INFORMATION TO USERS

This manuscript has been reproduced from the microfilm master. UMI films the text directly from the original or copy submitted. Thus, some thesis and dissertation copies are in typewriter face, while others may be from any type of computer printer.

The quality of this reproduction is dependent upon the quality of the copy submitted. Broken or indistinct print, colored or poor quality illustrations and photographs, print bleedthrough, substandard margins, and improper alignment can adversely affect reproduction.

In the unlikely event that the author did not send UMI a complete manuscript and there are missing pages, these will be noted. Also, if unauthorized copyright material had to be removed, a note will indicate the deletion.

Oversize materials (e.g., maps, drawings, charts) are reproduced by sectioning the original, beginning at the upper left-hand corner and continuing from left to right in equal sections with small overlaps. Each original is also photographed in one exposure and is included in reduced form at the back of the book.

Photographs included in the original manuscript have been reproduced xerographically in this copy. Higher quality 6" x 9" black and white photographic prints are available for any photographs or illustrations appearing in this copy for an additional charge. Contact UMI directly to order.
Decoding of linear block codes based on ordered statistics

Fossorier, Marc Pierre C., Ph.D.

University of Hawaii, 1994
DECODING OF LINEAR BLOCK CODES
BASED ON ORDERED STATISTICS

A DISSERTATION SUBMITTED TO THE GRADUATE DIVISION OF THE
UNIVERSITY OF HAWAII IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE OF

DOCTOR OF PHILOSOPHY

IN

ELECTRICAL ENGINEERING

DECEMBER 1994

By

Marc P.C. Fossorier

Dissertation Committee:
Shu Lin, Chairperson
Mike Hilden
N. Thomas Gaarder
W. Wesley Peterson
Galen Sasaki
Acknowledgements

I would like to express sincere gratitude to Professor Shu Lin for his constant support, enthusiasm, and keen interest in this work. His deep knowledge of coding theory has always allowed the pursuit of different ideas to some conclusive state.

I would like to thank Professor N.T. Gaarder, not only for his comments and careful attention concerning this dissertation, but also for the numerous and fruitful discussions which opened other horizons of research. In addition, his enjoyable and excellent lectures have brought me a strong background in the field of communications and information theory, as well as a rigorous approach for research problems.

I would like to thank my other orals committee and readers: Professor M. Hilden, Professor W.W. Peterson and Professor G. Sasaki for their interest and assistance in this work.

All the faculty members of the University of Hawaii helped to create an enjoyable and challenging environment. I also wish to thank all the students I enjoyed working with: a special “mahalo” to Anurag for his constant encouragements during these five years together which brought me unforgettable memories; thanks to Amir, Brian, D.P., Harit, Manish, Ravi, Sandeep, Vikram, Zhong for sharing some enjoyable moments. I also thank all my teammates of the “M’zamba” soccer team, and especially Johnny, Jim, Joe, Roy, Steve and Tony for all these moments which make of this “Endless Summer” a very difficult place to leave.

Finally, I am grateful to my parents, my family and Lora that enabled me to remain motivated throughout my years in Hawaii. To them, I dedicate this dissertation.
Abstract

This dissertation presents a novel approach to soft decision decoding for binary linear block codes. The basic idea of this approach is to achieve a desired error performance progressively in a number of stages. For each decoding stage, the error performance is tightly bounded and the decoding is terminated at the stage where either near-optimum error performance or a desired level of error performance is achieved. As a result, more flexibility in the trade-off between performance and decoding complexity is provided. The proposed decoding is based on the reordering of the received symbols according to their reliability measure. In the dissertation, we evaluate the statistics of the noise after ordering. Based on these statistics, we derive two monotonic properties which dictate the reprocessing strategy. Each codeword is decoded in two steps: (1) hard-decision decoding based on reliability information and (2) reprocessing of the hard-decision decoded codeword in successive stages until the desired performance is achieved. The reprocessing is based on the monotonic properties of the ordering and is carried out using a cost function. A new resource test tightly related to the reprocessing strategy is introduced to reduce the number of computations for each reprocessing stage. For short codes of lengths $N \leq 32$ or medium codes with $32 < N \leq 64$ with rate $R \geq 0.6$, near-optimum bit error performance is achieved in two stages of reprocessing with at most a computation complexity of $O(K^2)$ constructed codewords, where $K$ is the dimension of the code. For longer codes, three or more reprocessing stages are required to achieve near-optimum decoding. However, most of the coding gain is obtained within the first two reprocessing stages for error performances of practical interest. The proposed decoding algorithm applies to any binary linear
code, does not require any data storage and is well suitable for parallel processing. Furthermore, the maximum number of computations required at each reprocessing stage is fixed, which prevents buffer overflow at low SNR. Further reductions of the number of computations based on probabilistic or decomposable properties of the code considered are also investigated. The generalization of the algorithm to other channels is discussed. Finally, the discrete time channel associated with the AWGN model after ordering is greatly simplified. This simplification allows an important reduction of the number of computations required to evaluate the theoretical error performance of any algorithm based on a partial or total ordering.
# Table of Contents

Acknowledgements ............................................................... iv  
Abstract ................................................................. v  
List of Tables ................................................................. xii  
List of Figures ................................................................. xiv  

1 Introduction ................................................................. 1  
  1.1 Channel Coding ........................................................... 1  
  1.2 Maximum-Likelihood-Decoding of Linear Binary Block Codes .... 2  
  1.3 Outline of the Thesis ....................................................... 5  

2 Hard Decision Decoding and Reliability Informations ............... 9  
  2.1 Channel Model ............................................................. 9  
  2.2 Hard Decision Decoding Based on Reliability Information .......... 11  
    2.2.1 Decoding process .................................................. 11  
    2.2.2 Computational complexity ....................................... 12  
    2.2.3 Performance ....................................................... 13  

3 Statistics of the Noise after Ordering .................................. 19  
  3.1 Ordered Sequence Statistics .......................................... 19  

vii
<table>
<thead>
<tr>
<th>Section</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.2</td>
<td>Monotonic Properties</td>
<td>23</td>
</tr>
<tr>
<td>3.3</td>
<td>Other Immediate Consequences</td>
<td>26</td>
</tr>
<tr>
<td>3.3.1</td>
<td>Recurrence relations</td>
<td>26</td>
</tr>
<tr>
<td>3.3.2</td>
<td>Properties</td>
<td>27</td>
</tr>
<tr>
<td>4</td>
<td>Maximum Likelihood Decoding Based on Statistics after Reliability</td>
<td>29</td>
</tr>
<tr>
<td>4.1</td>
<td>Reprocessing</td>
<td>29</td>
</tr>
<tr>
<td>4.1.1</td>
<td>Definitions</td>
<td>30</td>
</tr>
<tr>
<td>4.1.2</td>
<td>Reprocessing algorithm</td>
<td>32</td>
</tr>
<tr>
<td>4.1.3</td>
<td>Resource test</td>
<td>34</td>
</tr>
<tr>
<td>4.1.4</td>
<td>Computational analysis</td>
<td>38</td>
</tr>
<tr>
<td>4.2</td>
<td>Performance Analysis</td>
<td>38</td>
</tr>
<tr>
<td>4.2.1</td>
<td>Effects of the dependent positions on the performance</td>
<td>38</td>
</tr>
<tr>
<td>4.2.2</td>
<td>Overall performance based on ordering</td>
<td>41</td>
</tr>
<tr>
<td>4.2.3</td>
<td>Optimum soft decision performance</td>
<td>49</td>
</tr>
<tr>
<td>4.2.4</td>
<td>Algorithm performance</td>
<td>50</td>
</tr>
<tr>
<td>4.2.5</td>
<td>Asymptotic error performance</td>
<td>50</td>
</tr>
<tr>
<td>4.3</td>
<td>Simulations Results</td>
<td>51</td>
</tr>
<tr>
<td>4.3.1</td>
<td>Error performance</td>
<td>51</td>
</tr>
<tr>
<td>4.3.2</td>
<td>Number of computations</td>
<td>64</td>
</tr>
<tr>
<td>4.4</td>
<td>Equivalent Algorithm in the Dual Space of the Code</td>
<td>77</td>
</tr>
<tr>
<td>4.4.1</td>
<td>Definitions</td>
<td>77</td>
</tr>
<tr>
<td>4.4.2</td>
<td>Resource test</td>
<td>78</td>
</tr>
<tr>
<td>5</td>
<td>Further Reduction of the Number of Computations</td>
<td>81</td>
</tr>
<tr>
<td>5.1</td>
<td>Additional Optimum Tests</td>
<td>81</td>
</tr>
</tbody>
</table>
8 Majority-Logic-Decoding of Reed-Muller Codes

8.1 A Review of Majority-Logic Decoding of RM Codes over the Binary Symmetric Channel

8.2 Majority-Logic Decoding Based on the Reliability Information of the Received Symbols

8.2.1 BSC model based on received symbol ordering

8.2.2 Gaussian model

8.2.3 Conventional decoding with UMP test for ties breaking

8.3 Computation Complexity

8.4 Simulation Results

9 Conclusion and Future Research

A Duality Between $H_1$ and $G_1$

B Conditional Densities of the Noise after Ordering

B.1 Marginal Conditional Density of $W_i$

B.2 Joint Conditional Density of $W_i$ and $W_j$

B.3 Joint Conditional Density of $W_i$'s

C Statistics of $Z$ after Ordering with Respect to $|Z|$, $\sum|Z|

C.1 Marginal Density of $Z_i$

C.2 Joint Density of $Z_i$ and $Z_j$

C.3 $Z = -\alpha A + W$

D Algorithm for First Stage of Closest Coset Decoding

E Weight Distribution for Closest Coset Decoding of $|u|u + v|$ Constructed Codes
E.1 Equivalent Weight Distribution for CCD Based on the $|u|u + v|$ Construction .................................................. 166

E.2 Application to RM Codes .............................................. 170

F Reliability Measure for the Raleigh Fading Channel with Coherent Detection .................................................. 172

G Normal Approximation of $\tilde{W}_i|X_i = -1$ ........................................ 174
List of Tables

4.1 Computations required for decoding with order-1 reprocessing .......... 38
4.2 \(N_{\text{ee}}(1)\) vs \((N - K)/2 + 1\) for different linear block codes. .......... 40
4.3 Order-1 simulation results for \((24, 12, 8)\) extended Golay code. .......... 65
4.4 Order-2 simulation results for \((24, 12, 8)\) extended Golay code (*: union
bound). ........................................................................................................ 65
4.5 Order-1 simulation results for \((64, 45, 8)\) extended BCH code. .......... 67
4.6 Order-2 simulation results for \((64, 45, 8)\) extended BCH code. .......... 67
4.7 Order-2 simulation results for \((128, 64, 22)\) extended BCH code (*:
union bound). .................................................................................................. 71
4.8 Order-3 simulation results for \((128, 64, 22)\) extended BCH code (*:
union bound) .................................................................................................... 71
4.9 Order-4 simulation results for \((128, 64, 22)\) extended BCH code .......... 71
4.10 Decoding complexity for order-\(i\) decoding and trellis decoding of some
well known codes. ............................................................................................... 72

5.1 Computation savings with syndrome test for \((24, 12, 8)\) extended Golay
code. ............................................................................................................... 83
5.2 \(\alpha\)'s for phase-2 reprocessing of the \((24, 12, 8)\) Golay code ............ 89
5.3 Independent order-1 simulation results for the 32 cosets of the \((32, 16, 8)
RM code. .......................................................................................................... 102
5.4 Order-1 simulation results for the survivor cosets of the (32, 16, 8) RM code

6.1 Order-2 simulation results for (24, 12, 8) extended Golay code.

6.2 Order-2 simulation results for (128, 64, 22) extended BCH code.

6.3 Order-3 simulation results for (128, 64, 22) extended BCH code.

6.4 Order-4 simulation results for (128, 64, 22) extended BCH code.

8.1 Computations required for the majority logic decoding of RM(r, m).

E.1 \( \hat{c}_1 \) and \( \hat{c}_2 \) contributions to \( w(\hat{c}) \).
List of Figures

2.1 AWGN channel model. ................................................... 10
2.2 Error performances for the (32,16,8) Reed-Muller code with majority-
logic-decoding and hard-decision based on reliability information. .... 15
2.3 Error performances for the (64,22,16) Reed-Muller code with majority-
logic-decoding and hard-decision based on reliability information. .... 16
2.4 Error performances for the (64,42,8) Reed-Muller code with majority-
logic-decoding and hard-decision based on reliability information. .... 17
2.5 Error performances for the (64,45,8) Extended BCH code with Berlekamp-
Massey algebraic decoding and hard-decision based on reliability infor-
mation. ................................................................. 18

3.1 $f_{W|X}(w_i | X_i = -1)$ for $N = 16, N_0 = 0.5, X_i = -1$ and $w_i \in [-2,2]$. 21
3.2 $f_{W|X}(w_i | X_i = -1)$ for $N = 16, N_0 = 0.5, X_i = -1$ and $w_i \in [1,2.5]$. 22

4.1 Distribution of $X_P$ for the (24,12,8) extended Golay code. .......... 42
4.2 Distribution of $X_P$ for the (64,22,16) RM code. ........................ 43
4.3 Distribution of $X_P$ for the (64,42,8) RM code. ........................ 44
4.4 Distribution of $X_P$ for the (64,36,12) extended BCH code. .......... 45
4.5 Distribution of $X_P$ for the (64,45,8) extended BCH code. .......... 46
4.6 Distribution of $X_P$ for the (128,64,22) extended BCH code. ........ 47
4.7 Error performances for the (24,12,8) extended Golay code. .......... 53

xiv
4.8 Error performances for the (32,16,8) RM code. ......................... 54
4.9 Error performances for the (32,26,4) RM code. ......................... 55
4.10 Error performances for the (64,22,16) RM code. ....................... 56
4.11 Error performances for the (64,42,8) RM code. ....................... 57
4.12 Error performances for the (64,36,12) extended BCH code. ............ 58
4.13 Error performances for the (64,45,8) extended BCH code. ............. 59
4.14 Error performances for the (128,64,22) extended BCH code. .......... 60
4.15 Error performances for the (128,99,10) extended BCH code. ........... 61
4.16 Error performances for the (128,120,4) extended BCH code. .......... 62
4.17 Number of computations for order-1 (x) and order-2 (o) reprocessings of the (24,12,8) extended Golay code. ................ 66
4.18 Number of computations for order-1 (x) and order-2 (o) reprocessings of the (64,45,8) extended BCH code. ......................... 68
4.19 Number of computations for order-2 (o), order-3 (x) and order-4 (*) reprocessings of the (128,64,22) extended BCH code. ............... 70
4.20 Error performances for order-i reprocessing and Chase algorithm 2 decoding of the (32,16,8) RM code. ......................... 75
4.21 Error performances for order-i reprocessing and Chase algorithm 2 decoding of the (64,42,8) RM code. ......................... 76

5.1 Outer test modification. ........................................ 82
5.2 Percentage of the computation savings for order 1 and order 2 reprocessings of the (24,12,8) Golay code. .............................. 86
5.3 Percentage of computation savings when processing the 32 least reliable codewords of the (24,12,8) Golay code. .................... 90
5.4 Pr\((|D+(\bar{a})|=i \mid |S_e|=0)\) and \(\Pr\((|D+(\bar{a})|=i \mid |S_e|=1)\) for (128,64,22) extended BCH code with SNR = 3.0dB. ................................. 93
5.5 Simulation and theoretical results for order-3 reprocessing of the (128,64,22) extended BCH code with \(T \in \{15,17,20,22,\infty\}\). ................................. 95
5.6 Percentage of average number of computational savings for order-3 reprocessing of the (128,64,22) extended BCH code with \(T \in \{15,17,20,22\}\) and corresponding performance degradation. ................................. 96
5.7 Coset decoding of the (32,16,8) RM code with order-i reprocessing of the repetition code \([(16,11,4)|(16,11,4)]\). .................................................. 101
5.8 \(f_{Zi}(z_i)\) for \(N/2 = 16, N_0 = 0.5,\) and \(z_i \in [-2.5,1]\). .................................................. 108
5.9 \(f_{Zi}(z_i)\) for \(N/2 = 16, N_0 = 0.5,\) and \(z_i \in [0,1.5]\). .................................................. 109
5.10 Closest Coset Decoding for the (32,16,8) RM code. ................................. 112
5.11 Closest Coset Decoding for the (64,42,8) RM code. ................................. 113
6.1 \(f_Z(z)\) for \(N_0 = 0.5\). .................................................. 117
6.2 \(f_{Zi}(z_i)\) for \(N = 16, N_0 = 0.5,\) and \(z_i \in [-4,2]\). .................................................. 118
6.3 \(f_{Zi}(z_i)\) for \(N = 16, N_0 = 0.5,\) and \(z_i \in [0,2]\). .................................................. 119
6.4 Error performances for the (24,12,8) extended Golay code. ................................. 122
6.5 Error performances for the (128,64,22) extended BCH code. ................................. 123
7.1 Comparison between exact and approximated values of Pe(64;128). ................................. 128
7.2 Comparison between Pe(62,63,64;128) and (128/66) Pe(62;128) \cdot (128/65) Pe(63;128) \cdot Pe(64;128). .................................................. 133
7.3 Different error probabilities associated with the ordering of 2 bits. ................................. 135
7.4 Ordered Binary Symmetric Channel for \(N = 2\). .................................................. 136
7.5 Comparison of \(C_{2,ave} (a)\) and \(\tilde{C}_{2,ave} (b)\). .................................................. 137
7.6 Capacities of the first order approximation of the OBSC for $N = 2^i$, with $i \in \{0,7\}$. ........................................... 139

8.1 Error performances for RM(32, 16, 8) with majority-logic-decoding. .... 147
8.2 Error performances for RM(32, 26, 4) with majority-logic-decoding. .... 148
8.3 Error performances for RM(64, 22, 16) with majority-logic-decoding. ... 149
8.4 Error performances for RM(64, 42, 8) with majority-logic-decoding. .... 150
Chapter 1
Introduction

1.1 Channel Coding

In 1948, Shannon provided a mathematical model to represent any digital communication system [1, 2] and established a theoretical limit to protect a digital stream of information against transmission errors that randomly corrupt a message between the transmitting and the receiving ends. Based on his theory, many researchers rapidly found practical solutions, referred to as error-correcting codes.

Error-correcting codes are divided into two distinct categories: block codes and convolutional codes. Both types of codes rely on the same principle of encoding $K$ information bits into $N$ transmitted bits ($N > K$). However, their processing method is different. A block code encodes each block of $K$ bits independently while for a convolutional code, each encoded block depends not only on the present message block, but also on the previous $M$ blocks.

In practice, the choice of a code is dictated by two major parameters, its performance and its decoding complexity. Therefore, while the two categories achieve similar coding gains for a given rate $K/N$, convolutional codes have been mostly implemented since their decoding is simpler. The same decoding algorithm, known as the Viterbi Algorithm (VA), can be applied for both Hard-Decision (HD) and Soft-Decision (SD) Maximum Likelihood Decoding (MLD) decoding of convolutional
codes. For comparable decoding complexities, efficient HD decoding algorithms of block codes have been devised, based on the algebraic structure of the considered family of codes. However, up to recently, no efficient MLD algorithm for block codes had been found, except for particular codes of small dimension, such as the (24,12,8) Golay code. Practical MLD algorithms for block codes should make the application of these codes more competitive.

1.2 Maximum-Likelihood-Decoding of Linear Binary Block Codes

The brute force approach to MLD of a binary linear \((N,K)\) block code requires the computation of \(2^K\) conditional probabilities (or equivalently Euclidean distances from the received sequence for the Additive White Gaussian Noise (AWGN) channel). This method rapidly becomes too complex to be implemented and more effective methods are therefore needed. If a code possesses a trellis structure, the Viterbi decoding algorithm can be applied to reduce the number of computations. Although all linear block codes have a trellis structure [3], the number of states and the branch complexity become prohibitively large for long codes and the Viterbi decoding becomes impractical. Consequently, other efficient algorithms are needed to achieve optimum or near optimum decoding.

Maximum likelihood decoding of block codes has been investigated by many coding theorists; a detailed bibliography of contributions in this area can be found in [4]. Most of the early works trade off the optimal performance for reducing the decoding complexity. In Generalized Minimum Distance (GMD) decoding [5], Forney uses an algebraic decoder to produce a list of codeword candidates. This list is determined from the reliability measures of the symbols within each received block. For each candidate, a test is then performed, with respect to a sufficient condition for optimality.
The most likely candidate is chosen as decoded codeword. Following the same idea, Chase provided an algorithm where a fixed number of the error patterns corresponding to certain least reliable bits are systematically searched [6]. For this algorithm, the maximum number of codewords considered and the error performance depend on the set of tested positions. Chase's algorithm has then been modified to allow the search only on positions corresponding to reliabilities less than a predetermined threshold [7]. For a given set of positions, the error performance depends also on the choice of the threshold, while the maximum number of computations depends on both the choice of the threshold and the signal-to-noise ratio (SNR). These algorithms suffer a slight degradation in performance when used with codes of small dimension, but the gap in error performance with respect to MLD increases with the dimension of the code.

Recently, an optimum MLD algorithm based on the same idea was proposed [8]. No limitation of the search space is imposed at the beginning of the algorithm, but at each iteration, a new powerful sufficient condition for optimality is tested. After this test, the search space for the optimum codeword is reduced, up to convergence to a unique solution. Due to the effectiveness of its associated stopping criterion, this optimum algorithm improve the computational complexities of [6, 7] for short codes. However, the complexity of this new algorithm still increases exponentially with the dimension of the code.

Another proposed technique is to perform syndrome decoding on the received sequence and then, use the syndrome information to modify and improve the original hard decision decoding. This method was introduced in [9], where one of the devised schemes also orders the information bits according to their reliability measures to guide the search of the most likely codeword. Using the same general algorithm, different search schemes, based on binary trees and graphs were presented in [10]. However, the methods presented in [9, 10] require that $N - K$ be relatively small.
because the search is carried out over most of the column patterns of the parity check matrix of the code. For very high rate codes, an efficient method to reduce the search space of [9] was presented in [11]. For a particular code, a predetermined necessary and sufficient list of error patterns is established, based on both the parity check matrix of the code and only a partial ordering of the reliability measures. However, the technique becomes rapidly impractical whenever $N - K$ exceeds 8. Also, not many general conditions on the survivor error patterns which are valid for any codes can be derived [12].

Other methods take advantage of the decomposable structure of certain codes to reduce the overall complexity of trellis decoding [4, 13, 14, 15, 16, 17]. Nevertheless, the trellis complexity still grows exponentially with the dimension of any sequence of good codes [18]. To maintain the number of computations manageable, many suboptimal multi-stage decodings based on the trellis structure have also been devised [19, 20, 21].

Most recently, an optimum algorithm, generalizing an artificial intelligence technique (Dijkstra's algorithm) has been presented [22]. This algorithm first re-orders the received symbols according to their confidence values and then performs a tree search similar to sequential decoding. The search is guided by a cost function which not only evaluates the present cost of the extended path in the tree, but also estimates its future contributions, allowing a significant reduction of the search space. This algorithm allows optimal decoding of long block codes efficiently for high SNR's. However, for large code lengths, the worst case performance may require both numerous computations and very large memory for low to medium SNR's. Another drawback is the dependency of the cost function on the weight distribution of the code, which may remain unknown. A suboptimum version of this algorithm has also
been devised where the maximum number of codeword candidates is limited by a threshold [23].

1.3 Outline of the Thesis

Even though many of the above methods reorder the received sequence with respect to the symbol reliability measures, the statistics of the noise after ordering have not been computed. In this thesis, we determine these statistics; they allow one to analyze the error performance of any decoding scheme with a total or partial reordering of the received symbols. Based on these statistics, different efficient soft decision decoding algorithms are then devised and analyzed.

In Chapter 2, we introduce the Additive White Gaussian Noise (AWGN) channel model. We then describe the sequence ordering based on reliability information and present a HD decoding of the ordered sequence. This decoding is not new and is for instance used in [22]. However, we discuss how it relates to MLD and how to further exploit the ordering, based on the obtained codeword.

To this end, we derive the statistics of the noise after ordering in Chapter 3. From these statistics, monotonic properties of the error probability associated with each ordered symbol are derived. The efficiency of the proposed decoding algorithm strongly relies on these properties. Other immediate properties related to each ordered bit are also presented.

An algorithm exploiting the symbol ordering is described and analyzed in Chapter 4. Preserving the decoding optimality may result in many unnecessary searches or computations, as seen in optimum decoding methods presented in Section 1.2. In this chapter, a different approach is proposed. For a given range of bit error rates (BER), we simply guarantee the error performance associated with MLD. Therefore there exist two events for which the delivered codeword differs from the optimum
MLD codeword. First, the MLD codeword is correct and the decoded codeword is in error, but this event is sufficiently rare so that the optimum error performance is not significantly degraded. Second, both are in error, in which case no further degradation occurs. The algorithm is not optimal, but from a practical point of view, no difference in error performance is observed with respect to MLD, while many unnecessary computations are saved. Whenever optimality is negligibly degraded, we refer the obtained error performance as practically optimum, for the specified BER. The proposed algorithm consists of three steps: (1) reordering of the symbols of each received sequence based on their reliability as in described in Chapter 2. (2) hard decision decoding of the reordered received sequence which provides a codeword expected to have as few information bits in error as possible; and (3) reprocessing and improving the hard decision decoding progressively in stages until the optimal error performance or a desired error performance is achieved. These reprocessing stages follow directly from the statistics of the noise after ordering, and the corresponding algorithm is very straightforward. We also develop a sufficient condition to recognize the ML codeword and stop the algorithm. Since the reprocessing of the algorithm follows the monotonic properties of the error probability associated with the ordered symbols, the effectiveness of the sufficient condition increases at each step. A rapid convergence to the ML codeword is observed in most of the cases, resulting in a very low computational cost on average. Another feature of the algorithm is that after each reprocessing stage, we can tightly bound the error performance achieved based on the ordered statistics. This allows us to determine the number of reprocessing stages needed for a given code and evaluate precisely the maximum number of computations required for the worst case. This last fact is important since many optimum or suboptimum decoding algorithms, despite performing with low computational cost on average, possess a worst case computation cost which is extremely high.
and very difficult to evaluate exactly, such as the algorithms of [8, 22]. Furthermore, the proposed algorithm does not require the storage of survivors or many candidate codewords for the final decision. Simulation results on some well known codes of lengths up to 128 confirm our analysis. In this chapter, we finally show that our algorithm can be equivalently equivalent application in the dual space of the code.

In Chapter 5, different methods to further reduce the number of computations required by the algorithm are discussed. We first present additional tests which preserve the error performance. Then, the improvements and the limitations of threshold tests based on probabilistic information are discussed. Finally, both the optimum and the multi-stage adaptations of our algorithm to decomposable codes is presented.

Chapter 6 generalizes the results of the AWGN channel to other channel models. We derived the statistics of the noise after ordering for the Raleigh fading channel and apply the same decoding algorithm to this channel. Results similar to those for the AWGN model are obtained and confirmed by simulation results.

Chapter 7 constitutes the second important theoretical contribution of this thesis. We first derive a closed form approximation of the probability of error associated with each position after ordering. We then show that despite the fact that the random variables representing the noise after ordering are not independent, the error probability associated with each subset of positions behave as if they were, for a long enough sequence. Using this property, we approximate the $2^N$-state fully connected ordered channel model with $N$ parallel 2-state fully connected Binary Symmetric Channels (BSC), for an ordered sequence of length $N$. We finally show that as $N$ increases, the capacity of the discrete ordered channel converges to the capacity of the continuous Gaussian channel for BPSK transmission.

The different models using reliability information are illustrated in Chapter 8 by simple modifications of the conventional Majority-Logic-Decoding of Reed-Muller
codes. All proposed schemes achieve significant coding gain over conventional Majority-Logic-Decoding at the expense of slightly more computational complexity.

Finally, the main contributions of the thesis, as well as the directions for future research are exposed in Chapter 9.
Chapter 2

Hard Decision Decoding

and Reliability Informations

2.1 Channel Model

Figure 2.1 describes the channel model used in the main part of this thesis. Suppose an \((N, K, d_H)\) binary linear code \(C\) with generator matrix \(G\) and minimum Hamming distance \(d_H\) is used for error control over the AWGN channel. Let \(\bar{c} = (c_1, c_2, \cdots, c_N)\) be a codeword in \(C\). For BPSK transmission, the codeword \(\bar{c}\) is mapped into the bipolar sequence \(\bar{x} = (x_1, x_2, \cdots, x_N)\) with \(x_i = (-1)^{c_i} \in \{\pm 1\}\). After transmission, the received sequence at the output of the matched filter in the demodulator is \(\bar{r} = (r_1, r_2, \cdots, r_N)\) with \(r_i = x_i + w_i\), where \(w_i\) is a statistically independent Gaussian random variable with zero mean and variance \(N_0/2\). Each received noisy block is decoded independently at the receiving end. The decoder may either be an algebraic decoder which considers the IID of the received sequence \(\bar{r}\), or a soft decision decoder working directly on \(\bar{r}\). 
Figure 2.1. AWGN channel model.
2.2 Hard Decision Decoding Based on Reliability Information

2.2.1 Decoding process

If a hard decision is performed on each \( r_i \) independently, the natural choice for measure of reliability is \(| r_i |\) since for bipolar signaling, this value is proportional to the log-likelihood ratio associated with the symbol hard decision. The decoding begins with reordering the components of the received sequence \( \bar{r} \) in decreasing order of reliability value. The resultant sequence is denoted

\[
\bar{y} = (y_1, y_2, \ldots, y_N),
\]

(2.1)

with \(| y_1 | > | y_2 | > \cdots > | y_N |\). Since the noise is AWGN, we assume that \( y_i = y_j \), with \( i \neq j \) has zero probability of occurrence. This reordering defines a permutation function \( \lambda_1 \) for which \( \bar{y} = \lambda_1[\bar{r}] \). We permute the columns of the generator matrix \( G \) based on \( \lambda_1 \). This results in the following matrix

\[
G' = \lambda_1[G] = [g_1', g_2', \ldots, g_N'],
\]

(2.2)

where \( g_i' \) denotes the \( i^{th} \) column of \( G' \). Clearly the binary code \( C' \) generated by \( G' \) is equivalent to \( C \), and \( C' = \lambda_1[C] \). Since the \( i^{th} \) column \( g_i' \) of \( G' \) corresponds to the \( i^{th} \) component \( y_i \) of \( \bar{y} \) with reliability value \(| y_i |\), we call \(| y_i |\) the associated reliability value of \( g_i' \). Starting from the first column of \( G' \), we find the first \( K \) independent columns with the largest associated reliability values. Use these \( K \) independent columns as the first \( K \) columns of a new \( K \times N \) matrix \( G'' \) maintaining the decreasing order of their reliability values. The remaining \( N - K \) columns of \( G' \) form the next \( N - K \) columns of \( G'' \) arranged in the order of decreasing associated reliability values. The above process defines a second permutation function \( \lambda_2 \), such that \( G'' = \lambda_2[G'] = \lambda_2[\lambda_1[G]] \). Rearranging the components of \( \bar{y} \) according to the
permutation \( \lambda_2 \), we obtain the sequence

\[
\bar{z} = (z_1, z_2, \ldots, z_K, z_{K+1}, \ldots, z_N),
\]

(2.3)

with \( |z_1| > |z_2| > \cdots > |z_K| \) and \( |z_{K+1}| > \cdots > |z_N| \). It is clear that \( \bar{z} = \lambda_2[\bar{y}] = \lambda_2[\lambda_1[\bar{r}]] \). The first \( K \) components of \( \bar{z} \) are called the most reliable independent (MRI) symbols of \( \bar{z} \). Now we perform elementary row operations on \( G'' \) to obtain a generator matrix \( G_1 \) in systematic form,

\[
G_1 = [I_K P] = \begin{bmatrix}
1 & 0 & \cdots & 0 & p_{1,1} & \cdots & p_{1,N-K} \\
0 & 1 & \cdots & 0 & p_{2,1} & \cdots & p_{2,N-K} \\
\vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\
0 & 0 & \cdots & 1 & p_{K,1} & \cdots & p_{K,N-K}
\end{bmatrix},
\]

(2.4)

where \( I_K \) is the \( K \times K \) identity matrix and \( P \) is the \( K \times (N - K) \) parity check matrix. The code \( C_1 \) generated by \( G_1 \) is equivalent to \( C' \) and \( C \).

Next, we perform hard decisions on the \( K \) MRI symbols of \( \bar{z} \) such that, for \( 1 \leq i \leq K \),

\[
a_i = \begin{cases} 
0 & \text{for } z_i > 0 \\
1 & \text{for } z_i \leq 0
\end{cases}.
\]

(2.5)

The sequence \((a_1, a_2, \ldots, a_K)\) is used as an information sequence to form a codeword \( \bar{a} = (a_1, a_2, \ldots, a_K, a_{K+1}, \ldots, a_N) \) in \( C_1 \) based on \( G_1 \) where for \( 1 \leq j \leq N - K \),

\[
a_{K+j} = \sum_{1 \leq i \leq K} a_i p_{i,j}.
\]

The codeword \( \bar{a} \) is taken as the hard decision decoded codeword for the received sequence \( \bar{z} \). Of course, the estimate \( \hat{e}_{HD} \) for the transmitted codeword \( \bar{c} \) can be obtained by permuting the components of \( \bar{a} \) using the inverse permutation \( \lambda_1^{-1} \lambda_2^{-1} \), i.e.

\[
\hat{c}_{HD} = \lambda_1^{-1} \lambda_2^{-1}[\bar{a}].
\]

(2.6)

### 2.2.2 Computational complexity

Using "Mergesort", the ordering of the received sequence is achieved with about \( N \log_2(N) \) comparisons [24, p.172] and can be further reduced using a parallel implementation. The entire process to obtain \( G_1 \) from \( G \) can be realized in \( o(NK^2) \).
elementary binary additions [25]. However, two levels of parallelism are possible, resulting in \( K \) steps of at most \( K \) independent summations, each summation consisting of \( N \) independent binary additions. Also, for \( K > N - K \), if \( H \) represents the parity check matrix of \( G \), \( H' = \lambda_1[H] \) is clearly the parity check matrix of \( G' = \lambda_1[G] \). Then \( o(N(N - K)^2) \) elementary binary additions are now required to transform \( H' \) into \( H_1 = [P^T I_{N-K}] \) and \( G_1 \) is easily constructed from \( H_1 \). The same permutation \( \lambda_2 \) is determined when dependent columns are present. Note that while the construction of \( G_1 \) is carried out from left to right as described in [25], \( H_1 \) is formed from right to left. The exact equivalence between \( H_1 \) and \( G_1 \) is proved in Appendix A. The dominant computational term while constructing \( G_1 \) is therefore \( o(N \min(K, N - K)^2) \). Finally, computing \( \hat{c}_{HD} \) requires \( K \) sign comparisons and \( N - K \) parallel \( K \) elementary binary additions. After ordering, the whole process is realized with binary operations only.

### 2.2.3 Performance

The hard decision decoding described in Section 2.2.1 should be close to the optimum decoding. Figures 2.2 to 2.5 compare the performance of this decoding process with majority-logic-decoding for the \((32,16,8)\), \((64,22,16)\) and \((64,42,8)\) Reed-Muller (RM) codes, and with the Berlekamp-Massey algebraic decoding of the \((64,45,8)\) Extended BCH code [26]. We see that the hard decision decoding based on reliability information outperforms the algebraic decodings for low SNR's. For higher SNR's, algebraic decoding of these codes provides an improvement since its error performance curve possesses a larger slope. Note however that the decoding of \( \hat{c}_{HD} \) is not aimed to minimize the Euclidean distance between any transmitted codeword of \( C \) and the received sequence \( \mathbf{r} \), but instead it minimizes the number of information bits to be in error, which is the fact exploited by the algorithm described in Chapter 4. The observations on the error performance are confirmed by the analysis of Chapter 4. To
evaluate exactly the error performance of hard decision decoding based on ordering, we need to know the statistics of the noise after ordering. These statistics are derived in the next chapter.
Figure 2.2. Error performances for the (32,16,8) Reed-Muller code with majority-logic-decoding and hard-decision based on reliability information.
Figure 2.3. Error performances for the (64,22,16) Reed-Muller code with majority-logic-decoding and hard-decision based on reliability information.
Figure 2.4. Error performances for the \((64,42,8)\) Reed-Muller code with majority-logic-decoding and hard-decision based on reliability information.
Figure 2.5. Error performances for the (64,45,8) Extended BCH code with Berlekamp-Massey algebraic decoding and hard-decision based on reliability information.
As described in the previous chapter, reordering the received symbols with respect to their reliabilities provides an equivalent code for each block. However, the statistics of the noise after ordering have changed. In this chapter, we derive these statistics which allow one to tightly bound the error performance of any decoding algorithm involving complete or partial ordering. Note that the performance analysis of many published results should be greatly simplified when using the results of this chapter.

3.1 Ordered Sequence Statistics

We first determine the conditional distribution of the noise $W_i$ in the ordered sequence $\hat{y}$. From these statistics, we then evaluate the probability that the hard decisions of any group of information bits are jointly in error.

For $i \in \{1, N\}$, define $f_{W_i|X_i}(w_i \mid X_i = s)$ as the density function of the $i^{th}$ noise value in the ordered sequence $\hat{y}$ of length $N$, conditioned on the fact that $X_i = s$ was transmitted, where $s = \pm 1$. It can be shown that (see Appendix B)

$$f_{W_i|X_i}(w_i \mid X_i = s) = \frac{(\pi N_0)^{-N/2} N!}{(i-1)! (N-i)!} \left( \int_{-\infty}^{m(w_i)} e^{-x^2/2N_0} dx + \int_{M(w_i)}^{\infty} e^{-x^2/2N_0} dx \right)^{i-1}$$
where \( m(w_i) = \min \{2 + sw_i, -sw_i\} \) and \( M(w_i) = \max \{2 + sw_i, -sw_i\} \). Figure 3.1 depicts Equation 3.1 for \( N = 16, N_0 = 0.5 \) \( X_i = -1 \) and different values of \( i \). We observe that for \( i \neq N \), \( f_{W_i \mid X_i}(w_i \mid X_i = -1) \) is bi-modal with a null at \( w_i = 1 \) as expected, while \( f_{W_N \mid X_N}(w_N \mid X_N = -1) \) has only one mode with a discontinuity at \( w_N = 1 \). The main mode corresponds to \( w_i \in (-\infty, 1) \) and shifts to the left as \( i \) decreases. Symmetrically, the second mode shifts to the right of \( w_i = 1 \) as \( i \) decreases. We finally notice that as \( i \) decreases, the area under the right mode of \( f_{W_i \mid X_i}(w_i \mid X_i = -1) \) decreases rapidly, as shown in Figure 3.2. This remark is important since the error probability associated with position \( i \) simply corresponds to the area under the right mode, for \( i \neq N \), as shown next.

For equally likely transmission of bipolar signals normalized to \( \pm 1 \), the probability that the hard decision of the \( i^{th} \) symbol of the sequence \( \hat{y} \) of length \( N \) is in error is given by

\[
\begin{align*}
Pe(i; N) &= \int_{-\infty}^{\infty} f_{W_i \mid X_i}(w_i \mid X_i = -1) dw_i \\
&= \left( \frac{\pi N_0}{(i-1)!} \frac{1}{(N-i)!} \right) \int_{1}^{\infty} \left( \tilde{Q}(n) + 1 - \tilde{Q}(2-n) \right)^{i-1} \\
&\quad \times \left( \tilde{Q}(2-n) - \tilde{Q}(n) \right)^{N-i} e^{-n^2/N_0} dn,
\end{align*}
\]

where \( \tilde{Q}(x) = (\pi N_0)^{-1/2} \int_{x}^{\infty} e^{-n^2/N_0} dn \). For \( N > 2 \), no closed form expression for Equation 3.2 has been found. This is mainly due to the fact that the noise \( W \) is not the ordered random variable. This emphasizes the fact that our statistics differ from regular ordered statistics for which closed form expressions are possible \([28, 29, 30]\). A simple closed form approximation of \( Pe(i; N) \) is found in Chapter 7.
Figure 3.1. $f_{W|X_i}(w_i | X_i = -1)$ for $N = 16$, $N_0 = 0.5$, $X_i = -1$ and $w_i \in [-2, 2]$. 
Figure 3.2. $f_{W_i | X_i}(w_i | X_i = -1)$ for $N = 16$, $N_0 = 0.5$, $X_i = -1$ and $w_i \in [1, 2.5]$. 
Similarly, we obtain in Appendix B, for $i < j$ the joint density

\[
\begin{align*}
 f_{w_i,w_j|x_i,x_j}(w_i,w_j \mid X_i = s_i, X_j = s_j) \\
 = \frac{(\pi N_0)^{-N/2}N!}{(i-1)!(j-i-1)!(N-j)!} e^{-(w_i^2+w_j^2)/N_0} \\
 \cdot \left( \left( \int_{-\infty}^{m(w_i)} e^{-z^2/N_0} dz + \int_{M(w_i)}^{\infty} e^{-z^2/N_0} dz \right)^{i-1} \left( \int_{m(w_j)}^{M(w_j)} e^{-z^2/N_0} dz \right)^{N-j} \right) \\
 \cdot \left( \left( \int_{M(w_j)}^{\infty} e^{-z^2/N_0} dz + \int_{m(w_j)}^{m(w_i)} e^{-z^2/N_0} dz \right)^{j-i-1} \cdot 1_{[m(w_i),M(w_i)]}(w_j) \right),
\end{align*}
\]

where the indicator function of the set $A$ is $1_A(x) = 1$ if $x \in A$ and $1_A(x) = 0$ otherwise. Then, the probability that both hard decisions of the $i^{th}$ and $j^{th}$ symbols of the sequence $\hat{y}$ of length $N$ are in error is

\[
\text{Pe}(i,j;N) = \int_{-\infty}^{\infty} \int_{1}^{w_i} f_{w_i,w_j|x_i,x_j}(w_i,w_j \mid X_i = X_j = -1) dw_i dw_j.
\]

As for $\text{Pe}(i;N)$, no closed form expression has been found for $\text{Pe}(i,j;N)$ with $N > 2$, but Chapter 7 relates in a simple way the approximations of $\text{Pe}(i;N)$ and $\text{Pe}(j;N)$ to $\text{Pe}(i,j;N)$.

Generalization of these results to any larger number of considered positions becomes straightforward, as discussed in Appendix B. Equations 3.2, 3.4 or equivalent forms can also be computed from the general method described in Appendix G.

### 3.2 Monotonic Properties

In this section, we derive monotonic properties of the probabilities $\text{Pe}(i;N)$ and $\text{Pe}(i,j;N)$, which form the basis of the decoding algorithm presented in the next chapter.

**Theorem 1:**

For the ordered received sequence $\hat{y}$, the probability that the $i^{th}$ reliable digit is in
error is smaller than the probability that the $i + 1^{th}$ reliable digit is in error,

$$\text{Pe}(i; N) < \text{Pe}(i + 1; N),$$

for $1 \leq i < N$.

**Proof:**

Define $u = f(n) = \hat{Q}(2 - n) - \hat{Q}(n)$ which implies $du = (\pi N_0)^{-1/2} e^{-n^2/N_0} (1 + e^{t(n-1)/N_0}) \, dn$. From Equation 3.2 and the fact that $f(n)$ is one-to-one, we obtain

$$\text{Pe}(i + 1; N) - \text{Pe}(i; N) = \left( \begin{array}{c} N \\ i \end{array} \right) \int_0^1 (1 - u)^{i-1} u^{N-i-1} \frac{(N - i)(1 - u) - i u}{1 + e^{4(f^{-1}(u)-1)/N_0}} \, du.$$

Integrating by parts and defining $\alpha(u) = (1 + e^{4(f^{-1}(u)-1)/N_0})^{-1}$, one finds

$$\text{Pe}(i + 1; N) - \text{Pe}(i; N) = - \left( \begin{array}{c} N \\ i \end{array} \right) \int_0^1 (1 - u)^{i-1} u^{N-i-1} \frac{\partial \alpha(u)}{\partial u} \, du,$$

with

$$\frac{\partial \alpha(u)}{\partial u} = \frac{-4/N_0 \frac{\partial f^{-1}(u)}{\partial u} e^{4(f^{-1}(u)-1)/N_0}}{(1 + e^{4(f^{-1}(u)-1)/N_0})^2},$$

$$\frac{\partial f^{-1}(u)}{\partial u} = \left. \frac{\partial f(u)}{\partial n} \right|_{n=f^{-1}(u)} = \frac{(\pi N_0)^{1/2} e^{-n^2/N_0} (1 + e^{4(n-1)/N_0})}{(1 + e^{4(n-1)/N_0})^{1/2}}.$$

Equations 3.8 and 3.9 imply $\frac{\partial \alpha(u)}{\partial u} \leq 0$, so that, from Equation 3.7

$$\text{Pe}(i + 1; N) - \text{Pe}(i; N) > 0,$$

which completes the proof. □

Similarly, we can show that, for $i < j$,

$$\text{Pe}(i, j; N) < \text{Pe}(i, j + 1; N),$$

$$\text{Pe}(i, j + 1; N) < \text{Pe}(i + 1, j + 1; N),$$

24
and equivalent results hold for any number of positions considered, as expected.

**Corollary 1:** For $1 \leq i < N$,

$$
\text{Pe}(i; N) < \text{Pe}(i; N - 1). \tag{3.13}
$$

**Proof:**

Using Equation 3.2, we easily derive

$$
\text{Pe}(i + 1; N) - \text{Pe}(i; N) = \frac{N}{i} (\text{Pe}(i; N - 1) - \text{Pe}(i; N)), \tag{3.14}
$$

which implies the inequality of Equation 3.13. $\square$

Rearranging Equation 3.14, we finally obtain,

**Corollary 2:** For $1 \leq i < N$,

$$
\text{Pe}(i; N) < \text{Pe}(i + 1; N + 1). \tag{3.15}
$$

**Theorem 2:**

For $1 \leq h < i < j < N$,

$$
\text{Pe}(h, i; N) \leq \text{Pe}(i, j; N) \leq \text{Pe}(i; N). \tag{3.16}
$$

**Proof:**

The proof is immediate using the marginal density definition and the fact that, for $n \geq 1$, $0 \leq \hat{Q}(2 - n) - \hat{Q}(n) \leq 1$. $\square$

Generalization of Theorem 2 to a larger number of considered positions in $\mathcal{Y}$ follows the same way.

Some important conclusions can be derived from Theorems 1 and 2. Since the first $K$ positions of the re-ordered received sequence $\tilde{\mathbf{z}}$ satisfy Theorem 1, we are guaranteed that, when considering the positions from $K$ to 1 in $\tilde{\mathbf{a}}$, the single error probability associated with each position decreases. This result remains valid for
any number of positions considered among the $K$ MRI symbols $(z_1, z_2, \cdots, z_K)$. Also, Theorem 2 shows that the more positions we group together, the lower the probability of these positions to be jointly in error is. In the next chapter, we propose a decoding algorithm which exploits these two fundamental facts by testing a stopping criterion which becomes more effective at each decoding step.

3.3 Other Immediate Consequences

3.3.1 Recurrence relations

Next, we present several recurrence relations which can be used to evaluate $P_i(N)$ and other higher order probabilities of error. These recurrence relations reduce the computation complexity of $P_i(N)$ and $P_{i,j}(N)$ since Equations 3.2 and 3.4 are quite complex and hard to evaluate. We omit the proofs and refer to [28J, as $P_i(N)$ and $P_{i,j}(N)$ follow the same recurrence relations as the single moment of ordered statistics.

**Theorem 3:**

The following recurrence relations hold for $P_i(N)$

\[
P_i(N) = \sum_{l=N-i+1}^{N} (-1)^{i+l-n-1} \binom{N}{l} \binom{l-1}{N-i} P_{1,l}. \tag{3.17}
\]

\[
P_i(N) = \sum_{l=i}^{N} (-1)^{l-i} \binom{N}{l} \binom{l-1}{i-1} P_{1,l}. \tag{3.18}
\]

\[
P(i+1; N) = P(i; N) + \sum_{l=N-i}^{N} (-1)^{i+l-N} \binom{N}{l} \binom{l-1}{N-i} P_{1,l}. \tag{3.19}
\]

\[
P(i+1; N) = P(i; N) - \sum_{l=i}^{N} (-1)^{l-i} \binom{N}{l} \binom{l}{i} P_{1,l}. \tag{3.20}
\]

\[
P(i+1; N) = P(i; N) + \binom{N}{i} \sum_{l=0}^{i} (-1)^{l} \binom{i}{l} P_{1,N-i+l}. \tag{3.21}
\]

Similar relations hold for higher order probabilities of error, based on the relations for product moments [28]. We finally mention that while these properties are extremely
useful for computing $\text{Pe}(i; N)$'s from lower sequence orders, as with any recurrence
relation, they also tend to propagate errors.

### 3.3.2 Properties

A legitimate question is whether the statistics of $W_i$, based on the ordering, could
improve the decision device performance, assumed so far to be the usual hard limiter
given by Equation 2.5. We list in this section some properties which show that a
straightforward application of the statistics of $W_i$ can not provide any improvement.

**Lemma 1:**

The likelihood ratio, using the statistics of $W_i$, is the same as the likelihood ratio
obtained from the AWGN channel.

*Proof:*

From Equation 3.1, we observe

$$
\frac{f_{W_i | X_i} (y_i - s \mid X_i = s)}{f_{W_i | X_i} (y_i + s \mid X_i = -s)} = e^{-(y_i - s)^2 / N_0} \cdot e^{-(y_i + s)^2 / N_0},
$$

which is simply the likelihood ratio for the AWGN model. □

**Lemma 2:**

$$
\text{Pe}(N; N) \leq \frac{1}{2}.
$$

*Proof:*

Integrating Equation 3.2 by parts, one finds

$$
\text{Pe}(N; N) = 1 - (\pi N_0)^{-1/2} N \int_1^\infty \left( \tilde{Q}(n) + 1 - \tilde{Q}(2 - n) \right)^{N-1} e^{-(n-2)^2 / N_0} dn
\leq 1 - \text{Pe}(N; N),
$$

which completes the proof. □

Lemma 2 simply shows that even for the last ordered symbol of a sequence of any
length, the hard limiter provides the best symbol by symbol decision.

Lemma 3: For any integer $\alpha \in \{1, N\}$,

$$
\sum_{i_1=1}^{N-(\alpha-1)} \sum_{i_2=i_1+1}^{N-(\alpha-1)+1} \cdots \sum_{i_\alpha=i_{\alpha-1}+1}^{N} P_e(i_1, i_2, \ldots, i_\alpha; N) = \binom{N}{\alpha} \hat{Q}(1)^{\alpha}.
$$

(3.25)

Proof:

The proof is straightforward using the definition of the joint error probability considered; an equivalent expression with respect to the product moments [28] can be found also. □

Lemma 3 expresses that on average, the probability of $\alpha$ ordered bits to be in error is the same as for independent BPSK signaling.
Chapter 4
Maximum Likelihood Decoding
Based on Statistics
after Reliability Ordering

Exhaustively testing the $2^K$ possible changes in the first $K$ positions of $\bar{a}$ and selecting the codeword with smallest Euclidean distance from the received sequence $\bar{z}$ provides the optimum maximum likelihood solution. The main idea in this chapter is to take advantage of the ordering and the fact that $\bar{a}$ contains only a few information bits in error; This reduces the number of possible changes and the remaining discarded changes do not significantly affect the error performance.

4.1 Reprocessing

The proposed decoding algorithm reprocesses the hard decision decoded codeword $\bar{a}$ obtained in Section 2.2.1 until either practically optimum or a desired error performance is attained. In the following, we first describe the main idea of the reprocessing, then present the reprocessing algorithm, and finally introduce a resource test to reduce the computation complexity.
4.1.1 Definitions

Let \( \bar{a} = (a_1, a_2, \ldots, a_K, a_{K+1}, \ldots, a_N) \) be the hard decision decoded codeword at the first step of the algorithm. For \( 1 \leq l \leq K \), the order-\( l \) reprocessing of \( \bar{a} \) is defined as follows:

For \( 1 \leq i \leq l \), make all possible changes of \( i \) of the \( K \) MRI bits of \( \bar{a} \). For each change, reconstruct the corresponding codeword \( \bar{a}^o \) based on the generator matrix \( G_1 \) and determine its corresponding BPSK sequence \( \bar{x}^a \). Compute the squared Euclidean distance \( d^2(\bar{z}, \bar{x}^a) \) between the ordered received sequence \( \bar{z} \) and the signal sequence \( \bar{x}^a \), and record the codeword \( \bar{a}^* \) for which \( \bar{z}^* \) is closest to \( \bar{z} \). When all the \( \sum_{i=0}^{l} \binom{K}{i} \) possible codewords have been tested, order-\( l \) reprocessing of \( \bar{a} \) is completed and the recorded codeword \( \bar{a}^* \) is the final decoded codeword.

For \( 1 \leq i \leq l \), the process of changing all the possible \( i \) of the \( K \) MRI bits, reconstructing the corresponding codewords and computing their squared Euclidean distance from \( \bar{z} \) is referred to as phase-\( i \) of the order-\( l \) reprocessing. Clearly, the order-\( K \) reprocessing achieves the maximum likelihood decoding and requires \( 2^K \) computations.

Let \( I(K) = \{1, 2, \ldots, K\} \) be the index set for the first \( K \) positions of \( \bar{a} \). For \( 1 \leq j \leq K \), let \( \{n_1, n_2, \ldots, n_j\} \) be a proper subset of \( j \) elements of \( I(K) \). Then it follows from the ordered statistics developed in Section 3.1 that, for \( n_j \leq K \),

\[
\text{Pe}(n_1, n_2, \ldots, n_j; N) \leq \text{Pe}(n_1 + \sigma_1, n_2 + \sigma_2, \ldots, n_j + \sigma_j; N), \tag{4.1}
\]

where \( 0 \leq \sigma_i < n_{i+1} - n_i + \sigma_{i+1} \) for \( 1 \leq i \leq j \), with \( \sigma_{j+1} = 1 \) and \( n_{j+1} = K \). For \( i < j \), let \( \{m_1, m_2, \ldots, m_i\} \) be a proper subset of \( \{n_1, n_2, \ldots, n_j\} \). Then

\[
\text{Pe}(n_1, n_2, \ldots, n_j; N) < \text{Pe}(m_1, m_2, \ldots, m_i; N). \tag{4.2}
\]
From Equation 4.1, it is clear that

\[ \text{Pe}(n_1, n_2, \cdots, n_j; N) \leq \text{Pe}(K - j + 1, \cdots, K - 1, K; N), \quad (4.3) \]

where equality holds if and only if \( n_1 = K - j + 1, \cdots, n_j = K \). Let \( \pi(j) \) be the probability that there are \( j \) or more errors in the first \( K \) positions of \( \tilde{a} \). Equations 4.1 and 4.2 imply that

\[ \pi(j) \leq \binom{K}{j} \text{Pe}(K - j + 1, \cdots, K - 1, K; N). \quad (4.4) \]

Clearly, if \( \pi(l + 1) \) is sufficiently small, the order-\( l \) reprocessing of \( \tilde{a} \) would give near-optimum performance. If \( P_B \) represents the block error probability of maximum likelihood decoding, then

\[ \left( \begin{array}{c} K \vspace{1ex} \\ l + 1 \end{array} \right) \text{Pe}(K - l, \cdots, K - 1, K; N) \ll P_B \quad (4.5) \]

is a sufficient condition for order-\( l \) to provide near-optimum error performance, assuming the second permutation \( \lambda_2 \) does not significantly affect the performance. The performance analysis of the complete order-\( l \) decoding scheme given in Section 4.2 verifies this last fact. It also provides a better evaluation of the error performance of order-\( l \) reprocessing, since the bound depicted in Equation 4.5 becomes very loose for long codes. For short codes \((N \leq 32)\) and medium length codes \((32 < N \leq 64)\) with rate \( R \geq 0.6 \), we find that order-2 reprocessing achieves optimum error performance. For longer codes, order-2 reprocessing provides near-optimum performance only for high rate codes while at least order-3 reprocessing is required to achieve near optimum performance for medium to low rate long codes. In the following, we use the reliability information and the statistics after ordering to carry out the reprocessing of \( \tilde{a} \) in a systematic manner that minimizes the number of computations.

For each ordered received symbol \( z_i \), we define its two possible contributions to
the maximum likelihood decoding rule by the two squared Euclidean distances

\[ d_i(-1) = (z_i + 1)^2, \]
\[ d_i(+1) = (z_i - 1)^2. \]  

(4.6) (4.7)

For each BPSK signal sequence \( \bar{x}^\alpha = (\alpha_1, \alpha_2, \cdots, \alpha_N) \) representing a codeword \( \bar{a}^\alpha \) of \( C_1 \), we define, for \( i \in [1, N] \),

\[ \delta_i(\bar{a}^\alpha) = \delta_i(\bar{x}^\alpha) = 1/4 (d_i(\alpha_i) - d_i(-\alpha_i)) = \pm z_i, \]  

(4.8)

\[ D^+(\bar{a}^\alpha) = D^+(\bar{x}^\alpha) = \{ \delta_i(\bar{x}^\alpha) : \delta_i(\bar{x}^\alpha) > 0, i \in [1, N] \}, \]  

(4.9)

\[ D^-(\bar{a}^\alpha) = D^-(\bar{x}^\alpha) = \{ \delta_i(\bar{x}^\alpha) : \delta_i(\bar{x}^\alpha) \leq 0, i \in [1, N] \}. \]  

(4.10)

Recall that at the first step of decoding, the \( K \) MRI symbols of the ordered received sequence \( \bar{z} \) are first decoded into \( a_1, a_2, \cdots, a_K \) based on Equation 2.5. Then the remaining \( N - K \) bits \( a_{K+1}, \cdots, a_N \) are formed from \( a_1, a_2, \cdots, a_K \) and the generator matrix \( G_1 \) given in Equation 2.4. Therefore, for \( i \in [1, K] \), \( \delta_i(\bar{a}) \in D^-(\bar{a}) \) since \( \delta_i(\bar{a}) = -|z_i| \). Define

\[ \tilde{D}(\bar{a}^\alpha) = \tilde{D}(\bar{x}^\alpha) = 1/4 d^2(\bar{x}^\alpha, \bar{z}) = 1/4 \sum_{i=1}^{N}(1 + z_i^2) - 1/2 \sum_{i=1}^{N} \alpha_i z_i. \]  

(4.11)

We see that if \( \alpha_i \) is changed into \( -\alpha_i \) in Equation 4.11, \( \tilde{D}(\bar{a}^\alpha) \) is changed into \( \tilde{D}(\bar{a}^\alpha) - \delta_i(\bar{a}^\alpha) \). We define the cost function \( \Delta \) (to be minimized over \( \bar{x}^\alpha \) for maximum likelihood decoding) associated with each pair \((\bar{a}^\alpha, \bar{x}^\alpha)\) as the inner product

\[ \Delta(\bar{a}^\alpha) = \Delta(\bar{x}^\alpha) = -1/2 \sum_{i=1}^{N} \alpha_i z_i. \]  

(4.12)

Equation 4.12 shows that \( \Delta(\bar{x}^\alpha) \) is now computed only with additions.

### 4.1.2 Reprocessing algorithm

We start order-1 reprocessing by evaluating the \( \binom{K}{1} \) solutions associated by changing the decision of the \( i^{th} \) MRI symbol \( a_i \) of \( \bar{a} \), for \( i \) varying from \( K \) to 1. Since each of
these positions corresponds to a specific row of the parity check matrix \( P \), we obtain a row cost
\[
\Delta_i = \delta_i(\bar{a}) + \sum_{j=1}^{N-K} p_{ij} \delta_{K+j}(\bar{a}),
\]
where \( \delta_i(\bar{a}) < 0 \) corresponds to the cost of changing the \( i^{th} \) MRI symbol while \( \sum_{j=1}^{N-K} p_{ij} \delta_{K+j}(\bar{a}) \) represents the cost associated with this change for the parity check bits depending on the \( i^{th} \) dimension. We obtain a new codeword \( \bar{a}^1 \) with associated cost \( \Delta(\bar{a}^1) = \Delta(\bar{a}) - \Delta_i \). If \( \Delta_i > 0 \), changing the \( i^{th} \) MRI symbol reduces the cost associated with the initial decoding \( \hat{e}_{HD} \) by the amount \( \Delta_i \). At step \( K - j \), with \( j \in \{1, K\} \), we assume that we have stored the maximum value \( \Delta_{\max} = \max_{k > j} \{\Delta_k\} \) (set initially to zero). If \( \Delta_j > \Delta_{\max} \), changing the \( j^{th} \) MRI symbol improves \( \Delta(\bar{a}) \) by the biggest amount considered so far in the decoding process. In this case, we set \( \Delta_{\max} \) to \( \Delta_j \) and record the position \( j \) of the change. We only record the changed position but do not execute it as it may lead to a local minimum. When the algorithm terminates, the recorded change will be made.

For phase-\( i \) of order-\( l \) reprocessing, with \( i \leq l \), the computations of the \( \binom{K}{i} \) changes of the \( K \) MRI symbols of \( \bar{a} \) are similar, when defining,
\[
\Delta_{j_1, \ldots, j_i} = \delta_{j_1}(\bar{a}) + \cdots + \delta_{j_i}(\bar{a}) + \sum_{k=1}^{N-K} (p_{j_1,k} \oplus \cdots \oplus p_{j_i,k}) \delta_{K+k}(\bar{a}),
\]
for \( j_i > \cdots > j_1 \), where \( \oplus \) represents the addition in GF(2). When the process terminates, we change the MRI symbols at the position(s) recorded and recompute the parity check symbols. It is then straightforward to permute the symbols back based on the permutation \( \lambda_1^{-1} \lambda_2^{-1} \) and declare the resulting codeword as the (near)-optimum solution. Equation 4.14 shows that only additions are required by this algorithm.
4.1.3 Resource test

In general, decoding with order-l reprocessing requires \( \sum_{j=0}^{l} \binom{k}{j} \) computations. In the following, we introduce a resource test in the reprocessing to reduce the number of computations.

The maximum likelihood decoding process can be modeled by different graph representations. In this section, we consider a complete weighted graph \( G(E, V) \) whose vertices are the \( 2^K \) BPSK signals sequences representing the code \( C_1 \). To each directed edge joining vertex \( \bar{x}^\alpha \) to vertex \( \bar{x}^\beta \), we associate the edge cost

\[
E(\bar{x}^\alpha, \bar{x}^\beta) = 1/4 \left( d^2(\bar{x}^\alpha, \bar{z}) - d^2(\bar{x}^\beta, \bar{z}) \right). \tag{4.15}
\]

Equation 4.15 implies

\[
E(\bar{x}^\alpha, \bar{x}^\beta) = -E(\bar{x}^\beta, \bar{x}^\alpha), \tag{4.16}
\]

\[
E(\bar{x}^\alpha, \bar{x}^\beta) + E(\bar{x}^\gamma, \bar{x}^\alpha) = E(\bar{x}^\alpha, \bar{x}^\gamma). \tag{4.17}
\]

The maximum likelihood decoding rule can be formulated as follows: Given a starting node \( \bar{x}_S \) of \( G(E, V) \), find the node \( \bar{x}_{ML} \) of \( G(E, V) \) which maximizes \( E(\bar{x}_S, \bar{x}^\alpha) \) over all vertices \( \bar{x}^\alpha \) of \( G(E, V) \).

We choose for starting node \( \bar{x}_S \) the BPSK signal sequence representing the codeword \( \bar{a} \) obtained in Section 2.2 and upper bound, for all \( \bar{x}^\alpha \)'s of \( G(E, V) \),

\[
0 \leq E(\bar{x}^\alpha, \bar{x}_{ML}) \leq R(\bar{x}^\alpha), \tag{4.18}
\]

Using Equation 4.17, we obtain, for all vertices \( \bar{x}^\alpha \) of \( G(E, V) \),

\[
E(\bar{x}_S, \bar{x}_{ML}) \leq R(\bar{x}_S), \tag{4.19}
\]

\[
E(\bar{x}_S, \bar{x}_{ML}) \leq E(\bar{x}_S, \bar{x}^\alpha) + R(\bar{x}^\alpha). \tag{4.20}
\]

We now use Equations 4.19 and 4.20 to reduce the number of computations of the reprocessing algorithm.
For phase-i of order-l reprocessing, \( R(\bar{x}^o) \) introduced above is defined as

\[
R_i(\bar{x}^o) = \sum_{\delta_i(\bar{x}^o) \in D^+(\bar{x}^o)} \delta_i(\bar{x}^o) + \sum_{\delta_i(\bar{x}^o) \in D^-R(\bar{x}^o)} \delta_i(\bar{x}^o),
\]

(4.21)

where \( D^-R(\bar{x}^o) \) represents the \( \max\{0, d_H- |D^+(\bar{x}^o) | -i \} \) values of \( D^-(\bar{x}^o) \) with smallest magnitude (corresponding to parity check positions), and \( |D^+(\bar{x}^o)| \) denotes the cardinality of the set \( D^+(\bar{x}^o) \). The first term of Equation 4.21 represents the best improvement possible when modifying the \( |D^+(\bar{x}^o)| \) bits of \( \bar{x}^o \). Since phase-i of order-l reprocessing modifies at most \( i \) bits of \( D^-(\bar{x}^o) \) in the first \( K \) positions of \( \bar{x}^o \) and provides a codeword, at least \( \max\{0, d_H- |D^+(\bar{x}^o) | -i \} \) other bits are modified, whose (negative) contribution is bounded by the second summation of Equation 4.21. The resource function \( R_i(\bar{x}_S) \) evaluated from Equation 4.21 for \( \bar{x}^o = \bar{x}_S \) represents the best improvement possible (but hardly reachable) on the initial cost \( \Delta(\bar{a}) \) defined by Equation 4.12 for the hard decision decoded codeword \( \bar{a} \).

We identify \( \Delta_{j_1,\cdots,j_i} \) defined in Equation 4.14 as

\[
\Delta_{j_1,\cdots,j_i} = E(\bar{x}_S, \bar{x}_{j_1,\cdots,j_i}),
\]

(4.22)

where \( \bar{x}_{j_1,\cdots,j_i} \) is the BPSK signal sequence representing the codeword obtained after inverting the decoded symbols of \( \bar{x}_S \) in the MRI positions \( j_1, \cdots, j_i \). For phase-i of order-l reprocessing, we define for the BPSK signal sequence \( \bar{x}_C \) such that \( \Delta_{max} = E(\bar{x}_S, \bar{x}_C) \),

\[
R_{\text{available}}(i) = \min \{R_i(\bar{x}_S) - E(\bar{x}_S, \bar{x}_C), R_i(\bar{x}_C) \}.
\]

(4.23)

We call \( R_{\text{available}}(i) \) the available resource at phase-i. If \( \bar{x}_C \neq \bar{x}_{ML} \), Equations 4.19 and 4.20 imply that for some \( k \geq i \), there exists at least one \( \bar{x}_{j_1,\cdots,j_k} \) such that

\[
E(\bar{x}_S, \bar{x}_C) < E(\bar{x}_S, \bar{x}_{j_1,\cdots,j_k}) \leq R_{\text{available}}(i) + E(\bar{x}_S, \bar{x}_C).
\]

(4.24)

Therefore, \( -R_{\text{available}}(i) < 0 \) represents the smallest value that the contributions from the \( i \) MRI bits of \( D^-(\bar{x}_S) \) should provide to further improve \( \Delta_{max} = E(\bar{x}_S, \bar{x}_C) \).
for phase-i of order-l reprocessing. The resource test developed in the following
exploits this fact to save many unnecessary computations without affecting practically
the optimum performance of the algorithm. Note that if both \( |D^+(\bar{x}_S)| \geq d_H - i \) and \( |D^+(\bar{x}_C)| \geq d_H - i \), which is usually the case when neither \( \bar{x}_S \), nor \( \bar{x}_C \) represents the maximum likelihood codeword, then the two quantities to be minimized
in Equation 4.23 are equal.

We first derive the following simple lemma

**Lemma 4:** For \( j_1 < j_2 \leq K \),

\[
\delta_{j_1} (\bar{a}) < \delta_{j_2} (\bar{a}) < 0.
\]  

**(Proof):** For \( j_1 < j_2 \leq K \), Equations 2.5 and 4.8 give \( \delta_{j_1} (\bar{a}) = -|z_{j_1}| < 0 \) with \( |z_{j_1}| < |z_{j_2}| \) from the labeling ordering. We also assumed in Section 2.2 that \( \delta_{j_1} (\bar{a}) = \delta_{j_2} (\bar{a}) \) has zero probability of occurrence for AWGN, which completes the
proof. □

It is important to notice that the last assumption no longer holds in practical cases
where the received symbols have been previously quantized. The algorithm is then
modified in a straightforward fashion by considering the sets of positions correspond­
ing to each quantized value instead of individual positions. In the remaining parts,
we ignore this fact and implicitly assume distinct values \( \delta_i (\bar{a}) \)'s.

From Lemma 4, \(-\delta_K (\bar{x}_S) > 0 \) represents the minimum cost necessary to change
any MRI symbol(s) of \( \bar{x}_S \). \( R_1 (\bar{x}_S) \leq -\delta_K (\bar{x}_S) \) guarantees that \( \hat{\delta}_{HD} \) is optimum and
no additional computations are required. In the reprocessing algorithm presented
above, each time \( \Delta_i > \Delta_{max} \), where \( \Delta_i \) is computed according to Equation 4.13,
\( R_{available}(1) \) is updated according to Equation 4.23. As soon as \(-\delta_i (\bar{x}_S) > R_{available}(1) \)
for some \( i \), we stop computing the remaining \( \Delta_i \)'s. Lemma 4 and Equation 4.24 guar­
antee that no further improvement is possible since \(-\delta_i (\bar{x}_S) \) constitutes a decreasing
series. Including a test on $R_{\text{available}}(1)$ in the reprocessing algorithm reduces the number of computations per step without requiring a significant amount of additional operations.

In general, we update $R_{\text{available}}(i)$ according to Equation 4.23 at the beginning of phase-$i$ of order-$l$ decoding since $\Delta_{j_{i-1}, j_i} < \Delta_{\text{max}}$ does not improve the recorded decoding. In Equation 4.14, $j_i$ is incremented by “1” and for $k \in [1, i - 1]$, $j_k$ is reset to $j_{k+1} - 1$ as soon as $-\delta_{j_i}(\bar{a}) \cdots - \delta_{j_1}(\bar{a}) > R_{\text{available}}(i)$. From the above discussion, we have the following theorem.

**Theorem 4 (optimality test):**

For decoding with order-$l$ reprocessing, if there exists an $i \leq l$ such that

$$- \sum_{j=0}^{i} \delta_{K-j}(\bar{a}) \geq R_{\text{available}}(i + 1),$$

then the reprocessing stops after phase-$i$ and modifying $\bar{a}$ at the recorded positions provides the maximum likelihood decoding solution. □

Theorem 4 improves the extended distance test introduced in [31], due to reprocessing strategy described in Section 4.1.2, which exploits the ordering information. In fact, the test of [31] is simply the value $R(\bar{x}^0)$ depicted in Equation 4.18 and valid for any node of the graph $G(V, E)$, while the test of Theorem 4 considers the particular state $\bar{x}_G$. Consequently, for phase-$i$ reprocessing, $i$ contributions to the resource test from the $K$ MRI positions not only replace contributions from the least reliable positions, but also these contributions monotonically increase. In addition, for phase-$i$ reprocessing, while the available resource defined in Equation 4.23 always contains the contributions from $i$ MRI bits of $\bar{x}_C$, least reliable bits of $D^-(\bar{x}_C)$ contribute to the test of [31] only when $|D^+(\bar{x}_C)| < d_H$. Therefore, the efficiency of our new resource test is greatly improved. However, this test is tightly related to the reprocessing algorithm.
Table 4.1. Computations required for decoding with order-\(l\) reprocessing

<table>
<thead>
<tr>
<th>Operations</th>
<th>Floating point</th>
<th>Binary</th>
</tr>
</thead>
<tbody>
<tr>
<td>(</td>
<td>y</td>
<td>)</td>
</tr>
<tr>
<td>Sorting</td>
<td>(N \log_2(N))</td>
<td>(N \cdot \min(K, N - K)^2)</td>
</tr>
<tr>
<td>(G_1)</td>
<td></td>
<td>([K] + K(N - K))</td>
</tr>
<tr>
<td>(\delta_i(\tilde{a}))</td>
<td></td>
<td>([N - K])</td>
</tr>
<tr>
<td>(\Delta_{i_1,\ldots,i_l})</td>
<td>(\leq (N - K - 1) \sum_{j=1}^{l} \binom{K}{j})</td>
<td>(\leq \sum_{j=1}^{l} \binom{K}{j})</td>
</tr>
<tr>
<td>(R_i(\tilde{x}_S))</td>
<td>(\leq N - K - 1)</td>
<td></td>
</tr>
</tbody>
</table>

4.1.4 Computational analysis

The different computation costs of the algorithm are summarized in Table 4.1. In this table, the binary operations in brackets ([\(-\)]) represent sign operations. We ignore the resource updates depicted in Equation 4.23 since the computations of the \(\Delta_{i_1,\ldots,i_l}\) are largely over estimated, and consider only the dominant order of binary operations when processing the generator matrix \(G_1\).

4.2 Performance Analysis

4.2.1 Effects of the dependent positions on the performance

The statistics derived for the ordered sequence \(\tilde{y}\) in Section 3.1 do not directly apply to the reordered sequence \(\tilde{z}\) since a second permutation \(\lambda_2\) is required to obtain the generator matrix \(G_1\) with the first \(K\) columns to be linearly independent. This permutation changes the initial sorted order of \(\tilde{y}\). The dependency effect on the performance requires consideration of the generator matrix of each studied code individually. In this thesis however, we look for a general expression which could be applied to any code considered with only a slight dependency on the code parameters. We first notice that the AWGN assumption renders the column ordering independent.
of each other, and of the noise variance value $N_0/2$. The problem can be restated as: What is the probability of choosing $K + i$ columns of the code generator matrix such that the $K + i^{th}$ pick provides the $K^{th}$ independent column, for $i \in [0, N - K]$?

We represent the number of dependent columns before the $K^{th}$ independent one by the random variable $X_P$ and define $E_i$ as the event of picking $i$ dependent columns before the $K^{th}$ independent one. Then, we have

$$E_0 = \bar{E}_1,$$

$$E_{N-K} \subseteq \cdots \subseteq E_1,$$

$$\bar{E}_1 \subseteq \cdots \subseteq \bar{E}_{N-K},$$

where $\bar{E}_i$ represents the complement of $E_i$. It follows from the definitions of $X_P$ and $E_i$ that

$$P_i = P(X_P = i) = P \left( E_i \cap \bar{E}_{i+1} \right)$$

$$= P \left( E_i \cap \bar{E}_{i+1} \cap \bar{E}_{i+2} \cdots \cap \bar{E}_{N-K} \right).$$

By using total probability and Equation 4.29, we rewrite Equation 4.30 as

$$P_i = P(\bar{E}_{N-K}) \cdot P(\bar{E}_{N-K-1} | \bar{E}_{N-K}) \cdots P(\bar{E}_{i+1} | \bar{E}_{i+2}) \cdot (1 - P(\bar{E}_i | \bar{E}_{i+1})).$$

Define $N_{\text{ave}}(i)$ as the average number of columns depending on $i$ dimensions for a given code $C$ whose generator matrix is in systematic form. For the set $R$ of all possible row combinations of a given systematic generator matrix $G$,

$$N_{\text{ave}}(i, G) = \frac{\sum_{R_i \in R} n_1(R_i)}{\binom{K}{i}},$$

where $R_i$ represents any combination of $i$ rows and $n_1(R_i)$ represents the number of columns of $R_i$ containing at least one “1”. $N_{\text{ave}}(i)$ is then obtained by averaging
Table 4.2. $N_{\text{ave}}(1)$ vs $(N - K)/2 + 1$ for different linear block codes. (R=RM, G=Golay, B=BCH, e=extented)

<table>
<thead>
<tr>
<th>Code</th>
<th>$N_{\text{ave}}(1)$</th>
<th>$(N - K)/2 + 1$</th>
</tr>
</thead>
<tbody>
<tr>
<td>eG (24,12,8)</td>
<td>8.08</td>
<td>7.00</td>
</tr>
<tr>
<td>R (32,16,8)</td>
<td>9.09</td>
<td>9.00</td>
</tr>
<tr>
<td>R (32,26,4)</td>
<td>4.46</td>
<td>4.00</td>
</tr>
<tr>
<td>R (64,22,16)</td>
<td>20.24</td>
<td>22.00</td>
</tr>
<tr>
<td>R (64,42,8)</td>
<td>11.11</td>
<td>12.00</td>
</tr>
<tr>
<td>eB (64,36,12)</td>
<td>15.22</td>
<td>15.00</td>
</tr>
<tr>
<td>eB (64,45,8)</td>
<td>10.40</td>
<td>10.50</td>
</tr>
<tr>
<td>eB (128,64,22)</td>
<td>33.00</td>
<td>33.00</td>
</tr>
<tr>
<td>eB (128,99,10)</td>
<td>15.50</td>
<td>15.50</td>
</tr>
<tr>
<td>eB (128,120,4)</td>
<td>5.20</td>
<td>5.00</td>
</tr>
</tbody>
</table>

$N_{\text{ave}}(i, G)$ over all possible systematic forms of $G$. Table 4.2 compares the values $N_{\text{ave}}(1)$ obtained from simulation with $(N - K)/2 + 1$ which represents the average number of “1” per row of a generator matrix in systematic form randomly constructed. For all considered codes, both values closely match and as expected, the matching is excellent for the extended BCH codes since their weight distribution is almost binomial [25, 33]. Since $N_{\text{ave}}(1)$ represents the average number of ways to add a new dimension for each pick and at most $N - K + 1$ column choices are left before the $K^{th}$ independent column pick, we approximate the average probability to pick one independent column by

$$p = P(\bar{E}_1 \mid \bar{E}_2) = P(A_j^i) = \frac{N_{\text{ave}}(1)}{N - K + 1}, \quad (4.33)$$

where $A_j^i$ represents the event that the $j^{th}$ column of at most $i$ dependent columns is independent. With this definition, we obtain

$$P(\bar{E}_i \mid \bar{E}_{i+1}) = P \left( \cup_{j=1}^{i} A_j^i \right), \quad (4.34)$$
and as each event $A^i_j$ is assumed to be equiprobable and independent of $j$, Equation 4.34 becomes

$$P(E_i | E_{i+1}) = 1 - (1 - p)^i = 1 - \left(1 - \frac{N_{\text{ave}}(1)}{N - K + 1}\right)^i. \quad (4.35)$$

Since each dimension is present in at least $d_H$ columns of the generator matrix of any linear block code, we are guaranteed to obtain the last dependent column at the $(N - d_H)_{th}$ position in the worst case. Therefore, the maximum number of dependent columns is $N - K - d_H + 1$ and $P(E_i | E_{i+1}) = P(E_i) = 1$ for $i > N - K - d_H + 1$.

Equation 4.31 can be rewritten as

$$P_i = \left(1 - \frac{N_{\text{ave}}(1)}{N - K + 1}\right)^i \left[\prod_{j=i+1}^{N - K - d_H + 1} \left(1 - \left(1 - \frac{N_{\text{ave}}(1)}{N - K + 1}\right)^j\right)\right]. \quad (4.36)$$

For a strict analysis of the random variable $X_P$, the probability of picking each dependent column depends on both the position it occurs and the number of dependent columns already picked, for every systematic generator matrix. This analysis is beyond the scope of this study. Figures 4.1 to 4.6 depict the distribution approximation obtained from Equation 4.36 with the real distribution of $X_P$ for respectively the (24,12,8) extended Golay code, the (64,22,16) and (64,42,8) RM codes, and the (64,36,12), (64,45,8) and (128,64,22) extended BCH code. For all codes, the approximated distribution closely matches the real distribution.

### 4.2.2 Overall performance based on ordering

 Define $P_{si}$ as the probability that a contains more than $i$ errors in the first $K$ MRI positions.

**Phase-0 error performance**
Figure 4.1. Distribution of $X_P$ for the $(24,12,8)$ extended Golay code.
Figure 4.2. Distribution of $X_P$ for the (64,22,16) RM code.
Figure 4.3. Distribution of $X_P$ for the (61,42,8) RM code.
Figure 4.4. Distribution of $X_P$ for the $(64,36,12)$ extended BCH code.
Figure 4.5. Distribution of $X_p$ for the $(64,45,8)$ extended BCH code.
Figure 4.6. Distribution of $X_P$ for the $(128,64,22)$ extended BCH code.
By referring to the previous notations, we obtain

\[ P_{so} = \sum_{j=0}^{N-K} P \{ \text{at least one of the first } K + j \text{ ordered symbols is in error} | X_P = j \} P_j \]  

(4.37)

and by the using the union bound, we find

\[ P_{so} \leq \sum_{j=0}^{N-K} P_j \left( \sum_{i=0}^{K+j-1} P_e(K + j - i; N) \right). \]  

(4.38)

Equivalently,

\[ P_{so} \leq \sum_{i=0}^{K-1} P_e(K - i; N) + \sum_{i=1}^{N-K} \left( 1 - \sum_{j=0}^{i-1} P_j \right) P_e(K + i; N). \]  

(4.39)

Assuming that a single error in the MRI positions constitutes the dominant factor for errors in these \( K \) positions, \( N_{ave}(1) \) represents the average number of symbol errors in an error block after order-0 reprocessing. Therefore, for a \( (N, K, d_H) \) linear block code, the associated bit error probability \( P_{bo} \) is approximated by

\[ P_{bo} \approx \frac{N_{ave}(1)}{N} \left( \sum_{i=0}^{K-1} P_e(K - i; N) + \sum_{i=1}^{N-K} \left( 1 - \sum_{j=0}^{i-1} P_j \right) P_e(K + i; N) \right). \]  

(4.40)

In Equation 4.40, all error terms \( P_e(i; N) \) corresponding to positions \( i \leq K \) contribute to \( P_{bo} \) while error terms \( P_e(i; N) \), with \( i > K \) are weighted by their probability of occurrence. From Equation 3.25, Equation 4.40 can be rewritten as

\[ P_{bo} \approx \frac{N_{ave}(1)}{N} \left( N \hat{Q}(1) - \sum_{i=1}^{N-K} \left( \sum_{j=0}^{i-1} P_j \right) P_e(K + i; N) \right). \]  

(4.41)

Finally, when discarding the dependency factor, the bit error probability \( P_{bo} \) can be approximatively bounded by

\[ \frac{N_{ave}(1)}{N} P_e(K; N) \leq P_{bo} \leq \frac{N_{ave}(1)}{N} K P_e(K; N), \]  

(4.42)

where \( P_e(K; N) \) is defined in Equation 3.2, for \( i = K \).
Phase-i error performance

More generally, following the same method as previously,

\[
P_{bi} \approx \frac{N_{ave}(1)}{N} \left( \left( \sum_{i=1}^{N} Q(1)^{i+1} - \sum_{j_2=1}^{K+1} \sum_{j_1=i+1}^{K+j_1-1} \sum_{j_1+j_2=j_1+1}^{j_1+1} \sum_{j=0}^{j_1-1} P_j \right) \cdot \text{Pe}(K + j_1 - j_{i+1}, \ldots, K + j_1 - j_2, K + j_1; N) \right) \approx \frac{N_{ave}(1)}{N} \left( \sum_{j_1=0}^{K-(i+1)} \sum_{j_1+j_2=j_1+1}^{K-1} \text{Pe}(K - j_1, \ldots, K - j_1; N) + \sum_{j_2=1}^{N-K} \sum_{j_2=1}^{j_1-1} \sum_{j_1+j_2=j_1+1}^{j_1+1} \text{Pe}(K + j_1 - j_{i+1}, \ldots, K + j_1 - j_2, K + j_1; N) \right), \quad (4.43)
\]

and

\[
\frac{N_{ave}(1)}{N} \text{Pe}(K - i, \ldots, K - 1; K; N) \leq P_{bi} \leq \frac{N_{ave}(1)}{N} \left( \frac{K}{i} \right) \text{Pe}(K - i, \ldots, K - 1, K; N). \quad (4.44)
\]

For \( i \geq 1 \), since phase-i reprocesses the decisions delivered by order-0 decoding, the average number \( \tilde{N}_i \) of symbols in error in an error block is no longer \( N_{ave}(1) \) after modifying \( \tilde{a} \). Despite the fact that minimizing the Euclidean distance between two BPSK signal sequences does not necessarily minimize their corresponding Hamming distance, we expect \( \tilde{N}_i \leq N_{ave}(1) \). Due to the difficulty of accurately determining \( \tilde{N}_i \), we kept \( N_{ave}(1) \) in Equations 4.43 and 4.44.

4.2.3 Optimum soft decision performance

The decoding symbol error probability for maximum likelihood decoding is given by [32, p.52]

\[
\text{Pr}(\epsilon) \approx \left( \frac{d_H}{N} \right)^n d \tilde{Q} \left( \sqrt{d_H} \right), \quad (4.45)
\]

when the energy per transmitted bit is normalized to unity. In Equation 4.45, \( n_d \) represents the number of minimum weight codewords of the code. Equation 4.45 is obtained by considering only the first term of the union bound on the probability
of error. It provides a good approximation for small to medium dimension codes at medium to high SNR. For higher dimensional codes, Equation 4.45 becomes loose, even at moderate SNR. In this case, the total union bound represents a more accurate performance measure. Since most of the weight distributions of linear block codes remain unknown, the union bound is evaluated from the approximations developed in [33].

4.2.4 Algorithm performance

Define $P_s(i)$ as the probability that a decoded codeword is in error after phase-$i$ of the reprocessing. Then,

$$P_s(i) = P(\exists \tilde{x}' : d^2(\tilde{r}, \tilde{x}') \leq d^2(\tilde{r}, \tilde{x}) | \tilde{x} \text{ was transmitted,}$$

or more than $i$ MRI symbols are in error). (4.46)

Let $P_b(i)$ be the bit error probability associated with $P_s(i)$. It follows from Equation 4.45 and the union bound that

$$P_b(i) \leq Pr(\epsilon) + P_{bi}. \quad (4.47)$$

When $P_{bi} \ll Pr(\epsilon)$, the probability of having at least $i$ of the MRI symbols in error is negligible compared to $Pr(\epsilon)$ and we can stop the decoding process after phase-$(i - 1)$ reprocessing. The maximum likelihood optimum performance is not altered while at least $2^K - \sum_{j=1}^{K} \binom{K}{j}$ unnecessary computations are saved.

4.2.5 Asymptotic error performance

In this section, we describe the asymptotic behavior of order-$l$ reprocessing. As $N_0$ approaches 0, $Pr(\epsilon)$ approaches [32]

$$Pr(\epsilon) \approx e^{-d_H/N_0}. \quad (4.48)$$

50
Based on Equation 7.4, we obtain, as $N_0$ approaches 0,

$$P_{bl} \approx e^{4(1+l-\sum_{j=0}^{l} m_{K-j})/N_0},$$  \hspace{1cm} (4.49)

when assuming that the dependency factor does not influence the asymptotic performance of order-$l$ reprocessing. Also, as $N_0$ approaches 0, $m_{K-j} \approx 2 - \tilde{Q}^{-1}(1 - (K - j)/N)$ approaches 2, so that

$$P_{bl} \approx e^{-4(1+l)/N_0}.$$  \hspace{1cm} (4.50)

Combining Equations 4.48 and 4.50, the asymptotic performance of the order-$l$ reprocessing is given by

$$P_b(l) \approx \max \left\{ e^{-d_H/N_0}, e^{-4(1+l)/N_0} \right\}.$$  \hspace{1cm} (4.51)

Equation 4.51 implies that for

$$l \geq \min \{ \lfloor d_H/4 - 1 \rfloor, K \},$$  \hspace{1cm} (4.52)

order-$l$ reprocessing is asymptotically optimum. Also, whenever Equation 7.4 dominates Equation 4.51, an asymptotic coding gain is still achieved by order-$l$ reprocessing. Equation 4.52 is reached at BER off any practical range, since Equation 4.50 implies that the error probability for any subset of $l + 1$ positions is the same. However, it shows that for many codes of dimension $N$ large enough, our algorithm is not only practically optimum, but also asymptotically optimum.

### 4.3 Simulations Results

#### 4.3.1 Error performance

Figures 4.7 to 4.16 depict the error performances of the (24,12,8) extended Golay code, the (32,16,8), (32,26,4), (64,22,16), (64,42,8) RM codes, and the (64,36,12), (64,45,8), (128,64,8), (128,99,10), (128,120,4) extended BCH codes. For each code,
the simulated results for various orders of reprocessing are plotted and compared with
the theoretical results obtained from Equation 4.47. Note that for \( i \geq 2 \), the number
of computations involved in Equation 4.43 becomes extremely large. In Chapter 7,
we show that, for \( n_j > N \),

\[
\text{Pe}(n_1, n_2, \ldots, n_j; N) = \prod_{i=1}^{j-1} \left( \frac{N}{N - n_i} \right) \text{Pe}(n_i; N) \cdot \text{Pe}(n_j; N).
\] (4.53)

Therefore, to evaluate Equation 4.47 for \( i \geq 2 \), we compute the exact values for the
most significant terms which correspond to the least reliable positions considered,
and use the approximation of Equation 4.53 for the remaining secondary ones. Since
Equation 4.53 is not tight for \( n_j \) close to \( N \), Equation 4.43 must imperatively be
computed in its second form.

For all codes considered, we observe a close match between the theoretical and sim-
ulated results. For the \((24,12,8)\) extended Golay code, order-2 reprocessing achieves
the optimum error performance. In fact, for this code, we also simulated the optimum
decoding and found absolutely no difference in error performance between the decoding
with order-2 reprocessing and the optimum decoding. The two decoded streams
are not identical, but whenever the optimum decoded block and the order-2 repro-
cessing block differ, both decoded codewords are always in error. This fact perfectly
illustrates the efficiency of our approach, since despite the fact that the decoded codewords
are not always the ML optimum codeword, no practical difference is observed.
There is a small performance degradation with order-1 reprocessing. At the BER
\( \text{Pe} = 10^{-6} \), we observe at most a 0.16 dB loss compared to the optimum decoding
and at most a loss of 0.3 dB at \( \text{Pe} = 10^{-11} \), when using the bound of Equation 4.47
with \( i = 1 \). Similar results hold for short codes of length \( N \leq 32 \) and rate \( R \geq 0.3 \), as
well as for medium length codes of length \( 32 < N \leq 64 \) and rate \( R \geq 0.6 \). For these
two classes of codes, at a practical error performance, order-2 reprocessing achieves
Figure 4.7. Error performances for the $(24,12,8)$ extended Golay code.
Figure 4.8. Error performances for the (32,16,8) RM code.
Figure 4.9. Error performances for the (32,26,4) RM code.
Figure 4.10. Error performances for the (64,22,16) RM code.
Figure 4.11. Error performances for the (64,42,8) RM code.
Figure 4.12. Error performances for the (64,36,12) extended BCH code.
Figure 4.13. Error performances for the \((64,45,8)\) extended BCH code.
Figure 4.14. Error performances for the (128,64,22) extended BCH code.
Figure 4.15. Error performances for the (128,99,10) extended BCH code.
Figure 4.16. Error performances for the (128,120,4) extended BCH code.
optimum error performance, while order-1 reprocessing results in a good trade-off between error performance and computational complexity. At the BER $P_e = 10^{-6}$, for order-1 reprocessing, the SNR loss is at most $0.3$ dB for the $(32,16,8)$ RM code, $0.4$ dB for the $(64,42,8)$ RM code and $0.47$ dB for the $(64,45,8)$ extended BCH code. Order-1 reprocessing achieves optimum error performance for the very high rate $(32,26,4)$ RM code. For codes of length $32 < N \leq 64$ and rate $0.3 \leq R < 0.6$, order-3 reprocessing achieves optimum error performance, while near optimum performance is reached by order-2 reprocessing. At the BER $P_e = 10^{-6}$, we observe a SNR gap between order-2 and order-3 performance curves of at most $0.25$ dB for both the $(64,36,12)$ extended BCH code and the $(64,22,16)$ RM code. Therefore, order-2 reprocessing offers an excellent trade-off between error performance and computation complexity for codes of length $N = 64$ and rate $0.3 \leq R < 0.6$. Also, despite the fact that order-3 reprocessing is required to achieve optimum error performance, the corresponding number of computations remains manageable since $K$ is relatively small. Finally, as discussed previously, we observe that the union bound computed from Equation 4.43 is rather loose. In general, for a given $N$, the SNR degradation of order-$i$ reprocessing with respect to ML decreases as the rate of the code increases.

For longer codes, order-3 or even higher orders of reprocessing might be required to achieve optimum or near optimum error performance. This order depends on both the code length and the rate. For the $(128,120,4)$ extended BCH code, order-1 reprocessing achieves optimum error performance for low and high SNR's, while about $0.1$ dB SNR loss is observed at medium SNR. This unusual behavior is due to the fact that $P_b$ dominates Equation 4.47 only for medium SNR values, as observed from the theoretical curves of Figure 4.16. For the $(128,99,10)$ extended BCH code, decoding with order-3 reprocessing achieves optimum error performance, while order-2 reprocessing provides near optimum performance, within $0.1$ dB of the optimum per-
formance for BER $\text{Pe} \geq 10^{-6}$. In fact, order-1 reprocessing for this code provides an excellent trade-off between the error performance and decoding complexity. Despite a degradation in performance of 1.1 dB compared with the optimum decoding, order-1 reprocessing still achieves a 5.5 dB coding gain over the uncoded BPSK with very few computations. For the (128,64,22) extended BCH code, order-4 reprocessing is required to achieve optimum error performance for $\text{BER} \geq 10^{-6}$. At the BER $\text{Pe} = 10^{-6}$, the optimum coding gain is 7.0 dB over the uncoded BPSK, while order-2 and order-3 reprocessings achieve 5.6 and 6.5 dB coding gains respectively. Note finally that for $\text{Pe} < 10^{-7}$, even order-4 reprocessing no longer achieves optimum error performance for this code.

4.3.2 Number of computations

**(24,12,8) extended Golay code**

The best known optimum decoding algorithm for the (24,12,8) extended Golay code is provided in [36]. This decoding method requires at most 651 addition-equivalent operations. For this code, order-2 decoding achieves optimum maximum likelihood error performance, as shown in Figure 4.7. Evaluating the extreme worst case from Table 4.1 while ignoring the binary operations provides 89 comparisons for ordering, at most $N - K - 1 = 11$ operations for order-0, at most $(N - K) K = 144$ operations for phase-1 and at most $(N - K) \left( \frac{K}{2} \right) = 792$ operations for phase-2, so a total of 1,036 addition-equivalent operations for order-2 reprocessing. We define $c_{\text{ave}}$ and $c_{\text{max}}$ as respectively the average and the maximum number of processed code-words per block. When simulating 250,000 uncoded blocks, Tables 4.3 and 4.4 depict the corresponding average number of computations $N_{\text{ave}}$ and maximum number of computations $N_{\text{max}}$ for respectively order-1 and order-2 reprocessing of this code at different probabilities of error. These values are also compared in Figure 4.17. For

64
Table 4.3. Order-1 simulation results for \((24, 12, 8)\) extended Golay code.

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>(P_e)</th>
<th>(c_{ave})</th>
<th>(c_{max})</th>
<th>(N_{ave} = 100 + 12c_{ave})</th>
<th>(N_{max} = 100 + 12c_{max})</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.55</td>
<td>(10^{-1.56})</td>
<td>1.60</td>
<td>12</td>
<td>120</td>
<td>244</td>
</tr>
<tr>
<td>2.22</td>
<td>(10^{-1.87})</td>
<td>0.95</td>
<td>12</td>
<td>112</td>
<td>244</td>
</tr>
<tr>
<td>3.01</td>
<td>(10^{-2.37})</td>
<td>0.43</td>
<td>12</td>
<td>106</td>
<td>244</td>
</tr>
<tr>
<td>3.98</td>
<td>(10^{-3.12})</td>
<td>0.13</td>
<td>12</td>
<td>102</td>
<td>244</td>
</tr>
<tr>
<td>5.23</td>
<td>(10^{-4.47})</td>
<td>0.018</td>
<td>12</td>
<td>101</td>
<td>244</td>
</tr>
<tr>
<td>6.02</td>
<td>(10^{-5.48})</td>
<td>0.004</td>
<td>12</td>
<td>101</td>
<td>244</td>
</tr>
</tbody>
</table>

Table 4.4. Order-2 simulation results for \((24, 12, 8)\) extended Golay code (*: union bound).

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>(P_e)</th>
<th>(c_{ave})</th>
<th>(c_{max})</th>
<th>(N_{ave} = 100 + 12c_{ave})</th>
<th>(N_{max} = 100 + 12c_{max})</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.55</td>
<td>(10^{-1.56})</td>
<td>2.39</td>
<td>77</td>
<td>129</td>
<td>1,024</td>
</tr>
<tr>
<td>2.22</td>
<td>(10^{-1.90})</td>
<td>1.33</td>
<td>76</td>
<td>116</td>
<td>1,012</td>
</tr>
<tr>
<td>3.01</td>
<td>(10^{-2.40})</td>
<td>0.55</td>
<td>67</td>
<td>107</td>
<td>904</td>
</tr>
<tr>
<td>3.98</td>
<td>(10^{-3.16})</td>
<td>0.15</td>
<td>43</td>
<td>102</td>
<td>616</td>
</tr>
<tr>
<td>5.23</td>
<td>(10^{-4.57})</td>
<td>0.021</td>
<td>21</td>
<td>101</td>
<td>352</td>
</tr>
<tr>
<td>6.02</td>
<td>(10^{-5.72})</td>
<td>0.005</td>
<td>12</td>
<td>101</td>
<td>244</td>
</tr>
<tr>
<td>6.99</td>
<td>(10^{-7.5}^*)</td>
<td>0.001</td>
<td>8</td>
<td>101</td>
<td>196</td>
</tr>
</tbody>
</table>

\(P_e \leq 10^{-3}\), \(N_{max}\) becomes smaller than 651 and further decreases as the SNR increases, while \(N_{ave}\) approaches the order-0 decoding complexity. It is important to notice that after ordering and order-0 decoding, the additional 2.5 dB asymptotic gain is achieved at the expense of very few computations on average. These average number of computations differ slightly between order-1 and order-2 reprocessings, which is not surprising since the corresponding error performances are very closed. Also, most computation is due to the ordering, even at low SNR.

\((64,45,8)\) extended BCH code

The number of computations for order-1 and order-2 reprocessings of the \((64,45,8)\)
Figure 4.17. Number of computations for order-1 (x) and order-2 (o) reprocessings of the (24,12,8) extended Golay code.
extended BCH code are given in Tables 4.5 and 4.6, while Figure 4.18 compares these numbers. We observe that contrarily to the (24,12,8) Golay code, the maximum number of computations remains very high for order-2 reprocessing at practical SNR's. Also, while the average number of computations decreases exponentially for both order-1 and order-2 reprocessings, the difference between them remains important. Finally, we mention that the cost of ordering becomes secondary for both orders of reprocessing at low to medium SNR's.

### (128,64,22) extended BCH code

The simulations for order-2, order-3 and order-4 decodings of the (128,64,22) extended BCH code are recorded respectively in Tables 4.7 to 4.9. For this code, the
Figure 4.18. Number of computations for order-1 (x) and order-2 (o) reprocessings of the (64,45,8) extended BCH code.
ordering requires 769 comparisons and order-0 reprocessing is achieved with at most 63 additions. Then, the number of computations for phase-i reprocessing is evaluated from Table 4.1. These numbers are also compared in Figure 4.19. As expected, the number of computations involved in both order-3 and order-4 reprocessings are enormous. We observe that $N_{\text{ave}}$ decreases exponentially as the SNR increases and for order-3 reprocessing, reaches manageable decoding complexities at BER's met in practice. However, only the maximum number of computations of order-2 reprocessing allows a practical implementation of this decoding scheme.

This code was simulated in [22] and for the BER $P_e = 1.57 \times 10^{-12}$, their optimum decoding algorithm requires the visit of at most 216,052 graph nodes and 42 on average, for 35,000 simulated coded blocks. For 50,000 coded blocks and a similar SNR, order-4 reprocessing suffers about 0.25 dB SNR loss but requires to consider at most 21,812 codewords and 1.17 on average. Note however that in both cases, the number of simulated blocks is far too small to obtain reliable information at such an error performance and in practice, a concatenated coding scheme instead of a single code is implemented for such BER. In fact, the same number of computations was required for order-3 reprocessing of the same blocks at this particular SNR. We conjecture that for order-3 decoding, $c_{\text{max}}$ is reached within a sequence sufficiently long for the error performance considered. However, for error performances of practical interest, the order-3 decoding SNR loss is less than 0.4 dB with respect to the optimum one, with $c_{\text{max}} = 43,744$. In [22], for a similar BER, the number of nodes visited becomes enormous. We finally point out that our algorithm does not need any storage requirement.

Comparison with Trellis Decoding

Table 4.10 compares the maximum number of operations needed for order-i re-
Figure 4.19. Number of computations for order-2 (o), order-3 (x) and order-4 (*) reprocessings of the (128,64,22) extended BCH code.
Table 4.7. Order-2 simulation results for (128,64,22) extended BCH code (*: union bound).

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>$P_e$</th>
<th>$c_{ave}$</th>
<th>$c_{max}$</th>
<th>$N_{ave} = 832 + 64c_{ave}$</th>
<th>$N_{max} = 832 + 64c_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.22</td>
<td>$10^{-4.2}$</td>
<td>1,174</td>
<td>2,080</td>
<td>75,968</td>
<td>133,952</td>
</tr>
<tr>
<td>3.01</td>
<td>$10^{-3.0}$</td>
<td>502</td>
<td>2,080</td>
<td>32,960</td>
<td>133,952</td>
</tr>
<tr>
<td>3.47</td>
<td>$10^{-3.6}$</td>
<td>236</td>
<td>2,080</td>
<td>15,936</td>
<td>133,952</td>
</tr>
<tr>
<td>3.98</td>
<td>$10^{-4.4}$</td>
<td>64.0</td>
<td>2,080</td>
<td>4,928</td>
<td>133,952</td>
</tr>
<tr>
<td>4.56</td>
<td>$10^{-5.7}$</td>
<td>9.9</td>
<td>2,060</td>
<td>1,466</td>
<td>132,672</td>
</tr>
<tr>
<td>5.23</td>
<td>$10^{-6.5}$</td>
<td>0.95</td>
<td>2,035</td>
<td>893</td>
<td>131,072</td>
</tr>
</tbody>
</table>

Table 4.8. Order-3 simulation results for (128,64,22) extended BCH code (*: union bound).

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>$P_e$</th>
<th>$c_{ave}$</th>
<th>$c_{max}$</th>
<th>$N_{ave} = 832 + 64c_{ave}$</th>
<th>$N_{max} = 832 + 64c_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.22</td>
<td>$10^{-2.8}$</td>
<td>14,819</td>
<td>43,744</td>
<td>949,248</td>
<td>2,800,448</td>
</tr>
<tr>
<td>3.01</td>
<td>$10^{-4.0}$</td>
<td>4,415</td>
<td>43,744</td>
<td>283,392</td>
<td>2,800,448</td>
</tr>
<tr>
<td>3.47</td>
<td>$10^{-4.9}$</td>
<td>1,505</td>
<td>43,744</td>
<td>97,152</td>
<td>2,800,448</td>
</tr>
<tr>
<td>3.98</td>
<td>$10^{-5.8}$</td>
<td>310</td>
<td>43,237</td>
<td>20,672</td>
<td>2,768,000</td>
</tr>
<tr>
<td>4.56</td>
<td>$10^{-7.1}$</td>
<td>32.9</td>
<td>30,372</td>
<td>2,938</td>
<td>1,944,640</td>
</tr>
<tr>
<td>5.23</td>
<td>$10^{-8.3}$</td>
<td>1.17</td>
<td>21,812</td>
<td>907</td>
<td>1,396,800</td>
</tr>
</tbody>
</table>

Table 4.9. Order-4 simulation results for (128,64,22) extended BCH code (*: union bound).

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>$P_e$</th>
<th>$c_{ave}$</th>
<th>$c_{max}$</th>
<th>$N_{ave} = 832 + 64c_{ave}$</th>
<th>$N_{max} = 832 + 64c_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>2.22</td>
<td>$10^{-3.0}$</td>
<td>108,377</td>
<td>679,120</td>
<td>7,045,466</td>
<td>43,464,512</td>
</tr>
<tr>
<td>3.01</td>
<td>$10^{-4.5}$</td>
<td>21,104</td>
<td>679,096</td>
<td>1,372,721</td>
<td>43,462,976</td>
</tr>
<tr>
<td>3.47</td>
<td>$10^{-5.5}$</td>
<td>5,531</td>
<td>667,221</td>
<td>360,476</td>
<td>42,702,976</td>
</tr>
<tr>
<td>3.98</td>
<td>$10^{-7.2}$</td>
<td>847</td>
<td>480,286</td>
<td>56,016</td>
<td>30,739,136</td>
</tr>
<tr>
<td>4.56</td>
<td>$10^{-9.1}$</td>
<td>49</td>
<td>91,860</td>
<td>4,146</td>
<td>5,879,872</td>
</tr>
<tr>
<td>5.23</td>
<td>$10^{-11.3}$</td>
<td>1.17</td>
<td>21,812</td>
<td>1,038</td>
<td>1,396,800</td>
</tr>
</tbody>
</table>
Table 4.10. Decoding complexity for order-i decoding and trellis decoding of some well known codes.

<table>
<thead>
<tr>
<th>code</th>
<th>trellis decoding</th>
<th>order-i decoding</th>
<th>SNR loss (Pe = 10^-6)</th>
</tr>
</thead>
<tbody>
<tr>
<td>G(24,12,8)</td>
<td>1,351</td>
<td>1</td>
<td>≤ 0.16 dB</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>244, 1,036</td>
<td></td>
</tr>
<tr>
<td>RM(32,16,8)</td>
<td>2,399</td>
<td>1</td>
<td>≤ 0.3 dB</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>400, 2,320</td>
<td></td>
</tr>
<tr>
<td>RM(32,26,4)</td>
<td>2,095</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>290</td>
<td></td>
</tr>
<tr>
<td>RM(64,22,16)</td>
<td>131,071</td>
<td>2</td>
<td>≤ 0.25 dB</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>10,988, 75,668</td>
<td></td>
</tr>
<tr>
<td>RM(64,42,8)</td>
<td>544,640</td>
<td>1</td>
<td>≤ 0.4 dB</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1,266, 20,208</td>
<td></td>
</tr>
<tr>
<td>eBCH(64,36,12)</td>
<td>13,572,000</td>
<td>2</td>
<td>≤ 0.25 dB</td>
</tr>
<tr>
<td></td>
<td>3</td>
<td>18,996, 218,916</td>
<td></td>
</tr>
<tr>
<td>eBCH(64,45,8)</td>
<td>985,095</td>
<td>1</td>
<td>≤ 0.47 dB</td>
</tr>
<tr>
<td></td>
<td>2</td>
<td>1,194, 20,004</td>
<td></td>
</tr>
</tbody>
</table>

Processing of some well known codes with the complexity of Viterbi decoding based on their trellises. The decoding complexity of order-i reprocessing is evaluated from Table 4.1 with the assumptions of Section 4.1.4. The trellis decoding complexities are respectively taken from [15] for the (24,12,8) Golay code and the (32,16,8) RM code, [37] for the other RM codes and [38] for the extended BCH codes. For order-i reprocessing, we also indicate the loss in coding gain with respect to the practically optimum error performance at Pe = 10^-6. For the optimum error performance, our decoding complexity is much less than that of the corresponding trellis decoding. The ratio of the number of computations of both decoding schemes is about 50 for the extended BCH codes, 25 for the (64,42,8) RM code, 5 for the (32,26,8) RM code and about 1 for the other codes. The advantage of our decoding for the extended BCH codes is due to the fact that these codes do not have as good decomposable structures as RM codes or the Golay code. Therefore trellis decoding is not as efficient. For RM codes with R ≥ 0.6, our decoding approach still largely outperforms
the corresponding trellis decoding from a computational point of view since at most order-2 reprocessing is performed. For rate $R < 0.6$ RM codes or the $(24,12,8)$ Golay code, only a slight advantage remains in favour of our decoding method. Also, when reprocessing with the resource test described in Section 4.1.3, the average number of computations becomes much smaller than the maximum one for $P_e = 10^{-6}$. However, for speedup purposes, it is possible to ignore the resource test and implement order-$i$ reprocessing in parallel. Each parallel decoder is assigned a particular subset of positions to process. In such case, the comparisons of Table 4.10 become significant and suggest an advantage of our decoding scheme over trellis decoding at practical BER's. Finally, Table 4.10 also shows that further important reductions of computations are obtained for suboptimum decodings with good trade-off between error performance and computational complexity.

**Comparison with the Chase Algorithm 2**

In [6], Chase adds to the HD decoding $\bar{d}$ of the received sequence all possible error patterns $\bar{e}$ obtained by modifying any of the $[d_H/2]$ least reliable bits. These $2^{[d_H/2]}$ vectors are successively decoded by an algebraic decoder, and the best solution is recorded. Therefore, whenever the ML error performance is not achieved, the error performance of this algorithm is dominated by the event that $t+1$ transmission errors are present in the $N - [d_H/2]$ first most reliable positions of the ordered received sequence [6]. In addition to compare the Chase algorithm 2 with our reprocessing method, we propose to use the results of Section 3.1 to provide a new analysis of the error performance of this algorithm. Tighter bounds on the error performance than in [6] are derived.

From the results of Section 4.2.4, the bit error probability $P_b(C2)$ associated with
the Chase algorithm 2 is bounded by

$$\left(\frac{d_H}{N}\right) \max \left\{ \tilde{Q} \left(\sqrt{d_H}\right), \text{Pe} \left(N - \lfloor d_H/2 \rfloor - t, \ldots, N - \lfloor d_H/2 \rfloor; N\right) \right\} \leq P_e(C2)$$

$$\leq \left(\frac{d_H}{N}\right) \left( \sum_{w_i \in W_d} n_{w_i} \frac{1}{\sqrt{w_i}} \right) \sum_{j_1=1}^{N-\lfloor d_H/2 \rfloor} \ldots \sum_{j_{t+1}=j_{t+1}}^{N-\lfloor d_H/2 \rfloor} \text{Pe} \left(j_1, j_2, \ldots, j_{t+1}; N\right),$$

(4.54)

with $j_1 < j_2 < \cdots < j_{t+1}$, $W_d$ representing the weight distribution of the code considered and $n_{w_i}$ the number of codewords of weight $w_i$. Equation 4.54 shows that at high SNR, the error performance of the Chase algorithm 2 is dominated by

$$\max \left\{ \tilde{Q} \left(\sqrt{d_H}\right), \text{Pe} \left(N - \lfloor d_H/2 \rfloor - t, \ldots, N - \lfloor d_H/2 \rfloor; N\right) \right\}. \quad (4.55)$$

Figures 4.20 and 4.21 compare the simulation results of the Chase algorithm 2 with our reprocessing algorithm for the (32,16,8) and (64,42,8) RM codes. We also plotted the theoretical bounds of Equation 4.54. We use Equation 4.53 to evaluate all terms with secondary contribution to the union bound. We verify, using Equations 4.47 with $i = 1$ and the upper bound of Equation 4.54 that the Chase algorithm 2 starts outperforming order-1 reprocessing for the BER $P_e = 10^{-7.6}$ for the (32,16,8) RM code and $P_e = 10^{-10.1}$ for the (64,42,8) RM code. At these crossovers, respectively 0.35 dB for the (32,16,8) RM code and 0.65 dB for the (64,42,8) RM code SNR gaps remain between the the Chase algorithm 2 and the order-2 reprocessing error performance curves, obtained from Equations 4.47 and 4.54. For both codes no practical difference in error performance is observed between order-2 reprocessing and ML decoding. As expected from Corollary 2 generalized, for a fixed $d_H$, the performance degradation of the Chase algorithm 2 with respect to the optimum ML error performance at a particular BER increases with $N$. Therefore, at practical BER's, few orders of reprocessing are sufficient to outperform the Chase algorithm 2.
Figure 4.20. Error performances for order-$i$ reprocessing and Chase algorithm 2 decoding of the $(32,16,8)$ RM code.
Figure 4.21. Error performances for order-\(i\) reprocessing and Chase algorithm 2 decoding of the (64,42,8) RM code.
4.4 Equivalent Algorithm in the Dual Space of the Code

In Section 2.2.2, we showed how to obtain the parity check matrix $H_1 = [P^T I_{N-K}]$ of the equivalent code after ordering. In this section, we apply our reprocessing algorithm directly to $H_1$, which is equivalent to the approach of [9, 10, 11]. For a given code, error performances and decoding complexities equivalent to the original algorithm are achieved. Based on the ordering, an efficient and structured search on the columns of $H_1$ is realized.

4.4.1 Definitions

From the BPSK sequence $\bar{d}$ representing the HD decoding of $\bar{z}$ and defined in Section 4.1.3, we compute the syndrome

$$\bar{s} = \bar{d} H_1^T.$$  \hspace{1cm} (4.56)

As defined in [9], the ML decoding rule becomes: "If $\bar{s} \neq \bar{0}$, find the set $L$ of columns $\bar{h}_{1,i}$ of $H_1$, $i \in L$, such that the sum of these columns is $\bar{s}$ and the sum of the decoding costs associated with each column minimizes the sum of the decoding costs of any group of columns summing to $\bar{s}$". If we define $\bar{e} = (e_1, e_2, \cdots, e_N)$ with $e_i = 1$ if $i \in L$ and $e_i = 0$ otherwise, then $\lambda_1^{-1} \lambda_2^{-1} [\bar{c}_d @ \bar{e}]$ is the ML codeword, where $\bar{c}_d$ is the binary $N$-tuple represented by the BPSK sequence $\bar{d}$.

For $1 \leq l \leq K$, the order-$l$ reprocessing of $H_1 = [P_1 I_{N-K}]$ is now defined as follows:

For $1 \leq i \leq l$, sum all possible groups $L_i$ of $i$ columns of $P_1$. Denote $\bar{s}_1 = (s_{1,1}, s_{1,2}, \cdots, s_{1,N-K})$ as the obtained vector. Construct the vector $\bar{s}_2 = \bar{s} \oplus \bar{s}_1$, and then the codeword $\bar{c}_d \oplus [\bar{e}_1 \bar{e}_2]$, where $\bar{e}_1 =$
with \( e_1,i = 1 \) if \( j \in L_i \) and \( e_1,i = 0 \) otherwise, and \( \bar{e}_2 = \bar{e}_2 \). For each reconstructed codeword, determine its corresponding BPSK sequence \( \bar{\mathbf{r}}^o \). Compute the squared Euclidean distance \( d^2(\bar{z}, \bar{\mathbf{r}}^o) \) between the ordered received sequence \( \bar{z} \) and the signal sequence \( \bar{\mathbf{r}}^o \), and record the codeword \( \bar{\mathbf{a}}^* \) for which \( \bar{\mathbf{r}}^o \) is closest to \( \bar{z} \). When all the \( \sum_{i=0}^{l} \binom{K}{i} \) possible codewords have been tested, order-l reprocessing of \( \bar{\mathbf{a}} \) is completed and the recorded codeword \( \bar{\mathbf{a}}^* \) is the final decoded codeword.

For \( 1 \leq i \leq l \), the process of summing all the possible groups of \( i \) columns of \( P_1 \), reconstructing the corresponding codewords and computing their squared Euclidean distance from \( \bar{z} \) is referred to phase-i of the order-l reprocessing. Note that phase-i reprocessing considers the sum of any group of \( i \) columns in the \( K \) MRI positions, which are indeed the same as before, due to the duality between \( H_1 \) and \( G_1 \). Then it adds the obtained \((N-K)\)-tuple to columns of \( I_{N-K} \) to form the syndrome \( \bar{\mathbf{s}} \). After order-l reprocessing, the recorded codeword is not the ML codeword if and only if more than \( l \) HD errors are present in the \( K \) MRI positions. The order-l reprocessing algorithms from \( G_1 \) and from \( H_1 \) are therefore equivalent, and have the same error performance.

### 4.4.2 Resource test

Based on Equation 4.8, we associate with \( \bar{\mathbf{e}} = [\bar{e}_1 \bar{e}_2] \) the decoding cost

\[
\Delta(\bar{\mathbf{e}}) = \sum_{j=1}^{N} e_j \delta_j(\bar{\mathbf{e}}) = \sum_{j=1}^{K} e_1,j \delta_{1,j}(\bar{\mathbf{e}}) + \sum_{j=1}^{N-K} e_2,j \delta_{2,j}(\bar{\mathbf{e}}),
\]

and record the minimum decoding cost \( \Delta_{\text{min}} \) and its corresponding codeword \( \bar{\mathbf{a}}^* \).

For any codeword candidate \( \bar{\mathbf{a}}^o \), phase-i reprocessing sums \( i \) columns of \( P_1 \). Then,
columns of $I_{N-K}$ are also added to form the syndrome $\bar{s}$. If $w(\bar{s})$ denotes the Hamming weight of $\bar{s}$, we need

$$i + N(\bar{a}^\omega) + w(\bar{s}) \geq d_H,$$  \hspace{1cm} (4.58)

since for a code of minimum distance $d_H$, no less than $d_H$ columns of its parity check matrix sum to 0 [26, p.64]. Let $D_i^R(\bar{s})$ denotes the set of the smallest $\max\{0, d_H - i - w(\bar{s})\}$ $\delta_j(\bar{c}_d)$'s for which $s_j = 0$. If $\bar{e}^\omega$ represents the error associated with $\bar{a}^\omega$, $\bar{e}^\omega = \bar{a}^\omega \oplus \bar{c}_d$ and we obtain

$$\Delta(\bar{e}^\omega) \geq \sum_{x=1}^{i} \delta_{1,x}(\bar{c}_d) + \sum_{j_x \in D_i^R(\bar{s})} \delta_{2,x}(\bar{c}_d).$$ \hspace{1cm} (4.59)

Let assume $\bar{a}^*$ has been recorded at phase-$k$ reprocessing, $k \leq i$, so that $\Delta(\bar{e}^*) = \Delta_{\text{min}}$. Then, at phase-$i$ reprocessing, for $\bar{a}^\omega \neq \bar{a}^*$,

$$i + N(\bar{a}^\omega) + k + N(\bar{a}^*) \geq d_H.$$ \hspace{1cm} (4.60)

We define $D_i^R(\bar{a}^*)$ as the set of the smallest $\max\{0, d_H - i - k - N(\bar{a}^*)\}$ $\delta_j(\bar{c}_d)$'s for which $e_{2,j} = 0$. This definition generalizes the definition of $D_i^R(\bar{s})$ since for order-0 reprocessing, $\bar{e}_2 = \bar{s}$. Similarly to Equation 4.59, we get

$$\Delta(\bar{e}^\omega) \geq \sum_{x=1}^{i} \delta_{1,x}(\bar{c}_d) + \sum_{j_x \in D_i^R(\bar{a}^*)} \delta_{2,x}(\bar{c}_d).$$ \hspace{1cm} (4.61)

For phase-$i$ of order-$l$ reprocessing, we compute for each codeword $\bar{a}^\omega$ its associated cost

$$\Delta(\bar{e}^\omega) = \sum_{x=1}^{i} \delta_{1,x}(\bar{c}_d) + \sum_{j_x \in S(\bar{a}^\omega)} \delta_{2,x}(\bar{c}_d),$$ \hspace{1cm} (4.62)

so that $S(\bar{a}^\omega) \cup \{j_1, j_2, \cdots, j_i\}$ represents the set of positions where the HD sequence $\bar{d}$ is modified. Equation 4.9 implies

$$S(\bar{a}^\omega) \cup \{j_1, j_2, \cdots, j_i\} = D^+(\bar{a}^\omega).$$ \hspace{1cm} (4.63)
Combining Equations 4.59 and 4.61, a necessary condition to process $\bar{a}^\alpha$ is

$$\sum_{x=1}^{i} \delta_{1,j_x}(\bar{c}_d) \leq \Delta_{\text{min}} - \max \left\{ \sum_{j_x \in D^n(x)} \delta_{2,j_x}(\bar{c}_d), \sum_{j_x \in D^n(x^*)} \delta_{2,j_x}(\bar{c}_d) \right\}. \quad (4.64)$$

When considering the positions $j_1, j_2, \cdots, j_i$ in systematic decreasing order for phase-$i$ reprocessing, the equality of Equation 4.63 implies that Equation 4.64 is equivalent to the resource test described in Section 4.1.3. Therefore, the described algorithm in the dual space is exactly the same as before, both from a performance and computation point of views. Its only advantage is whenever $N - K < K$, we store and process a smaller matrix. Also, the derivation of the resource test is simpler with this approach since $\bar{d}$ is the starting point of the reprocessing algorithm. In Section 4.1.3, a cost difference with respect to $\bar{d}$ is introduced, which complicates the analysis. With this version of our algorithm, an efficient search method valid for any code is now available for ML decoding in the dual space of the code.
Chapter 5
Further Reduction of
the Number of Computations

5.1 Additional Optimum Tests

5.1.1 Outer test

The resource test presented in Section 4.1 applies only for the reprocessing part of the algorithm. However, it might be useless to sort the received sequence when only few errors are present within the coded block. This suggests an outer test $T_{\text{init}}$ based on an initial algorithm which is computationally cheap. This outer test provides a first decision and decides either to accept this decoding, or to ignore it and process the algorithm $A_{\text{order}}$ of Section 4.1.2, as depicted in Figure 5.1. The overall error performance $P_e(A)$ is upper bounded by

$$P_e(A) \leq P_e(A_{\text{order}}) + P_e(T_{\text{init}}),$$

(5.1)

where $P_e(A_{\text{order}})$ and $P_e(T_{\text{init}})$ represent the error performances of respectively the algorithm $A_{\text{order}}$ and the outer test ($T_{\text{init}}$). For $P_e(T_{\text{init}}) \ll P_e(A_{\text{order}})$, $P_e(A) \approx P_e(A_{\text{order}}) \approx Pr(\epsilon)$.

Suppose we choose $T_{\text{init}}$ as the syndrome test for the hard decision of the received sequence $\tilde{r}$ and the AWGN channel is modeled by the Binary Symmetric Channel
Figure 5.1. Outer test modification.
Table 5.1. Computation savings with syndrome test for (24,12,8) extended Golay code.

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>Pe</th>
<th>$(1 - \tilde{Q}(1))^N$</th>
<th>$N_T/N_{tot}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.55</td>
<td>$10^{-1.8}$</td>
<td>0.052</td>
<td>0.052</td>
</tr>
<tr>
<td>3.01</td>
<td>$10^{-2.4}$</td>
<td>0.140</td>
<td>0.141</td>
</tr>
<tr>
<td>3.98</td>
<td>$10^{-3.1}$</td>
<td>0.245</td>
<td>0.246</td>
</tr>
<tr>
<td>5.23</td>
<td>$10^{-4.8}$</td>
<td>0.437</td>
<td>0.434</td>
</tr>
<tr>
<td>6.02</td>
<td>$10^{-6.0}$</td>
<td>0.576</td>
<td>0.574</td>
</tr>
<tr>
<td>6.99</td>
<td>$10^{-7.5}$</td>
<td>0.736</td>
<td>0.733</td>
</tr>
</tbody>
</table>

(BSC) with cross-over probability $\epsilon = \tilde{Q}(1)$, where $\tilde{Q}(x)$ is given after Equation 3.2. First, we check that

$$\text{Pe}(T_{init}) \leq \sum_{i: \omega_i \in W(C)} \omega_i \tilde{Q}(1)^i \left(1 - \tilde{Q}(1)\right)^{N-i} \ll \text{Pr}(\epsilon), \quad (5.2)$$

where $W(C)$ denotes the weight distribution of the code $C$ considered. The average number of computations required becomes

$$N_{ave}(A) = \left(1 - \tilde{Q}(1)^N\right) N_{ave}(A_{order} | T_{init} < 0) + \left(1 - \tilde{Q}(1)^N\right) N(T_{init}), \quad (5.3)$$

where $N(T_{init})$ is the number of computations for computing the syndrome, and $N_{ave}(A_{order} | T_{init} < 0)$ is the average number of computations realized by the algorithm $A_{order}$ when the syndrome is non-zero. Indeed, $N_{ave}(A_{order} | T_{init} < 0) > N_{ave}(A_{order})$, and for the syndrome test, $T_{init} > 0$ implies order-0 is optimum since Equation 5.2 guarantees no transmission errors in the coded block considered, so that $D^+(\bar{x}_S) = \emptyset$. Therefore, $\left(1 - \tilde{Q}(1)^N\right)$ represents the fraction of sorting savings in this case. These savings were not taken into account when we evaluated the performance of our algorithm since outer tests can be adapted to any algorithm. However, the savings may vary significantly from one algorithm to another.

Table 5.1 depicts the savings on sorting for the (24,12,8) extended Golay code.
when initially testing the syndrome. The ratio $N_T/N_{tot}$ represents the ratio of the number of satisfied syndrome tests to the total number of simulated blocks. We observe no relevant difference between the simulated and theoretical savings. For this code, at $Pe \approx 10^{-6}$, the syndrome test is satisfied at 57%. About the same saving is achieved for the number of computations since at this SNR, the total computational cost is almost entirely due to the sorting operation. For the (128,64,22) extended BCH code, the syndrome test provides almost no savings (0.012% at $Pe \approx 10^{-6}$). As expected, the syndrome test is efficient only when the sorting cost dominates the computational cost, which is generally the case only for codes with $N \leq 32$ at medium to high SNR's.

5.1.2 Resource improvement

In this section, we improve $R_i(x_{Ot})$ defined in Equation 4.21 by generalizing a result of [8]. Other existing algorithms, such as [22], should also benefit from the same improvement.

As in Section 4.1.3, $\bar{x}_C$ is the codeword candidate recorded by the algorithm, for which $m = |D^+(\bar{x}_C)|$. In addition, we also define $\bar{x}_0$ as the codeword candidate which minimizes $\bar{m}_0 = |D^+(\bar{x}^0)|$ over all codewords $\bar{x}^0$ processed so far by the algorithm. We let $\bar{x}'$ be any new codeword candidate to be processed by the algorithm. As in [8], we define

$$h_1 = |D^-(\bar{x}_C) \cap D^+(\bar{x}')|.$$ (5.4)

$h_1$ represents the number of positions where $\bar{x}_C$ provides a better decoding cost than $\bar{x}'$. We easily verify that the proof of Equation 7 of [8] still holds when substituting $\bar{m}_0$ to $m_0 = |D^+(\bar{x}_A)|$, where $\bar{x}_A$ is the BPSK sequence representing the codeword.
delivered by an algebraic decoder. The generalization of Equation 7 of [8] provides

$$h_1 \geq \max \left\{ d_H - \left\lfloor \frac{m + \hat{m}_0}{2} \right\rfloor, 0 \right\}. \quad (5.5)$$

Since phase-\(i\) of order-\(l\) reprocessing of \(\tilde{x}^o\) modifies at most \(i\) of the \(h_1\) bits defined in Equation 5.4 in the MRI positions of \(\tilde{x}^o\), at least \(h_1 - i\) other bits of \(D^{-}(\tilde{x}^o)\) have to be changed. For phase-\(i\) of order-\(l\) reprocessing, we define \(\tilde{D}_R(\tilde{x}^o)\) as the set of the \(d_H - [(m + \hat{m}_0)/2] - i\) values of \(D^{-}(\tilde{x})\) with smallest magnitudes. Then, Equation 4.21 can be rewritten, when substituting \(\tilde{D}_R(\tilde{x}^o)\) by \(D_R(\tilde{x}^o)\), as

$$R_i(\tilde{x}^o) = \sum_{\delta_j(\tilde{x}^o) \in \tilde{D}^+(\tilde{x}^o)} \delta_j(\tilde{x}^o) + \sum_{\delta_j(\tilde{x}^o) \in D_R(\tilde{x}^o)} \delta_j(\tilde{x}^o). \quad (5.6)$$

With the use of Equation 5.6, our algorithm becomes slightly more complex. As in Section 4.1.3, we test whether the stopping condition for phase-\(i\) of order-\(l\) reprocessing is satisfied for \(\tilde{x}'\). In addition, we also check whether \(|D^+(\tilde{x}')| < \hat{m}_0\), in which case \(\hat{m}_0\) is set to \(|D^+(\tilde{x}')|\) and \(R_{\text{available}}(i)\), defined in Equation 4.23 is updated with the new definition of Equation 5.6. Note that if the algorithm starts by testing the codeword delivered by an algebraic decoder, then \(\hat{m}_0\) takes immediately its minimum value \(m_0\) and we obtain the case considered in [8]. Also, in the worst case, \(\hat{m}_0 = m\) or equivalently, \(\tilde{x}_C = \tilde{x}_0\), which provides the criterion derived in Section 4.1.3, based on [31]. We finally mention that whenever \(\hat{m}_0 \leq t\), where \(t = [(d_H - 1)/2]\) is the error capability of the code, then \(\hat{m}_0 = m_0\).

Figure 5.2 represents the percentage of computation savings for order 1 and order 2 reprocessings of the (24,12,8) Golay code, when simulating 250,000 blocks. We observe that the percentage of savings is not very important and decreases as the SNR increases, both for order 1 and order 2 reprocessings. Also, from the simulations, we observed that \(\hat{m}_0 = m_0\) is almost never satisfied. If we update Equation 4.23 only when \(\hat{m}_0 \leq t\), the percentage of computation savings falls to less than 0.04%. For
Figure 5.2. Percentage of the computation savings for order 1 and order 2 reprocessings of the (24,12,8) Golay code.
the (128,64,22) extended BCH code, the percentage of savings provided by order 3 and order 4 reprocessings of 50,000 blocks are respectively less than 0.03% and 0.05% for $P_e \leq 10^{-2}$. In addition, $\tilde{m}_0 \leq t$ was never satisfied. Therefore, for short codes, such as the (24,12,8) Golay code, a small but interesting reduction of computations is provided by the more complex version of Equation 4.23. However, the gain becomes negligible for long codes such as the (128,64,22) extended BCH code.

These simulations confirm the fact that starting the decoding algorithm based either on a codeword which minimizes the Hamming distance from the received sequence, or equivalently the number of bits in error in the received sequence, or on a codeword which minimizes the the number of MRI bits in error in the received sequence provides two codewords which, whenever they differ, are far apart in the Euclidean space. Therefore, one can perform in parallel order-0 reprocessing as described in Section 2.2.1 and algebraic decoding. $R_{\text{available}}(i)$ is then updated with $m_0$ provided by the algebraic decoder. Since the simulations have shown that $\tilde{m}_0 = m_0$ was hardly satisfied, a more powerful resource test than the test based on the updating of $\tilde{m}_0$ is immediately available, at the expense of the algebraic decoding. It is important to notice that the algebraic decoder must always deliver the closest codeword in Hamming distance to the codeword represented by the HD BPSK sequence $\tilde{d}$ defined in Section 4.1.3. Otherwise, the efficiency of the test is greatly reduced. However, at medium to low SNR's for ML decoding, and thus at SNR's where a powerful test is mostly needed, usual algebraic decoders such as the Berlekamp-Massey decoder or Majority Logic based decoders perform poorly. They hardly deliver the codeword with smallest $m_0$ whenever more than $t$ errors occur, which is common at such low SNR's for HD decoding. More elaborated algebraic decoders, based for instance on the step by step procedure described in [39, p.38] have to be implemented.
5.2 Probabilistic Tests

The stopping criteria considered so far reduce the number of computations while preserving optimality. It should be however possible to further reduce the number of computations based on probabilistic tests.

5.2.1 On discarding some "reliable" positions

A natural way to reduce the number of computations for phase-\(i\) reprocessing is to discard \(i\)-tuples which correspond to the first MRI positions. Based on Equation 4.47, the decoding strategy for phase-\(i\) reprocessing of a given code becomes,

Find the set of \(i\)-tuples \(S_i = \{(j_1, j_2, \ldots, j_i)\}\) with largest cardinality such that

\[
\frac{N_{\text{ave}}(1)}{N} \sum_{(j_1, j_2, \ldots, j_i) \in S_i} P_e(j_1, j_2, \ldots, j_i; N) < < P_b(i). \tag{5.7}
\]

During phase-\(i\) reprocessing, do not consider the positions in \(S_i\).

Determining \(S_i\) may be extremely long and tedious, especially for medium to high dimension codes. However, the effort has to be done only once for a particular code. A weaker but simpler test is, for phase-\(i\) reprocessing of a given code,

For each \(i-1\)-tuples \((j_1, j_2, \ldots, j_{i-1})\), determine the largest \(\alpha < j_1\) such that

\[
\frac{N_{\text{ave}}(1)}{N} \left( \begin{array}{c} K \\ i \end{array} \right) P_e(\alpha, j_1, j_2, \ldots, j_{i-1}; N) < < P_b(i). \tag{5.8}
\]

During phase-\(i\) reprocessing, stop at position \(\alpha\) for the corresponding \(i-1\)-tuple \((j_1, j_2, \ldots, j_{i-1})\).

Table 5.2 contains the values of \(\alpha\) satisfying Equation 5.8 for \(P_e \geq 10^{-10}\), when considering phase-2 reprocessing of the \((24,12,8)\) Golay code. No reduction based on
Table 5.2. $\alpha$'s for phase-2 reprocessing of the (24,12,8) Golay code

<table>
<thead>
<tr>
<th>$j_1$</th>
<th>12</th>
<th>11</th>
<th>10</th>
<th>9</th>
<th>8</th>
<th>7</th>
<th>6</th>
<th>5</th>
<th>4</th>
<th>3</th>
<th>2</th>
</tr>
</thead>
<tbody>
<tr>
<td>$\alpha$</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>7</td>
<td>7</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>candidates</td>
<td>7</td>
<td>5</td>
<td>4</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>19</td>
</tr>
<tr>
<td>savings</td>
<td>4</td>
<td>5</td>
<td>5</td>
<td>6</td>
<td>6</td>
<td>6</td>
<td>5</td>
<td>4</td>
<td>3</td>
<td>2</td>
<td>1</td>
</tr>
</tbody>
</table>

Equation 5.8 is possible for phase-1 reprocessing of this code. From this table, we conclude that for error performances of practical interest, only 19 codewords have to be considered by phase-2 reprocessing. This signifies that after ordering, near optimum performance is achieved while considering only 32 codeword candidates out of the 4,096 possible. When processing only these 32 codewords, we observe from simulations that the optimum error performance of order-2 reprocessing is altered by at most 0.01 dB. Figure 5.3 depicts the corresponding computational savings. Despite reducing the maximum number of codewords from 79 candidates to 32, only a small amount of computations is saved on average. This is not totally surprising since for this code, order-1 and order-2 reprocessings have similar error performances. However, the maximum number of computations decreases from 1,036 to 472 operations. This represents a very significant reduction from a practical point of view since the worse case the decoder has to be designed for is reduced by more than half.

For the (32,26,4) RM code, the same approach shows that considering the 10 least reliable codewords of order-1 reprocessing is sufficient to achieve near-optimum performance. From Table 4.1, this corresponds to at most 194 addition-equivalent operations, with respect to 290 for the complete order-1 reprocessing. In [11], only 188 operations are required in the worst case to achieve optimum decoding. The computation gain of the algorithm of [11] is due to the ordering cost. Our algorithm requires a total ordering of the received sequence, which is achieved with at most

89
Figure 5.3. Percentage of computation savings when processing the 32 least reliable codewords of the (24,12,8) Golay code.
129 operations. In [11], at most 63 additions are sufficient to determine 9 minima. This suggests that for very high rate code, combining the approach described in this section with the algorithm of [11] should provide near optimum error performance with very few computations. For such codes, a total ordering is not necessary.

5.2.2 Threshold test

In [34], Dorsch proposes a decoding algorithm similar to the algorithm of Section 4.1, for a J-ary output channel. Starting from the same codeword \( \bar{a} \) as in Section 2.2.1, the codeword candidates are processed in increasing costs order until a given threshold is reached. At that point, the remaining codewords are discarded. In this section, we analyze the error performance of a simple stopping criterion based on a similar idea. We show the savings in computations of this approach, but also its limitations for practical applications.

**Definition**

Using the BSC model, the average number of errors per received block is simply \( N \tilde{Q}(1) \). However, for a code whose weight distribution approaches a binomial distribution, we showed in Section 4.2.1 that on average, \( (N - K)/2 \) least reliable parity check bits are in error whenever one MRI bit is wrong. In general, \( N \tilde{Q}(1) \ll (N - K)/2 \), which suggests a threshold \( T \) to distinguish between the two cases. Note however that, as in [34], the optimum value of \( T \) depends on \( N_0 \), which may limit the practical applications of this approach.

The threshold test becomes, for \( D^+(\bar{a}) \) defined in Equation 4.9.

If \( |D^+(\bar{a})| \leq T \), then accept \( \hat{c}_{HD} \) as the decoded codeword.

Else, start phase-1 reprocessing as in Section 4.1.2.
With this additional test, the algorithm stops after order-0 instead of starting phase-1 reprocessing in cases where the resource test is not satisfied. Therefore, additional computations are spared.

Performance analysis

We define $P(T)$ as the probability that $\tilde{a}$ is not the ML codeword, but satisfies the threshold test of Section 5.2.2. For order-\(i\) reprocessing, Equation 4.47 becomes

\[ P_b(i) \leq Pr(\epsilon) + P_{bi} + P_b(T), \quad (5.9) \]

where $P_b(T)$ is the bit error probability associated with $P(T)$. Based on Equation 5.9, we need to choose $T$ such as $P_b(T) \ll Pr(\epsilon) + P_{bi}$.

Let $S_e$ be the set of ordered positions $i \in [1, K]$ where $\tilde{a}$ differs from the ML codeword. The probabilities corresponding to a miss or a false alarm are respectively

\[ P_M = Pr \left( |D^+(\tilde{a})| \leq T \mid S_e \neq \emptyset \right) \quad (5.10) \]
\[ P_{FA} = Pr \left( |D^+(\tilde{a})| > T \mid S_e = \emptyset \right). \quad (5.11) \]

We observe that $P_M$ results in a decoding error while whenever $P_{FA}$ occurs, only unnecessary operations are realized.

Since $Pe(i, j; N) \ll Pe(i; N)$, we assume that single errors dominate the error performance, so that $|S_e| = 0$ or $|S_e| = 1$. If $|S_e| = 0$, a parity check position is in error with probability $p_0 = N\hat{Q}(1)/(N - K)$ on average. Therefore,

\[ Pr \left( |D^+ (\tilde{a})| = i \mid |S_e| = 0 \right) = \binom{N-K}{i} p_0^i (1-p_0)^{N-K-i}. \quad (5.12) \]

If $|S_e| = 1$, a parity check position is in error with probability $p_1 = 1/2$ on average, when assuming a binomial weight distribution, and

\[ Pr \left( |D^+ (\tilde{a})| = i \mid |S_e| = 1 \right) = 2^{-N-K} \binom{N-K}{i}. \quad (5.13) \]
Figure 5.4. $\Pr (|D^+(\bar{a})| = i \mid |S_e| = 0)$ and $\Pr (|D^+(\bar{a})| = i \mid |S_e| = 1)$ for (128,64,22) extended BCH code with SNR = 3.0dB.
Figure 5.4 depicts the theoretical distributions of Equations 5.12 and 5.13 with the simulation results for the (128,64,22) extended BCH code, with SNR = 3.0 dB. We observe an excellent matching, which justifies the validity of our assumption. Equations 5.12 and 5.13 also confirm that the optimum value of $T$ varies with $N_0$, since the distribution associated with Equation 5.12 changes with $N_0$ while the distribution described by Equation 5.13 remains fixed.

From Equations 5.10 and 5.13, we obtain

$$P_M \leq 2^{-(N-K)} \sum_{i=0}^{T} \binom{N-K}{i},$$

(5.14)

which is independent of $N_0$. Then, by applying the union bound to our assumption and since $P(T) \approx P_M \cdot \Pr(|S_e| = 1)$, we find

$$P(T) \leq \left( \sum_{i=1}^{K} P_e(i;N) \right) P_M,$$

(5.15)

or equivalently, when combining Equations 5.14 and 5.15,

$$P_e(T) \leq \left( \frac{(N-K)/2 + 1}{N} \right) \left( 2^{-(N-K)} \sum_{i=0}^{T} \binom{N-K}{i} \right) \left( \sum_{i=1}^{K} P_e(i;N) \right).$$

(5.16)

Simulation results

Figure 5.5 represents the simulation results for order-3 reprocessing of the (128,64,22) extended BCH code with $T$ taking the values 15, 17, 20 and 22. For these values, we also plotted the theoretical error performances of Equation 5.9 after substituting Equation 5.16. As expected, the error performance depends on the SNR for a fixed threshold. Figure 5.5 shows that as soon as $P_e(T)$ becomes the dominant term in Equation 5.14, the error performance rapidly diverges from the optimum performance, resulting in a poor decoding scheme. The average number of computations saved is represented on Figure 5.6, and for each point, the corresponding performance
Figure 5.5. Simulation and theoretical results for order-3 reprocessing of the (128,64,22) extended BCH code with $T \in \{15, 17, 20, 22, \infty \}$. 
Figure 5.6. Percentage of average number of computational savings for order-3 re-processing of the (128,64,22) extended BCII code with $T \in \{15, 17, 20, 22\}$ and corresponding performance degradation.
degradation with respect to the optimum order-3 reprocessing is indicated in brackets. As long as $T$ is chosen small enough to preserve the optimum error performance, important savings are obtained, as shown in Figure 5.6 for $T = 17$. We also notice from this figure that a too small value of $T$, such as $T = 15$, results in a large reduction of the full potential of computational savings.

From these results, we conclude that the simple threshold test proposed in this section may result in 30 to 40% computational savings for order-3 reprocessing of the (128,64,22) extended BCH code, which is indeed very attractive. However, these savings are possible only with an accurate choice of $T$ for a given SNR. The practical application of this scheme, as well as any probabilistic stopping criterion based on the same approach is therefore limited, since a too small value of $T$ significantly reduces the number of savings while a too large value drastically alters the error performance. Only an adaptive threshold test can provide the optimum computational savings.

### 5.3 Decomposables Codes

For long codes with large dimensions, high order reprocessing is generally required to achieve optimum performance or a desired level of error performance. High order reprocessing results in a large number of computations which render the decoding algorithm of Chapter 4 impractical for implementation. In this case, certain structural properties of a code may be used to reduce the computation complexity, e.g. decomposable properties [14, 17]. If a code can be decomposed as a multilevel code, e.g. RM codes, the proposed reprocessing scheme may be incorporated with either optimal coset decoding [15] or suboptimal multi-stage decoding [19, 20] to reduce the computational complexity. Since each component code of a decomposable code has a smaller dimension than that of the code itself, the number of computations required to reprocess the hard decision decoding for each component code will be
much reduced. This will result in a significant reduction in the overall computation complexity.

In this section, we present how to adapt the decoding algorithm of Chapter 4 to decomposable codes. We choose the \(|u|u + v| \) construction and discuss both optimum coset decoding and suboptimum closest coset decoding. Generalization to other decompositions, which may further reduce the computational complexity for some codes such as RM codes, follows in a similar way.

### 5.3.1 Coset decoding of the \(|u|u + v| \) construction

#### Definitions

We assume the generator matrix \( G \) of the \((N, K, d_H)\) code considered is of the form

\[
G = \begin{bmatrix} G_1 & G_1 \\ 0 & G_2 \end{bmatrix}
\]

(5.17)

where \( G_i \) is the generator matrix of an \((N/2, K_i, d_H)\) code \( C_i \), and \( d_H = \min(2d_{H_1}, d_{H_2}) \) [27, p.76]. Let \( c_u \) and \( c_v \) represent the codewords corresponding to the codes \( C_1 \) and \( C_2 \), and define \( c_1 = c_u, c_2 = c_u \oplus c_v \), with \( c_i = (c_{i,1}, c_{i,2}, \cdots, c_{i,N/2}) \). Then, the codeword \( c = |c_1|c_2| = c_u|c_u \oplus c_v| \) is mapped into the BPSK sequence \( \bar{x} = |x_1|x_2| \). The corresponding noisy received sequence is denoted \( \bar{y} = |y_1|y_2| = |x_1 + w_1|x_2 + w_2| \), where \( w_i \) is a \( N/2 \)-dimensional Gaussian with mean \( 0 \) and covariance matrix \( N_0/2 \cdot I_{N/2} \).

\( C \) can be viewed as the union of \( 2^{K_2} \) cosets of the \((N, K_1, 2d_{H_1})\) repetition code \( \tilde{C}_1 \) with generator matrix \([G_1G_1]\). The \( 2^{K_2} \) coset leaders are obtained by concatenating the \( N/2 \) all-0 vector with each codeword of \( C_2 \). At the receiver, each coset can be decoded independently, after removing the contribution of its coset leader from the received sequence \( \bar{y} = |\bar{y}_1|\bar{y}_2| \). For BPSK transmission of \( \bar{x} = |x_1|x_2| \), \( x_{1,i} = x_{2,i} \) whenever the \( i^{th} \) bit of the coset leader is 0 and \( x_{1,i} = -x_{2,i} \) whenever this bit is 1. Therefore, for each coset, the contribution of its coset leader is easily cancelled by
changing the sign of $y_{2,i}$ whenever the $i^{th}$ bit of the coset leader is 1. After this trivial operation, the $2^{K_2}$ decodings of the repetition code $C_1$ can be realized in parallel. Then, the final decoding is obtained by comparing the $2^{K_2}$ decoding costs of each coset decoding.

For the repetition of a BPSK symbol $\alpha = \pm 1$ over the AWGN model, the associated a posteriori probability is

$$P(x_{1,i} = \alpha \mid y_{1,i}, y_{2,i}) = (\pi N_0)^{-1/2} e^{-\frac{(y_{1,i} - \alpha)^2 + (y_{2,i} - \alpha)^2}{2N_0}}, \quad (5.18)$$

if $y_{1,i}$ and $y_{2,i}$ are the two corresponding noisy received symbols. The log-likelihood ratio $L(x_{1,i})$ associated with the repetition of a BPSK symbol over the AWGN model is therefore proportional to

$$L(x_{1,i}) \propto |y_{1,i} + y_{2,i}|. \quad (5.19)$$

Also, after scaling the contribution to the ML decoding of $\bar{y}$ as

$$d_i(\alpha) = 1/4 \left( (y_{1,i} - \alpha)^2 + (y_{2,i} - \alpha)^2 \right), \quad (5.20)$$

for $\alpha \in \{\pm 1\}$, we obtain

$$d_i(\alpha) - d_i(-\alpha) = -\alpha(|y_{1,i} + y_{2,i}|) = \pm |y_{1,i} + y_{2,i}|. \quad (5.21)$$

Equation 5.19 and 5.21 show that the decoding of the repetition code $C_1$ can be realized by decoding the code $C_1$ with the algorithm described in Chapter 4. The ordering is done with respect to $|y_{1,i} + y_{2,i}|$, which also becomes the decoding cost $\delta_i(\bar{y})$ associated with the $i^{th}$ bit.

**Performance analysis**

Without loss of generality, we assume that the all-(-1) BPSK sequence is transmitted. The corresponding received reliability measure is $| - 2 + w_{1,i} + w_{2,i}|$, which is the
magnitude of a normal random variable with mean \(-2\) and variance \(N_0\). Based on Appendix C, the same error performance as in Section 4.2 remains valid. After phase-\(i\) of order-\(l\) reprocessing, Equation 4.47 becomes

\[
P_b(i) \leq \Pr(\varepsilon) + \tilde{P}_{bi},
\]

where \(\tilde{P}_{bi}\) is obtained after substituting \(RN_0/R_1\) for \(N_0/2\) in Equation 4.43, which is defined for BPSK transmission of \(\pm 1\) over the AWGN channel with associated variance \(N_0/2\) and the code \(C_1\) with rate \(R_1 = K_1/N_1\). The ratio \(R/R_1\) is due to the energy normalization since we now encode with a rate \(R = K/N = (K_1 + K_2)/N\) code. When \(\tilde{P}_{bi} \ll \Pr(\varepsilon)\), order-\((i - 1)\) reprocessing of the repetition code \(\tilde{C}_1\) is sufficient to obtain optimum error performance.

Simulation results

Figure 5.7 compares Equation 5.22 with the simulation results for the (32,16,8) RM code. At \(\text{Pe} = 10^{-6}\), order-0 decoding of the \([16,11,4]\langle 16,11,4\rangle\) code \(\tilde{C}_1\) suffers a 0.6 dB coding loss with respect to the ML error performance and a 0.3 dB coding loss with respect to order-1 decoding of the (32,16,8) RM coder, while order-1 decoding of \(\tilde{C}_1\) achieves the optimum error performance. Based on Table 4.1, for each coset, this corresponds to at most 16 operations for the reliability measures, 49 for sorting 16 values, 4 for order-0 decoding and 55 for order-1 reprocessing, so a maximum of 124. Then, at most 31 comparisons are required to determine the decoded codeword from the coset solutions. The practically optimum error performance of the (32,16,8) RM code is achieved with Order-2 reprocessing, which requires a maximum of 2,320 operations. Indeed, \(32 \cdot 124 + 31 > 2,320\), but a parallel implementation of the 32 cosets speeds up the decoding process and it is usually cheaper to implement 32 simple order-1 decoders instead of the single order-2. The minimum trellis of the
Figure 5.7. Coset decoding of the (32, 16, 8) RM code with order-\(i\) reprocessing of the repetition code \([(16, 11, 4)][(16, 11, 4)]\).
Table 5.3. Independent order-1 simulation results for the 32 cosets of the (32, 16, 8) RM code.

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>$P_e$</th>
<th>$c_{ave}$</th>
<th>$c_{max}$</th>
<th>$N_{ave} = 69 + 5c_{ave}$</th>
<th>$N_{max} = 69 + 5c_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.55</td>
<td>$10^{-1.59}$</td>
<td>1.46</td>
<td>11</td>
<td>86</td>
<td>124</td>
</tr>
<tr>
<td>2.22</td>
<td>$10^{-1.94}$</td>
<td>1.44</td>
<td>11</td>
<td>85</td>
<td>124</td>
</tr>
<tr>
<td>3.01</td>
<td>$10^{-2.45}$</td>
<td>1.41</td>
<td>11</td>
<td>85</td>
<td>124</td>
</tr>
<tr>
<td>3.47</td>
<td>$10^{-2.78}$</td>
<td>1.39</td>
<td>11</td>
<td>85</td>
<td>124</td>
</tr>
<tr>
<td>3.98</td>
<td>$10^{-3.22}$</td>
<td>1.36</td>
<td>11</td>
<td>84</td>
<td>124</td>
</tr>
<tr>
<td>4.56</td>
<td>$10^{-3.84}$</td>
<td>1.32</td>
<td>11</td>
<td>84</td>
<td>124</td>
</tr>
<tr>
<td>5.24</td>
<td>$10^{-4.70}$</td>
<td>1.26</td>
<td>11</td>
<td>83</td>
<td>124</td>
</tr>
</tbody>
</table>

(32, 16, 8) RM code contains 8 parallel sections, each of 8 states [15]. For each section, the decoding is achieved with 479 operations [37]. While these numbers cannot be directly compared since the number of independent parallel decoders is different, they still provide information about the decoding complexity of both schemes. Also, as mentioned in [11], each coset can determine its ordering from the sorting of the 32 values $|y_{1,i} \pm y_{2,i}|$, which requires 32+129 operations. Then order-1 reprocessing is achieved within each coset with at most 59 operations. The number of computations for 1,000,000 simulated blocks with order-1 decoding of the repetition code $[(16, 11, 4)](16, 11, 4)$ are presented in Table 5.3. In this first implementation, each of the 32 cosets orders its corresponding received sequence independently and delivers its final solution after achieving order-1 reprocessing. $N_{ave}$ and $N_{max}$ represent the average and maximum number of computations per coset for order-1 reprocessing, at each decoding step. We observe that at any SNR, the maximum number of computations 124 is reached, and more surprisingly, that the average number of computations is rather stable around 85. This is due to the fact that after removing the contribution of each coset leader, some resulting sequences become so improbable that the 11 codeword candidates of order-1 reprocessing are checked. These sequences are dis-
Table 5.4. Order-1 simulation results for the survivor cosets of the (32,16,8) RM code.

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>$P_e$</th>
<th>$C_{ave}$</th>
<th>$C_{max}$</th>
<th>$c_{ave}$</th>
<th>$c_{max}$</th>
<th>$N_{ave}$</th>
<th>$N_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.55</td>
<td>$10^{-1.59}$</td>
<td>0.257</td>
<td>9</td>
<td>0.929</td>
<td>11</td>
<td>70</td>
<td>132</td>
</tr>
<tr>
<td>2.22</td>
<td>$10^{-1.94}$</td>
<td>0.144</td>
<td>8</td>
<td>0.786</td>
<td>11</td>
<td>70</td>
<td>131</td>
</tr>
<tr>
<td>3.01</td>
<td>$10^{-2.45}$</td>
<td>0.0593</td>
<td>6</td>
<td>0.580</td>
<td>9</td>
<td>70</td>
<td>119</td>
</tr>
<tr>
<td>3.47</td>
<td>$10^{-2.78}$</td>
<td>0.0325</td>
<td>5</td>
<td>0.500</td>
<td>9</td>
<td>70</td>
<td>118</td>
</tr>
<tr>
<td>3.98</td>
<td>$10^{-3.22}$</td>
<td>0.0152</td>
<td>4</td>
<td>0.402</td>
<td>9</td>
<td>70</td>
<td>117</td>
</tr>
<tr>
<td>4.56</td>
<td>$10^{-3.84}$</td>
<td>0.0056</td>
<td>3</td>
<td>0.305</td>
<td>5</td>
<td>70</td>
<td>96</td>
</tr>
<tr>
<td>5.24</td>
<td>$10^{-4.70}$</td>
<td>0.0014</td>
<td>1</td>
<td>0.05</td>
<td>1</td>
<td>70</td>
<td>74</td>
</tr>
</tbody>
</table>

carded at the final comparison stage but they dominate the computation complexity of each coset decoder.

Another approach is to include a resource test after order-0 reprocessing. As in Section 4.1.3, the available resource within each coset is determined based on order-0 reprocessing. Then, this resource is tested with respect to the minimum cost of the 32 cosets. Whenever the available resource within a coset is not sufficient to improve the minimum cost of the 32 cosets, this coset is discarded and order-1 reprocessing, as well as a final comparison are spared. This way, no significant additional cost is introduced since the 31 comparisons to select the best of 32 candidates are now realized after order-0 decoding. Only the comparisons required to select the best candidate of the survivor cosets are added, but this number is expected to be small. The number of computations for order-1 reprocessing is thus greatly reduced, as shown in Table 5.4. In this table, $N_{ave}$ and $N_{max}$ still represent the average and maximum number of computations per coset for order-1 reprocessing, at each decoding step. We also indicated the average and maximum number of cosets $C_{ave}$ and $C_{max}$ performing order-1 reprocessing at each decoding step. While depending on the SNR, these numbers remain extremely small, which indicates that this two-stage decoding
method is extremely efficient in terms of computation complexity. Since the number of remaining cosets for order-1 reprocessing is always much smaller than 32, a significant reduction of computations for order-1 reprocessing with respect to the previous implementation is achieved. For example, for 50,000 blocks at a SNR of 3.0 dB, the independent implementation of the 32 cosets requires $32 \cdot 1.41 \cdot 50,000 = 2,256,000$ operations for order-1 reprocessing, while the second version achieves the same performance with $0.0593 \cdot 0.580 \cdot 50,000 = 1,720$ operations. However, for this code, in both cases, ordering and order-0 reprocessing represent the dominating cost. Comparing Tables 5.3 and 5.4 confirms the fact that in the first implementation, most of the computations for order-1 reprocessing are processed within unprobable cosets which are discarded at the final selection stage. In the second approach, order-0 reprocessing is often optimum for the survivor cosets since $N_{\text{ave}} < 1$. In fact, $c_{\text{ave}} << 1$ indicates that in most of the cases, no survivor cosets are found. Then the code-word corresponding to the coset of minimum cost immediately becomes the decoding solution. This fact is not surprising since for the (32,16,8) RM code, order-0 coset decoding performs close to optimality, as shown on Figure 5.7. In general, for the second coset decoding approach, order-0 decoding allows one to dynamically select the survivor cosets in which order-1 reprocessing should be pursued. Generalization of this method to other decomposable codes, with possibly higher reprocessing orders, should also provide excellent results, as long as the number of cosets or the optimum reprocessing order remain manageable.

Similarly, we find that order-2 reprocessing of the repetition code $[(32,16,8)](32,16,8)$ for each of the 64 cosets of the (64,22,16) RM code achieves optimum error performance, while order-1 reprocessing achieves near optimum error performance. For each coset, we realize at most $32 + 129 + 15 + 256 + 1,920 = 2,352$ operations for order-2 reprocessing and 432 operations for order-1 reprocessing, while for similar error per-
performances, order-3 and order-2 reprocessings of the (64,22,16) RM code require at most respectively $321 + 41 + 924 + 9,702 + 64,680 = 75,668$ and $10,988$ operations. Also, the trellis decoding of the (32,16,8) RM code requires 2,399 operations [15]. The minimum trellis of the (64,22,16) RM code based on the 4-level decomposition also has 64 parallel sections, each of 16 states [15]. The trellis decoding is achieved with 2,047 operations per section [37], which suggests that application of our algorithm to the 4-level decomposition should further improve the decoding cost of this code. Also, the algorithm of [11] can no longer be applied efficiently to this code since $N_1 - K_1 = 16$ is too large.

5.3.2 Closest coset decoding of the $|u|u + v|$ construction

The coset decoding of Section 5.3.2 becomes impractical whenever $K_2$ is large. For instance, optimum coset decoding of the (64,42,8) RM code based on the $|u|u + v|$ construction requires a decomposition into $2^{16}$ cosets. For this code as well as any RM code, more efficient decompositions exist [15], but in general, this may not be the case. In [19], Hemmati proposes a 2-stage decoding of codes constructed from the $|u|u + v|$ construction. The first stage selects one of the $2^{K_2}$ coset candidates and then, the decoding of the repetition code $\tilde{C}_1$ within this coset is realized. This approach was then generalized to other decompositions in [20].

Definitions

We follow the notations introduced in Section 5.3.1. For $\alpha = \pm 1$, the a posteriori probability associated with the symbol $x_{u,i} = (-1)^{x_{u,i}} = \alpha$ corresponding to the code $C_2$ is

$$P(x_{u,i} = \alpha \mid y_{1,i}, y_{2,i}) = \frac{1}{2} \left( \frac{\pi N_0}{2} \right)^{-1/2} \left( e^{-\frac{(y_{1,i}-1)^2+(y_{2,i}-\alpha)^2}{N_0}} + e^{-\frac{(y_{1,i}+1)^2+(y_{2,i}+\alpha)^2}{N_0}} \right),$$

(5.23)
if \( y_{1,i} \) and \( y_{2,i} \) are the two corresponding noisy received symbols. Taking the maximum of the two terms in both the numerator and the denominator of the likelihood ratio formed from Equation 5.23, the log-likelihood ratio \( L(x_{v,i}) \) associated with \( x_{v,i} \) is well approximated by

\[
L(x_{v,i}) \approx \min \left( 2(y_{1,i} + y_{2,i}), -2(y_{1,i} + y_{2,i}) \right) - \min \left( 2(y_{1,i} - y_{2,i}), -2(y_{1,i} - y_{2,i}) \right),
\]

which is equivalent to

\[
L(x_{v,i}) \propto \min(|y_{1,i}|, |y_{2,i}|)
\]

after proper scaling. Based on Equation 5.25, we order the pairs \((|y_{1,i}|, |y_{2,i}|)\) with respect to \(\min(|y_{1,i}|, |y_{2,i}|)\) and then process the generator matrix \(G_2\) of \(C_2\) as in Section 2.2.1 to determine the \(K_2\) MRI positions. The resulting matrix is denoted \(G_v\).

For \(i \in [1, N/2]\), we define \(d_u^i\) and \(d_v^i\) as the decisions associated with the \(i^{th}\) position of the ordering and \(y_{1,i}^i\) and \(y_{2,i}^i\) as the received symbols in the ordered sequence. The bit \(d_u^i\) is allowed any binary value while only the \(K_2\) MRI bits \(d_v^i\), \(i \in [1, K_2]\) can be chosen independently. Based on these choices, the remaining \(d_v^i\)'s, \(i \in [K_2 + 1, N/2]\) are computed from \(G_v\). For \(i \in [1, N/2]\), the contribution of \(d_u^i\) and \(d_v^i\) to the ML cost function is

\[
\Delta_i(\tilde{a}) = (-1)^{d_v^i}y_1^i + (-1)^{d_u^i\oplus d_v^i}y_2^i.
\]

Equation 5.26 shows that if \(d_u^i\) is changed into \(d_u^i \oplus 1\), it is always possible to choose \(d_v^i\) such that \(\Delta_i(\tilde{a})\) is changed into \(\Delta_i(\tilde{a}) \pm \min(|y_{1,i}^i|, |y_{2,i}^i|)\). As in Section 5.3.1, this suggests that the algorithm of Chapter 4 can be adapted to choose the most likely coset in the first stage of closest coset decoding. This algorithm is described in Appendix D. It is then straightforward to remove the contribution of the coset leader and perform the algorithm described in Section 5.3.1 for the second stage decoding.
Performance analysis

We again assume that the all-(-1) sequence is transmitted and define the random variable

$$Z = \min_{i \in [1, a]}(|X_i| | X_i = -1).$$

(5.27)

Then, from Equation C.5, the density function of $Z$ is

$$f_Z(z) = 2(\pi N_0)^{-1/2} \left( \int_{-\infty}^{m_2} e^{-\frac{(z+z)^2}{2N_0}} \, dz + \int_{M_2}^{\infty} e^{-\frac{(z+z)^2}{2N_0}} \, dz \right) e^{-\frac{(z+z)^2}{2N_0}},$$

(5.28)

where $m_2 = \min\{-z, z\}$ and $M_2 = \max\{-z, z\}$. We observe that $f_Z(z)$ is not symmetrical with respect to the origin, but for $X = 1$, we obtain its mirror image with respect to the origin.

After ordering a sequence of length $N/2$ with respect to $|Z|$, the density of the $i^{th}$ value $Z_i$ in the ordering is, from Equation C.5,

$$f_{Z_i}(z_i) = \frac{(N/2)!}{(i-1)!(N/2-i)!} \left( \int_{-\infty}^{m_i} f_Z(z) \, dz + \int_{M_i}^{\infty} f_Z(z) \, dz \right)^{i-1} \left( \int_{m_i}^{M_i} f_Z(z) \, dz \right)^{N/2-i} f_Z(z_i),$$

(5.29)

where $m_i = \min\{-z_i, z_i\}$, $M_i = \max\{-z_i, z_i\}$ and $f_Z(z)$ is defined in Equation 5.28. Equation 5.29 is depicted in Figure 5.8 for $N/2 = 16$ and $N_0 = 0.5$. Comparing this figure with Figure 3.1 shifted by -1, we observe similar bimodal density curves, with the curves of Figure 5.8 closer together and narrower. Smaller noise magnitudes are therefore more probable for each position $i$, which is expected since we consider the worst value of a pair of random variables represented on Figure 3.1. By comparing Figure 5.9 with Figure 3.2, one finds the second mode of the density curve is closer to the origin and has a much larger mass. Therefore, the corresponding $Pe(i; N)$ will be larger for the studied case, as expected.

For equiprobable BPSK transmission of $X = \pm 1$, since $f_Z(z) = f_Z(-z)$, where
Figure 5.8. $f_{x_i}(z_i)$ for $N/2 = 16$, $N_0 = 0.5$, and $z_i \in [-2.5, 1]$. 

108
Figure 5.9. $f_{Z_i}(z_i)$ for $N/2 = 16$, $N_0 = 0.5$, and $z_i \in [0, 1.5]$. 
\( f_{Z'}(z) \) represents the density function of \( Z' = \min_{i \in [1,2]}(|Y_i| X_i = 1) \), the probability that the hard decision of the \( i^{th} \) symbol of the ordered sequence formed by the minimum value of each pair \((|y_{1,i}|, |y_{2,i}|)\) of length \( N/2 \) is in error is given by

\[
\text{Pe}(i; N) = \Pr(Z_i > 0) = \int_{0}^{\infty} f_{Z_i}(z_i) \, dz_i. \tag{5.30}
\]

For larger numbers of positions in error, expressions similar to Equation 5.30 hold (see Appendix C).

Based on the algorithm described in Appendix D, the probability that the coset leader after Step 4, or equivalently order-0 reprocessing, contains more than \( i_1 \) errors in the first \( K_2 \) MRI positions is dominated by \( \tilde{P}_{bi_1} \), computed from Equation 4.43 with the density function of Equation 5.28. If order-\( i_2 \) reprocessing is processed at the second decoding stage, its associated bit error probability \( \tilde{P}_{bi_2} \) is defined as in Equation 5.22. Therefore, the bit error probability associated with the event that a codeword is in error after phase-\( i_1 \) reprocessing at stage 1 and phase-\( i_2 \) reprocessing at stage 2 is upper bounded by,

\[
\tilde{P}_b(i) \leq \tilde{P}_r(\epsilon) + \tilde{P}_{bi_1} + \tilde{P}_{bi_2}, \tag{5.31}
\]

where \( \tilde{P}_r(\epsilon) \) represents the bit error probability associated with the optimum closest coset decoding. In [20], it is shown that \( \tilde{P}_r(\epsilon) \) is upper bounded by

\[
\tilde{P}_r(\epsilon) \leq 2^{N_1-K_1}\Pr(\epsilon), \tag{5.32}
\]

where \( \Pr(\epsilon) \) is the bit error probability associated with the ML decoding. Equation 5.32 is based on the fact that any \( N/2 \)-tuple is allowed to represent \( C_1 \) during the coset selection. A much tighter bound has been derived in [40]. For the \(|u|u + v| \) construction, the exact equivalent weight distribution for CCD can be determined, as presented in Appendix E. This new result is used to compare theoretical and simulated error performance in the next section.
Simulation results

Figures 5.10 and 5.11 depict the simulation results for closest coset decoding of the (32,16,8) and (64,42,8) RM codes. On these figures, the legend "Orders $i + j$" indicates order-$i$ reprocessing at the first stage and order-$j$ reprocessing at the second stage. For the (32,16,8) RM code, we observe that order-2 reprocessing at the first stage and order-1 reprocessing at the second stage achieve the CCD error performance. Also, order-1 reprocessing at the first stage and order-0 reprocessing at the second stage perform within 0.1 dB of the CCD error performance, for an important reduction of computations. Similar conclusions hold for the (64,42,8) RM code, with a loss within 0.2 dB for the second scheme. Therefore, for $P_e = 10^{-6}$, since closest coset decoding suffers respectively a 0.61 dB coding loss for the (32,16,8) RM code and a 0.57 dB coding loss for the (64,42,8) RM code, the scheme with order-1 reprocessing at the first stage and order-0 reprocessing at the second stage offers the best trade off between error performance and computation complexity. This decoding for the (32,16,8) RM code is realized with at most 16 comparisons to determine each minimum, 49 for sorting, 10 operations for order-0 and $5 \cdot 11 = 55$ operations for order-1 reprocessings of the first stage, and $16 + 49 + 4$ operations for order-0 reprocessing of the second stage, so a total of 199 real operations for a 3.8 dB coding gain. Similarly, for the (64,42,8) RM code, we obtain a maximum number of $32 + 129 + 15 + 16 \cdot 16$ operations for order-1 reprocessing at the first stage and $32 + 129 + 5$ operations for order-0 reprocessing at the second stage, so a total of 598 operations for a 4.4 dB coding gain. Also, the average number of operations is much less than these maxima for both schemes when using the resource test described in Chapter 4.
Figure 5.10. Closest Coset Decoding for the (32,16,8) RM code.
Figure 5.11. Closest Coset Decoding for the (64,42,8) RM code.
Chapter 6
Generalization to
the Raleigh Fading Channel

In this chapter, we present how to generalize the statistics obtained in Chapter 3 for the AWGN model to another channel model, the Raleigh fading channel with coherent detection.

6.1 Channel Model

Whenever scattering alter the signal transmission, fading effects have to be included in the channel model. We assume the in-phase and quadrature components of the analog signal $x(t)$ modulate the carriers $\sqrt{2}\cos \omega_0 t$ and $\sqrt{2}\sin \omega_0 t$. If no particular path dominates among all paths, then the received signal $y(t)$ associated with $x(t)$ is [41, 42, p.529]

$$y(t) = a x(t) \sqrt{2} \cos (\omega_0 t - \theta) + w(t),$$

(6.1)

where $w(t)$ represents the usual AWGN, and $\theta$ and $a$ are respectively the uniformly distributed phase and the Raleigh distributed amplitude due to the channel. If coherent detection is possible, the discrete time channel model corresponding to Equation 6.1 for BPSK transmission of $X \in \{\pm 1\}$ is simply

$$Y = AX + W,$$

(6.2)
where \( W \) has a Gaussian distribution and \( A \) is Raleigh distributed. Despite the fact that the phase ambiguity might be difficult to resolve practically, this model has been widely used in the literature [6, 43, 44, 45, 46, 47]. We therefore assume that the phase changes are slow enough to implement a phase-locked loop type receiver [41].

Based on the model described by Equation 6.2, we first show in Appendix F that for BPSK transmission with \( X_i \in \{-1, +1\} \), \( p_{Y_i|X} (y_1 \mid X = s_1) > p_{Y_i|X} (y_2 \mid X = s_2) \) is equivalent to \( |y_1| > |y_2| \), where \( s_i = \text{sgn}(y_i) \) represents the HD decoding of \( y_i \). As in Section 2.2.1, we can therefore associate with each received symbol \( y \) its reliability measure \( |y| \) and order the symbols within each received coded block with respect to this measure.

### 6.2 Statistics after Ordering

To evaluate the error performance of our decoding algorithm, as well as any decoding algorithm based on the ordering with respect to \(|y|\), we need to determine the appropriate statistics after ordering. In Chapter 3, the statistics of the noise after ordering are sufficient to evaluate the error performance of the algorithm. For a particular transmitted symbol \( X \), we have now to consider the joint contribution of the noise value \( W \) and the fading amplitude \( A \). Assuming a sufficient level of interleaving, the three random variables \( X, W \) and \( A \) are independent of each other. Since \( X \) takes equiprobable values in \( \{\pm1\} \), we assume that the all-\((-1)\) BPSK sequence has been transmitted to evaluate the error performance of the considered decoding algorithm.

To this end, we define the new random variable \( Z = (Y|X = -1) = W - A \), which implies \( f_Z(z) = p_{Y|X} (z \mid X = -1) \), where \( f_Z(z) \) represents the density function of \( Z \).
We find in Appendix F that

\[ f_Z(z) = \left( \frac{N_0}{N_0 + 1} \right) \left( \pi N_0 \right)^{-1/2} e^{-\frac{z^2}{2N_0}} - 2 (N_0 + 1)^{-3/2} z e^{-\frac{z^2}{2(N_0 + 1)}} \left( \frac{\sqrt{2z}}{\sqrt{N_0(N_0 + 1)}} \right). \]  

(6.3)

The density function \( f_Z(z) \), as depicted in Figure 6.1 for \( N_0 = 0.5 \), presents a bell-shape with its maximum between 0 and -1. As expected, the mean of \( Z \) is therefore smaller than -1, the mean of the random variable \(-1 + W\) corresponding to the AWGN model.

We define \( Z_i \) as the random variable representing \( Z \) at the \( i^{th} \) position of the ordering. Following the same approach as in Section 3.1, we obtain in Appendix C.1, for BPSK equiprobable signaling,

\[ f_{Z_i}(z_i) = \frac{N!}{(i-1)!(N-i)!} \left( \int_{-\infty}^{m_i} f_Z(z)dz + \int_{M_i}^{\infty} f_Z(z)dz \right)^{i-1} \left( \int_{m_i}^{M_i} f_Z(z)dz \right)^{N-i} f_Z(z_i), \]

(6.4)

where \( m_i = \min\{-z_i, z_i\} \) and \( M_i = \max\{-z_i, z_i\} \).  

(6.5)

Figure 6.2 depicts \( f_{Z_i}(z_i) \) for \( N = 16, N_0 = 0.5 \) and different values of \( i \). After shifting Figure 3.1 by -1 for comparison purpose, we observe the same general shapes for both figures, except for \( i = 16 \) for which \( f_{Z_i}(z_i) \) is narrower and has much more mass at \( z_i = 0 \). When comparing Figure 6.3 with Figure 3.2, we observe that at the same SNR, the mass under the secondary mode is larger in Figure 6.3. This results in a larger \( \text{Pe}(i; N) \) for the Raleigh channel model, as expected.

Based on Section C.3, the probability that the hard decision of the \( i^{th} \) symbol of the sequence \( \tilde{y} \) of length \( N \) is in error is given by

\[ \text{Pe}(i; N) = \Pr(Z_i > 0) = \int_{0}^{\infty} f_{Z_i}(z_i)dz_i. \]  

(6.6)
Figure 6.1. $f_2(z)$ for $N_0 = 0.5$. 
Figure 6.2. $f_{z_i}(z_i)$ for $N = 16$, $N_0 = 0.5$, and $z_i \in [-4, 2]$. 
Figure 6.3. \( f_{z_i}(z_i) \) for \( N = 16, N_0 = 0.5, \) and \( z_i \in [0, 2] \).
Similarly, for $i < j$, we determine in Appendix C.2 the joint distribution of $Z_i$ and $Z_j$ as

$$f_{Z_i,Z_j}(z_i,z_j) = \frac{N!}{(i-1)!(j-i-1)!(N-j)!} \left( \int_{-\infty}^{z_i} f_Z(z)dz + \int_{z_i}^{\infty} f_Z(z)dz \right)^{i-1} \cdot \left( \int_{z_j}^{\infty} f_Z(z)dz + \int_{-\infty}^{z_j} f_Z(z)dz \right)^{j-i-1} \cdot \int_{z_i}^{z_j} f_Z(z)dz \cdot 1_{[m_i,M_i]}(z_j). \quad (6.7)$$

The probability that both hard decisions of the $i^{th}$ and $j^{th}$ symbols of the sequence $\tilde{y}$ of length $N$ are in error is given as follows,

$$\text{Pe}(i,j;N) = \Pr(Z_i > Z_j > 0) = \int_{z_i}^{z_j} f_{Z_i,Z_j}(z_i,z_j)dz_idz_j. \quad (6.8)$$

The probabilities that the hard decisions of three or more ordered symbols of the sequence $\tilde{y}$ of length $N$ are in error are derived in the same way. We finally mention that the statistics derived in Chapter 3 can be obtained from the general method of this section by substituting $\delta(a - 1)$ for the Raleigh density function $p_A(a)$ when defining $Z_i$.

### 6.3 Algorithm Error Performance

Since the same ordering as in Section 2.2.1 is realized, the reprocessing algorithm described in Section 4.1.2 can be directly applied for this channel. $P_b(i)$, the bit error probability associated with the probability that a codeword is in error after phase-$i$ of the reprocessing remains, as in Equation 4.47,

$$P_b(i) \leq \Pr(\epsilon) + P_{bi}. \quad (6.9)$$

$P_{bi}$ is defined as in Section 4.2.2 after substituting Equations 6.6, 6.8 or equivalent forms for $i > 2$ in Equation 4.43. For antipodal binary signaling normalized to 1,
Pr(ε) can be upper bounded, when assuming adequate interleaving, by [44]

\[
Pr(ε) \leq n_d \frac{d_H}{N} \left( \frac{c N_0}{2} \right)^{d_H},
\]

(6.10)

where \( n_d \) is the error coefficient. Equation 6.10 assumes that no side information is available at the decoder. It shows that for BPSK transmission over the Raleigh fading channel with coherent detection, \( d_H \) remains the code parameter to optimize. Also, at high SNR, a good approximation of \( Pr(ε) \) is [6, 44, 46]

\[
Pr(ε) \leq n_d \frac{d_H}{N} N_0^{d_H}. \tag{6.11}
\]

### 6.4 Simulations Results

#### 6.4.1 Error performance

Figures 6.4 and 6.5 depict the error performances of the (24,12,8) extended Golay code and the (128,64,8) extended BCH code. For both codes, the simulated results for various orders of reprocessing are plotted and compared with the theoretical results obtained from Equation 6.9. For the (24,12,8) Golay code, order-2 reprocessing achieves nearly optimum error performance while order-1 reprocessing is near-optimum and order-0 reprocessing also offer a good trade-off between error performance and computation complexity. At \( Pe = 10^{-3} \), order-0 reprocessing achieves a 13.44 dB coding gain over BPSK, while order-1 and order-2 reprocessing respectively provide 16.03 dB and 16.04 dB coding gains. For the (128,64,22) extended BCH code, the coding gains at \( Pe = 10^{-3} \) are 13.23 dB for order-0, 16.25 dB for order-1, 17.62 dB for order-2, 18.53 dB for order-3 and 18.71 dB for order-4 reprocessings. As for the AWGN model, we observe that most of the coding gain is obtained within the first three phases of reprocessing. Therefore, for long codes such as the (128,64,22) extended BCH code, the last tenths of dB of the coding gain are achieved at a very high
Figure 6.4. Error performances for the (24,12,8) extended Golay code.
Figure 6.5. Error performances for the (128,64,22) extended BCH code.
Table 6.1. Order-2 simulation results for (24,12,8) extended Golay code.

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>$P_e$</th>
<th>$c_{ave}$</th>
<th>$c_{max}$</th>
<th>$N_{ave} = 100 + 12c_{ave}$</th>
<th>$N_{max} = 100 + 12c_{max}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>3.01</td>
<td>$10^{-1.2}$</td>
<td>6.13</td>
<td>78</td>
<td>174</td>
<td>1,036</td>
</tr>
<tr>
<td>3.98</td>
<td>$10^{-1.4}$</td>
<td>3.87</td>
<td>78</td>
<td>147</td>
<td>1,036</td>
</tr>
<tr>
<td>5.23</td>
<td>$10^{-1.8}$</td>
<td>1.98</td>
<td>78</td>
<td>124</td>
<td>1,036</td>
</tr>
<tr>
<td>6.99</td>
<td>$10^{-2.6}$</td>
<td>0.52</td>
<td>75</td>
<td>107</td>
<td>1,000</td>
</tr>
<tr>
<td>8.24</td>
<td>$10^{-3.1}$</td>
<td>0.16</td>
<td>40</td>
<td>102</td>
<td>580</td>
</tr>
<tr>
<td>10.0</td>
<td>$10^{-4.1}$</td>
<td>0.025</td>
<td>28</td>
<td>101</td>
<td>436</td>
</tr>
<tr>
<td>11.25</td>
<td>$10^{-4.9}$</td>
<td>0.0055</td>
<td>19</td>
<td>101</td>
<td>328</td>
</tr>
</tbody>
</table>

computational cost which might not be worthwhile, due to the large SNR gap already bridged. For example, at $P_e = 10^{-3}$, the 17.62 dB offered by order-2 reprocessing of the (128,64,22) extended BCH code seems sufficient for a practical implementation, since only 1.09 dB are left to reach the optimum performance, which requires order-4 reprocessing.

6.4.2 Number of computations

The number of computations for 250,000 blocks of the (24,12,8) Golay code are recorded in Table 6.1. We observe similar numbers as for the AWGN model whose corresponding results are given in Table 4.4. The same observation also holds for the (128,64,22) extended BCH code for which the number of computations obtained when simulating 50,000 blocks are given in Tables 6.2 to 6.4 for order-2, order-3 and order-4 reprocessings. Therefore, the algorithm of Chapter 4 is also suitable for practical applications of BPSK transmission on the coherent Raleigh fading channel.
Table 6.2. Order-2 simulation results for (128,64,22) extended BCH code.

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>Pe</th>
<th>$c_{\text{ave}}$</th>
<th>$c_{\text{max}}$</th>
<th>$N_{\text{ave}} = 832 + 64c_{\text{ave}}$</th>
<th>$N_{\text{max}} = 832 + 64c_{\text{max}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.22</td>
<td>$10^{-2.2}$</td>
<td>1,293</td>
<td>2,080</td>
<td>83,584</td>
<td>133,952</td>
</tr>
<tr>
<td>6.99</td>
<td>$10^{-3.4}$</td>
<td>464</td>
<td>2,080</td>
<td>30,528</td>
<td>133,952</td>
</tr>
<tr>
<td>8.24</td>
<td>$10^{-4.6}$</td>
<td>114</td>
<td>2,080</td>
<td>8,128</td>
<td>133,952</td>
</tr>
</tbody>
</table>

Table 6.3. Order-3 simulation results for (128,64,22) extended BCH code.

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>Pe</th>
<th>$c_{\text{ave}}$</th>
<th>$c_{\text{max}}$</th>
<th>$N_{\text{ave}} = 832 + 64c_{\text{ave}}$</th>
<th>$N_{\text{max}} = 832 + 64c_{\text{max}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.22</td>
<td>$10^{-2.7}$</td>
<td>15,850</td>
<td>43,744</td>
<td>1,015,232</td>
<td>2,800,448</td>
</tr>
<tr>
<td>6.02</td>
<td>$10^{-3.7}$</td>
<td>7,422</td>
<td>43,744</td>
<td>475,840</td>
<td>2,800,448</td>
</tr>
<tr>
<td>6.99</td>
<td>$10^{-5.1}$</td>
<td>2,414</td>
<td>43,744</td>
<td>155,328</td>
<td>2,800,448</td>
</tr>
</tbody>
</table>

Table 6.4. Order-4 simulation results for (128,64,22) extended BCH code.

<table>
<thead>
<tr>
<th>SNR (dB)</th>
<th>Pe</th>
<th>$c_{\text{ave}}$</th>
<th>$c_{\text{max}}$</th>
<th>$N_{\text{ave}} = 832 + 64c_{\text{ave}}$</th>
<th>$N_{\text{max}} = 832 + 64c_{\text{max}}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>5.22</td>
<td>$10^{-2.9}$</td>
<td>111,927</td>
<td>679,120</td>
<td>7,164,160</td>
<td>43,464,512</td>
</tr>
<tr>
<td>6.02</td>
<td>$10^{-4.0}$</td>
<td>46,340</td>
<td>679,120</td>
<td>2,966,592</td>
<td>43,464,512</td>
</tr>
<tr>
<td>6.99</td>
<td>$10^{-5.7}$</td>
<td>11,502</td>
<td>677,342</td>
<td>736,960</td>
<td>43,350,720</td>
</tr>
</tbody>
</table>
Chapter 7
The Ordered
Binary Symmetric Channel

7.1 Normal Approximation of the Noise Statistics after Ordering

7.1.1 Approximation of $\text{Pe}(i; N)$

In this chapter, we assume without loss of generality that all considered transmitted symbols are $X_p = -1$, for $p \in [1, N]$. We then remove the conditioning on $X_p$'s to simplify the notations and write for example $f_{W_i|X_i} (w_i | X_i = -1)$ defined in Equation 3.1 as $f_{W_i} (w_i)$. All possible symmetrical cases with some $X_p = 1$ follow in a similar fashion. We define $W_i$ as the restriction of $W_i$ to the interval $[1, \infty)$, so that

$$f_{\tilde{W}_i} (\tilde{w}_i) = \frac{f_{W_i} (\tilde{w}_i)}{\text{Pe}(i; N)}. \quad (7.1)$$

Based on the central limit theorem, we show in Appendix G that for $N$ large enough, the distribution of $\tilde{W}_i$ is well approximated by the distribution of the normal random variable $\eta \left( E[\tilde{W}_i], \sigma_{\tilde{W}_i}^2 \right)$ with mean $E[\tilde{W}_i] = m_i = \alpha^{-1} (1 - i/N)$ and variance $\sigma_{\tilde{W}_i}^2 = \pi N_0 \frac{i(N - i)/N^3 \left(e^{-(m_i - 2)^2/N_0} + e^{-m_i^2/N_0}\right) - 2}{N_0}$. We notice that for fixed $i/N$ and $N_0$, $E[\tilde{W}_i]$ remains the same while $\sigma_{\tilde{W}_i}^2$ decreases in $1/N$. Therefore, for $N$ large enough, many positions $i$ have most of the area under their density contained in the interval
This remark will be used in the next section.

Using the fact that
\[ f_{\tilde{W}_i}(\tilde{w}_i) |_{\tilde{w}_i = E[\tilde{W}_i]} = \frac{1}{\sqrt{2\pi} \sigma_{\tilde{W}_i}}, \] (7.2)
we obtain, from Equation 7.1,
\[ \Pe(i; N) \equiv \sqrt{2\pi} \sigma_{\tilde{W}_i} \cdot f_{\tilde{W}_i}(\tilde{w}_i) |_{\tilde{w}_i = E[\tilde{W}_i]} \]
\[ \equiv \left( \frac{N - 1}{i - 1} \right) \sqrt{\frac{i(N - i)}{N}} \left( \frac{i}{N} \right)^{i-1} \left( \frac{N - i}{N} \right)^{N-i} \frac{\sqrt{2\pi} e^{-\frac{m^2}{2\sigma^2}}}{e^{-\frac{(m-m_0)^2}{2\sigma^2}} + e^{-\frac{m^2}{\sigma^2}}} \] (7.3)
As discussed at the end of Appendix G, this approximation becomes less accurate as \(i\) approaches the extreme values, and especially \(N\). Also, for \(N_0\) of practical interest and positions \(i\) not to close to \(N\), Equation 7.3 is further approximated by
\[ \Pe(i; N) \equiv \sqrt{2\pi} \left( \frac{N - 1}{i - 1} \right) \sqrt{\frac{i(N - i)}{N}} \left( \frac{i}{N} \right)^{i-1} \left( \frac{N - i}{N} \right)^{N-i} e^{\frac{4(1-m_0)}{N_0}}. \] (7.4)
For large values of \(N\), applying Stirling’s formula \(n! \approx \sqrt{2\pi n} (n/e)^n\) to Equation 7.4 finally provides
\[ \Pe(i; N) \equiv e^{\frac{4(1-m_0)}{N_0}}. \] (7.5)
When \(N\) is large enough, both Equations 7.4 and 7.5 provide extremely tight bounds, as depicted in Figure 7.1 for \(N = 128\) and \(i = 64\).

### 7.1.2 Approximation of \(\Pe(i, j; N)\)

From Equations 3.1 and 3.3, we compute the conditional density of \(W_j | W_i = w_i\) as
\[ f_{W_i|W_j}(w_i | W_j = w_j) = \frac{(\pi N_0)^{-1/2}(N - i)!}{(j - i - 1)! (N - j)!} \cdot \frac{e^{-u^2/N_0}}{\alpha(w_i)} \cdot \left( 1 - \frac{\alpha(w_j)}{\alpha(w_i)} \right)^{j-i-1} \left( \frac{\alpha(w_j)}{\alpha(w_i)} \right)^{N-j} \cdot 1_{[m(w_i), M(w_i)]}(w_j). \] (7.6)
Equation 7.6 shows that \(W_j | W_i = w_i\) has the density function of the \((j - i)^{th}\) noise value after ordering a sample of size \(N - i\) from a population with distribution trun-
Figure 7.1. Comparison between exact and approximated values of $P_e(64;128)$. 

+ : Exact $P_e(64;128)$: Eq. (3.2)

x: Approximated $P_e(64;128)$: Eq. (7.4)

o: Approximated $P_e(64;128)$: Eq. (7.5)
cated to the interval \([m(w_i), M(w_i)]\). This result holds for ordered statistics in general [28]. However, it was not directly applicable to our case since the noise \(W\) is not the ordered random variable. Also, a symmetrical expression to Equation 7.6 is obtained when considering \(W_i | W_j = w_j\).

As in Section 7.1.1, we define \(\tilde{W}_{iji}\) as the restriction of \(W_j | W_i = w_i\) to the interval \([1, w_i)\), for \(w_i \geq 1\). Therefore,

\[
f_{\tilde{W}_{iji}}(\tilde{w}_j | W_i = w_i) = \frac{f_{W_j | W_i}(\tilde{w}_j | W_i = w_i)}{I(w_i, j; N)},
\]

where

\[
I(w_i, j; N) = \int_1^{w_i} f_{W_j | W_i}(w_j | W_i = w_i)dw_j.
\]

Defining \(\beta(t, w_i) = \alpha(t)/\alpha(w_i)\), Appendix G implies, as in Section 7.1.1 that for \(N\) large enough, the distribution of \(\tilde{W}_{iji}\) tends towards the distribution of a normal random variable with mean \(E[\tilde{W}_{iji}] = \beta^{-1} (1 - (j - i)/(N - i), w_i)\) and variance \(\sigma_{\tilde{W}_{iji}}^2 = \pi N_0 (j - i)(N - j)/(N - i)^3 \cdot (e^{-(m_{ji} - 2)^2/N_0} + e^{-m_{ji}^2/N_0})^{-2}\).

For \(w_i = E[\tilde{W}_i]\), \(\beta^{-1} (1 - (j - i)/(N - i), w_i) = \alpha^{-1} (1 - j/N) = E[\tilde{W}_j] = m_j\), as expected from the general properties of normal random variables. Also, for \(w_i = E[\tilde{W}_i]\), we compute

\[
\frac{\sigma_{\tilde{W}_{iji}}^2}{\sigma_{\tilde{W}_j}^2} = \frac{j - i}{j} \frac{N^3}{(N - i)^3}.
\]

For \(j >> i\), Equation 7.9 shows that \(\sigma_{\tilde{W}_{iji}}^2 \approx \sigma_{\tilde{W}_j}^2\), which implies that \(\tilde{W}_{iji}\) tends towards \(\tilde{W}_j\) in distribution. This is no longer the case when \(j\) and \(i\) are close. In fact, for \(N\) large enough, knowing \(\tilde{w}_i = E[\tilde{W}_i]\) almost surely fixes \(\tilde{w}_{i+1} = E[\tilde{W}_i] - \epsilon\), so that \(\tilde{W}_{iji}\) tends towards the density \(\delta(\tilde{w}_j - E[\tilde{W}_i])\). Note finally that the asymptotic convergence of \(\tilde{W}_j\) and \(\tilde{W}_{iji}\) holds only for \(\tilde{w}_i = E[\tilde{W}_i]\), so that \(\tilde{W}_j\) and \(\tilde{W}_i\) are not independent.

Considering the normal approximation for the density of \(\tilde{W}_{iji}\), we derive as for
Equation 7.4,

\[ I(E[\tilde{W}_i], j; N) = \frac{N}{N-i} \binom{N-i-1}{j-i-1} \sqrt{\frac{(j-i)(N-j)}{N-i}} \left( \frac{j-i}{N-i} \right)^{j-i-1} \left( \frac{N-j}{N-i} \right)^{N-j} \] 

while Stirling's formula provides

\[ I(E[\tilde{W}_i], j; N) = \frac{N}{N-i} \cdot e^{\frac{i(1-m_i)}{\kappa_0}}. \] (7.10)

\( I(E[\tilde{W}_i], j; N) \) represents the probability that hard decision of the \( j^{th} \) ordered symbol of the sequence \( \tilde{y} \) of length \( N \) is in error, when knowing the \( i^{th} \) noise value \( w_i = E[\tilde{W}_i] \).

Combining Equations 7.5 and 7.11, we finally obtain

\[ I(E[\tilde{W}_i], j; N) = \left( \frac{N}{N-i} \right) \text{Pe}(j; N). \] (7.12)

Equation 7.12 expresses that knowing \( w_i = E[\tilde{W}_i] \) provides almost no information about \( \text{Pe}(j; N) \), especially for \( i \ll N \).

From Equation 7.8, we verify that

\[ \lim_{w_i \to 1} I(w_i, j; N) = 1/2, \]

\[ \lim_{w_i \to \infty} I(w_i, j; N) = \text{Pe}(j - i; N - i). \] (7.13)

After integrating Equation 7.8 by parts, we check that \( \partial I(w_i, j; N)/\partial w_i \leq 0 \). \( I(w_i, j; N) \) is therefore non increasing between the values 1/2 and \( \text{Pe}(j - i; N - i) \). Finally, for \( N_0 \) of practical interest, we verify that \( \sigma_{\tilde{W}_i} \) is small enough to contain most of the area under \( f_{\tilde{W}_i}(\tilde{w}_i) \) (or equivalently \( f_{W_i}(w_i) \)) for \( \tilde{w}_i \in [E[\tilde{W}_i] - \sigma_{\tilde{W}_i}, E[\tilde{W}_i] + \sigma_{\tilde{W}_i}] \) and for approximating \( I(w_i, j; N) \) by a straight line for \( \tilde{w}_i \) in the same interval. Within these assumptions, Equation 3.4 becomes

\[ \text{Pe}(i, j; N) = \int_{\tilde{w}_i}^{\infty} f_{W_i}(w_i) I(w_i, j; N) dw_i \]

\[ = \int_{E[\tilde{W}_i] - \sigma_{\tilde{W}_i}}^{E[\tilde{W}_i] + \sigma_{\tilde{W}_i}} f_{W_i}(w_i) I(w_i, j; N) dw_i, \] (7.14)
which implies,

\[ Pe(i, j; N) \equiv \left( \frac{N}{N - i} \right) Pe(i; N) Pe(j; N). \]  

(7.15)

For \( N \gg i \), Equation 7.15 is equivalent to \( Pe(i, j; N) \equiv Pe(i; N) Pe(j; N) \). Therefore, despite the fact that \( \hat{W}_i \) and \( \hat{W}_j \) are dependent, their associated error probabilities tends to behave as if they were independent, for \( N \) large enough.

### 7.1.3 Approximation of \( Pe(n_1, n_2, \cdots, n_j; N) \)

We now generalize Equation 7.15 to any ordered set of indices \( I_j = \{n_1, n_2, \cdots, n_j\} \) corresponding to positions in error after ordering. Using the definitions of the joint densities derived in Section 3.1, we compute

\[ \frac{e^{-\frac{w_{n_j}}{N_0}}}{\alpha(w_{n_{j-1}})} \left( 1 - \frac{\alpha(w_{n_j})}{\alpha(w_{n_{j-1}})} \right)^{n_j - n_{j-1} - 1} \left( \frac{\alpha(w_{n_j})}{\alpha(w_{n_{j-1}})} \right)^{N-n_j} \cdot 1_{[m(w_{n_{j-1}}), M(w_{n_{j-1}})]}(w_{n_j}). \]  

(7.16)

Equation 7.16 expresses that \( W_{n_j} \mid W_{n_1}, \cdots, W_{n_{j-1}} \) has the density function of the \((n_j - n_{j-1})^{th}\) noise value after ordering a sample of size \( N - n_{j-1} \) from a population with distribution truncated to the interval \([m(w_{n_{j-1}}), M(w_{n_{j-1}})]\) and therefore, is equivalent to \( W_{n_j} \mid W_{n_{j-1}} \). As in the case of ordered statistics, we obtain a Markov chain [28]. When applying a chain argument based on Section 7.1.2, we finally obtain

\[ Pe(n_1, n_2, \cdots, n_j; N) \equiv \prod_{i=1}^{j-1} \left( \frac{N}{N - n_i} \right) Pe(n_i; N) \cdot Pe(n_j; N). \]  

(7.17)

In addition to its theoretical importance already discussed at the end of Section 7.1.2, Equation 7.17 is also useful when evaluating \( Pe(n_1, n_2, \cdots, n_j; N) \) numerically. An \( n_j^{th}\)-order integral is replaced by the product of \( n_j \) single ones.

Figure 7.2 depicts the validity of Equation 7.17 for \( N = 128 \), and \( n_1 = 62 \), \( n_2 = 63 \) and \( n_3 = 64 \). We observe that the matching between the two curves is
excellent. Similar results are obtained when considering any ordered set of indeces \( \{n_1, n_2, \ldots, n_j\} \) where \( n_j \) does not take values around \( N \).

7.2 First Order Approximation of the Ordered Binary Symmetric Channel

7.2.1 Definitions

For a subset of positions \( \{n_1, n_2, \ldots, n_j\} \), \( \text{Pe}(n_1, n_2, \ldots, n_j; N) \) as defined in Equation 4.1 represents the probability that at least the bits in position \( n_1, n_2, \ldots, n_j \) are in error after ordering a sequence of length \( N \). We now also define \( \text{Pe}_N(n_1, n_2, \ldots, n_j) \) as the probability that only the bits at position \( n_1, n_2, \ldots, n_j \) are in error after ordering a sequence of length \( N \). Note that while \( \text{Pe}(n_1, n_2, \ldots, n_j; N) \) is computed by integrating the joint distribution of the \( n_j \) ordered random variables \( W_{n_1}, W_{n_2}, \ldots, W_{n_j} \), the computation of \( \text{Pe}_N(n_1, n_2, \ldots, n_j) \) requires one to integrate the joint distribution of the \( N \) ordered random variables \( W_1, W_2, \ldots, W_N \).

7.2.2 Case \( N = 2 \)

The case \( N = 2 \) constitutes the only non-trivial case for which all error probabilities have a closed form, as resumed in Figure 7.3. The exact Ordered Binary Symmetric Channel (OBSC) corresponding to this ordering contains 4 fully connected states, as depicted in Figure 7.4, with \( \epsilon_1 = \text{Pe}_2(1) \), \( \epsilon_2 = \text{Pe}_2(2) \) and \( \epsilon_3 = \text{Pe}_2(1,2) = \text{Pe}(1,2;2) \). Noticing that \( \text{Pe}_2(1) = \text{Pe}(1;2) - \text{Pe}(1,2;2) \), \( \text{Pe}_2(2) = \text{Pe}(2;2) - \text{Pe}(1,2;2) \) and assuming that Equation 7.15 holds while disregarding the factor 2, we can approximate the 4-state OBSC of Figure 7.4 by the 2 time-shared 2-state BSC's corresponding to each ordered position, with crossover probabilities \( \epsilon(i) = \text{Pe}(i;2) \) for bit \( i, i \in \{1, 2\} \).
Figure 7.2. Comparison between $\text{Pe}(62, 63, 64; 128)$ and $(128/66) \text{Pe}(62; 128) \cdot (128/65) \text{Pe}(63; 128) \cdot \text{Pe}(64; 128)$. 
Defining $\epsilon_0 = 1 - \epsilon_1 - \epsilon_2 - \epsilon_3$, the capacity of the OBSC for $N = 2$ is [48]

$$C_2 = 2 + \sum_{i=0}^{3} \epsilon_i \log_2 \epsilon_i \text{ bits,} \quad (7.18)$$

so that the average capacity per channel use becomes

$$C_{2,ave} = 1 + \frac{1}{2} \sum_{i=0}^{3} \epsilon_i \log_2 \epsilon_i \text{ bit.} \quad (7.19)$$

Defining the entropy function

$$h(p) = -p \log_2 p - (1 - p) \log_2 (1 - p), \quad (7.20)$$

the average capacity of the approximate model of 2 BSC's is given by [48]

$$\tilde{C}_{2,ave} = 1 - \frac{1}{2} \sum_{i=1}^{2} h(Pe(i; 2)) \text{ bit.} \quad (7.21)$$

As expected, we observe in Figure 7.5 that $C_{2,ave} \approx \tilde{C}_{2,ave}$, which validates our approximation of the channel. We name this approximation the first order approximation of the OBSC.

### 7.2.3 General Case

We now generalize the results of Section 7.2.2 to any ordered sequence of length $N$. Based on Equation 7.17, we approximate

$$\text{Pe}(n_1, n_2, \ldots, n_j; N) \equiv \prod_{i=1}^{j} \text{Pe}(n_i; N). \quad (7.22)$$

Equation 7.22 expresses that after ordering, the events of having errors at positions $n_1, n_2, \ldots, n_j$ remain independent. Therefore, the $2^N$-state fully connected OBSC is equivalent to $N$ time-shared BSC's corresponding to each ordered position.

As $N$ increases, the capacity of the OBSC,

$$C_{N,ave} = 1 + \frac{1}{N} \sum_{i=0}^{2^N-1} \epsilon_i \log_2 \epsilon_i \text{ bit,} \quad (7.23)$$
Pe (1;2) = \tilde{Q} (\sqrt{2})

Pe (2;2) = 2 \tilde{Q}(1) - \tilde{Q} (\sqrt{2})

Pe_2 (1) = \tilde{Q} (\sqrt{2}) - \tilde{Q} (1)^2

Pe_2 (2) = 2 \tilde{Q}(1) - \tilde{Q} (\sqrt{2}) - \tilde{Q}(1)^2

Pe (1,2;2) = Q(1)^2

Figure 7.3. Different error probabilities associated with the ordering of 2 bits.
Figure 7.4. Ordered Binary Symmetric Channel for $N = 2$. 
Figure 7.5. Comparison of $C_{2,\text{ave}}$ (a) and $\tilde{C}_{2,\text{ave}}$ (b).
rapidly becomes too complex to evaluate since each of the $2^N \epsilon_i$'s corresponds to an $N$-order integral. In contrast, the capacity of first order approximation of the OBSC,

$$\hat{C}_{N,\text{ave}} = 1 - \frac{1}{N} \sum_{i=1}^{N} h(\text{Pe}(i; N)) \text{ bit}, \quad (7.24)$$

is easily derived. Figure 7.6 plots $\hat{C}_{N,\text{ave}}$ for $N = 2^i$, with $i \in \{0, 7\}$. For $N = 1$, $\hat{C}_{1,\text{ave}}$ is simply the capacity of the BSC with crossover probability $\hat{Q}(1)$, while $\lim_{N \to \infty} \hat{C}_{N,\text{ave}}$ should provide the capacity of the continuous Gaussian channel for BPSK transmission, given by [49]

$$C_{\text{bpsk}} = (\pi N_0)^{1/2} \int_{-\infty}^{\infty} e^{-(y-1)^2/N_0} \log_2 \left( \frac{2 e^{-(y-1)^2/N_0}}{e^{-(y-1)^2/N_0} + e^{-(y+1)^2/N_0}} \right) dy. \quad (7.25)$$

Figure 7.6 shows that the convergence to this limit is very fast as $N$ increases, so that

$$\hat{C}_{N,\text{ave}} \approx C_{\text{bpsk}}, \quad (7.26)$$

for $N$ large enough. Equation 7.26 indicates that when considering a sufficiently long ordered sequence, the first order approximation of the OBSC should provide a good approximation to the continuous Gaussian channel, for BPSK transmission. Therefore, for a given SNR, knowing the position in the ordering instead of the exact received value should be sufficient from a performance point of view. One of the algorithms for Majority-Logic-Decoding of RM codes presented in the next chapter illustrates this fact.
Figure 7.6. Capacities of the first order approximation of the OBSC for $N = 2^i$, with $i \in \{0, 7\}$.
Chapter 8
Majority-Logic-Decoding
of Reed-Muller Codes

8.1 A Review of Majority-Logic Decoding of RM Codes over the Binary Symmetric Channel

Consider binary RM codes in their original form described in [25]. For any non-negative integers \( r \) and \( m \) with \( r < m \), there exists a binary \( r \)-th order RM code, denoted \( \text{RM}(r,m) \), of length \( N = 2^m \), minimum Hamming distance \( d_H = 2^{m-r} \) and dimension \( K = \sum_{i=0}^{r} \binom{m}{i} \). Suppose this RM\((r,m)\) code is used for error control over the AWGN channel. For BPSK transmission, a codeword \( \tilde{c} = (c_1, c_2, \ldots, c_N) \) is mapped into a bipolar sequence \( \tilde{x} = (x_1, x_2, \ldots, x_N) \) with \( x_i = (-1)^{c_i} \in \{ \pm 1 \} \). After transmission, the received sequence at the output of the correlation detector is \( \tilde{y} = (y_1, y_2, \ldots, y_N) \) with \( y_i = x_i + w_i \), where the \( w_i \) are independent Gaussian random variables with zero mean and variance \( N_0/2 \).

For hard-decision decoding, \( \tilde{y} \) is converted into a binary sequence \( \bar{y} = (r_1, r_2, \ldots, r_N) \) with \( r_i = 0 \) for \( y_i > 0 \) and \( r_i = 1 \) otherwise. For majority-logic decoding, a set \( S_j \) of check-sums for each information bit \( a_j \), \( 1 \leq j \leq K \), is formed. Each check-sum is a modulo-2 sum of the hard-decision decoded bits \( r_i \) of certain received symbols \( y_i \). For RM codes, the sets of bits of any two distinct check-sums are disjoint. The
check-sums in \( S_j \) form a set of \textit{independent estimates} of the information bit \( a_j \).

In the following, we briefly review the main steps of majority-logic decoding of RM codes, closely following the derivations of [50]. For equiprobable signaling, the information bit \( a_j \) is decoded into its estimate \( \hat{a}_j \in [0,1] \) which maximizes the conditional probability \( \text{Prob}(S_j \mid a_j = s) \). Let \( S_j = \{A_{j,l} : 1 \leq l \leq |S_j|\} \), where \( A_{j,l} \) denotes the \( l^{th} \) check-sum in \( S_j \) and \( |S_j| \) denotes the cardinality of \( S_j \). Since the check sums in \( S_j \) are disjoint, we obtain

\[
\ln \left( \text{Prob}(S_j \mid a_j = s) \right) = \sum_{l=1}^{|S_j|} \ln \left( \text{Prob}(A_{j,l} \mid a_j = s) \right) .
\]

\( (8.1) \)

Let \( p_l \) be the probability that the number of independent errors in the check sum \( A_{j,l} \) is odd. Then it is shown in [50] that

\[
p_l = \text{Prob}(A_{j,l} = 0 \mid a_j = 1) = \text{Prob}(A_{j,l} = 1 \mid a_j = 0)
\]

\[
= \frac{1}{2} \left[ 1 - \prod_{i \in N_j(l)} (1 - 2 \gamma_i) \right],
\]

\( (8.2) \)

where \( N_j(l) \) represents the set of positions associated with the digits constituting the check-sum \( A_{j,l} \) and \( \gamma_i \) is the probability that the \( i^{th} \) digit is in error. It is clear that \( p_l \) is simply the probability that the check-sum \( A_{j,l} \) mismatches the transmitted information bit \( a_j \).

For the BSC with crossover probability \( \gamma_i = \tilde{Q}(1) \), \( p_l \) is the same for all \( l \) and independent of \( j \). It follows from Equation 8.1 that [50]

\[
\sum_{i=1}^{|S_j|} \ln \left( \frac{\text{Pr}(A_{j,l} \mid a_j = 1)}{\text{Pr}(A_{j,l} \mid a_j = 0)} \right) = \sum_{i=1}^{|S_j|} (2A_{j,l} - 1) \ln \left( \frac{1 - p_l}{p_l} \right).
\]

\( (8.3) \)

Since \( \ln((1 - p_l)/p_l) \) is a positive constant, the decision rule is then

\[
\sum_{i=1}^{|S_j|} (2A_{j,l} - 1) \leq 0,
\]

\( (8.4) \)

which simply compares \( \sum_{i=1}^{|S_j|} A_{j,l} \) with \( |S_j|/2 \). If \( \sum_{i=1}^{|S_j|} A_{j,l} > |S_j|/2 \), set the estimate \( \hat{a}_j \) of \( a_j \) to 1, and if \( \sum_{i=1}^{|S_j|} A_{j,l} < |S_j|/2 \), set \( \hat{a}_j = 0 \). Whenever \( \sum_{i=1}^{|S_j|} A_{j,l} = |S_j|/2 \), we
flip a fair coin to determine \( \hat{a}_j \), which results in an erroneous decision in half of the cases on average. This kind of error dominates the error performance. The above decoding is the conventional majority-logic decoding. Many majority-logic decodable codes besides RM codes have been constructed based on this hard-decision decoding scheme.

8.2 Majority-Logic Decoding Based on the Reliability Information of the Received Symbols

In this section, we improve the above conventional majority-logic decoding by using the reliability information of the received symbols. Three improvements are presented.

8.2.1 BSC model based on received symbol ordering

Consider the received sequence \( \bar{y} = (y_1, y_2, \cdots, y_N) \) at the output of the correlation detector of the receiver. As described in Chapter 2, we order the received symbols \( y_j \) in decreasing order with respect to their reliability measures \( |y_j| \). Based on Section 7.2, to each ordered position \( i_o \in [1, 2^m] \), we associate the BSC channel with crossover probability \( \gamma_{i_o} = \text{Pe}(i_o; 2^m) \) as defined in Equation 3.2. This crossover probability \( \gamma_{i_o} \) represents the probability that the \( i_o \)th ordered digit is in error after hard decision of the received sequence \( \bar{y} \).

From Equation 8.2, we notice that \( \binom{2^m}{|\tilde{N}_j(l)|} \) possible values of \( p_l \) need to be preprocessed for the check sum \( S_j \); this becomes rapidly impractical. Defining \( \tilde{N}_j(l) \) as the set of ordered positions associated with the digits constituting the parity check sum \( A_{j,l} \), we propose to approximate \( \prod_{i_o \in \tilde{N}_j(l)} (1 - 2 \gamma_{i_o}) \approx (1 - 2 \max_{i_o \in \tilde{N}_j(l)} \{\gamma_{i_o}\}) \), so that \( p_l \approx \max_{i_o \in \tilde{N}_j(l)} \{\gamma_{i_o}\} \). This approximation is equivalent to considering only the least reliable position in each check sum. Only the \( 2^m \) \( k_l = \ln((1 - p_l)/p_l) \)'s need
now to be stored and the decision rule becomes
\[ \sum_{i=1}^{|S_j|} (2A_{j,i} - 1) k_i \leq 0. \] (8.5)

The decoding based on Equation 8.5 is: if \( \sum_{i=1}^{|S_j|} (2A_{j,i} - 1) k_i > 0 \), set the estimate \( \hat{a}_j \) of \( a_j \) to 1, else set \( \hat{a}_j \) to 0.

### 8.2.2 Gaussian model

While the two models considered so far are time invariant, we now present a time variant model based on the Gaussian statistics, as first introduced in [50]. We obtain
\[ \gamma_i = \frac{e^{-|L_i|}}{1 + e^{-|L_i|}}, \] (8.6)

where \( L_i = 4y_i/N_0 \) represents the log-likelihood ratio associated with the received symbol \( y_i \). As in Section 8.2.1, the optimum decision rule based on Equation 8.6 is computationally expensive. Using the same approximation, we obtain
\[ p_t \approx \frac{e^{-4|y|_{\text{min}}/N_0}}{1 + e^{-4|y|_{\text{min}}/N_0}}, \] (8.7)

where \( |y|_{\text{min}} = \min_{i\in N_j} \{|y_i|\} \), which provides the Uniformly Most Powerful (UMP) test
\[ \sum_{i=1}^{|S_j|} (2A_{j,i} - 1) |y_{i}\text{min} \leq 0. \] (8.8)

As opposed to the time invariant BSC model, ordering does not provide further improvement for the Gaussian model, since the log-likelihood ratio \( |L_j| \) remains identical with or without ordering, as shown in Lemma 1 of Section 3.3.2.

In both Equations 8.5 and 8.8, each term of Equation 8.4 is weighted with respect to the least reliable digit of its corresponding check sum. Comparing Equations 8.5 and 8.8, we notice that Equation 8.5 depends only on the position after ordering, but not on the exact received value. On the other hand, it also depends on the noise.
power spectral density $N_0$ and requires a complete ordering of the received symbols, while Equation 8.8 is a UMP test and needs only to evaluate the minimum received value within each check sum.

### 8.2.3 Conventional decoding with UMP test for ties breaking

As mentioned in Section 8.1, the error performance of the conventional majority-logic decoding is dominated by the event $\sum_{i=1}^{S_j} A_{ij} = |S_j|/2$. In such cases, we propose to break the tie using the UMP test of Section 8.2.2 instead of flipping a fair coin. Such a test requires very few additional computations.

### 8.3 Computation Complexity

The different computational costs of each scheme are given in Table 8.1. We respectively name Scheme 1 the conventional decoding of Section 8.1, Scheme 2 the scheme described in Section 8.2.1 and Scheme 3 the scheme described in Section 8.2.2. The different costs are based on the facts that majority-logic-decoding of RM($r, m$) is achieved in $r + 1$ stages [26]. Each stage $l \in [0, r]$ decodes $\binom{r}{l}$ symbols, each symbol corresponds to $2^{m-l}$ check sums and each check sum itself sums $2^l$ bits. We assume that ordering $N$ symbols requires about $N \log_2 N$ comparisons while $2^l - 1$ comparisons are needed to determine $|y_i|_{\min}$ in each check sum. Note however that both processes can be speeded up considerably using a parallel implementation of respectively simple sorting algorithms, such as “Merge-sort” and a binary tree search [24]. Also, in the simplified algorithm of Section 8.2.1, $\max_{i \in N_{j} \cup \{j\}} \{\gamma_i\}$ can be simultaneously determined for each check sum by reading backwards the set of indices corresponding to the ordered sequence. We ignore this cost, assuming the proper data structures are used to efficiently access the permutations between the initial positions and the
Table 8.1. Computations required for the majority logic decoding of RM(r, m).

<table>
<thead>
<tr>
<th>Operation</th>
<th>Scheme 1</th>
<th>Scheme 2</th>
<th>Scheme 3</th>
</tr>
</thead>
<tbody>
<tr>
<td>H D</td>
<td>$2^m$</td>
<td>$2^m$</td>
<td>$2^m$</td>
</tr>
<tr>
<td>Check sum</td>
<td>$\sum_{l((2^l-1)[2^m-l]}\binom{m}{i}$</td>
<td>$\sum_{l((2^l-1)[2^m-l]}\binom{m}{i}$</td>
<td>$\sum_{l((2^l-1)[2^m-l]}\binom{m}{i}$</td>
</tr>
<tr>
<td>Ordering</td>
<td>0</td>
<td>$m \cdot 2^m$</td>
<td>$\sum_{l((2^l-1)[2^m-l]}\binom{m}{i}$</td>
</tr>
<tr>
<td>Test</td>
<td>$\sum_{l\in\mathbb{N}}(2^{m-l}-1)\binom{m}{l}$</td>
<td>$\sum_{l\in\mathbb{N}}(2^{m-l}-1)\binom{m}{l}$</td>
<td>$\sum_{l\in\mathbb{N}}(2^{m-l}-1)\binom{m}{l}$</td>
</tr>
</tbody>
</table>

ordered positions, as well as the table look-up between initial positions and check sums.

In Table 8.1, an exponent in brackets represents the number of parallel realizations of the expression considered. While the conventional decoding of Section 8.1 requires only elementary binary operations, the ordering and testing operations for Sections 8.2.1 and 8.2.2 both require real number computations. We finally point out that more efficient decoding schemes can be constructed using hybrid versions of the three presented schemes. For example, using the UMP test of Section 8.2.2 for the first decoding stage and the conventional majority-logic decoding for the remaining stages would save many computations while resulting in very little performance degradation since the number of check sums doubles at each stage.

8.4 Simulation Results

The majority-logic decoding schemes described in the previous sections are simulated for the (32,16,8), (32,26,4), (64,22,16) and (64,42,8) RM codes. Their error performance is shown in Figures 8.1 to 8.4. We see that all the proposed schemes achieve significant coding gains (at least 1 dB) over the conventional majority-logic decoding scheme.
We first notice that the approximation introduced in Sections 8.2.1 and 8.2.2 does not significantly alter the corresponding optimum error performance, for any practical SNR. Therefore, both schemes provide low computational near-optimum decoding which particularly fits applications requiring high speed decoding. As the SNR increases, the time varying scheme of Section 8.2.2 slightly outperforms the scheme of Section 8.2.1 for high rate codes. This is mostly due to the fact that the proposed BSC model after ordering becomes less accurate for the last 3 to 5 ordered symbols, as discussed in Section 7.1. Therefore, high rate codes are more affected since at the first stage, each symbol is decoded with few corresponding check sums. Also, for these last positions, the accuracy of the model increases as the code length $N$ increases. We finally notice that using the UMP test to break the ties in the conventional majority-logic decoding provides significant coding gains, especially for high rate codes.
Figure 8.1. Error performances for RM(32, 16, 8) with majority-logic-decoding.
(a) Maximum-likelihood-decoding (Union bound); (b) Majority-logic-decoding, scheme 3 (optimal); (c) Majority-logic-decoding, scheme 3 (ump test); (d) Majority-logic-decoding, scheme 2 (optimal and approximation); (e) Majority-logic-decoding, scheme 1 with ties decided by scheme 3 (ump test); (f) Majority-logic-decoding, scheme 1; (g) BPSK.
Figure 8.2. Error performances for RM(32, 26, 4) with majority-logic-decoding.
(a) Maximum-likelihood-decoding (Union bound); (b) Majority-logic-decoding, scheme 3 (optimal); (c) Majority-logic-decoding, scheme 3 (ump test); (d) Majority-logic-decoding, scheme 2 (optimal and approximation); (e) Majority-logic-decoding, scheme 1 with ties decided by scheme 3 (ump test); (f) Majority-logic-decoding, scheme 1; (g) BPSK.
Figure 8.3. Error performances for RM(64, 22, 16) with majority-logic-decoding.
(a) Maximum-likelihood-decoding (Union bound); (b) Majority-logic-decoding, scheme 3 (optimal); (c) Majority-logic-decoding, scheme 3 (ump test); (d) Majority-logic-decoding, scheme 2 (optimal and approximation); (e) Majority-logic-decoding, scheme 1 with ties decided by scheme 3 (ump test); (f) Majority-logic-decoding, scheme 1; (g) BPSK.
Figure 8.4. Error performances for RM(64, 42, 8) with majority-logic-decoding.
(a) Maximum-likelihood-decoding (Union bound); (b) Majority-logic-decoding, scheme 3 (optimal); (c) Majority-logic-decoding, scheme 3 (ump test); (d) Majority-logic-decoding, scheme 2 (optimal and approximation); (e) Majority-logic-decoding, scheme 1 with ties decided by scheme 3 (ump test); (f) Majority-logic-decoding, scheme 1; (g) BPSK.
ML trellis decoding of linear block codes requires the same number of computations for each coded block. On the other hand, dynamic ML decoding algorithms depend on two parameters, the maximum number of processed codewords per coded block $c_{\text{max}}$ as for trellis decoding, and the average number of processed codewords per coded block $c_{\text{ave}}$. While $c_{\text{ave}}$ provides information about the decoding speed of the algorithm, $c_{\text{max}}$ remains a limiting factor since the decoder must be designed for this value. The recent decoding algorithms presented in [8, 22] provide efficient ways to reduce the search space and thus, $c_{\text{ave}}$. However, $c_{\text{max}}$ is ignored and, at the beginning of the algorithm, in fact $c_{\text{max}} = 2^K$ is allowed. Then, as the algorithm progresses, the search space is progressively reduced up to convergence. If a small $c_{\text{ave}}$ guarantees a fast convergence in most of the cases, no indication about the worst case is provided.

In this dissertation, based on the statistics after ordering derived in Chapter 3, we proposed in Chapter 4 an algorithm which limits $c_{\text{max}}$ to $\sum_{j=0}^{i} \binom{N}{j}$ at the start of order-$i$ reprocessing. Then, the search space is further reduced as the algorithm advances, in a fashion similar to [8, 22]. This way, while maintaining the efficiency of the algorithm with a small $c_{\text{ave}}$, we also provide the limiting worst case $c_{\text{max}}$. Note
that we no longer guarantee the decoding of the ML codeword, but the error performance associated with ML decoding. For codes of length $N \leq 64$, the algorithm is both extremely efficient and easily implementable. For longer codes, despite a very low $c_{\text{ave}}$, optimality is achieved with $c_{\text{max}}$ out of practical range. However, the same approach provides implementable decoding algorithms with good trade-off between error performance and computational complexity. The development of more powerful criteria than in [8] to reduce $c_{\text{ave}}$, or different approaches to reduce $c_{\text{max}}$ would provide significant contribution to ML decoding of linear block codes.

It is also obvious that combining the statistical results of this thesis with the algorithm of [22] immediately reduces the computational cost of [22], since $c_{\text{max}}$ decreases from $2^K$ to $\sum_{j=0}^{i} \binom{N}{j}$. This is achieved by opening only nodes which differ in at most $i$ MRI information symbols with $a$. Simulation results of the combined algorithm would be extremely interesting from a practical point of view. In a similar way, using the information of an algebraic decoder as in [8] with the monotonic properties of the ordering provides a stronger stopping criterion, and thus further reduces $c_{\text{ave}}$.

We also described in Chapter 4 the equivalent algorithm in the dual space of the code, after ordering. The same approach as [9, 10, 11] is therefore considered and the new version of our algorithm provides an efficient way to search the columns of the parity check matrix of the code, for any $N - K$. In addition, with this approach, information about the dual code may lead to new improvements for the resource test.

Efficient trellis decoding of any block code is inherently related to the decomposable structure of the code considered. In Section 5.3, we adapted our algorithm for decoding codes based on the $|u|u+v|$ construction. For these codes, we presented practically optimum, as well as suboptimum applications of our reprocessing algorithm which can reduce the total or average number of computations, or speed
up the decoding process. Based on the same principle, devising algorithms for other code constructions seems a promising area of future research.

In Chapter 6, generalization of the algorithm to other channel model was illustrated by the Raleigh fading channel with coherent detection. Besides the theoretical and practical results presented, this chapter also provides the major steps to analyse other channel models.

For large $N$, approximations for the probability that a group of positions is in error after ordering are derived in Chapter 7. These results are useful to analyse suboptimum algorithms based on ordering, since no close form solution of the corresponding error performance has been found, for $N > 2$. However, the results become less accurate as the positions approach $N$, and do not even hold if $N$ is the last position considered. While this fact did not affect our results, it would be interesting to specifically approximate the statistics for the last positions of the ordering.

Based on these results, we provide a good simple model for the OBSC. Further theoretical work on this model, as well as higher order approximations can be realised. This model was also considered in Chapter 8, where improvements of conventional Majority-Logic-Decoding of RM codes are presented. The decoding of any majority decodable code based on the same principle worth being investigated.
Appendix A

Duality Between $H_1$ and $G_1$

The first permutation $\lambda_1$ can be represented by the $N \times N$ matrix $M = [m_{i,j}]$ such that, for $G' = \lambda_1[G] = G \cdot M$

$$m_{i,j} = \begin{cases} 1 & \text{if the } j^{th} \text{ column of } G \text{ becomes the } i^{th} \text{ column of } G' \\ 0 & \text{otherwise} \end{cases} \quad (A.1)$$

Since $MM^T = I_N$, $(\lambda_1[G])(\lambda_1[H])^T = G \cdot M \cdot M^T \cdot H = G \cdot H^T = [0]$, which verifies that $\lambda_1[H]$ is the parity check matrix of $\lambda_1[G]$.

Putting $G'$ into systematic form requires $K$ steps [25]. For $j \in [1, K]$, let $\tilde{G}_j$ denote the intermediate matrix obtained at step $j$ and define $\tilde{G}_0 = G'$. Then, we easily show $\tilde{G}_j = N_j \tilde{G}_{j-1}$, with, for the $K \times K$ matrix $N_j = [n_{r,c}]$,

$$n_{r,c} = \begin{cases} \delta_{r,c} & \text{if } c \neq j \\ g_{r,c} & \text{otherwise} \end{cases} \quad (A.2)$$

where $\delta_{r,c} = 1$ if $r = c$ and $\delta_{r,c} = 0$ if $r \neq c$. After the $K$ steps, the final matrix becomes $\tilde{G}_K = N_K \cdot N_{K-1} \cdots N_1 \cdot (G \cdot M)$. Then, the second permutation $\lambda_2$ repositions the dependent columns, as described in Section 2.2.1. This defines a second matrix $M_2$ as in Equation A.1. Finally, we re-order the rows of the obtained matrix to have $I_K$ in the first $K$ positions by a matrix $L_2$, so that

$$G_1 = L_2 \cdot N_K \cdot N_{K-1} \cdots N_1 \cdot G \cdot M \cdot M_2. \quad (A.3)$$
Since $G_1 H_1^T = [0]$, we finally obtain

$$H_1 = L_2' N_{N-K}' N_{N-K-1}' \cdots N_1' H M M_2,$$

(A.4)

which defines the same second permutation $\lambda_2$. 

155
Appendix B

Conditional Densities of the Noise after Ordering

B.1 Marginal Conditional Density of $W_i$

In this section, we determine the density function of the $i^{th}$ noise value $W_i$ after ordering, conditioned on the knowledge of the corresponding transmitted symbol $X_i = x_i$. A capital letter represents a random variable while a lower case letter stands for a deterministic value. As described in Section 2.2, we order the received symbols with the new labeling $i < j$ for $|x_i + w_i| > |x_j + w_j|$, where $x_i$ represents the transmitted symbol and $w_i$ the noise value associated with this symbol.

We first assume $X_i = -1$. For $j > 0$,

$$|W_{i+j} + x_{i+j}| \leq |w_i - 1|$$

implies that, for $x_{i+j} = -1$,

$$W_{i+j} \in [2 - w_i, w_i] \quad \text{if} \quad w_i \geq 1,$$

$$W_{i+j} \in [w_i, 2 - w_i] \quad \text{if} \quad w_i \leq 1.$$  \hspace{1cm} (B.2)

and, for $x_{i+j} = 1$,

$$W_{i+j} \in [-w_i, w_i - 2] \quad \text{if} \quad w_i \geq 1,$$

$$W_{i+j} \in [w_i - 2, -w_i] \quad \text{if} \quad w_i \leq 1.$$  \hspace{1cm} (B.3)
Since the bipolar transmitted digits are independent, equiprobable and independent of the zero-mean white Gaussian noise, we observe that, for $w_i \geq 1,$

$$
P(W_{i+j} \in [2-w_i, w_i] | X_{i+j} = -1) = P(W_{i+j} \in [-w_i, w_i-2] | X_{i+j} = 1) = P(W \in [2-w_i, w_i]),
$$

where $W$ represents the white noise before ordering. The same equation holds for $w_i \leq 1$ after permuting the limits of the interval considered.

The case $X_i = 1$ is identical when comparing $w_i$ with respect to $-1.$ Equation B.4 becomes, for $w_i \leq -1$,

$$
P(W_{i+j} \in [2+w_i, -w_i] | X_{i+j} = -1) = P(W_{i+j} \in [w_i, -w_i-2] | X_{i+j} = 1) = P(W \in [2+w_i, -w_i]).
$$

By comparing Equation B.4 and Equation B.5, we obtain, for $j > 0$

$$
P(|W_{i+j} + x_{i+j} \leq| w_i + s_i | | X_i = s_i) = P(W \in [m(w_i), M(w_i)] | X_i = s_i),
$$

where

$$
m(w_i) = \min(2+s_iw_i, -s_iw_i),
$$

$$
M(w_i) = \max(2+s_iw_i, -s_iw_i).
$$

For $j > 0,$ the same development leads to the complementary conclusion

$$
P(|W_{i-j} + x_{i-j} | \geq | w_i + s_i | | X_i = s_i) = P(W \in (-\infty, m(w_i)] \cup [M(w_i), \infty) | X_i = s_i),
$$

and we note

$$
f_W(w_i)dw_i = P(w_i \leq W \leq w_i + dw_i).
$$

Following the proof of [35, p.185], the three events $E_1 = \{| W + x | > | w_i + s_i |\}$, $E_2 = \{| w_i \leq W \leq w_i + dw_i\}$ and $E_3 = \{| W + x | < | w_i + dw_i + s_i |\}$ are disjoint and
from Equations B.6, B.7 and B.8, have respective associated probabilities

\[ P(E_1) = (\pi N_0)^{-1/2} \left( \int_{-\infty}^{m(w_i)} e^{-x^2/2N_0} dx + \int_{M(w_i)}^{\infty} e^{-x^2/2N_0} dx \right), \quad \text{(B.9)} \]

\[ P(E_2) = (\pi N_0)^{-1/2} e^{-w_i^2/2N_0} dw_i, \quad \text{(B.10)} \]

\[ P(E_3) = (\pi N_0)^{-1/2} \int_{m(w_i)}^{M(w_i)} e^{-x^2/2N_0} dx. \quad \text{(B.11)} \]

When \( E_1 \) occurs \( i - 1 \) times, \( E_2 \) once and \( E_3 \) \( N - i \) times, we obtain

\[ f_{W_iX_i}(w_i | X_i = s_i) = \frac{(\pi N_0)^{-N/2} N!}{(i-1)!(N-i)!} \left( \int_{-\infty}^{m(w_i)} e^{-x^2/2N_0} dx + \int_{M(w_i)}^{\infty} e^{-x^2/2N_0} dx \right)^{i-1} e^{-w_i^2/2N_0} \left( \int_{m(w_i)}^{M(w_i)} e^{-x^2/2N_0} dx \right)^{N-i}. \quad \text{(B.12)} \]

The marginal density of the noise after ordering is easily obtained as, for equiprobable signaling,

\[ f_{W_i}(w_i) = 1/2 \left( f_{W_i|X_i}(w_i | X_i = -1) + f_{W_i|X_i}(w_i | X_i = 1) \right) = f_{W_i}(-w_i). \quad \text{(B.13)} \]

### B.2 Joint Conditional Density of \( W_i \) and \( W_j \)

The joint conditional density is obtained in a similar way. We first assume \( X_i = X_j = -1 \) and \( i < j \). For \( k > 0 \),

\[ | W_{j+k} + x_{j+k} | \leq | w_j - 1 | \leq | w_i - 1 | \quad \text{(B.14)} \]

is equivalent to Equation B.1 for \( | W_{j+k} + x_{j+k} | \leq | w_j - 1 | \). Similarly,

\[ | w_j - 1 | \leq | w_i - 1 | \leq | W_{i-k} + x_{i-k} | \quad \text{(B.15)} \]

is equivalent to Equation B.7 for \( | w_i - 1 | \leq | W_{i-k} + x_{i-k} | \). For \( k \in (i, j) \), we obtain

\[ | w_j - 1 | \leq | W_k + x_k | \leq | w_i - 1 |, \quad \text{(B.16)} \]
which implies that, for \( w_j > 1 \) and \( w_i > 1 \),

\[
W_k \in [w_j, w_i] \cup [2 - w_i, 2 - w_j] \quad \text{if} \quad x_k = -1,
\]

\[
W_k \in [-w_i, -w_j] \cup [w_i - 2, w_j - 2] \quad \text{if} \quad x_k = 1. \tag{B.17}
\]

All symmetrical cases for \( w_i \) and \( w_j \), as well as the three other possible assumptions for \( X_i = \pm 1 \) and \( X_j = \pm 1 \) follow in a straightforward way, as previously. We finally obtain, for \( i < j \)

\[
f_{W_i, W_j | X_i, X_j}(w_i, w_j \mid X_i = s_i, X_j = s_j) = \frac{(\pi N_0)^{-N/2} N!}{(i - 1)! (j - i - 1)! (N - j)!} e^{-(w_i^2 + w_j^2)/N_0}
\]

\[
\cdot \left( \int_{-\infty}^{m(w_i)} e^{-x^2/N_0} dx + \int_{M(w_i)}^{\infty} e^{-x^2/N_0} dx \right)^{i-1} \left( \int_{m(w_j)}^{M(w_j)} e^{-x^2/N_0} dx \right)^{N-j}
\]

\[
\cdot \left( \int_{M(w_i)}^{\infty} e^{-x^2/N_0} dx + \int_{m(w_i)}^{M(w_j)} e^{-x^2/N_0} dx \right)^{j-i-1} \cdot 1_{[m(w_i), M(w_i)]}(w_j), \tag{B.18}
\]

where \( m(w) \) and \( M(w) \) are defined in Equation B.6 for \( s_i \) and \( s_j \), and \( 1_A(x) = 1 \) if \( x \in A \) and \( 1_A(x) = 0 \) otherwise.

The joint density of \( W_i \) and \( W_j \) is easily obtained when considering the 4 possible conditional joint densities.

## B.3 Joint Conditional Density of \( W_i \)'s

The same method remains available when considering more than two ordered noise values. We obtain for \( i < j < l \)

\[
f_{W_i, W_j, W_l | X_i, X_j, X_l}(w_i, w_j, w_l \mid X_i = s_i, X_j = s_j, X_l = s_l)
\]

\[
= \frac{(\pi N_0)^{-N/2} N!}{(i - 1)! (j - i - 1)! (l - j - 1)! (N - l)!} e^{-(w_i^2 + w_j^2 + w_l^2)/N_0}
\]

\[
\cdot \left( \int_{-\infty}^{m(w_i)} e^{-x^2/N_0} dx + \int_{M(w_i)}^{\infty} e^{-x^2/N_0} dx \right)^{i-1} \left( \int_{m(w_j)}^{M(w_j)} e^{-x^2/N_0} dx \right)^{N-j}
\]

\[
\cdot \left( \int_{M(w_i)}^{\infty} e^{-x^2/N_0} dx + \int_{m(w_i)}^{M(w_j)} e^{-x^2/N_0} dx \right)^{j-i-1} \cdot 1_{[m(w_i), M(w_i)]}(w_j), \tag{B.18}
\]
\[
\left( \int_{M(w_j)}^{M(w_i)} e^{-x^2/N_0} dx + \int_{m(w_j)}^{m(w_i)} e^{-x^2/N_0} dx \right)^{j-i-1} \cdot 1_{[m(w_i), M(w_i)]}(w_j)
\]
\[
\left( \int_{M(w_i)}^{M(w_j)} e^{-x^2/N_0} dx + \int_{m(w_i)}^{m(w_j)} e^{-x^2/N_0} dx \right)^{j-j-1} \cdot 1_{[m(w_j), M(w_j)]}(w_i).
\]

(B.19)

It becomes straightforward to generalize Equation B.19 to any number of noise values.
Appendix C

Statistics of $Z$ after Ordering with Respect to $|Z|$.

C.1 Marginal Density of $Z_i$

We first determine the density function of the $i^{th}$ value $Z_i$ after ordering with respect to $|Z|$ a sequence of length $N$. For $i < j$,

$$|Z_{i+j}| \leq |z_i|$$  \hspace{1cm} (C.1)

implies that,

$$Z_{i+j} \in [-z_i, z_i] \quad \text{if} \quad z_i \geq 0,$$

$$Z_{i+j} \in [z_i, -z_i] \quad \text{if} \quad z_i \leq 0. \hspace{1cm} (C.2)$$

and, similarly,

$$|Z_{i-j}| \geq |z_i|$$  \hspace{1cm} (C.3)

implies that,

$$Z_{i-j} \in (-\infty, -z_i] \cup [z_i, \infty) \quad \text{if} \quad z_i \geq 0,$$

$$Z_{i-j} \in (-\infty, z_i] \cup [-z_i, \infty) \quad \text{if} \quad z_i \leq 0. \hspace{1cm} (C.4)$$
Following the same steps as in Appendix B, the marginal density of $Z_i$ after ordering is easily obtained as

$$f_{Z_i}(z_i) = \frac{N!}{(i - 1)!(N - i)!} \left( \int_{-\infty}^{m_i} f_Z(z)dz + \int_{M_i}^{\infty} f_Z(z)dz \right)^{i-1} \left( \int_{m_i}^{M_i} f_Z(z)dz \right)^{N-i} f_Z(z_i),$$

with $m_i = \min(-z_i, z_i)$ and $M_i = \max(-z_i, z_i).$ (C.5)

\[C.6\]

\section*{C.2 Joint Density of $Z_i$ and $Z_j$}

In addition to Equations C.1 and C.3, we consider, for $i < k < j,$

$$| z_i | \geq | Z_k | \geq | z_j |,$$

which is equivalent to

$$Z_k \in [-z_i, -z_j] \cup [z_j, z_i] \text{ if } z_i > z_j > 0.$$ (C.8)

The three symmetrical cases with respect to $z_i$ and $z_j$ follow in a straightforward way. Therefore, combining the proof given in Appendix B.2 with the approach of Section C.1, we obtain

$$f_{Z_i,Z_j}(z_i, z_j) = \frac{N!}{(i - 1)!(j - i - 1)!(N - j)!} \left( \int_{-\infty}^{m_i} f_Z(z)dz + \int_{M_i}^{\infty} f_Z(z)dz \right)^{i-1} \left( \int_{m_j}^{M_j} f_Z(z)dz \right)^{j-i} \left( \int_{m_j}^{M_j} f_Z(z)dz \right)^{N-j} \cdot f_Z(z_i) f_Z(z_j) 1_{[m_i,M_j]}(z_j).$$ (C.9)

The same method remains available when considering more than two ordered values.

\section*{C.3 $Z = -\alpha A + W$}

In this section, we consider a random variable $Y$ of the form $Y = AX + W$. We assume that $X$ takes symmetrical values $\pm \alpha$ with respect to the origin with equal
probability 1/2, and that both the density functions of the random variable \( A \) and \( W \) are known. Therefore, \( Y \) can represent a bipolar equiprobable transmission scheme perturbed by additive noise \( W \) and distorted by \( A \). We then define the random variable \( Z = (Y|X = -\alpha) \).

By the definition of \( Z \), we consider only the all-\((\alpha)\) transmitted sequence. The signaling represented by \( X \) is equiprobable and symmetrical with respect to the origin. If in addition \( X \), \( W \) and \( A \) are independent, if the density of \( W \) is symmetrical with respect to the origin and if \( A \) takes only non negative or non positive values, then Equations C.5, C.9 or equivalent higher order forms are sufficient to evaluate the error performance of any decoding scheme based on a hard decision \( \text{sgn}(Y) \) of \( Y \) after the ordering with respect to \( |Y| \). This statement is easily justified by generalizing the lengthy and tedious approach of Appendix B.
Appendix D
Algorithm for First Stage of Closest Coset Decoding

The algorithm to obtain the most likely coset in the first stage of closest coset decoding follows.

• Step 1: For \( i \in [1, N/2] \), order the received sequence with respect to \( \min(|y_{1,i}|, |y_{2,i}|) \) and reprocess \( G_2 \) to obtain \( G_v \) in systematic form.

• Step 2: For \( i \in [1, K_2] \), perform the hard decision decodings,

\[
\begin{align*}
  d_u^i &= 0 \quad \text{if} \quad y_1^i \geq 0 \\
  d_u^i &= 1 \quad \text{if} \quad y_1^i < 0, \\
\end{align*}
\]

and,

\[
\begin{align*}
  d_v^i &= d_u^i \quad \text{if} \quad y_2^i \geq 0 \\
  d_v^i &= d_u^i \oplus 1 \quad \text{if} \quad y_2^i < 0. \\
\end{align*}
\]

• Step 3: For \( i \in [K_2, N/2] \), compute in \( GF(2) \)

\[
d_v^i = \sum_{j=1}^{K_2} g_v[j][i] \cdot d_v^j, \tag{D.3}
\]

where \( G_v = (g_v[j][i]) \).

• Step 4: For \( i \in [K_2, N/2] \), perform the hard decision decoding,
If $|y_1| \geq |y_2|$, 

\[
d_u^i = 0 \quad \text{if} \quad y_1^i \geq 0 \\
d_u^i = 1 \quad \text{if} \quad y_1^i < 0,
\]

Else

\[
d_u^i = d_v^i \quad \text{if} \quad y_2^i \geq 0 \\
d_u^i = d_v^i \oplus 1 \quad \text{if} \quad y_2^i < 0.
\]

**Step 5:** For $i \in [1, N/2]$, compute

\[
\Delta_i(\bar{\alpha}) = \frac{1}{2} \left( (-1)^{d_u^i} y_1^i + (-1)^{d_v^i} y_2^i \right).
\]

**Step 6:** Apply order-$i$ reprocessing to the $(N/2, K_2)$ code with generator matrix $G_\upsilon$ as described in Chapter 4. The resource is computed with $\delta_i(\bar{\alpha}) = \min(|y_1^i|, |y_2^i|)$.

The number of real value operations for this algorithm is $N/2 + N/2 \log_2(N/2)$ for step 1, $2K_2$ for step 2, $2(N/2 - K_2)$ for step 4, so a total of $N/2 \log_2(N) + N$ for steps 1 to 4. Then, step 5 corresponds to order-$0$ reprocessing and step(s) 6 to order-$i$, for $i \geq 1$. 

165
Appendix E

Weight Distribution for Closest Coset Decoding of $|u|u + v|$ Constructed Codes

In this appendix, we derive the exact weight distribution of the equivalent code when performing Closest Coset Decoding (CCD), as described in [19].

E.1 Equivalent Weight Distribution for CCD Based on the $|u|u + v|$ Construction

The generator matrix $G$ of the $(N, K, d_H)$ code $C$ considered is of the form

$$G = \begin{bmatrix} G_1 & G_1 \\ 0 & G_2 \end{bmatrix}, \quad (E.1)$$

where $G_i$ is the generator matrix of an $(N/2, K_i, d_{H_i})$ code $C_i$. Then, it is well known that $d_H = \min(2d_{H_1}, d_{H_2})$. We assume $C_i$'s are binary linear codes, so that, without loss of generality, we consider the transmitted codeword $\bar{c} = [\bar{c}_1 | \bar{c}_1 \oplus \bar{c}_2] = 0$. For CCD, the decoder at the first stage assumes $\bar{c}_1$ can take any $N/2$-tuples value and selects the coset $[\bar{c}_2]$. For the first stage, this defines an equivalent supercode $C_{s1}$ which contains $C$, with generator matrix

$$G_{s1} = \begin{bmatrix} I_{N/2} & I_{N/2} \\ 0 & G_2 \end{bmatrix}, \quad (E.2)$$

166
Table E.1. $\hat{c}_1$ and $\hat{c}_2$ contributions to $w(\hat{c})$.

<table>
<thead>
<tr>
<th>$\hat{c}_{1,i}$</th>
<th>$\hat{c}_{2,i}$</th>
<th>$w(\hat{c}_i)$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

where $I_{N/2}$ represents the $N/2 \times N/2$ identity matrix. Then, at the second stage, the decoder decodes within the coset $[\hat{c}_2]$ the remaining $K_1$ bits based on the generator matrix

$$G_{s2} = [G_1 \ G_1].$$  \hfill (E.3)

A decoding error occurs either if the wrong coset is chosen at the first stage, or if the right coset is chosen but the wrong codeword is delivered by the second stage.

We first state a fact that not only shows that CCD based on the $|u|u+v|$ construction does not reduce the Hamming distance $d_H$ of the code $C$, but also implies that if $d_{H_2} > d_H$, then the equivalent weight distribution for CCD of $C$ between $d_H = 2d_{H_1}$ and $d_{H_2} - 1$ remains unchanged.

**Fact 1:** If $\hat{c} = [\hat{c}_1 \hat{c}_1 \oplus \hat{c}_2] \in C_{s1}$ is delivered by the first decoding stage with $w(\hat{c}) < d_{H_2}$, then the right coset $[\hat{c}_2]$ is chosen.

**Proof:**

For $\bar{c} = \bar{0}$, let assume $\hat{c}_2 \neq \bar{0}$, in which case the wrong coset is chosen. Table E.1 resumes the possible contributions from $\hat{c}_1$ and $\hat{c}_2$ to $w(\hat{c})$. $\hat{c}_2 \neq \bar{0}$ implies $w(\hat{c}_2) \geq d_{H_2}$, so that from Table E.1, $w(\hat{c}) \leq d_{H_2}$. The proof is completed by using the negation of this implication.

Let $A_w$ represent the number of codewords of weight $w$ of $C_{s1}$ which are not in the coset $[\hat{c}_2] = [\bar{0}]$. 

167
Fact 2: Let \( \hat{e}_2 \in C_2 \) with \( w_2 = w(\hat{e}_2) > 0 \). Then \( w_2 \) contributes to \( A_{w_2}, A_{w_2+2}, A_{w_2+4}, \ldots, A_{w_2+2(N/2-w_2)} \).

Proof:

From Table E.1, if \( \hat{e}_1 = \emptyset, \hat{e}_2 \) obviously contributes to \( A_{w_2} \), which corresponds in fact to the case where \( \hat{e}_1 \) is correct but the wrong codeword of \( C_2 \), and thus the wrong coset is selected. If \( \hat{e}_1 \neq \emptyset, \) Table E.1 indicates that whenever \( \hat{e}_{1,i} = 1 \) and \( \hat{e}_{2,i} = 0 \), a contribution of 2 is provided to \( w(\hat{e}_i) \), while in any other case, the contribution of \( \hat{e}_2 \) remains unchanged. Finally, the maximum number of extra contributions from the \( N/2 \)-tuple \( \hat{e}_1 \) is simply \( N/2 - w_2 \).

Fact 2 shows that even \( w_2 \)'s contributes to even weights only, and reciprocally for odd weights. Therefore, if \( C_2 \) has only even weight codewords, \( A_{2i+1} = 0 \) for \( i \in [0, N/2 - 1] \).

Fact 3: For \( i \in [0, N/2 - w_2] \), the contribution of \( \hat{e}_2 \in C_2 \) with \( w_2 = w(\hat{e}_2) > 0 \) to \( A_{w_2+2i} \) is \( 2^{w_2} \cdot \left( \begin{array}{c} N/2 \vDash w_2 \end{array} \right) \).

Proof:

Let \( 0 \leq \alpha \leq N/2 - w_2 \) be the number of positions where \( \hat{e}_{1,i} = 1 \) and \( \hat{e}_{2,i} = 0 \). By the proof of Fact 2, the pair \((\hat{e}_1, \hat{e}_2)\) contributes to \( A_{w_2+2\alpha} \). To choose \( \hat{e}_1 \), from Table E.1, we observe that we can pick any \( w_2 \)-tuple corresponding to the positions where \( \hat{e}_{2,i} = 1 \), while \( \left( \begin{array}{c} N/2 \vDash w_2 \end{array} \right) \) different choices are also possible for the \( N/2 - w_2 \) positions where \( \hat{e}_{2,i} = 0 \). The corresponding number of choices for \( \hat{e}_1 \) is therefore \( 2^{w_2} \cdot \left( \begin{array}{c} N/2 \vDash w_2 \end{array} \right) \).

Regrouping Facts 2 and 3, we obtain, for \( N/2 \) even and \( i \in [0, N/4] \),

\[
A_{N/2-2i} = \sum_{j=0}^{N/4-i-1} 2^{N/2-2i-2j} \left( \begin{array}{c} 2(i+j) \\ j \end{array} \right) N_{2,N/2-2i-2j} \quad (E.4)
\]

\[
A_{N/2+2i} = \sum_{j=2i}^{N/4+i-1} 2^{N/2+2i-2j} \left( \begin{array}{c} 2(j-i) \\ j \end{array} \right) N_{2,N/2+2i-2j} \quad (E.5)
\]
where \( N_{i,w} \) denotes the number of codewords of weight \( w \) in \( C_i, i \in [1,2] \). Similar expressions are derived for the complementary cases.

If \( B_w \) represents the number of codewords of weight \( w \) for the repetition code \( C_{22} \), we immediately obtain, for \( i \in [0,N/2] \),

\[
B_{2i} = N_{1,i} \\
B_{2i+1} = 0.
\]

(E.6)

(E.7)

The weight distribution corresponding to CCD based on the \( u|u+v| \) construction is finally given by:

For \( N/2 \) even and \( i \in [0,N/4] \),

\[
W_{N/2-2i} = N_{1,N/4-i} + \sum_{j=0}^{N/4-i-1} 2^{N/2-2i-2j} \binom{2(i+j)}{j} N_{2,N/2-2i-2j}, \\
W_{N/2+2i} = N_{1,N/4+i} + \sum_{j=2i}^{N/4+i-1} 2^{N/2+2i-2j} \binom{2(j-i)}{j} N_{2,N/2+2i-2j}.
\]

(E.8)

(E.9)

For \( N/2 \) even and \( i \in [0,N/4-1] \),

\[
W_{N/2-1-2i} = \sum_{j=0}^{N/4-i-1} 2^{N/2-2i-2j-1} \binom{2(i+j)+1}{j} N_{2,N/2-2i-2j-1}, \\
W_{N/2+1+2i} = \sum_{j=2i+1}^{N/4+i-1} 2^{N/2+2i-2j+1} \binom{2(j-i)-1}{j} N_{2,N/2+2i-2j+1}.
\]

(E.10)

(E.11)

For \( N/2 \) odd and \( i \in [0,N/4-1/2] \),

\[
W_{N/2-2i} = \sum_{j=0}^{N/4-i-1/2} 2^{N/2-2i-2j} \binom{2(i+j)}{j} N_{2,N/2-2i-2j}, \\
W_{N/2+2i} = \sum_{j=2i}^{N/4+i-1/2} 2^{N/2+2i-2j} \binom{2(j-i)}{j} N_{2,N/2+2i-2j}.
\]

(E.12)

(E.13)

For \( N/2 \) odd and \( i \in [0,N/4-1/2] \),

\[
W_{N/2-2i-1} = N_{1,N/4-i-1/2}
\]
We verify that all \( N_2,w \) but \( N_2,0 \) contributes to these equations, as expected. Also, the total number of codewords is now

\[
N_{\text{tot}} = 2^{N/2} \left( 2^{K_2} - 1 \right) + 2^{K_1}. \tag{E.16}
\]

Since \( C \) has \( 2^{K_1+K_2} \) codewords, the code \( C_1 \) has to be chosen with small redundancy to maintain CCD efficient, as mentioned in [20].

### E.2 Application to RM Codes

In this section, using the previous results, we evaluate the error performance of CCD based on the \(|u|u+v|\) construction for Reed-Muller (RM) codes. We denote \( RM(r, m) \) the \( r \)th order RM code with generator matrix \( G(r, m) \), and parameters \( N = 2^m \), \( K = K(r, m) = \sum_{i=0}^{r} \binom{m}{i} \) and \( d_H = 2^{m-r} \). We also define \( \tilde{W}(r, m) \) as the number of codewords of minimum weight for the \( RM(r, m) \) code.

From Equation E.9, we obtain for CCD of the \( RM(r, m) \) code,

\[
W_{2^m-r} = 2^{2^m-r} \tilde{W}(r-1, m-1) + \tilde{W}(r-1, m). \tag{E.17}
\]

Therefore, \( W_{2^m-r} \) depends only on \( m-r \). For fixed \( m-r \), the influence of CCD on the first term of the equivalent weight distribution does not increase with the dimension of the code. For \( m-r \geq 2 \), the second coefficient of the equivalent weight distribution becomes, according to Equation E.9,

\[
W_{2^m-r+2} = \left( 2^{m-1} - 2^{m-r} \right) \left( W_{2^m-r} - \tilde{W}(r-1, m) \right), \tag{E.18}
\]

170
and thus increases with the dimension of the code.

We now compare \( W_{d_H} \) defined in Section E.1 with the upper bound of [20]

\[
W_{d_H} \leq 2^{N/2 - K_1(N_{d_H} - N_{1,d_H})} + N_{1,d_H},
\]

(E.19)

where \( N_{d_H} \) is the number of codewords of weight \( d_H \) in the code \( C \), or equivalently,

\[
W_{2m-r} \leq 2^{m-1-K(r,m-1)} \left( \tilde{W}(r,m) - \tilde{W}(r,m-1) \right) + \tilde{W}(r,m-1).
\]

(E.20)

Using the inductive relation

\[
\tilde{W}(r,m) - \tilde{W}(r,m-1) = 2^{m-r+1} \tilde{W}(r-1,m-1),
\]

(E.21)

Equation E.17 can be rewritten as

\[
W_{2m-r} = 2^{m-r-K(1,m-r)} \left( \tilde{W}(r,m) - \tilde{W}(r,m-1) \right) + \tilde{W}(r,m-1).
\]

(E.22)

To compare Equations E.20 and E.22, we define for \( r \leq m - 2 \) the difference

\[
\Delta(r,m) = \sum_{i=r+1}^{m-1} \left( \begin{array}{c} m-1 \\ i \end{array} \right) - \sum_{i=2}^{m-r} \left( \begin{array}{c} m-r \\ i \end{array} \right) = \sum_{i=r+1}^{m-1} \left( \begin{array}{c} m-1 \\ i \end{array} \right) - \left( \begin{array}{c} m-r \\ i+1-r \end{array} \right).
\]

(E.23)

Note that for \( r = m - 1 \), CCD is equivalent to ML decoding since \( G(m-1,m-1) = I_{2m-1} \). From Equation E.23, we observe that \( \Delta(1,m) = \Delta(m-2,m) = 0 \), so that the bound of [20] is in fact the exact value for CCD of the \( RM(1,m) \) and \( RM(m-2,m) \) codes. For other RM codes with \( r \in [2,m-3] \), \( \Delta(r,m) < 0 \), so that Equation E.22 improves the error performance evaluation of CCD based on the \( |u|u+v| \) construction.
Appendix F

Reliability Measure

for the Raleigh Fading Channel

with Coherent Detection

Refering to the notations introduced in Chapter 6, the channel model is described by

\[ Y = AX + N. \]  

Then,

\[ \Pr(Y \leq y \mid X = x) = \int_0^\infty p_A(a) \Pr(N \leq y - ax) \, da, \]  \hspace{1cm} (F.2)

which implies

\[ p_{Y \mid X}(y \mid X = x) = 2(\pi N_0)^{-1/2} \int_0^\infty a e^{-a^2} e^{-\frac{(y-ax)^2}{N_0}} \, da. \]  \hspace{1cm} (F.3)

After completing the square in the exponent and integrating by parts, we obtain

\[ p_{Y \mid X}(y \mid X = x) = \left( \frac{N_0}{N_0 + x^2} \right)^{1/2} e^{-\frac{x^2}{N_0}} \]
\[ + \frac{2xy(N_0 + x^2)^{-3/2} e^{-\frac{x^2}{N_0}}}{\sqrt{N_0(N_0 + x^2)}} \]
\[ Q \left( \frac{\sqrt{2xy}}{\sqrt{N_0(N_0 + x^2)}} \right) \]  \hspace{1cm} (F.4)

For \( x_i \in \{-1, +1\} \), \( p_{Y \mid X}(y_1 \mid X = x_1) > p_{Y \mid X}(y_2 \mid X = x_2) \) is equivalent to, from Equation F.4,

\[ x_1y_1 Q \left( \frac{\sqrt{2x_1y_1}}{\sqrt{N_0(N_0 + 1)}} \right) > x_2y_2 Q \left( \frac{\sqrt{2x_2y_2}}{\sqrt{N_0(N_0 + 1)}} \right), \]  \hspace{1cm} (F.5)
or simply, for $x_i = \text{sgn}(y_i)$,

$$|y_1| > |y_2|.$$  \hspace{1cm} (F.6)

Note finally that Equation F.4 represents the density function of the sum of a Gaussian distributed random variable and a Raleigh distributed random variable scaled by the scalar $x$. 


Appendix G

Normal Approximation of

$$\tilde{W}_i | X_i = -1$$

For $t > 1$, we observe the equivalence between the two events

$$\{\tilde{W}_i > t\} \equiv \left\{ \sum_{j=1}^{N} 1_{[t-1,t]}(N_j) < N - (i - 1) \right\}, \quad (G.1)$$

where $N_j$ represents the $j^{th}$ noise value before ordering. We define the normalized random variable

$$Z_N = -\frac{1}{\sqrt{N}} \sum_{j=1}^{N} 1_{[t-1,t]}(N_j), \quad (G.2)$$

with mean $E[Z_N] = -\sqrt{N} \alpha(t)$ and variance $\sigma_{Z_N}^2 = \alpha(t)(1 - \alpha(t)) < 1$, where $\alpha(t) = \bar{Q}(2 - t) - \bar{Q}(t) \approx \bar{Q}(2 - t)$ for $t > 1$. For a particular value $t > 1$ and $N$ large enough, the central limit theorem allows to approximate the distribution of $Z_N$ by the distribution of a normal random variable $\eta \left( -\sqrt{N} \alpha(t), \alpha(t)(1 - \alpha(t)) \right)$ with mean $E[Z_N]$ and variance $\sigma_{Z_N}^2$ [51]. Also, since the random variable $1_{[t-1,t]}(N_j)$ has finite third moment, a theorem due to Esseen and Berry ensures a convergence at an $O(N^{-1/2})$ speed [52, p.201]. Therefore

$$\{\tilde{W}_i > t\} \equiv \left\{ Z_N > -\frac{N - i}{\sqrt{N}} \right\} \equiv \left\{ \eta(0,1) > \frac{-N(t) + N \alpha(t)}{\sqrt{N} \alpha(t)(1 - \alpha(t))} \right\}$$
\[
\begin{align*}
\mathbb{E} \{ \eta(0, 1) > \frac{it - N(1 - \alpha(t))t}{\sqrt{N\alpha(t)(1 - \alpha(t))t^2}} \} \\
\mathbb{E} \{ \eta \left( \frac{N(1 - \alpha(t))t}{i}, \frac{N\alpha(t)(1 - \alpha(t))t^2}{i^2} \right) > t \}.
\end{align*}
\] (G.3)

However, we seek a solution independent of \(t\). From Equation G.3, the equivalence

\[
\{ \bar{W}_i > E[\bar{W}_i] \} \equiv \{ \eta \left( E[\bar{W}_i], \sigma_{\bar{W}_i}^2 \right) > E[\bar{W}_i] \}
\] (G.4)

is obtained for \(t_0 = E[\bar{W}_i]\). This implies \(N/i (1 - \alpha(t_0)) = 1\), or equivalently,

\[
t_0 = E[\bar{W}_i] = \alpha^{-1} \left( 1 - \frac{i}{N} \right).
\] (G.5)

For \(t > 1\), \(\alpha(t) = u\) is bijective, so that \(\alpha^{-1}(u) = t\) is uniquely defined. Also, for \(i \neq N\), Equation G.5 is well approximated by \(t_0 \approx 2 - \tilde{q}^{-1} \left( 1 - \frac{i}{N} \right)\).

Using the equivalence between the events \{\(\eta(m, \sigma^2) > t\)\} and \{\(\eta(0, 1) > -m/\sigma + t/\sigma\)\} and rewriting, for \(K > 0\),

\[
\frac{i - N(1 - \alpha(t))}{\sqrt{N\alpha(t)(1 - \alpha(t))}} = -\frac{K}{\sqrt{N\alpha(t)(1 - \alpha(t))}} = t_0 + \frac{Kt_0 + i - N(1 - \alpha(t))}{\sqrt{N\alpha(t)(1 - \alpha(t))}} t,
\] (G.6)

we identify,

\[
K = \frac{i - N(1 - \alpha(t))}{t - t_0},
\] (G.7)

which implies

\[
\sigma^2(t) = \frac{N\alpha(t)(1 - \alpha(t))(t - t_0)^2}{(i - N(1 - \alpha(t)))^2}.
\] (G.8)

Applying twice L'Hospital's rule to Equation G.9, we obtain

\[
\sigma_{\bar{W}_i}^2 = \lim_{t \to t_0} \sigma^2(t) = \pi N_0 \frac{i(N - i)}{N^2} \left( e^{-t_0 - 2} / N_0 + e^{-t_0^2 / N_0} \right)^{-2}.
\] (G.9)

Equation G.9 shows that \(\sigma_{\bar{W}_i}^2\) is maximum for \(i = N/2\) and decreases monotonically to 0 when \(i\) tends to either 0 or \(N\). Therefore, we expect the convergence in distribution to be worst at the extremes (and particularly for \(i = N\)) and to improve as \(i \)
becomes closer to $N/2$. We verified experimentally that both Equations G.5 and G.9 closely match the experimental results, and that the matching improves as $i$ tends to $N/2$, for $N$ large enough.
Bibliography


