View metadata, citation and similar papers at core.ac.uk

brought to you by CORE

provided by Journal of Algebra Combinatorics Discrete Structures and Applications (JACODESMATH,...

J. Algebra Comb. Discrete Appl.
5(3) • 153–165

Received: 24 October 2017
Accepted: 5 September 2018

Journal of Algebra Combinatorics Discrete Structures and Applications

Batch codes from Hamming and Reed-Muller codes*

Research Article

Travis Baumbaugh, Yariana Diaz, Sophia Friesenhahn, Felice Manganiello, Alexander Vetter

Abstract: Batch codes, introduced by Ishai *et al.*, encode a string $x \in \Sigma^k$ into an m-tuple of strings, called buckets. In this paper we consider multiset batch codes wherein a set of t-users wish to access one bit of information each from the original string. We introduce a concept of optimal batch codes. We first show that binary Hamming codes are optimal batch codes. The main body of this work provides batch properties of Reed-Muller codes. We look at locality and availability properties of first order Reed-Muller codes over any finite field. We then show that binary first order Reed-Muller codes are optimal batch codes when the number of users is 4 and generalize our study to the family of binary Reed-Muller codes which have order less than half their length.

2010 MSC: 14G50

Keywords: Batch codes, Hamming codes, Reed-Muller codes

1. Introduction

Consider the situation where a certain amount of data, such as information to be downloaded, is distributed over a number of devices. We could have multiple users who wish to download this data. In order to reduce wait time, we look at locally repairable codes with availability as noted in [4]. A locally repairable code, with locality r and availability δ, provides us the opportunity to reconstruct a particular bit of data using δ disjoint sets of size at most r [12]. When we want to reconstruct a certain bit of information, this is the same as a Private Information Retrieval (PIR) code. However, we wish to examine a scenario where we reconstruct not necessarily distinct bits of information.

A possible answer to the more complex scenario seems to be a family of codes called batch codes, introduced by Ishai *et al.* in [6]. Batch codes were originally studied as a schematic for distributing data

[*] *This work was supported by NSF Grant DMS #1547399.*

Travis Baumbaugh (Corresponding Author), Felice Manganiello; Clemson University, United States (email: tbaumba@g.clemson.edu, manganm@clemson.edu).
Yariana Diaz; The University of Iowa, United States (email: yariana-diaz@uiowa.edu).
Sophia Friesenhahn; Willamette University, United States (email: sophmf@gmail.com).
Alexander Vetter; Villanova University, United States (email: avetter@villanova.edu).

http://dx.doi.org/10.13069/jacodesmath.466634

ISSN 2148-838X

across multiple devices and minimizing the load on each device and total amount of storage. We study (n, k, t, m, τ) batch codes, where n is the code length, k is the dimension of the code, t is the number of bits we wish to retrieve, m is the number of buckets, and τ is the maximum number of bits used from each bucket for any reconstruction of t bits. In this paper we seek to minimize the number of devices in the system and the load on each device while maximizing the amount of reconstructed data. In other words, we want to minimize $m\tau$ while maximizing t.

In Section 2, we formally introduce batch codes and summarize results from previous work on batch codes. We then introduce the concepts of locality and availability of a code. We conclude the section by introducing a concept of optimal batch codes.

After the background, we study the batch properties of binary Hamming codes and Reed-Muller codes. Section 3 focuses on batch properties of binary Hamming codes. We show that Hamming codes are optimal $(2^{s-1}, 2^s - 1 - s, 2, m, \tau)$ batch codes for $m, \tau \in \mathbb{N}$ such that $m\tau = 2^{s-1}$.

Section 4 is the main body of this work and provides batch properties of Reed-Muller codes. We first study the induced batch properties of a code $\mathcal{C}$ given that $\mathcal{C}^\perp$ is of a $(u \mid u + v)$-code construction with determined batch properties.

In Section 4.1 we study the locality and availability properties of first order Reed-Muller codes over any finite field. We find that the locality of $\mathcal{RM}_q(1, \mu)$ is 2 when $q \neq 2$ and 3 when the $q = 2$. Furthermore, we also show that its availability is $\left\lfloor \frac{q^\mu - 1}{2} \right\rfloor$ when $q \neq 2$, whereas when $q = 2$, the availability is $\frac{2^\mu - 1}{3}$ if μ is even and at least $\frac{2^\mu - 4}{4}$ otherwise. In Section 4.2 we show that binary first order Reed-Muller codes are optimal batch codes for $t = 4$. We first look at the specific $\mathcal{RM}(1, 4)$ case and achieve parameters $(16, 5, 4, m, \tau)$ such that $m\tau = 10$. We then prove a general result that any Reed-Muller code with $\rho = 1$ and $\mu \geq 4$ has batch properties $(2^\mu, \mu + 1, 4, m, \tau)$ for any $m, \tau \in \mathbb{N}$ such that $m\tau = 10$.

Finally, in Section 4.3 we generalize our study of Reed-Muller codes and look at properties of $\mathcal{RM}(\rho, \mu)$ for all values of ρ and conclude our study by presenting batch properties $(2^\mu, k, 4, m, \tau)$ such that $m\tau = 10 \cdot 2^{2\rho - 2}$ for $\mathcal{RM}(\rho', \mu)$ where $\mu \in \{2\rho + 2, 2\rho + 3\}$ and $\rho' \leq \rho$.

2. Background

In 2004, Ishai *et al.* [6] introduced the following definition of batch codes:

Definition 2.1. *An (n, k, t, m, τ) batch code over an alphabet Σ encodes a string $x \in \Sigma^k$ into an m-tuple of strings, called buckets, of total length n such that for each t-tuple of distinct indices $i_1, \ldots, i_t \in [k] = \{1, ..., k\}$, the entries $x_{i_1}, \ldots, x_{i_t}$ can be decoded by reading at most τ symbols from each bucket.*

We can view the buckets as servers and the symbols used from each bucket as the maximal load on each server. In the above scenario, a single user is trying to reconstruct t bits of information. This definition naturally leads to the concept of multiset batch codes which have nearly the same definition as above, but the indices $i_1, \ldots, i_k \in [k]$ are not necessarily distinct. This means we have t users who each wish to reconstruct a single element. This definition in turn relates to private information retrieval (PIR) codes which are similar to batch codes but instead look to reconstruct the same bit of information t times. Another notable type of batch code defined in [6] is a primitive multiset batch code where the number of buckets is $m = n$. In this research, the queries are considered to happen at the same time, while the asynchronous case is considered in [10].

The following are useful lemmas proven in [6]:

Lemma 2.2. *An (n, k, t, m, τ) batch code for any τ implies an $(n\tau, k, t, m\tau, 1)$ batch code.*

Lemma 2.3. *An $(n, k, t, m, 1)$ batch code implies an $(n, k, t, \lceil \frac{m}{\tau} \rceil, \tau)$ batch code.*

Much of the related research involves primitive multiset batch codes with a systematic generator matrix. In [6], the authors give results for some multiset batch codes using subcube codes and Reed-Muller codes. They use a systematic generator matrix, which often allows for better parameters. Their goal was to maximize the efficiency of the code for a fixed number of queries t. The focus of research on batch codes then shifted to combinatorial batch codes. These were first introduced by [9]. They are replication based codes using various combinatorial objects that allow for efficient decoding procedures. We do not consider combinatorial batch codes but some relevant results can be found in [9], [2], [3], and [11].

Next, the focus of research turned to linear batch codes, which use classical error-correcting codes. The following useful results are proven in [7]:

Theorem 2.4. *Let $\mathcal{C}$ be an $[n, k, t, n, 1]$ linear batch code over $\mathbb{F}_2$ with generator G. Then, G is a generator matrix of the classical error-correcting $[n, k, d]_2$ linear code where $d \geq t$.*

Theorem 2.5. *Let $\mathcal{C}_1$ be an $[n_1, k, t_1, n_1, 1]_q$ linear batch code and $\mathcal{C}_2$ be an $[n_2, k, t_2, n_2, 1]_q$ linear batch code. Then, there exists an $[n_1 + n_2, k, t_1 + t_2, n_1 + n_2, 1]_q$ linear batch code.*

Theorem 2.6. *Let $\mathcal{C}_1$ be an $[n_1, k_1, t_1, n_1, 1]_q$ linear batch code and $\mathcal{C}_2$ be an $[n_2, k_2, t_2, n_2, 1]_q$ linear batch code. Then, there exists an $[n_1 + n_2, k_1 + k_2, min(t_1, t_2), n_1 + n_2, 1]_q$ linear batch code.*

Because of the vast amount of information on classical error-correcting codes, we use these as our central focus in this paper. As is often the case, studying the properties of the dual codes helps us find efficient batch codes.

Next, [13] considers restricting the size of reconstruction sets. These are similar to codes with locality and availability:

Definition 2.7. *A code $\mathcal{C}$ has locality $r \geq 1$ if for any $c \in \mathcal{C}$, any entry in c can be recovered by using at most r other entries of c.*

Definition 2.8. *A code $\mathcal{C}$ with locality $r \geq 1$ has availability $\delta \geq 1$ if for any $c \in \mathcal{C}$, any entry of c can be recovered by using one of δ disjoint sets of at most r other entries of y*

The restriction on the size of reconstruction sets can be viewed as trying to minimize total data distribution. We restrict the size of our reconstruction sets to the locality of the code. By using this restriction, we find multiset batch codes with optimal data distribution. For cyclic codes, the locality can be derived from the following in [5]:

Lemma 2.9. *Let $\mathcal{C}$ be an $[n, k, d]$ cyclic code, and let d' be the minimum distance of its dual code $\mathcal{C}^{\perp}$. Then, the code $\mathcal{C}$ has all symbol locality $d' - 1$.*

This relies on each entry being in the support of a minimal weight dual code word. We generalize this lemma to the following one.

Lemma 2.10. *Let $\mathcal{C} \subseteq \mathbb{F}_q^n$ be a linear code and let d' be the minimum distance of $\mathcal{C}^{\perp}$. If $\mathcal{C}^{\perp}$ is generated by its minimum weight codewords and*

$$\bigcup_{\lambda \in \mathcal{C}^{\perp}} \operatorname{supp}(\lambda) = [n], \tag{1}$$

then $\mathcal{C}$ has all symbol locality $d' - 1$.

Proof. Condition (1) implies that no coordinate of $\mathcal{C}$ is independent on the others. If the minimum weight codewords generate $\mathcal{C}^{\perp}$, then each coordinate of $\mathcal{C}$ is in the support of at least one minimum weight codeword of $\mathcal{C}^{\perp}$. This implies the all symbol locality $d' - 1$ of $\mathcal{C}$. □

Condition (1) is a reasonable condition for a code. Without it the code $\mathcal{C}$ would have non recoverable coordinates.

We give here a bound that relates the locality property of a linear code with its batch properties.

Lemma 2.11. *Let $\mathcal{C}$ be an $[n, k, t, m, \tau]$ linear batch code with minimal locality r. Then it holds that*

$$m\tau \geq (t-1)r + 1. \tag{2}$$

Proof. We consider such a code $\mathcal{C}$. If each entry has at least one reconstruction set with fewer than r elements, then by the definition of locality, $\mathcal{C}$ has locality less than r, a contradiction to r being the minimal locality. Therefore, there exists at least one entry for which all recovery sets are of size at least r. If we wish to recover this entry t times, then we may read the entry itself and then make use of $t-1$ disjoint recovery sets, each of size r. This implies reading at least $(t-1)r+1$ entries, and since we may read only τ entries from each of the m buckets, we must have that $m\tau \geq (t-1)r+1$. □

From the perspective of individual devices storing bits of data, $m\tau$ represents the total amount of data read to provide t pieces of the original data. To minimize bandwidth usage in the case where the entries of a codeword represent nodes on a network, we must minimize $m\tau$.

Definition 2.12. *A $[n, k, t, m, \tau]$ linear batch code $\mathcal{C}$ with minimal locality r is optimal if it satisfies Condition (2) with equality.*

We now show that binary Hamming codes are optimal linear batch codes.

3. Hamming codes

Hamming codes were first introduced in 1950 by Richard Hamming. In what follows, we consider binary Hamming codes over $\mathbb{F}_2$. The parameters of binary Hamming codes are shown in [8].

Definition 3.1. *For some $s \geq 2$, let $H \in \mathbb{F}_2^{2^s-1\times s}$ be a matrix whose columns are all of the nonzero vectors of $\mathbb{F}_2^s$. Let $n = 2^s - 1$. We use H as our parity check matrix and define the binary Hamming Code:*

$$\mathcal{H}_s := \{c \in \mathbb{F}_2^n \mid cH^T = 0\}$$

It is well-known that $\mathcal{H}_s$ is a $[2^s-1, 2^s-1-s, 3]$ cyclic code. Its dual code, the simplex code, is a $[2^s-1, s, 2^{s-1}]$ cyclic code. Thus, by Lemmma 2.9, the locality of $\mathcal{H}_s$ is $2^{s-1}-1$. We now present the batch properties of binary Hamming Codes.

Theorem 3.2. *A binary $[n = 2^s-1, k = 2^s-1-s]$ Hamming code is an optimal batch code with properties $[2^s-1, 2^s-1-s, 2, m, \tau]$, for any $m, \tau \in \mathbb{N}$ such that $m\tau = 2^{s-1}$.*

Proof. First, note that $m\tau \geq (2-1)(2^{s-1}-1)+1 = 2^{s-1}$. The buckets for $m = 2^{s-1}$, $\tau = 1$ are constructed as follows. Let H be the parity check matrix of a binary Hamming code, $\mathcal{H}$, with columns $h_j \in \mathbb{F}_2^s$ for $1 \leq j \leq n$. If $h_a + h_b = 1$ (the all ones column), then we place a and b into the same bucket. Note that because $h_\ell = 1$ in H, ℓ is placed into its own bucket. Let $r_d \in \mathbb{F}_2^n$ be the rows in H such that $1 \leq d \leq (n-k) = s$. For any $c \in \mathcal{H}$, $r_d \cdot c^T = 0$, and thus $\sum_{i\in\mathrm{supp}(r_d)} c_i = 0$. As a result of this construction, if $a, b \in \mathrm{supp}(r_d)$, then entry d of $h_a + h_b$ is 0, so a and b cannot be in the same bucket. Therefore, $\sum_{i\in\mathrm{supp}(r_d)} c_i = 0$ only involves bits in separate buckets. Hence, any bit in a codeword can be written as a linear combination of bits in separate buckets. Now, we show how to reconstruct any two bits of information.

- Case 1: If a and b are in separate buckets, then use c_a and c_b.
- Case 2: Suppose a and b are in the same bucket. We can take c_a itself. To construct c_b, we choose an r_d such that $b \in \mathrm{supp}(r_d)$. Then, we can write $c_b = \sum_{i\in\mathrm{supp}(r_d)\setminus\{b\}} c_i$, which we know only contains bits in disjoint buckets.

Every bucket has cardinality 2 aside from the bucket corresponding to the all ones column in H, so this construction gives us exactly 2^{s-1} buckets. Thus, we have shown that the batch properties hold for $m = 2^{s-1}$ and $\tau = 1$. Further, Lemma 2.3 implies that this is true for any m, τ such that $m\tau = 2^{s-1}$.

Note that the locality of $\mathcal{H}$ is $2^{s-1} - 1$, and therefore, $t = 2$ is also maximal. Suppose instead that we could have $t \geq 3$. Then in particular, each entry must be reconstructible at least 3 times. We may take the entry itself, but then there must be at least 2 other reconstruction sets used which are disjoint and of size $2^{s-1} - 1$. These would correspond to two codewords in the dual code of weight 2^{s-1} with the intersection of their support being only the given entry. The sum of these codewords will thus have weight $2^{s-1} + 2^{s-1} - 2 = 2^s - 2$. However, the all ones vector is also in the dual code. Adding this vector to the sum will produce a codeword of weight one, a contradiction. Thus, $t = 2$ is maximal. □

Example 3.3. *We now give an example for $s = 3$. This Hamming Code is a $[7, 4]$-linear code, and the dual code is a $[7, 3]$-linear code.*

The parity check matrix H is as follows:

$$H = \begin{pmatrix} 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Thus, the buckets are:

$$\{1, 6\}, \{2, 5\}, \{3, 4\}, \{7\}$$

We note that for general batch codes we are only interested in reconstructing information bits. However, we are able to obtain any pair of bits in the codeword. Additionally, we note that although $m\tau$ is optimized, we wish to find batch codes where $t > 2$. We desire a larger value as we are concerned with practical applications, and the goal is to quickly distribute data. Thus we move on to Reed-Muller codes, where we are able to obtain larger t values.

4. Reed-Muller codes

Reed-Muller codes are well known linear codes. We give some basic properties of these codes, but an interested reader can find more information in [1].

Definition 4.1. *Let $\mathbb{F}_q[X_1, \ldots, X_\mu]$ be the ring of polynomials in μ variables with coefficients in $\mathbb{F}_q$ and let $\mathbb{F}_q^\mu = \{P_1, \ldots, P_n\}$ (so $n = q^\mu$). The q-ary Reed-Muller code, $\mathcal{RM}_q(\rho, \mu)$ is defined as:*

$$\mathcal{RM}_q(\rho, \mu) \coloneqq \{(f(P_1), \ldots, f(P_n)) \mid f \in \mathbb{F}_q[X_1, \ldots, X_\mu]^\rho\},$$

where $\mathbb{F}_q[X_1, \ldots, X_\mu]^\rho$ is the set of all multivariate polynomials over $\mathbb{F}_q$ of total degree at most ρ.

It is known that if $\rho < \mu(q-1)$ then the dual of a Reed-Muller code $\mathcal{RM}_q(\rho, \mu)$ is the Reed-Muller code $\mathcal{RM}_q(\mu(q-1) - 1 - \rho, \mu)$ [1].

In the binary case, Reed-Muller codes can be equivalently defined using the $(u \mid u+v)$-code construction. For completeness, we first give a description of the $(u \mid u+v)$-code construction and the related generator matrix, which can be found in [8].

Definition 4.2. *Given two linear codes $\mathcal{C}_1, \mathcal{C}_2$ with identical alphabets and block lengths, we construct a new code $\mathcal{C}$ as follows:*

$$\mathcal{C} \coloneqq \{(u \mid u+v) \mid u \in \mathcal{C}_1,\, v \in \mathcal{C}_2\}.$$

Let G, G_1, and G_2 be the generator matrices for the codes $\mathcal{C}$, $\mathcal{C}_1$, and $\mathcal{C}_2$, respectively, where $\mathcal{C}$ is obtained from $\mathcal{C}_1$ and $\mathcal{C}_2$ via the $(u \mid u+v)$-construction. Then we have

$$G := \begin{pmatrix} G_1 & G_1 \\ 0 & G_2 \end{pmatrix}$$

From this, we have the following proposition.

Proposition 4.3. *Let $\mathcal{C}_1$ be an $[n, k_1, d_1]$-linear code, $\mathcal{C}_2$ an $[n, k_2, d_2]$-linear code, and $\mathcal{C}$ the code obtained from $\mathcal{C}_1, \mathcal{C}_2$ via the $(u \mid u+v)$-construction. Then, $\mathcal{C}$ is an $[2n, k, d]$-code where $k = k_1 + k_2$ and $d = \min\{2d_1, d_2\}$.*

Later, our focus will be on $q = 2$, so when referring to $\mathcal{RM}_2(\rho, \mu)$ we omit the 2 for convenience. We obtain the following equivalent definition of a binary Reed-Muller code.

Definition 4.4. *Let $\rho < \mu$. A binary Reed-Muller code $\mathcal{RM}(\rho, \mu)$ is defined as follows:*

$$\mathcal{RM}(\rho, \mu) := \{(u \mid u+v) \mid u \in \mathcal{RM}(\rho, \mu-1),\ v \in \mathcal{RM}(\rho-1, \mu-1)\}$$

where $\mathcal{RM}(0, \mu) := 1$ of length 2^μ, and $\mathcal{RM}(\mu, \mu) := I_{2^\mu}$.

As a consequence if $G_{\rho,\mu}$ is the generator matrix of the code $\mathcal{RM}(\rho, \mu)$, then

$$G_{\rho,\mu} := \begin{pmatrix} G_{\rho,\mu-1} & G_{\rho,\mu-1} \\ 0 & G_{\rho-1,\mu-1} \end{pmatrix}$$

We now examine the batch properties of the $(u \mid v+u)$-code construction. The first notable result comes from codes that are contained in other codes:

Theorem 4.5. *Let $\mathcal{C}_1, \mathcal{C}_2$ be codes of length n and dimension k_1 and k_2, respectively such that $\mathcal{C}_2 \subseteq \mathcal{C}_1$. If $\mathcal{C}_1$ is a (n, k_1, t, m, τ) batch code, then $\mathcal{C}_2$ is a (n, k_2, t, m, τ) batch code.*

Proof. Note that $\mathcal{C}_1^\perp \subseteq \mathcal{C}_2^\perp$ because $\mathcal{C}_2 \subseteq \mathcal{C}_1$. Therefore the same parity check equations for $\mathcal{C}_1$ apply to $\mathcal{C}_2$. Thus, to recover information in $\mathcal{C}_2$, we may use the same parity check equations we would in $\mathcal{C}_1$, which implies $\mathcal{C}_2$ is at least a (n, k_2, t, m, τ) batch code. □

We now introduce results for a $(u \mid u+v)$-code construction.

Theorem 4.6. *Let $n, k_1, k_2 \in \mathbb{N}$ such that $n \geq k_1 \geq k_2$, and let $\mathcal{C}$ be a $[n, k_1 + k_2]$ linear code. Then let $\mathcal{C}^\perp$ be a $(u \mid u+v)$-code construction of $\mathcal{C}_1^\perp$ and $\mathcal{C}_2^\perp$, where*

- *${\mathcal{C}_1}^\perp$ is an $[n, n-k_1]$ linear code and ${\mathcal{C}_2}^\perp$ is an $[n, n-k_2]$ linear code, and*
- *$\mathcal{C}_2^\perp \subseteq \mathcal{C}_1^\perp$.*

If $\mathcal{C}_2$ is a $(n, k, t, m\tau)$ batch code, then $\mathcal{C}$ is a $(2n, k_1 + k_2, t, m, \tau)$ batch code.

Proof. The first two parameters of $\mathcal{C}$ follow from the definition of a $(u \mid u+v)$ construction. Let $\mathcal{C}$ be constructed as described, and let $\mathcal{C}_2$ be an (n, k, t, m, τ) batch code. Then consider any t-tuple of indices $i_1, \ldots, i_t \in [2n]$ and let $i'_j = i_j$ if $i_j \in [n]$ and $i'_j = i_j - n$ otherwise. Then we know that $i'_1, \ldots, i'_t \in [n]$, and so by the batch properties of $\mathcal{C}_2$ there exist t disjoint recovery sets for the entries in these indices, the union of which consists of at most τ entries in each of the m buckets.

If for all $i \in \{n+1, \ldots, 2n\}$, we place i in the same bucket as $i - n$, then this results in m buckets for $\mathcal{C}$. If we consider the recovery sets from above, if $R_{i'_j}$ is the recovery set for i'_j, and $i_j \in [n]$, then we claim that $R'_{i'_j} = R_{i'_j}$ is a recovery set for i_j in $\mathcal{C}$. This is because the recovery set comes from a vector

$v \in \mathcal{C}_2^{\perp}$, and by construction, since $\mathcal{C}_2^{\perp} \subseteq \mathcal{C}_1^{\perp}$, we have that $(v|0), (0|v) \in \mathcal{C}^{\perp}$. Likewise, if $i_j \notin [n]$, then by using $(0|v)$, we find that $R'_{i'_j} = \{i + n | i \in R_{i'_j}\}$ is a recovery set for i_j.

Since the original $R_{i'_j}$ are all disjoint, so are the $R'_{i'_j}$, and so we have t disjoint recovery sets, the union of which consists of at most τ elements from each of m buckets, and so $\mathcal{C}$ is a $(2n, k_1 + k_2, t, m, \tau)$ batch code. □

Because binary Reed-Muller codes have parity check matrices that satisfy the above properties, we turn to that family of codes.

4.1. Locality and availability properties of $\mathcal{RM}_q(1, \mu)$

Reed-Muller codes for which $\rho = 1$ are known as first order Reed-Muller codes. We look at the properties using the polynomial evaluation definition of Reed-Muller codes. We begin with a result in the q-ary case.

Theorem 4.7. *Let $\mathbb{F}_q^{\mu} = \{P_i \mid 1 \leq i \leq 2^{\mu} = n\}$ be the set of evaluation points for $\mathcal{RM}_q(1, \mu)$. Then $(\lambda_1, \ldots, \lambda_n) \in \mathcal{RM}_q(1, \mu)^{\perp}$ if and only if*

$$\sum_{i=1}^{n} \lambda_i P_i = 0 \textit{ and } \sum_{i=1}^{n} \lambda_i = 0. \tag{3}$$

Proof. First, if $(\lambda_1, \ldots, \lambda_n)$ is in the dual code, then by definition,

$$\sum_{i=1}^{n} \lambda_i f(P_i) = 0 \tag{4}$$

for every polynomial $f \in \mathbb{F}_q[x_1, \ldots, x_{\mu}]^1$. In particular, note that for any $1 \leq k \leq \mu$, from $f_k(x_1, \ldots, x_{\mu}) = x_k$, we have

$$\sum_{i=1}^{n} \lambda_i f_k(P_i) = \sum_{i=1}^{n} \lambda_i p_{i,k} = 0,$$

where $p_{i,k}$ is the kth entry of point P_i. We may gather these equations together for $1 \leq k \leq \mu$ to write the linear combination

$$\sum_{i=1}^{n} \lambda_i P_i = 0.$$

We then consider $f_0 = 1$, and Equation (4) becomes $\sum_{i=1}^{n} \lambda_i = 0$, and so we have this direction.

For the other direction, assume that

$$\sum_{i=1}^{n} \lambda_i P_i = 0 \text{ and } \sum_{i=1}^{n} \lambda_i = 0.$$

Then in particular, for any $1 \leq k \leq \mu$, we have $\sum_{i=1}^{n} \lambda_i p_{i,k} = 0$, and so for any $f \in \mathbb{F}_q[x_1, \ldots, x_{\mu}]^1$, by linearity we have that

$$\sum_{i=1}^{n} \lambda_i f(P_i) = \sum_{i=1}^{n} \lambda_i \left[f_0 + \sum_{k=1}^{\mu} f_k p_{i,k} \right] = f_0 \sum_{i=1}^{n} \lambda_i + \sum_{k=1}^{\mu} f_k \sum_{i=1}^{n} \lambda_i p_{i,k} = 0,$$

and thus $(\lambda_1, \ldots, \lambda_{\mu}) \in \mathcal{RM}(1, \mu)^{\perp}$. □

From Theorem 4.7 we obtain the following corollaries:

Corollary 4.8. *The minimum distance of $\mathcal{RM}_q(1,\mu)^\perp$ is 4 if $q=2$ and 3 otherwise.*

Proof. Let $q=2$ and suppose by way of contradiction that the minimum weight is 2. Then there exist two distinct points that sum to zero. This is not possible, and thus the minimum weight must be greater than 2. Note that the only choice of λ_i is 1, and thus the sum is 0 if and only if there are an even number of points. Therefore, the weight of the codewords is a multiple of 2, and thus the minimum weight is not 3. The following points are in $\mathcal{P}$ (for $\mu \geq 2$):

$$P_0 = (0,0,0,\dots,0)^T, P_1 = (1,0,0,\dots,0)^T, P_2 = (0,1,0,\dots,0)^T, \text{ and } P_3 = P_1 + P_2.$$

These points satisfy the conditions, and thus the minimum distance for characteristic 2 is 4.

If $q \neq 2$, let $P_1 = (0,0,0,0,\dots,0)^T, P_2 = (1,0,0,0,\dots,0)^T, P_3 = (-a,0,0,0,\dots,0)^T \in \mathbb{F}_q^\mu$ and the entries of λ be $-(a+1), a$, and 1 corresponding to the positions of P_1, P_2, and P_3, respectively, and 0 otherwise. Then, If $a \neq -1, 0$, λ satisfies Equations (3).

Suppose there exists a $\lambda \in \mathcal{RM}_q(1,\mu)^\perp$ with weight 2. Then we have two distinct points $P_i, P_j \in \mathbb{F}_q^\mu$ and $\lambda_i, \lambda_j \in \mathbb{F}_q$ such that $\lambda_j = -\lambda_i$ and $\lambda_k = 0$ for all $k \neq i, j$. Our two conditions imply:

$$\lambda_i P_i - \lambda_i P_j = 0 \implies P_i = P_j,$$

A contradiction to the two points being distinct. □

Corollary 4.9. *When $q=2$, every codeword in $\mathcal{RM}(\rho,\mu)$ satisfies Equation 3 for $\rho \leq \mu - 2$.*

Proof. The dual code of $\mathcal{RM}(1,\mu)$ is $\mathcal{RM}(\mu-2,\mu)$ and $\mathcal{RM}(\rho_1,\mu) \subset \mathcal{RM}(\rho_2,\mu)$ if $\rho_1 < \rho_2$. Thus, any codeword in $\mathcal{RM}(\rho,\mu)$ is also in $\mathcal{RM}(\mu-2,\mu)$. Therefore, Theorem 4.7 implies our claim. □

Theorem 4.10. *Let $q \neq 2$. Then $\mathcal{RM}_q(1,\mu)$ has locality 2 and availability $\delta = \left\lfloor \frac{q^\mu - 1}{2} \right\rfloor$.*

Proof. Let $P_a \in \mathbb{F}_q^\mu$ be an evaluation point. Then consider any $\alpha \in \mathbb{F}_q$ such that $\alpha \neq 0, -1$. We have that $1 + \alpha + (-\alpha - 1) = 0$, and will find corresponding points to use in the reconstruction of P_a. For any choice of $P_b \in \mathbb{F}_q^\mu$, $P_b \neq P_a$, take

$$P_c = (\alpha+1)^{-1}(P_a + \alpha P_b).$$

Upon rearrangement, we have that $P_a + \alpha P_b + (-\alpha-1)P_c = 0$. Furthermore, we find that $P_c \neq P_a, P_b$. If $P_c = P_a$, then our equation becomes $P_a + \alpha P_b + (-\alpha - 1)P_a = 0$, which simplifies to $\alpha P_b - \alpha P_a = 0$, which would contradict $P_b \neq P_a$. Likewise, $P_c = P_b$ would imply $P_a + \alpha P_b + (-\alpha-1)P_b = 0$, which becomes $P_a - P_b = 0$, another contradiction. A simple counting argument tells us that there are $\left\lfloor \frac{q^\mu - 1}{2} \right\rfloor$ choices of pairs P_b, P_c for P_a, and each of these corresponds to a unique $\lambda \in \mathcal{RM}_q(1,\mu)^\perp$ of weight 3 that can be used to recover c_a, the supports of which intersect only in $\{a\}$. Thus, the locality is 2 and the availability is $\left\lfloor \frac{q^\mu - 1}{2} \right\rfloor$. □

As proven in [1] the minimum-weight codewords are generators of a Reed-Muller code $\mathcal{RM}_{p^s}(\mu,\rho)$ where p is a prime number and $0 \leq \rho \leq \mu(p^s - 1)$ if and only if either $m = 1$ or $\mu = 1$ or $\rho < p$ or $\rho > (\mu-1)(p^s-1) + p^{s-1} - 2$. Together with Lemma 2.10 it follows that these Reed-Muller codes have all symbol availability. Thus, in the following theorems, we only consider the availability of P_1 as this implies all symbol availability.

Theorem 4.11. *$\mathcal{RM}(1,\mu)$ has availability $\delta = \frac{2^\mu - 1}{3}$ when μ is even.*

Proof. An inductive argument on μ satisfies this claim. We look at the sum of evaluation points to prove our claim. We are looking for $\frac{2^\mu-1}{3}$ disjoint sets of three points of $\mathbb{F}_q^\mu$ that sum to $(0,\dots,0)^T \in \mathbb{F}_q^\mu$.

It is easy to verify the claim for $\mu = 2$ since there is only one equation for which this is true

$$(0,1)^T + (1,0)^T + (1,1)^T = (0,0)^T.$$

Now assume the claim is true for $\mu = 2k$, we show that it is true also for $\mu = 2k+2$. We have $\frac{2^{2k}-1}{3}$ disjoint sets of three points that all sum to $\bar{P}_1 = (0,\dots,0)^T \in \mathbb{F}_q^{2k}$. Let $P_1 = (0,\dots,0)^T \in \mathbb{F}_q^{2k+2}$. For any choice of set of points $\{S_1, S_2, S_3\} \subset \mathbb{F}_q^{2k}$ that sum to $\bar{P}_1$ in $\mathbb{F}_q^{2k}$ it holds that

$$\begin{aligned}
(S_1^T|0,0)^T + (S_2^T|0,0)^T + (S_3^T|0,0)^T &= P_1\\
(S_1^T|1,0)^T + (S_2^T|0,1)^T + (S_3^T|1,1)^T &= P_1\\
(S_1^T|0,1)^T + (S_2^T|1,1)^T + (S_3^T|1,0)^T &= P_1\\
(S_1^T|1,1)^T + (S_2^T|1,0)^T + (S_3^T|0,1)^T &= P_1.
\end{aligned} \tag{5}$$

Additionally it also holds that

$$(\bar{P}_1^T|1,1)^T + (\bar{P}_1^T|1,0)^T + (\bar{P}_1^T|0,1)^T = P_1^T. \tag{6}$$

The four equations in (5) all use distinct sets of points because S_1, S_2, and S_3 are distinct. Now, there are $\frac{2^{2k}-1}{3}$ sets of distinct points like S_1, S_2, and S_3. Thus, we have a total of

$$4 \cdot \frac{2^{2k}-1}{3} + 1 = \frac{2^{2k+2}-1}{3}.$$

Note that the extra one comes from Equation (6). Also note that our total is an integer as

$$2^{2k+2} \equiv 4^{k+1} \equiv 1^{k+1} \equiv 1 \pmod 3.$$

Because of this, every single coordinate is used, and thus we have maximal availability. □

Theorem 4.12. *$\mathcal{RM}(1,\mu)$ has availability δ at least $\frac{2^\mu-4}{4}$ when μ is odd.*

Proof. We prove this by induction on μ. For $\mu = 3$, let $S_1 = (1,0,0)^T$, $S_2 = (0,1,0)^T$ and $S_3 = S_1 + S_2$, then

$$S_1 + S_2 + S_3 = \bar{P}_1 = (0,0,0)^T.$$

No combination of the four remaining points of $\mathbb{F}_2^3$ sum to $\bar{P}_1$. Here we have availability $1 = \frac{2^3-4}{4}$, and so we have our base case.

Now assume that, for $\mu = 2k+1$, where $k \geq 1$, we have that the availability of $\mathcal{RM}(1,\mu)$ is at least $\frac{2^\mu-4}{4}$, and there are at least 3 points that are not used in any recovery set for P_1. Let S_1, S_2, and S_3 be any three points in a recovery set of $P_1 \in \mathbb{F}_2^\mu$. Then for $\mu+2$, the disjoint sets of three points that sum to $\tilde{P}_1 = (0,\dots,0) \in \mathbb{F}_2^{\mu+5}$ can be defined by the equations:

$$\begin{aligned}
(S_1^T|0,0)^T + (S_2^T|0,0)^T + (S_3^T|0,0)^T &= \tilde{P}_1\\
(S_1^T|1,0)^T + (S_2^T|0,1)^T + (S_3^T|1,1)^T &= \tilde{P}_1\\
(S_1^T|0,1)^T + (S_2^T|1,1)^T + (S_3^T|1,0)^T &= \tilde{P}_1\\
(S_1^T|1,1)^T + (S_2^T|1,0)^T + (S_3^T|0,1)^T &= \tilde{P}_1
\end{aligned}$$

This results in at least $4\left(\frac{2^\mu-4}{4}\right) = 2^\mu - 4$ possible recovery sets. However, we also have points T_1, T_2, and T_3 that are not used in $\mathbb{F}_2^\mu$, and so we may also define the following equations:

$$(\bar{P}_1^T|1,0)^T + (T_1^T|0,1)^T + (T_1^T|1,1)^T = \tilde{P}_1$$
$$(\bar{P}_1^T|0,1)^T + (T_2^T|1,1)^T + (T_2^T|1,0)^T = \tilde{P}_1$$
$$(\bar{P}_1^T|1,1)^T + (T_3^T|1,0)^T + (T_3^T|0,1)^T = \tilde{P}_1$$

Adding these, we have availability of at least $2^\mu - 4 + 3 = 2^\mu - 1 = \frac{2^{\mu+2}-4}{4}$, and so our property holds for $\mu+2$ as well. We also note that $3(2^\mu - 1) + 3 = 3 \cdot 2^\mu < 4 \cdot 2^\mu = 2^{\mu+2}$, and so there are at least 3 unused points. Thus, by induction, we have that for every $k \geq 1$, when $\mu = 2k+1$, the availability of $\mathcal{RM}(1,\mu)$ is at least $\frac{2^\mu-4}{4}$

Thus we have achieved a lower bound on δ. Note, however, that we have not shown that this is necessarily an optimal construction. □

We now study the batch properties of Reed-Muller codes.

4.2. Batch properties of $\mathcal{RM}(1,\mu)$

Theorem 4.13. *The linear code $\mathcal{RM}(1,4)$ is a $(16,5,4,m,\tau)$ batch code for any $m,\tau \in \mathbb{N}$ such that $m\tau = 10$.*

Proof. First, note that the dual code of $\mathcal{RM}(1,4)$ is $\mathcal{RM}(2,4)$, which informs us how to reconstruct elements of the codewords. The generator matrix for $\mathcal{RM}(1,4)$ can be recursively constructed as follows:

$$G_{1,4} = \begin{pmatrix} G_{1,3} & G_{1,3} \\ 0 & G_{0,3} \end{pmatrix}$$

It can be verified that any query of 4 coordinates of a codeword in $\mathcal{RM}(1,4)$ is possible with the following partition into buckets:

$$\{1\}, \{2\}, \{3\}, \{4\}, \{5,6\}, \{7,8\}, \{9,11\}, \{10,12\}, \{13,16\}, \{14,15\}.$$

In the above case, $m = 10$ and $\tau = 1$. By Lemma 2.3, this holds for any $m,\tau \in \mathbb{N}$ such that $m\tau = 10$. □

We now show how to extend this construction to $\mathcal{RM}(1,\mu)$ for any $\mu \geq 4$.

Theorem 4.14. *Any first order Reed-Muller code, $\mathcal{RM}(1,\mu)$, with $\mu \geq 4$, has batch properties $(n,k,4,m,\tau)$ for any $m,\tau \in \mathbb{N}$ such that $m\tau = 10$.*

Proof. We will proceed by induction. First, we have just shown this for the base case where $\mu = 4$. Now, assume that for some $\mu - 1 \geq 4$, we have that $\mathcal{RM}(1,\mu-1)$ has batch properties $(n,k,4,m,\tau)$. Recall that the dual code of $\mathcal{C} = \mathcal{RM}(1,\mu)$ is $\mathcal{C}^\perp = \mathcal{RM}(\mu-2,\mu)$. Then as Reed-Muller codes follow the $(u \mid u+v)$-construction, $\mathcal{C}^\perp$ is the $(u \mid u+v)$-code construction of $\mathcal{C}_1^\perp = \mathcal{RM}(\mu-2,\mu-1)$ and $\mathcal{C}_2^\perp = \mathcal{RM}(\mu-3,\mu-1)$. Since $\mathcal{RM}(\mu-3,\mu-1) \subseteq \mathcal{RM}(\mu-2,\mu-1)$, we have $\mathcal{C}_2^\perp \subseteq \mathcal{C}_1^\perp$. This means we may apply Theorem 4.6. Since $\mathcal{C}_2 = \mathcal{RM}(1,\mu-1)$, we know that $\mathcal{C}$ is also a $(n,k,4,m,\tau)$ batch code. By induction, this is now true for any $\mu \geq 4$. □

Since the locality of these codes is $r = 3$, for $t = 4$, we have $m\tau = 10 = (4-1)\cdot 3 + 1 = (t-1)r + 1$, and thus we have optimal batch properties. We now extend this even further for most $\mathcal{RM}(\rho,\mu)$.

4.3. Batch properties of $\mathcal{RM}(\rho, \mu)$

We begin with a preliminary result that uses the recursive construction of Reed-Muller binary codes.

Lemma 4.15. *Let $a \in \mathcal{RM}(\rho - 1, \mu - 2)$. Then*

$$(a|a|0|0), (a|0|a|0), (a|0|0|a), (0|a|a|0), (0|a|0|a), (0|0|a|a) \in \mathcal{RM}(\rho, \mu),$$

where $0 \in \mathbb{F}_2^\mu$.

Proof. Let

$$G = \begin{pmatrix} G_{\rho,\mu-1} & G_{\rho,\mu-1} \\ 0 & G_{\rho-1,\mu-1} \end{pmatrix}$$

be the generator of $\mathcal{RM}(\rho, \mu)$. Recursively, we obtain that

$$G = \begin{pmatrix} G_{\rho,\mu-2} & G_{\rho,\mu-2} & G_{\rho,\mu-2} & G_{\rho,\mu-2} \\ 0 & G_{\rho-1,\mu-2} & 0 & G_{\rho-1,\mu-2} \\ 0 & 0 & G_{\rho-1,\mu-2} & G_{\rho-1,\mu-2} \\ 0 & 0 & 0 & G_{\rho-2,\mu-2} \end{pmatrix} \tag{7}$$

From the second and third block rows in matrix (7), we see that for any $a \in \mathcal{RM}(\rho - 1, \mu - 2)$, the second row implies $(0|a|0|a) \in \mathcal{RM}(\rho, \mu)$, and the third row implies $(0|0|a|a) \in \mathcal{RM}(\rho, \mu)$. Note that our code is linear, and thus $(0|a|0|a) + (0|0|a|a) = (0|a|a|0) \in \mathcal{RM}(\rho, \mu)$. Finally, note that since $\mathcal{RM}(\rho - 1, \mu - 2) \subseteq \mathcal{RM}(\rho, \mu - 2)$, the first row implies $(a|a|a|a) \in \mathcal{RM}(\rho, \mu)$, and so combining this with the previous vectors, we find that $(a|a|0|0), (a|0|a|0), (a|0|0|a) \in \mathcal{RM}(\rho, \mu)$. □

We now show the batch properties of $\mathcal{RM}(\rho, \mu)$ for $\rho \geq 1$ and $\mu \geq 2\rho + 2$.

Theorem 4.16. *Let $\rho \geq 1$ and $\mu \in \{2\rho + 2, 2\rho + 3\}$. Then, for $\rho' \leq \rho$, $\mathcal{RM}(\rho', \mu)$ is a $(n, k, 4, m, \tau)$ linear batch code for any $m, \tau \in \mathbb{N}$ such that $m\tau = 10 \cdot 2^{2\rho-2}$.*

Proof. We focus on the case where $\mu = 2\rho + 2$ as the case $\mu = 2\rho + 3$ proceeds with similar steps.

If $\mathcal{RM}(\rho, 2\rho + 2)$ is a $(n, k, 4, m, \tau)$ linear batch code for any $m, \tau \in \mathbb{N}$ such that $m\tau = 10 \cdot 2^{2\rho-2}$, then it follows from Theorem 4.5 that for any $\rho' \leq \rho$, the code $\mathcal{RM}(\rho', 2\rho + 2)$ is a $(n, k, 4, m, \tau)$ batch code for any $m, \tau \in \mathbb{N}$ such that $m\tau = 10 \cdot 2^{2\rho-2}$. Thus, we need only prove that this property holds for $\mathcal{RM}(\rho, 2\rho + 2)$.

We proceed by induction on ρ. Note that in Section 4.2, the claim is true for $\rho = 1$, the base cases with $\mu = 4, 5$. Assume that $\rho \geq 1$. We show that the properties hold for $\mathcal{RM}(\rho + 1, 2(\rho + 1) + 2) = \mathcal{RM}(\rho + 1, 2\rho + 4)$, assuming that $\mathcal{RM}(\rho, 2\rho + 2)$ is a $(n, k, 4, m, \tau)$ linear batch code for any $m, \tau \in \mathbb{N}$ such that $m\tau = 10 \cdot 2^{2\rho-2}$. In particular, we may choose $\tau = 1$ and have $m = 10 \cdot 2^{2\rho-2}$ buckets.

We now examine $\mathcal{RM}(\rho + 1, 2\rho + 4)$. The dual code of $\mathcal{RM}(\rho + 1, 2\rho + 4)$ is $\mathcal{RM}(\rho + 2, 2\rho + 4)$. By Lemma 4.15, for any $a \in \mathcal{RM}(\rho + 1, 2\rho + 2) = \mathcal{RM}(\rho, 2\rho + 2)^\perp$, we have

$$(a|a|0|0), (a|0|a|0), (a|0|0|a), (0|a|a|0), (0|a|0|a), (0|0|a|a) \in \mathcal{RM}(\rho + 2, 2\rho + 4).$$

This provides a way to produce parity check equations for $\mathcal{RM}(\rho+1, 2\rho+4)$ from those for $\mathcal{RM}(\rho, 2\rho+1)$, which in turn provides a way to make recovery sets for the former from those for the latter, as each vector corresponds to a recovery set for every index in its support.

Each bucket $B = \{i_1, \ldots, i_\ell\}$ for $\mathcal{RM}(\rho, 2\rho + 2)$ can be made into 4 buckets for $\mathcal{RM}(\rho + 1, 2\rho + 4)$: $B_1 = B$, $B_2 = B + n = \{i_1 + n, \ldots, i_\ell + n\}$, $B_3 = B + 2n$, and $B_4 = B + 3n$. This results in $4 \cdot 10 \cdot 2^{2\rho-2} = 10 \cdot 2^{2\rho} = 10 \cdot 2^{2(\rho+1)-2}$ buckets, and we must show that any set of 4 indices may be recovered by drawing at most 1 entry from each bucket.

Consider any tuple of 4 indices $i_1, i_2, i_3, i_4 \in [4n] = \bigcup_{s=0}^{3}([n] + sn)$ and let $s_k = \left\lfloor \frac{i_k - 1}{n} \right\rfloor$ for $k \in [4]$. Then let $i'_k = i_k - s_k n$, so that $i'_1, i'_2, i'_3, i'_4 \in [n]$. By the induction hypothesis, we have recovery sets $R'_1, R'_2, R'_3, R'_4 \subseteq [n]$ for these indices in $\mathcal{RM}(\rho, 2\rho + 2)$. These recovery sets are sets such that $i'_k \in R'_k$ for all $k \in [4]$ and

1. $(R'_k \setminus \{i'_k\}) \cap (R_j \setminus \{i'_j\}) = \emptyset$ for $k, j \in [4]$ with $k \neq j$
2. $\bigcup_{k=1}^{4}(R'_k \setminus \{i'_k\})$ consists of at most 1 index in each bucket

Each R'_k is either $\{i'_k\}$ or the support of some vector $a \in \mathcal{RM}(\rho + 1, 2\rho + 2)$. If $|R'_k| = 1$, then let $R_k = R'_k + s_k n$. Otherwise, let $R_k = (R'_k + s_k n) \cup (R'_k + s'_k n)$. By Lemma 4.15, we know that if $s'_k \neq s_k$, R_k is the support of some vector in $\mathcal{RM}(\rho + 2, 2\rho + 4)$, and so this is a valid recovery set. We must now show that these recovery sets have the desired properties given the correct choice of s'_k values.

We now note that since indices are being recreated from $d = |\{s_1, s_2, s_3, s_4\}|$ different quarters of $[4n]$, we can take at least d of the recovery sets to be singletons. Further, assume that we take as many recovery sets to be singletons as possible. In particular, this means that no recovery set will contain more than one index in each bucket. This is because the only way R_k could contain two indices in a bucket would be if R'_k did. Since R'_k is a proper recovery set, it could only contain two indices in one bucket if one of those indices was i'_k. Then that bucket is not used in any other recovery set, and so we could instead take $R'_k = \{i'_k\}$, as per our assumption.

This leaves at most $4 - d$ recovery sets which are not singletons and require a subset in a second quarter. Assume without loss of generality that these are $R_1, \ldots, R_{d-4}$. Then we may let $s'_1, \ldots, s'_{d-4}$ be the elements of $\{0, 1, 2, 3\} \setminus \{s_1, s_2, s_3, s_4\}$. Since these are distinct, the only way some $R_k \setminus \{i_k\}$ and $R_j \setminus \{i_j\}$ could have a nonempty intersection would be if condition 1 was violated. Thus, condition 1 also holds for the R_k. We have already covered the fact that none of the $R'_k + s'_k n$ will not contain more than one index in each bucket, and since these are in separate quarters, the only way $\bigcup_{k=1}^{4}(R_k \setminus \{i_k\})$ would contain more than one index in a bucket would be if some elements being recovered in the same quarter had $(R'_k \setminus \{i'_k\}) \cup (R'_j \setminus \{i_j\})$ consisting of more than 1 index in some bucket. This would violate condition 2, and so we know that the R_ks also satisfy 2.

Thus, this code is a $(4n, k', 4, 10 \cdot 2^{2(\rho+1)-2}, 1)$ batch code, and by Lemma 2.3, we know that $\mathcal{RM}(\rho + 1, 2(\rho + 1) + 2)$ is a $(4n, k', 4, m, \tau)$ batch code for any $m, \tau \in \mathbb{N}$ such that $m\tau = 10 \cdot 2^{2(\rho+1)-2}$. This completes the induction step, and so for any $\rho \geq 1$, $\mathcal{RM}(\rho, 2\rho + 2)$ is a $(4n, k', 4, m, \tau)$ batch code for any $m, \tau \in \mathbb{N}$ such that $m\tau = 10 \cdot 2^{2\rho-2}$. □

5. Conclusion

This work focuses on batch properties of binary Hamming and Reed-Muller code.

The high locality of binary Hamming codes implies their availability to be at most 1. Binary Hamming codes can be viewed as linear batch codes retrieving queries of at most 2 indices, the trivial case. Nonetheless, we prove that for $t = 2$, binary Hamming codes are actually optimal $(2^{s-1}, 2^s - 1 - s, 2, m, \tau)$ batch codes for $m, \tau \in \mathbb{N}$ such that $m\tau = 2^{s-1}$.

We turn to binary Reed-Muller codes for optimal batch codes that allow larger queries, meaning t-tuples with t larger than 2. This research direction is motivated by the large availability of first order Reed-Muller codes as showed in the paper. We prove the optimality of first order Reed-Muller codes for $t = 4$.

Finally we generalize our study to Reed-Muller codes $\mathcal{RM}(\rho, \mu)$ which have order less than half their length by proving that they have batch properties $(2^\mu, k, 4, m, \tau)$ such that $m\tau = 10 \cdot 2^{2\rho-2}$ for $\mathcal{RM}(\rho', \mu)$ where $\mu \in \{2\rho + 2, 2\rho + 3\}$ and $\rho' \leq \rho$.

Acknowledgment: The authors are grateful to Clemson University for hosting the REU at which this work was completed. The REU was made possible by an NSF Research Training Group (RTG) grant

(DMS #1547399) promoting Coding Theory, Cryptography, and Number Theory at Clemson.

References

[1] E. F. Assmus, J. D. Key, Designs and Their Codes, volume 103 of Cambridge Tracts in Mathematics, Cambridge University Press, Cambridge, 1992.

[2] S. Bhattacharya, S. Ruj, B. Roy, Combinatorial batch codes: A lower bound and optimal constructions, Adv. Math. Commun. 6(2) (2012) 165–174.

[3] C. Bujtás, Z. Tuza, Optimal combinatorial batch codes derived from dual systems, Miskolc Math. Notes 12(1) (2011) 11–23.

[4] A. G. Dimakis, K. Ramchandran, Y. Wu, C. Suh, A survey on network codes for distributed storage, Proc. IEEE 99(3) (2011) 476–489.

[5] P. Huang, E. Yaakobi, H. Uchikawa, P. H. Siegel, Binary linear locally repairable codes, IEEE Trans. Inform. Theory 62(11) (2016) 6268–6283.

[6] Y. Ishai, E. Kushilevitz, R. Ostrovsky, A. Sahai, Batch codes and their applications, Proc. 36th Annu. ACM Symp. Theory Comput. (2004) 262–271.

[7] H. Lipmaa, V. Skachek, Linear batch codes, In Coding Theory and Applications, CIM Series in Mathematical Sciences 3 (2015) 245–253.

[8] F. J. MacWilliams, N. J. A. Sloane. The Theory of Error–correcting Codes, North–Holland Publishing Co., Amsterdam–New York–Oxford, 1977.

[9] M. B. Paterson, D. R. Stinson, R. Wei, Combinatorial batch codes, Adv. Math. Commun. 3(1) (2009) 13–27.

[10] A.–E. Riet, V. Skachek, E. K. Thomas, Asynchronous batch and PIR codes from hypergraphs, preprint, 2018, arXiv:1806.00592.

[11] N. Silberstein, A. Gál, Optimal combinatorial batch codes based on block designs, Des. Codes Cryptogr. 78(2) (2016) 409–424.

[12] V. Skachek, Batch and PIR codes and their connections to locally repairable codes, In Network Coding and Subspace Designs, Signals Commun. Technol., Springer, Cham, (2018) 427–442.

[13] E. K. Thomas, V. Skachek, Constructions and bounds for batch codes with small parameters, In Coding Theory and Applications, Springer, Cham, (2017) 283–295.