Foreword

This chapter is based on lecture notes from coding theory courses taught by Venkatesan Guruswami at University at Washington and CMU; by Atri Rudra at University at Buffalo, SUNY and by Madhu Sudan at MIT.

This version is dated **May 1, 2013**. For the latest version, please go to

http://www.cse.buffalo.edu/ atri/courses/coding-theory/book/

The material in this chapter is supported in part by the National Science Foundation under CAREER grant CCF-0844796. Any opinions, findings and conclusions or recomendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the National Science Foundation (NSF).

Chapter 11

Decoding Concatenated Codes

In this chapter we study Question 9.3.1. Recall that the concatenated code $C_{\text{out}} \circ C_{\text{in}}$ consists of an outer $[N, K, D]_{Q=q^k}$ code C_{out} and an inner $[n, k, d]_q$ code C_{in}, where $Q = O(N)$. (Figure 11.1 illustrates the encoding function.) Then $C_{\text{out}} \circ C_{\text{in}}$ has design distance Dd and Question 9.3.1 asks if we can decode concatenated codes up to half the design distance (say for concatenated codes that we saw in Section 9.2 that lie on the Zyablov bound). In this chapter, we begin with a very natural unique decoding algorithm that can correct up to $Dd/4$ errors. Then we will consider a more sophisticated algorithm that will allow us to answer Question 9.3.1 in the affirmative.

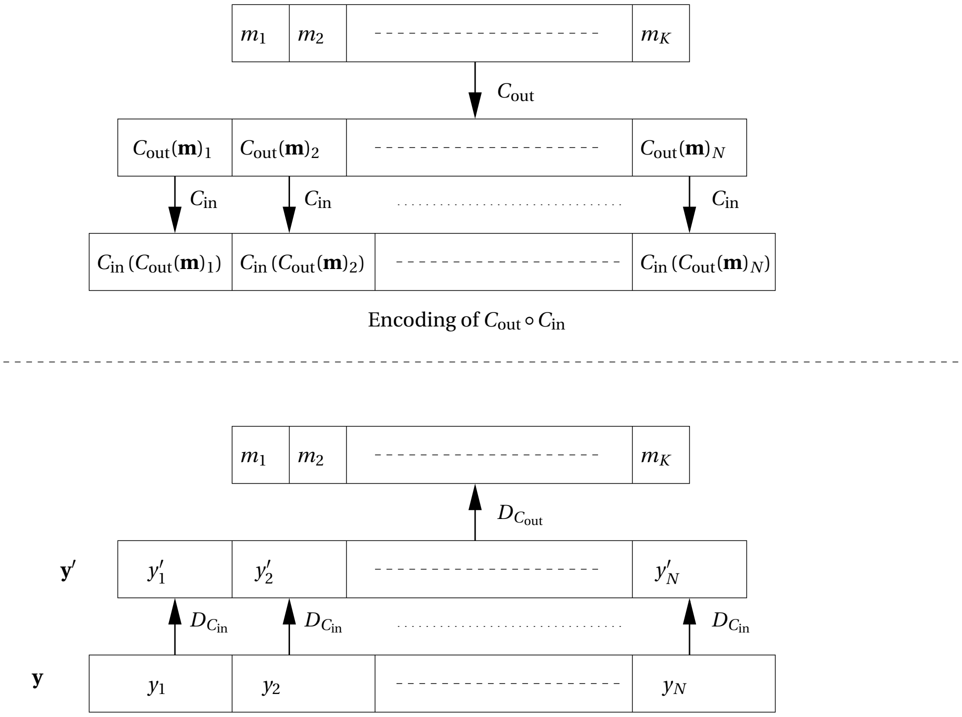
11.1 A Natural Decoding Algorithm

We begin with a natural decoding algorithm for concatenated codes that “reverses" the encoding process (as illustrated in Figure 11.1). In particular, the algorithm first decodes the inner code and then decodes the outer code.

For the time being let us assume that we have a polynomial time unique decoding algorithm $D_{C_{\text{out}}} : \left[q^k\right]^N \to \left[q^k\right]^K$ for the outer code that can correct up to $D/2$ errors.

This leaves us with the task of coming up with a polynomial time decoding algorithm for the inner code. Our task of coming up with such a decoder is made easier by the fact that the running time needs to be polynomial in the *final* block length. This in turn implies that we would be fine if we pick a decoding algorithm that runs in singly exponential time in the inner block length as long as the inner block length is logarithmic in the outer code block length. (Recall that we put this fact to good use in Section 9.2 when we constructed explicit codes on the Zyablov bound.) Note that the latter is what we have assumed so far and thus, we can use the Maximum Likelihood Decoder (or MLD) (e.g. its implementation in Algorithm 1, which we will refer to as $D_{C_{\text{in}}}$). Algorithm 7 formalizes this algorithm.

It is easy to check that each step of Algorithm 7 can be implemented in polynomial time. In particular,



Encoding of $C_{\text{out}} \circ C_{\text{in}}$

Decoding of $C_{\text{out}} \circ C_{\text{in}}$

Figure 11.1: Encoding and Decoding of the concatenated code $C_{\text{out}} \circ C_{\text{in}}$. $D_{C_{\text{out}}}$ is a unique decoding algorithm for C_{out} and $D_{C_{\text{in}}}$ is a unique decoding algorithm for the inner code (e.g. MLD).

Algorithm 7 Natural Decoder for $C_{\text{out}} \circ C_{\text{in}}$

```
INPUT: Received word $\mathbf{y} = (y_1, \cdots, y_N) \in [q^n]^N$
OUTPUT: Message $\mathbf{m}' \in [q^k]^K$

$\mathbf{y}' \leftarrow (y'_1, \cdots, y'_N) \in [q^k]^N$ where
    $$C_{\text{in}}(y'_i) = D_{C_{\text{in}}}(y_i) \;\; 1 \le i \le N.$$
$\mathbf{m}' \leftarrow D_{C_{\text{out}}}(\mathbf{y}')$
RETURN $\mathbf{m}'$
```

1. The time complexity of Step 1 is $O(nq^k)$, which for our choice of $k = O(\log N)$ (and constant rate) for the inner code, is $(nN)^{O(1)}$ time.

2. Step 2 needs polynomial time by our assumption that the unique decoding algorithm $D_{C_{\text{out}}}$ takes $N^{O(1)}$ time.

Next, we analyze the error-correction capabilities of Algorithm 7:

Proposition 11.1.1. *Algorithm 7 can correct* $< \frac{Dd}{4}$ *many errors.*

Proof. Let $\mathbf{m}$ be the (unique) message such that $\Delta\left(C_{\text{out}} \circ C_{\text{in}}(\mathbf{m}), \mathbf{y}\right) < \frac{Dd}{4}$.

We begin the proof by defining a bad event as follows. We say a *bad event* has occurred (at position $1 \le i \le N$) if $y_i \ne C_{\text{in}}(C_{\text{out}}(\mathbf{m})_i)$. More precisely, define the set of all bad events to be

$$\mathscr{B} = \left\{ i | y_i \ne C_{\text{in}}(C_{\text{out}}(\mathbf{m})_i) \right\}.$$

Note that if $|\mathscr{B}| < \frac{D}{2}$, then the decoder in Step 2 will output the message $\mathbf{m}$. Thus, to complete the proof, we only need to show that $|\mathscr{B}| < D/2$. To do this, we will define a superset $\mathscr{B}' \supseteq \mathscr{B}$ and then argue that $|\mathscr{B}'| < D/2$, which would complete the proof.

Note that if $\Delta\left(y_i, C_{\text{in}}(C_{\text{out}}(\mathbf{m})_i)\right) < \frac{d}{2}$ then $i \not\in \mathscr{B}$ (by the proof of Proposition 1.4.1)– though the other direction does not hold. We define $\mathscr{B}'$ to be the set of indices where $i \in \mathscr{B}'$ if and only if

$$\Delta\left(y_i, C_{\text{in}}(C_{\text{out}}(\mathbf{m})_i)\right) \ge \frac{d}{2}.$$

Note that $\mathscr{B} \subseteq \mathscr{B}'$.

Now by definition, note that the total number of errors is at least $|\mathscr{B}'| \cdot \frac{d}{2}$. Thus, if $|\mathscr{B}'| \ge \frac{D}{2}$, then the total number of errors is at least $\frac{D}{2} \cdot \frac{d}{2} = \frac{Dd}{4}$, which is a contradiction. Thus, $|\mathscr{B}'| < \frac{D}{2}$, which completes the proof. □

Note that Algorithm 7 (as well the proof of Proposition 11.1.1) can be easily adapted to work for the case where the inner codes are different, e.g. Justesen codes (Section 9.3).

Thus, Proposition 11.1.1 and Theorem 11.3.3 imply that

Theorem 11.1.2. *There exist an explicit linear code on the Zyablov bound that can be decoded up to a fourth of the Zyablov bound in polynomial time.*

This of course is predicated on the fact that we need a polynomial time unique decoder for the outer code. Note that Theorem 11.1.2 implies the existence of an explicit asymptotically good code that can be decoded from a constant fraction of errors.

We now state two obvious open questions. The first is to get rid of the assumption on the existence of $D_{C_{\text{out}}}$:

Question 11.1.1. *Does there exist a polynomial time unique decoding algorithm for outer codes, e.g. for Reed-Solomon codes?*

Next, note that Proposition 11.1.1 does not quite answer Question 9.3.1. We move to answering this latter question next.

11.2 Decoding From Errors and Erasures

Now we digress a bit from answering Question 9.3.1 and talk about decoding Reed-Solomon codes. For the rest of the chapter, we will assume the following result.

Theorem 11.2.1. *An $[N,K]_q$ Reed-Solomon code can be corrected from e errors (or s erasures) as long as $e < \frac{N-K+1}{2}$ (or $s < N-K+1$) in $O(N^3)$ time.*

We defer the proof of the result on decoding from errors to Chapter 13 and leave the proof of the erasure decoder as an exercise. Next, we show that we can get the best of both worlds by correcting errors and erasures simultaneously:

Theorem 11.2.2. *An $[N,K]_q$ Reed-Solomon code can be corrected from e errors and s erasures in $O(N^3)$ time as long as*

$$2e + s < N - K + 1. \tag{11.1}$$

Proof. Given a received word $\mathbf{y} \in (\mathbb{F}_q \cup \{?\})^N$ with s erasures and e errors, let $\mathbf{y}'$ be the sub-vector with no erasures. This implies that $\mathbf{y}' \in \mathbb{F}_q^{N-s}$ is a valid received word for an $[N-s,K]_q$ Reed-Solomon code. (Note that this new Reed-Solomon code has evaluation points that corresponding to evaluation points of the original code, in the positions where an erasure did not occur.) Now run the error decoder algorithm from Theorem 11.2.1 on $\mathbf{y}'$. It can correct $\mathbf{y}'$ as long as

$$e < \frac{(N-s)-K+1}{2}.$$

This condition is implied by (11.1). Thus, we have proved one can correct e errors under (11.1). Now we have to prove that one can correct the s erasures under (11.1). Let $\mathbf{z}'$ be the output after correcting e errors. Now we extend $\mathbf{z}'$ to $\mathbf{z} \in (\mathbb{F}_q \cup \{?\})^N$ in the natural way. Finally, run the erasure decoding algorithm from Theorem 11.2.1 on $\mathbf{z}$. This works as long as $s < (N-K+1)$, which in turn is true by (11.1).

The time complexity of the above algorithm is $O(N^3)$ as both the algorithms from Theorem 11.2.1 can be implemented in cubic time. □

Next, we will use the above errors and erasure decoding algorithm to design decoding algorithms for certain concatenated codes that can be decoded up to half their design distance (i.e. up to $Dd/2$).

11.3 Generalized Minimum Distance Decoding

Recall the natural decoding algorithm for concatenated codes from Algorithm 7. In particular, we performed MLD on the inner code and then fed the resulting vector to a unique decoding algorithm for the outer code. A drawback of this algorithm is that it does not take into account the information that MLD provides. For example, it does not distinguish between the situations where a given inner code's received word has a Hamming distance of one vs where the received word has a Hamming distance of (almost) half the inner code distance from the closest codeword. It seems natural to make use of this information. Next, we study an algorithm called the Generalized Minimum Distance (or GMD) decoder, which precisely exploits this extra information.

In the rest of the section, we will assume C_{out} to be an $[N,K,D]_{q^k}$ code that can be decoded (by $D_{C_{\text{out}}}$) from e errors and s erasures in polynomial time as long as $2e+s<D$. Further, let C_{in} be an $[n,k,d]_q$ code with $k=O(\log N)$ which has a unique decoder $D_{C_{\text{in}}}$ (which we will assume is the MLD implementation from Algorithm 1).

We will in fact look at three versions of the GMD decoding algorithm. The first two will be randomized algorithms while the last will be a deterministic algorithm. We will begin with the first randomized version, which will present most of the ideas in the final algorithm.

11.3.1 GMD algorithm- I

Before we state the algorithm, let us look at two special cases of the problem to build some intuition.

Consider the received word $\mathbf{y}=(y_1,\ldots,y_N)\in[q^n]^N$ with the following special property: for every i such that $1\le i\le N$, either $y_i=y_i'$ or $\Delta(y_i,y_i')\ge d/2$, where $y_i'=MLD_{C_{\text{in}}}(y_i)$. Now we claim that if $\Delta(\mathbf{y},C_{\text{out}}\circ C_{\text{in}})<dD/2$, then there are $<D$ positions in $\mathbf{y}$ such that $\Delta(y_i,C_{\text{in}}(y_i'))\ge d/2$ (we call such a position *bad*). This is because, for every bad position i, by the definition of y_i', $\Delta(y_i,C_{\text{in}})\ge d/2$. Now if there are $\ge D$ bad positions, this implies that $\Delta(\mathbf{y},C_{\text{out}}\circ C_{\text{in}})\ge dD/2$, which is a contradiction. Now note that we can decode $\mathbf{y}$ by just declaring an erasure at every bad position and running the erasure decoding algorithm for C_{out} on the resulting vector.

Now consider the received word $\mathbf{y}=(y_1,\ldots,y_N)$ with the special property: for every i such that $i\in[N]$, $y_i\in C_{\text{in}}$. In other words, if there is an error at position $i\in[N]$, then a valid codeword in C_{in} gets mapped to another valid codeword $y_i\in C_{\text{in}}$. Note that this implies that a position with error has at least d errors. By a counting argument similar to the ones used in the previous paragraph, we have that there can be $<D/2$ such error positions. Note that we can now decode $\mathbf{y}$ by essentially running a unique decoder for C_{out} on $\mathbf{y}$ (or more precisely on $(x_1,\ldots,x_N)$, where $y_i=C_{\text{in}}(x_i)$).

Algorithm 8 generalizes these observations to decode arbitrary received words. In particular, it smoothly “interpolates" between the two extreme scenarios considered above.

Note that if $\mathbf{y}$ satisfies one of the two extreme scenarios considered earlier, then Algorithm 8 works exactly the same as discussed above.

By our choice of $D_{C_{\text{out}}}$ and $D_{C_{\text{in}}}$, it is easy to see that Algorithm 8 runs in polynomial time (in the final block length). More importantly, we will show that the final (deterministic) version of

Algorithm 8 Generalized Minimum Decoder (ver 1)

```
INPUT: Received word $\mathbf{y} = \left(y_1, \cdots, y_N\right) \in \left[q^n\right]^N$
OUTPUT: Message $\mathbf{m}' \in \left[q^k\right]^K$

FOR $1 \le i \le N$ DO
    $y'_i \leftarrow D_{C_{\text{in}}}(y_i)$.
    $w_i \leftarrow \min\left(\Delta(y'_i, y_i), \frac{d}{2}\right)$.
    With probability $\frac{2w_i}{d}$, set $y''_i \leftarrow ?$, otherwise set $y''_i \leftarrow x$, where $y'_i = C_{\text{in}}(x)$.
$\mathbf{m}' \leftarrow D_{C_{\text{out}}}(\mathbf{y}'')$, where $\mathbf{y}'' = (y''_1, \ldots, y''_N)$.
RETURN $\mathbf{m}'$
```

Algorithm 8 can do unique decoding of $C_{\text{out}} \circ C_{\text{in}}$ up to half of its design distance.

As a first step, we will show that in expectation, Algorithm 8 works.

Lemma 11.3.1. *Let* $\mathbf{y}$ *be a received word such that there exists a codeword* $C_{\text{out}} \circ C_{\text{in}}(\mathbf{m}) = (c_1, \ldots, c_N) \in [q^n]^N$ *such that* $\Delta(C_{\text{out}} \circ C_{\text{in}}(\mathbf{m}), \mathbf{y}) < \frac{Dd}{2}$. *Further, if* $\mathbf{y}''$ *has* e' *errors and* s' *erasures (when compared with* $C_{\text{out}} \circ C_{\text{in}}(\mathbf{m})$*), then*

$$\mathbb{E}\left[2e' + s'\right] < D.$$

Note that if $2e' + s' < D$, then by Theorem 11.2.2, Algorithm 8 will output $\mathbf{m}$. The lemma above says that in expectation, this is indeed the case.

Proof of Lemma 11.3.1. For every $1 \le i \le N$, define $e_i = \Delta(y_i, c_i)$. Note that this implies that

$$\sum_{i=1}^{N} e_i < \frac{Dd}{2}. \tag{11.2}$$

Next for every $1 \le i \le N$, we define two indicator variables:

$$X_i^? = \mathbb{1}_{y''_i = ?},$$

and

$$X_i^e = \mathbb{1}_{C_{\text{in}}(y''_i) \ne c_i \text{ and } y''_i \ne ?}.$$

We claim that we are done if we can show that for every $1 \le i \le N$:

$$\mathbb{E}\left[2X_i^e + X_i^?\right] \le \frac{2e_i}{d}. \tag{11.3}$$

Indeed, by definition we have: $e' = \sum_i X_i^e$ and $s' = \sum_i X_i^?$. Further, by the linearity of expectation (Proposition 3.1.2), we get

$$\mathbb{E}\left[2e' + s'\right] \le \frac{2}{d} \sum_i e_i < D,$$

where the inequality follows from (11.2).

To complete the proof, we will prove (11.3) by a case analysis. Towards this end, fix an arbitrary $1 \le i \le N$.

Case 1: ($c_i = y'_i$) First, we note that if $y''_i \ne ?$ then since $c_i = y'_i$, we have $X^e_i = 0$. This along with the fact that $\Pr[y''_i = ?] = \frac{2w_i}{d}$ implies

$$\mathbb{E}\left[X^?_i\right] = \Pr[X^?_i = 1] = \frac{2w_i}{d},$$

and

$$\mathbb{E}\left[X^e_i\right] = \Pr[X^e_i = 1] = 0.$$

Further, by definition we have

$$w_i = \min\left(\Delta(y'_i, y_i), \frac{d}{2}\right) \le \Delta(y'_i, y_i) = \Delta(c_i, y_i) = e_i.$$

The three relations above prove (11.3) for this case.

Case 2: ($c_i \ne y'_i$) As in the previous case, we still have

$$\mathbb{E}\left[X^?_i\right] = \frac{2w_i}{d}.$$

Now in this case, if an erasure is not declared at position i, then $X^e_i = 1$. Thus, we have

$$\mathbb{E}\left[X^e_i\right] = \Pr[X^e_i = 1] = 1 - \frac{2w_i}{d}.$$

Next, we claim that as $c_i \ne y'_i$,

$$e_i + w_i \ge d, \tag{11.4}$$

which implies

$$\mathbb{E}\left[2X^e_i + X^?_i\right] = 2 - \frac{2w_i}{d} \le \frac{2e_i}{d},$$

as desired.

To complete the proof, we show (11.4) via yet another case analysis.

Case 2.1: ($w_i = \Delta(y'_i, y_i) < d/2$) By definition of e_i, we have

$$e_i + w_i = \Delta(y_i, c_i) + \Delta(y'_i, y_i) \ge \Delta(c_i, y'_i) \ge d,$$

where the first inequality follows from the triangle inequality and the second inequality follows from the fact that C_{in} has distance d.

Case 2.2: ($w_i = \frac{d}{2} \le \Delta(y'_i, y_i)$) As y'_i is obtained from MLD, we have

$$\Delta(y'_i, y_i) \le \Delta(c_i, y_i).$$

This along with the assumption on $\Delta(y'_i, y_i)$, we get

$$e_i = \Delta(c_i, y_i) \ge \Delta(y'_i, y_i) \ge \frac{d}{2}.$$

This in turn implies that

$$e_i + w_i \ge d,$$

as desired. □

11.3.2 GMD Algorithm- II

Note that Step 4 in Algorithm 8 uses “fresh" randomness for each i. Next we look at another randomized version of the GMD algorithm that uses the *same* randomness for every i. In particular, consider Algorithm 9.

Algorithm 9 Generalized Minimum Decoder (ver 2)

```
INPUT: Received word $\mathbf{y} = (y_1, \cdots, y_N) \in [q^n]^N$
OUTPUT: Message $\mathbf{m}' \in [q^k]^K$

1: Pick $\theta \in [0,1]$ uniformly at random.
2: FOR $1 \le i \le N$ DO
3:     $y'_i \leftarrow D_{C_{\text{in}}}(y_i)$.
4:     $w_i \leftarrow \min\left(\Delta(y'_i, y_i), \frac{d}{2}\right)$.
5:     If $\theta < \frac{2w_i}{d}$, set $y''_i \leftarrow ?$, otherwise set $y''_i \leftarrow x$, where $y'_i = C_{\text{in}}(x)$.
6: $\mathbf{m}' \leftarrow D_{C_{\text{out}}}(\mathbf{y}'')$, where $\mathbf{y}'' = (y''_1, \ldots, y''_N)$.
7: RETURN $\mathbf{m}'$
```

We note that in the proof of Lemma 11.3.1, we only use the randomness to show that

$$\Pr\left[y''_i = ?\right] = \frac{2w_i}{d}.$$

In Algorithm 9, we note that

$$\Pr\left[y''_i = ?\right] = \Pr\left[\theta \in \left[0, \frac{2w_i}{d}\right)\right] = \frac{2w_i}{d},$$

as before (the last equality follows from our choice of θ). One can verify that the proof of Lemma 11.3.1 can be used to show the following lemma:

Lemma 11.3.2. *Let* $\mathbf{y}$ *be a received word such that there exists a codeword* $C_{\text{out}} \circ C_{\text{in}}(\mathbf{m}) = (c_1, \ldots, c_N) \in [q^n]^N$ *such that* $\Delta(C_{\text{out}} \circ C_{\text{in}}(\mathbf{m}), \mathbf{y}) < \frac{Dd}{2}$. *Further, if* $\mathbf{y}''$ *has* e' *errors and* s' *erasures (when compared with* $C_{\text{out}} \circ C_{\text{in}}(\mathbf{m})$*), then*

$$\mathbb{E}_\theta\left[2e' + s'\right] < D.$$

Next, we will see that Algorithm 9 can be easily “derandomized."

11.3.3 Derandomized GMD algorithm

Lemma 11.3.2 along with the probabilistic method shows that there exists a value $\theta^* \in [0,1]$ such that Algorithm 9 works correctly even if we fix θ to be θ^* in Step 1. Obviously we can obtain such a θ^* by doing an exhaustive search for θ. Unfortunately, there are uncountable choices of θ because $\theta \in [0,1]$. However, this problem can be taken care of by the following discretization trick.

Define $Q = \{0,1\} \cup \{\frac{2w_1}{d}, \cdots, \frac{2w_N}{d}\}$. Then because for each i, $w_i = \min(\Delta(y'_i, y_i), d/2)$, we have

$$Q = \{0,1\} \cup \{q_1, \cdots, q_m\}$$

where $q_1 < q_2 < \cdots < q_m$ for some $m \le \left\lfloor \frac{d}{2} \right\rfloor$. Notice that for every $\theta \in [q_i, q_{i+1})$, just before Step 6, Algorithm 9 computes the same $\mathbf{y}''$. (See Figure 11.2 for an illustration as to why this is the case.)

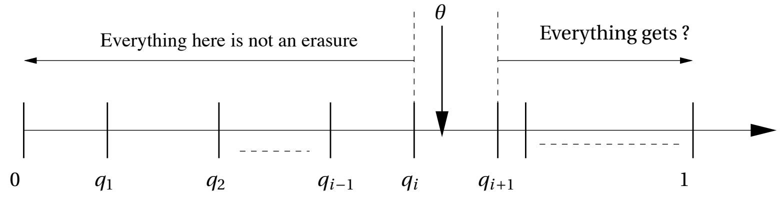


Figure 11.2: All values of $\theta \in [q_i, q_{i+1})$ lead to the same outcome

Thus, we need to cycle through all possible values of $\theta \in Q$, leading to Algorithm 10.

Algorithm 10 Deterministic Generalized Minimum Decoder‘

INPUT: Received word $\mathbf{y} = (y_1, \cdots, y_N) \in [q^n]^N$

OUTPUT: Message $\mathbf{m}' \in [q^k]^K$

```
Q ← {2w_1/d, ···, 2w_N/d} ∪ {0,1}.
FOR θ ∈ Q DO
    FOR 1 ≤ i ≤ N DO
        $y'_i \leftarrow D_{C_{\text{in}}}(y_i)$.
        $w_i \leftarrow \min\left(\Delta(y'_i, y_i), \frac{d}{2}\right)$.
        If $\theta < \frac{2w_i}{d}$, set $y''_i \leftarrow ?$, otherwise set $y''_i \leftarrow x$, where $y'_i = C_{\text{in}}(x)$.
    $\mathbf{m}'_\theta \leftarrow D_{C_{\text{out}}}(\mathbf{y}'')$, where $\mathbf{y}'' = (y''_1, \ldots, y''_N)$.
RETURN $\mathbf{m}'_{\theta^*}$ for $\theta^* = \arg\min_{\theta \in Q} \Delta\left(C_{\text{out}} \circ C_{\text{in}}\left(\mathbf{m}'_\theta\right), \mathbf{y}\right)$
```

Note that Algorithm 10 is Algorithm 9 repeated $|Q|$ times. Since $|Q|$ is $O(n)$, this implies that Algorithm 10 runs in polynomial time. This along with Theorem 9.2.1 implies that

Theorem 11.3.3. *For every constant rate, there exists an explicit linear binary code on the Zyablov bound. Further, the code can be decoded up to half of the Zyablov bound in polynomial time.*

Note that the above answers Question 9.3.1 in the affirmative.

11.4 Bibliographic Notes

Forney in 1966 designed the Generalized Minimum Distance (or GMD) decoding [13].