Contemporary Engineering Sciences, Vol. 9, 2016, no. 13, 609 - 617
HIKARI Ltd, www.m-hikari.com
http://dx.doi.org/10.12988/ces.2016.6448

Efficient Calculation of the Weight Distributions for Linear Codes over Large Finite Fields

Sunghyu Han

School of Liberal Arts & HRD
KoreaTech, South Korea

Hee Suk Seo

Department of Computer Science and Engineering
KoreaTech, South Korea
Corresponding author

Seunghwan Ju

Department of Computer Science and Engineering
KoreaTech, South Korea

Abstract

In an error correcting code, the performance of a specific linear code is dependent on the minimum weight and the weight distribution of the code. Therefore it is very importance to know or calculate the minimum weight and the weight distribution of a code. In this paper we propose a very efficient calculation method for the minimum weight and the weight distribution for linear codes over large finite fields. Computation results show that our algorithm is much faster than Magma, the computational algebra system, for the minimum weight and the weight distribution of specific random linear codes over large finite fields.

Keywords: Linear codes, Minimum weight, Weight distributions

1 Introduction

Error correcting codes are very important in data communication. There are various types of error correcting codes. Among them, linear codes over finite fields are basic and fundamental. Also they have strong application in data communication. The performance of a specific linear code is dependent on the minimum weight and the weight distribution of the code.

In 1978, Berlekamp, McEliece and van Tilborg proved that the computation of weight distribution of linear codes over GF(2) is NP-complete [1], and in 1997, Alexander Vardy proved that the computation of minimum weight of linear codes over GF(2) is also NPcomplete [11]. Therefore, we cannot expect a polynomial-time algorithm for these computation unless P = NP.

Many scientists investigated the problem of the computation for minimum weight and the weight distribution. A lot of them were not published and used in personal purpose. A. E. Brouwer and K.-H. Zimmerman proposed an algorithm for minimum weight and the algorithm is implemented in Magma [2, 3]. They used many different generator matrices. In [10], Leon also proposed an algorithm for minimum weight, which is probabilistic, and the algorithm is used in GUAVA [6]. In [4], I. Bouyukliev and V. Bakoev proposed an algorithm for the weight distribution using a sequence of different generator matrices. In [9], S. Han proposed an algorithm for the weight distribution and using the algorithm calculated the weight distribution of the projections of the 2-adic Golay code of length 24 to Z_2^e.

The purpose of this paper is to suggest an efficient calculation method for the minimum weight and the weight distribution of linear codes over finite fields. The key idea of our algorithm is based on the idea in [9]. In this paper, we investigate the complexity of the algorithm, and apply the algorithm to random linear codes over large finite fields. Our calculation results show that our method is much faster than Magma [2] for random linear codes over large finite fields.

This paper is organized in the following way. First we describe the previous well known basic method in Previous Method, and our algorithm is given in Proposed Method with detailed complexity discussion of the algorithm. In Computation time, we give calculation results of our algorithm and Magma. All the computations of this paper were done using a laptop (2GHz, 8GB RAM) and Magma(V2.19-4).

2 Previous Method

2.1 Method 1

Let C be an [n, k] linear code over F_q with a k × n generator matrix G. Our object is to calculate the number of codewords of C with given weight w. To do this, we consider the following linear combinations.

$$c = a_1 G_1 + a_2 G_2 + \cdots + a_k G_k \tag{1}$$

Where $a_i \in F_q$ and G_i is the i-th row of G, $(1 \le i \le k)$. We only have to calculate the codewords in Eqn. (1) of weight w. The number of linear combinations of Eqn. (1) is

$$q^k \tag{2}$$

For each linear combination, we require nk multiplications and n(k − 1) additions over F_q. Therefore the total multiplication and total additions are $q^k nk$, $q^k n(k-1)$, respectively.

2.2 Method 2

Above method can be improved if we using a standard generator matrix. We can obtain G′ = [I|A], where I is a k×k identity matrix, using a Gauss-Jordan elimination and column permutation. Let C′ be the linear code generated by G′. Then C and C′ are equivalent and have the same weight distribution. Therefore without loss of generality, we can assume that C has a standard generator G = [I|A]. Let's consider the following linear combinations

$$c = a_1 G_{i_1} + a_2 G_{i_2} + \cdots + a_t G_{i_t}, \tag{3}$$

where $a_j \in F^*_q$, $(F^*_q = F_q - \{0\}, 1 \le j \le t)$, $1 \le i_1 < i_2 < \cdots < i_t \le k$, and $t \le \min\{k,w\}$. Let $c = (u_1|u_2)$, where u_1 is the first k coordinates and u_2 is the last n−k coordinates of c. Since $a_i \in F_q^*$, the weight of u_1 is t. Therefore we have to count the codeword c of weight $wt(u_2) = w-t$, where $wt(u_2)$ means the weight of u_2. This fact leads to the following linear combinations.

$$u_2 = a_1 A_{i_1} + a_2 A_{i_2} + \cdots + a_t A_{i_t}, \tag{4}$$

where $a_j \in F^*_q$, $(1 \le j \le t)$, A_j is the j-th row of A. We only have to calculate the number of u_2 in Eqn. (4) of weight w − t. The total number of linear combinations in Eqn. (4) is

$$(q-1)\binom{k}{1} + (q-1)^2\binom{k}{2} + \cdots + (q-1)^{\min\{k,w\}}\binom{k}{\min\{k,w\}} = \sum_{1 \le t \le \min\{k,w\}} (q-1)^t \binom{k}{t} \tag{5}$$

We can reduce the number of linear combinations in Eqn. (5) by the following observation. Let

$$u_2' = \frac{1}{a_1} u_2 = A_{i_1} + \frac{a_2}{a_1} A_{i_2} + \cdots + \frac{a_t}{a_1} A_{i_t} \tag{6}$$

Then the weight of u'_2 is the same as the weight of u_2. Therefore in Eqn. (4), we restrict $a_1 = 1$, and then calculate the number of codewords u_2 of weight w-t, and then we multiply the number by q-1. This fact leads to the following computational complexity for the number of linear combinations.

$$\binom{k}{1}+(q-1)\binom{k}{2}+\cdots+(q-1)^{\min\{k,w\}-1}\binom{k}{\min\{k,w\}}=\sum_{1\le t\le\min\{k,w\}}(q-1)^{t-1}\binom{k}{t} \tag{7}$$

Now we calculate the number of multiplication and addition of F_q in Eqn. (4) with $a_1 = 1$. The number is (t-1)*(n-k) for both multiplication and addition. Using this fact and Eqn. (7) we have the total number of multiplication and the total number of addition over F_q by the following.

$$(n-k)\sum_{1\le t\le\min\{k,w\}}(q-1)^{t-1}(t-1)\binom{k}{t} \tag{8}$$

3 Proposed method

3.1 Method 3

Let C be an [n, k] linear code over F_q with a k × n generator matrix G. Our object is to calculate the number of codewords of C with given weight w. First, we consider the null space of each column vector of G.

$$Z_i = Null((G^i)^T) = \{v \in F_q^k \mid vG^i = 0\},\ (i = 1,2,\cdots,n) \tag{9}$$

where G^i is the i-th column vector of G. The number of codewords of C with given weight w is given by the following formula.

$$f(G,w) = \sum_{I\subseteq\{1,2,\cdots,n\},|I|=n-w} \left|\bigcap_{j\in I} Z_j - \bigcup_{j\notin I} Z_j\right| \tag{10}$$

Note that if $v \in (\cap_{j\in I} Z_j - \cup_{j\notin I} Z_j)$ and c = vG = (c_1, c2,…, c_n) then $c_j = 0$ for j ∈ I and $c_j \neq 0$ for j ∉ I. Therefore wt(c) = w. In other words, in Eqn. (10), $\cap_{j\in I} Z_j$ corresponds to the zero components of a codeword and $\cup_{j\notin I} Z_j$ corresponds to the nonzero components of a codeword. The number of zero components is |I| = n-w and the number of nonzero components is $|I^C|$ = w. Therefore the weight of a codeword is w.

The number of terms of the summation in Eqn. (10) is

$$\binom{n}{n-w}=\binom{n}{w} \tag{11}$$

For each term of the summation, we have to calculate the value $|\cap_{j\in I} Z_j - \cup_{j\notin I} Z_j|$. In the first, $\cap_{j\in I} Z_j$ can be calculated by Gauss-Jordan Elimination. The complexity of multiplication and addition of Gauss-Jordan Elimination is $\frac{1}{3}k^3 + k^2 + \frac{1}{3}k$, $\frac{1}{3}k^3 + \frac{1}{2}k^2 - \frac{5}{6}k$, respectively, for k × k matrix. For each element v ∈ $\cap_{j\in I} Z_j$ we have to check that $v \notin Z_j$ for all $j \in I^C$. This can be done by checking $vG^j \neq 0$ for $j \in I^C$. Therefore the number of multiplication is k for each $j \in I^C$ and

the number of addition is k-1 for each $j \in f$. Since $|I^C|$ is w, the the number of multiplication and addition is kw and (k-1)w, respectively. The complexity depends on the number of elements of $\cap_{j\in I} Z_j$, i.e., $|\cap_{j\in I} Z_j|$. This value depends on the rank of G^I, where G^I is the column submatrix of G consists of the G^j, $j \in I$. Let r(I) be the rank of G^I. Then $|\cap_{j\in I} Z_j| = q^{k-r(I)}$. For checking $vG^j \neq 0$, we only have to check for a normalized vector v, i.e., the first nonzero component is 1, the total multiplication is $\frac{q^{k-r(I)}-1}{q-1} \times kw$ and the total addition is $\frac{q^{k-r(I)}-1}{q-1} \times (k-1)w$. Finally, the number of multiplication of f(G, w) is

$$\sum_{I\subseteq\{1,2,\ldots,n\},|I|=n-w} \left((\frac{1}{3}k^3 + k^2 + \frac{1}{3}k) + \frac{q^{k-r(I)}-1}{q-1} \times kw \right). \tag{12}$$

the number of addition of f(G, w) is

$$\sum_{I\subseteq\{1,2,\ldots,n\},|I|=n-w} \left((\frac{1}{3}k^3 + k^2 - \frac{5}{6}k) + \frac{q^{k-r(I)}-1}{q-1} \times (k-1)w \right). \tag{13}$$

3.2 Method 4

Combining Method 2 and Method 3, we give Method 4. We start with Method 2. Let's consider the following linear combinations

$$c = a_1 G_{i_1} + a_2 G_{i_2} + \cdots + a_t G_{i_t}, \tag{14}$$

where $a_j \in F^*_q$, ($F^*_q = F_q - \{0\}$, $1 \le j \le t$), $1 \le i_1 < i_2 < \cdots < i_t \le k$, and $t \le \min\{k,w\}$. Let $c = (u_1|u_2)$, where u_1 is the first k coordinates and u_2 is the last n−k coordinates of c. Since $a_i \in F_q^*$, the weight of u_1 is t. Therefore we have to count the codeword c of weight $wt(u_2) = w-t$, where $wt(u_2)$ means the weight of u_2. This fact leads to the following linear combinations.

$$u_2 = a_1 A_{i_1} + a_2 A_{i_2} + \cdots + a_t A_{i_t}, \tag{15}$$

where $a_j \in F^*_q$, ($1 \le j \le t$), A_j is the j-th row of A. We only have to calculate the number of u_2 in Eqn. (15) of weight w − t. Now we follow Method 3. The number of u_2 in Eqn. (58) of weight w-t can be written by the following formula.

$$f(A(\{i_1, i_2, \ldots, i_t\}), w-t) = \sum_{I\subseteq\{1,2,\cdots,n-k\},|I|=(n-k)-(w-t)} |\bigcap_{j\in I} Z_j^* - \bigcup_{j\notin I} Z_j^*| \tag{16}$$

where $A(\{i_1, i_2,\ldots, i_t\})$ be the t × (n-k) submatrix of A with i_j ($1\le j\le t$) rows of A and $Z_j^* = \{v \in (F_q^*)^t \mid vA(\{i_1, i_2, \ldots, i_t\})^j = 0\}$, (j = 1,2,…,n-k) and $A(\{i_1, i_2,\ldots, i_t\})^j$ is the j-th column vector of $A(\{i_1, i_2,\ldots, i_t\})$.

Then the number of codewords of weight w in C is

$$f(G,w) = \sum_{1 \le t \le \min\{k,w\}} \sum_{\{i_1, i_2, \ldots i_t\} \in S(k,t)} f(A\{\{i_1, i_2, \ldots i_t\}, w-t\} \tag{17}$$

where S(k, t) is the set of all subsets of {1, 2,…, k} with size t.
Therefore the total complexity for multiplication is the following.

$$\sum_{t=1}^{\min\{k,w\}} \sum_{\{i_1, i_2, \ldots i_t\} \in S(k,t)} \sum_{I \subseteq \{1,2,\ldots,n-k\}, |I| = (n-k)-(w-t)} \left(O(t^3) + \frac{q^{t-r(I)}-1}{q-1} \times t \times (w-t) \right) \tag{18}$$

And the total complexity for addition is the following.

$$\sum_{t=1}^{\min\{k,w\}} \sum_{\{i_1, i_2, \ldots i_t\} \in S(k,t)} \sum_{I \subseteq \{1,2,\ldots,n-k\}, |I| = (n-k)-(w-t)} \left(O(t^3) + \frac{q^{t-r(I)}-1}{q-1} \times (t-1) \times (w-t) \right) \tag{19}$$

There are many factors for the complexity. But we focus on the size of q. We assume that q is very large and the other factors are small, i.e., we assume that |S(k, t)| and $\binom{n-k}{w-t}$ are small. In this case, the main factor is $q^{t-r(I)}$. Therefore if t-r(I) is small, Method 4 has better performance than previous algorithms.

4 Computation time

In this section, we compare our proposed method with the method in Magma. We have two kinds of computations. First one is the computation of minimum weight. Second one is the computation of partial weight distribution. For this purpose, we generate random linear codes over GF(q) for various values of q. More specifically, we make random matrix 7×5 matrix A over GF(q) and then we make generator matrix [I,A] for the random linear [12, 7, d] codes over GF(q). The computation results are summarized in Table 1 and Table 2.

In Table 1, we give the calculation time for the minimum weight. For example, for a random linear [12, 7, d] codes over GF(2^{13}) (in this case the minimum weight is d = 6), our proposed Method 4 takes 0.016 seconds but the Magma built-in function takes 288.578 seconds. In Table 2, we give the calculation time for the minimum weight and the partial weight distribution up to minimum weight. For example, for a random linear [12, 7, d] codes over GF(2^{10}) (in this case the minimum weight is d = 6 and the number of minimum weight codewords is # = 945252), our proposed Method 4 takes 0.031 seconds for the minimum weight and 0.422 seconds for the number of minimum weight codewords. But the Magma built-in function takes 4.734 seconds for the minimum weight and more than 2000.000 seconds for the number of minimum weight codewords.

From the Table 1 and Table 2, our method is more efficient than the Magma method.

Table 1: Minimum weight calculation time for random linear [12, 7, d] codes over GF(q)

q	d	Method 4	Magma
2^1	2	0.000	0.000
2^2	4	0.000	0.000
2^3	3	0.000	0.000
2^4	4	0.015	0.000
2^5	5	0.016	0.000
2^6	5	0.015	0.016
2^7	5	0.016	0.015
2^8	5	0.031	0.062
2^9	5	0.016	0.625
2^{10}	6	0.031	4.734
2^{11}	6	0.016	18.390
2^{12}	5	0.015	36.297
2^{13}	6	0.016	288.578
2^{14}	6	0.015	?
2^{15}	6	0.016	?
2^{20}	6	0.031	?
2^{50}	6	0.031	?
2^{100}	6	0.062	?

Table 2: Minimum weight calculation time for random linear [12, 7, d] codes over GF(q)

q	d	# min. cw.	Method 4 for d	Method 4 for #	Magma for d	Magma for #
2^1	3	5	0.016	0.000	0.000	0.000
2^2	2	3	0.016	0.000	0.000	0.000
2^3	4	98	0.000	0.000	0.000	0.000
2^4	3	15	0.000	0.000	0.000	0.000
2^5	4	31	0.000	0.000	0.015	0.000
2^6	4	63	0.000	0.000	0.000	0.000
2^7	5	127	0.031	0.015	0.016	0.031
2^8	6	235620	0.016	0.141	0.109	12.781
2^9	5	511	0.000	0.031	0.610	1.203
2^{10}	6	945252	0.031	0.422	4.734	>2000.000
2^{11}	5	2047	0.016	0.031	?	
2^{12}	6	3783780	0.016	1.594	?	
2^{13}	6	7568484	0.031	3.156	?	
2^{14}	6	15137892	0.015	6.219	?	
2^{15}	5	32767	0.016	0.031	?	
2^{16}	6	60554340	0.031	24.766	?	
2^{17}	6	121109604	0.031	49.609	?	
2^{18}	6	242220132	0.015	98.172	?	
2^{19}	6	484441188	0.016	195.718	?	
2^{20}	6	968883300	0.016	393.172	?	

5 Conclusion

In this paper, we proposed an efficient algorithm for computations of the minimum weight and the weight distribution of linear codes over finite fields. Our method is very efficient if the size of finite field is large. In this case, our algorithm is much faster than Magma. In the future work, it is worth while to find another application of our algorithm. For example, we can investigate whether our algorithm can be applied to the computation of the minimum weight and the weight distribution of quadratic residue codes.

Acknowledgements. This work (Grants No. C0328635) was partially supported by Business for Cooperative R&D between Industry, Academy, and Research Institute funded Korea Small and Medium Business Administration in 2015.
This paper was partially supported by the Education and Research Promotion Program of KOREATECH.

References

[1] E.R. Berlekamp, R.J. McEliece, H.C. van Tilborg, On the inherent intractability of certain coding problems, *IEEE Trans. Inform. Theory*, **24** (1978), 384-386. http://dx.doi.org/10.1109/tit.1978.1055873

[2] W. Bosma, J. Cannon and C. Playoust, The Magma Algebra System I: The User Language, *J. Symbolic Comput.*, **24** (1997), 235-265. http://dx.doi.org/10.1006/jsco.1996.0125

[3] A. Betten, H. Fripertinger, A. Kerber, A. Wassermann, K.-H. Zimmermann, *Codierungstheorie Konstruktion und Anwendung linearer Codes*, Springer-Verlag, Berlin, Heidelberg, New York, 1998. http://dx.doi.org/10.1007/978-3-642-58973-7

[4] I. Bouyukliev, V. Bakoev, A method for efficiently computing the number of codewords of fixed weights in linear codes, *Discrete Applied Mathematics*, **156** (2008), 2986-3004. http://dx.doi.org/10.1016/j.dam.2008.01.003

[5] A.R. Calderbank, N.J.A. Sloane, Modular and p-adic Cyclic Codes, *Designs, Codes Cryptogr*., **6** (1995), 21-35. http://dx.doi.org/10.1007/bf01390768

[6] J. Cramwinckel, E. Roijackers, R. Baart, E. Minkes, L. Ruscio, D. Joyner, GAP package GUAVA.

[7] S.T. Dougherty, S.Y. Kim, Y.H. Park, Lifted codes and their weight enumerators, *Discr. Math.*, **305** (2005), 123-135. http://dx.doi.org/10.1016/j.disc.2005.08.004

[8] P. Gaborit, C.-S. Nedeloaia, A. Wassermann, On the Weight Enumerators of Duadic and Quadratic Residue Codes, *IEEE Trans. Inf. Theory*, **51** (2005), 402-407. http://dx.doi.org/10.1109/tit.2004.839522

[9] S. Han, On the Weight Enumerators of the Projections of the 2-adic Golay Code of Length 24 to $\mathbf{Z}_2^e$, *Chapter in Mathematical Software - ICMS 2014*, 2014, 111-114. http://dx.doi.org/10.1007/978-3-662-44199-2_19

[10] J. Leon, A probabilistic algorithm for computing minimum weights of large error-correcting codes, *IEEE Trans. Inform. Theory*, **34** (1988), 1354-1359. http://dx.doi.org/10.1109/18.21270

[11] A. Vardy, The intractability of computing the minimum distance of a code, *IEEE Trans. Inform. Theory*, **43** (1997), no. 6, 1757-1766. http://dx.doi.org/10.1109/18.641542

Received: April 11, 2016; Published: June 2, 2016