G.U. Journal of Science
20(3): 127-132 (2007)

GUJS
www.gujs.org

A Simulation Study on Decoding of Binary Linear Codes Using Projective Geometry PG(3,2)

Fikri GÖKPINAR ♠

Gazi University, Faculty of Arts and Sciences, Department of Statistics, Ankara, Turkey

Received: 20.09.2005 Revised: 04.08.2007 Accepted: 17.09.2007

ABSTRACT

Projective geometry is used to decode and represent codes easily. Cameron [1] generated a binary linear code from PG(2,2). In this paper we construct a binary linear code from PG(3,2). Also we give a decoding rule for this code. A simulation study is given to compare this decoding algorithm with maximum likelihood decoding algorithm.

Key Words: Projective geometry, Binary linear codes, Error-correcting codes, Decoding

1. INTRODUCTION

In coding theory, there are several methods decoding of linear codes. Maximum likelihood(ML) decoding, syndrome decoding are the most important of these decoding technique. One of the most important ways is to use projective geometry(PG). This method is very useful and easy. The most immediate way in which a linear code can be associated with a design or, indeed, with any incidence structure. Assmus and Key [2] give a coding theoretic approach to Hadamard matrices and their designs. Assmus[3] gives a relation between affine plane and codes. Assmus [4] investigates minimum weight vectors in code generated by the incidence matrix of a finite affine plane. Codes are used in many practical situations. Few of these are as following.

Error-correction code is used for getting pictures and data about the Solar System back to earth. Codes also are used combinatorial search problem, and human genome project.

In second part of this paper, basic definitions of coding theory and projective geometry were given. In third part, we gave construction and decoding rules of codes from PG(3,2). Also we gave relation between parameters of projective geometry and binary linear codes. In forth section, decoding algorithm using projective geometry is compared with maximum likelihood decoding algorithm using different error rate.

2. BASIC DEFINITIONS OF CODING THEORY AND PROJECTIVE GEOMETRY

The object of coding theory is the transmission of messages over noisy channels. In here, the basic definitions of coding theory are given.

Definition 2.1: [5] Let A be a finite set of v elements (the alphabet). A v-ary code C of length n is a family of n-tuples with entries in A as follows

$C \subseteq A_n$.

Definition 2.2: [6] Let $x=(x_1,x_2,\ldots,x_n)$, $y=(y_1,y_2,\ldots,y_n) \in A^n$. The (Hamming) distance between x and y is

$d(x,y)$=number of i such that $x_i \neq y_i$.

The minimum distance d(C) of code $C \subseteq A^n$ is the minimum of the Hamming distances d(x,y), where $x,y \in C$, $x \neq y$.

It is easy to see that the Hamming distance is a metric. Its meaning in our setting is obvious: if $x \in A_n$ is sent and $y \in A_n$ is received, then at least d(x,y) errors must have occurred, and the most probable explanation is that precisely d(x,y) errors have occurred. Here is how codes are used for information transmission: T

♠Corresponding author, e-mail: fikri@gazi.edu.tr

(Transmitter) and R (Receiver) agree on a code $C \subseteq A_n$ of minimum distance d. The tuples of C are called code words. Naturally there must be at least as many code words as source states. Consider the case d=3. Let $y=(y_1,y_2,\ldots,y_n)$ be the received vector. Observe that R knows a codeword x was sent. If $y \in C$, then R will decode $y \rightarrow x=y$. Moreover this will be correct unless at least 3 errors have occurred. Assume $y \notin C$. Then R will search for a code word x at distance 1 from y. There can be at most one such word (if X' was a second such code word, then $d(x, X') \leq 2$ contradiction). If such a word exists, then R will decode $y \rightarrow x$. The important observation is that x is the word having been sent unless more than one error occurred under transmission. So a code with minimum distance 3 corrects one error. By the same reasoning, if d>2e then a code with minimum distance d corrects e errors. The injective mapping of source states into codewords is called the encoding.

Definition 2.3 (linear codes): [7] Let $\mathbf{F}_q$ be the field with q elements. A q-ary linear code of length n and dimension k is a linear subspace $C \subseteq \mathbf{F}_q^n$ of vector space dimension k. If its minimum distance is d, then we record the parameters of C as;

$[n,k,d]_q$.

Observe that a q-ary linear code of dimension k has q^k code words. It contains the all 0-word 0=(0,0…0).

For fixed q we wish to construct codes [n,k,d] with large d, large k and small n. Recall that a code with minimum distance d allows the correction of e transmission errors, when 2e<d.

d(x, y)=d(x-z, y –z) from the definition of Hamming distance. If x, y are code words of a linear code, we get d(x,y)=d(0, y-x) and y-x is a code word, because of linearity. Define the weight wt(x) as the number of coordinates with a nonzero entry, in other words wt(x)=d(x,0). Then the minimum distance of a linear code equals the minimum weight of its nonzero code words.

Projective geometry can be defined as following definition.

Definition 2.5.(projective geometry) [8] Let V be an (n+1)-dimensional vectorspace over $\mathbf{F}_q$. Call the (i+1)-dimensional subspaces of V the i-flats of geometry. Here i=1,2,...n . Incidence is defined by inclusion. That is to say flats a and b are called incident if either $a \subset b$ or $b \subset a$. These flats and their incidence relation define the n-dimensional projective geometry PG(n,q) of order q:

0-flats are points (1-dimensional subspaces), 1-flats are lines (2-dimensional subspaces)…and (n-1)-flats are hyperplanes (n-dimensional subspaces).

Definition 2.6. (Gauss polynomial) [8] The number of i-flats of PG(n,q) is

$$\begin{bmatrix} n+1 \\ i+1 \end{bmatrix} = \frac{\left(q^{n+1}-1\right)\left(q^{n+1}-q\right)..\left(q^{n+1}-q^i\right)}{\left(q^{i+1}-1\right)\left(q^{i+1}-q\right)...\left(q^{i+1}-q^i\right)}$$

In particular the number of points and the number of hyperplanes is $\left(q^{n+1}-1\right)/(q-1)$.

3. DECODING OF BINARY LINEAR CODES USING PROJECTIVE GEOMETRY

Using PG(n,2), we can obtain binary linear codes and its parameters as follows:

First we write all points in base 2, coordinates of points on any hyperplanes up to 0(where addition is binary).

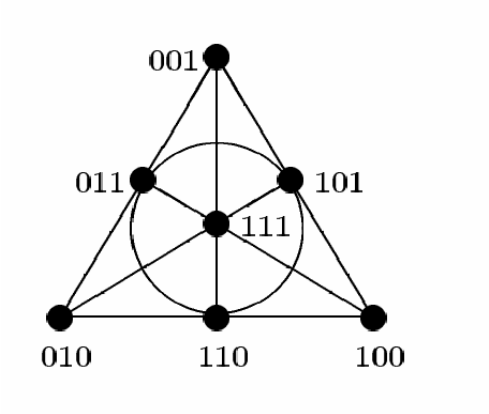
Now we can construct a binary linear code. As known before, there is 2^{n+1}-1 points and hyperplane, every hyperplane of PG(n,2)has 2^n-1 points, every point of PG(n,2) pass 2^n-1 hyperplanes, every pair of points occur in 2^{n-1}-1 distinct hyperplanes and every pair of hyperplanes meet 2^{n-1}-1 distinct points.

There are 2^{n+2} codewords in binary linear codes which obtained from PG(n,2). Length of this code is 2^{n+1}-1. This code's minimum distance d is 2^n-1 so this code can correct maximum 2^{n-1}-1 errors.

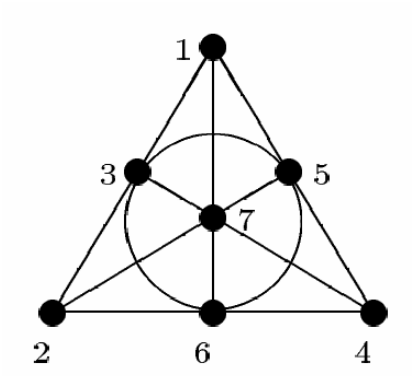
From these informations and definition 2.2 and definition 2.3, codes can be obtained as follows. Binary linear code must include 2^{n+1}-1 codeword which weights' 2^n-1 and 2^{n+1}-1 codeword which weights' 2^n in other word these 2^{n+1}-1 codeword has 2^n-1 0s. One of the other 2 codewords is all 0s and the other one is all 1s. In 2^n-1 weighted codewords, 1s' place is determined by points of hyperplane. In 2^n-weighted codewords, 0s' place is determined by points of hyperplane.

Cameron[1] generated a binary linear code from PG(2,2). For PG(2,2), parameters of this plane and generating code from this plane are as follows.

This plane has 7 points and 7 lines and it is shown at figure 1. The code has 16 codeword, lengt of codewords is 7. Distance is 3 and error correcting number e=1.



a) PG(2,2) with all points in base 2



b) points of PG(2,2) which corresponding to (a)

Figure 1. Point and lines of PG(2,2).

The code has 16 words. One is all-zeros, and one is all-ones. Of the others, seven have three zeros and four ones, and the lines of the plane give the positions of the three zeros. The other seven have three ones and four zeros, and the lines of the plane give the positions of the three ones. So Cameron[1] gives decoding rules as follows:

Look at the received word. If they are all 0s, then no error was occurred. If there was just one 1, then it was the error. If there were two 1s in positions *i* and *j*, find the third point *k* on the line through *i* and *j*; this is the position of the error. If there are three 1s which form a line of the plane, no error has occur. If there are three 1s which do not form a line, then the positions of the zeros contain just one line; the odd point out is the error. If there are more 1s than 0s, just reverse zeros and ones.

Table 1: Code which is generated from PG(2,2).

1	0 0 0 0 0 0 0
2	0 0 0 1 1 1 1
3	0 0 1 0 1 1 0
4	0 0 1 1 0 0 1
5	0 1 0 0 1 0 1
6	0 1 0 1 0 1 0
7	0 1 1 0 0 1 1
8	0 1 1 1 1 0 0
9	1 0 0 0 0 1 1
10	1 0 0 1 1 0 0
11	1 0 1 0 1 0 1
12	1 0 1 1 0 1 0
13	1 1 0 0 1 1 0
14	1 1 0 1 0 0 1
15	1 1 1 0 0 0 0
16	1 1 1 1 1 1 1

We generate a binary linear code from PG(3,2). PG(3,2) has 15 planes and 15 points. Binary linear code generated from PG(3,2) has 32 codeword, minimum distance d=7, error correcting number e=3, and length of codeword of this code is 15. Planes of PG(3,2) which satisfy addition rule(coordinates of points on any plane up to 0) is at Table 2 [9].

Table 2: Planes of PG(3,2).

Planes	Points
1	1,2,3,4,5,6,7
2	1,2,3,8,9,10,11
3	1,2,3,12,13,14,15
4	1,4,5,8,9,14,15
5	1,4,5,10,11,12,13
6	1,6,7,8,9,12,13
7	1,6,7,10,11,14,15
8	2,4,6,8,10,13,15
9	2,4,6,9,11,12,14
10	2,5,7,8,10,12,14
11	2,5,7,9,11,13,15
12	3,4,7,8,11,13,14
13	3,4,7,9,10,12,15
14	3,5,6,8,11,12,15
15	3,5,6,9,10,13,14

So, codewords of code which satisfy conditions that give above, are at Table 3.

Table 3: Code which is generated from PG(3,2).

1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
2	1	0	0	0	0	1	1	1	1	0	0	1	1	0	0
3	0	1	0	0	1	0	1	1	0	1	0	1	0	1	0
4	0	0	1	0	1	1	0	1	0	0	1	1	0	0	1
5	0	1	0	1	0	1	0	0	1	0	1	1	0	1	0
6	1	0	0	1	1	0	0	1	1	0	0	0	0	1	1
7	0	1	0	0	1	0	1	0	1	0	1	0	1	0	1
8	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0
9	1	1	1	0	0	0	0	0	0	0	0	1	1	1	1
10	0	1	0	1	0	1	0	1	0	1	0	0	1	0	1
11	1	0	0	1	1	0	0	0	0	1	1	1	1	0	0
12	0	0	1	0	1	1	0	0	1	1	0	0	1	1	0
13	0	0	1	1	0	0	1	1	0	0	1	0	1	1	0
14	0	0	1	1	0	0	1	0	1	1	0	1	0	0	1
15	1	1	1	0	0	0	0	1	1	1	1	0	0	0	0
16	1	0	0	0	0	1	1	0	0	1	1	0	0	1	1
17	0	1	1	1	1	0	0	0	0	1	1	0	0	1	1
18	1	0	1	1	0	1	0	0	1	0	1	0	1	0	1
19	1	1	0	1	0	0	1	0	1	1	0	0	1	1	0
20	1	0	1	0	1	0	1	1	0	1	0	0	1	0	1
21	0	1	1	0	0	1	1	0	0	1	1	1	1	0	0
22	1	0	1	1	0	1	0	1	0	1	0	1	0	1	0
23	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
24	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0
25	1	0	1	0	1	0	1	0	1	0	1	1	0	1	0
26	0	1	1	0	0	1	1	1	1	0	0	0	0	1	1
27	1	1	0	1	0	0	1	1	0	0	1	1	0	0	1
28	1	1	0	0	1	1	0	0	1	1	0	1	0	0	1
29	1	1	0	0	1	1	0	1	0	0	1	0	1	1	0
30	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
31	0	1	1	1	1	0	0	1	1	0	0	1	1	0	0
32	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1

Decoding rule of this code is as follows:

Look at the received word. If they are all 0s, then no error was occurred. If there were one, two or three 1, then there were the errors.

If there were four 1s and they are on a plane then find the other points on the plane. These are the errors. But if there aren't on a plane then there are more than three errors so we can't correct this codeword then we want new transmission.

If there were five 1s and they are on a plane then find the other points on the plane. These are the errors. If there aren't on a plane then look at the place of 0s; if they include a plane then the other 0s are the errors. On the other cases, we want new transmission.

If there are six 1s and they are on a plane then find the other point on the plane. This is the error. If five of them are on a plane then the other one and other two digits of plane are the errors. But If these six 1s aren't on a plane then look at the place of 0s; if 0s include a plane then the other 0s are the errors. On the other cases, we want new transmission.

If there are seven 1s and they are on a plane then this is a true transmission. But if six of them are on a plane then the other one and other digit of plane are the errors. If there aren't on a plane then look at the place of 0s; if they include a plane then the other 0 is the error. But if six of them are on a plane then the other one and other one digit of plane are the errors. On the other cases, we want new transmission.

If there are more 1s then 0s then just reverse zeros and ones and apply the rules that were given above.

4. SIMULATION STUDY

In this section, to compare ML decoding and PG decoding, we generated different size of numbers from 1-32. All numbers are matched with a codeword. This number and corresponding codewords are given at table 3. For example we use codeword 00..0 for 1 and 11…1 codeword for 32, and the other numbers between 2 and 31. First we encode the numbers to the corresponding codewords and send these codewords then under different error rate we encode the codewords. From section 3, code has parameters $(15,5,7)_2$ so it can corrects up to 3 errors. 1000(1000)100000 codewords are generated under 0.05, 0.10, 0.15 error rates and Figures 4.1-4.3 are given.

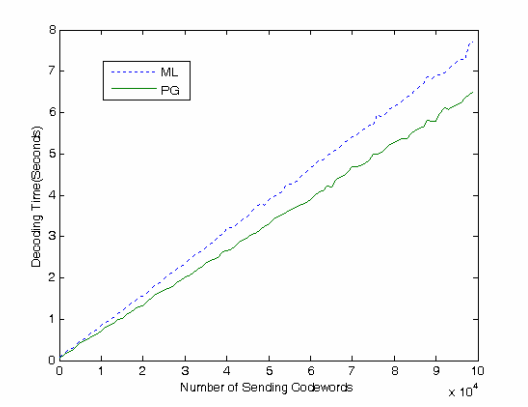


Figure 4.1. Decoding times for error rate 0.05.

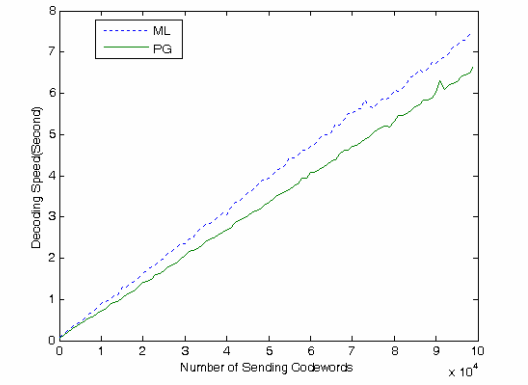


Figure 4.2. Decoding times for error rate 0.10.

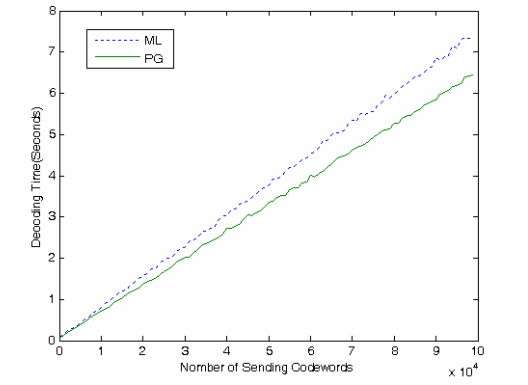


Figure 4.3. Decoding times for error rate 0.05.

Also for this code encoding and sending time are give at Figures 4.4 and 4.5 respectively.

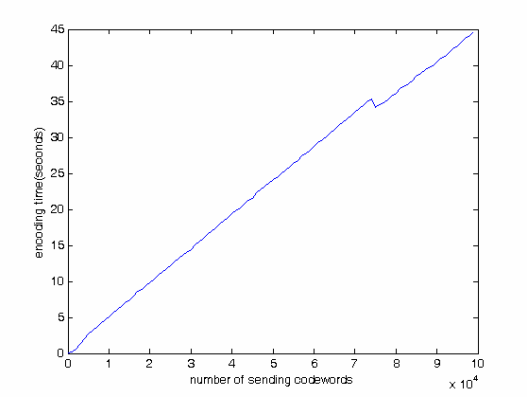


Figure 4.4. Encoding times for the code from PG(3,2).

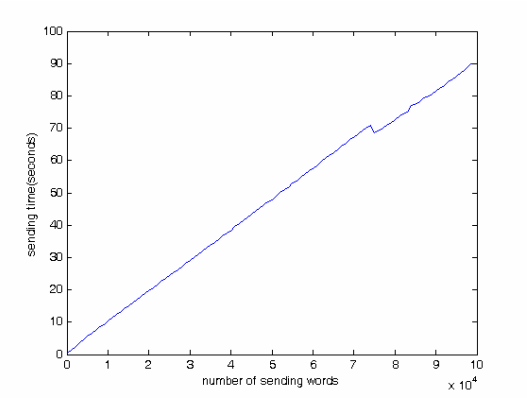


Figure 4.5. Encoding times the code from PG(3,2).

From these results, ML and PG decoding techniques are not slow but PG is much faster than ML. The difference between these techniques is about %20 for all error rates. In greater number of sending words, this difference is getting much more. Also constructing of linear codes from is easier than the other constructing techniques. For 0.05 error rate, total true decoding ratio is 0.995, for 0.10 error rate, total true decoding ratio is 0.95 and for 0.15 error rate, total true decoding ratio is 0.85. So this code can be used up to 0.10 error rate. Sending and encoding times are same for ML and PG decoding.

4. CONCLUSION

Using projective geometry, constructing of binary linear codes is simple. Also decoding this kind of codes is faster than the most of the other decoding methods. Also codes from Projective Geometry can be generalized for greater dimension.

REFERENCES

[1] Cameron, P. J., "Combinatorics: Topics, Techniques, Algorithms", ***Cambridge University Press***, (1996).

[2] Assmus, E. F., Key, J. D., "Hadamard Matrices And Their Design", ***Transactions of the American Mathematical Society***, 330: 269-293 (1989).

[3] Assmus, E. F., "On the theory of designs, Surveys in Combinatorics", 141, ***Cambridge University Press***, 1-21 (1989).

[4] Assmus, E. F., "The coding theory of finite geometries and designs", ***Lecture Notes in Computer Science***, Edited by T. Mora, Springer-Verlag, 357: 1-6 (1998).

[5] Xambo- Descamps, S., "Block error-Correcting Codes: A computational premier", ***Springer-Verlag***, New York, 112-114 (2003).

[6] Garret, P., "The Mathematics of coding theory", ***Prentice Hall***, New York (2004).

[7] Wicker, S. B, Kim, S., "Fundementals of codes, graphs and iterative decoding", ***Kluvwer Academic Publishers***, London, 78-80 (2003).

[8] Beutelspacher, A., Rosenbaum, U., "Projective Geometry". ***Cambridge University Press***, London, 25-30 (1998).

[9] Raghavarao, D., "Constructions and Combinatorial Problems in Design of Experiments", ***Dover***, New York, 56-57 (1971).