Available at
www.ElsevierMathematics.com
POWERED BY SCIENCE DIRECT®
Discrete Mathematics 274 (2004) 213–231

DISCRETE MATHEMATICS

www.elsevier.com/locate/disc

On the computation of coset leaders with high Hamming weight

Håvard Raddum

Department of Informatics, University of Bergen, N-5020 Bergen, Norway

Received 11 March 2002; received in revised form 4 November 2002; accepted 2 January 2003

Abstract

The Newton radius of a code is the largest weight of a uniquely correctable error. The covering radius is the largest distance between a vector and the code. In this paper, we use the modular representation of a linear code to give an efficient algorithm for computing coset leaders of relatively high Hamming weight. The weights of these coset leaders serve as lower bounds on the Newton radius and the covering radius for linear codes.

Keywords: Newton radius; Covering radius; Modular representation

1. Introduction

When using a linear $[n,k]_q$-code to communicate over a noisy channel, maximum likelihood decoding is used to remove errors introduced during transmission. If the number of errors is less than or equal to $t = \lfloor (d-1)/2 \rfloor$, where d is the minimum distance, the transmitted codeword will always be the codeword closest to the received vector. If the number of errors is more than t, the codeword closest to the received vector may or may not be the transmitted codeword, depending on the error pattern and the code.

In [2], the term Newton radius was introduced as the largest weight of a uniquely correctable error. In that paper, some general bounds on the Newton radius for binary linear codes are given with some improved bounds on the Newton radius for binary first-order Reed–Muller codes. The paper also determines the Newton radius exactly for binary equidistant codes.

E-mail address: haavardr@ii.uib.no (H. Raddum).

0012-365X/03/$ - see front matter
doi:10.1016/S0012-365X(03)00090-6

The idea of using modular representation of a code for studying the Newton and covering radii was introduced in [3]. The work in our paper generalizes the methods used in [3]. In [1], general lower and upper bounds on the Newton radius for q-ary linear codes are given.

When trying to compute the Newton radius of a code, it is necessary to improve on these bounds, and narrow the interval where the Newton radius can be found. In this paper, we give an algorithm for computing uniquely correctable vectors of high weight for any q-ary linear code. This algorithm makes it possible to improve on the lower bounds, by exhibiting a uniquely correctable vector of higher weight than the known lower bound. One nice feature of this algorithm is that its complexity does not depend on n, the length of the code, but only on the dimension k. The algorithm can also be used to give lower bounds on the covering radius of a code.

Using this algorithm we have been able to improve some of the lower bounds for the Newton radius for binary first-order Reed–Muller codes given in [2].

2. Newton radius and covering radius using modular representation

Let C be an $[n,k]_q$ code, that is, a linear code of length n and dimension k over the finite field $\mathbb{F}_q$. An error $\mathbf{e}$ is (uniquely) correctable if and only if

$$w(\mathbf{e}) = d(\mathbf{e},0) < d(\mathbf{e},\mathbf{c})$$

for all non-zero code words $\mathbf{c}$, that is, it is the unique coset leader in its coset. The *Newton radius* $\nu(C)$ of C is the largest weight of a uniquely correctable error

$$\nu(C) = \max\{w(\mathbf{x}) \mid w(\mathbf{x}) < d(\mathbf{x},\mathbf{c}) \text{ for all } \mathbf{c} \in C\backslash\{0\}\}.$$

The *covering radius* $r(C)$ is the maximal distance of a vector from the code

$$r(C) = \max\{w(\mathbf{x}) \mid w(\mathbf{x}) \leqslant d(\mathbf{x},\mathbf{c}) \text{ for all } \mathbf{c} \in C\backslash\{0\}\}.$$

From the definitions of the Newton and covering radii, it immediately follows that $\nu(C) \leqslant r(C)$. A simple proof (see e.g. [2]) shows that if an $[n,k]_q$ code has a zero-position (that is, all code words are zero in this position) and the code is shortened to an $[n-1,k]_q$ code by removing the zero-position, then both the Newton radius and the covering radius decrease by one. Therefore, we will assume from now on that the codes do not have zero-positions.

Two $[n,k]_q$ codes C_1 and C_2 are *equivalent* if one is obtained from the other by some permutation of the columns of the generator matrix and multiplication of columns with non-zero scalars. Since equivalent codes have the same covering radii and the same Newton radii, it is convenient for our purpose to look at classes of equivalent codes.

2.1. Modular representation

To represent an equivalence class of codes we will use *modular representation* (see [4]).

By multiplying by non-zero scalars if necessary, we will assume that all columns in the generator matrix G of C has a 1 as the first non-zero entry. Thus, there are

$(q^k-1)/(q-1)$ vectors of length k that may appear in G. Let $A=\{1,2,\ldots,(q^k-1)/(q-1)\}$, order the columns in some order, and denote a column as $\mathbf{g}_a=(g_{a,1}, g_{a,2},\ldots,g_{a,k})^{\mathrm{T}}$ for $a\in A$. Let $\mathbf{u}=(u_a)_{a\in A}$ be a vector of length $(q^k-1)/(q-1)$, where u_a is the number of times $\mathbf{g}_a$ appears as a column in G. Then the vector $\mathbf{u}$ specifies the code C up to equivalence.

For vectors $\mathbf{a}=(a_1,a_2,\ldots,a_m)$ and $\mathbf{b}=(b_1,b_2,\ldots,b_m)$ of real numbers, we define the relations $\mathbf{a}\leqslant\mathbf{b}$, $\mathbf{a}<\mathbf{b}$, etc. to be component-wise.

$$\mathbf{a}\leqslant\mathbf{b} \;\Leftrightarrow\; a_i\leqslant b_i,\quad i=1,\ldots,m,$$

$$\mathbf{a}<\mathbf{b} \;\Leftrightarrow\; a_i< b_i,\quad i=1,\ldots,m,$$

etc.

Let $I_a(G)$ be the set of positions where $\mathbf{g}_a$ appears in G:

$$I_a(G)=\{i\,|\,\text{column no. } i \text{ of } G=\mathbf{g}_a\}.$$

For a vector $\mathbf{x}\in\mathbb{F}_q^n$, let $\mathbf{v}(\mathbf{x})=(v_{a,\beta}(\mathbf{x}))_{(a,\beta)\in A\times\mathbb{F}_q}$ where

$$v_{a,\beta}(\mathbf{x})=v_{a,\beta}(G,\mathbf{x})=|\{i\in I_a(G)\,|\,x_i=\beta\}|.$$

Clearly, we have

$$\sum_{\beta\in\mathbb{F}_q} v_{a,\beta}(\mathbf{x})=u_a. \tag{1}$$

In this notation, the weight of $\mathbf{x}$ can be written as

$$w(\mathbf{x})=\sum_{a\in A}(u_a-v_{a,0}(\mathbf{x})). \tag{2}$$

We are interested in measuring the distance between a vector $\mathbf{x}$ and the codewords of C. For $\mathbf{m}\in\mathbb{F}_q^k$, let $\mathbf{c}_\mathbf{m}=\mathbf{m}\cdot G$. In our notation we then have

$$d(\mathbf{x},\mathbf{c}_\mathbf{m})=\sum_{a\in A}(u_a-v_{a,\mathbf{m}\cdot\mathbf{g}_a}(\mathbf{x})), \tag{3}$$

where $\mathbf{m}\cdot\mathbf{g}_a=m_1g_{a,1}+\cdots+m_kg_{a,k}$ is the usual inner product.

Let $M=\mathbb{F}_q^k\backslash\{0\}$, and let $\mathbf{d}(\mathbf{x})=(d_\mathbf{m}(\mathbf{x}))_{\mathbf{m}\in M}$, where

$$d_\mathbf{m}(\mathbf{x})=d(\mathbf{x},\mathbf{c}_\mathbf{m})-w(\mathbf{x}).$$

In other words, $d_\mathbf{m}$ tells us how much closer $\mathbf{x}$ is to the all-zero codeword than to $\mathbf{c}_\mathbf{m}$.

Translating this into modular representation using (2) and (3) we get

$$\begin{aligned} d_\mathbf{m}(\mathbf{x}) &= \sum_{a\in A}(u_a-v_{a,\mathbf{m}\cdot\mathbf{g}_a}(\mathbf{x}))-\sum_{a\in A}(u_a-v_{a,0}(\mathbf{x})) \\ &= \sum_{a\in A}(v_{a,0}(\mathbf{x})-v_{a,\mathbf{m}\cdot\mathbf{g}_a}(\mathbf{x})). \end{aligned} \tag{4}$$

If $\mathbf{x}$ is a coset leader, we know that $0\leqslant d(\mathbf{x},\mathbf{c}_\mathbf{m})-w(\mathbf{x})$ for all $\mathbf{m}\in M$. If $\mathbf{x}$ is a unique coset leader we have $1\leqslant d(\mathbf{x},\mathbf{c}_\mathbf{m})-w(\mathbf{x})$ for all $\mathbf{m}\in M$. Using $\mathbf{d}(\mathbf{x})$ we can say that $\mathbf{x}$ is a coset leader if and only if $\mathbf{d}(\mathbf{x})\geqslant 0$, and that $\mathbf{x}$ is a unique coset leader if and only if $\mathbf{d}(\mathbf{x})\geqslant 1$.

Thus we are interested in finding $\mathbf{v}(\mathbf{x})$ that, by using (4), yields $\mathbf{d}(\mathbf{x}) \geqslant 0$ or $\mathbf{d}(\mathbf{x}) \geqslant 1$. With this notation the Newton radius and the covering radius can be described as

$$v(C) = \max\left\{ \sum_{a\in A} (u_a - v_{a,0}(\mathbf{x})) \,|\, \mathbf{d}(\mathbf{x}) \geqslant 1 \right\},$$

$$r(C) = \max\left\{ \sum_{a\in A} (u_a - v_{a,0}(\mathbf{x})) \,|\, \mathbf{d}(\mathbf{x}) \geqslant 0 \right\}.$$

We proceed to show that for a given code there is a 1–1 correspondence between $\mathbf{v}(\mathbf{x})$-vectors and $\mathbf{d}(\mathbf{x})$-vectors. Assume that $\mathbf{d}(\mathbf{x})$ and $\mathbf{u}$ are given, and that the $v_{a,\beta}(\mathbf{x})$ are unknown. Then (1) gives us $(q^k - 1)/(q - 1)$ equations, and (4) gives us $q^k - 1$ equations. Together this is $q\cdot(q^k - 1)/(q-1)$ equations in $q\cdot(q^k - 1)/(q-1)$ unknown. We prove that these equations are independent and, therefore, uniquely determine the unknown $\mathbf{v}(\mathbf{x})$.

We can write the equation set as a vector/matrix equation. To this end, we first fix some order for the elements of $\mathbb{F}_q = \{\sigma_0 = 0, \sigma_1 = 1, \ldots, \sigma_{q-1}\}$. Next, we fix some order of the variables (columns) and equations (rows). The variables (columns) are indexed by (a, β) and are listed in the following order:

$$(1,\sigma_0),(1,\sigma_1),(1,\sigma_2),\ldots,(1,\sigma_{q-1}),(2,\sigma_0),(2,\sigma_1),(2,\sigma_2),\ldots,(2,\sigma_{q-1}),\ldots,$$

$$\left(\frac{q^k-1}{q-1},\sigma_0\right),\left(\frac{q^k-1}{q-1},\sigma_1\right),\left(\frac{q^k-1}{q-1},\sigma_2\right),\ldots,\left(\frac{q^k-1}{q-1},\sigma_{q-1}\right).$$

The rows (equations) are indexed by a or $\mathbf{m}$ and are listed in the following order:

$$1,2,3,\ldots,\frac{q^k-1}{q-1},\mathbf{m}_1,\mathbf{m}_2,\mathbf{m}_3,\ldots,\mathbf{m}_{q^k-1},$$

where $\mathbf{m}_i$ is the q-ary representation of the integer i with the digits $\sigma_0,\ldots,\sigma_{q-1}$. The equations indexed by A correspond to Eq. (1), and the equations indexed by M correspond to Eq. (4). We denote the coefficient matrix by $B = (b_{z,(a,\beta)})$, where $z \in A \cup M$, $a \in A$ and $\beta \in \mathbb{F}_q$.

Using the same indexing as with B, define the matrix H as follows:

$$h_{a,(a,\beta)} = q^{k-2} - 1 \quad \text{for all } \beta \in \mathbb{F}_q,\ a \in A,$$

$$h_{a,(a',\beta)} = 0 \quad \text{for all } a, a' \in A, a \neq a',\ \beta \in \mathbb{F}_q,$$

$$h_{\mathbf{m},(a,0)} = q^{-1} - 1 \quad \text{for all } \mathbf{m} \in M,\ a \in A,$$

$$h_{\mathbf{m},(a,\beta)} = q^{-1} \quad \text{for all } \mathbf{m} \in M,\ a \in A,\ \beta \in \mathbb{F}_q \backslash \{0\}.$$

Then

$$B^{-1} = (q^{1-k}(B + H))^{\mathrm{T}}.$$

To show this, we find it convenient to introduce some extra notations

$$\mathbf{b}_{z,a} = (b_{z,(a,\sigma_0)}, b_{z,(a,\sigma_1)}, \ldots, b_{z,(a,\sigma_{q-1})}),$$

$$\mathbf{b}_z = (\mathbf{b}_{z,1}, \mathbf{b}_{z,2}, \mathbf{b}_{z,3}, \ldots, \mathbf{b}_{z,(q^k-1)/(q-1)}).$$

In this notation, $\mathbf{b}_z$ is the row of B indexed with z. In the following, we will call a segment $\mathbf{b}_{z,a}$ of the row $\mathbf{b}_z$ a *block*. Note that

$\mathbf{b}_{a,a}$ is all one for all $a \in A$,

$\mathbf{b}_{a,a'}$ is all zero for all $a, a' \in A,\ a \neq a'$,

$\mathbf{b}_{\mathbf{m},a}$ is all zero for all $\mathbf{m} \in M$ such that $\mathbf{m} \cdot \mathbf{g}_a = 0$.

In all other cases, $\mathbf{b}_{\mathbf{m},a}$ contains a 1 in the first position, a -1 in position i, where $\mathbf{m} \cdot \mathbf{g}_a = \sigma_i$, and zero in the remaining positions.

Lemma 2.1. *For all* $\mathbf{m} \in M$, *the number of all-zero blocks in* $\mathbf{b}_\mathbf{m}$ is $(q^{k-1}-1)/(q-1)$.

Proof. We get an all-zero block every time $\mathbf{m} \cdot \mathbf{g}_a = 0$. This can be seen as a linear equation in the k unknown $g_{a,1}, \ldots, g_{a,k}$, and thus has q^{k-1} solutions. After removing the all-zero solution and normalizing (only keeping those solutions that have a one as the first non-zero element), we are left with $(q^{k-1}-1)/(q-1)$ solutions in $\{\mathbf{g}_a \mid a \in A\}$. □

We consider some inner products:

Lemma 2.2. *We have*

(a) $\mathbf{b}_a \cdot \mathbf{b}_a = q$ *for all* $a \in A$,

(b) $\mathbf{b}_a \cdot \mathbf{b}_{a'} = 0$ *for all* $a, a' \in A,\ a \neq a'$,

(c) $\mathbf{b}_\mathbf{m} \cdot \mathbf{b}_a = 0$ *for all* $\mathbf{m} \in M,\ a \in A$,

(d) $\mathbf{b}_\mathbf{m} \cdot \mathbf{b}_\mathbf{m} = 2q^{k-1}$ *for all* $\mathbf{m} \in M$,

(e) $\mathbf{b}_\mathbf{m} \cdot \mathbf{b}_{\mathbf{m}'} = q^{k-1}$ *for all* $\mathbf{m}, \mathbf{m}' \in M,\ \mathbf{m} \neq \mathbf{m}'$.

Proof. Parts (a) and (b) are trivial. For part (c), we only need to observe that the sum of the numbers in one block of $\mathbf{b}_\mathbf{m}, \mathbf{m} \in M$ is always 0. For the inner product in part (d), we get a contribution of 2 from the blocks that contain a 1 and a -1, and a contribution of 0 from the all-zero blocks. By Lemma 2.1, we have $((q^k-1)/(q-1)) - ((q^{k-1}-1)/(q-1)) = q^{k-1}$ blocks that have a 1 and a -1. For part (e), we first find in how many blocks $\mathbf{b}_\mathbf{m}$ and $\mathbf{b}_{\mathbf{m}'}$ are equal. They are equal in the block corresponding to a if $\mathbf{m} \cdot \mathbf{g}_a = \mathbf{m}' \cdot \mathbf{g}_a$. As in Lemma 2.1, the equation $(\mathbf{m} - \mathbf{m}') \cdot \mathbf{g}_a = 0$ has $(q^{k-1}-1)/(q-1)$ solutions in $\{\mathbf{g}_a \mid a \in A\}$. Let t be the number of blocks where $\mathbf{b}_\mathbf{m}$ and $\mathbf{b}_{\mathbf{m}'}$ are both all-zero. For the inner product we then get a contribution of

2 from $(q^{k-1}-1)/(q-1)-t$ of the blocks, and we get a contribution of 0 from $2\cdot(q^{k-1}-1)/(q-1)-t$ blocks. For the rest of the blocks we get a contribution of 1, since every block that is not all-zero has a 1 in the first position. Summing up we get

$$\begin{aligned}\mathbf{b}_{\mathbf{m}}\cdot\mathbf{b}_{\mathbf{m}'} &= 1\cdot\left(\frac{q^k-1}{q-1}-\left(2\cdot\frac{q^{k-1}-1}{q-1}-t\right)-\left(\frac{q^{k-1}-1}{q-1}-t\right)\right)\\ &\quad+2\cdot\left(\frac{q^{k-1}-1}{q-1}-t\right)\\ &= q^{k-1}. \qquad \square\end{aligned}$$

Lemma 2.3. *Let* $\mathbf{h}_z$ *be defined similarly as* $\mathbf{b}_z$ *for* $z\in A\cup M$. *We then have*

(a) $\mathbf{b}_a\cdot\mathbf{h}_a = q^{k-1}-q$ *for all* $a\in A$,

(b) $\mathbf{b}_a\cdot\mathbf{h}_{a'} = 0$ *for all* $a,a'\in A,\ a\neq a'$,

(c) $\mathbf{b}_a\cdot\mathbf{h}_{\mathbf{m}} = 0$ *for all* $a\in A,\ \mathbf{m}\in M$,

(d) $\mathbf{b}_{\mathbf{m}}\cdot\mathbf{h}_a = 0$ *for all* $a\in A,\ \mathbf{m}\in M$,

(e) $\mathbf{b}_{\mathbf{m}}\cdot\mathbf{h}_{\mathbf{m}'} = -q^{k-1}$ *for all* $\mathbf{m},\mathbf{m}'\in M$.

The proof is a straightforward check of the stated equalities from the definitions of B and H, where part (e) also makes use of Lemma 2.1. Combining Lemmas 2.2 and 2.3 we get the following result.

Theorem 2.4. $q^{1-k}(B+H)B^{\mathrm{T}}$ *is the identity matrix of order* $q\cdot(q^k-1)/(q-1)$, *in particular* B *is non-singular*.

Theorem 2.4 proves that for a given code there is a 1–1 correspondence between $\mathbf{v}(\mathbf{x})$-vectors and $\mathbf{d}(\mathbf{x})$-vectors. Searching for $\mathbf{v}(\mathbf{x})$'s that give $\mathbf{d}(\mathbf{x})\geqslant 0$ or $\mathbf{d}(\mathbf{x})\geqslant 1$ is equivalent to searching for $\mathbf{x}$'s that are coset leaders, respectively, unique coset leaders. In the following, we will let $\mathbf{d}$ denote any vector of length q^k-1 with integer components.

Instead of searching for $\mathbf{v}(\mathbf{x})$'s, we will turn the problem around and search for $\mathbf{d}(\mathbf{x})$'s among the vectors with components in the non-negative integers. For a given $\mathbf{d}\geqslant 0$ or $\mathbf{d}\geqslant 1$, we can use Theorem 2.4 to compute the corresponding $\mathbf{v}$. If the resulting $\mathbf{v}$ only has non-negative integers as components we have $\mathbf{d}=\mathbf{d}(\mathbf{x})$ and $\mathbf{v}=\mathbf{v}(\mathbf{x})$ for some coset leader $\mathbf{x}$ (unique if $\mathbf{d}\geqslant 1$). In other words, the strategy is to specify the distance properties we want our coset leader to have, and then use B^{-1} to show us what this coset leader is.

Of course, it is very easy to specify distance properties which no coset leader has, in which case we will get a $\mathbf{v}$ with negative or non-integer components. In this case

$\mathbf{d} \notin \{\mathbf{d}(\mathbf{x}) \mid \mathbf{x} \in \mathbb{F}_q^n\}$ and $\mathbf{v} \notin \{\mathbf{v}(\mathbf{x}) \mid \mathbf{x} \in \mathbb{F}_q^n\}$. The next sections are concerned with how to choose $\mathbf{d}$ such that with high probability, $\mathbf{d} = \mathbf{d}(\mathbf{x})$ for some coset leader $\mathbf{x}$.

3. Constraints on the selection of d

First we need to introduce some more notation. For a $\mathbf{d}$-vector, let $\omega(\mathbf{d})$ be the sum of all components of $\mathbf{d}$:

$$\omega(\mathbf{d}) = \sum_{\mathbf{m} \in M} d_{\mathbf{m}}.$$

Let $\omega_{a,\beta}(\mathbf{d})$ be the sum of all elements of $\mathbf{d}$ with coordinates corresponding to points in the hyperplane in $\mathbb{F}_q^k$ given by $\mathbf{g}_a \cdot \mathbf{m} = \beta$:

$$\omega_{a,\beta}(\mathbf{d}) = \sum_{\{\mathbf{m} \in M \mid \mathbf{g}_a \cdot \mathbf{m} = \beta\}} d_{\mathbf{m}}.$$

By using Theorem 2.4 we can now compute each $v_{a,\beta}$ in terms of $\mathbf{d}$.

Lemma 3.1. *For each $(a,\beta) \in A \times \mathbb{F}_q$ we have*

$$v_{a,\beta} = \frac{u_a}{q} + \frac{\omega(\mathbf{d})}{q^k} - \frac{\omega_{a,\beta}(\mathbf{d})}{q^{k-1}}. \tag{5}$$

Proof. In matrix notation, the set of equations is given as $B \cdot \mathbf{v}(\mathbf{x}) = \binom{\mathbf{u}^{\mathrm{T}}}{\mathbf{d}^{\mathrm{T}}}$. Multiplying by B^{-1} on both sides and transposing we get

$$\mathbf{v}^{\mathrm{T}} = q^{1-k}((\mathbf{u},\mathbf{d}) \cdot B + (\mathbf{u},\mathbf{d}) \cdot H).$$

We focus on the calculation of one entry $v_{a,\beta}$, and break the proof into two cases.

Case 1: $\beta \neq 0$. In the product of $(\mathbf{u},\mathbf{d})$ with column (a,β) of B, the $\mathbf{u}$-part will meet a vector that contains a 1 in position a and zero otherwise. Since $\beta \neq 0$ the $\mathbf{d}$-part will meet a vector that contains -1 in the positions where $\mathbf{m} \cdot \mathbf{g}_a = \beta$, and zero otherwise. The inner product of $(\mathbf{u},\mathbf{d})$ with column (a,β) of B will therefore be $u_a - \omega_{a,\beta}(\mathbf{d})$. In the product of $(\mathbf{u},\mathbf{d})$ with column (a,β) of H, the $\mathbf{u}$-part will meet a vector that contains $q^{k-2} - 1$ in position a, and zero otherwise. Since $\beta \neq 0$, the $\mathbf{d}$-part will meet a vector with all entries q^{-1}. This inner product will be $u_a(q^{k-2} - 1) + q^{-1}\omega(\mathbf{d})$, and all together we get

$$v_{a,\beta} = q^{1-k}(u_a - \omega_{a,\beta}(\mathbf{d}) + u_a(q^{k-2} - 1) + q^{-1}\omega(\mathbf{d})) = \frac{u_a}{q} + \frac{\omega(\mathbf{d})}{q^k} - \frac{\omega_{a,\beta}(\mathbf{d})}{q^{k-1}}.$$

Case 2: $\beta = 0$. The calculation of the $\mathbf{u}$-part in the inner products is the same as above. In the product of $(\mathbf{u},\mathbf{d})$ and column $(a,0)$ of B, the $\mathbf{d}$-part will meet a vector that contains a 1 in all positions where $\mathbf{m} \cdot \mathbf{g}_a \neq 0$, and zero otherwise. So this inner product will be $u_a + \omega(\mathbf{d}) - \omega_{a,0}(\mathbf{d})$. In the product of $(\mathbf{u},\mathbf{d})$ and column $(a,0)$ of H,

the $\mathbf{d}$-part will meet a vector that has $q^{-1}-1$ in all coordinates, so this inner product will be $u_a(q^{k-2}-1)+(q^{-1}-1)\omega(\mathbf{d})$. Summing up we get

$$v_{a,0} = q^{1-k}(u_a+\omega(\mathbf{d})-\omega_{a,0}(\mathbf{d})+u_a(q^{k-2}-1)+(q^{-1}-1)\omega(\mathbf{d})$$
$$= \frac{u_a}{q}+\frac{\omega(\mathbf{d})}{q^k}-\frac{\omega_{a,0}(\mathbf{d})}{q^{k-1}}. \qquad \square$$

As explained earlier, we want to choose $\mathbf{d}$-vectors such that $\mathbf{d}=\mathbf{d}(\mathbf{x})$ for some $\mathbf{x}\in\mathbb{F}_q^n$. The following two propositions give us two constraints on the selection of $\mathbf{d}$.

Proposition 3.2. *If each* $v_{a,\beta}\in\mathbb{Z}$, *then* $\omega(\mathbf{d})\equiv 0 \pmod{q^{k-1}}$.

Proof. Using (4) we get the following:

$$\omega(\mathbf{d}) = \sum_{\mathbf{m}\in M}\sum_{a\in A}(v_{a,0}-v_{a,\mathbf{m}\cdot\mathbf{g}_a}) = \sum_{a\in A}\sum_{\mathbf{m}\in M}(v_{a,0}-v_{a,\mathbf{m}\cdot\mathbf{g}_a}).$$

For any given $\mathbf{g}_a$ and $\beta\neq 0$, there are q^{k-1} solutions in M to the equation $\mathbf{m}\cdot\mathbf{g}_a=\beta$. When $\mathbf{m}\cdot\mathbf{g}_a=0$, the term $v_{a,0}-v_{a,\mathbf{m}\cdot\mathbf{g}_a}$ vanishes. The equation can then be written as

$$\omega(\mathbf{d}) = \sum_{a\in A}\sum_{\beta\in\mathbb{F}_q^*} q^{k-1}(v_{a,0}-v_{a,\beta}) = q^{k-1}\sum_{a\in A}\sum_{\beta\in\mathbb{F}_q^*}(v_{a,0}-v_{a,\beta}),$$

where $F_q^* = F_q\backslash\{0\}$. $\square$

In other words, we only need to consider those $\mathbf{d}$'s for which $\omega(\mathbf{d})$ is a multiple of q^{k-1}.

Proposition 3.3. *If each* $v_{a,\beta}\in\mathbb{Z}$, *then* $\omega_{b,\alpha}(\mathbf{d})\equiv 0 \pmod{q^{k-2}}$ *for all* $(b,\alpha)\in A\times\mathbb{F}_q$.

Proof.

Case 1: $\alpha=0$. We sum (4) over those $\mathbf{m}\in M$ for which $\mathbf{m}\cdot\mathbf{g}_b=0$. This set has $q^{k-1}-1$ elements.

$$\omega_{b,0}(\mathbf{d}) = \sum_{a\in A}\sum_{\mathbf{m}\cdot\mathbf{g}_b=0}(v_{a,0}-v_{a,\mathbf{m}\cdot\mathbf{g}_a})$$
$$= (q^{k-1}-1)\sum_{a\in A}v_{a,0} - \sum_{a\in A\backslash\{b\}}\sum_{\mathbf{m}\cdot\mathbf{g}_b=0}v_{a,\mathbf{m}\cdot\mathbf{g}_a} - \sum_{\mathbf{m}\cdot\mathbf{g}_b=0}v_{b,\mathbf{m}\cdot\mathbf{g}_b}.$$

When $a\neq b$, the number of vectors in M that satisfy $\mathbf{m}\cdot\mathbf{g}_b=0$ and $\mathbf{m}\cdot\mathbf{g}_a=0$ is $q^{k-2}-1$, and the number of vectors in M that satisfy $\mathbf{m}\cdot\mathbf{g}_b=0$ and $\mathbf{m}\cdot\mathbf{g}_a=\beta\neq 0$ is

q^{k-2}. The equation above can then be written as

$$\omega_{b,0}(\mathbf{d}) = (q^{k-1}-1)\sum_{a\in A} v_{a,0} - \sum_{a\in A\setminus\{b\}} (q^{k-2}-1)v_{a,0}$$
$$- \sum_{a\in A\setminus\{b\}}\sum_{\beta\in\mathbb{F}_q^*} q^{k-2}v_{a,\beta} - (q^{k-1}-1)v_{b,0}$$
$$= (q^{k-1}-q^{k-2})\sum_{a\in A\setminus\{b\}} v_{a,0} - q^{k-2}\sum_{a\in A\setminus\{b\}}\sum_{\beta\in\mathbb{F}_q^*} v_{a,\beta}.$$

Case 2: $\alpha \neq 0$. We sum (4) over those $\mathbf{m}\in M$ that have $\mathbf{m}\cdot\mathbf{g}_b = \alpha$. This set has q^{k-1} elements. We get

$$\omega_{b,\alpha}(\mathbf{d}) = \sum_{a\in A}\sum_{\mathbf{m}\cdot\mathbf{g}_b=\alpha} (v_{a,0} - v_{a,\mathbf{m}\cdot\mathbf{g}_a})$$
$$= q^{k-1}\sum_{a\in A} v_{a,0} - \sum_{a\in A\setminus\{b\}}\sum_{\mathbf{m}\cdot\mathbf{g}_b=\alpha} v_{a,\mathbf{m}\cdot\mathbf{g}_a} - \sum_{\mathbf{m}\cdot\mathbf{g}_b=\alpha} v_{b,\mathbf{m}\cdot\mathbf{g}_b}.$$

When $a\neq b$, the number of solutions in M to the equations $\mathbf{m}\cdot\mathbf{g}_b=\alpha$ and $\mathbf{m}\cdot\mathbf{g}_a=\beta$ is q^{k-2}. We can then write the equation as

$$\omega_{b,\alpha}(\mathbf{d}) = q^{k-1}\sum_{a\in A} v_{a,0} - q^{k-2}\sum_{a\in A\setminus\{b\}}\sum_{\beta\in\mathbb{F}_q} v_{a,\beta} - q^{k-1}v_{b,\alpha}. \qquad \square$$

By Proposition 3.2 we see that if $\mathbf{d}=\mathbf{d}(\mathbf{x})$ for some $\mathbf{x}\in\mathbb{F}_q^n$, it is necessary that $\omega(\mathbf{d})$ is a multiple of q^{k-1}. Proposition 3.3 says that the sum of all coordinates of $\mathbf{d}$ indexed by the points of a hyperplane must be a multiple of q^{k-2}. We proceed to show how one can construct $\mathbf{d}$'s that give $v_{a,\beta}\in\mathbb{Z}$ for each $(a,\beta)\in A\times\mathbb{F}_q$.

4. Creating d-vectors

In this section, we will show how one can construct $\mathbf{d}$-vectors that meet the requirements from Propositions 3.2 and 3.3. In the following, if $\mathbf{d}\in\{\mathbf{d}(\mathbf{x})\mid\mathbf{x}\in\mathbb{F}_q^n\}$ we will say that $\mathbf{d}$ is *good*. In this section, we will show how one can construct a $\mathbf{d}$ such that the corresponding $\mathbf{v}$ only has integer components.

To facilitate the analysis, we will from now on let $\mathbf{d}$ be indexed by all the points in $\mathbb{F}_q^k$. However, we shall always insist that $d_0=0$ as it should be according to the definition given by (4), so that this slight abuse of notation will not invalidate any results from earlier sections. In particular, there will still be a 1–1 correspondence between $\mathbf{d}$-vectors and $\mathbf{v}$-vectors, and $\omega(\mathbf{d})$ and $\omega_{a,\beta}(\mathbf{d})$ will be the same regardless of whether d_0 is included or not. When writing $\mathbf{d}\geqslant 1$ it will be understood that this condition does not apply to d_0.

The analysis will be explained in the language of hyperplanes and linear spaces of codimension 2 in $\mathbb{F}_q^k$, so we start by introducing some notation for this.

For $(a,\beta)\in A\times\mathbb{F}_q$, let $\Pi_{a,\beta}$ be the hyperplane defined by $\mathbf{g}_a\cdot\mathbf{m}=\beta$. For $e\neq f$, let $h_{i,j}^{e,f}$ be the affine subspace of $\mathbb{F}_q^k$ of dimension $k-2$ defined by $h_{i,j}^{e,f}=\Pi_{e,\sigma_i}\cap\Pi_{f,\sigma_j}$. For given e and f in A, we denote the class of all the q^2 spaces $h_{i,j}^{e,f}$ by $H_{e,f}$:

$$H_{e,f}=\{h_{i,j}^{e,f}\mid 0\leqslant i,j\leqslant q-1\}.$$

We can write the equations for $h_{i,j}^{e,f}\cap\Pi_{a,\beta}$ in matrix notation

$$\begin{bmatrix} g_{e,k} & g_{e,k-1} & \cdots & g_{e,1}\\ g_{f,k} & g_{f,k-1} & \cdots & g_{f,1}\\ g_{a,k} & g_{a,k-1} & \cdots & g_{a,1}\end{bmatrix}\begin{bmatrix} m_k\\ m_{k-1}\\ \vdots\\ m_1\end{bmatrix}=\begin{bmatrix}\sigma_i\\ \sigma_j\\ \beta\end{bmatrix}.$$

Let the coefficient matrix of the above equations be Z. Since $h_{i,j}^{e,f}$ is a space of dimension $k-2$, the rank of Z is either 2 or 3. If $\operatorname{rank}(Z)=3$, $\Pi_{a,\beta}\cap h_{i,j}^{e,f}$ will be a linear space of dimension $k-3$ for any i,j and any β.

If $\operatorname{rank}(Z)=2$, then either $\Pi_{a,\beta}\cap h_{i,j}^{e,f}=\emptyset$ or $h_{i,j}^{e,f}\subseteq\Pi_{a,\beta}$, depending on i,j and β. All the q^2 different $h_{i,j}^{e,f}$ in one $H_{e,f}$ cover all of $\mathbb{F}_q^k$. Because of this, and the fact that the $h_{i,j}^{e,f}$ have dimension one lower than $\Pi_{a,\beta}$, there will be exactly q indices (i,j) such that $h_{i,j}^{e,f}\subseteq\Pi_{a,\beta}$ for a given β when $\operatorname{rank}(Z)=2$. Likewise, given a pair of indices (i,j), there will be exactly one β such that $h_{i,j}^{e,f}\subseteq\Pi_{a,\beta}$. We will use the notation $H_{e,f}\|\Pi_{a,\beta}$ to mean that e,f and a are chosen such that $\operatorname{rank}(Z)=2$.

Given e and f, to have $H_{e,f}\|\Pi_{a,\beta}$, $\mathbf{g}_a$ must be a non-zero linear combination of $\mathbf{g}_e$ and $\mathbf{g}_f$, normalized to have first non-zero entry 1. There are, therefore, $(q^2-1)/(q-1)=q+1$ different $\mathbf{g}_a$'s that give $H_{e,f}\|\Pi_{a,\beta}$ for a given e and f. We denote this set of indices as $P_{e,f}$:

$$P_{e,f}=\{a\mid H_{e,f}\|\Pi_{a,\beta}\}.$$

Whenever $H_{e,f}\|\Pi_{a,\beta}$, we define the set of indices (i,j) such that $h_{i,j}^{e,f}\subseteq\Pi_{a,\beta}$ to be

$$J_{a,\beta}^{e,f}=\{(i,j)|h_{i,j}^{e,f}\in H_{e,f}\text{ and }h_{i,j}^{e,f}\subseteq\Pi_{a,\beta}\}.$$

Notice that $|J_{a,\beta}^{e,f}|=q$, and that $a\in P_{e,f}$ if and only if $H_{e,f}\|\Pi_{a,\beta}$.

For each class $H_{e,f}$, let $\mathbf{d}_{e,f}$ be a vector indexed the same way as $\mathbf{d}$. In particular, $\mathbf{d}_{e,f}$ shall have the value 0 in the component corresponding to 0. Let $\mathbf{d}_{e,f}$ have the value $t_{i,j}^{e,f}$ in the coordinates indexed by points in $h_{i,j}^{e,f}$, where

$$\sum_{i=0}^{q-1}\sum_{j=0}^{q-1}t_{i,j}^{e,f}=qT_{e,f}\quad\text{(constraint 1)}.$$

Lemma 4.1. *$\omega(\mathbf{d}_{e,f})\equiv 0\,(\operatorname{mod}q^{k-1})$ and $\omega_{a,\beta}(\mathbf{d}_{e,f})\equiv 0\,(\operatorname{mod}q^{k-2})$ for any $(a,\beta)\in A\times\mathbb{F}_q$.*

Proof. Each $h_{i,j}^{e,f}$ contains q^{k-2} points, so we get

$$\omega(\mathbf{d}_{e,f}) = q^{k-2} \sum_{i=0}^{q-1} \sum_{j=0}^{q-1} t_{i,j}^{e,f}.$$

Since $\sum_{i=0}^{q-1} \sum_{j=0}^{q-1} t_{i,j}^{e,f}$ is a multiple of q we get that $\omega(\mathbf{d}_{e,f}) \equiv 0 \pmod{q^{k-1}}$.

For $\omega_{a,\beta}(\mathbf{d}_{e,f})$ we get two cases to consider.

Case 1: $H_{e,f} \nparallel \Pi_{a,\beta}$. In this case $\Pi_{a,\beta} \cap h_{i,j}^{e,f}$ will consist of q^{k-3} points for each i,j. We then get

$$\omega_{a,\beta}(\mathbf{d}_{e,f}) = q^{k-3} \sum_{i=0}^{q-1} \sum_{j=0}^{q-1} t_{i,j}^{e,f}.$$

Again, since $\sum_{i=0}^{q-1} \sum_{j=0}^{q-1} t_{i,j}^{e,f}$ is a multiple of q we get $\omega_{a,\beta}(\mathbf{d}_{e,f}) \equiv 0 \pmod{q^{k-2}}$.

Case 2: $H_{e,f} \| \Pi_{a,\beta}$. For indices (i,j) not in $J_{a,\beta}^{e,f}$ we have $h_{i,j}^{e,f} \cap \Pi_{a,\beta} = \emptyset$. We then get

$$\omega_{a,\beta}(\mathbf{d}_{e,f}) = q^{k-2} \sum_{(i,j) \in J_{a,\beta}^{e,f}} t_{i,j}^{e,f} \equiv 0 \pmod{q^{k-2}}. \qquad \square$$

The lemma above shows us how we can choose $\mathbf{d}_{e,f}$'s that meet the requirements given in Propositions 3.2 and 3.3. There is one general constraint on the selection of $t_{i,j}^{e,f}$ in addition to constraint 1. Since we require that the component in $\mathbf{d}_{e,f}$ indexed by 0 has the value 0, we also get the constraint

$$t_{0,0}^{e,f} = 0 \quad \text{(constraint 2)}.$$

We give now an algorithm for selecting each $t_{i,j}^{e,f}$. In this algorithm we need a new set of vectors. For $a \in A$ and $1 \leqslant r \leqslant u_a$, let $\mathbf{y}^{a,r}$ be a vector indexed in the same manner as $\mathbf{t}^{e,f}$.

Algorithm 1.

For each $a \in P_{e,f}$:
  For each $1 \leqslant r \leqslant u_a$:
    Randomly choose $\gamma \in \mathbb{F}_q$, but make sure that when the algorithm is finished, the number of times $\gamma = 0$ has been chosen is a multiple of q.
    Let $y_{i,j}^{a,r}$ have the value $q-1$ when $(i,j) \in J_{a,\gamma}^{e,f}$, and the value 0 when $(i,j) \notin J_{a,\gamma}^{e,f}$.
Let $t_{i,j}^{e,f} = \sum_{a \in P_{e,f}} \sum_{r=1}^{u_a} y_{i,j}^{a,r} \pmod{q}$.

Notice that $(0,0) \in J_{a,\gamma}^{e,f}$ if and only if $\gamma = 0$. Since the number of times $\gamma = 0$ is chosen is a multiple of q, the vector $\mathbf{t}^{e,f}$ will satisfy constraint 2.

We know that $|J_{a,\gamma}^{e,f}| = q$, so we get that $\sum_{i=0}^{q-1} \sum_{j=0}^{q-1} y_{i,j}^{a,r} = q(q-1)$ for any $a \in P_{e,f}$ and $1 \leqslant r \leqslant u_a$. We can then check that a $\mathbf{t}^{e,f}$-vector chosen according to Algorithm 1 also meets constraint 1:

$$\sum_{i=0}^{q-1}\sum_{j=0}^{q-1} t_{i,j}^{e,f} \equiv \sum_{i=0}^{q-1}\sum_{j=0}^{q-1}\sum_{a\in P_{e,f}}\sum_{r=1}^{u_a} y_{i,j}^{a,r}$$

$$\equiv \sum_{a\in P_{e,f}}\sum_{r=1}^{u_a} q(q-1) \equiv 0 \pmod q.$$

We need two more lemmas before we are ready to prove that a $\mathbf{d}_{e,f}$ constructed using a $\mathbf{t}^{e,f}$ from Algorithm 1 is useful for constructing a good $\mathbf{d}$.

Lemma 4.2. *Given $e, f \in A$, let $a, a' \in P_{e,f}$, $a \neq a'$. For any $\beta \in \mathbb{F}_q$ we then have*

$$\sum_{(i,j)\in J_{a,\beta}^{e,f}} y_{i,j}^{a',r} = q-1 \quad \textit{and} \quad \sum_{(i,j)\in J_{a,\beta}^{e,f}} y_{i,j}^{a,r} \equiv 0 \pmod q.$$

Proof. The two hyperplanes given by $\mathbf{g}_a$ and $\mathbf{g}_{a'}$ are not parallel, and so they will intersect in a subspace of dimension $k-2$. Since both of them can be written as unions of spaces from $H_{e,f}$, they must intersect in exactly one of the $h_{i,j}^{e,f}$ in $H_{e,f}$. From this we see that one of the terms in $\sum_{(i,j)\in J_{a,\beta}^{e,f}} y_{i,j}^{a',r}$ will be $q-1$ and the others will be 0.

From the construction of $\mathbf{y}^{a,r}$ in Algorithm 1, we get that the sum $\sum_{(i,j)\in J_{a,\beta}^{e,f}} y_{i,j}^{a,r}$ will be either 0 or $q(q-1)$, depending of the choice of γ made when constructing $\mathbf{y}^{a,r}$. In either case the sum is $0 \bmod q$. □

We need to define one final variable before the next lemma. For given $e, f \in A$, let $X_{e,f} \equiv \sum_{a\in P_{e,f}} u_a \pmod q$.

Lemma 4.3. *Let $e, f \in A$ be given, and let $\mathbf{t}^{e,f}$ be chosen according to Algorithm 1. Then for any $\beta \in \mathbb{F}_q$*

$$\sum_{(i,j)\in J_{a,\beta}^{e,f}} t_{i,j}^{e,f} \equiv u_a - X_{e,f} \bmod q.$$

Proof. Substituting for $t_{i,j}^{e,f}$ and changing the order of summations we get

$$\sum_{(i,j)\in J_{a,\beta}^{e,f}} t_{i,j}^{e,f} \equiv \sum_{a'\in P_{e,f}}\sum_{r=1}^{u_{a'}}\sum_{(i,j)\in J_{a,\beta}^{e,f}} y_{i,j}^{a',r} \pmod q.$$

Splitting the outer sum into the cases $a = a'$ and $a \neq a'$, and using Lemma 4.2 we get

$$\sum_{(i,j)\in J_{a,\beta}^{e,f}} t_{i,j}^{e,f} \equiv \sum_{r=1}^{u_a} \sum_{(i,j)\in J_{a,\beta}^{e,f}} y_{i,j}^{a,r} + \sum_{a'\in P_{e,f}\setminus\{a\}} \sum_{r=1}^{u_{a'}} \sum_{(i,j)\in J_{a,\beta}^{e,f}} y_{i,j}^{a',r}$$

$$\equiv 0 + (q-1)(X_{e,f} - u_a) \equiv u_a - X_{e,f} \pmod q. \qquad \square$$

Notice that adding a multiple of q to a component of a $\mathbf{t}^{e,f}$-vector constructed by Algorithm 1 will not change the validity of Lemma 4.3, since all computations are done mod q. However, adding q to some $t_{i,j}^{e,f}$ will increase $T_{e,f}$ by one.

4.1. Constructing $\mathbf{d}$

The idea now is to take a set H of classes $H_{e,f}$, construct a $\mathbf{d}_{e,f}$ for each class, and let $\mathbf{d}$ be the sum of the $\mathbf{d}_{e,f}$'s. However, the set H must be chosen carefully. In order to get a good $\mathbf{d}$, it turns out to be necessary that

$$|\{H_{e,f} \in H \mid a \in P_{e,f}\}| \equiv 1 \pmod q \text{ for each } a \in A \quad \text{(constraint 3)}.$$

We construct H as follows. Start with $H = \emptyset$ and select $e \in A$ at random. Choose $f_1 \in A\setminus\{e\}$, and put H_{e,f_1} in H. Continue recursively: Given $H_{e,f_1}, \ldots, H_{e,f_{l-1}}$, select $f_l \in A\setminus(\bigcup_{i=1}^{l-1} P_{e,f_i})$ and put H_{e,f_l} in H. The process stops when $A = \bigcup_{i=1}^{l} P_{e,f_i}$. Creating H this way we will have $P_{e,f_i} \cap P_{e,f_j} = \{e\}$ when $i \neq j$. At the start of the recursion there will be $(q^k - 1)/(q-1) - 1$ elements in A to choose from. Since $|P_{e,f_l}| = q + 1$, for each new H_{e,f_l} that gets added to H, the set from which the next f_{l+1} can be chosen from is reduced by q elements. We, therefore, get $|H| = (1/q)(((q^k - 1)/(q-1)) - 1) = (q^{k-1} - 1)(q-1)$. When $a \neq e$ we then have $|\{H_{e,f} \in H \mid a \in P_{e,f}\}| = 1$, and when $a = e$ we get $|\{H_{e,f} \in H \mid e \in P_{e,f}\}| = (q^{k-1} - 1)(q-1) \equiv 1 \pmod q$. This construction will then give us a set H satisfying constraint 3.

We sum this up in a second algorithm:

Algorithm 2.

Compute H such that constraint 3 is satisfied
For each $H_{e,f} \in H$:
  Select $\mathbf{t}^{e,f}$ according to Algorithm 1.
  Compute $T_{e,f}$ and $X_{e,f}$.
  While $T_{e,f} \not\equiv -X_{e,f} \pmod q$:
    Randomly select $(i,j) \in \mathbb{F}_q \times \mathbb{F}_q \setminus \{(0,0)\}$.
    Let $t_{i,j}^{e,f} = t_{i,j}^{e,f} + q$.
  Construct $\mathbf{d}_{e,f}$ using $\mathbf{t}^{e,f}$.
Let $\mathbf{d} = \sum \mathbf{d}_{e,f}$.

Lemma 4.4. *Let e, f and a be given such that $H_{e,f} \nparallel \Pi_{a,\beta}$. Then*

$$\frac{\omega(\mathbf{d}_{e,f})}{q^k} - \frac{\omega_{a,\beta}(\mathbf{d}_{e,f})}{q^{k-1}} = 0.$$

Proof. Since $H_{e,f} \nparallel \Pi_{a,\beta}$, the hyperplane $\Pi_{a,\beta}$ will meet each $h_{i,j}^{e,f}$ in q^{k-3} points. This gives us $\omega_{a,\beta}(\mathbf{d}_{e,f}) = q^{k-3} \sum_{i=0}^{q-1} \sum_{j=0}^{q-1} t_{i,j}^{e,f}$. The lemma follows from the fact that $\omega(\mathbf{d}_{e,f}) = q^{k-2} \sum_{i=0}^{q-1} \sum_{j=0}^{q-1} t_{i,j}^{e,f} = q\omega_{a,\beta}(\mathbf{d}_{e,f})$. □

We are now ready to prove the main result in this section.

Theorem 4.5. *Construct* $\mathbf{d}$ *according to Algorithm* 2. *Then*

$$v_{a,\beta} = \frac{u_a}{q} + \frac{\omega(\mathbf{d})}{q^k} - \frac{\omega_{a,\beta}(\mathbf{d})}{q^{k-1}} \in \mathbb{Z}.$$

Proof. Collecting the fractions we get

$$v_{a,\beta} = \frac{q^{k-1}u_a + \omega(\sum_{H_{e,f} \in H} \mathbf{d}_{e,f}) - q\omega_{a,\beta}(\sum_{H_{e,f} \in H} \mathbf{d}_{e,f})}{q^k}.$$

By Lemma 4.4 we only need to include the terms where $a \in P_{e,f}$ in the sum. By substituting $\omega(\mathbf{d}_{e,f})$ and $\omega_{a,\beta}(\mathbf{d}_{e,f})$ with the expressions given in Lemma 4.1 we get

$$\begin{aligned} v_{a,\beta} &= \frac{q^{k-1}u_a + \sum_{\{H_{e,f} \in H | a \in P_{e,f}\}} (q^{k-2} \cdot qT_{e,f} - q \cdot q^{k-2} \sum_{(i,j) \in J_{a,\beta}^{e,f}} t_{i,j}^{e,f})}{q^k} \\ &= \frac{u_a + \sum_{\{H_{e,f} \in H | a \in P_{e,f}\}} (T_{e,f} - \sum_{(i,j) \in J_{a,\beta}^{e,f}} t_{i,j}^{e,f})}{q}. \end{aligned}$$

To prove the theorem it is therefore sufficient to show

$$u_a + \sum_{\{H_{e,f} \in H | a \in P_{e,f}\}} \left(T_{e,f} - \sum_{(i,j) \in J_{a,\beta}^{e,f}} t_{i,j}^{e,f} \right) \equiv 0 \pmod{q}.$$

Since each $\mathbf{d}_{e,f}$ is constructed using Algorithm 2, we know that $T_{e,f} \equiv -X_{e,f}$, and by Lemma 4.3 that

$$\sum_{(i,j) \in J_{a,\beta}^{e,f}} t_{i,j}^{e,f} \equiv u_a - X_{e,f} \pmod{q}.$$

Substituting this into the expression above we get

$$u_a + \sum_{\{H_{e,f} \in H | a \in P_{e,f}\}} - u_a \pmod{q}.$$

The number of terms in the sum is congruent to 1 modulo q since H satisfies constraint 3. Finally, the expression can then be written as

$$u_a + \sum_{\{H_{e,f} \in H | a \in P_{e,f}\}} - u_a \equiv u_a - u_a \equiv 0 \pmod{q}. \qquad \square$$

5. Adjusting d

Theorem 4.5 solves one part of the problem on how to find good **d**'s. However, several components in the corresponding **v** may still be negative, in which case **d** is not good. This section is concerned with how to alter **d** to remove negative numbers in the corresponding **v**.

For $(a,\beta)\in A\times\mathbb{F}_q^*$, let $\mathbf{z}_{a,\beta}$ be a vector indexed by the points in $\mathbb{F}_q^k$. Let $\mathbf{z}_{a,\beta}$ have the value q in all components corresponding to points in $\Pi_{a,\beta}$, and have the value 0 otherwise.

The vector **v** is given by **d** and **u**, where we consider **u** to be a given constant vector. In the remainder of this section we will write $v_{a,\beta}(\mathbf{d})$ to mean the value $v_{a,\beta}$ gets using **d** in (5). The following lemma explains how **v** changes when adding or subtracting a $\mathbf{z}_{a,\beta}$ to **d**.

Lemma 5.1. *Let $a,b\in A$ and $\alpha,\beta\in\mathbb{F}_q$. We then have*

(a) $v_{a,\beta}(\mathbf{d}\pm\mathbf{z}_{b,\alpha})=v_{a,\beta}(\mathbf{d})$ *for* $b\neq a$,

(b) $v_{a,\beta}(\mathbf{d}\pm\mathbf{z}_{a,\alpha})=v_{a,\beta}(\mathbf{d})\pm 1$ *for* $\alpha\neq\beta$,

(c) $v_{a,\beta}(\mathbf{d}\pm\mathbf{z}_{a,\beta})=v_{a,\beta}(\mathbf{d})\mp(q-1)$.

Proof. For any b and α we have $\omega(\mathbf{z}_{b,\alpha})/q^k=q\cdot q^{k-1}/q^k=1$.

(a) We isolate the terms involving $\mathbf{z}_{b,\alpha}$.

$$\begin{aligned}v_{a,\beta}(\mathbf{d}\pm\mathbf{z}_{b,\alpha})&=\frac{u_a}{q}+\frac{\omega(\mathbf{d}\pm\mathbf{z}_{b,\alpha})}{q^k}-\frac{\omega_{a,\beta}(\mathbf{d}\pm\mathbf{z}_{b,\alpha})}{q^{k-1}}\\&=\left(\frac{u_a}{q}+\frac{\omega(\mathbf{d})}{q^k}-\frac{\omega_{a,\beta}(\mathbf{d})}{q^{k-1}}\right)\pm\frac{\omega(\mathbf{z}_{b,\alpha})}{q^k}\mp\frac{\omega_{a,\beta}(\mathbf{z}_{b,\alpha})}{q^{k-1}}\\&=v_{a,\beta}(\mathbf{d})\pm\frac{\omega(\mathbf{z}_{b,\alpha})}{q^k}\mp\frac{\omega_{a,\beta}(\mathbf{z}_{b,\alpha})}{q^{k-1}}.\end{aligned}$$

Since $a\neq b$, the hyperplane $\Pi_{a,\beta}$ will meet $\Pi_{b,\alpha}$ in q^{k-2} points and so we have $\omega_{a,\beta}(\mathbf{z}_{b,\alpha})=q\cdot q^{k-2}$. The two last terms in the expression above will then cancel out, and so $v_{a,\beta}$ will remain unchanged.

(b) Separating as above we get

$$v_{a,\beta}(\mathbf{d}\pm\mathbf{z}_{a,\alpha})=v_{a,\beta}(\mathbf{d})\pm\frac{\omega(\mathbf{z}_{a,\alpha})}{q^k}\mp\frac{\omega_{a,\beta}(\mathbf{z}_{a,\alpha})}{q^{k-1}}.$$

The two hyperplanes $\Pi_{a,\beta}$ and $\Pi_{a,\alpha}$ are parallell, and so we get $\omega_{a,\beta}(\mathbf{z}_{a,\alpha})=0$.

(c) Separating we get

$$v_{a,\beta}(\mathbf{d}\pm\mathbf{z}_{a,\beta})=v_{a,\beta}(\mathbf{d})\pm\frac{\omega(\mathbf{z}_{a,\beta})}{q^k}\mp\frac{\omega_{a,\beta}(\mathbf{z}_{a,\beta})}{q^{k-1}}.$$

We have $\omega_{a,\beta}(\mathbf{z}_{a,\beta})=q\cdot q^{k-1}=q^k$, so the last term of the expression is q. □

From Lemma 5.1 we see that adding and/or subtracting any number of $\mathbf{z}_{a,\beta}$'s to a $\mathbf{d}$ constructed using Algorithm 2, will not change the property that all components of the corresponding $\mathbf{v}$ will be integers.

It is possible to use Lemma 5.1 to change $\mathbf{d}$ so that some or all negative components of the corresponding $\mathbf{v}$ are changed to non-negative integers. However, in order to keep $d_0 = 0$ we cannot use $\mathbf{z}_{a,0}$ for any $a \in A$. Unfortunately, we have not been able to find an algorithm based on Lemma 5.1 that is guaranteed to leave $\mathbf{v} \geqslant 0$.

There are basically two ways one can change $\mathbf{d}$ to remove negative components of $\mathbf{v}$, but certain conditions must be met in order not to introduce new negative components in $\mathbf{v}$, and to keep $\mathbf{d} \geqslant 0$ or $\mathbf{d} \geqslant 1$.

Lemma 5.2. *Assume $-l \leqslant v_{a,\beta}(\mathbf{d}) < 0$, and that there exists $\beta_1, \ldots, \beta_l \in \mathbb{F}_q^*$, such that $v_{a,\beta_i}(\mathbf{d}) \geqslant (q-l)$, $i = 1, \ldots, l$. Then the number of negative components in $\{v_{b,\alpha}(\mathbf{d} + \sum_{i=1}^{l} \mathbf{z}_{a,\beta_i}) \,|\, (b,\alpha) \in A \times \mathbb{F}_q\}$ is less than the number of negative components in $\{v_{b,\alpha}(\mathbf{d}) \,|\, (b,\alpha) \in A \times \mathbb{F}_q\}$.*

Proof. By part (a) of Lemma 5.1 only the components $v_{a,\sigma_0}, \ldots, v_{a,\sigma_{q-1}}$ will change when going from $\mathbf{d}$ to $\mathbf{d} + \sum_{i=1}^{l} \mathbf{z}_{a,\beta_i}$. Since $\beta \neq \beta_i$, $1 \leqslant i \leqslant l$, part (b) of Lemma 5.1 gives us

$$v_{a,\beta}\left(\mathbf{d} + \sum_{i=1}^{l} \mathbf{z}_{a,\beta_i}\right) = v_{a,\beta}(\mathbf{d}) + l \geqslant 0,$$

so there is at least one negative component that has been changed to non-negative. From part (c) of Lemma 5.1 it follows that each v_{a,β_i} will decrease by $(q-1)$ when $\mathbf{z}_{a,\beta_i}$ is added, but we also get from part (b) that v_{a,β_i} will increase by 1 when each of the $l-1$ other $\mathbf{z}_{a,\beta_j}$'s are added. We then get for $i = 1, \ldots, l$

$$\begin{aligned} v_{a,\beta_i}\left(\mathbf{d} + \sum_{i=1}^{l} \mathbf{z}_{a,\beta_i}\right) &= v_{a,\beta_i}(\mathbf{d}) - (q-1) + (l-1) \\ &\geqslant (q-l) - (q-1) + (l-1) = 0, \end{aligned}$$

so no new negative components have been introduced. □

In the next lemma, we let $l = q$ if we require $\mathbf{d} \geqslant 0$ (searching for coset leaders), and we let $l = q+1$ if we require $\mathbf{d} \geqslant 1$ (searching for unique coset leaders).

Lemma 5.3. *Assume that $(1-q) \leqslant v_{a,\beta} < 0$ for one $\beta \in \mathbb{F}_q^*$, and that $v_{a,\alpha} \geqslant 1$ for all $\alpha \in \mathbb{F}_q \backslash \{\beta\}$. Assume also that $\min\{d_{\mathbf{m}} \,|\, \mathbf{m} \in \Pi_{a,\beta}\} \geqslant l$. Then the number of negative components in $\{v_{b,\alpha}(\mathbf{d} - \mathbf{z}_{a,\beta}) \,|\, (b,\alpha) \in A \times \mathbb{F}_q\}$ is less than the number of negative components in $\{v_{b,\alpha}(\mathbf{d}) \,|\, (b,\alpha) \in A \times \mathbb{F}_q\}$.*

Proof. By part (a) of Lemma 5.1, only the components $v_{a,\sigma_0}, \ldots, v_{a,\sigma_{q-1}}$ will change when going from $\mathbf{d}$ to $\mathbf{d} - \mathbf{z}_{a,\beta}$. By part (c) of Lemma 5.1 $v_{a,\beta}$ will increase by $(q-1)$ when going from $\mathbf{d}$ to $\mathbf{d} - \mathbf{z}_{a,\beta}$. This gives $v_{a,\beta}(\mathbf{d} - \mathbf{z}_{a,\beta}) = v_{a,\beta}(\mathbf{d}) + q - 1 \geqslant 0$, so

at least one of the negative components in $\mathbf{v}$ becomes non-negative when subtracting $\mathbf{z}_{a,\beta}$. By part (b) of Lemma 5.1, all the other $v_{a,\alpha}$, $\alpha \in \mathbb{F}_q \backslash \{\beta\}$ will decrease by 1. We then have $v_{a,\alpha}(\mathbf{d} - \mathbf{z}_{a,\beta}) = v_{a,\alpha}(\mathbf{d}) - 1 \geqslant 0$, so no components in $\mathbf{v}$ have been changed from non-negative to negative. Since $\min\{d_{\mathbf{m}} \mid \mathbf{m} \in \Pi_{a,\beta}\} \geqslant l$, we still have $\mathbf{d} \geqslant 0$ or $\mathbf{d} \geqslant 1$. □

In the implementation of the ideas presented in this paper, after $\mathbf{d}$ was constructed using Algorithm 2, the corresponding $\mathbf{v}$-vector was checked to see if the conditions in Lemmas 5.2 or 5.3 were met. This was repeated until $\mathbf{v} \geqslant 0$, or until neither lemma could be used to construct a better $\mathbf{d}$.

We want the eventual coset leaders we find to have high weight. Recall that if we find a good $\mathbf{d}$, producing coset leader $\mathbf{x}$, we have $d_{\mathbf{m}}(\mathbf{x}) = d(\mathbf{x}, \mathbf{c}_{\mathbf{m}}) - w(\mathbf{x})$. We can expect that at least a few of the codewords have large distance to $\mathbf{x}$. In other words, if $\max\{d_{\mathbf{m}}(\mathbf{x}) \mid \mathbf{m} \in \mathbb{F}_q^k\}$ is small, then $w(\mathbf{x})$ must be large, relatively speaking.

This means that before the process using Lemmas 5.2 and 5.3 was started, we subtracted as many $\mathbf{z}_{a,\beta}$'s from $\mathbf{d}$ as possible, all the time keeping $\mathbf{d} \geqslant 0$ or $\mathbf{d} \geqslant 1$. The order in which the $\mathbf{z}_{a,\beta}$ are subtracted turns out to be very important. Several methods were tested. One was to always make sure that the largest value in $\mathbf{d}$ was decreased with each subtraction, another was to subtract $\mathbf{z}_{a,\beta}$ when $\min\{d_{\mathbf{m}} \mid \mathbf{m} \in \Pi_{a,\beta}\}$ was the largest. In practice, it turned out that subtracting $\mathbf{z}_{a,\beta}$ when $\omega_{a,\beta}(\mathbf{d}) = \max\{\omega_{b,\alpha}(\mathbf{d}) \mid (b,\alpha) \in A \times \mathbb{F}_q\}$ works best. In light of Lemma 5.1, this is equivalent to always increase the smallest value in $\mathbf{v}$ by $q-1$.

6. Test results and further work

The algorithms for finding coset leaders with high weight for a given code described in this paper have been implemented on a computer. So far we have only concentrated on some codes for $q = 2$ and 3. This paper describes how to construct one $\mathbf{d}$-vector that hopefully will be a good $\mathbf{d}$. There are many random choices done in Algorithms 1 and 2, so iterating the construction of $\mathbf{d}$'s several times will, with high probability, result in different vectors each time. By constructing many $\mathbf{d}$-vectors, we hope to find a few which are good and produce coset leaders of high weight.

Not many results on the Newton radius for different classes of codes are known. In [2] the Newton radius is determined for all equidistant binary codes. The first-order binary Reed–Muller codes are studied in the same paper, and several bounds on the Newton radius are given for these codes. The authors of [2] have also conducted a straightforward search for unique coset leaders by randomly selecting $\mathbf{x}$ of weight w, and checking whether $\mathbf{x}$ is a unique coset leader. For the $[64,7]_2$ Reed–Muller code they tried 300 000 000 different $\mathbf{x}$-vectors of weight 24 without finding any of them to be unique coset leaders. For the $[128,8]_2$ Reed–Muller code they tried 200 000 000 vectors of weight 52 without finding any of them to be unique coset leaders.

Several unique coset leaders for the $[64,7]_2$ Reed–Muller code of weight 24 have been found using the algorithm described above. On three different executions, the

number of **d**'s that needed to be produced before one resulted in a unique coset leader of weight 24 were 96, 64 and 95, respectively.

For the $[128, 8]_2$ Reed–Muller code, many unique coset leaders of weight 52, and a few of weight 53 have been found. We ran the algorithm three times in search for a unique coset leader of weight 52. To produce one of weight 52 we needed to construct 37, 40 and 23 **d**'s, respectively. On three different searches for a unique coset leader of weight 53 we needed to try 4843, 4072 and 3756 different **d**'s before a unique coset leader of weight 53 was found.

Only a few general bounds on the Newton radius are known for codes over other alphabets than GF(2). The binary simplex codes can be generalized in two ways. One way is to let C be the $[(q^k-1)/(q-1), k]_q$ code where each column appears exactly once in the generator matrix. We will refer to this type of code as the *short* generalized simplex code. The $[q^k-1, k]_q$ code where each column appears exactly $q-1$ times in the generator matrix will be called the *long* generalized simplex code. Since the Newton radius and the covering radius are determined for the binary simplex codes, we have done some small searches for coset leaders of generalized simplex codes over $\mathbb{F}_3$. The results are listed below, and serve as first lower bounds for the covering radius and the Newton radius for these codes.

k	Short gen. simplex	Long gen. simplex
3	$\nu(C) \geqslant 5 \mid r(C) \geqslant 7$	$\nu(C) \geqslant 14 \mid r(C) \geqslant 16$
4	$\nu(C) \geqslant 21 \mid r(C) \geqslant 22$	$\nu(C) \geqslant 48 \mid r(C) \geqslant 49$
5	$\nu(C) \geqslant 67 \mid r(C) \geqslant 69$	$\nu(C) \geqslant 148 \mid r(C) \geqslant 150$

In [1] the following relation between the covering radius and the Newton radius is proven:

$$r(C) + (q-1)\nu(C) \leqslant (q-1)n - k - (q-2).$$

It is conjectured that this relation can be improved to

$$r(C) + (q-1)\nu(C) \leqslant (q-1)(n-k).$$

We have done some searching among codes over $\mathbb{F}_3$ for counter-examples to this relation, but have not been able to produce any. However, there are cases where the bound is met with equality, so if the improved bound is true, it is tight.

6.1. Further work

The tests done here indicate that the method for computing coset leaders presented in this paper is far better than doing a more straightforward search. On the author's workstation, finding a unique coset leader of weight 52 for the $[128, 8]_2$ Reed–Muller code takes less than a minute, and to find one of weight 53 takes less than one and a half hour.

On the other hand, it is not clear whether all coset leaders can be produced using our method. Maybe one needs to go deeper than spaces of dimension $k-2$ when

constructing **d** in order to find some particular coset leaders. When $v(C)$ and $r(C)$ are known, it should be noted that our algorithm has problems when trying to find (unique) coset leaders of these weights when the dimension is greater than 6.

It would also be nice to have an algorithm that produces a good **d** with probability 1 on each execution. When testing the algorithm, we have mostly used only binary and ternary codes. It appears that it is harder to produce good **d**'s when $q = 3$ than it is when $q = 2$. In general, it is probably easier to meet the requirements needed for removing negative numbers in **v** when q is small.

Acknowledgements

The author would like to thank Torleiv Kløve for his many helpful comments.

References

[1] E. Gabidulin, T. Kløve, On the Newton and covering radii of linear codes, IEEE Trans. Inform. Theory 45 (1999) 2534–2536.
[2] T. Helleseth, T. Kløve, The Newton radius of codes, IEEE Trans. Inform. Theory 43 (1997) 1820–1831.
[3] T. Kløve, Relations between the covering and Newton radii of binary codes, Discrete Math. 238 (2001) 81–88.
[4] W.W. Peterson, Error-Correcting Codes, MIT Press, Cambridge, MA, 1961.