

3

General Methods of Decoding of Linear Codes

In this chapter we consider the general methods of linear codes decoding, i.e. the decoding methods that do not need any special properties of codes, except linearity. We start with minimum distance (Hamming distance) decoding, which coincides with the maximum likelihood decoding for channels with independent errors, then we consider more comprehensive methods of decoding linear codes.

3.1 MINIMUM DISTANCE DECODING

Usually we take the words minimum distance decoding to mean the procedure of searching for the codeword that is the closest one to the received word. Such procedures can be realised by an exhaustive search on the set of codewords or on the set of syndromes of probable error vectors.

The exhaustive search algorithm on the set of codewords consists of comparing the received word with all codewords and choosing of the closest codeword.

The exhaustive search algorithm on the set of syndromes of error vectors can be realised if we have table T, in which syndromes correspond to coset leaders. The algorithm is as follows:

1. Calculation of the syndrome \mathbf{s} of the received word \mathbf{b} ;
2. The search of the syndrome \mathbf{s} and the corresponding coset leader \mathbf{e} in the table T;
3. Calculation of decoded vector $\hat{\mathbf{a}}$:

$$\hat{\mathbf{a}} = \mathbf{b} + \mathbf{e}.$$

The first algorithm usually needs more operations, which is related to the necessity to generate all codewords (to execute q^k encoding procedures) and to compare them with the received word (q^k comparisons). The decoding on the syndromes requires keeping in memory the table with q^r words of length n .

It is usually difficult to implement these algorithms in practice due to their high complexity. Now we will consider less complex algorithms which, however, cannot provide full decoding, i.e. the decoding of all errors, which can be corrected with the help of given code (all coset leaders).

3.2 INFORMATION SET DECODING

The key concept for many general methods of decoding is the concept of the *information set* of the code. It is known that for systematic code first k symbols a_0, \dots, a_{k-1} fully define the word $a_0, \dots, a_{k-1}, a_k, \dots, a_{n-1}$ of the (n, k) code. That means there is only one word in the code where the symbols a_0, \dots, a_{k-1} occupy the positions $\{0, \dots, k-1\}$.

However, the set of positions $\{0, \dots, k-1\}$ is not a unique one. For every code there exists many other sets $\{j_1, \dots, j_k\}$, $(0 \leq j_1 < \dots < j_k \leq n-1)$, such that for any a_{j_1}, \dots, a_{j_k} there exists only one word in the code where the symbols a_{j_1}, \dots, a_{j_k} occupy the positions $\{j_1, \dots, j_k\}$. Hence, the symbols on the positions $\{j_1, \dots, j_k\}$ also fully define the codeword.

Definition 3.1 The set of positions $\{j_1, \dots, j_k\}$, $(0 \leq j_1 < \dots < j_k \leq n-1)$ is called the information set of the code V if the symbols a_{j_1}, \dots, a_{j_k} uniquely define the codeword from V .

Let \mathbf{G} be the generator matrix of the code V . Let us denote by $\mathbf{G}(\gamma)$ the matrix constructed from the columns of \mathbf{G} enumerated by the elements of set γ . It is obvious that set γ is the information set if, and only if, the mapping $f_\gamma: V \rightarrow A^k$, which put in correspondence to the codeword its coordinates with numbers from γ , is one-to-one mapping. This fact as was shown in chapter 2, is equivalent to the nonsingularity of matrix of mapping f_γ , which is the matrix $\mathbf{G}(\gamma)$ in the basis $\{\mathbf{v}_1, \dots, \mathbf{v}_k\}$. Thus, the following statement is correct:

Lemma 3.1 The set of positions $\gamma = \{j_1, \dots, j_k\}$ is the information set if, and only if, the matrix $\mathbf{G}(\gamma)$ is nonsingular.

If in the received erroneous word there is at least one information set without erroneous symbols (i.e. symbols with indexes from this information set do not contain errors), then the transmitted word can be restored on the basis of this information set. In this case the decoding procedure can be regarded as the search of an information set that is free of errors. As this takes place, the issue of the 'stop rule' is very important, i.e. we should choose the rule according to whether it is possible to identify that the information set free of errors is found. Hereafter we will consider the decoding of t -fold errors.

Let us describe now the decoding algorithm based on the information sets. Let $\gamma = \{j_1, \dots, j_k\}$ be the information set of the code V , and let \mathbf{G} and \mathbf{H} be the generator and the parity matrix of this code. Let us denote by \mathbf{G}_γ the matrix

$$\mathbf{G}_\gamma = (\mathbf{G}(\gamma))^{-1} \cdot \mathbf{G}. \quad (3.1)$$

It is obvious that the columns of matrix \mathbf{G}_γ with numbers $\{j_1, \dots, j_k\}$ form the identity $(k \times k)$ -matrix. Multiplying the vector $(a_{j_1}, \dots, a_{j_k})$ by matrix \mathbf{G}_γ results in the codeword with symbols a_{j_1}, \dots, a_{j_k} on the positions $\{j_1, \dots, j_k\}$:

$$\begin{aligned} (a_{j_1}, \dots, a_{j_k}) \cdot \mathbf{G}_\gamma &= (a_{j_1}, \dots, a_{j_k}) \cdot \begin{bmatrix} & j_1 & & j_2 & & j_k & & \\ \dots & 1 & \dots & 0 & \dots & 0 & \dots & \\ \dots & 0 & \dots & 1 & \dots & 0 & \dots & \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \\ \dots & 0 & \dots & 0 & \dots & 1 & \dots & \end{bmatrix} \\ &= \begin{pmatrix} & j_1 & & j_2 & & j_k & & \\ \dots & a_{j_1}, & \dots, & a_{j_2}, & \dots, & a_{j_k}, & \dots & \end{pmatrix} \end{aligned}$$

Let us put matrix \mathbf{G}_γ in correspondence to parity matrix

$$\mathbf{H}_\gamma = (\mathbf{H}(\bar{\gamma}))^{-1} \cdot \mathbf{H}, \quad (3.2)$$

where $\bar{\gamma} = \{1, 2, \dots, n\} \setminus \gamma$, i.e. the set of positions that are not included in γ , and $\mathbf{H}(\bar{\gamma})$ is the matrix formed by the columns of matrix \mathbf{H} with indexes from $\bar{\gamma}$. The columns of matrix \mathbf{H}_γ with indexes not included in γ form the identity $(r \times r)$ -matrix. Hence, if all nonzero elements of error vector \mathbf{e} are located in the set $\bar{\gamma}$ (i.e. γ is free of errors), then the weight of the syndrome

$$\mathbf{s}_\gamma(\mathbf{e}) = \mathbf{e} \cdot \mathbf{H}_\gamma^T, \quad (3.3)$$

is equal to weight of error vector. Therefore, the algorithm of the information set decoding of ν -fold errors, $\nu \leq t = \left\lfloor \frac{d-1}{2} \right\rfloor$, can be formulated as follows:

Let $\Gamma = \{\gamma_1, \gamma_2, \dots, \gamma_l\}$ be the set of information sets of the code. Let us assume that the set Γ contains a reasonable number of information sets to correct of ν -fold errors.

1. Calculation of the syndromes $\mathbf{s}_{\gamma_i}(\mathbf{b})$, where $\mathbf{b} = (b_0, b_1, \dots, b_{n-1})$ is the received vector, $\mathbf{b} = \mathbf{a} + \mathbf{e}$, \mathbf{a} is the transmitted vector and \mathbf{e} is the error vector, until information set $\gamma = \{j_1, j_2, \dots, j_k\}$ is found such that the weight of the corresponding syndrome

$$w(\mathbf{s}_\gamma(\mathbf{b})) \leq \nu. \quad (3.4)$$

2. If the condition (3.4) is satisfied the codeword $\hat{\mathbf{a}}$ is regarded as the decoded word.

$$\hat{\mathbf{a}} = \mathbf{b}(\gamma) \cdot \mathbf{G}_\gamma = (b_{j_1}, b_{j_2}, \dots, b_{j_k}) \cdot \mathbf{G}_\gamma. \quad (3.5)$$

3. If none of the information sets γ_i satisfy the condition (3.4) the calculation (3.5) is not executed and it is assumed that an uncorrectable error is detected.

Example 3.1 Consider the decoding of the binary (7, 4) code with $d = 3$. Let the generator matrix \mathbf{G} and the parity matrix \mathbf{H} be as follows

$$\mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}; \quad \mathbf{H} = \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix}. \quad (3.6)$$

The set of symbols $\gamma' = \{0, 1, 2, 6\}$ is the information set of the code and the set $\gamma'' = \{0, 1, 2, 4\}$ is not the information set of the code because the determinant of the matrix $\mathbf{G}(\gamma')$

$$|\mathbf{G}(\gamma')| = \begin{vmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{vmatrix} \neq 0$$

and the determinant of the matrix $\mathbf{G}(\gamma'')$

$$|\mathbf{G}(\gamma'')| = \begin{vmatrix} 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 \end{vmatrix} = 0.$$

In the same way we can verify that $\gamma^{(0)} = \{0, 1, 2, 3\}$, $\gamma^{(1)} = \{3, 4, 5, 6\}$, $\gamma^{(2)} = \{0, 1, 2, 6\}$ are the information sets. For these information sets $\mathbf{H}_{\gamma^{(0)}} = \mathbf{H}$, $\mathbf{G}_{\gamma^{(0)}} = \mathbf{G}$;

$$\begin{aligned} \mathbf{H}_{\gamma^{(1)}} &= (\mathbf{H}(0, 1, 2))^{-1} \cdot \mathbf{H} = \begin{bmatrix} 1 & 1 & 1 \\ 0 & 1 & 1 \\ 1 & 1 & 0 \end{bmatrix}^{-1} \cdot \mathbf{H} = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 1 & 1 \\ 1 & 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 & 0 & 1 & 0 \\ 1 & 1 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} \\ &= \begin{bmatrix} 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 & 0 & 1 \end{bmatrix}, \end{aligned}$$

$$\mathbf{G}_{\gamma^{(1)}} = (\mathbf{G}(3, 4, 5, 6))^{-1} \cdot \mathbf{G} = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix};$$

$$\mathbf{H}_{\gamma^{(2)}} = (\mathbf{H}(3, 4, 5))^{-1} \cdot \mathbf{H} = \begin{bmatrix} 1 & 1 & 0 & 1 & 0 & 0 & 1 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \end{bmatrix};$$

$$\mathbf{G}_{\gamma^{(2)}} = (\mathbf{G}(0, 1, 2, 6))^{-1} \cdot \mathbf{G} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

The set of information sets $\Gamma = \{\gamma^{(0)}, \gamma^{(1)}, \gamma^{(2)}\}$ allows decoding any 1-fold errors in (7,4) code. Let $\mathbf{a} = (0, 0, 0, 0, 0, 0, 0)$ be the transmitted word and $\mathbf{b} = (0, 0, 0, 1, 0, 0, 0)$ be the received word. The decoding procedure in accordance with the algorithm described above is as follows:

$$\begin{aligned} \mathbf{s}_{\gamma^{(0)}}(\mathbf{b}) &= \mathbf{b} \cdot \mathbf{H}_{\gamma^{(0)}}^T = (0, 1, 1) \Rightarrow w(\mathbf{s}_{\gamma^{(0)}}(\mathbf{b})) = 2 > 1 = \frac{d-1}{2}; \\ \mathbf{s}_{\gamma^{(1)}}(\mathbf{b}) &= \mathbf{b} \cdot \mathbf{H}_{\gamma^{(1)}}^T = (1, 0, 1) \Rightarrow w(\mathbf{s}_{\gamma^{(1)}}(\mathbf{b})) = 2 > 1 = \frac{d-1}{2}; \\ \mathbf{s}_{\gamma^{(2)}}(\mathbf{b}) &= \mathbf{b} \cdot \mathbf{H}_{\gamma^{(2)}}^T = (1, 0, 0) \Rightarrow w(\mathbf{s}_{\gamma^{(2)}}(\mathbf{b})) = 1 = \frac{d-1}{2}. \end{aligned}$$

With the help of the information set $\gamma^{(2)}$ we can calculate the decoded word

$$\hat{\mathbf{a}} = (b_0, b_1, b_2, b_6) \cdot \mathbf{G}_{\gamma^{(2)}} = (0000) \cdot \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 \end{bmatrix} = (0000000) = \mathbf{a}.$$

The cardinal number of set Γ , i.e. the number of information sets, which is required for decoding generally increases very quickly with the increasing of the code length and the number of correctable errors. Since each information set γ from Γ needs in keeping or calculating matrices \mathbf{G}_γ and (or) \mathbf{H}_γ , then the complexity of the algorithm increases. The simplification of the information set decoding is associated with two modifications of this algorithm - *permutation decoding* and decoding with the help of *covering polynomials* (or *covering-set decoding*).

As was defined above, the permutation π ($\pi \in S_n$) is the one-to-one self-mapping of set $\{1, 2, \dots, n\}$. Each permutation $\pi \in S_n$ corresponds to the linear operator on space A^n , i.e. $\pi(a_1, \dots, a_n) = (a_{\pi(1)}, \dots, a_{\pi(n)})$. Arbitrary permutation transfers the word of the code V to some other word (normally this word does not belong to the code V). However, there exist some permutations that transfer any codeword to the codeword of the same code. Such permutations are said to be preserving the code permutation, and the code is said to be invariant relative to this permutation. It is easy to verify that the set of permutations preserving the code V forms the subgroup in the group S_n of all permutations. This subgroup is denoted as $\text{Aut } V$.

Example 3.2 Consider the binary linear (3, 2)-code consisting of 4 words (000), (110), (100), (010). The permutation of the first and the second symbol of any codeword transfers it to the codeword, but the permutation of the second and the third symbol transfers the codeword (110) to the word (101), which does not belong to the code.

Let \mathbf{G} be the generator matrix and \mathbf{H} be the parity matrix of the code V

$$\mathbf{G} = \begin{bmatrix} \mathbf{g}_1 \\ \vdots \\ \mathbf{g}_k \end{bmatrix}; \quad \mathbf{H} = \begin{bmatrix} \mathbf{h}_1 \\ \vdots \\ \mathbf{h}_r \end{bmatrix}.$$

It is a necessary condition for some permutation π to preserve the code V if matrix

$$\pi(\mathbf{G}) = \begin{bmatrix} \pi(\mathbf{g}_1) \\ \vdots \\ \pi(\mathbf{g}_k) \end{bmatrix}$$

satisfies the following equation

$$\pi(\mathbf{G}) \cdot \mathbf{H}^T = \mathbf{0}.$$

Let π be the permutation preserving the code V . Then for any vector $\mathbf{b} = \mathbf{a} + \mathbf{e}$, $\mathbf{a} \in V$,

$$\pi(\mathbf{b}) = \pi(\mathbf{a}) + \pi(\mathbf{e}) = \mathbf{a}' + \mathbf{e}' ,$$

where \mathbf{a}' is some codeword and \mathbf{e}' is the error vector, which has the same weight as vector \mathbf{e} . If the weight of vector \mathbf{e} is no more than t , then the weight of vector \mathbf{e}' also is no more than t .

With the help of permutations preserving the code, information set decoding can be realised as follows. Let \mathbf{G} be the generator matrix and \mathbf{H} be the parity matrix of the code V , and let both these matrices be in systematic form, i.e. corresponding to the information set $\gamma^{(0)} = \{0, 1, \dots, k-1\}$. Let $\text{Aut } V = \{\pi_1, \dots, \pi_l\}$ be the set of permutations preserving the code. Let us calculate the syndromes of vectors $\pi_i(\mathbf{b})$ with the help of information set $\gamma^{(0)}$

$$\mathbf{s}(\pi_i(\mathbf{b})) = \pi_i(\mathbf{b}) \cdot \mathbf{H}^T, \quad (3.7)$$

then we calculate the weight of syndromes (3.7). If some permutation π transfers vector \mathbf{e} to vector \mathbf{e}' , where all nonzero components are located on the positions $\{k, \dots, n-1\}$, then the information set $\gamma^{(0)}$ is free of errors, and the weight of the corresponding syndrome is no more than t . In this case it is sufficient to use the permutation π^{-1} to vector \mathbf{a}' to restore the transmitted codeword.

Example 3.3 Consider the permutation decoding of (7, 4)-code with $d = 3$, which was defined in the Example 3.1. This code is invariant relative to the cyclic permutation T , $T(i) = (i + 1) \bmod 7$. Really,

$$T(\mathbf{G}) = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 1 \end{bmatrix},$$

$$T(\mathbf{G}) \cdot \mathbf{H}^T = \mathbf{0}.$$

Let us decode the word $\mathbf{b} = \mathbf{a} + \mathbf{e} = (1011000) + (0100000) = (1111000)$. Let us calculate

$$\mathbf{s}_{\gamma^{(0)}}(T^i(\mathbf{b})) \text{ for } i = 0, 1, \dots, 6 \text{ since } T^7 = I = T^0.$$

$$\mathbf{s}_{\gamma^{(0)}}(\mathbf{b}) = \mathbf{b} \cdot \mathbf{H}^T = (111) \Rightarrow w(\mathbf{s}_{\gamma^{(0)}}(\mathbf{b})) = 3 > 1 = \frac{d-1}{2};$$

$$\mathbf{s}_{\gamma^{(0)}}(T(\mathbf{b})) = T(\mathbf{b}) \cdot \mathbf{H}^T = (110) \Rightarrow w(\mathbf{s}_{\gamma^{(0)}}(T(\mathbf{b}))) = 2 > 1 = \frac{d-1}{2};$$

$$\mathbf{s}_{\gamma^{(0)}}(T^2(\mathbf{b})) = T^2(\mathbf{b}) \cdot \mathbf{H}^T = (011) \Rightarrow w(\mathbf{s}_{\gamma^{(0)}}(T^2(\mathbf{b}))) = 2 > 1 = \frac{d-1}{2};$$

$\mathbf{s}_{\gamma^{(0)}}(T^3(\mathbf{b})) = T^3(\mathbf{b}) \cdot \mathbf{H}^T = (100) \Rightarrow w(\mathbf{s}_{\gamma^{(0)}}(T^3(\mathbf{b}))) = 1 = \frac{d-1}{2}$, and the condition (3.4) is satisfied. With the help of information set $\gamma^{(0)}$ calculate the word $\mathbf{a}' = T^3(\hat{\mathbf{a}})$, $\hat{\mathbf{a}}$ is the decoded word

$$\mathbf{a}' = T^3(\hat{\mathbf{a}}) = (0001) \cdot \mathbf{G} = (0001011),$$

and

$$\hat{\mathbf{a}} = T^{-3}(\mathbf{a}') = (1011000) = \mathbf{a}.$$

Another way of ‘clearing’ the information set from errors is the covering of errors in the information set. This method is called decoding with the help of covering polynomials¹.

Let θ be the vector (covering polynomial), which coincides with error vector \mathbf{e} on the positions of the information set γ and with zeroes on the other positions. Then for vector $(\mathbf{b} - \theta)$ the information set γ is free of errors, and the weight of the corresponding syndrome

$$\mathbf{s}_{\gamma}(\mathbf{b} - \theta) = (\mathbf{e} - \theta) \cdot \mathbf{H}^T$$

is

$$w(\mathbf{s}_{\gamma}(\mathbf{b} - \theta)) \leq t - w(\theta). \tag{3.8}$$

If we search vectors θ in increasing order of their weights (starting with $\theta_0 = (0 \dots 0)$) until some θ^* satisfy (3.8), then it will be possible to restore the transmitted vector with the help of vector $\mathbf{b}^* = (\mathbf{b} - \theta^*)$ and the information set γ (of course, if the weight of error vector does not exceed t).

Example 3.4 Let us use decoding with the help of covering polynomials for the case considered in Example 3.1. Let the set of covering polynomials be $\theta_0 = (0000000)$, $\theta_1 = (1000000)$, $\theta_2 = (0100000)$, $\theta_3 = (0010000)$, $\theta_4 = (0001000)$. Consider the

¹The term ‘covering polynomial’ is well-established in coding theory but is not exactly correct. The proper term is ‘covering vector’ ‘covering word’.

information set $\gamma^{(0)}$. Let the received vector be $\mathbf{b} = (0001000) = \mathbf{a} + \mathbf{e} = (0000000) + (0001000)$. Calculate the syndromes

$$\begin{aligned} \mathbf{s}_{\gamma^{(0)}}(\mathbf{b} - \theta_0) &= (0001000) \cdot \mathbf{H}^T = (011) \Rightarrow w(\mathbf{s}_{\gamma^{(0)}}(\mathbf{b} - \theta_0)) = 2 > 1 = \frac{d-1}{2}; \\ \mathbf{s}_{\gamma^{(0)}}(\mathbf{b} - \theta_1) &= (1001000) \cdot \mathbf{H}^T = (110) \Rightarrow w(\mathbf{s}_{\gamma^{(0)}}(\mathbf{b} - \theta_1)) = 2 > 1 = \frac{d-1}{2} - w(\theta_1); \\ \mathbf{s}_{\gamma^{(0)}}(\mathbf{b} - \theta_2) &= (0101000) \cdot \mathbf{H}^T = (100) \Rightarrow w(\mathbf{s}_{\gamma^{(0)}}(\mathbf{b} - \theta_2)) = 1 > 0 = \frac{d-1}{2} - w(\theta_2); \\ \mathbf{s}_{\gamma^{(0)}}(\mathbf{b} - \theta_3) &= (0011000) \cdot \mathbf{H}^T = (101) \Rightarrow w(\mathbf{s}_{\gamma^{(0)}}(\mathbf{b} - \theta_3)) = 2 > 0 = \frac{d-1}{2} - w(\theta_3); \\ \mathbf{s}_{\gamma^{(0)}}(\mathbf{b} - \theta_4) &= (0000000) \cdot \mathbf{H}^T = (000) \Rightarrow w(\mathbf{s}_{\gamma^{(0)}}(\mathbf{b} - \theta_4)) = 0 = \frac{d-1}{2} - w(\theta_4). \end{aligned}$$

Then $\theta^* = \theta_4$, $\mathbf{b}^* = (\mathbf{b} - \theta^*) = (0000000)$, and the transmitted word can be restored with the help of $\gamma^{(0)}$ and vector \mathbf{b}^* :

$$\hat{\mathbf{a}} = (b_0^*, b_1^*, \dots, b_{k-1}^*) \cdot \mathbf{G}_{\gamma^{(0)}} = (0000000) = \mathbf{a}.$$

The best results can be obtained with the joint use of the algorithms considered above. Let $\Gamma = \{\gamma^{(0)}, \dots, \gamma^{(m)}\}$ be the set of the information sets of code V and $\theta^{(0)}, \dots, \theta^{(m)}$ be the sets of covering polynomials corresponding to information sets $\gamma^{(0)}, \dots, \gamma^{(m)}$: $\theta^{(0)} = \{\theta_{00}, \dots, \theta_{0l_0}\}, \dots, \theta^{(m)} = \{\theta_{m0}, \dots, \theta_{ml_m}\}$ with $\theta_{j0} = \mathbf{0}$ and $w(\theta_{j0}) < w(\theta_{j1}) \leq \dots \leq w(\theta_{jl_j}), j = 0, \dots, m$. The decoding algorithm based on the joint use of information sets and covering polynomials is as follows:

Covering-Set Decoding:

1. Calculate the vector $\tilde{\mathbf{b}}_{ij} = \mathbf{b} - \theta_{ij}$ and the syndrome $\mathbf{s}_{\gamma^{(i)}}(\tilde{\mathbf{b}}_{ij}) = \tilde{\mathbf{b}}_{ij} \cdot \mathbf{H}^T$ for each pair $\gamma^{(i)}, \theta_{ij}$ ($i = 0, \dots, m; j = 0, \dots, l_j$) until the pair i^*, j^* will be found, such that the following condition is satisfied

$$w(\mathbf{s}_{\gamma^{(i^*)}}(\tilde{\mathbf{b}}_{i^*j^*} - \theta)) \leq t - w(\theta_{i^*j^*}). \quad (3.9)$$

2. If the condition (3.9) is satisfied, calculate

$$\hat{\mathbf{a}} = \tilde{\mathbf{b}}(\gamma^{(i^*)}) \cdot \mathbf{G}_{\gamma^{(i^*)}},$$

where $\tilde{\mathbf{b}}(\gamma^{(i^*)})$ is the subvector of vector $\tilde{\mathbf{b}}_{i^*j^*}$ combined from the elements of vector $\tilde{\mathbf{b}}_{i^*j^*}$, which belong to the information set $\gamma^{(i^*)}$

3. If any pair $\gamma^{(i)}, \theta_{ij}$ does not satisfy the condition (3.9), then it is assumed that the transmitted word was corrupted by the uncorrectable error.

The set of set Γ and sets of the covering polynomials $\Theta = \{\theta^{(0)}, \dots, \theta^{(m)}\}$ is called the decoding set and is denoted as $DS = \{\Gamma, \Theta\}$.

It is often convenient not to use all covering polynomials for decoding but only those with weight no more than ν , i.e. vectors with nonzero elements located on the positions of set γ , and the number of these nonzero elements does not exceed ν . Such kinds of vector we denote as $\theta_\gamma(\nu)$.

Example 3.5 Consider (7,4) code from the Example 3.1. Consider $\gamma_0 = \{0, 1, 2, 3\}$. The set of polynomials $\theta_{\gamma^{(0)}}(1)$ consists of five polynomials $\theta_{00} = (0000000)$, $\theta_{01} = (1000000)$, $\theta_{02} = (0100000)$, $\theta_{03} = (0010000)$, $\theta_{04} = (0001000)$.

It is easy to verify that if γ is the information set of the code V and π is the permutation preserving the code, then the set $\pi(\gamma)$ is also the information set of code V .

Example 3.6 Consider the decoding of the (15, 5) Hamming code with distance $d = 7$. It is possible to use the decoding set $DS = \{\Gamma, \Theta(1)\}$ to decode this code, where $\Gamma = \{\gamma^{(0)}, \gamma^{(1)} = T^5(\gamma^{(0)})\}$, $\gamma^{(0)} = \{0, 1, 2, 3, 4\}$, and $T^5(\gamma^{(0)}) = \{5, 6, 7, 8, 9\}$ is the cyclic shift of the information set $\gamma^{(0)}$ by 5 positions. Let the received word \mathbf{b} be $\mathbf{b} = \mathbf{a} + \mathbf{e} = (001110110010100) + (010100010000000)$, i.e. the received word is corrupted in the first, third and seventh position. Calculate $\mathbf{s}_{0j}(\mathbf{b}) = \tilde{\mathbf{b}}_{0j} \cdot \mathbf{H}_{\gamma^{(0)}}^T$ with the help of information set $\gamma^{(0)}$:

$$\begin{aligned} \tilde{\mathbf{b}}_{00} &= \mathbf{b} + \theta_{00} = \mathbf{b} + (000000000000000), \\ \mathbf{s}_{00}(\mathbf{b}) &= \tilde{\mathbf{b}}_{00} \cdot \mathbf{H}_{\gamma^{(0)}}^T = (0011110110) \Rightarrow w(\mathbf{s}_{00}(\mathbf{b})) = 6 > 3 = \frac{d-1}{2} - w(\theta_{00}); \\ \tilde{\mathbf{b}}_{01} &= \mathbf{b} + \theta_{01} = \mathbf{b} + (100000000000000), \\ \mathbf{s}_{01}(\mathbf{b}) &= \tilde{\mathbf{b}}_{01} \cdot \mathbf{H}_{\gamma^{(0)}}^T = (1101000100) \Rightarrow w(\mathbf{s}_{01}(\mathbf{b})) = 4 > 2 = \frac{d-1}{2} - w(\theta_{01}); \\ &\dots\dots\dots \\ \tilde{\mathbf{b}}_{05} &= \mathbf{b} + \theta_{05} = \mathbf{b} + (000010000000000), \\ \mathbf{s}_{05}(\mathbf{b}) &= \tilde{\mathbf{b}}_{05} \cdot \mathbf{H}_{\gamma^{(0)}}^T = (1110010011) \Rightarrow w(\mathbf{s}_{05}(\mathbf{b})) = 6 > 2 = \frac{d-1}{2} - w(\theta_{05}); \end{aligned}$$

Now let us use the information set $\gamma^{(1)} = T^5(\gamma^{(0)})$, and the corresponding matrices $\mathbf{G}_{\gamma^{(1)}} = T^5(\mathbf{G}_{\gamma^{(0)}})$ and $\mathbf{H}_{\gamma^{(1)}} = T^5(\mathbf{H}_{\gamma^{(0)}})$ are

$$\mathbf{G}_{\gamma^{(1)}} = \begin{bmatrix} 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$\mathbf{H}_{\gamma^{(1)}} = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}.$$

corresponding partition of the parity-check matrix \mathbf{H} . Any error vector $\mathbf{e} = (\mathbf{e}_l | \mathbf{e}_r)$ with $\mathbf{e} \cdot \mathbf{H}^T = \mathbf{e}_l \cdot \mathbf{H}_l^T + \mathbf{e}_r \cdot \mathbf{H}_r^T = \mathbf{s}$ is a plausible candidate for the decoding output. Assume, in addition, that the number of errors within the subset l equals u , where the numbers u and m are chosen in accordance with the natural restrictions $u \leq m$, $t - u \leq n - m$. For every possible m -vector \mathbf{e}_l , compute the product $\mathbf{s}_l = \mathbf{e}_l \cdot \mathbf{H}_l^T$ and store it, together with the vector \mathbf{e}_l , as an entry of the table X_l . Likewise, form the table X_r and look for a pair of entries $(\mathbf{s}_l, \mathbf{s}_r)$ that add up to the received syndrome \mathbf{s} . Therefore, for every given \mathbf{s}_r occurring in X_r , we should inspect X_l for the occurrence of $\mathbf{s} - \mathbf{s}_r$. One practical way to do this is to order X_l with respect to the entries \mathbf{s}_l .

However, in reality we know neither the number of errors nor their distribution. Therefore, we have to repeat the described procedure for several choices of m and u . In doing so, we may optimise the choice in order to reduce the total size of memory used for the tables X_l and X_r . For every choice of m there are no more than t different options for the choice of u . Hence, by repeatedly building the tables, though not more than nt times, we shall capture any distribution of t errors. Finally, the entire procedure should be repeated for all $t = 1, 2, \dots, d$ until we find the error vector that has the ‘received’ syndrome \mathbf{s} . Let us give a more formal description of the algorithm.

Split Syndrome Decoding:

Precomputation stage: For every weight t , $1 \leq t \leq d$ find the point m such that the tables X_l and X_r have an (almost) equal size. Store the pair (m, u) in the set $E(t)$.

1. Compute $\mathbf{s} = \mathbf{b} \cdot \mathbf{H}^T$ and set $t = 1$.
2. For every entry of $E(t)$, form the tables X_l and X_r as described.
3. Order X_l with respect to the entries \mathbf{s}_l .
4. For every entry of X_r check whether X_l contains the vector $\mathbf{s}_l = \mathbf{s} - \mathbf{s}_r$. If this is found, then output $\hat{\mathbf{a}} = \mathbf{b} - (\mathbf{e}_l | \mathbf{e}_r)$ and STOP.
5. Otherwise, set $t = t + 1$ and repeat Steps 2–5 while $t < d$.

3.3 A SUPERCODE DECODING ALGORITHM

A supercode decoding algorithm is based on the ideas illustrated in the previous section. A more detailed description of this algorithm can be found in [7]. The basic idea of the supercode algorithm is to combine lists of candidates obtained after decoding of several ‘supercodes’ of V , i.e., linear codes V' such that $V \subset V'$. We begin with an example that illustrates some of the ideas of the algorithm.

Example 3.7 Consider the $(48, 24, 12)$ Binary Extended QR Code. The aim is to construct a decoder that corrects five errors. Suppose the first 24 coordinates form an information set of the code. Since the code is self-dual, the last 24 coordinates also form an information set. The decoding algorithm consists of two stages, one for each of the two choices of information sets. We only explain one of them; the other is symmetric. Suppose the parity-check matrix \mathbf{H} is reduced to the form $[\mathbf{I}_{24} | \mathbf{A}]$. Let

$\mathbf{e} = (\mathbf{e}_l | \mathbf{e}_r)$ be an error vector of weight ≤ 5 , where \mathbf{e}_l and \mathbf{e}_r are the first and the second halves, respectively. At this stage we aim at correcting error vectors satisfying $wt(\mathbf{e}_l | \mathbf{e}_r) = (5, 0)$ or $(4, 1)$ or $(3, 2)$; the remaining possibilities will be covered by the second stage.

Let $\mathbf{b} = (\mathbf{b}_l | \mathbf{b}_r)$ be the received vector. First, the decoder assumes that $wt(\mathbf{e}_r) \leq 1$. There are 25 such error vectors. Each of them is subtracted from \mathbf{b}_r . The obtained vector is then encoded with the code and compared to \mathbf{b} . If the distance between them is less than or equal to 5, the decoding stops.

Otherwise, let $\mathbf{s} = \mathbf{b} \cdot \mathbf{H}^T$ be the received syndrome. Let \mathbf{H}_i , $1 \leq i \leq 4$, be the submatrix of \mathbf{H} formed by rows $6(i-1) + j$, $1 \leq j \leq 6$, and let $\mathbf{s}_i = \mathbf{b} \cdot \mathbf{H}_i^T$ be the corresponding part of the syndrome. Denote by \mathbf{A}_i the corresponding six rows of the matrix \mathbf{A} . The partition into submatrices defines a partition of the first 24 coordinates of the code into four parts

$$N_i = [6(i-1) + 1, 6(i-1) + 2, \dots, 6(i-1) + 6], \quad 1 \leq i \leq 4.$$

The syndrome \mathbf{s} is divided into four parts $\mathbf{s}_1, \mathbf{s}_2, \mathbf{s}_3, \mathbf{s}_4$ where part \mathbf{s}_i is formed by the sum of the columns in \mathbf{H}_i that correspond to the positions of errors. If a part, say N_1 , is error-free, then there is an error pattern \mathbf{e} with $wt(\mathbf{e}_r) \leq 2$ such that $\mathbf{e}_r \cdot \mathbf{A}_1^T = \mathbf{s}_1$. Any single error in N_1 affects one coordinate in the syndrome. Therefore, if N_1 contains one error, by inspecting all error patterns \mathbf{e}_r of weight ≤ 2 in the information part we shall find a syndrome $\mathbf{s}' = \mathbf{e}_r \cdot \mathbf{A}_1^T$ at a distance one from \mathbf{s}_1 . Therefore, this step can be accomplished as follows. For each i , $1 \leq i \leq 4$, we make a list of error patterns \mathbf{e}_r with $wt(\mathbf{e}_r) \leq 2$ that yield a syndrome \mathbf{s}' at a distance ≤ 1 from \mathbf{s}_i . An error pattern is a plausible candidate if it appears in three out of four lists.

Since we do not know the actual error distribution, we need to store four tables of error patterns for each of the two message sets. Each table consists of 64 records, one for each possible value of \mathbf{s}_i . Finally, each record is formed by all error patterns \mathbf{e}_r of weight 2 or less for which $dist(\mathbf{s}_i, \mathbf{e}_r \cdot \mathbf{A}_i^T) \leq 1$.

The decoding is repeated for each of the two information sets. For a given information set, we compile a list of error patterns that appear in three out of four tables T_i in the record corresponding to the received syndrome \mathbf{s}_i . Each error pattern is subtracted from the 24 coordinates of \mathbf{b} that correspond to the message part. The obtained message set is then encoded with the code. The decoding stops when we find a vector at a distance of at most 5 from \mathbf{b} .

The total size of memory used by tables T_i is 8 kbytes. The decoding requires about 3000 operations with binary vectors of length 24 and seems to be the simplest known for this code. Note that the straightforward search over all error patterns of weight ≤ 2 in the message part would require about twice as many operations.

Let us now pass to the general case. The algorithm involves an (exponential) number of iterations. Each iteration is performed for a given information set with respect to V and consists of $O(n-k)$ steps of decoding different codes V' . Let $\hat{\mathbf{a}}$ be the current decision, which is updated in the course of the decoding. The initial value is set to 0.

First, we describe what happens after we fix an information set $\gamma \subset \{0, 1, \dots, n-1\}$ with respect to the code V . Let \mathbf{H} be an $((n-k) \times n)$ parity-check matrix of V . Choose the basis of V in such a way that \mathbf{H} is diagonal on $\{0, 1, \dots, n-1\} \setminus \gamma$, i.e., $\mathbf{H} = [\mathbf{A} | \mathbf{I}_{n-k}]$. The

idea is to look for the part of the syndrome least affected by errors. Let y , $0 \leq y \leq n - k$ be an integer parameter whose value will be chosen later. Represent the matrix \mathbf{H} in the form

$$\mathbf{H} = \begin{bmatrix} \mathbf{A}_1 & \mathbf{I}_y & 0 \\ \mathbf{A}_2 & 0 & \mathbf{I}_{n-k-y} \end{bmatrix}.$$

Let \mathbf{b} be a received vector, i.e., $\mathbf{b} = \mathbf{a} + \mathbf{e}$, where $\mathbf{a} \in V$ is the closest codeword to \mathbf{b} . Isolate the first $k + y$ coordinates of V and denote by $V(y)$ the linear code orthogonal to $\mathbf{H}_y = [\mathbf{A}_1 | \mathbf{I}_y]$. Let \mathbf{s}_y be the syndrome of \mathbf{b} with respect to \mathbf{H}_y . Decoding in $V(y)$ amounts to solving the equation

$$\mathbf{u} \cdot \mathbf{H}_y^T = \mathbf{s}_y \quad (3.10)$$

with respect to the unknown vector \mathbf{u} . Suppose \mathbf{u} is represented as $\mathbf{u} = (\mathbf{u}_1 | \mathbf{u}_2)$, where \mathbf{u}_1 is a k -vector and \mathbf{u}_2 is a y -vector. Then in (3.10) we are looking for vectors satisfying

$$(\mathbf{u}_1 | \mathbf{u}_2) \cdot [\mathbf{A}_1 | \mathbf{I}_y]^T = \mathbf{s}_y \quad (3.11)$$

To build the list of solutions to this equation, we again use the split-syndrome algorithm. Suppose that we also know that $wt(\mathbf{u}_1) \leq e_1$ and $wt(\mathbf{u}_2) \leq e_2$ (below we abbreviate this as $wt(\mathbf{u}_1 | \mathbf{u}_2) \leq (e_1 | e_2)$). This restriction allows us to reduce the size of the list of solutions to (3.11). Here e_1 and e_2 again are integer parameters whose values are chosen below. Partition the subset $\{0, 1, \dots, n - 1\} \setminus \gamma$ of size $n - k$ into $s = (n - k)/y$ consecutive segments of length y (we assume that s is integer). Repeat the decoding for all s placements of the y -segment within the check part of V . The i th placement supplies us with a list $K_i = \{\mathbf{u} = (\mathbf{u}_1 | \mathbf{u}_2)\}$, where every vector \mathbf{u} satisfies (3.11). We are only going to test those error vectors \mathbf{u}_1 that appear as first parts of \mathbf{u} for at least l lists K_i , where l is another parameter of the procedure. Form a list

$$K = K(\gamma) = \left\{ \mathbf{u}_1 | \mathbf{u} = (\mathbf{u}_1 | \mathbf{u}_2) \in \bigcap_{j=1}^l K_{i_j} \text{ for some } 1 \leq i_1 < i_2 \cdots < i_l \leq s \right\} \quad (3.12)$$

Entries of this list are possible error vectors in the coordinates of γ . Therefore, we subtract them from $\mathbf{b}(\gamma)$ to form a list $J(\gamma) = \{\mathbf{b}(\gamma) - \mathbf{u}_1 | \mathbf{u}_1 \in K(\gamma)\}$. For every vector \mathbf{z} in this list we examine all code vectors \mathbf{c}' with $\mathbf{c}'(\gamma) = \mathbf{z}$. We update the current decision $\hat{\mathbf{a}}$ by setting $\hat{\mathbf{a}} = \mathbf{c}'$ if this procedure finds a vector with $dist(\mathbf{b}, \mathbf{c}') < dist(\mathbf{b}, \hat{\mathbf{a}})$. Now let us give the formal description of the algorithm [7].

Supercode Decoding:

1. Compute the syndrome $\mathbf{s} = \mathbf{b} \cdot \mathbf{H}^T$. Set $\hat{\mathbf{a}} = \mathbf{0}$.
2. Choose a random subset $\gamma \in \{0, 1, \dots, n - 1\}$, $|\gamma| = k$. Bring the matrix \mathbf{H} to the form $\mathbf{H}' = [\mathbf{A} | \mathbf{I}_{n-k}]$, where the columns of \mathbf{A} have their numbers in γ .

3. Split the subset $n - k$ into $s = (n - k)/y$ segments of length y . For every i , $1 \leq i \leq s$, do the following two steps:
 - 3.1. Form the $(y \times (k + y))$ matrix $\mathbf{H}_i = [\mathbf{A}_i | \mathbf{I}_y]$, isolating rows $y(i - 1) + j$, $1 \leq j \leq y$, of the parity-check matrix \mathbf{H} . Form the vector $\mathbf{s}_i = (\mathbf{s}_{y(i-1)+j})$, $1 \leq j \leq y$.
 - 3.2. Apply the split-syndrome algorithm to form a list K_i of vectors $\{\mathbf{u} = (\mathbf{u}_1 | \mathbf{u}_2)\}$ with $wt(\mathbf{u}_1 | \mathbf{u}_2) \leq (e_1 | e_2)$ that satisfy the equation $\mathbf{s}_i = \mathbf{u} \cdot \mathbf{H}_i^T$.
4. Form the list $K(\gamma)$ of those vectors \mathbf{u}_1 that appear in at least l lists K_i , see (3.12).
5. For every k -vector $\mathbf{m} = \mathbf{b}(\gamma) - \mathbf{u}_1$, $\mathbf{u}_1 \in K(\gamma)$ generate successively all code vectors $\mathbf{c}' \in V$ whose projection on the chosen k -subset γ equals \mathbf{m} . If $dist(\mathbf{b}, \mathbf{c}') < dist(\mathbf{b}, \hat{\mathbf{a}})$, assign $\hat{\mathbf{a}} = \mathbf{c}'$.
6. Output $\hat{\mathbf{a}}$.

The steps 2–5 are performed $L_n(k, e_1)$ times. The value of $L_n(k, e_1)$ will be discussed later.

This algorithm rests on a number of assumptions. First, we have assumed that γ is an information set. In reality, however, γ is not to be found immediately; therefore, we may not be able to diagonalise the parity-check and generator matrices of V . Next we assume that we are able to control the weight of errors on different parts of the received vector. The detailed explanation why these assumptions hold true can be found in [7].

We conclude this section by estimating the performance of our algorithm.

Theorem 3.1 [7]. For almost all long linear codes the supercode algorithm performs complete minimum-distance decoding.

Proof If all our assumptions about the code and the number of errors hold true, the ‘true’ error vector will appear in at least l lists K_i for a certain choice of γ . Then it will be included in the list $J(\gamma)$ and will be encoded on Step 5 into a code vector (a list of code vectors). Obviously, one of these code vectors is the transmitted one. It will be chosen as the decoding result if it is the closest to the received vector \mathbf{b} . Therefore, decoding with the supercode algorithm can result in a wrong codeword, i.e., a codeword other than the transmitted one if one of the following events takes place.

1. The weight of the error is greater than d ;
2. The correct code vector appears in one of the lists $K(\gamma)$ but is not the closest to the transmitted vector \mathbf{b} ;
3. Repeated random choice fails to produce a partition with the desired error distribution.

For almost all codes, the first and second events form the part of the inherent error of any decoding algorithm (even an exhaustive search would yield an error). The error probability of the complete maximum-likelihood decoding for most codes is known to behave as $p_c = q^{-O(n)}$ [8]. The third event occurs only with probability n^{-n} . We conclude that for almost all codes, except for a fraction of codes that decays exponentially in the code length, the decoding error probability up to $o(1)$ terms behaves as the probability p_c of the complete maximum-likelihood decoding.

3.4 THE COMPLEXITY OF DECODING IN THE CHANNEL WITH INDEPENDENT ERRORS

The complexity of decoding algorithms is currently a subject of extensive study in coding theory. However, the way of measuring complexity itself is seldom discussed or specified, which sometimes results in algorithms that are performed under different computation models being listed as comparable. Here we work with the following two models: random-access machines (RAMs) [9] and Boolean circuits. The RAM is a computing device that has an unrestricted amount of memory with direct access and performs basic operations similar to those performed by Turing machines. This computational model corresponds to ‘real-life’ computers. Most algorithms in coding theory that are currently being studied are formulated under the implicit assumption of this model. The complexity is measured by the number of operations (*time complexity*) and the amount of memory used by the algorithm (*space complexity*). Implementation of decoders by Boolean circuits allows basic operations to be performed in parallel. This approach has a long history in coding theory [10]. The complexity is measured by the number of gates in the circuit (*size*) and the length of the longest path from an input gate to an output gate (*depth*). We discuss implementation of decoders under both models.

The analysis of the complexity of the maximum likelihood (ML) decoding is based on Lemma 2.2, which reduces the ML decoding to the combinatorial problem of the correction of the given number of errors. This lemma claims that in a channel with independent errors it is possible to provide an error probability of not more than two times worse than that provided by ML decoding if decoded in the sphere of radius d_{GV} (see Section 2.5), i.e. to correct t -fold errors, where $t < d_{GV}$, and d_{GV} is the maximal number such that

$$\sum_{i=1}^{d_{GV}-1} \binom{n}{i} \cdot (q-1)^i \leq q^{n-k},$$

where q is the number of symbols in the alphabet, n is the code length, and k is the number of information symbols of the code. The *relative* GV distance δ_{GV} is the limit value of δ_{GV}/n as $n \rightarrow \infty$. Let $R = k/n$ be the rate of the code, then $\delta_{GV} = \delta_{GV}(R)$ is the smallest positive root of the equation $R = 1 - H_q(\delta)$, where $H_q(x) = x \cdot \log_q(q-1) - x \log_q x - (1-x) \cdot x \log_q(1-x)$ is the entropy function.

By their nature, decoding algorithms allow a certain error rate due to the occasional high number of errors in the channel. The idea of reducing the decoding complexity is to allow an algorithmic error whose rate has, at most, the same order as that of inherent error events. The overall decoding error rate for long codes is then essentially the same as that for minimum-distance decoding.

This line of research was initiated by the work of Evseev [11]. He has studied decoding in discrete additive channels. Let X be a finite input alphabet and $Y \supseteq X$ a finite output alphabet of the channel (say, an additive group). A channel is called *additive* if $\Pr(\mathbf{y}|\mathbf{x}) = \Pr(\mathbf{y} - \mathbf{x})$, i.e., the error process does not depend on the message transmitted. Let p_{ML} be the error probability of maximum-likelihood decoding. Evseev has proved that any decoding algorithm that examines q^{n-k} most probable error patterns has error probability $p \leq 2p_{ML}$. Specialising this for the q -ary symmetric channel and using the definition of d_{GV} , we observe that given a (n, k, d) linear code, inspecting all possible errors in the sphere of radius

$d_{GV}(n, k)$ around the received word rather than all error patterns, at most doubles the decoding error probability. Based on this, Evseev [11] proposed a general decoding algorithm whose asymptotic complexity $q^{k(1-R)(1+o(1))}$ improved all the methods known at that time.

This work opened a new page in the study of decoding algorithms of general linear codes. Papers [3], [5], [6], [12], [13] and [14] introduced new decoding methods and provided theoretical justification of those already known.

In the asymptotic setting this approach was later supplemented by the following important result.

Theorem 3.2 [15]: The covering radius of almost all (n, k) random linear codes equals $d_{GV} \cdot (1 + o(1))$.

Remark Most of the results hereafter are formulated for long linear codes. Strictly speaking, this means that we study families of (n, k_n, d_n) codes of growing length. Let V_n be such a family. Suppose there exists the limit $R = \lim_{n \rightarrow \infty} k_n/n$ called the rate of the family. The statement of the theorem means that if ρ_n is the covering radius of V_n , then with probability $\rightarrow \infty$ the quotient $\rho_n/n \rightarrow \delta_{GV}(R)$.

Theorem 3.2 implies that for long linear codes, correcting a little more than d_{GV} errors ensures the same output error rate as complete minimum-distance decoding. For these reasons, the algorithms considered below restrict themselves to decoding in the sphere of radius $d_{GV} = n \cdot \delta_{GV}(R)$. In the rare case that the algorithms find no codeword at all, we can take any codeword as the decoding result. Asymptotically this will not affect the decoding error rate. We wish to underline that these algorithms, in contrast to gradient-like methods [16], [17], perform complete maximum-likelihood decoding only in the limit as $n \rightarrow \infty$. Their error probability as a function approaches the error probability of complete maximum likelihood decoding (the limit of their quotient is one).

Recently, Dumer [18], [19] extended both results to the case of much more general channels. Namely, he has proved [18] that for a symmetric channel with finite input X and arbitrary output $Y \supseteq X$, a decoding algorithm that examines $N > q^{n-k}$ most probable vectors of X^n has error probability $p \leq p_{ML} \cdot (1 + (q^{n-k}/N - q^{n-k}))$. This enabled him to construct general maximum-likelihood soft-decision decoding algorithms with reduced complexity similar in spirit to the known hard-decision decoding methods.

Algorithms that we discuss examine a list of plausible candidates for the decoder output. Our sole concern will be that the transmitted codeword appears in this list. If it later fails to be chosen, this is a part of the inherent error event rather than the fault of a specific decoder. Note that in practice we often do not need to store the whole list, keeping only the most plausible candidates obtained so far.

Minimum-distance decoding can be accomplished either by inspecting all codewords of V (time complexity $O(nq^k)$) or by storing the table of syndromes and coset leaders (space complexity $O(nq^{n-k})$). The last method is called *syndrome decoding*. The only known algorithm that yields an asymptotic improvement of these methods based on geometric properties of the code itself, i.e., without introducing an additional algorithmic error rate, is the *zero-neighbours algorithm* [17]. The decoding is accomplished by iterative refinements of the current decision in much the same way as are standard optimisation methods in continuous spaces. However, since our space is discrete, in order to determine the direction

that reduces the value of the objective function (the distance between the received vector \mathbf{y} and the closest codeword found), one has to inspect a certain subset of codewords called zero neighbors. It is shown in [17] that this subset lies entirely inside the sphere of radius $2t + 1$ about $\mathbf{0}$, where t is the covering radius of the code. Both time and space complexity of this decoding are governed by the size of this set. By Theorem 3.2, the covering radius for almost all codes of rate R grows as $n\delta_{GV}(R)$. Thus the complexity of this algorithm for codes of rate R is dominated by the asymptotic size of the sphere of radius $2n\delta_{GV}(R)$, given in Lemma 3.5 below. This leads to the following result.

Theorem 3.3 [17]: Let V be an (n, Rn) linear code. For any $\mathbf{y} \in E_q^n$, zero-neighbors decoding always finds a closest codeword. For almost all codes it can be implemented by a sequential algorithm with both time and space complexity $q^{\eta_q(R)(1+o(1))}$, where

$$\eta_q(R) = \begin{cases} R, & 0 \leq R \leq 1 - H_q\left(\frac{q-1}{2q}\right) \\ H_q(2\delta_{GV}) - (1-R), & 1 - H_q\left(\frac{q-1}{2q}\right) < R < 1. \end{cases}$$

A parallel implementation of this decoding requires a Boolean circuit of size $q^{\eta_q(R)(1+o(1))}$ and depth $O(n^2)$.

For instance, for $q = 2$, the complexity of this decoding is exponentially smaller than that of the exhaustive search for $R \geq 1 - H_2(1/4) = 0.189$. This is also smaller than the time complexity $O(nq^{k(1-R)})$ of the decoding algorithm in [11] for high code rates.

However, as shown in [20], this approach seems to have already reached its limits. Namely, a result in [20] shows that any ‘gradient-like’ decoding method for binary codes, all of whose codewords have even weight, has to examine all zero neighbors. (See [20] for definitions and exact formulations.) For this reason we turn our attention to information-set decoding.

Let us choose the number of steps $L_n(k)$ in Covering-Set Decoding algorithm as follows

$$L_n(k) = (n \log n) \cdot \binom{n}{d_{GV}} / \binom{n-k}{d_{GV}}. \tag{3.13}$$

A key result in [13] and [21] states that as the length of the code grows, any k coordinates form an ‘almost’ information set, i.e., that the codimension of the space of code vectors in any k coordinates is small. This means that the number of code vectors that project identically on any k coordinates, i.e., the size of the list $M(\gamma)$ for any γ , is small. The following lemma shows that this size grows as $q^{O(n^{1/2})}$ and, therefore, does not contribute to the main term of the complexity estimate. In [7] a slightly better estimate was proved for the corank of square submatrices of a random matrix than the one in [13] and [21].

Lemma 3.2 Let \mathbf{A} be a random $(k \times n)$ matrix over F_q , $k < n$, $k, n \rightarrow \infty$. For almost all matrices, the corank of every square $(k \times k)$ submatrix \mathbf{B} is

$$\text{corank } \mathbf{B} \leq \sqrt{\log_q \binom{n}{k}}.$$

Proof Let us estimate the probability $\pi(k, u)$ that a random $k \times k$ matrix has rank u . Every $(k \times k)$ -matrix of rank u corresponds to a linear mapping that takes E_q^k to a u -dimensional space F that can be viewed as a subspace of E_q^k . The number of image subspaces is

$$\begin{bmatrix} k \\ u \end{bmatrix} = \prod_{j=0}^{u-1} (q^k - q^j) / (q^u - q^j).$$

Likewise, the number of kernels of a rank u mapping is $\begin{bmatrix} k \\ u \end{bmatrix}$. Every choice of the basis in the subspace $F \subset E_q^k$ accounts for a different matrix. The number of bases in F equals $\prod_{j=0}^{u-1} (q^u - q^j)$. Therefore, the number of square matrices of order k and rank u equals

$$\prod_{j=0}^{u-1} (q^k - q^j)^2 / (q^u - q^j).$$

Thus

$$\pi(k, u) = q^{-k^2} \prod_{j=0}^{u-1} \frac{(q^k - q^j)^2}{(q^u - q^j)} < q^{-k^2 + ku} \prod_{j=0}^{u-1} \frac{q^{k-j} - 1}{q^{u-j} - 1},$$

where we have estimated $\prod_{j=0}^{u-1} (q^k - q^j) = q^{(1/2)u(u-1)} \prod_{j=0}^{u-1} (q^{k-j} - 1)$ by q^{ku} . Estimating the Gaussian binomial, we obtain

$$\begin{aligned} \prod_{j=0}^{u-1} \frac{q^{k-j} - 1}{q^{u-j} - 1} &< q^{u(k-u)} \prod_{j=0}^{u-1} \frac{q^{u-j}}{q^{u-j} - 1} = q^{u(k-u)} \prod_{j=0}^{u-1} \left(1 + \frac{1}{q^j - 1} \right) \\ &= q^{u(k-u)} \cdot \frac{q}{q-1} \cdot \left(1 + \frac{1}{q^2 - 1} + \dots \right). \end{aligned}$$

For $q \geq 2$, the constant factor here is less than 5 since the omitted terms are always less than 1. Thus

$$\pi(k, u) < 5q^{-(k-u)^2}. \quad (3.14)$$

Since there are $\binom{n}{k}$ possibilities for \mathbf{B} , the probability that there is a submatrix of \mathbf{A} with corank $> k - l$ is

$$\sum_{u=0}^{l-1} \binom{n}{k} \pi(k, u). \quad (3.15)$$

Substituting (3.14) in (3.15) we obtain that the value of probability (3.15) does not exceed

$$\sum_{u=0}^{l-1} \binom{n}{k} \pi(k, u) < 5 \binom{n}{k} q^{-(k-l)^2} \sum_{u=0}^{l-1} q^{-(k-u)^2 + (k-l)^2} < 5 \binom{n}{k} q^{-(k-l)^2} \sum_{i=1}^l q^{-(i+1)^2 + 1}.$$

The last sum is maximal for $q = 2$. It can be checked not to exceed 0.2. Therefore, the required probability is at most $\binom{n}{k} \cdot q^{-(k-l)^2}$, which falls exponentially if

$$\text{corank } \mathbf{B} > \sqrt{\log_q \binom{n}{k}}. \quad \text{Q.E.D.}$$

The time complexity of the generalised covering-set decoding is determined by the quantity $L_n(k)$ (3.13). Let us formulate the properties of the generalised covering-set decoding algorithm as a theorem.

Theorem 3.4 [13], [21]: The covering-set decoding for almost all codes performs minimum distance decoding. The decoding can be implemented by a sequential algorithm with time complexity at most

$$O(n^4 (\log n) q^{n\alpha^{(q)}(R) + \sqrt{nH_2(R)/\log_2 q}}) = q^{n\alpha^{(q)}(R)(1+o(1))}, \quad (3.16)$$

where

$$\alpha^{(q)}(R) = (\log_q 2) \left[H_2(\delta_{GV}) - (1-R) \cdot H_2\left(\frac{\delta_{GV}}{1-R}\right) \right] \quad (3.17)$$

and space complexity $O(n^3)$. A parallel implementation of this algorithm requires a circuit of size $q^{n\alpha^{(q)}(R)(1+o(1))}$ and depth $O(n)$.

Now let us consider the complexity of the split syndrome decoding. The total size of tables X_l and X_r used in this algorithm is $O\left(n \cdot \binom{m}{u} \cdot (q-1)^u\right)$ and $O\left(n \cdot \binom{n-m}{t-u} \cdot (q-1)^{t-u}\right)$ correspondingly (see section 3.2). We may optimise on the choice of m and u in order to reduce the total size of memory used for the tables X_l and X_r . Since this size is a sum of two exponential functions, for every error distribution we must choose the point so that both tables are (roughly) equally populated. The size of the memory is then bounded as $O(M(t))$, where

$$M(t) = n \cdot \left(\binom{m}{u} \cdot (q-1)^u \right)^{1/2} \cdot n \cdot q^{nH_q(t/n)/2} \quad (3.18)$$

Note that in Step 3 this algorithm makes a call to a sorting subroutine. Suppose we need to order an array of N binary vectors of length n . Sorting can be accomplished by a sequential algorithm (RAM computation) with both time and space complexity $O(nN)$. Alternatively, sorting an array can be implemented by a Boolean circuit of size $O(nN)$ and depth $O(n^2)$

[22], [23]. The complexity of sorting dominates the space complexity of split syndrome decoding and all algorithms dependent on it. The properties of the algorithm can be summarized as follows.

Theorem 3.5 [5]. For most long linear codes the split syndrome decoding algorithm performs minimum-distance decoding. Its sequential implementation has for any code of rate R time complexity $O(n \cdot d_{GV}^2 \cdot M(d_{GV})) = q^{(1/2)n(1-R)(1+o(1))}$ and space complexity $O(n \cdot M(d_{GV}))$. Its parallel implementation requires a Boolean circuit of size $O(n^4 \cdot q^{(1/2)n(1-R)(1+o(1))})$ and depth $O(n^2)$.

The time complexity of the split-syndrome algorithm is smaller than the complexity of the generalised covering-set decoding for high code rates. For instance, for $q = 2$ the complexity exponent of this algorithm is better than $\alpha^{(2)}(R)$ in (3.17) for $R \geq 0.954$. Therefore, if we puncture the code V (i.e., cast aside some of its parity symbols) so that its rate becomes large, we can gain in complexity by applying split syndrome decoding to the punctured code V' . By Lemma 3.2 we may regard any symbols as parity ones provided that their number is less than $n - k$. This is the basic idea of the second improvement of covering-set decoding, undertaken in [6]. We shall call this algorithm *punctured split syndrome decoding*. The main result of [6] is expressed by the following theorem, which we state for $q = 2$. Let

$$\beta^{(2)}(R) = \min_{u, \alpha} \max \left\{ (1-u) \cdot \left[1 - H_2 \left(\frac{\delta_{GV} - \alpha}{1-u} \right) \right], \right. \\ \left. 1 - R - \frac{1}{2} u H_2 \left(\frac{\alpha}{u} \right) - (1-u) \cdot \left[1 - H_2 \left(\frac{\delta_{GV} - \alpha}{1-u} \right) \right] \right\} \quad (3.19)$$

with $R \leq u \leq 1$ and $\max(0, \delta_{GV} + u - 1) < \alpha < \min(\delta_{GV}, u)$, and let $\alpha^* = \alpha^*(R)$ and $u^* = u^*(R)$ be the values that furnish this minimum for a given rate R .

Theorem 3.6 [6]. For most long binary linear codes the punctured split syndrome algorithm performs complete minimum-distance decoding. Its sequential implementation for a code of rate R has time complexity $2^{n\beta^{(2)}(R)(1+o(1))}$ and space complexity $2^{(1/2)u^*nH_2(\alpha^*/u^*)(1+o(1))}$.

Setting the parameters α and u to values other than α^* and u^* , we can trade time complexity for space complexity. Taking $\alpha = \alpha^*$ and $u = u^*$ furnishes the minimum time complexity of the algorithm.

Now let us discuss the complexity of the supercode decoding algorithm (see section 3.3). Given a code rate R , we choose the values of v , α , l and that furnish the minimum to the complexity exponent, found below in Theorem 3.7. This determines the algorithm parameters $y = v \cdot n$ and $e_1 = \alpha \cdot n$. Let

$$L_n(k, e_1) = (n \log n) \cdot \left[\binom{n}{d_{GV}} / \binom{k}{e_1} \cdot \binom{n-k}{d_{GV} - e_1} \right], \quad (3.20)$$

$$e_2 = \frac{(\delta_{GV} - \alpha) \cdot y}{1 - R - (l-1) \cdot v}. \quad (3.21)$$

The supercode decoding implies that we need to find a partition of the set $\{0, 1, \dots, n-1\}$ into a k -subset with at most e_1 errors and an $(n-k)$ -subset. This $(n-k)$ -subset must have the property that once it is partitioned into s consecutive segments, at least l of them have at most e_2 errors. This is done in two stages. In the first stage we choose a partition of $\{0, 1, \dots, n-1\}$ into subsets of size k and $n-k$ randomly and independently many times. The probability that one such choice does not generate the required distribution of errors equals

$$1 - \binom{k}{e_1} \cdot \binom{n-k}{d_{GV} - e_1} / \binom{n}{d_{GV}}.$$

By repeating this choice independently $L_n(k, e_1)$ times (see (3.20)) we ensure that the probability of failing to construct the mentioned above system decays as $e^{-n \log n}$.

Letting $k = Rn$ and $e_1 = \alpha \cdot n$, we obtain the estimate of (3.20) as follows

$$L_n(k, e_1) \leq q^{n\varepsilon_1(R, \alpha) \cdot (1+o(1))}, \quad (3.22)$$

where

$$\varepsilon_1(R, \alpha) = (\log_q 2) \cdot \left[H_2(\delta_{GV}) - R \cdot H_2\left(\frac{\alpha}{R}\right) - (1-R) \cdot H_2\left(\frac{\delta_{GV} - \alpha}{1-u}\right) \right]. \quad (3.23)$$

The second stage is performed for each partition of $\{0, 1, \dots, n-1\}$ generated in the first stage. Taking the second subset, we split it into $s = (n-k)/y = O(n-k)$ consecutive segments of length y (except maybe the last, shorter, one). Every choice of this segment supplies us with a desired partition of $\{0, 1, \dots, n-1\}$ into three parts. The total number of partitions equals $O(n-k) \cdot L_n(k, e_1)$.

Each partition enables us to isolate the coordinates of $V(y)$. We intend to perform the decoding of $V(y)$ by solving (3.11), i.e., to form a list K_i , $1 \leq i \leq s$ of vectors $\{\mathbf{u} = (\mathbf{u}_1 | \mathbf{u}_2)\}$ that satisfy it. The goal of this decoding is that $wt(\mathbf{e}_1) \leq e_1$, provided that the ‘true’ error vector appears in K_i for at least l different indexes i . To achieve this, we compute the minimal value of e_2 such that $(s-l+1)$ y -segments of the error vector cannot all be heavier than e_2 . Since $s-l+1 = \frac{n-k-(l-1) \cdot y}{y}$ we obtain for e_2 the inequality

$$e_2 > \frac{\delta_{GV} - e_1}{s-l+1} = \frac{(\delta_{GV} - \alpha) \cdot y}{1-R-(l-1) \cdot v}.$$

Note that the greater the value of e_2 , the greater is the size of the resulting list of vectors that should be tested in further steps. Therefore, e_2 should be as small as possible. This justifies our choice of e_2 in (3.21).

Looking at the description of the supercode decoding algorithm, we see that Steps 1–3.1 have algebraic complexity. The most computationally involved parts are Steps 3.2–5. Let us estimate their asymptotic complexity.

We begin with Step 3.1. Suppose that the values of e_1 and e_2 are fixed. Let us describe the process of solving (3.11) with respect to the unknown vector $(\mathbf{u}_1 | \mathbf{u}_2)$ with

$wt(\mathbf{u}_1|\mathbf{u}_2) \leq (e_1|e_2)$. To find a list of solutions of (3.11) we apply a version of split syndrome decoding.

Lemma 3.3 The list of solutions of (3.11) with $wt(\mathbf{u}_1|\mathbf{u}_2) \leq (e_1|e_2)$, where $e_1 \leq \alpha \cdot n$ and e_2 satisfies (3.21), can be compiled using the split syndrome algorithm. Both time and space complexity of this stage are bounded as $q^{(1/2) \cdot n \varepsilon_2(R, v, \alpha, l) \cdot (1+o(1))}$, where

$$\varepsilon_2(R, v, \alpha, l) = R \cdot H_2\left(\frac{\alpha}{R}\right) + v \cdot H_2\left(\frac{\delta_{GV} - \alpha}{1 - R - (l-1) \cdot v}\right). \quad (3.24)$$

The proof can be found in [7].

It is quite essential for us not to write out the lists K_i explicitly but to store them as pairs of lists (X_l, X_r) since the size $|K_i|$ can be much greater than the size of its ‘building blocks.’ This happens because each entry in X_l can be coupled with many entries in X_r to form error vectors \mathbf{u} in the list K_i . We remark without further discussion that writing out the lists K_i explicitly would yield an algorithm of complexity asymptotically equal to that of punctured split syndrome decoding (Theorem 3.6).

Thus we need to find intersections of the lists K_i each of which is specified by two ‘halves,’ i.e., lists X_l and X_r . Our goal is to find error vectors that appear in at least l out of the s lists K_i , where l is a constant that depends only on the rate of the code V . Therefore, we can afford to examine all the possible $\binom{s}{l}$ groups of l lists.

The complexity of constructing the intersection of a given group of l lists K_i is the sum of the number of operations needed to compute the intersection and the size of the resulting list. The number of operations is estimated in the following lemma.

Lemma 3.4 The intersection of given l lists K_i can be computed in time of order $q^{(1/2) \cdot n \varepsilon_2(R, v, \alpha, l) + vm}$. The size of the memory used by the computation is at most $q^{(1/2) \cdot n \varepsilon_2(R, v, \alpha, l)}$.

Proof See in [7].

Let us estimate the size of $K(\gamma)$. For this we use the following lemma.

Lemma 3.5 Let V be an (n, k) code of rate $R = k/n$. Suppose $S \subset E_q^n$ is a set of size $q^{\sigma n}$, $1 - R < \sigma < 1$ and let $U_\sigma = |V \cap S|$ be the number of codewords of V in this set. Then

$$U_\sigma \leq \frac{|S|}{q^{n-k}} \cdot q^{n \cdot o(1)} = q^{n \cdot (\sigma - (1-R)) \cdot (1+o(1))}. \quad (3.25)$$

for all codes except for an n^{-n} fraction of them.

Proof See in [7].

Note that the decrease rate of the fraction of ‘bad’ codes, i.e., codes that do not satisfy the statement of this lemma, is quite important for us since we are going to choose the

order of $q^{O(n)}$ codes $V(y)$ and need the estimate (3.25) to hold for all of them at a time. A more accurate estimate shows that this decay rate can be brought down to $q^{-O(n^2)}$. The last lemma allows us to estimate the size of the list of those vectors \mathbf{u}_1 that appear in at least l of the codes K_i , see (3.12).

Corollary 3.1 For almost all codes V and almost all choices of l supercodes $V(y)$, the size of the list $K(\gamma)$ is at most $q^{n \cdot (\varepsilon_2(R, v, \alpha, l) - lv) \cdot (1+o(1))}$, where the function ε_2 is defined in (3.24).

Proof See in [7].

We are now able to estimate the complexity of Step 4 of the supercode decoding algorithm.

Lemma 3.6 The time complexity of implementing Step 4 of the supercode decoding algorithm has the exponential order $\max \left\{ \frac{1}{2} \varepsilon_2(R, v, \alpha, l) + v, \varepsilon_2(R, v, \alpha, l) - lv \right\}$. The space complexity is bounded above by $q^{(1/2)n \cdot \varepsilon_2(R, v, \alpha, l)}$.

Proof As said before Lemma 3.4, the complexity of Step 3.2 is a sum of two terms. The first term is estimated in Lemma 3.4. The second term is the size of the resulting list, estimated in the previous corollary. The complexity of this step is estimated from above by the sum of these two exponential functions. Therefore, the exponent of the time complexity of this step is at most the maximum of the two exponents. *Q.E.D.*

Combining Lemmas 3.3, 3.6, and formula (3.23), we can prove the following result.

Theorem 3.7 The supercode decoding algorithm for almost all long linear (n, k) codes of rate $R = k/n$ performs minimum-distance decoding. The time complexity of its sequential implementation for almost all codes is at most $q^{n \zeta^{(q)}(R) \cdot (1+o(1))}$, where

$$\zeta^{(q)}(R) = \min_{v, \alpha, l} \left\{ \varepsilon_1(R, \alpha) + \max \left(\frac{1}{2} \varepsilon_2(R, \alpha, v, l) + v, \varepsilon_2(R, \alpha, v, l) - lv \right) \right\}. \quad (3.26)$$

and the functions ε_1 and ε_2 are defined in (3.23) and (3.24), respectively. The optimisation parameters are restricted to

$$\begin{aligned} \max(0, \delta_{GV} + R - 1) < \alpha < \min(\delta_{GV}, R) \\ \delta_{GV} - \alpha < 1 - R - (l - 1) \cdot v \end{aligned} \quad (3.27)$$

The space complexity of the algorithm is estimated from above as $q^{(1/2)n \cdot \varepsilon_2(R, v, \alpha, l) \cdot (1+o(1))}$. A parallel implementation of the algorithm requires a Boolean circuit of size $q^{n \zeta^{(q)}(R) \cdot (1+o(1))}$ and depth $O(nl)$.

Proof The complexity of Steps 2–3.1 is algebraic and contributes only to $o(1)$ terms in the exponent. Let us estimate the complexity of Step 5. As said above, for each vector $\mathbf{m} \in K$ we compute a list of at most $q^{O(n^{1/2})}$ code vectors that agree with it in the

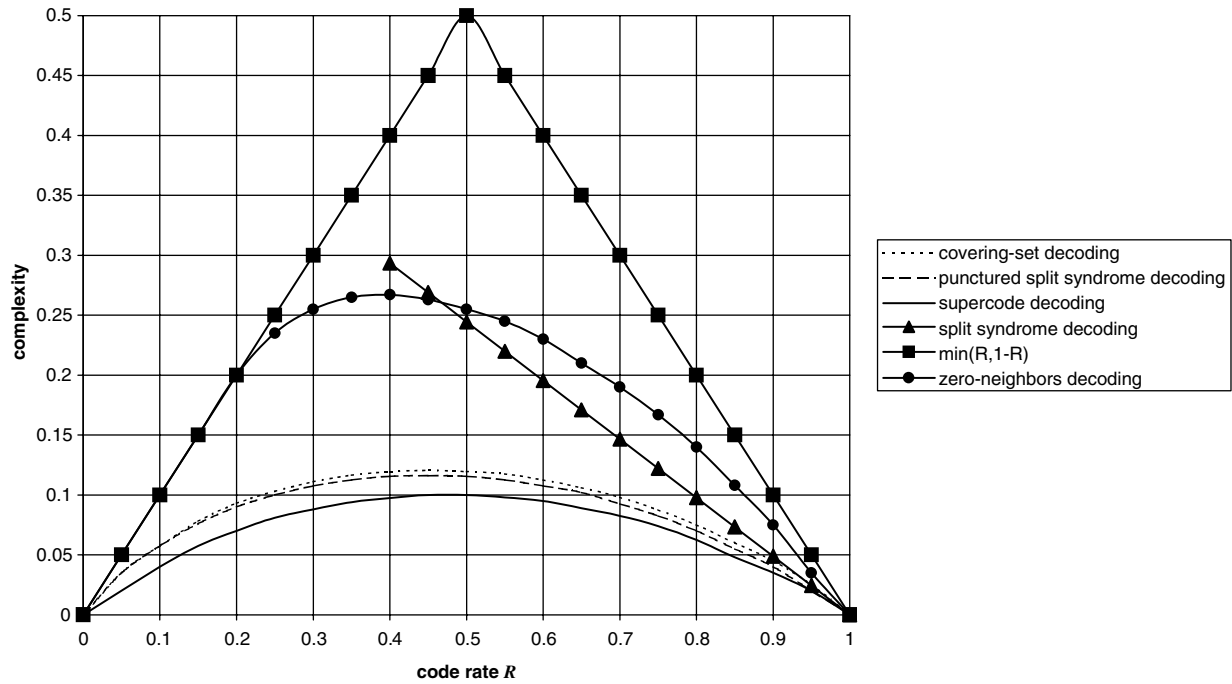


Figure 3.1 Complexity of the hard-decision decoding algorithms for binary codes

k -part. If one of these vectors, say \mathbf{c}' , is closer to the received vector \mathbf{b} than the current decision $\hat{\mathbf{a}}$, we update it by assigning $\hat{\mathbf{a}} = \mathbf{c}'$. Thus Step 5 has time complexity of the same exponential order as Step 4. Therefore, by Lemma 3.6 we see that the most time-consuming steps of the algorithm are Steps 4 and 5. The entire sequence of steps is repeated $L_n(k, e_1)$ times. Therefore, the complexity exponent of the algorithm equals $\log_q L_n(k, e_1)$, given by (3.23), plus the exponent found in Lemma 3.6. The parameters α, ν, l should be chosen to minimise this value. The first of inequalities (3.27) is obvious; the second is implied by the definition of e_2 . *Q.E.D.*

The asymptotic complexity of the supercode decoding algorithm is exponentially smaller than the best known result [6] for any $q \geq 2$ and any code rate $R, 0 < R < 1$. The complexity exponents of the algorithms mentioned in this section for binary codes are shown in Figure 3.1 (the complexity of the zero-neighbors decoding is represented in Theorem 3.3, covering-set decoding in Theorem 3.4, split syndrome decoding in Theorem 3.5, punctured split syndrome decoding in Theorem 3.6, and the complexity of the supercode decoding is represented in Theorem 3.7).

Thus the asymptotic complexity of the supercode decoding algorithm is less than the complexity of all other hard-decision decoding methods known.

REFERENCES

1. Gordon, D. M. (1982). Minimal permutation sets for decoding the binary Golay code, *IEEE Trans. Inform. Theory*, **IT-28**, 541–3.
2. Wolfmann, J. (1983). A permutation decoding of the (24, 12, 8) Golay code, *IEEE Trans. Inform. Theory*, **IT-29**, 748–51.
3. Krouk, E. A. and Fedorenko, S. V. (1995) Decoding by generalized information sets, *Probl. Inform. Transm.*, **31**, (2), 54–61 (in Russian) and 134–9 (English translation).
4. Barg, A. (1998). Complexity issues in coding theory, in *Handbook of Coding Theory*, vol. 1, (ed. V. Pless and Huffman, W. C.) Elsevier Science, Amsterdam, The Netherlands: pp. 649–754.
5. Dumer, I. (1989). Two decoding algorithms for linear codes, *Probl. Inform. Transm.*, **25**, (1), 24–32 (in Russian) and 17–23 (English translation).
6. Dumer, I. (1999). On minimum distance decoding of linear codes, in *Proc. 5th Joint Soviet–Swedish Int. Workshop Information Theory*, pp. 50–52, Moscow, Russia.
7. Barg, A., Krouk, E. and van Tilborg H. C. A. (1999). On the Complexity of Minimum Distance Decoding of Long Linear Codes, *IEEE Trans. Inform. Theory*, **IT-45**, 1392–1405.
8. Gallager, R. (1963). *Low-Density Parity-Check Codes*. MIT Press, Cambridge, MA.
9. Aho, A. V., Hopcroft, J. E., and Ullman, J. D. (1974). *The Design and Analysis of Computer Algorithms*. Addison-Wesley, London, U.K..
10. Savage, J. E. (1969 and 1971). The complexity of decoders, I, II, *IEEE Trans. Inform. Theory*, **IT-15**, 689–95, **IT-17**, 77–85.
11. Evseev, G. S. (1983). Complexity of decoding for linear codes, *Probl. Inform. Transm.*, **19**, (1), 3–8 (in Russian) and 1–6 (English translation).
12. Barg, A. and Dumer, I. (1986). Concatenated decoding algorithm with incomplete inspection of code vectors, *Probl. Inform. Transm.*, **22**, (1), 3–8 (in Russian) and 1–7 (English translation).
13. Coffey, J. T. and Goodman, R. M. F. (1990). The complexity of information set decoding, *IEEE Trans. Inform. Theory*, **35**, 1031–7.
14. Coffey, J. T., Goodman, R. M. F., and Farrell, P. (1991). New approaches to reduced complexity decoding, *Discr. Appl. Math.*, **33**, 43–60.

15. Blinovskii, V. M. (1987). Lower asymptotic bound on the number of linear code words in a sphere of given radius in F_q^n , *Probl. Inform. Transm.*, **23**, (2) 50–3 (in Russian) and 130–2 (English translation).
16. Hwang, T.-Y. (1979). Decoding linear block codes for minimizing word error rate, *IEEE Trans. Inform. Theory*, **IT-25**, 733–7.
17. Levitin, L. and Hartmann, C. R. P. (1985). A new approach to the general minimum distance decoding problem: The zero-neighbors algorithm, *IEEE Trans. Inform. Theory*, **IT-31**, 378–84.
18. Dumer, I. (1996). Suboptimal decoding of linear codes: Partition technique, *IEEE Trans. Inform. Theory*, **42**, 1971–86.
19. Dumer, I. (1996). Covering lists in maximum likelihood decoding, in *Proc. 34th Annu. Allerton Conf. Communications, Control, and Computing*, 683–92.
20. Ashikhmin, A. and Barg, A. (1998). Minimal vectors in linear codes, *IEEE Trans. Inform. Theory*, **44**, 2010–17.
21. Krouk, E. A. (1989). Decoding complexity bound for linear block codes, *Problems of Info. Trans.*, **25**, (3), 103–7 (in Russian) and 251–4 (English translation).
22. Cormen, T. H., Leiserson, C. E., and Rivest, R. L. (1990). *Introduction to Algorithms*. MIT Press, Cambridge, MA, USA.
23. Knuth, D. E. (1973). *The Art of Computer Programming*, vol. 3. Reading, MA: Addison-Wesley, Reading, MA, USA.