each other. As the theoretical limits

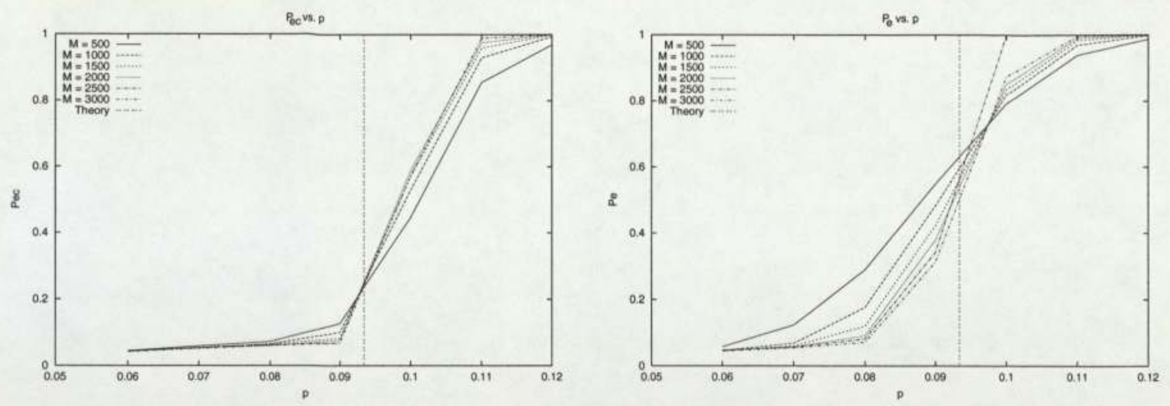

Figure 4.3: Practical behaviour around the spinodal point. P_{ec} and P_e represent the block error probability. As expected, with increasing values of the message length M, the block error probability looks more and more like a step function. The left graph refers to P_{ec}. These graphs were obtained recovering both signal and noise.

are defined for ideal encoding/decoding mechanisms, the corresponding block error probability is lower than the practical values obtained. Hence, the numerical results obtained in our experiments are slightly different than the theoretical limit due to the sub-optimal encoding/decoding techniques.

Signal and noise retrieval

From a theoretical point of view, we need to consider the retrieval of the noise as well as the signal. In the following experiments, we use a corruption level from 0.06 to 0.12 to concentrate on the behaviour around the spinodal point p_s and the message lengths spread from 500 to 3000 in steps of 500. The results are summarised in figure 4.5.

We first notice an almost flat behaviour of $\log(P_e)$ and $\log(P_{ec})$ in the logarithmic scale with respect to M. This indicates a vanishingly small reliability function. As a consequence, to reach a given threshold of confidence, we will need a much longer message length, revealing that the finite-size effects are more important than previously expected. This behaviour could come from the algorithm itself, either due to the inherent approximation in the decoding process, or from the fact that the practical code we use, is not optimal as expected to achieve the theoretical limit.

Another important characteristic of these graphs is the appearance of an error floor

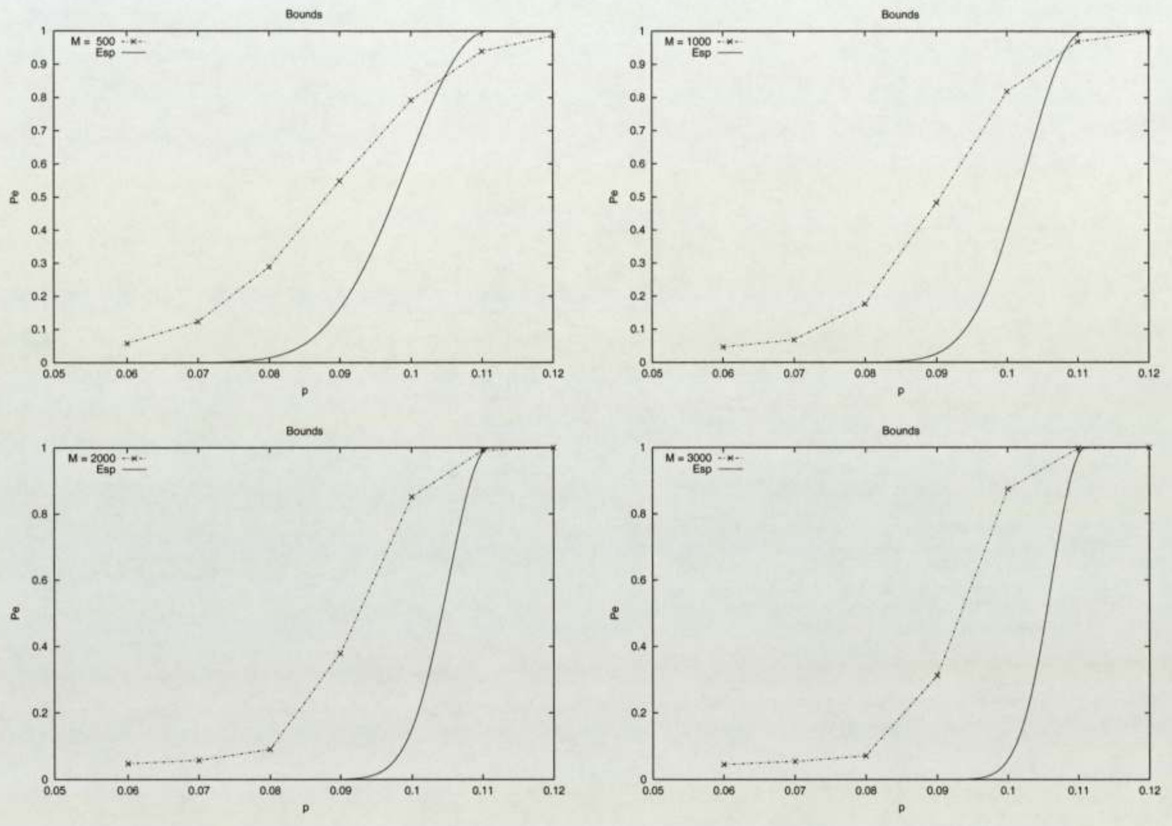

Figure 4.4: Block error probability and bounds. The slope of the graphs, both theoretical and practical, is steeper close to the transition point with increasing M values. The practical results follow the behaviour of the theoretical limits with some difference. This comes from the fact that the theoretical limits are defined for ideal cases; it is also natural that for practical codes of finite size the block error probability may be above the limit. For instance, for $M = 500$, some cases of perfect retrieval exist even beyond Shannon's limit due to the finite-size effects. This phenomenon disappears as M increases.

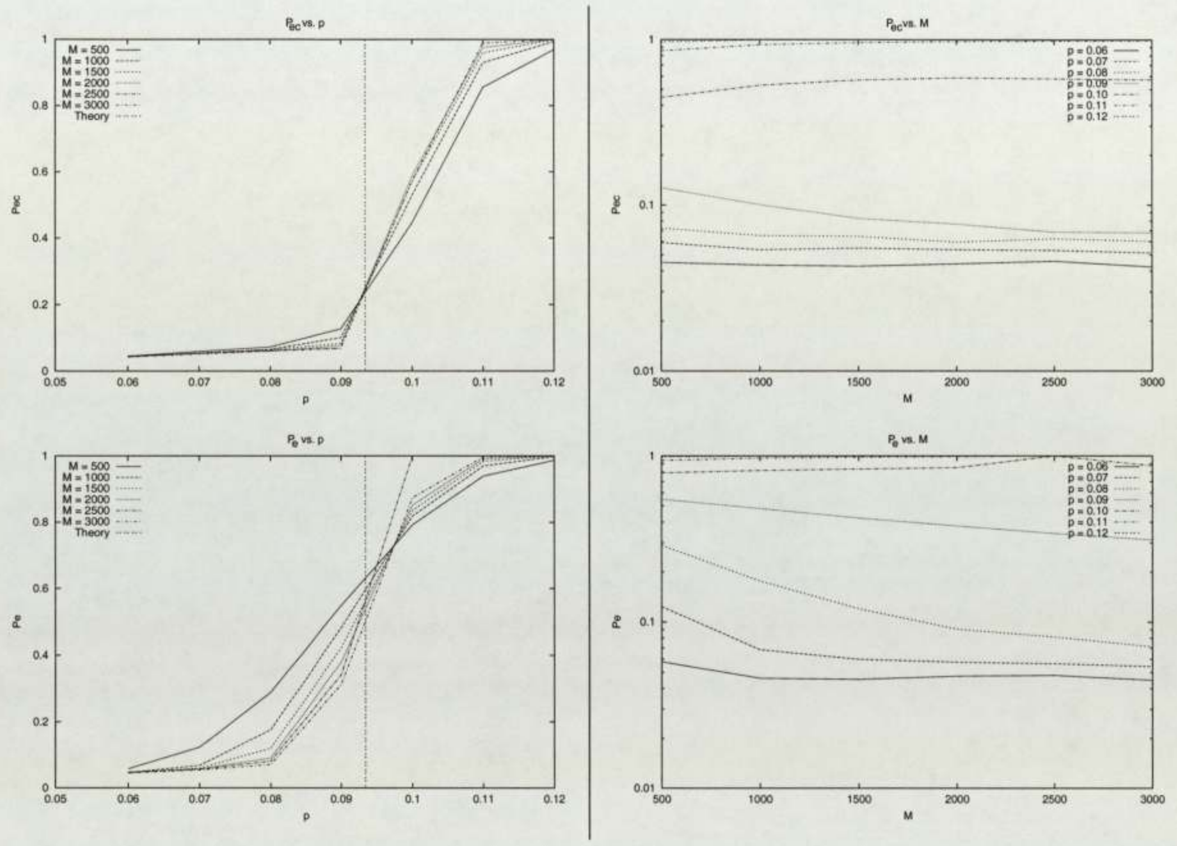

Figure 4.5: Left column: the error probability increases with the corruption level. Right column: In a logarithmic scale on the Y-axis, the block error probability decreases with the length M of the cipher-text below the spinodal point; for corruption levels above the spinodal point, the behaviour is inverted. Below p_s, the behaviour of $\log(P_e)$ and $\log(P_{ec})$ is almost flat.

at low corruption levels. This seems to be related to the existence of small loops in the matrices. These loops force the algorithm to oscillate asymptotically mainly in the noise part of the estimate, due to the high bias, increasing the block error probability value.

p	intercept	slope ($\times 10^{-5}$)	E_{sp}
0.06	0.044795	1.8116	0.026971
0.07	0.058708	4.6399	0.016003
0.08	0.071936	6.9803	0.008485
0.09	0.13057	24.121	0.003640

Table 4.2: Intercepts and slopes of the probabilities in a logarithmic scale to point out the linear behaviour, for different corruption levels. The slope is almost zero in comparison to the theoretical bound, leading to small value of the reliability function. this is obtained for a code rate $R = 1/2$, with $K = L = 2$.

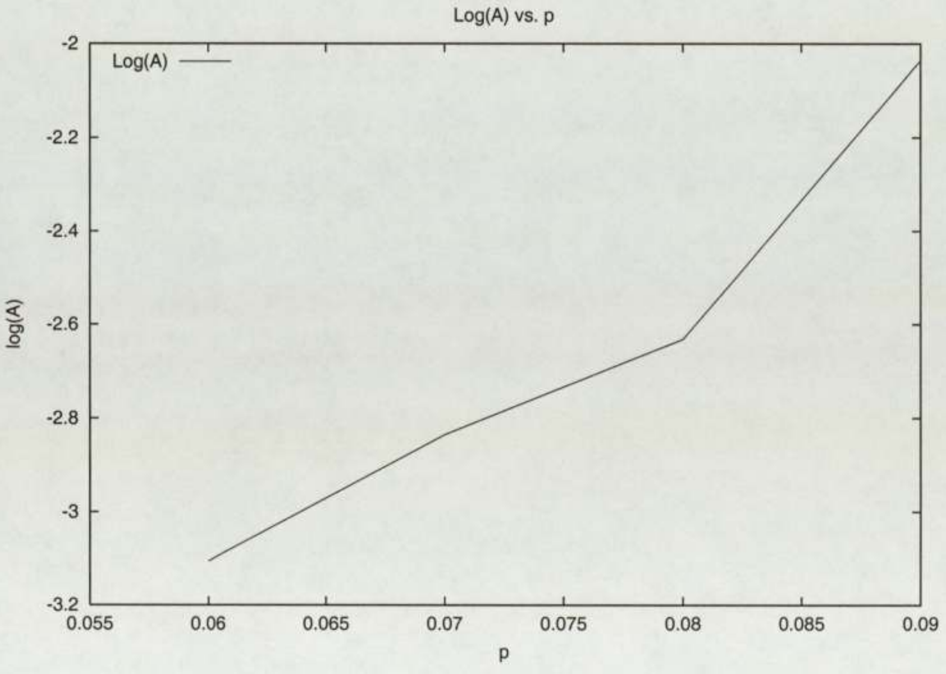

Figure 4.6: Behaviour of the pre-factor $\log(A)$ with the corruption level p. This behaviour appears to be quite linear.

If we consider the pre-factor A, we can see that it decreases exponentially with p, as shown on figure 4.6, as the logarithm seems to decrease linearly with p. The significant difference between the theoretical limits and the numerical results raises doubts about the numerical results we obtained. We currently try to verify the accuracy of the results. However, if the results are accurate, they reveal strong finite-size effects, and an important pre-factor A. The results of table 4.3 cannot be fully trusted as the flat behaviour of $E(p)$ is very sensitive to fluctuations, a minor alteration of $E(p)$ would give rise to a drastic modification in the numerical results. Finally, as this flat behaviour is not expected, it may result from computational errors, or even from the fact that the code used here is not an optimal encoding/decoding process.

p	$P_e = 10^{-4}$	$P_e = 10^{-5}$	$P_e = 10^{-6}$
0.06	100,330	137,080	173,830
0.07	25,880	34,420	42,960
0.08	14,715	18,915	23,115
0.09	39,105	49,445	59,790

Table 4.3: Extrapolation of message length M to achieve a level of reliability P_e for a given corruption level p. On the assumption of accurate encoding/decoding, these results indicate strong finite-size effects for a code rate $R = 1/2$.

Some experiments have been carried out for different code rates to check the dependency of the results with the code rate. The behaviour is generally similar, as one can see in figure 4.7.

Signal retrieval

In this part, we focus on the signal retrieval alone, as it is the aim of the cryptosystem. We therefore, measure the decryption success with respect to the signal alone. The results are summarised in figure 4.8. These graphs point to the exponential behaviour of P_e and P_{ec} for large M values. Some irregularities appear, that may be due to the low M values or to an error in the decoding process. Note that these results are relevant only in a practical sense as the theoretical bound considers retrieval of both the noise and the signal.

In these experiments, we focus on the range for p below the spinodal point to stay in the satisfiable range of the cryptosystem. The corruption level runs from 0.05 to 0.09 and the message lengths M are { 50, 100, 200, 500, 1000, 2000, 5000 }.

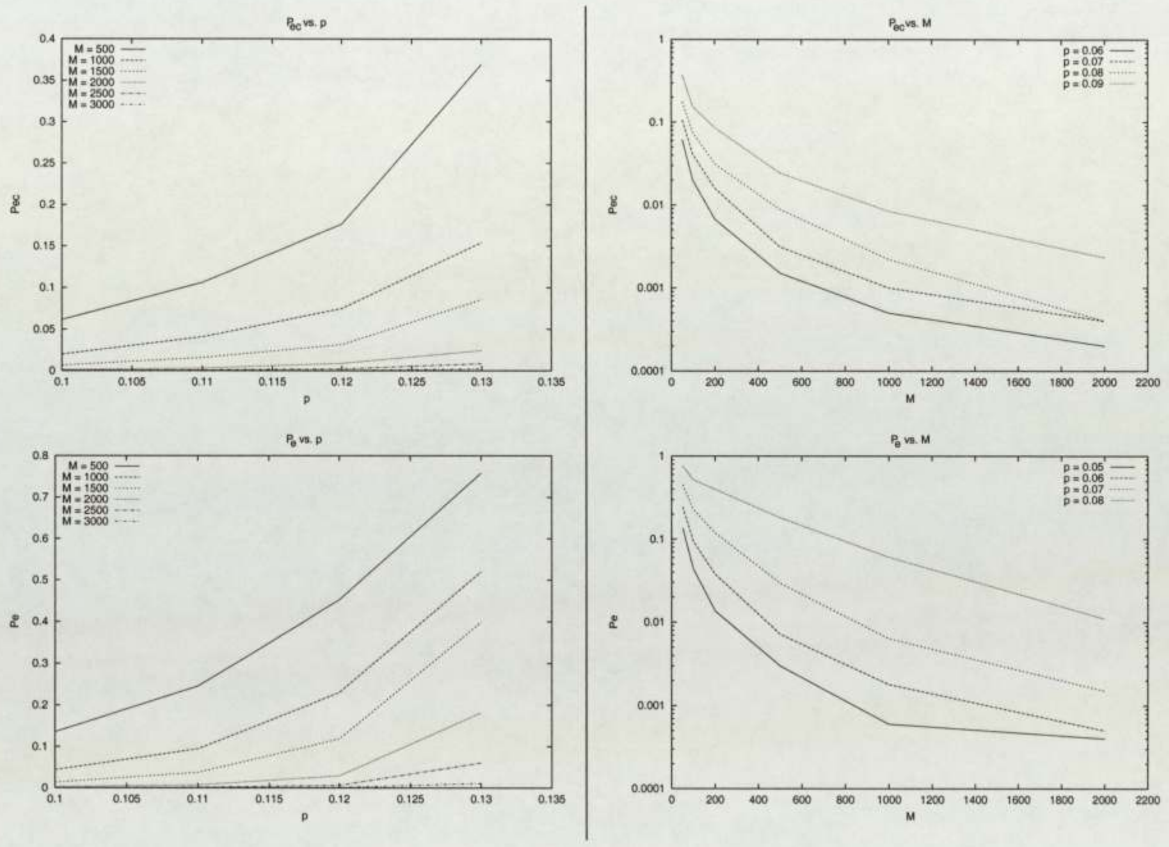

Figure 4.7: Behaviour of P_e and P_{ec} for a rate of R = 1/3. The behaviour of the block error probability is similar to the one for a rate of 1/2.

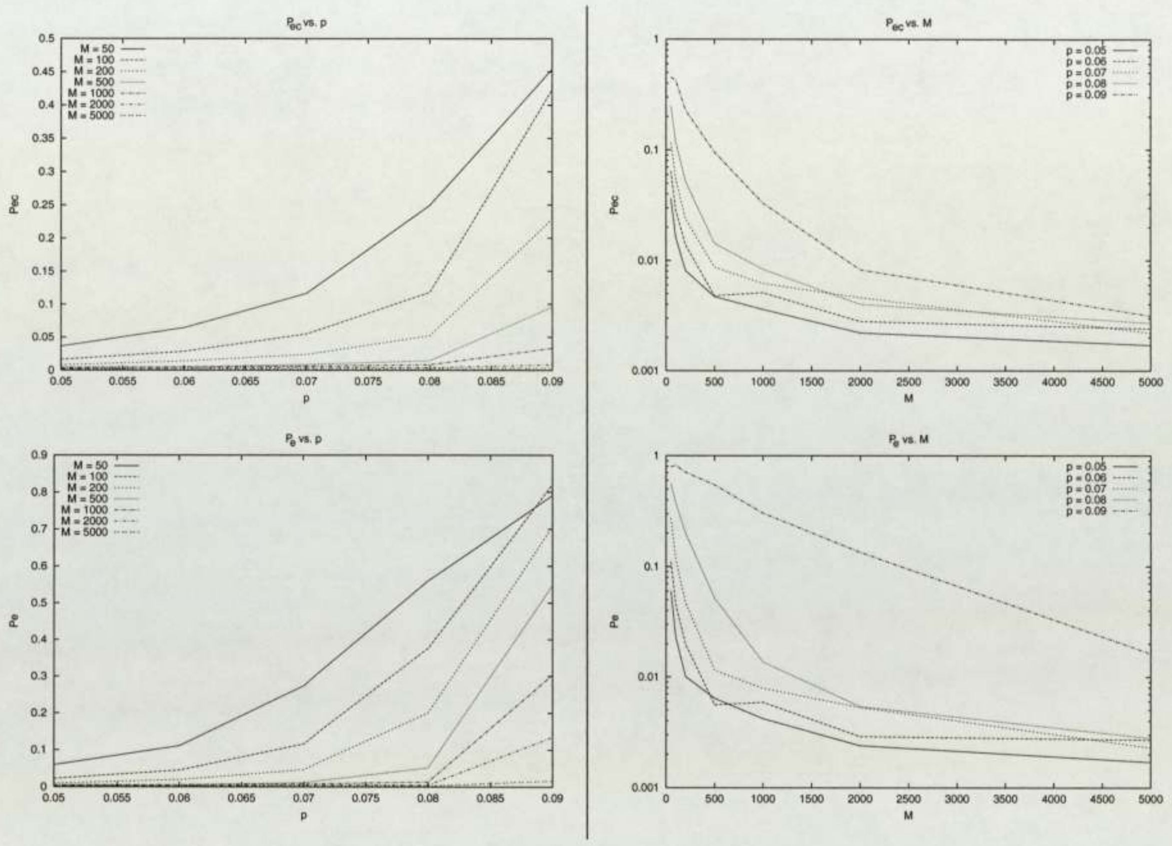

Figure 4.8: Signal retrieval. Left column: the error probability increases with the corruption level. Right column: In a logarithmic scale on the Y-axis, the block error probability decreases with the length M of the cipher-text. For sufficiently large values of M the behaviour becomes linear.

Correction about the results

As shown previously, it appears that the obtained results follow the expected behaviour but with some bias to the theoretical solution. This is a result of a non-optimal algorithm, and the existence of small loops in the sparse ramdom matrices.

These small loops can be interpreted as cycles the algorithm cannot deal with, the decoding process is non-optimal and may lead to sub-optimal solutions. The clear appearence of these loops can be seen on the threshold we reach when $p \leftarrow 0$. Theoretically, when p is approaching 0, the block error probability should be 0. In figure 4.3, we can clearly see this unexpected threshold.

To get more consistent results, the small loops that appear in the sparse matrices generation must be avoided. Some works should be done in that way.

Chapter 5

Digital signatures

In this chapter, the role of the cryptosystem is reversed; it is used to define a signature using the private key in conjunction with the message, to generate a signature. The public key, combined with a comparison function, will serve as authentication. The main attack consists in the falsification of the signature with the full knowledge of the public key. Using the public knowledge and the message, it seems that many schemes remain insecure. Even though we made many attempts to devise a reliable electronic signature, these schemes remain insecure. We now describe the general scheme before developing the different solutions studied.

5.1 Digital signature scheme

In this chapter, the cryptosystem is reversed to define a signature using the private key in conjunction with the message to generate a signature. The public key combined with a comparison function will serve as authentication. In order to test the reliability of such an electronic signature, we define some attacks that can corrupt the authentication data. The main attack consists in the falsification of the signature with the knowledge of the public key. Here is the description of the main ideas of digital signature before turning to the different specific solutions studied.

5.1.1 Generating the signature

For this approach, the notations need to be redefined. The message is not encrypted and refers to $\boldsymbol{J}$. It is important to underline that the message remains open all the time and one concentrates on the authentication of the sender. From the message and the private key, the signature is generated using the decryption process. It can be interpreted as the most likely sequence of bits that could have produced the vector $\boldsymbol{J}$. Clearly, this signature depends both on the text and on the personal key. Therefore, the signature is added to the original message and they are sent as one unique message. As the signature depends on the private key, it may be necessary to make some changes to the keys or to the belief propagation algorithm, for security problems, even if the main structure of the procedure remains similar.

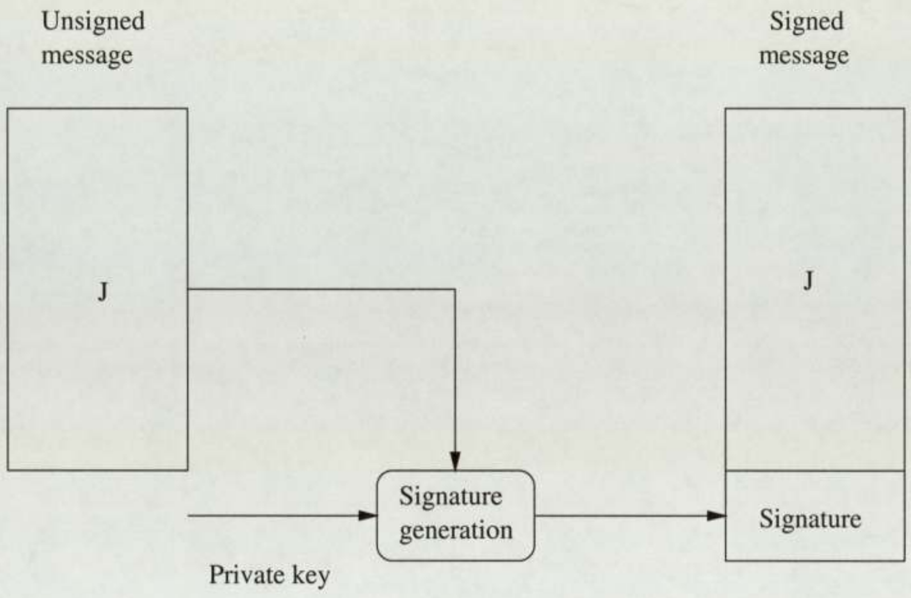

Figure 5.1: From the initial message and the private key, the signature is created using the belief propagation algorithm. Therefore, the recipient should be able to identify the sender with this extra information. As the private key allows the construction of the public key, the signature contains information that should be regenerated from the message by the public key.

5.1.2 Authentication

In order to identify the sender, the recipient has to compare the message with the signature with a constraint depending on the public key. In this scheme, the receiver will generate a cipher text from the signature and verify its dependence on the message itself corresponding to the corruption level defined in the public key.

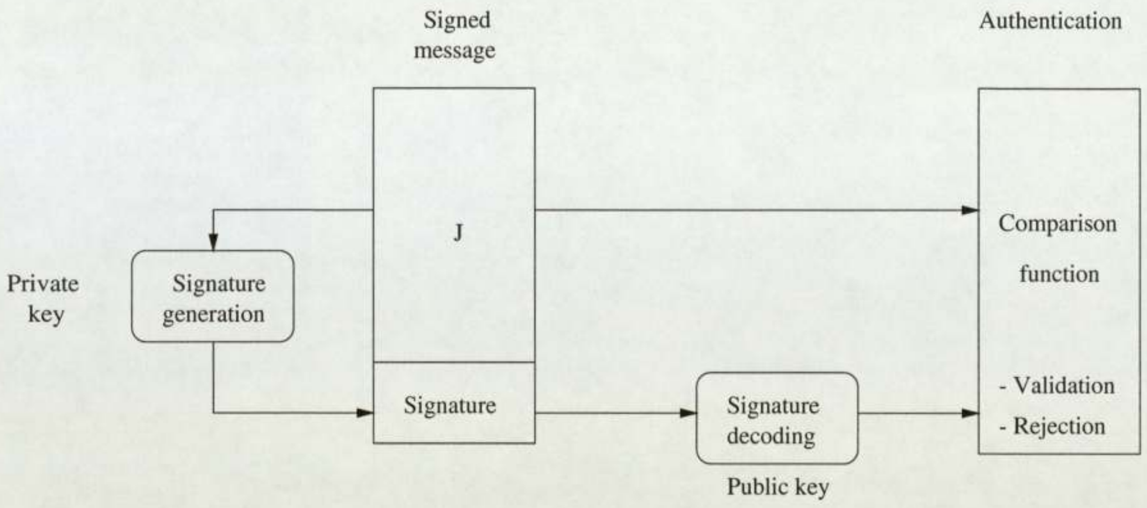

Figure 5.2: Identification process. The receiver should validate the signature by some predefined operations using the public key (e.g., matrix multiplication) and the message. If the predefined relation is obeyed, the authentication is defined successful and fails otherwise.

5.2 Attempts to generate a secure scheme

5.2.1 Main scheme

Starting from a message $\boldsymbol{J}$ and our private key (A, B, D, p), we try to generate a secure digital signature. Naturally, we try to reverse the process of the cryptosystem, applying the decrypting procedure as described in equation (3.3):

$$\boldsymbol{z} = B\boldsymbol{J} = A\boldsymbol{s} + B\boldsymbol{\tau}$$

This leads to the most probable vector $\boldsymbol{s}$ that could have generated $\boldsymbol{J}$ given the element of the private key. Thus $\boldsymbol{s}$ depends on both the message $\boldsymbol{J}$ and the private key. The authentication process will consist in generating the vector $\boldsymbol{J}^0 = G\boldsymbol{s}$ using information of public key and comparing it to the original message $\boldsymbol{J}$. The identification is then certified if $\boldsymbol{J}^0 - \boldsymbol{J}$ is a (noise) vector with a flip probability p.

The previous scheme is the theoretical basis for a secure digital signature process but is not yet unbreakable. The actual weakness of this scheme is the identity substitution in which we manage to substitute the original signature by a faked one. Even some more evoluted version of this scheme seem to fail.

Initial scheme

The most obvious attack appears when one considers the way we check the authentication. As one is able to generate ones own corruption vector $\boldsymbol{\zeta}'$ with probability p, one can generate ones own signature $\boldsymbol{s}'$ solving $\boldsymbol{J} - \boldsymbol{\zeta}' = G\boldsymbol{s}'$. The attacker has only to substitute the real signature with his own to fake the identity of the sender (note that it is not yet clear from this that the above equation can be solved, as we will see below).

Signal and noise combined scheme

We considered sending both the most probable signal and noise, $\boldsymbol{s}$ and $\boldsymbol{\tau}$ as the signature, but the previous attack remains, due to the ability of an attacker to generate his own corruption vector, independently of the generation of the signature.

Signal and noise dependence scheme

An attempt to link the corruption vector and the signal vector during the generation of the signature was then considered. Any scheme $\boldsymbol{s} = H\boldsymbol{\tau}$ is subject to the obvious attack, consisting of solving a linear system with a constraint that remains solvable. In this scheme, H do not have any property, it can be any binary matrix that introduces some dependences between $\boldsymbol{s}$ and $\boldsymbol{\tau}$, as the weakness is based on linear system solving and that we are still able to generate our random noise $\boldsymbol{\tau}$.

We considered a dependence between the corruption vector and the signal vector following the scheme $\boldsymbol{\tau} = H\boldsymbol{s}$. In such a model, the previous attack is not that obvious to apply. Generating the corruption vector does not implicate anymore that the signature is reachable through the resolution of a linear system, as we need to satisfy a much more specific constraint. The main problem is the ability to generate these vectors for the sender. The complexity to generate the signature is to be considered for specific choices of the dependence matrix H.

Although, if the signature generation is much more difficult, it may be necessary to change the algorithm that generates it and/or the key data. Note that the difficulty to generate the signature for the sender and the attacker differs only in the knowledge and the structure of the matrix H.

5.2.2 Space covering

It seems that the main problem in this digital signature scheme is the space covering problem. If we consider a signal vector $\boldsymbol{s}$ of length N, we have 2^N possible vectors. Thus, we can map 2^N vectors onto the 2^M possible messages using the redundancy generation $\boldsymbol{J}^0 = G\boldsymbol{s}, M > N$. Introducing the corruption vector $\boldsymbol{\tau}$, we define a sphere in which the algorithm tends to converge to $\boldsymbol{J}^0$. This is represented in figure 5.3.

This problem of mapping relies on the corruption level used. Below Shannon's limit, the space is not fully covered and messages can be out of all spheres; i.e., there is no guarantee that a solution (signature) exists for a given message. The situation is even worse, such a solution exists only for an exponentially small fraction of all possible messages.

To counter this point, two attempts were made. The first was to generate the signature *above* Shannon's limit. The second one was to consider any vector of the message space as the combination of two most probable signal messages, $\boldsymbol{J} = \boldsymbol{J}^0 + \boldsymbol{J}^1$ where $\boldsymbol{J}^0$ and $\boldsymbol{J}^1$ correspond to two different mapped vectors $\boldsymbol{s}^0$ and $\boldsymbol{s}^1$. Both of these attempts failed due to the breakdown of the belief propagation algorithm. We therefore, did not manage to devise a secure electronic signature, leaving this field open for further research.

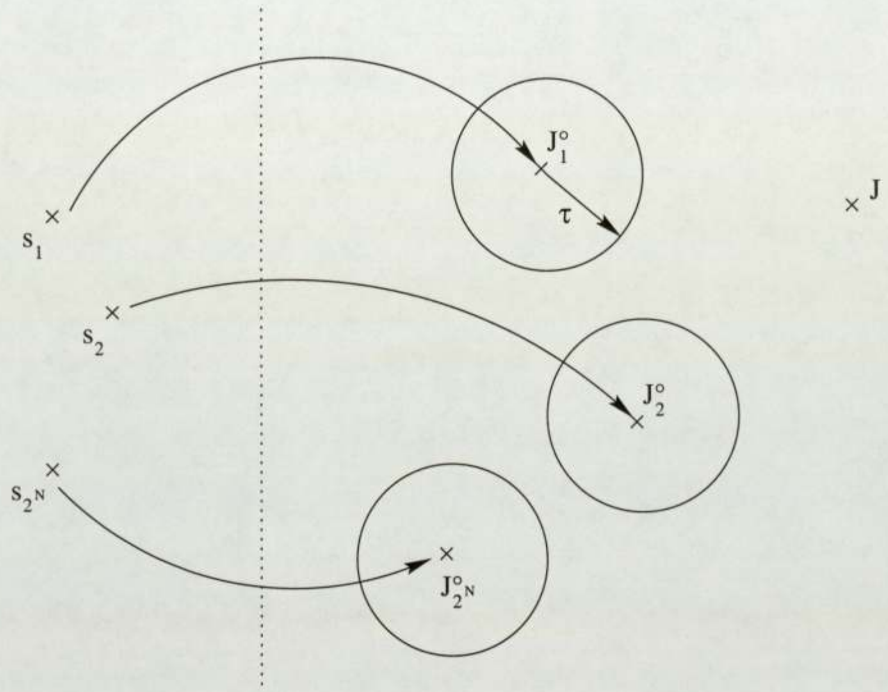

Figure 5.3: Mapping of 2^N vectors into a 2^M space, $N < M$. The vectors from the 2^M space that are in a sphere of radius $\boldsymbol{\tau}$ will converge to mapped element using the decryption process. On the other hand, for a corruption below Shannon's limit, some vectors $\boldsymbol{J}$ do not lead to a mapped element. For our signature process that means some messages among all the possible could not give rise to a valid signature, given a corruption probability.

5.3 Conclusion

Using the public knowledge and the message, it seems that many schemes remain insecure. Linear dependencies and space covering that are the main approaches seem to be unefficient. Although, some attempts have been made [9], none leads to a completely secure scheme. Works is still in progress in order to achieve a fully secure process.

Chapter 6

Conclusion

In this thesis, we consider low-density parity-check codes, derived from error-correcting codes, as a secure transmission solution, using belief propagation as a statistical decoder. After describing the specificities of the algorithm and the main scheme of the encryption/decryption process, we have studied finite-size effects.

It was shown that, although the practical decoding roughly follows the theoretical behaviour, the finite-size effects appear to be stronger than expected, assuming that there is no error in our scheme. The main weakness of the analysis seems to be in the presence of small loops in the sparse matrices generation, creating cycles in the decoding/decrypting algorithm. As a consequence, the reliability function is found to be much lower in practice than its theoretical value. As the reliability function flattens, it appears that the pre-factor is more important than expected. The validation of these results are still in process during the conclusion of this thesis. There is still some work to be done to remove small loops to verify that their occurence is not negligible and to put the results much closer to theoretical ones.

Finally, we have described a scheme to implement digital signature. Some security holes are pointed out, and different attempts to encounter them are specified. Some ideas are still to be tested, but for the moment none of them achieves a secure electronic signature scheme.

As future research directions, the big gap between practical results and theoretical values for the cryptosystem needs to be investigated, to determine whether this difference is inherent to the algorithm or due to some other factors. The problem of the digital signature is still an open one, although the main idea is to devise the signature as a link between both signal and corruption vectors after solving the problem of space covering.

Bibliography

[1] M. C. Davey. *Error-correction using low-density parity-check codes.* PhD thesis, University of Cambridge, Cambridge, UK, Dec. 1999.

[2] C. Domb and M. S. Green. *Phase transitions and critical phenomena*, volume 2. Academics Press (London), 1972.

[3] C. E. Froberg. *Introduction to numerical analysis.* Addison-Wesley, second edition, 1970.

[4] R. G. Gallager. Low density parity check codes. *IRE Trans. Info. Theory*, IT-8:21–28, Jan. 1962.

[5] R. G. Gallager. *Information Theory and Reliable Communication*, chapter 5. John Wiley & Sons, (New York), 1968.

[6] Y. Iba. The Nishimori line and Bayesian statistics. *J. Phys. A: Math. Gen.*, 32:3875–3888, 1999.

[7] Y. Kabashima, T. Murayama, and D. Saad. Cryptography properties of Ising spin systems. *Phys. Rev. Lett.*, 84(9):2030–2033, Feb. 2000.

[8] Y. Kabashima, T. Murayama, and D. Saad. Typical performance of gallager-type error-correcting codes. *Phys. Rev. Lett.*, 84(6):1355–1358, Feb. 2000.

[9] I. Kanter, E. Kanter, and L. Ein-Dor. Secure linear cryptosystem using error-correcting codes. *Europhy. Lett.*, 51(2):244, Jul. 2000.

[10] I. Kanter and D. Saad. Error-correcting codes that nearly saturate Shannon's bound. *Phys. Rev. Lett.*, 83(13):2660–2663, Sept. 1999.

[11] I. Kanter and D. Saad. Finite-size effects and error-free communication in Gaussian channels. *J. Phys. A: Math. Gen.*, 33:1675–1681, 2000.

[12] D. J. C. MacKay. Information theory, inference and learning algorithms. Cavendish Lab. Cambridge, UK, 1995-1998. Draft 1.4.8.

[13] D. J. C. MacKay. Good error-correcting codes based on very sparse matrices. *IEEE Trans. on Info. Theory*, 45:399–431, 1999.

[14] R. J. M. McEliece and J. Omura. An improved upper bound on the block coding error exponent for binary-input discrete memoryless channels. *IEEE Trans. on Info. Theory*, 23(5):611–613, Sept. 1977.

[15] T. Murayama, Y. Kabashima, D. Saad, and R. Vicente. Statistical physics of regular low-density parity-check error-correcting codes. *Phys. Rev. E.*, 62(2):1577–1591, Aug. 2000.

[16] M. Mézard, G. Parisi, and M. Virasoro. *Spin glass theory and beyond.* World Scientific, (Singapore), 1st edition, 1987. Lecture notes in Physics, Vol. 9.

[17] H. Nishimori. *Theory of Spin glasses and statistical mechanics of information.* Evaluation draft edition, 2000. available at http://www.stat.phys.titech.ac.jp/ nishi/index-e.html.

[18] D. R. Stinson. *Cryptography theory and practice.* CRC Press, (Florida), 1995.

[19] R. Vicente. *Statistical physics of error-correcting codes.* PhD thesis, Aston University, Oct. 2001.

[20] R. Vicente, D. Saad, and Y. Kabashima. Error-correcting code on a cactus: A solvable model. *Europhy. Lett.*, 51(6):698–704, Sept. 2000.

[21] R. Vicente, D. Saad, and Y. Kabashima. Statistical physics of irregular low-density parity-check codes. *J. Phys. A: Math. Gen.*, 33:6527–6542, 2000.

[22] Y. Weiss. Belief propagation and revision in networks with loops. Technical Report A.I. Memo 1616, M.I.T., July 1998.