Serdica J. Computing **1** (2007), 157–170

Serdica
Journal of Computing

Bulgarian Academy of Sciences
Institute of Mathematics and Informatics

ON THE ERROR-CORRECTING PERFORMANCE OF SOME BINARY AND TERNARY LINEAR CODES

Tsonka S. Baicheva

Abstract. In this work, we determine the coset weight spectra of all binary cyclic codes of lengths up to 33, ternary cyclic and negacyclic codes of lengths up to 20 and of some binary linear codes of lengths up to 33 which are distance-optimal, by using some of the algebraic properties of the codes and a computer assisted search. Having these weight spectra the monotony of the function of the undetected error probability after t-error correction $P_{ue}^{(t)}(C,p)$ could be checked with any precision for a linear time. We have used a programm written in Maple to check the monotony of $P_{ue}^{(t)}(C,p)$ for the investigated codes for a finite set of points of $p \in [0, \frac{p}{q-1}]$ and in this way to determine which of them are not proper.

I. Introduction. Environmental interference and physical defects in the communication medium can cause random bit errors during data transmission. One way of achieving more reliable communication is to incorporate some kind of error-correcting codes. Then an encoder forms a codeword from an incoming message. The codeword consists of the message and some redundant information.

ACM Computing Classification System (1998): G.2.3.
Key words: Proper codes, binary cyclic codes, ternary cyclic and negacyclic codes.

The codeword is transmitted over the channel and at the receiver a decoder is used to decide upon which codeword is most likely the transmitted one. Provided that not too many errors have appeared during transmission the receiver will be able to recover the message.

In this work, we will focus our attention on the most important class among error-correcting block codes – the linear codes. These codes have an accessible mathematical structure which leads to effective coding and decoding methods and allows to analyze their performance. A particular class of linear codes – cyclic codes – have properties which make them very easily to implement and have a wide range of applications.

When we would like to choose the most appropriate code for a given application we are interested of its error-control capabilities. A measure of these capabilities is the minimum distance of the code which gives the number of detectable and correctable errors. Another important measures are the probability of undetected error and the probability of correct decoding which are the object of the investigation in this work.

II. Preliminaries. Let F_q^n be an n-dimensional vector space over the finite field with q elements. A *linear code* C is a k-dimensional subspace of F_q^n. A k-by-n matrix G having as rows the vectors of a basis of C is called a *generator matrix* of C.

Let $x, y \in F_q^n$. The *Hamming distance* $d(x, y)$ between x and y is the number of positions in which x and y differ, i.e. $d(x, y) = |\{\imath \mid x_i \neq y_i\}|$. The *minimum distance* of a linear code C is the minimum Hamming distance between all distinct pairs of codewords in C.

With these notations a linear code with length n, dimension k and minimum distance d over F_q is denoted by $[n, k, d]_q$.

The set of all vectors of F_q^n orthogonal, with respect to the usual inner product, to all codewords from C is called the *dual code* $C^\perp$ to C which is a linear $[n, n-k]_q$ code. A generator matrix of the code $C^\perp$ is a *parity check* matrix of C.

Let A_i denote the number of codewords of C of weight i. Then the numbers $A_0, \ldots, A_n$ are called the *weight distribution* of the code C.

A *coset* of the code C defined by the vector $x \in F_q^n$ is the set $x + C = \{x + c \mid c \in C\}$. A *coset leader* of $x + C$ is a vector in $x + C$ of smallest weight. We will denote by α_i for $i = 0, 1, \ldots, n$ the number of coset leaders of weight i.

Let us for brevity denote the set of all A_i and α_i for $i = 0, 1, \ldots, n$ with $\bar{A}$ and $\bar{\alpha}$ correspondingly.

The *covering radius* R of a code is the largest weight in the set of coset leaders and the smallest integer R, such that the spheres of radius R around the codewords completely cover F_q^n.

We will say that the block code C *detects* t *errors* if each word a', obtained from a codeword a by changing of $1, 2, \ldots, t$ symbols, is not a codeword and *corrects* t *errors* if the Hamming distance $d(a, a')$ is strictly smaller then the Hamming distance of any other codeword to a'. Then every linear $[n, k, d]_q$ code detects up to $d-1$ errors and corrects up to $t = \left\lfloor \frac{d-1}{2} \right\rfloor$ errors.

Consider what happens when a codeword x from an $[n, k, d]_q$ code C is transmitted over a channel and errors occurs during transmission. A transmitter which transmits codewords through a communication channel to the receiver where the word y is obtained. The channel adds to the codeword an *error vector* which is of the same length and has nonzero entries in the positions where errors occur, i.e. $y = x + e$.

A channel model is a description of the probability of receiving a vector y of length n when a vector x of the same length was transmitted. We say that the channel has no memory if the errors in different positions occur independently. The model of a channel we will consider is a q-ary *symmetric channel without memory* (qSC). This is a discrete channel with q-ary input, q-ary output and channel error probability p, where $0 \leq p \leq \frac{p}{(q-1)}$. Any symbol has probability $1-p$ of being received correctly and a probability $\frac{p}{(q-1)}$ of being transformed into each of the other $q-1$ symbols.

On the receiver site decoding is performed to a codeword which is nearest to the received word in the Hamming distance. We assume that the error vector is always a coset leader. Following this procedure

1. we may find a unique codeword y for which the Hamming distance $d(x, y)$ is minimal and decode the received word y to x. We decode correctly in this case. Clearly the probability of correct decoding is given by

$$P_{corr} = \sum_{i=0}^{n} \alpha_i \left(\frac{p}{q-1} \right)^i (1-p)^{n-i}$$

and the probability of error by

$$P_{err} = 1 - P_{corr} = 1 - \sum_{i=0}^{n} \alpha_i \left(\frac{p}{q-1} \right)^i (1-p)^{n-i}.$$

2. We may detect an error if there are more than one codeword with minimal Hamming distance $d(x, y)$.

3. We decode incorrectly if the channel error have changed x in such a way that the closest codeword to y is $x' \neq x$; i.e. we have an *undetectable error.*

Since the code is linear, an undetected error occurs (assuming the code is used for error detection) iff the error vector e is a nonzero codeword. If i positions of the codeword are corrupted, i.e. e has Hamming weight i, then the probability of this error pattern is $\left(\frac{p}{q-1}\right)^i (1-p)^{n-i}$ and the probability of an undetected error is given by

$$P_{ue}(C, p) = \sum_{i=1}^{n} A_i \left(\frac{p}{q-1}\right)^i (1-p)^{n-i}.$$

Let $P_{ue}^{(t)}(C, p)$ denote the probability of an undetected error after t error correction and $P_h(p)$ denote the probability that an undetectable error pattern in a coset of weight h occurs with $0 \leq h \leq t$. Let $Q_{h,l}$ be the number of vectors of weight l in the cosets of minimum weight h, excluding the coset leaders. Then (see [7, 9])

$$P_h(p) = \sum_{l=0}^{n} Q_{h,l} \left(\frac{p}{q-1}\right)^l (1-p)^{n-l}$$

and

$$P_{ue}^{(t)}(C, p) = \sum_{h=0}^{t} P_h(p).$$

III. Criteria for t-proper linear code. When we want to find an $[n, k]$ code for error detection or correction in some applications, the best choice is a code with minimum $P_{ue}^{(t)}(C, p)$ (optimal code). There are two problems when we would like to find such a code. Very often the channel error probability is not a fixed value, i.e. it changes during the time of the transmission. Then a code optimal for $p' \neq p$ may not be optimal for p. Moreover even if we know p, there is in general no method to find an optimal code, except exhaustive search, and this is in most cases not feasible. Therefore, it is useful to have some criteria by which we can judge the usefulness of a given code for error detection.

A code C is called *t-proper* (or only proper when t=0 and the code is used for error detection only) if the function $P_{ue}^{(t)}(C,p)$ is monotonous in the whole interval $\left[0, \frac{p}{(q-1)}\right]$ [8].

Unfortunately, we can check this criterion only for a finite set of points of p. Discrete sufficient condition for a linear $[n,k,d]_q$ code to be *t-good* was derived by Dodunekova and Dodunekov [4].

Theorem. *If for* $l = t+1, \ldots, n$

$$(q^{-(n-k)} - q^{-n})V_q(t) \geq q^{-l} \sum_{i=t+1}^{l} \frac{l_{(i)}}{n_{(i)}} A_i^{(t)}$$

then C is t-good for error correction.

Here $A_i^{(t)} = \sum_{h=0}^{t} Q_{h,i},\ i = t+1, \ldots, n$ is the weight distribution of the vectors in the cosets of weight at most t, excluding the leaders. $V_q(t)$ is the volume of the q-ary sphere of radius t in the n-dimensional vector space over $GF(q)$ and $m_{(i)} = m(m-1)\ldots(m-i+1)$.

If we would like to check monotony of the function $P_{ue}^{(t)}(C,p)$ or the sufficient condition for a code to be proper, or to evaluate $P_{ue}^{(t)}(C,p)$ and P_{corr} we have to know $\bar{A}$, $\bar{\alpha}$ and $Q_{h,l}$. But how hard is the problem of the determination of these values?

IV. Complexity of checking the conditions for proper linear error correcting codes. Berlecamp, McEliece and Van Tilborg [3] showed that the following problem is NP complete: given a $k \times n$ (binary) generator matrix G and an integer w, decide if the code C generated by G contains a codeword of weight w. In particular, this implies that the problem of finding the weight distribution of C is NP hard.

McLoughlin [10] proved that determination of the covering radius of the code is an NP hard problem, i.e determination of $\bar{\alpha}$ and $Q_{h,l}$ are also computationally hard problems.

To check the properness of a linear code we have to know the weight distributions of the code and of its cosets. Therefore we have to solve computationally hard problems. That is why these characteristics are known only for a few classes of codes and there is known only one example of a class of codes (MDS codes) which are *t-proper* [7].

In this work $\bar{A}$, $\bar{\alpha}$ and $Q_{h,l}$ of all binary cyclic codes of lengths up to 33, ternary cyclic and negacyclic codes of lengths up to 20 and of some binary distance-optimal linear codes of lengths up to 33 have been computed with programmes written on C. To determine $\bar{A}$ we have to generate all the codewords of the code and to check their weights. To determine $\bar{\alpha}$ and $Q_{h,l}$ we can use some of the algebraic properties of linear codes and the following two methods are suitable:

Method 1: It is based on the fact that if $H = (h_1, h_2, \ldots, h_n)$ is any parity check matrix of C, then the covering radius $R(C)$ of the code is the smallest integer ρ such that every nonzero column vector of $n-k$ entries (where k is the dimension of the code) is a linear combination of not more than ρ columns of H. To obtain $\bar{\alpha}$ we have to make all the combinations of $\rho = t+1, \ldots, R(C)$ columns of H and for each value of ρ to count how many different vectors of $n-k$ entries have been obtained. We will note that it is not necessary to check values of ρ from 1 to t because we know that each $n-k$ dimensional vector of weight $1, \ldots, t$ is a unique coset leader.

The number of the steps required to find $\bar{\alpha}$ is $\sum_{i=t+1}^{R(C)} \binom{n}{i} 2^i$ and q^{n-k} words of storage are needed. Therefore, this method is suitable for codes with big dimensions.

When we have not enough memory to store these q^{n-k} words and when we would like to determine the values of $Q_{h,l}$ the following method can be used:

Method 2: It uses the definition of $R(C)$ as the weight of the coset leader of greatest weight. The weight of a coset leader is the minimum Hamming distance between any vector of the coset and all code vectors. For a code in a systematic form with generator matrix $G = [I|A]$, where I is the $k \times k$ identity matrix, a vector of each coset can be found by generating all vectors of the form $(\underbrace{0, \ldots, 0}_{k}, a)$, $a \in GF(3)^{n-k}$.

The number of steps for this method is proportional to nq^n and q^k words are needed to store the code in memory. These q^k words can not be stored in memory if the code is too long. In this case the code has to be generated once again for each vector.

V. Results. Classifications of binary cyclic and distance-optimal codes, and of ternary cyclic and negacyclic codes have been done in [6], [11], [1], [2] correspondingly. Using these classifications as a source $\bar{A}$, $\bar{\alpha}$ and $Q_{h,l}$ for all the codes have been computed by the methods from the previous section. Then their properness with any given precision as well as the discrete sufficient condition

can be checked in a linear time. Different codes can be compared and the best according to the undetected error probability or probability of correct decoding can be chosen.

We have used a programm written in Maple to check the monotony of $P_{ue}^{(t)}(C,p)$ for the investigated codes for a finite set of points of $p \in \left[0, \frac{p}{q-1}\right]$ with a step of 10^{-5} and in this way to determine all not *t-proper* codes. The results are presented in the tables below.

Table 1. Binary cyclic codes.

No	[n,k,d]	generator polynomial	proper
1.	$[7,4,3]^*$	1101	$t=0,1$
2.	[7,3,4]	10111	$t=0,1$
3.	$[9,3,3]^*$	1001001	$t=0,1$
4.	$[9,2,6]^{o*}$	11011011	$t=0,1,2$
5.	$[15,11,3]^{o*}$	11001	$t=0,1$
6.	$[15,10,4]^o$	101011	$t=0,1$
7.	[15,9,3]	1001111	$t=0,1$
8.	$[15,9,4]^o$	1011101	$t=0,1$
9.	$[15,8,4]^o$	11010001	$t=0,1$
10.	$[15,8,4]^o$	11100111	$t=0,1$
11.	$[15,7,3]^*$	110111011	$t=0$
12.	$[15,7,5]^{o*}$	100010111	$t=0,1,2$
13.	$[15,6,6]^o$	1011001101	$t=0,1,2$
14.	$[15,6,6]^{o*}$	1100111001	$t=0,1,2$
15.	$[15,5,7]^{o*}$	10000100001	$t=0,1,2,3$
16.	[15,4,6]	110001100011	$t=0,1$
17.	$[15,4,8]^o$	100110101111	$t=0,1,2,3$
18.	$[15,2,10]^{o*}$	11011011011011	$t=0,1,2,3,4$
19.	$[17,9,5]^{o*}$	100111001	$t=0,1,2$
20.	$[17,8,6]^o$	1101001011	$t=0,1,2$
21.	$[21,16,3]^{o*}$	100011	$t=0,1$
22.	$[21,15,3]^*$	1110101	$t=0,1$
23.	$[21,15,4]^o$	1100101	$t=0,1$
24.	$[21,14,4]^o$	0011111	$t=0,1$
25.	$[21,13,3]^*$	01001011	$t=0,1$
26.	$[21,13,4]^{o*}$	101111101	$t=0,1$
27.	[21,12,4]	1111011101	$t=0,1$
28.	$[21,12,5]^*$	1100110111	$t=0,1,2$
29.	$[21,11,6]^o$	10101011001	$t=0,1,2$
30.	[21,10,5]	100110000101	$t=0,1,2$

No	[n,k,d]	generator polynomial	proper
31.	[21,9,6]	1011001010011	$t = 0, 1, 2$
32.	$[21,9,8]^o$	1001001000001	$t = 0, 1, 2, 3$
33.	[21,8,6]	11101011110101	$t = 0, 1, 2$
34.	[21,8,6]	10110111101101	$t = 0, 1, 2$
35.	$[21,7,8]^o$	110001110111001	$t = 0, 1, 2, 3$
36.	$[21,6,7]^*$	10101100111011111	$t = 0, 1, 2, 3$
37.	$[21,6,8]^o$	10100100110010111	$t = 0, 1, 2, 3$
37.	$[21,5,10]^{o*}$	11111010100110001	$t = 0, 1, 2, 3, 4$
39.	$[21,4,9]^*$	110100011010001101	$t = 0, 1, 2, 3, 4$
40.	$[21,3,12]^o$	1011100101110010111	$t = 0, 1, 2, 3, 4, 5$
41.	$[21,2,14]^{o*}$	11011011011011011011	$t = 0, 1, 2, 3, 4, 5, 6$
42.	$[23,12,7]^{o*}$	110001110101	$t = 0, 1, 2, 3$
43.	$[23,11,8]^o$	10100100111111	$t = 0, 1, 2, 3$
44.	$[27,2,18]^{o*}$	110110110110110110110110111	$t = 0, 1, 2, 3, 4, 5, 6, 7, 8$
45.	$[31,26,3]^{o*}$	101001	$t = 0, 1$
46.	$[31,25,4]^o$	1111011	$t = 0, 1$
47.	$[31,21,5]^{o*}$	10110101101	$t = 0, 1, 2$
48.	$[31,21,5]^{o*}$	11001110101	$t = 0, 1, 2$
49.	$[31,21,5]^{o*}$	10010110111	$t = 0, 1, 2$
50.	$[31,20,6]^o$	111011110111	$t = 0, 1, 2$
51.	$[31,20,6]^o$	101010011111	$t = 0, 1, 2$
52.	$[31,20,6]^o$	110111011001	$t = 0, 1, 2$
53.	[31,16,5]	1001000011000111	$t = 0, 1, 2$
54.	[31,16,6]	1100011110110101	$t = 0, 1, 2$
55.	[31,16,7]	1101000100000001	$t = 0, 1, 2, 3$
56.	[31,16,7]	1001110000101101	$t = 0, 1, 2, 3$
57.	[31,15,6]	10100100011011111	$t = 0, 1, 2$
58.	$[31,15,8]^o$	10111001100000011	$t = 0, 1, 2, 3$
59.	$[31,15,8]^o$	11011000101001001	$t = 0, 1, 2, 3$
60.	$[31,15,8]^o$	11100111111001101	$t = 0, 1, 2, 3$
61.	$[31,11,11]^{o*}$	100001100101100111011	$t = 0, 1, 2, 3, 4, 5$
62.	$[31,11,11]^{o*}$	101010000011100110111	$t = 0, 1, 2, 3, 4, 5$
63.	[31,11,10]	111011001110000010101	$t = 0, 1, 2, 3, 4$
64.	$[31,10,12]^o$	1100010101110101001101	$t = 0, 1, 2, 3, 4, 5$
65.	$[31,10,12]^o$	1111110000100101011001	$t = 0, 1, 2, 3, 4, 5$
66.	[31,10,10]	1000111010000101110001	$t = 0, 1, 2, 3, 4$
67.	$[31,6,15]^{o*}$	11011001111010010101110001	$t = 0, 1, 2, 3, 4, 5, 6, 7$
68.	$[31,5,16]^{o*}$	10110101000111011111010011	$t = 0, 1, 2, 3, 4, 5, 6, 7$

In the first three tables cyclic and negacyclic codes are given. Only codes which are proper for error correction are included and their length, dimension, minimum

Table 2. Ternary cyclic codes.

No	[n,k,d]	generator polynomial	proper
1.	$[4,1,4]^{o*}$	1211	$t = 0, 1$
2.	$[8,5,3]^{o*}$	1011	$t = 0, 1$
3.	$[8,4,4]^{o}$	11012	$t = 0, 1$
4.	$[8,3,5]^{o}$	102111	$t = 0, 1, 2$
5.	[8,3,4]	120012	$t = 0, 1$
6.	[8,2,6]	1120221	$t = 0, 1, 2$
7.	[8,2,4]	1020102	$t = 0, 1$
8.	$[8,1,8]^{o*}$	12121212	$t = 0, 1, 2, 3$
9.	$[10,5,4]^{*}$	112122	$t = 0, 1$
10.	$[10,1,10]^{o*}$	1212121212	$t = 0, 1, 2, 3, 4$
11.	$[11,6,5]^{o*}$	102122	$t = 0, 1, 2$
12.	$[11,5,6]^{o}$	1222101	$t = 0, 1, 2$
13.	$[11,1,11]^{o*}$	11111111111	$t = 0, 1, 2, 3, 4, 5$
14.	$[13,10,3]^{o*}$	1112	$t = 0$
15.	$[13,9,3]^{o}$	10011	$t = 0$
16.	$[13,7,5]^{o*}$	1022201	$t = 0, 1, 2$
17.	$[13,7,4]^{*}$	1222121	$t = 0, 1$
18.	$[13,6,6]^{o}$	12200112	$t = 0, 1, 2$
19.	[13,6,6]	11002122	$t = 0, 1, 2$
20.	$[13,4,7]^{o}$	1120102201	$t = 0, 1, 2, 3$
21.	$[13,3,9]^{o*}$	10111220121	$t = 0, 1, 2, 3, 4$
22.	$[13,1,13]^{o*}$	1111111111111	$t = 0, 1, 2, 3, 4, 5, 6$
23.	$[14,1,14]^{o*}$	12121212121212	$t = 0, 1, 2, 3, 4, 5, 6$ g&p
24.	$[16,10,4]^{o*}$	1101121	$t = 0$
25.	$[16,9,5]^{o}$	10210122	$t = 0, 1$
26.	[16,8,5]	111210221	$t = 0, 1$
27.	$[16,7,6]^{o}$	1001222022	$t = 0, 1$
28.	[16,6,6]	11010112212	$t = 0, 1$
29.	$[16,3,10]^{o}$	10211100102111	$t = 0, 1, 2, 3, 4$
30.	$[16,2,12]^{o}$	112022101120221	$t = 0, 1, 2, 3, 4, 5$
31.	$[16,1,16]^{o*}$	1212121212121212	$t = 0, 1, 2, 3, 4, 5, 6, 7$ g&p
32.	[20,11,5]	1202000202	$t = 0$
33.	[20,9,6]	121102020102	$t = 0$
34.	[20,8,8]	1002122221122	$t = 0, 1$
35.	[20,7,8]	10022121211122	$t = 0, 1$
36.	$[20,6,10]^{o}$	112100101002221	$t = 0, 1, 3$
37.	[20,6,8]	122110102022112	$t = 0, 1, 3$
38.	[20,5,11]	1012201212020022	$t = 0, 1, 2$
39.	$[20,4,12]^{o}$	11101210002220212	$t = 0, 1, 2, 3$
40.	$[20,1,20]^{o*}$	12121212121212121212	$t = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9$

Table 3. Ternary Negacyclic Codes.

No	[n,k,d]	generator polynomial	good and proper
1.	[6,2,3]	10201	$t = 0, 1$
2.	$[10,6,4]^{o*}$	11021	$t = 0, 1$
3.	$[10,4,6]^{o}$	1110121	$t = 0, 1, 2$
4.	$[12,8,3]^{o*}$	12211	$t = 0, 1$
5.	[12,4,6]	122122211	$t = 0, 1, 2$
6.	$[12,2,9]^{o}$	12202110122	$t = 0, 1, 2, 3, 4$
7.	$[14,8,5]^{o*}$	1101011	$t = 0, 1, 2$
8.	$[14,6,6]^{o}$	111202111	$t = 0, 1, 2$
9.	[20,12,5]	112212211	$t = 1$
10.	[20,10,7]	12201101002	$t = 0$
11.	[20,10,6]*	11121011012	$t = 0, 1, 2$
12.	[20,8,8]	1122201022211	$t = 0, 1, 2$
13.	[20,6,9]	120121010211212	$t = 0, 1$
14.	[20,4,12]	11021100212101201	$t = 0, 1, 2, 3$
15.	[20,2,15]	1120221011202210112	$t = 0, 1, 2, 3, 4, 5, 6$

distance, generator polynomial and values of t for which the code is proper are presented. The generator polynomials are given as sequences of coefficients with the leading coefficient in the first place. With ^{o} distance-optimal codes are marked and with * the codes having the smallest possible covering radius among the codes with the given length and dimension. In the last table the results about binary distance-optimal codes are presented. It turned out that all the codes tested are proper. In addition their covering radii were determined.

As an illustration how the computed data can be used we present the graph (Fig. 1) of the undetected error probability of the $[21, 10, 4]$ binary cyclic

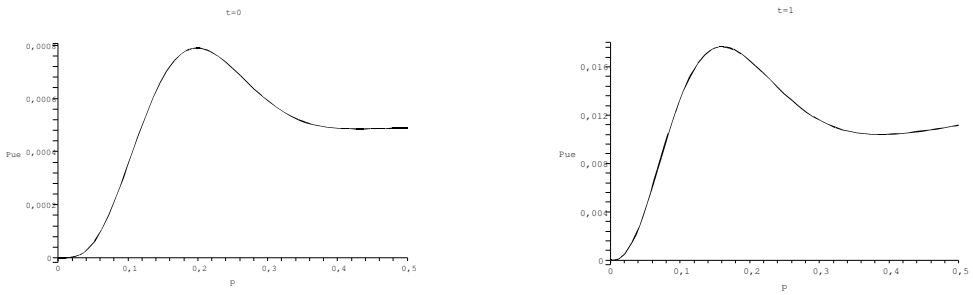

Fig. 1. P_{ue} of $[21, 10, 4]$ binary cyclic code.

Table 4. Binary distance-optimal codes

No	[n,k,d]	cov. radius	No	[n,k,d]	cov. radius
1.	[12,4,6]*	4	46.	[27,6,12]	11
2.	[16,11,4]*	2	47.	[27,6,12]	10
3.	[16,7,6]*	4	48.	[27,6,12]	10
4.	[16,7,6]*	4	49.	[27,6,12]	11
5.	[16,7,6]	5	50.	[27,6,12]	10
6.	[17,5,8]*	6	51.	[27,6,12]	10
7.	[17,5,8]	7	52.	[27,6,12]	11
8.	[18,9,6]	4	53.	[27,6,12]	11
9.	[18,6,8]	7	54.	[27,6,12]	10
10.	[18,6,8]	7	55.	[27,6,12]	11
11.	[19,7,8]	7	56.	[27,6,12]	10
12.	[20,8,8]	7	57.	[27,6,12]	11
13.	[22,10,8]	7	58.	[27,6,12]	10
14.	[23,12,7]*	3	59.	[27,6,12]	11
15.	[24,12,8]	7	60.	[27,6,12]	11
16.	[21,8,8]	6	61.	[27,6,12]	10
17.	[21,8,8]	7	62.	[27,6,12]	11
18.	[21,8,8]	7	63.	[27,6,12]	11
19.	[21,8,8]	7	64.	[27,6,12]	11
20.	[21,8,8]	7	65.	[28,10,10]	8
21.	[21,8,8]	7	66.	[28,10,10]	8
22.	[21,8,8]	8	67.	[28,10,10]	8
23.	[21,8,8]	8	68.	[28,10,10]	8
24.	[21,5,10]*	8	69.	[28,10,10]	8
25.	[24,14,6]	4	70.	[28,10,10]	8
26.	[24,7,10]	8	71.	[28,10,10]	8
27.	[24,7,10]	8	72.	[28,10,10]	8
28.	[24,7,10]	8	73.	[28,10,10]	8
29.	[24,7,10]	8	74.	[28,10,10]	8
30.	[24,7,10]	8	75.	[28,10,10]	8
31.	[24,7,10]	8	76.	[28,5,14]	12
32.	[24,5,12]	10	77.	[29,5,14]*	12
33.	[25,5,12]	11	78.	[29,5,14]*	12
34.	[25,5,12]	11	79.	[29,5,14]*	12
35.	[25,5,12]*	10	80.	[29,5,14]*	12

No	[n,k,d]	cov. radius	No	[n,k,d]	cov. radius
36.	[25,5,12]*	10	81.	[29,5,14]*	12
37.	[25,5,12]*	10	82.	[29,5,14]*	12
38.	[25,5,12]*	10	83.	[29,5,14]*	12
39.	[25,5,12]*	10	84.	[30,6,14]	11
40.	[26,6,12]	11	85.	[30,6,14]	11
41.	[26,6,12]	11	86.	[31,13,9]	7
42.	[27,7,12]	10	87.	[32,17,8]	6
43.	[27,6,12]	11	88.	[32,6,16]	12
44.	[27,6,12]	11	89.	[33,8,14]	11
45.	[27,6,12]	11	90.	[33,12,11]	9

code. The code has minimum distance 4 and can correct one error. Therefore $t = 0, 1$. It is clear from the graph that this code is neither $t = 0$ nor $t = 1$ proper if we would like to use it in the whole range of bit error probabilities, i.e. $p \in [0, \frac{1}{2}]$. Very often in practice, we are interested in using the code in a subinterval in the interval $p \in \left[0, \dfrac{q-1}{q}\right]$. Discrete sufficient condition for a binary code to be proper in the subinterval $[a, \frac{1}{2}]$ of the interval $[0, \frac{1}{2}]$ is derived in [5]. According to the results obtained in this work we can make conclusions about properness of the code also in the subinterval $[0, b]$. Namely, if the $[21, 10, 4]$ binary cyclic code is used for error detection in the interval $p \in [0, 0.21]$ and for 1-error correction in the interval $p \in [0, 0.16]$ it is proper.

On the next graph (Fig. 2) an example of $t = 0, 1, 2$ proper in the whole interval $[0, 1/2]$ binary cyclic $[21, 10, 5]$ code is given.

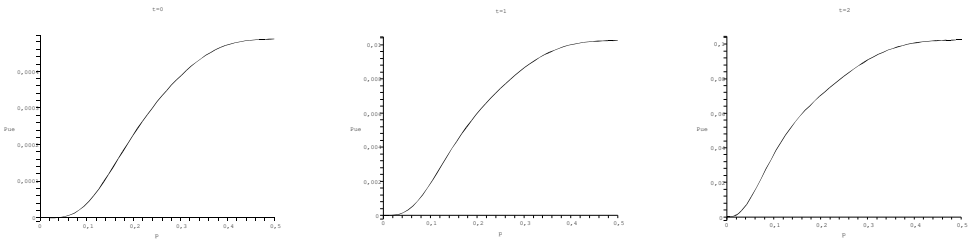

Fig. 2. P_{ue} of $[21, 10, 5]$ binary cyclic code

As a last example, comparison between the probability of correct decoding between two binary $[25, 5, 12]$ distance-optimal codes is presented (Fig. 3).

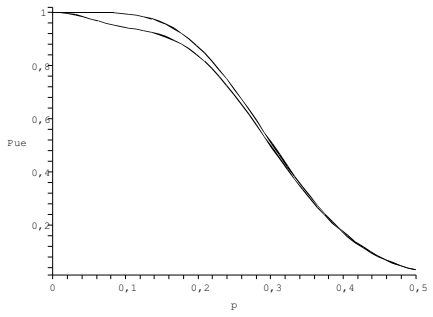

Fig. 3. P_{corr} of two binary $[25, 5, 12]$ distance-optimal codes

VI. Acknowledgement. This work was supported by "Finite Structures" Marie Curie Host Fellowship for the Transfer of Knowledge project carried out by the Alfred Rényi Institute of Mathematics in the framework of the European Community's Structuring the European Research Area programme.

REFERENCES

[1] Baicheva T. The Covering Radius of Ternary Cyclic Codes with Length up to 25. *Des. Codes Cryptogr.* **13** (1998), 223–227.

[2] Baicheva T. On the Covering Radius of Ternary Negacyclic Codes with Length up to 26. *IEEE Trans. Inform. Theory* **47** (2001), 413–416.

[3] Berlekamp E. R., R. J. McEliece, H. C. A. van Tilborg. On the inherent intractability of certain coding problems. *IEEE Trans. Inform. Theory* **24** (1978), 384–386.

[4] Dodunekova R., S. Dodunekov. Sufficient conditions for good and proper linear error correcting codes. Proc. Second International Workshop on Optimal Codes and Related Topics, Sozopol, Bulgaria, 1998, 62–67.

[5] Dodunekova R., E. Nikolova. Sufficient condotions for the monotonicity of the undetected error probability for large channel error probabilities. *Problemy Peredachi Informatsii* **41** (2005), 3–16 (in Russian); English translation: *Problems Inform. Transmission* **41** (2005), 187–198.

[6] Downie D., N. J. A. Sloane. The Covering Radius of Cyclic Codes of Length up to 31. *IEEE Trans. Inform. Theory* **31** (1985), 446–447.

[7] Kasami T., S. Lin. On the probability of undetected error for the Maximum Distance Separable codes. *IEEE Trans. Communications* **32** (1984), 998–1006.

[8] Klöve T., V. Korzhik. Error Detecting Codes. Boston, Kluwer Academic Publishers, 1995.

[9] MacWilliams F. J. A theorem on the distribution of weights in a systematic code. *Bell System Tech. J.* **42** (1963), 79–94.

[10] McLoughlin A. The complexity of computing the covering radius of a code. *IEEE Trans. Inform. Theory* **30** (1984), 800–804.

[11] Jaffe D. Binary Linear Codes: New Results on Nonexistence. Draft (Version 0.4), Department of Mathematics and Statistics, University of Nebraska, April 14, 1997.

Institute of Mathematics and Informatics
Bulgarian Academy of Sciences
P.O.Box 323
5000 Veliko Tarnovo, Bulgaria
e-mail: tsonka@moi.math.bas.bg

Received October 23, 2006
Final Accepted June 27, 2007