On the minimum distance of $q$-ary nonlinear codes

Jaume Pujol, Mercè Villanueva, Fanxuan Zeng

Combinatorics, Coding and Security Group (CCSG)
Department of Information and Communications Engineering (dEIC)
Autonomous University of Barcelona (UAB)

June 10-13, CanaDAM 2013, St.John’s
Outline

Introduction

Minimum weight and minimum distance

Improvements on minimum weight and minimum distance

Minimum weight and minimum distance of $q$-ary codes

Conclusions and future research
Hamming Weight: or simply weight, of a vector $v = (v_1, \ldots, v_n)$ is the number of its entries which are nonzero.

Hamming Distance: or simply distance, between two vectors $v = (v_1, \ldots, v_n)$ and $w = (w_1, \ldots, w_n)$ is the number of entries where they differ.

An $(n, M, d)_q$ code $C$ is a subset of $\mathbb{F}_q^n$, where $\mathbb{F}_q$ is a finite field of $q$ elements.

We start from Binary Codes, which are subsets of $\mathbb{F}_2^n$. 
Introduction

Given a code $C$, the problem of storing $C$ in memory and finding out the minimum (Hamming) weight $w(C)$ and minimum (Hamming) distance $d(C)$ are well-known problems.

- If $C$ is linear, then it can be compactly represented using a generator matrix. Moreover, $d(C) = w(C)$.
- If $C$ is nonlinear, then a solution would be to know whether it has another structure or not.

For example, there are binary codes which have a $\mathbb{Z}_4$-linear or $\mathbb{Z}_2 \mathbb{Z}_4$-linear structure and, therefore, they can also be compactly represented using a quaternary generator matrix.
Given a code $C$, the problem of storing $C$ in memory and finding out the minimum (Hamming) weight $w(C)$ and minimum (Hamming) distance $d(C)$ are well-known problems.

- If $C$ is linear, then it can be compactly represented using a generator matrix. Moreover, $d(C) = w(C)$.

- If $C$ is nonlinear, then a solution would be to know whether it has another structure or not.

For example, there are binary codes which have a $\mathbb{Z}_4$-linear or $\mathbb{Z}_2\mathbb{Z}_4$-linear structure and, therefore, they can also be compactly represented using a quaternary generator matrix.
In general, binary codes without any of these structures can be represented as the union of cosets of a binary linear subcode of $C$.

$$C = \bigcup_{i=0}^{t} \left( K + c_i \right),$$

where $c_0 = 0$, $t + 1 = M/2^k$, $M = |C|$ and

$$K = \{ x \in \mathbb{Z}_2^n \mid x + C = C \}.$$

The kernel $K$ is the largest linear subcode for which it is true. The coset leaders are given by the set of $t$ vectors: $\{c_1, \ldots, c_t\}$. 

---

On the minimum distance of $q$-ary nonlinear codes

---

Introduction

---

In general, binary codes without any of these structures can be represented as the union of cosets of a binary linear subcode of $C$. 

$$C = \bigcup_{i=0}^{t} \left( K + c_i \right),$$

where $c_0 = 0$, $t + 1 = M/2^k$, $M = |C|$ and

$$K = \{ x \in \mathbb{Z}_2^n \mid x + C = C \}.$$

The kernel $K$ is the largest linear subcode for which it is true. The coset leaders are given by the set of $t$ vectors: $\{c_1, \ldots, c_t\}$. 

---

---
Introduction

Since $K$ is linear, the binary code $C$ can be represented by the generator matrix of $K$ and the set of coset leaders instead of a set of all its codewords.

Then, the code $C$ take up a memory space of order $O(n(k + t))$.

Example 1

Memory space of binary codes of length $n$, $M = 2^{19}$ codewords and different dimension of kernel:

<table>
<thead>
<tr>
<th>dimension of kernel $k$</th>
<th>0</th>
<th>1</th>
<th>...</th>
<th>17</th>
<th>19</th>
</tr>
</thead>
<tbody>
<tr>
<td>number coset leaders $t$</td>
<td>$2^{19}$</td>
<td>$2^{18} - 1$</td>
<td>...</td>
<td>3</td>
<td>0</td>
</tr>
<tr>
<td>memory space</td>
<td>$2^{19}n$</td>
<td>$2^{18}n$</td>
<td>...</td>
<td>$20n$</td>
<td>$19n$</td>
</tr>
</tbody>
</table>
Minimum weight and distance

• For linear codes, the best algorithm to obtain the minimum weight/distance based on the enumeration of some codewords is the Brouwer-Zimmermann algorithm.

• For nonlinear codes, there is no other algorithms but the enumeration of all codeswords, that is, the brute force algorithm.
A short introduction to Brouwer-Zimmermann’s Algorithm

Minimum weight and minimum distance

Minimum weight of enumerated vectors

Lower bound of unenumerated vectors $h(r+1)$
Minimum weight and distance

**Extend Coset:** Given a binary code \( C \) and a vector \( v \), denote 
\( K_v = K \cup (K + v) \) as an extend coset of \( C \). Since \( K \) is linear, then \( K_v \) is also linear, and can be constructed as \( \langle \text{Basis}(K), v \rangle \).

Let \( C = \bigcup_{i=0}^{t} (K + c_i) \) with \( t \geq 2 \).

**Proposition / Algorithm 1 (MinW)**

*The minimum weight of \( C \) is \( \min(\{w(K_{c_i}) \mid 1 \leq i \leq t\}) \).*

**Proposition / Algorithm 2 (MinD)**

*The minimum distance of \( C \) is \( \min(\{w(K_{c_i}) \mid 1 \leq i \leq t\} \cup \{w(K_{c_i+c_j}) \mid 1 \leq i < j \leq t\}) \).*
On the minimum distance of $q$-ary nonlinear codes

Minimum weight and minimum distance

Explanation of Algorithm 2 (MinD)

\[ d(K + c_i, K + c_j) = w(K + c_i + c_j) \]
Example 2

Let $K$ be the binary linear code of length $n = 31$, dimension $k = 5$, and $d(K) = 16$, constructed as the dual of the binary Hamming code of length $n = 31$. Let $C = \bigcup_{i=0}^{3} (K + c_i)$, where $c_0 = 0$, and the coset leaders are:

\[c_1 = (0010001110011010011110001011110)\]
\[c_2 = (0101101010111100101110100111101)\]
\[c_3 = (0000011100011101101000111010111)\]

It is easy to check that the minimum weight of $C$ is $w(C) = 10$ and its minimum distance $d(C) = 8$.

<table>
<thead>
<tr>
<th></th>
<th>minimum weight</th>
<th>minimum distance</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brute Force</td>
<td>0.00018s</td>
<td>0.00840s</td>
</tr>
<tr>
<td>Algorithm 1 / 2</td>
<td>0.00060s</td>
<td>0.00126s</td>
</tr>
</tbody>
</table>
Example 2

Let $K$ be the binary linear code of length $n = 31$, dimension $k = 5$, and $d(K) = 16$, constructed as the dual of the binary Hamming code of length $n = 31$. Let $C = \bigcup_{i=0}^{3} (K + c_i)$, where $c_0 = 0$, and the coset leaders are:

$c_1 = (0010001110011010011110001011110)$
$c_2 = (0101101010111100101110100111101)$
$c_3 = (000001110001110110100011101011)$

It is easy to check that the minimum weight of $C$ is $w(C) = 10$ and its minimum distance $d(C) = 8$.

<table>
<thead>
<tr>
<th></th>
<th>minimum weight</th>
<th>minimum distance</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Brute Force</strong></td>
<td>0.00018s</td>
<td>0.00840s</td>
</tr>
<tr>
<td><strong>Algorithm 1 / 2</strong></td>
<td>0.00060s</td>
<td>0.00126s</td>
</tr>
</tbody>
</table>

Note that sometimes a brute force calculation can be a faster way.
Time of computing $w(C)$ using Algorithm 1 (MinW) compared with brute force, for binary nonlinear codes of length $n = 100$, size $M = 2^{15} \cdot 31$, and kernel of dimension $k \in \{7, \ldots, 15\}$. 
Time of computing $d(C')$ using Algorithm 2 (MinD) compared with brute force, for binary nonlinear codes of length $n = 100$, size $M = 2^7 \cdot 31$, and kernel of dimension $k \in \{3, \ldots, 7\}$. 

![Graph showing time comparison between Algorithm MinD and brute force. The x-axis represents the dimension of the kernel, ranging from 3 to 7, and the y-axis represents the time in seconds, ranging from 0 to 140.]
All these algorithms are based on the enumeration of codewords, adding together codewords and deciding the minimum weight of codewords.

The performance measurement of these computations is referred as \textbf{work}. An estimate of the total work an algorithm requires is referred as \textbf{work factor}. 
Work factor of binary linear codes

For a binary linear code $K$ of length $n$ and dimension $k$, the minimum weight and minimum distance are the same, the work factor of computing them using Zimmermann algorithm is:

$$WF_{Zim} = (n - k) \left\lceil \frac{n}{k} \right\rceil \sum_{r=1}^{\bar{r}} \binom{k}{r}$$

where $\bar{r}$ is the smallest natural number such that

$$\left\lceil \frac{n}{k} \right\rceil (\bar{r} + 1) + \max(0, \bar{r} + 1 - (k - n \mod k)) \geq w(C).$$
Work factor of minimum weight

For a binary nonlinear code of length $n$, dimension of kernel $k$ and number of coset leaders $t$, we can estimate the work factor:

**Brute Force Algorithm:** Enumerating every codeword of the code and examining their weights, the work factor is:

$$n \cdot M = n \cdot (t + 1) \cdot 2^k$$

**Algorithm MinW:** The work factor is:

$$\sum_{i=1}^{t} \left( (n - (k + 1)) \left\lfloor \frac{n}{k + 1} \right\rfloor \sum_{r=1}^{\bar{r}_i} \binom{k + 1}{r} \right)$$

where $\bar{r}_i$ is the smallest natural number such that

$$\left\lfloor \frac{n}{k + 1} \right\rfloor (\bar{r}_i + 1) + \max(0, \bar{r}_i + 1 - (k + 1 - n \mod k + 1)) \geq w(K_{ci}).$$
Work factor of minimum distance

**Brute Force Algorithm:** Enumerating every pair of codewords of the code, and exam the distance between them, the work factor is:

\[ n \cdot \binom{M}{2} = n \cdot \binom{2^k(t+1)}{2} \]

**Algorithm MinD:** The work factor is:

\[
\sum_{i=0}^{t-1} \left( \sum_{j=i+1}^{t} ((n - (k + 1))\lceil n/(k + 1) \rceil \sum_{r=1}^{\bar{r}_{i,j}} \binom{k + 1}{r} ) \right)
\]

where \( \bar{r}_{i,j} \) is the smallest natural number such that

\[
\left\lfloor \frac{n}{(k + 1)} \right\rfloor (\bar{r}_{i,j}+1) + \max(0, \bar{r}_{i,j}+1-(k+1-n \mod (k+1)) ) \geq w(K_{c_i+c_j}).
\]
Work factor upper bounds

The minimum weight of an extend coset $K_v = K \cup (K + v)$ relies on the minimum weight of itself, so it is hard to estimate the work factor.

However, note that $w(K_v) \leq w(K)$, where $w(K)$ is the minimum weight of the kernel. Therefore, we can obtain an upper bound by replacing $w(K_v)$ with $w(K)$. 
Work factor upper bounds

**Minimum Weight Upper Bound:**

\[ t \cdot ((n - (k + 1)) \lceil n/(k + 1) \rceil) \sum_{r=1}^{\bar{r}} \binom{k + 1}{r} \]

**Minimum Distance Upper Bound:**

\[ \left( \frac{t + 1}{2} \right) \cdot ((n - (k + 1)) \cdot \lceil n/(k + 1) \rceil) \sum_{r=1}^{\bar{r}} \binom{k + 1}{r} \]

where \( \bar{r} \) is the smallest natural number such that

\[ \lceil n/(k + 1) \rceil (\bar{r} + 1) + \max(0, \bar{r} + 1 - (k + 1 - n \mod k + 1)) \geq w(K). \]
Work factor and work factor upper bound of minimum weight for binary codes of length $n = 100$, size $M = 2^{17} \cdot 31$, and dimension of kernel $k \in \{7, \ldots, 17\}$. 
Work factor and work factor upper bound of minimum distance for binary codes of length $n = 100$, size $M = 2^7 \cdot 31$, and dimension of kernel $k \in \{3, \ldots, 7\}$. 

![Bar chart showing work factor and upper bound of Algorithm MinD for different dimensions of kernel](chart.png)

- **Upper bound of Algorithm MinD**
- **Algorithm MinD**
Improvement on Algorithm MinW and MinD

According to Algorithm MinW and MinD, we can obtain the minimum weight and distance of a binary code with $t$ coset leaders by getting the minimum weight of several extend cosets.

Therefore, the codewords in the kernel are checked several times:
• $t$ times for minimum weight in Algorithm MinW.
• $\binom{t+1}{2}$ times for minimum distance in Algorithm MinD.
An improved algorithm is brought to avoid the repetition.

Algorithm 3 (IMinD)

1. Add the all-zero codeword into the set of coset leaders.
2. Add a coset leader to the basis of the kernel to form a binary linear code of rank $k + 1$ (for the added all-zero coset leader, the rank is still $k$).
3. Use Gaussian elimination to make the coordinates of information set on the coset leader be 0. Then delete these coordinates in all rows.
4. During every stage of enumerating $r$ rows in Zimmermann’s algorithm, select $r$ rows from the basis of the kernel and always select the coset leader row.
On the minimum distance of $q$-ary nonlinear codes

Improvements on minimum weight and minimum distance

Explanation of Algorithm 3 (IMinD)
Work factor and upper bounds of Algorithm IMinD

**Minimum Distance:**

\[
\sum_{i=0}^{t} \left( \sum_{j=i+1}^{t} ((n - k)\lceil n/k \rceil \sum_{r=1}^{\bar{r}_{i,j}} \binom{k}{r}) \right)
\]

where \(\bar{r}_{i,j}\) is the smallest natural number such that

\[
\left\lfloor \frac{n}{k} \right\rfloor (\bar{r}_{i,j} + 1) + \max(0, \bar{r}_{i,j} + 1 - (k - n \mod k)) \geq w(K_{c_i + c_j}).
\]

**Upper Bounds:**

\[
\left( \binom{t+1}{2} \right) + 1 \cdot (n - k) \cdot \left\lceil \frac{n}{k} \right\rceil \sum_{r=1}^{\bar{r}} \binom{k}{r}.
\]

where \(\bar{r}\) is the smallest natural number such that

\[
\left\lfloor \frac{n}{k} \right\rfloor (\bar{r} + 1) + \max(0, \bar{r} + 1 - (k - n \mod k)) \geq w(K).
\]
Work factor of minimum distance using Algorithm 2 (MinW) and 3 on codes of length $n = 100$, kernel $k \in \{7, \ldots, 15\}$, size $M = 2^{17} \cdot 31$. 
Minimum weight/distance of $q$-ary codes

**Problem:** If $K$ is a linear code,

**Binary Codes:** $K \cup (K + c)$ is a linear code which can be obtained by adding $c$ in the generator matrix.

**$q$-ary Codes:** $K \cup (K + c)$ is not necessarily a linear code, and it can not be obtained by adding $c$ in the generator matrix.

**Lemma 3**

For a $[n, k, d]_q$ code $K$ and a vector $v \in \mathbb{F}_q^n$, the code $K_v = \langle \text{Basis}(K), v \rangle$ is a linear code, and the minimum weight $w(K_v) = w(K(v))$, where $K(v) = K \cup (K + v)$. 
Let $C = \bigcup_{i=0}^{t} (K + c_i)$ be an $(n, M, d)_q$ code given by kernel $K$ and a list of coset leaders $\{c_1, c_2, ..., c_t\}$.

**Proposition / Algorithm 4 (MinWq)**

The minimum weight of $C$ is $\text{min}(\{w(K_{c_i}) \mid 1 \leq i \leq t\})$.

**Proposition / Algorithm 5 (MinDq)**

The minimum distance of $C$ is $\text{min}(\{w(K_{c_i}) \mid 1 \leq i \leq t\} \cup \{w(K_{c_i+c_j}) \mid 1 \leq i < j \leq t\})$. 
On the minimum distance of $q$-ary nonlinear codes

Minimum weight and minimum distance of $q$-ary codes

Improved minimum weight/distance of $q$-ary codes

Let $C = \bigcup_{i=0}^{t} (K + c_i)$ be an $(n, M, d)_q$ code given by kernel $K$ and a list of coset leaders $\{c_1, c_2, ..., c_t\}$.

**Proposition / Algorithm 6 (IMinWq)**

_The minimum weight of $C$ is $\min(\{w(K + c_i) \mid 1 \leq i \leq t\})$._

**Proposition / Algorithm 7 (IMinDq)**

_The minimum distance of $C$ is
\[\min(\{w(K + c_i) \mid 1 \leq i \leq t\} \cup \{w(K + c_i + c_j) \mid 1 \leq i < j \leq t\})\]._

Note that for the $q$-ary codes, since the coset $K + c_i$ contains only $1/q$ codewords as $K_v$, this improvement save much time.
Minimum distance work factor of Algorithm IMinDq

**Work Factor:**

$$
\sum_{i=0}^{t} \left( \sum_{j=i+1}^{t} (n - k) \left\lceil \frac{n}{k} \right\rceil \sum_{r=1}^{\bar{r}_{i,j}} \binom{k}{r} (q - 1)^{r-1} \right)
$$

where $\bar{r}_{i,j}$ is the smallest natural number such that

$$
\left\lfloor \frac{n}{k} \right\rfloor (\bar{r}_{i,j} + 1) + \max(0, \bar{r}_{i,j} + 1 - (k - n \mod k)) \geq w(K_{c_i+c_j}).
$$

**Upper Bound:**

$$
\left( \binom{t+1}{2} + 1 \right) \cdot (n - k) \cdot \left\lceil \frac{n}{k} \right\rceil \sum_{r=1}^{\bar{r}} \binom{k}{r} (q - 1)^{r-1}.
$$

where $\bar{r}$ is the smallest natural number such that

$$
\left\lfloor \frac{n}{k} \right\rfloor (\bar{r} + 1) + \max(0, \bar{r} + 1 - (k - n \mod k)) \geq w(K).
$$
Upper bounds of work factor of minimum distance using Algorithm IMinDq and brute force on codes over $\mathbb{F}_3$ of length $n = 100$, dimension of kernel $k \in \{3, \ldots, 9\}$, size $M = 3^9 \cdot 31$, and $w(K) = 35$. 

![Graph showing work factor and dimension of kernel for different values of $k$.]
Conclusions

- We applied the structure of the kernel and a set of coset leaders to represent a code $C$ to store it more compactly.

- Based on this structure, we developed several algorithms to decide the minimum weight and minimum distance of a code $C$.

- We evaluated the performance of the algorithms in theory and partly in practical test.
Future research

- Apply the algorithms, especially for the general $q$-ary codes.

- Try to find a point based on the parameters from where we can decide to use the improved algorithms.

- Use the minimum weight to decode nonlinear binary codes.

- Use these algorithms to look for better nonlinear codes with big minimum distance.
Bibliography


Thank you!