Coding Theory

Arpon Basu

Last updated July 9, 2023

Contents

§0. Notation

For any $n \in \mathbb{N} := \{1, 2, \ldots\}$, we define $[n]$ to be the set $\{1, 2, \ldots, n\}$.

Given two strings s_1, s_2, we denote by $s_1||s_2$ the concatenation of s_1 and s_2.

Let q be a prime power. Then $\mathbb{F}_q$ denotes the finite field of order q.

We shall often identify $\mathbb{F}_q^m$ and $\mathbb{F}_{q^m}$ with each other, where q is a prime power. In most cases, the exact bijection (which is equivalent to specifying a degree m irreducible polynomial in $\mathbb{F}_q$) will not be important, and will thus be left unspecified.

For any matrix $X \in \mathbb{F}^{n \times m}$, where $\mathbb{F}$ is a field, we define the *spark* of X to be the size of the smallest linearly *dependent* set of *columns* of X.
Now, note that while appending an extra column to a maximal linearly independent set of columns yields a linearly dependent set of columns, this obtained set of columns may not be the *smallest* linearly dependent set of columns. Thus the spark of X is at most $\mathrm{rank}(X) + 1$, but it may not necessarily equal $\mathrm{rank}(X) + 1$.
Also, note that the spark of a matrix may not necessarily equal the spark of its transpose.

§1. A Note to the Reader

We have mainly followed this book [GRS22] (Chapters 1-6) and these [Gur10] notes for this report.
We would also like to make a special point about pre-requisites: Coding theory makes heavy use of the theory of finite fields. Unfortunately, as is the case with most mathematical theories, a comprehensive study of finite fields will consume too much of the reader's time, and distract her somewhat from her purpose of learning coding theory. Consequently, most texts on coding theory include a very brief description of finite fields for aiding the reader. However, the author feels that this is inadequate.
Thus, the reader is suggested this [For] reading for the theory of finite fields. This is not too extensive, yet it covers all the requisite material along with proofs wherever required.
Many proofs in coding theory use the properties of the entropy function. However, presenting the properties of the entropy function in between the chapter can disrupt the narrative. Thus, an entire chapter (Appendix A.2) worth of content discussing the properties of the entropy function has been relegated to the appendix. The reader is advised to consult it whenever required.

§2. Introduction and Basic Definitions

One of the characteristic features of human language is that it has a lot of redundancy: Consequently, as is quite common in colloquial speech, even if someone speaks in a manner that is grammatically or otherwise incorrect, the meaning of the sentence is usually transmitted through. Take for example this sentence: "Never know them let move next your". Most human readers can, after a moment's thought, get that this sentence is the scrambling of some correct sentence. The reason they can do this is that English speech contains information beyond what is required to be conveyed, and consequently, in cases of errors, some extra redundancy/structure helps us recover the original meaning of the sentence.
For similar reasons, some amount of error correction capabilities are desired within data transmission in digital systems too, especially because data is prone to be corrupted/mutilated, in which case it becomes important to extract what the original uncorrupted message was.
From this need arose the field of coding theory, which tells us how one may design *codes*, which, through some redundancy, can correct certain errors.

Definition 2.1. (Codes) Let Σ be a set of alphabets, with $|\Sigma| = q$. A **code** of block length n is a subset of Σ^n.
We often denote $|C|$ by M.

Remark. A few remarks are in order:

1. Note that $q \geq 2$.
2. Note that q need not be a constant: It can vary with n, as is the case with certain codes.

We also define

Definition 2.2. (Dimension of a Code) Given a code $C \subseteq \Sigma^n$, we define the dimension of C to be

$$k := \log_q |C|$$

Remark. Note that:-

1. Since $C \subseteq \Sigma^n$, $|C| \leq |\Sigma^n| = q^n$, and consequently $k \leq n$. Furthermore, assuming C is non-empty, we also have that $k \geq 0$.
2. The dimension of a code can be non-integral.

Finally, the degree of redundancy in a code is specified by its *rate*, which is defined below.

Definition 2.3 (Rate of a code)**.** The rate R of a code $C \subseteq \Sigma^n$ with dimension k is defined as

$$R := \frac{k}{n}$$

At this point, we give some concrete examples of codes to better elucidate the definitions given above.

Example (Parity Code)**.** Let $\Sigma = \{0, 1\}$, and thus $q = 2$. Then the parity code $C_\oplus$ for t-bit strings is defined as

$$C_\oplus(x_1, x_2, \ldots, x_t) := (x_1, x_2, \ldots, x_t, x_1 \oplus x_2 \oplus \cdots \oplus x_t), (x_1, x_2, \ldots, x_t) \in \{0, 1\}^t$$

Note that $C_\oplus = \{(x_1, x_2, \ldots, x_t, x_1 \oplus x_2 \oplus \cdots \oplus x_t) : (x_1, x_2, \ldots, x_t) \in \{0, 1\}^t\} \subseteq \{0, 1\}^{t+1}$, and consequently, we have $n = t + 1$. Moreover, we also have $k = t$, since $|C| = 2^t$, which implies that $k = \log_2(2^t) = t$. Furthermore, the rate of this code is $\frac{t}{t+1}$.

Example (Repetition Code)**.** Let $\Sigma = \{0, 1\}$, and thus $q = 2$. Then the 3-repetition code $C_{3,\text{rep}}$ for t-bit strings is defined as

$$C_{3,\text{rep}}(x_1, x_2, \ldots, x_t) := (x_1, x_1, x_1, x_2, x_2, x_2, \ldots, x_t, x_t, x_t), (x_1, x_2, \ldots, x_t) \in \{0, 1\}^t$$

Note that $C_{3,\text{rep}} = \{(x_1, x_1, x_1, x_2, x_2, x_2, \ldots, x_t, x_t, x_t) : (x_1, x_2, \ldots, x_t) \in \{0, 1\}^t\} \subseteq \{0, 1\}^{3t}$, and consequently, we have $n = 3t$. Moreover, we also have $k = t$, since $|C| = 2^t$, which implies that $k = \log_2(2^t) = t$. Furthermore, the rate of this code is $\frac{t}{3t} = \frac{1}{3}$.

2.1. Error Correction

Before we can come to error correction through codes, we must define the most central notion in coding theory: the Hamming distance.

Definition 2.4 (Hamming Distance)**.** Given $u, v \in \Sigma^n$, we define the Hamming distance $\Delta(u, v)$ between u and v as the number of positions in which u, and v differ, ie:-

$$\Delta(u, v) := |\{i : u_i \neq v_i\}|$$

Remark. One can verify that the Hamming distance is a metric on Σ^n. In particular, Δ obeys the triangle inequality.

A very similar notion is the notion of Hamming weight.

Definition 2.5 (Hamming Weight)**.** Let $\Sigma = \{0, 1, \ldots, q-1\}$. Then the Hamming weight $\text{wt}(v)$ of $v \in \Sigma^n$ is defined to be $\Delta(v, 0^n)$.

We now define our model of data corruption explicitly.

Definition 2.6 (e-error Channel)**.** A n-symbol e-error channel is defined to be a function Ch: $\Sigma^n \mapsto \Sigma^n$ such that $\Delta(\text{Ch}(v), v) \leq e$ for every $v \in \Sigma^n$.

Remark. Note that inherent in our conception of an e-error channel is the notion that at most a certain number of errors (namely e) can occur during our transmission, and these errors may occur on any subset of bit(s) of the transmitted code. Such a model of noise is known as the "worst-case" or adversarial model of noise, and this model was proposed by Hamming. It is important to know that there exist other models of noise: For example, the *Shannon model of noise* assumes that every bit of the transmitted message is equally likely to be flipped with some probability p, where p is the parameter of our noise model.
Unsurprisingly, the quantitative error-correction capabilities of a code are dependent on the noise model being used. Unless we specify explicitly, we shall always use the Hamming noise model.

Also, associated with every code is an *encoding and decoding mechanism*, which are described below.

Definition 2.7 (Encoding)**.** Let $C \subseteq \Sigma^n$ be a code, and consider some well-ordering of C, which naturally leads to an *encoding function* $E : [|C|] \mapsto \Sigma^n$. Consequently, if $\mathcal{M}$ is our well-ordered set of messages such that we have a bijection $\tau : \mathcal{M} \mapsto [|C|]$, then one can define the *encoding* of a message m to be $E(\tau(m)) =: C(m)$.

Remark. As we shall see later, the well-orderings on C and $\mathcal{M}$ are not completely arbitrary: We shall often impose some additional structure on Σ while constructing codes, which then dictates a canonical order on C and $\mathcal{M}$.

Definition 2.8 (Decoding)**.** Let C be a code. Then a function $D : \Sigma^n \mapsto \mathcal{M}$ is called a decoding function for C.

One of the most common decoding functions is the *Maximum Likelihood Decoding function*: The MLD decoding function is defined as follows:

$$D(v) := \text{argmin}_{c \in C} \Delta(v, c)$$

Note that for the notion of an MLD decoder to be consistently defined, it is necessary that *for every permissible $v \in \Sigma^n$, there is a unique codeword in C which is closest to v*. We shall examine this issue at length a bit later. Also, unless specified otherwise, we shall always assume our decoding function to be an MLD decoder.
We can now formally define what we mean by error detection.

Definition 2.9 (Error Correction)**.** A code $C \subseteq \Sigma^n$ is said to be e-error correcting if there exists a decoding function D such that for every message m, and any e-error channel Ch, we have $D(\text{Ch}(C(m))) = m$.

Remark. We don't assume any particular decoder in the above definition.
In a similar vein,

Definition 2.10 (Error Detection)**.** A code $C \subseteq \Sigma^n$ is said to be e-error detecting if there exists a detecting algorithm D such that for every message m, and for every $v \in \Sigma^n$ received across our channel satisfying $\Delta(C(m), v) \leq e$, D outputs a 1 if $v = C(m)$, and outputs 0 otherwise.

Here are some examples detailing how error correction and detection can be performed by codes.

Example (Error correction by $C_{3,\text{rep}}$)**.** We show that up to 1-bit flip can be corrected by this code: Indeed, consider the 3-bit block where the error occurred. If the block originally had 3 zeros, then it will now have 2 zeros, and similarly, if the block originally had 3 ones, then it will now have 2 ones.
Consequently, if we take the majority number of bits among every 3-bit block, we can reconstruct the original message.
Also, note that this code can *not* correct 2-bit flips: Indeed, consider the 3-bit block "000". Suppose the first two bits got flipped, so the transmitted message is "110". Now, the decoder on the other side can't guess if this "110" was produced by two-bit flips from "000", or one-bit flip from "111".

Example (Error detection by the parity code $C_\oplus$)**.** Note that $C_\oplus = \{(x_1, x_2, \ldots, x_t, x_1 \oplus x_2 \oplus \cdots \oplus x_t) : (x_1, x_2, \ldots, x_t) \in \{0,1\}^t\}$. Consequently, if we take the XOR of all $(t+1)$ bits of any element of $C_\oplus$, we get $x_1 \oplus x_2 \oplus \cdots \oplus x_t \oplus (x_1 \oplus x_2 \oplus \cdots \oplus x_t)$, which equals 0.
Consequently, the parity code can detect an odd number of errors (Note that when $\Sigma = \{0,1\}$, an error is just a bit flip): Indeed, if an odd number of bits are flipped in any element of $C_\oplus$, the collective XOR of all the values changes from 0 to 1, which tells us that there has been an error somewhere.
For similar reasons, note that the parity code will *fail* to detect an even number of errors.

2.1.1. Distance of a Code

Definition 2.11 (Distance of a Code)**.** Let C be a code. We define the distance d of C to be

$$d := \min_{\substack{x,y \in C \\ x \neq y}} \Delta(x, y)$$

The distance of a code is the minimum separation between any two codewords of a given code C. The reason this notion is so important is made clear by the theorem below.

Theorem 2.1. Let C be a code. Then the following are equivalent:

1. The distance of C is d.
2. C is $\left\lceil \frac{d-2}{2} \right\rceil$-error correcting.
3. C is $(d-1)$-error detecting.

Proof. We assume the MLD decoder for this theorem.

- $(1 \implies 2)$: We must prove that the MLD decoder finds a unique codeword $c \in C$ which is closest to any $y \in \Sigma^n$ for which $\min_{c \in C} \Delta(y, c) \leq \left\lceil \frac{d-2}{2} \right\rceil$. Assume for the sake of contradiction there exist $c_1, c_2 \in C$ such that c_1, c_2 are both the closest members of C to y, ie:- $\Delta(y, c_1) = \Delta(y, c_2)$. Then by the triangle inequality we have
$$\Delta(c_1, c_2) \leq \Delta(c_1, y) + \Delta(y, c_2) \leq 2 \cdot \left\lceil \frac{d-2}{2} \right\rceil < d$$
which contradicts the definition of the distance of C.
- $(1 \implies 3)$: Consider the transmitted message y. Since $\Delta(y, C) < d$, if y is corrupted, then $y \notin C$ [1]. Consequently, our error correction algorithm declares an error if and only if $y \notin C$.
- $(\neg 1 \implies \neg 2)$: Since $\neg 1$ holds, the distance of C is $< d$. Thus assume the distance of C is $d - k, k > 0$. Also assume for the sake of simplicity that $d - k$ is even [2], and let $c_1, c_2 \in C$ be such that $\Delta(c_1, c_2) = d - k$. WLOG let $c_1 = \ell || \ell_2^{(1)} || \ell_3^{(1)}, c_2 = \ell || \ell_2^{(2)} || \ell_3^{(2)}$, where the length of ℓ is $n - d + k$, and length of $\ell_2^{(i)}$ as well as $\ell_3^{(i)}$ is $\frac{d-k}{2}$ for $i \in \{1, 2\}$. Furthermore, $\ell_2^{(i)} \neq \ell_3^{(i)}$ for $i \in \{1, 2\}$. Then consider $y := \ell || \ell_2^{(1)} || \ell_3^{(2)}$. Clearly $\Delta(y, c_1) = \Delta(y, c_2) = \frac{d-k}{2}$. Now, a decoder can't decide if y was generated by corrupting c_1 or c_2, and thus we demonstrated an instance for which the decoder can't correct $\frac{d-k}{2} \leq \frac{d-1}{2}$ errors.
- $(\neg 1 \implies \neg 3)$: If there exist codewords c_1, c_2 such that $\Delta(c_1, c_2) < d$, then by changing $< d$ bits of c_1, we can get c_2. Consequently, if we receive c_2 across the channel, we can't be sure if it is the codeword c_2, or the mutilated codeword c_1.

■

2.2. Hamming Bound

Before we conclude this section, we shall establish a fundamental bound constraining the parameters of a code.

Definition 2.12. A code $C \subseteq \Sigma^n$ with dimension k and distance d is also known as a $(n, k, d)_q$ code.

Definition 2.13 (Hamming Ball)**.** For any $v \in \Sigma^n$, and any $\ell \in \mathbb{N}$, we define
$$B_{q,n}(v, \ell) := \{y \in \Sigma^n : \Delta(v, y) \leq \ell\}$$

[1] Note that since the distance between any two codewords is $\geq d$, if we corrupt any codeword in $< d$ positions, we can never obtain another codeword

[2] the odd case is similar, just some ceilings and floors may need to be added here and there

Remark. We shall often suppress the subscripts q, n if they are clear from the context.

Theorem 2.2 (Generalized Hamming Bound)**.** For any $(n, k, d)_q$ code, we have

$$k \leq n - \log_q \left(\sum_{i=0}^{\lfloor \frac{d-1}{2} \rfloor} \binom{n}{i} (q-1)^i \right)$$

Proof. For notational convenience, set $e = \left\lfloor \frac{d-1}{2} \right\rfloor$. Since the distance of our code is d, we have that $B(c_1, e) \cap B(c_2, e) = \varnothing$ for any $c_1, c_2 \in C$. Consequently,

$$\left| \bigcup_{c \in C} B(c, e) \right| = \sum_{c \in C} |B(c, e)| = |C| \cdot |B(0^n, e)|$$

where the last equality follows from the fact that the volume of a ball doesn't depend on the location of its center. Now, an elementary counting argument reveals that

$$|B(0^n, e)| = \sum_{i=0}^{e} \binom{n}{i} (q-1)^i$$

On the other hand, we also know that

$$\bigcup_{c \in C} B(c, e) \subseteq \Sigma^n \implies \left| \bigcup_{c \in C} B(c, e) \right| \leq |\Sigma^n| = q^n$$

Thus

$$|C| \cdot |B(0^n, e)| = q^k \cdot \sum_{i=0}^{e} \binom{n}{i} (q-1)^i \leq q^n$$

simplifying which yields the desired result. ∎

Definition 2.14 (Perfect Codes)**.** Codes that satisfy the Hamming Bound are known as perfect codes.

§3. Linear Codes

Definition 3.1 (Linear Codes)**.** Let q be a prime power, and let $\Sigma = \mathbb{F}_q$. Then $C \subseteq \mathbb{F}_q^n$ is a linear code if it is a subspace of $\mathbb{F}_q^n$.
A linear code $C \subseteq \mathbb{F}_q^n$ with dimension k and distance d is denoted as $[n, k, d]_q$, or simply $[n, k]_q$.

Remark. Note the subtle difference in notation between $[n, k, d]_q$ and $(n, k, d)_q$.

Linearity simplifies many aspects of general codes. For example,

Lemma 3.1 (Distance Equals Minimum Hamming Weight)**.** Let C be a $[n, k, d]_q$ code. Then

$$d = \min_{\substack{c \in C \\ c \neq 0}} \mathrm{wt}(c)$$

Proof. Firstly, note that $\min_{\substack{c \in C \\ c \neq 0}} \mathrm{wt}(c) = \min_{\substack{c \in C \\ c \neq 0}} \Delta(0, c) \geq \min_{\substack{c, c' \in C \\ c \neq c'}} \Delta(c, c') = d$.
Now, since d is the distance of C, there exist $c_1, c_2 \in C$ such that $\Delta(c_1, c_2) = d$. However, since C is also linear, $c_1 - c_2 \in C$, and consequently, $\min_{\substack{c \in C \\ c \neq 0}} \mathrm{wt}(c) \leq \mathrm{wt}(c_1 - c_2) = \Delta(c_1, c_2) = d$.
Combining the two inequalities yields the desired lemma. ■

We now elucidate the linear algebraic structure of linear codes more explicitly, in the form of the definitions below.

Definition 3.2 (Generator Matrices)**.** Let C be a $[n, k]_q$ code. Then there exists a full-rank matrix $G \in \mathbb{F}_q^{n \times k}$, also known as the *generator matrix of* C, such that

$$C = \{Gx : x \in \mathbb{F}_q^k\}$$

Remark. Note that the columns of G form a basis for C. This also tells us that G is not unique in general.

Definition 3.3 (Parity Check Matrices)**.** Let C be a $[n, k]_q$ code. Then there exists a full-rank matrix $H \in \mathbb{F}_q^{(n-k) \times n}$, also known as the *parity-check matrix of* C, such that

$$C = \{y \in \mathbb{F}_q^n : Hy = 0\}$$

Remark. C is the null space of H. Furthermore, note that $HG = O$.

Before we move on to the applications of generator and parity check matrices, we explore another powerful consequence of the linearity of codes.

Definition 3.4 (Systematic code)**.** A (linear) code generated by a matrix $G \in \mathbb{F}_q^{n\times k}$ is called systematic if G is of the form $\left[\frac{I_k}{A}\right]$ for some $A \in \mathbb{F}_q^{(n-k)\times k}$.

Note that if C is a systematic code, then $C(m) = Cm = \left[\frac{m}{m'}\right]$, ie:- the encoding of every message m contains m as a prefix. Thus decoding systematic codes is as easy as it gets.

Theorem 3.2. Let $C = [n,k]_q$ be any linear code. Then there exists a systematic (linear) code C' such that C and C' are in a linear bijection.

Proof. Let G be any matrix generating C. Since the rank of G is k, the rank of G^T is k too. Moreover, the reduced-row echelon form of G^T is of the form $A' := [I_k|A]$ for some $A \in \mathbb{F}_q^{k\times(n-k)}$. Now, we know that the reduced row-echelon form of G^T can be obtained by performing row operations on it. We also know that performing a row operation on a matrix is equivalent to pre-multiplying the matrix by some elementary matrix corresponding to the row operation. Consequently, $A' = EG^\mathsf{T}$, where E is the product of elementary matrices, and is thus invertible. Thus

$$A' = EG^\mathsf{T} \implies G^\mathsf{T} = E^{-1}A' \implies GE^{-1} = (A')^\mathsf{T}$$

Since $(A')^\mathsf{T}$ generates a systematic code, there exists a linear bijection between the code generated by G and a systematic code. ∎

Remark. Note that linear bijections don't necessarily preserve the distance of a code.

3.1. Applications of the Generator and Parity Check Matrices

One immediate consequence of the above definitions is

Lemma 3.3. Any $[n,k]_q$ code can be represented with $\mathcal{O}(n\cdot\min(k,n-k))$ symbols.

Proof. Depending on whether k is lesser (or greater than) $n-k$, once can *represent* a $[n,k]_q$ code by its generator or parity check matrix respectively. ∎

Now, note that if C is a $[n,k]_q$ code, then the encoding of C can be simply defined as $C(m) := Gm$, where $m \in \mathcal{M} = \mathbb{F}_q^k$. Consequently,

Lemma 3.4. Encoding for a $[n,k]_q$ code can be performed in $\mathcal{O}(nk)$ time.

Proof. It takes $\mathcal{O}(nk)$ time to multiply a $n\times k$ matrix with a $k\times 1$ vector. ∎

Lemma 3.5. For a $[n,k]_q$ code, error detection can be performed $\mathcal{O}(n(n-k))$ time.

Proof. Note that a particular $y \in \mathbb{F}_q^n$ belongs to C if and only if $Hy = 0$. Since Hy can be calculated in $\mathcal{O}(n(n-k))$ time, the lemma follows. ■

Finally, we can obtain another identity for d in terms of the parity check matrix H.

Lemma 3.6. Let C be a $[n, k, d]_q$ code, and let H be a parity-check matrix for C. Then $\text{spark}(H) = d$.

Proof. By Lemma 3.1, there must exist a $c \in C$ such that $\text{wt}(c) = d$. Now, by the definition of H, we have $Hc = 0$, ie:- $\sum_{j=1}^{n} H_{i,j}c_j = 0$ for every $i \in [n-k]$. Now since $\text{wt}(c) = d$, c has only d non-zero entries in it, say $c_{j_1}, \ldots, c_{j_d}$, which in turn implies that the set of columns $H_{\cdot,j_1}, \ldots, H_{\cdot,j_d}$ are linearly dependent, since their linear combination by the non-zero entries of c leads to them becoming zero. Thus the spark of H is at most d.
On the other hand, let $H_{\cdot,\ell_1}, \ldots, H_{\cdot,\ell_t}$ be a set of linearly dependent columns with non-zero weights $c'_{\ell_1}, \ldots, c'_{\ell_t}$ making them 0, ie:- $\sum_{i=1}^{t} H_{\cdot,\ell_i}c_{\ell_i} = 0$. Then consider $c' \in \mathbb{F}_q^n$ where the ℓ_i^{th} entries of c' are just c'_{ℓ_i}, while the other entries are 0. Then $Hc' = 0$, which implies that $c' \in C$, which further implies that the spark of H is atleast d.
Joining the two inequalities above yields that the spark of H is d. ■

Corollary 3.7 (Singleton Bound)**.** For any linear code $[n, k, d]_q$, we have $d \leq n - k + 1$.

Remark. The Singleton bound holds for *all* codes. We'll see the general proof later.

Proof. $d = \text{spark}(H) \leq \text{rank}(H) + 1 = n - k + 1$. ■

3.2. Hamming Codes

We are now in a position to describe a very important class of codes called Hamming Codes.

Definition 3.5. For any $r \in \mathbb{N}$, consider the matrix $H_r \in \mathbb{F}_2^{r \times (2^r - 1)}$, where the i^{th} column of H_r is defined to be the r-bit binary representation of i, for $i \in [2^r - 1]$.
The $[2^r - 1, 2^r - r - 1]_2$ Hamming code, denoted by $C_{H,r}$ is the code with parity check matrix H_r.

It turns out that all Hamming codes have a distance of 3.

Lemma 3.8. The distance of any Hamming code is 3.

Proof. Note that any two columns of H_r are linearly independent: Indeed, in $\mathbb{F}_2^n$, the *only* pair of linearly dependent vectors are $(0^n, 1^n)$, and by the definition of H_r, 0^r doesn't occur in H_r. Consequently, the spark of H_r is at least 3.
On the other hand, note that

$$H_r^1 + H_r^2 + H_r^3 = \begin{bmatrix} \vdots \\ 0 \\ 1 \end{bmatrix} + \begin{bmatrix} \vdots \\ 1 \\ 0 \end{bmatrix} + \begin{bmatrix} \vdots \\ 1 \\ 1 \end{bmatrix} = 0$$

which shows that the spark of H_r is 3, which implies that $d = 3$. ■

Corollary 3.9. Hamming Codes can correct up to 1 error.

Finally, one of the reasons why Hamming codes are so interesting, from a theoretical point of view, is because they are perfect codes.

Lemma 3.10. Hamming codes are perfect.

Proof. Note that Hamming codes have $d = 3$, and $q = 2$, which means that

$$\sum_{i=1}^{\lfloor \frac{d-1}{2} \rfloor} \binom{n}{i}(q-1)^i = \binom{2^r}{1} = 2^r$$

Consequently, $\log_q \left(\sum_{i=1}^{\lfloor \frac{d-1}{2} \rfloor} \binom{n}{i}(q-1)^i \right) = \log_2(2^r) = r$, and indeed $k = n-r-1$, showing that equality is achieved in the Hamming bound for Hamming codes. ■

Remark. A few remarks are in order:

1. The only perfect codes (non-linear codes included) with $q = 2$ are:
   (a) Hamming Codes
   (b) $C = \{0^n, 1^n\} \subseteq \{0,1\}^n$ is a (trivial) perfect code with specifications $[n, 1, n]_2$.
   (c) Golay Codes
2. Note that for every $v \in \{0,1\}^{2^r-1}$, which is *not* a Hamming codeword, there is a unique Hamming codeword at a distance of 1 to v.

To explicitly give an example of a Hamming code, consider the $[7, 4, 3]_2$ Hamming code, which is described as follows:

$$C_{H,3}(x_1, x_2, x_3, x_4) := (x_1, x_2, x_3, x_4, x_2 \oplus x_3 \oplus x_4, x_1 \oplus x_3 \oplus x_4, x_1 \oplus x_2 \oplus x_4)$$

Its generator matrix can just be read off by substituting the basis vectors $e_1, \ldots, e_4$ into $C_{H,3}$ to get

$$G = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \end{bmatrix}$$

The parity check matrix, of course, can be just written from the definition of $H_{\cdot,r}$:

$$H_{3,r} = \begin{bmatrix} 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 0 & 1 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 1 & 0 & 1 & 0 & 1 \end{bmatrix}$$

The special structure of Hamming codes also allows for the easy decoding of Hamming codes.

Theorem 3.11. Consider a $C = [n, k, d]_q = [2^r - 1, 2^r - r - 1, 3]_2$ Hamming code. Then there exists a decoder, which given a $y \in \mathbb{F}_2^n$,

1. Certifies y as a valid codeword if $y \in C$.
2. Detects the error, and gives its location if $\min_{c \in C} \Delta(y, c) = 1$.

Furthermore, this decoder works in $\mathcal{O}(n \log n)$ time.

Proof. Note that performing the matrix calculation $H_r y$ takes $\mathcal{O}(rn) = \mathcal{O}(n \log n)$ time.
Observe that $y \in C$ if and only if $H_r y = 0$. We can check if $H_r y = 0$ in $\mathcal{O}(n \log n)$ time.
Now suppose $\min_{c \in C} \Delta(y, c) = 1$: Then $y = c + e_i$ for some $i \in [n]$, where e_i is the unit vector with a 1 at the i^{th} position. Then $H_r y = H_r^{(i)}$, where $H_r^{(i)}$ is the i^{th} column of H. Consequently, if $\min_{c \in C} \Delta(y, c) = 1$, then we can detect and locate the error in $\mathcal{O}(n \log n)$ time. ■

3.2.1. *The Hat Problem

This section is optional and can be skipped in the first reading.
Hamming codes arise very naturally as a solution to the following popular puzzle:

> There are n prisoners in a room, each of whom is given a black/white hat, uniformly at random. Every prisoner can see the hat of every prisoner but his own.
> The jailor asks each prisoner the color of his hat. A prisoner can either guess or abstain from answering. The group of n prisoners wins collectively if they make at least one guess, and every guess they make is correct.
> Thus, the prisoners lose if they all abstain, or if any one of them makes a wrong guess.
> The prisoners are not allowed to talk to each other once the jailor has started asking them questions, although they can strategize beforehand.

We ask the usual questions: What is the maximum possible value of the probability that the prisoners win? What is a strategy that will maximize the probability of the prisoners winning?
We shall answer these questions, and see a surprising connection with Hamming codes along the way.

Lemma 3.12. Consider the undirected hypercube graph

$$G = (V, E) := (\{0,1\}^n, \{\{x, y\}, x, y \in \{0,1\}^n : \Delta(x, y) = 1\})$$

Let H be a subgraph of G. Now, to every edge of H, we assign a direction, thus turning H into a directed graph. We define $K(H)$ to be the number of vertices in H with an in-degree of at least 1, and an out-degree of 0.
Then the maximum probability of winning the hat game is

$$\max_{H \text{ is a directed subgraph of } G} \frac{K(H)}{2^n}$$

Proof. Suppose $n = 5$, and the third prisoner sees the colors 0, 1, 1, 1 (where 0 stands for white and 1 stands for black). Consequently, for the third prisoner, the hat color configuration is $(0, 1, ?, 1, 1)$.
Now suppose our subgraph 'H' contains an edge between $(0, 1, 0, 1, 1)$ and $(0, 1, 1, 1, 1)$. If the edge is towards the tuple with $? = 1$, we interpret that as the third prisoner's strategy being guessing 1. Similarly, if $? = 0$, the third prisoner guesses 0. Finally, if the edge is absent, then the prisoner abstains.
As described in the example above, it is easy to see how every directed subgraph of G encodes a strategy: Every edge present in our subgraph encodes a choice, and those edges of G which aren't present encode abstinence. Conversely,

every possible strategy is encoded by some directed subgraph of G.
Now, note that if the prisoners play by a strategy in which two prisoners make guesses which correspond to different configurations, then that strategy is bound to fail since at least one of the prisoners will be wrong.
Consequently, if the prisoners play by the strategy encoded by H, then the vertices which have an in-degree of ≥ 1, as well as an out-degree of ≥ 1, correspond to fatal configurations: If the prisoners make guesses according to that configuration, then they will lose. Stated alternatively, the only configurations which on being chosen could lead to victory are the configurations that have an in-degree of ≥ 1 and out-degree of 0 [3].
Since each configuration occurs with probability $\frac{1}{2^n}$, the lemma follows. ■

Lemma 3.13. The probability of victory in the hat game is at most $\frac{n}{n+1}$.

Proof. Let $H = (V_H, E_H)$ be a directed subgraph of G. A vertex of H with 0 out-degree and ≥ 1 in-degree is called nice. Let $N \subseteq V_H$ be the set of nice vertices of H, and let $U = \{0,1\}^n \setminus N$ be the set of vertices which are not nice. Then

$$|N| + |U| = 2^n$$

Note that $|N| = K(H)$.
At the same time, $|N| \leq n \cdot |U|$, since the out-degree of any vertex in G is at most n. Thus $2^n \geq |N| + \frac{|N|}{n} \implies \frac{|N|}{2^n} \leq \frac{n}{n+1}$, as desired. ■

Theorem 3.14. Let $r \in \mathbb{N}$, and $n = 2^r - 1$. Then the bound proved in Lemma 3.13 can be achieved through the use of Hamming codes.

Proof. Let $C \subseteq \{0,1\}^n$ be the Hamming code. Let $H = G$, and we assign directions to the edges of G as follows: Draw outward edges from every codeword in C to its neighbors. No other edges are drawn.
Now, for this graph,

$$K(H) = |\{0,1\}^n| - |C| = 2^{2^r-1} - 2^{2^r-r-1}$$

Thus

$$\frac{K(H)}{2^n} = \frac{2^{2^r-1} - 2^{2^r-r-1}}{2^{2^r-1}} = 1 - 2^{-r} = \frac{n}{n+1}$$

■

3.3. Dual of a Linear Code

Definition 3.6. Let C be a code, and let H be a parity-check matrix for C. Then the code generated by H^T is known as the dual of C, which is denoted as $C^\perp$.

Remark. If C is a $[n,k]_q$ code, then $C^\perp$ is a $[n, n-k]_q$ code. Furthermore, if G, H are the generator and parity-check matrices of C respectively, then $H^\mathsf{T}, G^\mathsf{T}$ are generator and parity-check matrices of $C^\perp$ respectively.

We digress a bit to point out a slightly counterintuitive fact: Note that if C is a subspace of a vector space over $\mathbb{R}$, then $C \cap C^\perp = \{0\}$ since no non-zero vector of real numbers is orthogonal to itself. However, this is not the case for finite fields (whose characteristic is positive): There exist non-zero self-orthogonal vectors over finite fields, and for subspaces C of vector spaces over some finite field, $C \cap C^\perp$ can contain non-zero vectors too.

[3] the configurations which have an out-degree ≥ 1 and in-degree 0 won't be chosen in the first place

Definition 3.7. A linear code C is called:

1. Self-orthogonal if $C \subseteq C^{\perp}$.
2. Self-dual if $C = C^{\perp}$.

Remark. We shall see some examples, later on, of self-dual codes.
We use the concept of duals to define two important classes of codes below.

Definition 3.8. (Simplex Code) For every $r \in \mathbb{N}$, the dual of the Hamming code $C_{H,r}$ is said to be the *simplex* code, $C_{\text{Sim},r}$, ie:- $C_{\text{Sim},r} := C_{H,r}^{\perp}$. Consequently, $C_{\text{Sim},r}$ is a $[2^r - 1, r]_2$ code.

Definition 3.9. (Hadamard Code) For any $r \in \mathbb{N}$, take the $r \times (2^r - 1)$ matrix H_r, and append the all-zero column to it to make a new matrix $H_r' \in \mathbb{F}_2^{r \times 2^r}$. The code generated by $(H_r')^{\mathsf{T}}$ is called the *Hadamard* code, which is denoted as $C_{\text{Had},r}$, and it is a $[2^r, r]_2$ code.

We'll now state a very interesting property of Hadamard codes.

Theorem 3.15. The weight of every non-zero codeword in $C_{\text{Had},r}$ is 2^{r-1}.

Proof. Fix some $i \in [r]$.
Partition all columns of H_r' into pairs, where two columns u and v are paired together if $u = v + e_i$. Since we are over $\mathbb{F}_2$, if $u = v + e_i$, then $v = u + e_i$, and thus the pairing scheme is symmetric.
Now, for any $x \in \mathbb{F}_2^r$ such that $x_i = 1$, note that

$$\langle x, u \rangle = \langle x, v + e_i \rangle = \langle x, v \rangle + x_i = \langle x, v \rangle + 1$$

Thus exactly one number among $\langle x, u \rangle, \langle x, v \rangle$ is 1. Consequently, exactly 2^{r-1} columns of H_r' have a dot product of 1 with x.
Now, the Hadamard code encodes x as $c_x := (H_r')^{\mathsf{T}} x$. Note that the weight of c_x is the number of rows of $(H_r')^{\mathsf{T}}$ (alternatively viewed as columns of H_r') which have a dot product of 1 with x. Assuming x is non-zero, there is some $i \in [r]$ such that $x_i = 1$, and then by the discussion above, we get that the weight of c_x is 2^{r-1}. Since every non-zero Hadamard codeword is generated by some non-zero vector in $\mathbb{F}_2^r$, the claim follows. ∎

Corollary 3.16. The distance of $C_{\text{Sim},r}$ is 2^{r-1}, the simplex code is $[2^r - 1, r, 2^{r-1}]_2$.

Corollary 3.17. The distance of $C_{\text{Had},r}$ is 2^{r-1}, the Hadamard code is $[2^r, r, 2^{r-1}]_2$.

We also define the *augmented Hadamard code*, which is a variant of the Hadamard code with a slightly better rate.

Definition 3.10. Let $r \geq 2$. For $x \in \mathbb{F}_2^r$, we define the augmented Hadamard encoding of x to be

$$\text{augHad}(x) := (\langle x, y \rangle)_{y \in \{1\} \times \{0,1\}^{r-1}} \in \mathbb{F}_2^{2^{r-1}}$$

The augmented Hadamard code is a $[2^{r-1}, r, 2^{r-2}]_2$ code.

The reader can refer to [Had23] for more on the augmented Hadamard code.

§4. Bounds on Codes

Before we begin this section, we introduce the concept of relative distance 'δ' of a $(n, k, d)_q$ code, which is defined as d/n. The notion of δ homogenizes the notion of distance and makes the comparison of different codes possible.

Definition 4.1. Given a $(n, k, d)_q$ code, we define its relative distance δ to be $\frac{d}{n}$.

Remark. The above definition has been given for *all* codes, not just linear codes.

After proving many of the bounds below, it is often fruitful to look for codes that form the equality case of the bound. This is because the codes satisfying these equality conditions often possess very desirable "extremal properties", which makes them useful candidates for real-life applications.

4.1. Asymptotic Hamming Bound

Just the notion of relative distance allows us to state and prove an inequality right away.

Theorem 4.1 (Asymptotic Hamming Bound)**.** Consider a *family* of codes $\{(n_i, k_i, d_i)_{q_i}\}_{i\in\mathbb{N}}$, with rates $R_i = k_i/n_i$ and relative distances $\delta_i = d_i/n_i$. Let $R = \lim_{i\to\infty} R_i$, and $\delta = \lim_{i\to\infty} \delta_i$. Then

$$R \leq 1 - H_q\left(\frac{\delta}{2}\right)$$

Proof. By Theorem A.10, we know that $\operatorname{vol}_{q_i,n_i}\left(\left\lfloor\frac{d_i-1}{2}\right\rfloor\right) \geq q_i^{H_{q_i}(\frac{\delta_i}{2})n_i - o(n_i)}$, and thus

$$\log_{q_i}\operatorname{vol}_{q_i,n_i}\left(\left\lfloor\frac{d_i-1}{2}\right\rfloor\right) \geq H_{q_i}\left(\frac{\delta_i}{2}\right)n_i - o(n_i) \implies \frac{1}{n_i}\log_{q_i}\operatorname{vol}_{q_i,n_i}\left(\left\lfloor\frac{d_i-1}{2}\right\rfloor\right) \geq H_{q_i}\left(\frac{\delta_i}{2}\right) - o(1)$$

On the other hand, from Theorem 2.2, we know that

$$k_i \leq n_i - \log_{q_i}\operatorname{vol}_{q_i,n_i}\left(\left\lfloor\frac{d_i-1}{2}\right\rfloor\right) \implies R_i \leq 1 - \frac{1}{n_i}\log_{q_i}\operatorname{vol}_{q_i,n_i}\left(\left\lfloor\frac{d_i-1}{2}\right\rfloor\right)$$

Combining the two inequalities above yields

$$R_i \leq 1 - H_{q_i}\left(\frac{\delta_i}{2}\right) + o(1)$$

Finally, passing into the limit yields the desired result. ∎

Remark. It is a good mathematical rule of thumb that wherever we see the entropy function $H(\cdot)$, the volume of the Hamming ball must have been involved somewhere in the proof.

4.2. Gilbert-Varshamov Bound

We now establish one of the most important bounds in coding theory, namely the Gilbert-Varshamov bound. The reason why this bound is so important is that it gives a lower bound for R in terms of δ: such inequalities are rare. Also, from now onwards, we shall directly work with codes, *assuming that they belong to some family, which justifies things like taking limits or talking in terms of asymptotic notation*, without explicitly mentioning so every time.

Lemma 4.2. For any $n, q \in \mathbb{N}$, and any $\delta \in \left[\frac{1}{n}, 1 - \frac{1}{q}\right)$, there exists a $(n, k, \geq \delta n)_q$ code satisfying $R \geq 1 - H_q\left(\delta - \frac{1}{n}\right)$.

Proof. We present a greedy algorithm for constructing a code with the desired properties.
Note that the code outputted by this algorithm indeed has a distance $\geq d$: Indeed, a codeword is added to C if

Algorithm 1: Gilbert Varshamov Construction

Data: $n, q, d = \delta n$
Result: $C = (n, k, \geq d)_q$
$C \leftarrow \varnothing$;
while $\exists v \in [q]^n, \min_{c \in C} \Delta(v, c) \geq d$ **do**
  Add v to C
return C

and only if it doesn't cause the distance of the code to fall below d, and when no such codeword can be found the algorithm terminates.
Now, note that $\bigcup_{c \in C} B_{q,n}(c, d-1) = [q]^n$, where C is the code returned by the algorithm: Indeed, if there existed $x \in [q]^n \setminus \bigcup_{c \in C} B_{q,n}(c, d-1)$, then $\Delta(x, c) \geq d$ for every $c \in C$, which is a contradiction since our algorithm would have found at least one candidate $x \in [q]^n$ for which $\min_{c \in C} \Delta(x, c) \geq d$, and wouldn't have terminated, returning C.
Thus

$$\left|\bigcup_{c \in C} B_{q,n}(c, d-1)\right| = q^n \implies \sum_{c \in C} |B_{q,n}(c, d-1)| \geq q^n \implies |C| \cdot \mathrm{vol}_{q,n}(d-1) \geq q^n$$

By Theorem A.10, $\mathrm{vol}_{q,n}(d-1) \leq q^{nH_q(\delta - \frac{1}{n})}$, and thus

$$|C| \geq \frac{q^n}{q^{nH_q(\delta - \frac{1}{n})}} = q^{n(1 - H_q(\delta - \frac{1}{n}))} \implies k \geq n\left(1 - H_q\left(\delta - \frac{1}{n}\right)\right) \implies R \geq 1 - H_q\left(\delta - \frac{1}{n}\right)$$

■

Remark. The $\delta < 1 - \frac{1}{q}$ condition is needed for Theorem A.10 to hold.

Corollary 4.3. For any $\delta \in \left(0, 1 - \frac{1}{q}\right)$, there exists a family of codes with asymptotic relative distance $\geq \delta$ and asymptotic rate satisfying $R \geq 1 - H_q(\delta)$.

However, there are multiple problems with the above construction: First of all, the code produced by the greedy algorithm may not be linear, in which case even storing the code can take an exponential amount of space. Secondly, it can be shown that the greedy algorithm outlined above takes $q^{\mathcal{O}(n)}$ time to run, which implies that even finding a code with the above properties can take exponential time.
Fortunately, a randomized construction saves the day.

Theorem 4.4. With high probability, one can find (a family of) linear codes satisfying the bounds in Corollary 4.3. A bit more precisely, let $\delta \in \left[\frac{1}{n}, 1 - \frac{1}{q}\right), \varepsilon \in (0, 1 - H_q(\delta)], n \in \mathbb{N}$ be any numbers. Then a uniformly random matrix in $\mathbb{F}_q^{n \times k}$, where $k = n(1 - H_q(\delta) - \varepsilon)$, generates a $[n, k, \geq \delta n]_q$ code with probability $\geq 1 - q^{-\varepsilon n}$.

Proof. We must show that a uniformly random matrix in $\mathbb{F}_q^{n\times k}$ is full rank [4]. Furthermore, by Lemma 3.1, it is enough to show that the following holds (with high probability) to establish that the distance is $\geq \delta n$:

$$\mathrm{wt}(Gx) \geq \delta n, \forall x \in \mathbb{F}_q^k \setminus \{0^k\}$$

Now, let G be a uniformly random matrix in $\mathbb{F}_q^{n\times k}$. Then, by Lemma A.2, Gx is a uniform random vector in $\mathbb{F}_q^n$. Consequently, for any $x \in \mathbb{F}_q^k \setminus \{0^k\}$

$$\Pr(\mathrm{wt}(Gx) < d) = \frac{\mathrm{vol}_{q,n}(d-1)}{q^n} \leq q^{-n\left(1-H_q\left(\delta-\frac{1}{n}\right)\right)}$$

where the last inequality is a familiar invocation of Theorem A.10.
Since $H_q\left(\delta - \frac{1}{n}\right) \leq H_q(\delta)$[5],

$$\Pr(\mathrm{wt}(Gx) < d) < q^{-n\left(1-H_q(\delta)\right)} = q^{-k}q^{-\varepsilon n}$$

Consequently, by Lemma A.1,

$$\Pr(\mathrm{wt}(Gx) \geq \delta n, \forall x \in \mathbb{F}_q^k \setminus \{0^k\}) \geq 1 - (q^k - 1)(q^{-k}q^{-\varepsilon n}) > 1 - q^{-\varepsilon n}$$

Suppose G is not full rank. Then there would exist $x \in \mathbb{F}_q^k \setminus \{0^k\}$ such that $Gx = 0 \implies \mathrm{wt}(Gx) = 0$.
Consequently, the event "$\mathrm{wt}(Gx) \geq \delta n, \forall x \in \mathbb{F}_q^k \setminus \{0^k\}$" implies that G is full-rank, and the lemma follows. ■

4.3. Singleton Bound

The Singleton bound is named after its discoverer, R. Singleton. Thus the "Singleton" in the Singleton bound refers to a person, not the set-theoretic term for a set with one element. That's why we spell this 'Singleton' with an upper case 'S'.
Note that we have already seen the Singleton bound for linear codes in Corollary 3.7. We shall prove it for all codes now.

Theorem 4.5. For any $(n, k, d)_q$ code, we have $d \leq n - k + 1$.

Proof. Let $c_1, c_2, \ldots, c_{q^k}$ be the codewords in our code. For any codeword c_i, let c_i' be the prefix of c_i of length $n-d+1$. Then note that if $i \neq j$, then $c_i' \neq c_j'$: Indeed, if $c_i' = c_j'$, then $\Delta(c_i, c_j) = \Delta(c_i'', c_j'')$ where $c_i = c_i'||c_i''$, $c_j = c_j'||c_j''$. But since the length of c_i'' is $d-1$, $\Delta(c_i'', c_j'') \leq d-1$, contradicting the fact that the distance of our code is d.
Consequently, the number of codewords in our code can be at most the number of distinct strings of length $n-d+1$ that can be formed from an alphabet of size q.
Casting the above reasoning in mathematical notation yields

$$|C| = q^k \leq q^{n-d+1} \implies k \leq n - d + 1$$

as desired. ■

Remark. Although we took a prefix of length $n-d+1$ in this proof, note that we could have taken *any* subset of the indices with size $n-d+1$ and our proof would have gone through.

Corollary 4.6. Consider a family of codes with asymptotic rate R and asymptotic relative distance δ. Then $R \leq 1-\delta$.

Proof. By the Singleton bound, $R_i \leq 1 - \delta_i + \frac{1}{n_i}$. Taking the limit yields the desired result. ■

[4] otherwise the dimension of the code it generates will not be k

[5] recall that $H_q(\cdot)$ is an increasing function on $\left[0, 1 - \frac{1}{q}\right)$

4.3.1. MDS codes

Definition 4.2. Codes that satisfy the Singleton bound are known as Maximum Distance Separable (MDS) codes.

Remark. Note that:-

1. MDS codes do exist (so we aren't talking about unicorns here)! As we shall see later, Reed-Solomon codes are a very important class of MDS codes, although they are not the only ones.
2. In case k is not an integer, satisfying the Singleton bound entails d being the largest integer smaller than $n-k+1$.

Definition 4.3. Given a code $C \subseteq \Sigma^n$, and given a subset $S = \{s_1, s_2, \ldots, s_m\} \subseteq [n]$, we define the projection of C onto S to be

$$C_S := \{c_S = (c_{s_1}, c_{s_2}, \ldots, c_{s_m}) : c = (c_1, c_2, \ldots, c_n) \in C\}$$

Theorem 4.7. Let C be a $(n,k)_q$ MDS code. Then for any $S \subseteq [n]$ with $|S| = k$, we have $|C_S| = q^k$.

Proof. As remarked after the proof of the Singleton bound, for any distinct $\alpha, \beta \in C$, $\alpha_S \neq \beta_S$, where S is any subset of n with size k. Consequently, if $|S| = k$, the projection $C \to C_S$ is actually a bijection. The desired result then follows. ■

Theorem 4.8. The dual of a linear MDS code is also a linear MDS code.

Proof. Consider the linear MDS code $C := [n, k, n-k+1]_q$, generated by the matrix G. Its dual is $C^\perp := [n, n-k]_q$. We must show that the distance of $C^\perp$ is $k+1$. Now, by Lemma 3.6, the distance of $C^\perp$ is the spark of G^T.
Now, we claim that every set of k rows of G is linearly independent. Assume for the sake of contradiction that some k rows of G, given by $G_{t_1,\cdot}, G_{t_2,\cdot}, \ldots, G_{t_k,\cdot}$ are linearly dependent. Then if we concatenate these n rows, we get a $k \times k$ matrix, whose rows are linearly dependent. Consequently, the columns of this matrix are also linearly dependent, which implies that the space spanned by these k columns is of dimension lesser than k, and consequently, the cardinality of the space spanned by these k columns is lesser than q^k, which contradicts Theorem 4.7.
Since any k rows of G are linearly independent, the spark of G^T is atleast $k+1$. Now, by Theorem 4.5, the distance of $C^\perp$ is at most $n-(n-k)+1 = k+1$. Consequently, the distance of $C^\perp$ is exactly $k+1$, as desired. ■

4.4. Plotkin Bound

Plotkin's bound has 3 régimes: δ greater than, lesser than, or equal to, $1 - \frac{1}{q}$. We shall see each of them one by one.

Theorem 4.9. For *any* $C = (n,k,d)_q$ code, if $\delta = \frac{d}{n} > 1 - \frac{1}{q}$, then $|C| = q^k \leq \frac{\delta}{\delta - (1-\frac{1}{q})}$.

Proof. Define

$$S := \sum_{\substack{c_1, c_2 \in C \\ c_1 \neq c_2}} \Delta(c_1, c_2)$$

Clearly $S \geq \binom{|C|}{2} d = \frac{1}{2}|C|(|C|-1)d$.
Also define the matrix $\mathcal{C} \in \mathbb{F}_q^{|C| \times n}$, where each row of $\mathcal{C}$ is a codeword in C. Now, note that

$$S = \sum_{j=1}^{n} \sum_{1 \leq i_1 < i_2 \leq |C|} \mathbb{1}_{\mathcal{C}_{i_1 j} \neq \mathcal{C}_{i_2 j}}$$

Now, fix any j, and let $\Sigma = \{\sigma_1, \ldots, \sigma_q\}$. Suppose in the j^{th} column of $\mathcal{C}$, $t_{\ell,j}$ entries equal σ_ℓ for $\ell \in [q]$. Then

$$\sum_{1 \leq i_1 < i_2 \leq |C|} \mathbb{1}_{\mathcal{C}_{i_1 j} \neq \mathcal{C}_{i_2 j}} = \binom{|C|}{2} - \sum_{\ell=1}^{q} \binom{t_{\ell,j}}{2}$$

Now, since $x \mapsto \frac{x(x-1)}{2}$ is a convex function, by Jensen's inequality we have

$$\sum_{\ell=1}^{q} \binom{t_{\ell,j}}{2} \geq q \binom{\frac{1}{q} \cdot \sum_{\ell=1}^{q} t_{\ell,j}}{2} = q \binom{\frac{|C|}{q}}{2} = \frac{|C|(|C|-q)}{2q}$$

$$\implies \sum_{1 \leq i_1 < i_2 \leq |C|} \mathbb{1}_{\mathcal{C}_{i_1 j} \neq \mathcal{C}_{i_2 j}} \leq \binom{|C|}{2} - \frac{|C|(|C|-q)}{2q} = \frac{|C|^2}{2}\left(1 - \frac{1}{q}\right)$$

Consequently

$$S = \sum_{j=1}^{n} \sum_{1 \leq i_1 < i_2 \leq |C|} \mathbb{1}_{\mathcal{C}_{i_1 j} \neq \mathcal{C}_{i_2 j}} \leq \sum_{j=1}^{n} \frac{|C|^2}{2}\left(1 - \frac{1}{q}\right) = \frac{n|C|^2}{2}\left(1 - \frac{1}{q}\right)$$

Thus

$$\frac{n|C|^2}{2}\left(1 - \frac{1}{q}\right) \geq \frac{1}{2}|C|(|C|-1)d$$

Simplifying this inequality yields the desired result. ■

Corollary 4.10. For some fixed $\delta > 1 - \frac{1}{q}$, $R = \frac{k}{n} \leq \frac{1}{n} \log_q \left(\frac{\delta}{\delta - (1 - \frac{1}{q})} \right) = o_{n \to \infty}(1)$.

Theorem 4.11. For *any* $C = (n, k, d)_q$ code, if $\delta = \frac{d}{n} \leq 1 - \frac{1}{q}$, then $R = \frac{k}{n} \leq 1 - \frac{q}{q-1}\delta + \frac{3 + \log_q(n)}{n} = 1 - \frac{q}{q-1}\delta + o(1)$.

Proof. Define $n' := \left\lfloor \frac{qd}{q-1} \right\rfloor - 1$. Now, for any $x \in \Sigma^{n-n'}$, define

$$\mathcal{C}_x := \{(c_{n-n'+1}, \ldots, c_n) : (c_1, \ldots, c_n) \in C, (c_1, \ldots, c_{n-n'}) = x\} \subseteq \Sigma^{n'}$$

Note that if $|\mathcal{C}_x| \geq 2$, then the distance of $\mathcal{C}_x$ is $\geq d$ [6]. We now claim that

$$|\mathcal{C}_x| \leq qd$$

Now, if $|\mathcal{C}_x| \leq 2$, then the inequality is satisfied trivially. Otherwise, $\mathcal{C}_x$ is a $(n', k', \geq d)_q$ code such that $n' < \frac{qd}{q-1} \implies \frac{d}{n'} \geq 1 - \frac{1}{q}$, and thus by Theorem 4.9 we have

$$|\mathcal{C}_x| \leq \frac{\frac{d}{n'}}{\frac{d}{n'} - (1 - \frac{1}{q})} = \frac{qd}{qd - (q-1)n'} \leq qd$$

[6] note that all codewords in C whose suffixes are in $\mathcal{C}_x$ have a distance of $\geq d$ among themselves by the very definition of d. Now, note that the distance between these codewords is entirely due to the distance between their suffixes since their prefixes are the same

as desired.
Now, note that

$$C = \bigsqcup_{x \in \Sigma^{n-n'}} \mathcal{C}_x \implies |C| \leq \sum_{x \in \Sigma^{n-n'}} |\mathcal{C}_x| \leq |\Sigma^{n-n'}| \cdot qd$$

$$\implies |C| \leq q^{n-n'+1}d < q^{n-\frac{q}{q-1}d+3}d \leq q^{n-\frac{q}{q-1}d+3}n = q^{n-\frac{q}{q-1}d+3+\log_q n} = q^{n-\frac{q}{q-1}d+o(n)} = q^{n(1-\frac{q}{q-1}\delta+o(1))}$$

Thus

$$q^k \leq q^{n(1-\frac{q}{q-1}\delta+o(1))} \implies k \leq n\left(1-\frac{q}{q-1}\delta+o(1)\right) \implies R \leq 1-\frac{q}{q-1}\delta+o(1)$$

■

Theorem 4.12. Let $\delta = \frac{d}{n} = 1 - \frac{1}{q}$. Then $|C| \leq 2qn$.

Proof. Let f be the map given by Lemma A.12. Then

$$\langle f(c_i), f(c_j)\rangle = 1 - \frac{q\Delta(c_i, c_j)}{n(q-1)} \leq 1 - \frac{qd}{n(q-1)} = 0, 1 \leq i < j \leq |C|$$

Thus $\{f(c_1), \ldots, f(c_{|C|})\}$ are a set of non-zero vectors in $\mathbb{R}^{nq}$ such that all of their mutual dot products are non-positive. Thus by Lemma A.11, we have that $|C| \leq 2qn$, as desired. ■

Corollary 4.13. For $\delta = 1 - \frac{1}{q}$, $R = \frac{k}{n} \leq \frac{1}{n}\log_q(2qn) = o_{n\to\infty}(1)$.

Corollary 4.14. Consider a family of codes with asymptotic relative distance $\delta \geq 1 - \frac{1}{q}$. Then $R = 0$.

Proof. Follows from Corollary 4.10, Theorem 4.11 and Corollary 4.13. ■

If $q = 2$, then Theorem 4.12 can be improved even further.

Theorem 4.15. Consider a binary code C of block length n and distance $\geq \frac{n}{2}$. Then $|C| \leq 2n$.

Proof. Define a map $f : \mathbb{F}_2^n \mapsto \mathbb{R}^n$ as

$$f(c) = f((c_1, \ldots, c_n)) := \frac{1}{\sqrt{n}}((-1)^{c_1}, \ldots, (-1)^{c_n})$$

Then for two different codewords c, d one can easily verify that

$$\langle f(c), f(d)\rangle = 1 - \frac{2\Delta(c,d)}{n} \leq 0$$

The result then follows by applying Lemma A.11 on the vectors produced by applying f on the codewords in C. Finally, note that this bound is tight, as is witnessed by augmented Hadamard codes (Definition 3.10). ■

4.5. Elias-Bassalygo Bound

This is a very powerful bound, that asymptotically supersedes most other bounds we have seen so far.
Before stating the Elias-Bassalygo bound, we prove some lemmata.

Lemma 4.16. Given any code $C \subseteq [q]^n$, and any $e \in \{0, \ldots, n\}$, there exists a Hamming ball of radius e containing $\geq \frac{|C| \operatorname{vol}_{q,n}(e)}{q^n}$ codewords in it.

Proof. Pick a $y \in [q]^n$ uniformly at random. Now note that

$$\mathbb{E}[|B(y,e) \cap C|] = \sum_{c \in C} \Pr(c \in B(y,e)) = \sum_{c \in C} \Pr(y \in B(c,e)) = \sum_{c \in C} \frac{\operatorname{vol}_{q,n}(e)}{q^n} = \frac{|C| \operatorname{vol}_{q,n}(e)}{q^n}$$

where the first equality follows by the linearity of expectation. Now, since the expectation of the random variable $|B(\cdot, e) \cap C|$ is $\frac{|C| \operatorname{vol}_{q,n}(e)}{q^n}$, there *must* exist some $y \in [q]^n$ for which $|B(y,e) \cap C| \geq \frac{|C| \operatorname{vol}_{q,n}(e)}{q^n}$ [7], as desired. ■

Definition 4.4. We define the q-ary Johnson function $J_q : \left[0, 1 - \frac{1}{q}\right] \mapsto \mathbb{R}$ as

$$J_q(x) := \left(1 - \frac{1}{q}\right)\left(1 - \sqrt{1 - \frac{qx}{q-1}}\right)$$

Lemma 4.17 (Johnson Bound)**.** Consider any binary code $C = (n, k, d)_2$, and let $e < J_2(\delta)n$, where $\delta = \frac{d}{n} \leq 1 - \frac{1}{2}$. Let $y \in \mathbb{F}_2^n$ be any arbitrary vector. Then $|B(y,e) \cap C| \leq 2dn$.

Remark. For general q-ary alphabets, we have $|B(y,e) \cap C| \leq qdn$, *mutatis mutandis*.

Proof. We mimic the proof of Theorem 4.9: Let $B(y,e) \cap C = \{c_1, c_2, \ldots, c_u\}$. We must show that $u \leq 2dn$. Define $c_i' = c_i - y$ for $i \in [u]$. Finally, define

$$S := \sum_{1 \leq i < j \leq u} \Delta(c_i', c_j') = \sum_{1 \leq i < j \leq u} \Delta(c_i, c_j)$$

Clearly $S \geq \binom{u}{2} d$.
Once again, construct the matrix $\mathcal{C} \in \mathbb{F}_2^{u \times n}$, where the i^{th} row of $\mathcal{C}$ is c_i. Now, note that

$$S = \sum_{j=1}^{n} \sum_{1 \leq i_1 < i_2 \leq u} \mathbb{1}_{\mathcal{C}_{i_1 j} \neq \mathcal{C}_{i_2 j}}$$

Now, fix any j, and suppose in the j^{th} column of $\mathcal{C}$, t_j entries equal 1. Then note that

$$S = \sum_{j=1}^{n} t_j(u - t_j)$$

[7] otherwise if $|B(y,e) \cap C| < \frac{|C| \operatorname{vol}_{q,n}(e)}{q^n}$ for every $y \in [q]^n$, then we'd have $\mathbb{E}[|B(y,e) \cap C|] < \frac{|C| \operatorname{vol}_{q,n}(e)}{q^n}$, which is a contradiction

Finally, define $\overline{e} := \frac{\sum_{j=1}^{n} t_j}{u}$. Then

$$S = u\sum_{j=1}^{n} t_j - \sum_{j=1}^{n} t_j^2 = u^2\overline{e} - \sum_{j=1}^{n} t_j^2 \leq u^2\overline{e} - \frac{(u\overline{e})^2}{n} = u^2\left(\overline{e} - \frac{\overline{e}^2}{n}\right)$$

Consequently,

$$u^2\left(\overline{e} - \frac{\overline{e}^2}{n}\right) \geq \binom{u}{2}d = \frac{u(u-1)d}{2} \implies u \leq \frac{1}{1 - \frac{2\overline{e}}{d}(1 - \frac{\overline{e}}{n})} = \frac{2dn}{(n-2\overline{e})^2 - n(n-2d)}$$

Finally, also note that $\sum_{j=1}^{n} t_j$ is the number of ones in $\mathcal{C}$, which is also the sum of weights of $c_i', i \in [u]$. Now, since $c_i \in B(y, e)$, $\mathrm{wt}(c_i') = \Delta(c_i, y) \leq e$. Consequently,

$$u\overline{e} = \sum_{j=1}^{n} t_j = \sum_{i=1}^{u} \mathrm{wt}(c_i') \leq ue \implies \overline{e} \leq e \implies \frac{2dn}{(n-2\overline{e})^2 - n(n-2d)} \leq \frac{2dn}{(n-2e)^2 - n(n-2d)}$$

$$\implies u \leq \frac{2dn}{(n-2e)^2 - n(n-2d)}$$

Now, since $e < J_2(\delta)n$, $n - 2e > n\left(1 - \sqrt{1 - 2\frac{d}{n}}\right) \implies (n-2e)^2 > n(n-2d)$. Since $(n-2e)^2, n(n-2d) \in \mathbb{Z}$, $(n-2e)^2 - n(n-2d) \geq 1 \implies u \leq 2dn$, as desired. ■

Theorem 4.18 (Elias-Bassalygo Bound)**.** Consider any code $C = (n, k, d)_q$, with $R = \frac{k}{n}, \delta = \frac{d}{n} \leq 1 - \frac{1}{q}$. Then $R \leq 1 - H_q(J_q(\delta)) + o(1)$.

Proof. Define $e = nJ_q(\delta) - 1$. By Lemma 4.16, there exists a Hamming ball $\mathcal{B}$ of radius e such that $|\mathcal{B}| \geq \frac{|C| \cdot \mathrm{vol}_{q,n}(e)}{q^n}$. Also, by Lemma 4.17, $|\mathcal{B}| \leq qdn$ [8].
Consequently

$$qdn \geq \frac{|C| \cdot \mathrm{vol}_{q,n}(e)}{q^n} \implies |C| \leq qdn \cdot \frac{q^n}{\mathrm{vol}_{q,n}(e)}$$

Invoking Theorem A.10 yields

$$q^k = |C| \leq q^{n(1 - H_q(J_q(\delta)) + o(1))}$$

as desired. ■

4.6. Summary

Throughout this chapter, we tried to explore the tradeoffs between the rate and relative distance, and we saw many bounds for the same. To recapitulate the central theme of this chapter,

> Consider a family of codes with asymptotic relative distance δ. What is the maximum possible asymptotic rate R^* that this family can achieve?

Although we have proved many bounds in this chapter, the best upper bounds for R in terms of δ are given by Theorem 4.18 and Corollary 4.14. The only lower bound we have is the Gilbert Varshamov bound. Combining these results yields:

[8] The Johnson Bound holds for general q-ary alphabets, although we only proved it for $q = 2$

Theorem 4.19. For $\delta < 1 - \frac{1}{q}$, the optimal value R^* is

$$R^* \geq 1 - H_q(\delta), \text{ Corollary 4.3}$$

$$R^* \leq 1 - H_q(J_q(\delta)), \text{ Theorem 4.18}$$

$$R^* \leq 1 - \frac{q\delta}{q-1}, \text{ Theorem 4.11}$$

For $\delta \geq 1 - \frac{1}{q}$, $R^* = 0$, by Corollary 4.14.

Remark. A few remarks are in order:

1. All other upper bounds proved in this chapter are weaker than the ones mentioned above: However, that doesn't mean that the effort that went into proving them was wasted. This is because bounds like the Singleton bound are very useful in a "finite setting" when we are not talking about asymptotic quantities, but rather trying to determine if some *particular* code is feasible or not. Moreover, the concept of MDS codes, which were defined as the extremal cases of the Singleton bound, is very useful and will be discussed later on too, in the context of Reed-Solomon codes. Finally, the crucial insight used in proving the Singleton bound, which is to observe that any set of $n - d + 1$ indices uniquely determines a codeword, forms the underpinning of many important constructions from codes, where the uniqueness of any set of indices translates into "independence" in a probabilistic setting.
2. It is worth noting that the Gilbert-Varshamov bound is one of the very few lower bounds on R^* in coding theory literature, while a multitude of upper bounds for the same exist.
3. Note that Theorem 4.11 states that for *any* code, $R \leq 1 - \frac{q\delta}{q-1} + o(1)$. Thus, if we keep the size of our alphabet q constant, then for large enough n, $1 - \frac{q\delta}{q-1} + o(1) < 1 - \delta$. Consequently,

   > A family of codes on a fixed alphabet size will eventually stop satisfying the Singleton bound. Alternatively stated, an MDS family of codes will necessarily see its alphabet size grow with n.

4. For $q = 2$, the Elias-Bassalygo bound outperforms even the Plotkin bound. However, for large q (say $q = 100$), the Plotkin bound is (mostly) better, except for small values of δ.

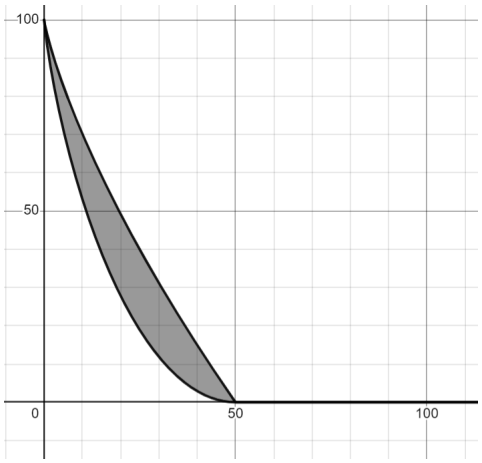

Figure 1: Feasible values of (R^*, δ) for $q = 2$, according to the bounds covered in this chapter. The upper curve is given by the Elias-Bassalygo bound, and the lower curve by the Gilbert-Varshamov bound. Both the axes have been scaled $100\times$ for clarity

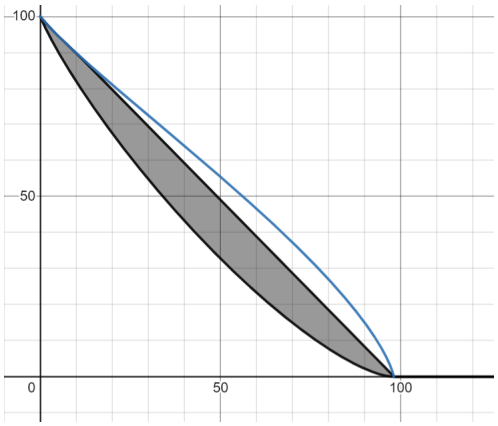

Figure 2: Feasible values of (R^*, δ) for $q = 53$, according to the bounds covered in this chapter. Note how the Plotkin bound (which is the straight line) outperforms the Elias-Bassalygo bound (the blue curve) except for small δ. The lower bound is still the Gilbert-Varshamov bound. Both the axes have been scaled $100\times$ for clarity

§5. Reed Solomon Codes

In this chapter, we shall see a lot of codes, and see many of the concepts we have seen so far in action.
Although we will most often be engrossed in deep technical details about Reed-Solomon codes, it should be kept in mind that aplications of Reed-Solomon codes can be found far beyond the boundaries of coding theory. Apart from being a very useful code in practice (Reed-Solomon codes and its variants are used for error-correction in DVD players, database storage systems such as RAID 6, satellite communications, and many other places), the properties of Reed-Solomon codes have found use in proving seminal results in computational complexity, such as the PCP theorem ([AS98]).
Thus, let us dive into the study of a very rich and powerful class of codes.

5.1. Reed-Solomon Codes: A definition

Definition 5.1. Let q be a prime power, and let $k \leq q$. For any $\mathbf{m} = (m_0, \ldots, m_{k-1}) \in \mathbb{F}_q^k$, we define the polynomial corresponding to $\mathbf{m}$ as

$$f_{\mathbf{m}}(X) := \sum_{i=0}^{k-1} m_i X^i \in \mathbb{F}_q[X]$$

Lemma 5.1. The map $\mathbb{F}_q^k \mapsto \mathbb{F}_q[X] : \mathbf{m} \mapsto f_{\mathbf{m}}$ is $\mathbb{F}_q$-linear.

Proof. The lemma merely says that for every $\mathbf{m}, \mathbf{m}_1, \mathbf{m}_2 \in \mathbb{F}_q^k, \lambda \in \mathbb{F}_q$, we have $f_{\mathbf{m}_1+\mathbf{m}_2} = f_{\mathbf{m}_1} + f_{\mathbf{m}_2}$, $f_{\lambda\mathbf{m}} = \lambda f_{\mathbf{m}}$, which can be easily seen to be true. ■

Definition 5.2 (Reed-Solomon Codes)**.** Let q be a prime power, and let $1 \leq k \leq n \leq q$ be integers. Let $\alpha_1, \alpha_2, \ldots, \alpha_n$ be distinct elements of $\mathbb{F}_q$. We then define the Reed-Solomon encoding function

$$\mathrm{RS} : \mathbb{F}_q^k \mapsto \mathbb{F}_q^n$$

$$\mathrm{RS}(\mathbf{m}) := (f_{\mathbf{m}}(\alpha_1), \ldots, f_{\mathbf{m}}(\alpha_n))$$

Remark. A few remarks are in order:

1. $\alpha_1, \alpha_2, \ldots, \alpha_n$ are also known as "evaluation points" of the RS code.
2. Note how the alphabet size is greater than or equal to the block length.

5.2. Some Basic Properties of the Reed-Solomon code

We explore the properties of the Reed-Solomon code.

Lemma 5.2. The Reed-Solomon code is linear.

Proof. Follows from Lemma 5.1. ■

Lemma 5.3. Reed-Solomon codes are MDS codes, ie:- RS codes are $[n, k, n-k+1]_q$ codes.

Proof. If there exist distinct $\mathbf{m}_1, \mathbf{m}_2 \in \mathbb{F}_q^k$ such that $\Delta(\mathrm{RS}(\mathbf{m}_1), \mathrm{RS}(\mathbf{m}_2)) \leq n-k$, then that would mean that for at least k values 'α' in $\{\alpha_1, \ldots, \alpha_n\}$, we have $f_{\mathbf{m}_1}(\alpha) = f_{\mathbf{m}_2}(\alpha) \implies f_{\mathbf{m}_1 - \mathbf{m}_2}(\alpha) = 0$, which would mean that the non-zero polynomial $f_{\mathbf{m}_1-\mathbf{m}_2}$ has $\geq k$ roots in $\mathbb{F}_q$, leading to a contradiction [9], since $\deg(f_{\mathbf{m}_1-\mathbf{m}_2}) \leq k-1$.
Consequently, the distance of an RS code is at least $n-k+1$. Now, consider the polynomial $p(X) := \prod_{i=1}^{k-1}(X-\alpha_i)$. Since p is a polynomial of degree $k-1$, there exists $\mathbf{m} \in \mathbb{F}_q^k$ such that $f_{\mathbf{m}}(X) = p(X)$. Now, note that

$$\Delta(\mathrm{RS}(\mathbf{m}), \mathrm{RS}(0)) = \sum_{i=1}^{n} \mathbb{1}_{f_{\mathbf{m}}(\alpha_i) \neq f_0(\alpha_i)} = \sum_{i=1}^{n} \mathbb{1}_{f_{\mathbf{m}}(\alpha_i) \neq 0} = \sum_{i=k}^{n} \mathbb{1}_{f_{\mathbf{m}}(\alpha_i) \neq 0} = n-k+1$$

where the last equality follows from the fact that $f_{\mathbf{m}}(\alpha_i) \neq 0$ for every $i \geq k$, since if we had $f_{\mathbf{m}}(\alpha_i) = 0$ for some $i > k$, then that would imply that $f_{\mathbf{m}}$ had $\geq k$ roots $(\alpha_1, \ldots, \alpha_{k-1}, \alpha_i)$, leading to a contradiction.
Consequently, the distance of an RS code is exactly equal to $n-k+1$, as desired. ■

For the next lemma, we'll state an identity about finite fields, without proof.

Lemma 5.4. Let q be a prime power. Then

$$\sum_{a \in \mathbb{F}_q} a^k = \begin{cases} 0, \text{ if } k \in \{0, 1, \ldots, q-2\} \\ -1, \text{ if } k = q-1 \end{cases}$$

Remark. In this lemma, we set 0^0 to be 1.
Note that the generator matrix for an RS code is

$$G = \begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & \ldots & \alpha_1^{k-1} \\ 1 & \alpha_2 & \alpha_2^2 & \ldots & \alpha_2^{k-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_n & \alpha_n^2 & \ldots & \alpha_n^{k-1} \end{bmatrix} \in \mathbb{F}_q^{n \times k}$$

Lemma 5.5. The dual of a $[q, k]_q$ RS code is once again an RS code.

Proof. Note that the generator matrix for the $[q, k]_q$ RS code is

$$G = \begin{bmatrix} 1 & \alpha_1 & \alpha_1^2 & \ldots & \alpha_1^{k-1} \\ 1 & \alpha_2 & \alpha_2^2 & \ldots & \alpha_2^{k-1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & \alpha_q & \alpha_q^2 & \ldots & \alpha_q^{k-1} \end{bmatrix} \in \mathbb{F}_q^{q \times k}$$

where $\mathbb{F}_q = \{\alpha_1, \alpha_2, \ldots, \alpha_q\}$. Now consider the matrix

$$H = \begin{bmatrix} 1 & 1 & 1 & \ldots & 1 \\ \alpha_1 & \alpha_2 & \alpha_3 & \ldots & \alpha_q \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha_1^{q-k-1} & \alpha_2^{q-k-1} & \alpha_3^{q-k-1} & \ldots & \alpha_q^{q-k-1} \end{bmatrix} \in \mathbb{F}_q^{(q-k) \times q}$$

[9] Recall that a non-zero polynomial of degree t on a field can have at most t roots

Then for $A = HG \in \mathbb{F}_q^{(q-k)\times k}$, we have

$$a_{ij} = \sum_{k=1}^{q} h_{ik}g_{kj} = \sum_{k=1}^{q} \alpha_k^{i-1} \cdot \alpha_k^{j-1} = \sum_{\alpha\in\mathbb{F}_q} \alpha^{i+j-2} = 0$$

where the last equality follows from Lemma 5.4.
Consequently, H is the parity-check matrix for the $[q,k]_q$ RS code. Very clearly, H^T is the generator matrix for the $[q,q-k]_q$ RS code, which establishes our desired result. ■

Corollary 5.6. $[2^t, 2^{t-1}]_{2^t}$ RS codes are self-dual, for $t \in \mathbb{N}$.

Proof. This follows from the above lemma and the fact that the $[q,k]_q$ RS code is the *unique* RS code (on $\mathbb{F}_q$) of dimension k. ■

5.3. Generalizations and Variants of Reed-Solomon Codes

Notation Alert: In this section, m will denote an integer. We shall instead use 'μ' to denote a message in this section. Before we state the generalized Reed-Solomon code, we re-examine Definition 5.1.

Lemma 5.7. Let $\mathbb{F}_q^{(m-1)}[X]$ denote the polynomials in $\mathbb{F}_q[X]$ with degree at most $m-1$. Then there is a bijection between $\mathbb{F}_q^{(m-1)}[X]$ and $\mathbb{F}_q^m$.

Proof. The proof is quite obvious: We identify a polynomial $p(X) := \sum_{i=0}^{m-1} \mu_i X^i$ with $(\mu_0, \ldots, \mu_{m-1})$. Clearly, this induces a bijection. ■

When we examine the above proof a bit more closely, we observe that we took a polynomial $p(X) \in \mathbb{F}[X]$, expressed it as a linear combination of the basis elements $(1, X, X^2, \ldots, X^{m-1})$, and constructed the vector out of the coefficients of the basis elements. Formalizing this notion,

Definition 5.3. Let $\mathbb{F}_q^{(m-1)}[X]$ denote the polynomials in $\mathbb{F}_q[X]$ with degree at most $m-1$. Also, let $\mathcal{B} = (b_0, \ldots, b_{m-1})$ be a basis for $\mathbb{F}_q^{(m-1)}[X]$.
We define the encoding of a polynomial $p \in \mathbb{F}_q^{(m-1)}[X]$ in the basis $\mathcal{B}$ as follows:

$$\mathrm{enc}_{\mathcal{B}}(p) = (\mu_0, \ldots, \mu_{m-1}) \in \mathbb{F}_q^m$$

where $p = \sum_{i=0}^{m-1} \mu_i b_i$.
In case our basis $\mathcal{B}$ is just the canonical basis $(1, X, \ldots, X^{m-1})$, we shall write $\mathrm{enc}_{\mathcal{B}}(p)$ simply as $\mathrm{enc}(p)$.

Before, we state the next theorem, we would like to recall that polynomials over fields also follow the Chinese Remainder Theorem.

Theorem 5.8 (Chinese Remainder Theorem for Polynomials)**.** Let $E_1(X), \ldots, E_n(X) \in \mathbb{F}[X]$ be polynomials such that $\deg(\gcd(E_i(X), E_j(X))) = 0$, for all $1 \le i < j \le n$, ie:- the polynomials are mutually co-prime. Let $d_i = \deg(E_i)$ for every $i \in [n]$, and let $D := \sum_{i=1}^n d_i$. If $A_1(X), \ldots, A_n(X) \in \mathbb{F}[X]$ are polynomials such that $\deg(A_i) < d_i$ or $A_i = 0$ for every $i \in [n]$, then there is a unique polynomial $p \in \mathbb{F}[X]$ such that $\deg(p) < D$, and $p(X) \bmod P_i(X) = A_i(X)$ for every $i \in [n]$.

Definition 5.4. Let $m \geq 1$ be an integer parameter such that $m < k \leq n$. Let $E_1(X), \ldots, E_n(X) \in \mathbb{F}_q[X]$ be polynomials of degree m such that $\deg(\gcd(E_i(X), E_j(X))) = 0$, for all $1 \leq i < j \leq n$, ie:- the polynomials are mutually co-prime.
We define the generalized RS code to be

$$\text{genRS} : \mathbb{F}_q^k \mapsto \mathbb{F}_{q^m}^n$$
$$\mu \mapsto (f_\mu(X) \bmod E_1(X), \ldots, f_\mu(X) \bmod E_n(X))$$

Now, note that we identify $\text{enc}(f_\mu(X) \bmod E_i(X)) \in \mathbb{F}_q^m$ as an element of $\mathbb{F}_{q^m}$.

Theorem 5.9. genRS is a $\left[n, \frac{k}{m}, n - \left\lfloor \frac{k-1}{m} \right\rfloor\right]_{q^m}$ code, ie:- genRS is a linear MDS code.

Remark. Note that the dimension of this code is not necessarily integral.

Proof. We first note that genRS is an injective function: Indeed, if $\text{genRS}(\mu_1) = \text{genRS}(\mu_2)$, then by Theorem 5.8 we must have $f_{\mu_1} = f_{\mu_2} \iff \mu_1 = \mu_2$. Consequently, the size of the genRS code is $|\mathbb{F}_q^k| = q^k$, and thus, the dimension is $\log_{q^m}(q^k) = \frac{k}{m}$.
Linearity of genRS can also be seen easily: Since both the maps $\mu \mapsto f_\mu$ and $f_\mu \mapsto (f_\mu \bmod E)$ are linear, and consequently, their composition is linear too.
Finally, we prove that the distance of this code is at least $n - \left\lfloor \frac{k-1}{m} \right\rfloor$: The Singleton bound then forces the distance to be exactly $n - \left\lfloor \frac{k-1}{m} \right\rfloor$. To that end, assume for the sake of contradiction that there existed some distinct $\mu_1, \mu_2 \in \mathbb{F}_q^k$ such that $\Delta(\text{genRS}(\mu_1), \text{genRS}(\mu_2)) \leq n - \left\lfloor \frac{k-1}{m} \right\rfloor - 1$. Then for at least $\left\lfloor \frac{k-1}{m} \right\rfloor + 1$ indices $i \in [n]$, we have that $f_{\mu_1} \bmod E_i = f_{\mu_2} \bmod E_i$. Now, note that $m \cdot \left(\left\lfloor \frac{k-1}{m} \right\rfloor + 1 \right) \geq k > k - 1$, which implies that by Theorem 5.8, $f_{\mu_1} = f_{\mu_2} \iff \mu_1 = \mu_2$, which is a contradiction. ■

Corollary 5.10. The generalized RS code, with $m = 1$ and $E_i(X) = X - \alpha_i, i \in [n]$ returns to us our original RS code, where $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{F}_q$ are distinct (Note that we just use the canonical encoding scheme 'enc' for each coordinate).

Proof. Follows from the fact that for any polynomial $f(X) \in \mathbb{F}[X]$, $f(X) \bmod (X - \alpha_i) = f(\alpha_i)$. ■

We now derive various codes by substituting different expressions for the polynomials $E_i(X)$. All of the codes mentioned below are very important in the context of coding theory.

5.3.1. Derivative Codes

For this section, we first formally define derivatives in finite fields.

Definition 5.5 (Derivatives)**.** Let $f = \sum_{i=0}^n f_i X^i \in \mathbb{F}[X]$ be a polynomial. We define the derivative of f to be

$$f' := \sum_{i=0}^{n-1} (i+1) f_{i+1} X^i$$

where for any $n \in \mathbb{N}$, we define $n := \underbrace{1_{\mathbb{F}} + \ldots + 1_{\mathbb{F}}}_{n \text{ times}}$.

One common theme in algebra is to extend results in analysis (ie:- results about $\mathbb{R}$ and $\mathbb{C}$) to fields with positive characteristic. In these endeavours, many a times, results which hold on fields with characteristic 0 also hold on fields with sufficiently large characteristic. We shall see an example of this in action now.

Theorem 5.11 (Taylor's theorem for positive characteristics)**.** Let q be a prime power, and let $k \leq \text{char}(\mathbb{F}_q)$. Consider a polynomial $f \in \mathbb{F}_q[X]$. Then

$$f(X) \bmod (X-\alpha)^k = f(\alpha) + \frac{f'(\alpha)}{1}(X-\alpha) + \ldots + \frac{f^{(k-1)}(\alpha)}{1 \cdot 2 \ldots \cdot (k-1)}(X-\alpha)^{k-1} = \sum_{\ell=0}^{k-1} f^{(\ell)}(\alpha)\frac{(X-\alpha)^\ell}{\ell!}$$

Now, let k, m, n be such that $k < \text{char}(\mathbb{F}_q), m < k < nm$. Consider the generalized RS code with $E_i(X) = (X - \alpha_i)^m, i \in [n]$, where $\alpha_1, \alpha_2, \ldots, \alpha_n \in \mathbb{F}_q$ are distinct.
Now, for every $\alpha_i \in \mathbb{F}_q$, denote by $\mathcal{T}_i$ the basis $\{1, (X-\alpha_i), \ldots, \frac{(X-\alpha_i)^{m-1}}{(m-1)!}\}$ of $\mathbb{F}_q^{(m-1)}[X]$. Then note that

$$\text{enc}_{\mathcal{T}_i}(f(X) \bmod E_i(X)) = \text{enc}_{\mathcal{T}_i}\left(\sum_{\ell=0}^{m-1} p^{(\ell)}(\alpha_i)\frac{(X-\alpha_i)^\ell}{\ell!}\right) = (p(\alpha_i), p'(\alpha_i), \ldots, p^{(\ell)}(\alpha_i))$$

where the first equality follows from Theorem 5.11.
Consequently, the generalized RS code, with $E_i(X) = (X-\alpha_i)^m, i \in [n]$, and with the encoding scheme provided by $\mathcal{T}_i$ for each coordinate i (note that, unlike RS codes, the encoding scheme for each coordinate is different), gives rise to what is known as a *derivative code*.

Definition 5.6 (Derivative Codes)**.** Let $m \geq 1$ be an integer, and consider k, n such that $m < k < mn$, $k < \text{char}(\mathbb{F}_q)$. Let $\alpha_1, \ldots, \alpha_n$ be distinct elements of $\mathbb{F}_q$. Then the codeword for some $\mu \in \mathbb{F}_q^k$ is

$$\begin{bmatrix} f_\mu(\alpha_1) & f_\mu(\alpha_2) & \ldots & f_\mu(\alpha_n) \\ f'_\mu(\alpha_1) & f'_\mu(\alpha_2) & \ldots & f'_\mu(\alpha_n) \\ \vdots & \vdots & \ddots & \vdots \\ f_\mu^{(m-1)}(\alpha_1) & f_\mu^{(m-1)}(\alpha_2) & \ldots & f_\mu^{(m-1)}(\alpha_n) \end{bmatrix}$$

By the properties of the generalized RS code, the derivative code is a $\left[n, \frac{k}{m}, n - \left\lfloor \frac{k-1}{m} \right\rfloor\right]_{q^m}$ code.

5.3.2. Folded Reed Solomon Codes

Let $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$ be distinct elements. Suppose there exists some $\gamma \in \mathbb{F}_q^* := \mathbb{F}_q \setminus \{0\}$ such that $S_i \cap S_j = \varnothing$ for every $1 \leq i < j \leq n$, where

$$S_i := \{\alpha_i, \gamma\alpha_i, \ldots, \gamma^{m-1}\alpha_i\}$$

We also require $\gamma^a \neq \gamma^b$ for every $a, b \in \{0, \ldots, m-1\}, a \neq b$.
Consider the generalized RS code with $E_i(X) := \prod_{j=0}^{m-1}(X - \gamma^j\alpha_i), i \in [n]$.

Now, for any $f(X) \in \mathbb{F}_q[X], i \in [n], \alpha_i \neq 0$, note that if $q(X) = f(X) \bmod E_i(X)$, then $q(\gamma^j \alpha_i) = f(\gamma^j \alpha_i)$. Now, since the degree of q is lesser than m, the expression for q can be given by Lagrange interpolation, and consequently,

$$q(X) = \sum_{j=0}^{m-1} f(\gamma^j \alpha_i) \prod_{\ell \neq j} \frac{X - \gamma^\ell \alpha_i}{\gamma^j \alpha_i - \gamma^\ell \alpha_i}$$

Thus, for every $i \in [n]$ for which $\alpha_i \neq 0$, define the basis $\mathcal{L}_i := \left\{ \prod_{\ell \neq j} \frac{X - \gamma^\ell \alpha_i}{\gamma^j \alpha_i - \gamma^\ell \alpha_i} : 0 \leq j \leq m-1 \right\}$ of $\mathbb{F}_q^{(m-1)}[X]$.
For indices i such that $\alpha_i = 0$, we fall back to our canonical basis, ie:- $\{1, X, \ldots, X^{m-1}\}$, for them.
Consequently, the generalized RS code, with $E_i(X) = (X - \alpha_i)^m, i \in [n]$, and with the encoding scheme described above gives rise to what is known as a *folded Reed Solomon code*.

Definition 5.7 (Folded Reed Solomon Code)**.** Let $\alpha_1, \ldots, \alpha_n \in \mathbb{F}_q$ be distinct elements for which there exists some $\gamma \in \mathbb{F}_q^* := \mathbb{F}_q \setminus \{0\}$ such that $S_i \cap S_j = \varnothing$ for every $1 \leq i < j \leq n$, where

$$S_i := \{\alpha_i, \gamma \alpha_i, \ldots, \gamma^{m-1} \alpha_i\}$$

Further suppose $\gamma^a \neq \gamma^b$ for every $a, b \in \{0, \ldots, m-1\}, a \neq b$.
Then the codeword for some $\mu \in \mathbb{F}_q^k$ is

$$\begin{bmatrix} f_\mu(\alpha_1) & f_\mu(\alpha_2) & \ldots & f_\mu(\alpha_n) \\ f_\mu(\gamma\alpha_1) & f_\mu(\gamma\alpha_2) & \ldots & f_\mu(\gamma\alpha_n) \\ \vdots & \vdots & \ddots & \vdots \\ f_\mu(\gamma^{m-1}\alpha_1) & f_\mu(\gamma^{m-1}\alpha_2) & \ldots & f_\mu(\gamma^{m-1}\alpha_n) \end{bmatrix}$$

By the properties of the generalized RS code, the folded Reed-Solomon code is a $\left[n, \frac{k}{m}, n - \left\lfloor \frac{k-1}{m} \right\rfloor \right]_{q^m}$ code.

5.3.3. Algebraic-Geometric Codes

Algebraic-Geometric Codes are another way to generalize Reed-Solomon codes. We shall first establish some lemmata before we define the code.

Lemma 5.12. Let p be a prime, and let $q = p^2$. Define

$$S := \{(\alpha, \beta) \in \mathbb{F}_q^2 : \beta^p + \beta = \alpha^{p+1}\}$$

Then $|S| = p^3$.

Proof. Throughout this argument, whenever we refer to $\mathbb{F}_p$, we shall be referring to the subfield of $\mathbb{F}_q$.
We make two cases:

1. $p = 2$: The equation then becomes $\beta^2 + \beta = \alpha^3$. We further make cases:
   (a) $\alpha = 0$: Then $\beta^2 + \beta = 0$. This has 2 solutions, $\beta = 0, 1$. Furthermore, since $\beta^2 + \beta$ is a quadratic equation over a field, it can't have more than 2 roots.
   (b) $\alpha \neq 0$: Note that since $\alpha \in \mathbb{F}_4$, $\alpha^4 = \alpha$. Since $\alpha \neq 0$, that implies $\alpha^3 = 1$. Now, note that the polynomial $X^2 + X + 1$ is irreducible over $\mathbb{F}_2$. Thus we identify $\mathbb{F}_4$ with $\frac{\mathbb{F}_2[X]}{(X^2+X+1)}$, and consequently there exists $r \in \mathbb{F}_4 \setminus \mathbb{F}_2$ such that $r^2 + r = 1$. Also note that if $r^2 + r = 1$, then $(1+r)^2 + (1+r) = r^2 + 2r + 1 + 1 + r =$

$r^2 + r = 1$. Thus the two elements in $\mathbb{F}_4 \setminus \mathbb{F}_2$ (ie:- r and $r+1$) are roots of the equation $\beta^2 + \beta = 1$, and consequently for every non-zero α we have exactly two β for which $\beta^2 + \beta = \alpha^3$.

Collecting the above solutions yield exactly 8 solutions, as desired.

2. $p \geq 3$: Let $r \in \mathbb{F}_p$ be a quadratic non-residue for p [10]. Then the polynomial $X^2 - r$ is irreducible over $\mathbb{F}_p$, and we thus identify $\mathbb{F}_q$ with $\frac{\mathbb{F}_p[X]}{(X^2-r)}$. Consequently, every element of $\mathbb{F}_q$ can be uniquely written as $x\eta + y$, where $\eta \in \mathbb{F}_q \setminus \mathbb{F}_p$ is such that $\eta^2 = r$, and $x, y \in \mathbb{F}_p$, ie:- if $x_1\eta + y_1 = x_2\eta + y_2$, then $x_1 = x_2, y_1 = y_2$. Consequently, set $\alpha = a_1\eta + a_2, \beta = b_1\eta + b_2$. Then

$$\beta^p + \beta = (b_1\eta + b_2)^p + (b_1\eta + b_2) = b_1^p\eta^p + b_2^p + b_1\eta + b_2$$

where the last equality follows from the fact that $(a+b)^p = a^p + b^p$ for all $a, b \in \mathbb{F}_q$. Furthermore, since $b_1, b_2 \in \mathbb{F}_p$, $b_1^p = b_1, b_2^p = b_2$, and thus

$$\beta^p + \beta = b_1\eta(r^{\frac{p-1}{2}} + 1) + 2b_2$$

Now note that since $(r^{\frac{p-1}{2}})^2 = 1$, we have $r^{\frac{p-1}{2}} = \pm 1$. However, since $r^{\frac{p-1}{2}}$ can be 1 if and only if the order of r in $\mathbb{F}_p^*$ is even, which can't be the case since r is a quadratic non-residue. Consequently, $r^{\frac{p-1}{2}} = -1$, implying that

$$\beta^p + \beta = 2b_2$$

Similarly,

$$a^{p+1} = (a_1\eta + a_2)(a_1\eta^p + a_2) = (a_1\eta + a_2)(-a_1\eta + a_2) = a_2^2 - ra_1^2$$

Consequently, note that we can choose a_1, a_2 and b_1 freely from $\mathbb{F}_p$, and then $b_2 = \frac{a_2^2 - ra_1^2}{2}$ gets fixed. Thus $|S| = p^3$.

■

For the next lemma, we shall need Eisenstein's criterion for multivariate polynomials.

Lemma 5.13 (Eisenstein's Criterion)**.** Let $\mathbb{F}$ be a field, and let $\mathbb{F}[X, Y]$ be the ring of bivariate polynomials over $\mathbb{F}$. A polynomial $f(X, Y) := Y^t + f_{t-1}(X)Y^{t-1} + \ldots + f_0(X)$ is irreducible over $\mathbb{F}[X, Y]$ if there exists a *prime* polynomial $p(X) \in \mathbb{F}[X]$ such that $p(X)$ divides $f_i(X)$ for every $i \in \{0, \ldots, t-1\}$, but $p(X)^2$ does *not* divide $f_0(X)$.
Recall that a polynomial $p(X) \in \mathbb{F}[X]$ is said to be *prime* if it has degree at least 1 and for any polynomials $a(X), b(X) \in \mathbb{F}[X]$, if $p(X)$ divides $a(X)b(X)$ then $p(X)$ either divides $a(X)$ or it divides $b(X)$.

Lemma 5.14. The polynomial $F(X, Y) := X^p + X - Y^{p+1}$ is irreducible over $\mathbb{F}_q$, where $q = p^2$ and p is a prime.

Proof. We will be done if we can show that $Y^{p+1} - (X^p + X)$ is irreducible over $\mathbb{F}_q$. Note that the polynomial $g(X) = X$ is prime in $\mathbb{F}_q[X]$, $g(X)$ divides $-(X^p + X)$ but $g(X)^2$ doesn't divide $-(X^p + X)$. We can thus conclude by Lemma 5.13. ■

As has become customary now, we shall need another fact from algebraic geometry before our next lemma.

Lemma 5.15 (Bezout's Bound)**.** If $f, g \in \mathbb{F}_q[X, Y]$ are non-zero polynomials with no common factors, then they have at most $\deg(f)\deg(g)$ common zeros.

[10] such a quadratic non-residue always exists for odd primes

Lemma 5.16. Let $n = p^3, q = p^2$, where p is a prime number. Consider the evaluation map $\mathrm{ev} : \mathbb{F}_q[X,Y] \mapsto \mathbb{F}_q^n$ defined by

$$\mathrm{ev}(f) := (f(\alpha,\beta) : (\alpha,\beta) \in S)$$

where S is defined as in Lemma 5.12.
If f is a non-zero polynomial *not* divisible by $Y^p + Y - X^{p+1}$, then the Hamming weight of $\mathrm{ev}(f)$ is atleast $n - \deg(f)(p+1)$.

Proof. Since $Y^p + Y - X^{p+1}$ is irreducible, if it doesn't divide f, then the gcd of f and $Y^p + Y - X^{p+1}$ must be of degree 0, and thus f and $Y^p + Y - X^{p+1}$ have no common factors.
Thus, by Lemma 5.15, f and $Y^p + Y - X^{p+1}$ have at most $(p+1)\deg(f)$ common zeros, implying that for at most $(p+1)\deg(f)$ elements of S, can f evaluate to 0.
The conclusion of the lemma now follows. ■

We now define a special class of bivariate polynomials which we are interested in.

Definition 5.8. For any $\ell \in \mathbb{N}$, we define

$$\mathscr{F}_\ell := \{f \in \mathbb{F}_q[X,Y] : \deg(f) \leq \ell, \deg_X(f) \leq p\}$$

where $\deg_X(f)$ is the degree of f interpreted as polynomial in X. For example, for $f(X,Y) := X^2Y^4 + Y^5 - X^4$, $\deg(f) = 6$ and $\deg_X(f) = 4$.

Note that every polynomial in $\mathscr{F}_\ell$ can be generated as a $\mathbb{F}_q$-linear combination of the polynomials $\{X^iY^j : i \leq p, i+j \leq \ell\}$. Consequently, $\mathscr{F}_\ell$ can be viewed as a $\mathbb{F}_q$-vector space of dimension $(\ell-p+1)+(\ell-p+2)+\ldots+(\ell+1) = (\ell+1)(p+1) - \frac{p(p+1)}{2}$.
We can finally define the Algebraic-Geometric code.

Definition 5.9 (Algebraic-Geometric Codes)**.** Let p be a prime number, and let $n = p^3, q = p^2$. Then the algebraic-geometric code is defined as

$$\mathrm{AG}_{p,\ell} := \{\mathrm{ev}(f) : f \in \mathscr{F}_\ell\} \subseteq \mathbb{F}_q^n$$

Lemma 5.17. $\mathrm{AG}_{p,\ell}$ is a $[n,k,d]_q$ code where

$$k \leq (\ell+1)(p+1) - \frac{p(p+1)}{2}, d \geq n - \ell(p+1)$$

Proof. The inequality for k follows from the fact that ev is $\mathbb{F}_q$-linear. Consequently, the image of $\mathscr{F}_\ell$ under ev is also a $\mathbb{F}_q$-vector space of dimension at most the dimension of $\mathscr{F}_\ell$.
Thus $k \leq (\ell+1)(p+1) - \frac{p(p+1)}{2}$.
The inequality for d follows from Lemma 5.16. Note that no polynomial in $\mathscr{F}_\ell$ is divisible by $Y^p + Y - X^{p+1}$ since the X-degree of polynomials in $\mathscr{F}_\ell$ is at most p. ■

Note that AG codes are $[n,k,d]_q$ codes where $d \geq n - k + 1 - \frac{p(p-1)}{2}$. Thus the bound for d has a deviation of $\frac{p(p-1)}{2}$ from the Singleton bound. However, the deviation is only $\mathcal{O}(p^2) = o(n)$. However, this slight deficiency from the

Singleton bound allows us to increase our block length n to $q^{\frac{3}{2}}$, whereas in RS codes we were constrained to keep $n \leq q$. In fact, AG codes have very good rate-distance tradeoffs, which we shall show next.

Theorem 5.18. AG codes beat the Gilbert-Varshamov bound for large enough p.

Proof. Note that the Gilbert Varshamov bound constructs codes with asymptotics $R \geq 1 - H_q(\delta) - \varepsilon = 1 - \delta - \mathcal{O}(\frac{1}{\log q}) - \varepsilon$.
Meanwhile, for the AG code, we have

$$d \geq n - k + 1 - \frac{p(p-1)}{2} \implies \delta \geq 1 - R + \underbrace{\frac{1}{n} - \frac{p(p-1)}{2n}}_{-\mathcal{O}(\frac{1}{\sqrt{q}})} \implies R \geq 1 - \delta - \mathcal{O}\left(\frac{1}{\sqrt{q}}\right)$$

Since $1 - \delta - \mathcal{O}\left(\frac{1}{\sqrt{q}}\right) > 1 - \delta - \mathcal{O}\left(\frac{1}{\log q}\right) - \varepsilon$ for large enough q for any fixed $\varepsilon > 0$, we have our desired result. ■

Remark. In general, in combinatorics, random objects are often extremal. In the context of coding theory, the Gilbert-Varshamov bound constructs a random linear code and achieves an asymptotic rate of $1 - H_q(\delta)$ from it. Thus, for many years, the consensus in the coding theory community was that the Gilbert-Varshamov bound was optimal. That notion was dispelled with the advent of algebraic-geometric codes, whose careful construction managed to beat the rate of a random linear code.

5.3.4. BCH Codes

BCH codes are named after their discoverers, Bose, Ray-Chaudhuri, and Hocquenghem.

Definition 5.10. Let q be a prime power. Consider the k-dimensional RS code $[q-1, k]_q$ on $\mathbb{F}_q$ generated by using all the non-zero elements of $\mathbb{F}_q$ as evaluation points, ie:- every element of $\mathbb{F}_q^*$ is used as an evaluation point.
We denote this RS code as $\mathrm{RS}_{\mathbb{F}_q^*}[k]$.

Now, recall that if $\mathbb{F}$ is a finite field, then $(\mathbb{F}^*, 1, \cdot)$ is a cyclic group, where $\mathbb{F}^* := \mathbb{F} \setminus \{0\}$. Consequently, there exists a *primitive element* $\alpha \in \mathbb{F}^*$ such that $\mathbb{F}^* = \{1 = \alpha^0, \alpha^1, \alpha^2, \ldots, \alpha^{|\mathbb{F}|-2}\}$. Thus the parity check matrix of $\mathrm{RS}_{\mathbb{F}^*}[k]$ is as given below:

$$H_{\mathbb{F}^*,k} = \begin{bmatrix} 1 & 1 & 1 & \ldots & 1 \\ \alpha^0 & \alpha^1 & \alpha^2 & \ldots & \alpha^{|\mathbb{F}|-2} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha^{0\cdot(|\mathbb{F}|-k-2)} & \alpha^{1\cdot(|\mathbb{F}|-k-2)} & \alpha^{2\cdot(|\mathbb{F}|-k-2)} & \ldots & \alpha^{(|\mathbb{F}|-2)\cdot(|\mathbb{F}|-k-2)} \end{bmatrix}$$

Definition 5.11 (BCH Codes)**.** Consider the code $\mathrm{RS}_{\mathbb{F}_{2^m}^*}[k]$, ie:- $n = 2^m - 1$. All of its codewords reside in $\mathbb{F}_{2^m}^n$. Now, consider the inclusion $\mathbb{F}_2 \hookrightarrow \mathbb{F}_{2^m}$, and consider the code

$$C_{\text{BCH}} := \mathrm{RS}_{\mathbb{F}_{2^m}^*}[k] \cap \mathbb{F}_2^n$$

ie:- the BCH code consists of all codewords in the Reed-Solomon code whose only entries are $0, 1$.

Before we analyze the parameters of the BCH code, we establish some small facts about finite fields.

Lemma 5.19. Consider n numbers $c_0, \ldots, c_{n-1} \in \mathbb{F}_2$. Let $m \in \mathbb{N}$, and consider an indeterminate $\alpha \in \mathbb{F}_{2^m}$ satisfying the equation

$$c_0 + c_1\alpha + \ldots + c_{n-1}\alpha^{n-1} = 0$$

The above equation is equivalent to a system of m linear equations in $\mathbb{F}_2$.

Proof. We endow $\mathbb{F}_{2^m}$ with a vector space structure over $\mathbb{F}_2$, ie:- we pick a basis $\mathcal{B} = \{\beta_1 = 1, \beta_2, \ldots, \beta_m\} \subseteq \mathbb{F}_{2^m}$ such that every element of $\mathbb{F}_{2^m}$ can be written as a unique linear combination $\sum_{i=1}^m \eta_i\beta_i$, where $\eta_i \in \mathbb{F}_2 = \{0, 1\}, i \in [m]$. Note that if we have $\eta \in \mathbb{F}_2 \hookrightarrow \mathbb{F}_{2^m}$, then the representation of η in the vector space $\mathbb{F}_2^m$ with $\mathcal{B}$ as a basis is just $\begin{bmatrix}\eta & 0 & \ldots & 0\end{bmatrix}^\mathsf{T}$.
Now, consider the map $\mathcal{M}_\alpha : \mathbb{F}_{2^m} \mapsto \mathbb{F}_{2^m}$, where $\mathcal{M}_\alpha(x) := \alpha x$. Clearly, $\mathcal{M}_\alpha$ is $\mathbb{F}_2$-linear, since $\mathcal{M}_\alpha(x + y) = \mathcal{M}_\alpha(x) + \mathcal{M}_\alpha(y)$ and $\mathcal{M}_\alpha(\eta x) = \eta\mathcal{M}_\alpha(x)$ for every $x, y \in \mathbb{F}_{2^m}, \eta \in \mathbb{F}_2$. Consequently $\mathcal{M}_\alpha$ can be associated with a matrix $\mathbf{M}_\alpha \in \mathbb{F}_2^{m\times m}$ such that for any $v \in \mathbb{F}_{2^m}$, if the representation of v in the vector space $\mathbb{F}_2^m$ (according to the basis $\mathcal{B}$) is $(v_1, \ldots, v_m)$, then $\mathbf{M}_\alpha \cdot \begin{bmatrix}v_1 & \ldots & v_m\end{bmatrix}^\mathsf{T}$ gives us the vector space representation of $\mathcal{M}_\alpha(v) = \alpha v$.
Consequently, the equation $c_0 + c_1\alpha + \ldots + c_{n-1}\alpha^{n-1} = 0$ translates to, in $\mathbb{F}_2^m$, to:

$$\begin{bmatrix}c_0\\0\\\vdots\\0\end{bmatrix} + \mathbf{M}_\alpha\begin{bmatrix}c_1\\0\\\vdots\\0\end{bmatrix} + \ldots + \mathbf{M}_\alpha^{n-1}\begin{bmatrix}c_{n-1}\\0\\\vdots\\0\end{bmatrix} = \begin{bmatrix}0\\0\\\vdots\\0\end{bmatrix}$$

Comparing this equation component-wise yields m linear equations over $\mathbb{F}_2$, as desired. ∎

Lemma 5.20. If $f(X) \in \mathbb{F}_2[X]$, then $f(X^2) = f(X)^2$.

Proof. Recall that if $x, y \in \mathbb{F}_{p^s}$, then $(x + y)^p = x^p + y^p$. Consequently, let $f(X) = c_0 + c_1X + \ldots + c_{n-1}X^{n-1}$, where $c_0, \ldots, c_{n-1} \in \mathbb{F}_2$. Then

$$f(X)^2 = (c_0 + c_1X + \ldots + c_{n-1}X^{n-1})^2 = c_0^2 + (c_1X)^2 + \ldots + (c_{n-1}X^{n-1})^2$$

Now, also note that for any $\eta \in \mathbb{F}_2, \eta^2 = \eta$. Thus

$$c_0^2 + (c_1X)^2 + \ldots + (c_{n-1}X^{n-1})^2 = c_0^2 + c_1^2X^2 + \ldots + c_{n-1}^2X^{2(n-1)} = c_0 + c_1X^2 + \ldots + c_{n-1}X^{2(n-1)} = f(X^2)$$

as desired. ∎

Corollary 5.21. Let $f(X) \in \mathbb{F}_2[X] \hookrightarrow \mathbb{F}_{2^m}[X]$, and suppose $f(\alpha) = 0$ for some $\alpha \in \mathbb{F}_{2^m}$. Then $f(\alpha^2) = 0$.

Lemma 5.22. The BCH code is a $[n, k', d']_2$ code, where $n = 2^m - 1$ for some $m \in \mathbb{N}$, $d' \geq n - k + 1$ and $k' \geq n - \left\lceil\frac{d'-1}{2}\right\rceil \log_2(n + 1)$.

Proof. Note that $\mathrm{RS}_{\mathbb{F}_{2^m}^*}[k]$ is a subspace of $\mathbb{F}_{2^m}^n$, and so is $\mathbb{F}_2^n$. Thus the BCH code, being the intersection of two subspaces is also a subspace and is consequently linear.

Note that since the BCH code is a subset of the RS code, and since the RS code has distance $n-k+1$, the distance between any two codewords in the BCH code must also be $\geq n-k+1$, as desired.
Now, let α be any primitive element of $\mathbb{F}_{2^m}$. Since the BCH code is a sub-code of an RS code, the parity check matrix of the RS code nullifies the codewords of the BCH code too. Thus, let $c=(c_0,c_1,\ldots,c_{n-1})\in\mathbb{F}_2^n$ be an element of the BCH code. Then we have

$$\begin{bmatrix} \alpha^{0\cdot 0} & \alpha^{1\cdot 0} & \alpha^{2\cdot 0} & \ldots & \alpha^{(n-1)\cdot 0} \\ \alpha^{0\cdot 1} & \alpha^{1\cdot 1} & \alpha^{2\cdot 1} & \ldots & \alpha^{(n-1)\cdot 1} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ \alpha^{0\cdot(n-k-1)} & \alpha^{1\cdot(n-k-1)} & \alpha^{2\cdot(n-k-1)} & \ldots & \alpha^{(n-1)\cdot(n-k-1)} \end{bmatrix} \begin{bmatrix} c_0 \\ c_1 \\ \vdots \\ c_{n-1} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix}$$

Thus

$$f_c(1)=f_c(\alpha)=\ldots=f_c(\alpha^{n-1-k})=0$$

where $f_c(X)=c_0+c_1X+\ldots+c_{n-1}X^{n-1}\in\mathbb{F}_2[X]$.
Now, from Corollary 5.21, we know that $f_c(\alpha^j)=0$ implies $f_c(\alpha^{2j})=0$. Thus the solution space of the system $f_c(1)=f_c(\alpha)=\ldots=f_c(\alpha^{n-1-k})=0$ is the same as the solution space of the system $f_c(\alpha^1)=f_c(\alpha^3)=f_c(\alpha^5)=\ldots=0$, ie:- we only care about the odd exponents of α in the system now. Now, we have at most $\left\lceil\frac{n-k}{2}\right\rceil$ equations in $f_c(\alpha^1)=f_c(\alpha^3)=f_c(\alpha^5)=\ldots=0$, and by Lemma 5.19, each equation $f_c(\alpha^j)=0$ translates to $m=\log_2(n+1)$ equations in $\mathbb{F}_2$.
Thus the solution space of the system $f_c(\alpha^1)=f_c(\alpha^3)=f_c(\alpha^5)=\ldots=0$ is in bijection with the solution space of $\left\lceil\frac{n-k}{2}\right\rceil\cdot m=\left\lceil\frac{n-k}{2}\right\rceil\cdot\log_2(n+1)$ linear equations in $\mathbb{F}_2$.
Thus the nullity of our BCH code is at most $\left\lceil\frac{n-k}{2}\right\rceil\cdot\log_2(n+1)$, and consequently the dimension of our code is at least $n-\left\lceil\frac{n-k}{2}\right\rceil\cdot\log_2(n+1)\geq n-\left\lceil\frac{d'-1}{2}\right\rceil\cdot\log_2(n+1)$, as desired. ■

5.4. Applications

We use the aforementioned construction of BCH codes to give a very efficient construction of a k-wise independent source.

Definition 5.12. A set of vectors $S\subseteq\mathbb{F}_q^n$ is called t-wise independent if for every $I\subseteq[n]$ with $|I|=t$, every vector in $\mathbb{F}_q^t$ appears an equal number of times as a projection of some element $s\in S$ onto I.

Remark. A few remarks are in order.

1. $(n,k)_q$ MDS codes are k-wise independent. In fact, every vector in $\mathbb{F}_q^k$ occurs exactly once as a projection of some codeword in $(n,k)_q$. This follows from Theorem 4.7.
2. Given n random variables $X_1,\ldots,X_n$, taking values in R, we say they are k-wise independent if $\Pr(X_{i_1}=r_1,\ldots,X_{i_k}=r_k)=\prod_{\ell=1}^k\Pr(X_{i_\ell}=r_\ell)$ holds for every $\{i_1,i_2,\ldots,i_k\}\subset[n]$ and every $(r_1,\ldots,r_k)\in R^k$. Then note that if $S\subseteq\mathbb{F}_q^n$ is k-wise independent, then the random vectors $X_1,X_2,\ldots,X_n$ are also k-wise independent, where X_i is the random variable denoting the i^{th} entry of a uniformly randomly chosen element of S.

Theorem 5.23. One can compute n random bits which are t-wise independent using $\left\lceil\frac{t}{2}\right\rceil\log_2(1+n)$ uniform independent random bits, using BCH codes.

Proof. This result follows (after some effort) from Proposition 6.5 in [ABI86]. ■

§6. Alternative Noise Models: Shannon's Theorem

So far we have implicitly worked with the Hamming model of a channel, which upper bounds the number of errors a transmitted codeword can have: However, those errors are allowed to occur on any bit of the codeword, which thus includes within itself the possibility that an adversary could selectively corrupt bits to make decoding impossible/erroneous.
However, another large class of noise models is derived from the assumption that noise is a stochastic process: Thus, in most stochastic noise models, there is no upper bound on the number of errors that may take place: There is a non-zero probability that every bit might be corrupted.
This class of noise models was introduced and investigated by Shannon, who proved an eponymous theorem regarding rate-error tradeoffs for stochastic noise models.
But before stating Shannon's theorem, we first define our error model precisely.

Definition 6.1 (Binary Symmetric Channel)**.** Fix a parameter $p \in \left[0, \frac{1}{2}\right]$. The binary symmetric channel is a channel that flips every bit with probability p, independently of other bits.
This channel is denoted as BSC_p, and we shall write $e \sim \mathrm{BSC}_p$ to denote an error e sampled from the BSC_p channel. Note that if our channel is a n-bit channel, then $e \in \{0,1\}^n$.

Remark. A few remarks are in order:

1. There are lots of other stochastic noise models, but we will only consider the BSC model. Consequently, we will only be talking about binary codes in this chapter.
2. If $x \in \{0,1\}^n$ is the original codeword, and $e \in \{0,1\}^n$ is the error, then the transmitted codeword is $x+e = x \oplus e$.

6.1. Shannon's Theorem

Theorem 6.1 (Shannon's Theorem)**.** Consider real numbers p, ε such that $p \in \left(0, \frac{1}{2}\right), \varepsilon \in \left[0, \frac{1}{2} - p\right]$. Then for large enough $n \in \mathbb{N}$, we have that:

1. There exists a real $\delta > 0$, a positive integer $k \leq \lfloor (1 - H(p+\varepsilon))n \rfloor$, and functions $E : \{0,1\}^k \mapsto \{0,1\}^n, D : \{0,1\}^n \mapsto \{0,1\}^k$, such that for every $\mu \in \{0,1\}^k$:
$$\Pr_{e \sim \mathrm{BSC}_p}(D(E(\mu) + e) \neq \mu) \leq 2^{-\delta n}$$
2. If $k \geq \lceil (1 - H(p) + \varepsilon)n \rceil$, then for *any* pair of functions $E : \{0,1\}^k \mapsto \{0,1\}^n, D : \{0,1\}^n \mapsto \{0,1\}^k$, there exists a $\mu \in \{0,1\}^k$ such that
$$\Pr_{e \sim \mathrm{BSC}_p}(D(E(\mu) + e) \neq \mu) \geq \frac{1}{2}$$

Remark. A few remarks are in order:

1. Note how in the first part we have ε *inside* the entropy function ('$1 - H(p+\varepsilon)$'), while in the second part, we have it outside the entropy function ('$1 - H(p) + \varepsilon$').
2. The reason why Shannon's theorem is so important is that it precisely identifies $1 - H(p)$ to be the "threshold" up to which reliable transmission is possible on the BSC_p channel. In this respect, it is somewhat similar to the Singleton Bound. Indeed, if we want to tolerate p fraction of errors in the Hamming model of noise, we need our relative distance δ to be at least $2p$, and then Theorem 4.5 forces our rate R to be at most $1 - \delta \leq 1 - 2p$.

We will give proofs for the two parts separately. We will prove the second part of the theorem first.

Proof of the second part. Suppose $k \geq \lceil (1 - H(p) + \varepsilon)n \rceil$, and assume for the sake of contradiction there exist $E : \{0,1\}^k \mapsto \{0,1\}^n, D : \{0,1\}^n \mapsto \{0,1\}^k$ such that for every $\mu \in \{0,1\}^k$,

$$\Pr_{e \sim \text{BSC}_p}(D(E(\mu) + e) \neq \mu) < \frac{1}{2} \iff \Pr_{e \sim \text{BSC}_p}(E(\mu) + e \notin D^{-1}(\mu)) < \frac{1}{2}$$

Now set $\gamma = \frac{\varepsilon}{2p \log_2\left(\frac{1}{p} - 1\right)}$, and for every $\mu \in \{0,1\}^k$, define

$$S_\mu := B(E(\mu), (1+\gamma)pn) \setminus B(E(\mu), (1-\gamma)pn)$$

Now, by Theorem A.3,

$$\Pr(E(\mu) + e \notin S_\mu) < 2e^{-\frac{\gamma^2 pn}{3}} = 2^{-\Omega(\gamma^2 n)}$$

Thus by Lemma A.1,

$$\Pr((E(\mu) + e) \notin S_\mu \cap D^{-1}(\mu)) < \frac{1}{2} + 2^{-\Omega(\gamma^2 n)} \implies \Pr((E(\mu) + e) \in S_\mu \cap D^{-1}(\mu)) > \frac{1}{2} - 2^{-\Omega(\gamma^2 n)} \geq \frac{1}{4}$$

where the last inequality holds for large enough n.
On the other hand, note that

$$\Pr((E(\mu)+e) \in S_\mu \cap D^{-1}(\mu)) \leq |S_\mu \cap D^{-1}(\mu)| \cdot \max_{y \in S_\mu} \Pr(E(\mu)+e = y) = |S_\mu \cap D^{-1}(\mu)| \cdot \max_{d \in [(1-\gamma)pn, (1+\gamma)pn]} p^d (1-p)^{n-d}$$

Some elementary calculus yields that $p^d(1-p)^{n-d}$ is a decreasing function of d for $p \leq \frac{1}{2}$, and consequently

$$\max_{d \in [(1-\gamma)pn, (1+\gamma)pn]} p^d (1-p)^{n-d} = p^{(1-\gamma)pn}(1-p)^{n-(1-\gamma)pn} = \left(\frac{1-p}{p}\right)^{\gamma pn} 2^{-nH(p)}$$

Thus

$$\frac{1}{4} \leq \Pr((E(\mu) + e) \in S_\mu \cap D^{-1}(\mu)) \leq |S_\mu \cap D^{-1}(\mu)| \cdot \left(\frac{1-p}{p}\right)^{\gamma pn} 2^{-nH(p)}$$

$$\implies |S_\mu \cap D^{-1}(\mu)| \geq \frac{1}{4}\left(\frac{1-p}{p}\right)^{-\gamma pn} 2^{nH(p)}$$

Finally, note that

$$\{0,1\}^n = \bigsqcup_{\mu \in \{0,1\}^k} D^{-1}(\mu) \implies 2^n = \sum_{\mu \in \{0,1\}^k} |D^{-1}(\mu)| \geq \sum_{\mu \in \{0,1\}^k} |D^{-1}(\mu) \cap S_\mu|$$

$$\geq 2^k \cdot \frac{1}{4}\left(\frac{1-p}{p}\right)^{-\gamma pn} 2^{nH(p)} > 2^{k + nH(p) - \varepsilon n}$$

where the last inequality follows, for large enough n, when we substitute $\gamma = \frac{\varepsilon}{2p \log_2\left(\frac{1}{p} - 1\right)}$.

But since $k \geq \lceil (1 - H(p) + \varepsilon)n \rceil$, $2^n > 2^{k + nH(p) - \varepsilon n}$ is a contradiction, as desired. ■

Before proving the first part, we establish some lemmata.

Lemma 6.2. Let $k \leq (1 - H(p + \varepsilon))n$, where p, ε are as they were defined in Theorem 6.1.
Let $E : \{0,1\}^k \mapsto \{0,1\}^n$ be a random encoding function, ie:- for every $\mu \in \{0,1\}^k$, $E(\mu)$ is uniformly random in $\{0,1\}^n$. Let D be the maximum likelihood decoding function corresponding to E: In case some $x \in \{0,1\}^n$ is equally close to $E(\mu_1), E(\mu_2), \ldots, E(\mu_t)$, we arbitrarily assign one of $\mu_1, \mu_2, \ldots, \mu_t$ as $D(x)$.
Then there exists some $\delta' > 0$ such that for any $\mu \in \{0,1\}^k$:

$$\mathbb{E}_E\left[\Pr_{e \sim \text{BSC}_p}\left(D(E(\mu) + e) \neq \mu\right)\right] \leq 2^{-\delta' n}$$

Proof. Let $\varepsilon' := \frac{\varepsilon}{2}$. Also define $y_\mu := E(\mu) + e$. Then

$$\Pr_{e\sim \mathrm{BSC}_p}\left(D(E(\mu)+e)\neq \mu\right) = \sum_{y\in B(E(\mu),(p+\varepsilon')n)} \Pr(y_\mu = y)\mathbb{1}_{D(y)\neq\mu} + \sum_{y\notin B(E(\mu),(p+\varepsilon')n)} \Pr(y_\mu = y)\mathbb{1}_{D(y)\neq\mu}$$

Now,

$$\sum_{y\notin B(E(\mu),(p+\varepsilon')n)} \Pr(y_\mu = y)\mathbb{1}_{D(y)\neq\mu} \leq \sum_{y\notin B(E(\mu),(p+\varepsilon')n)} \Pr(y_\mu = y) = \Pr(y_\mu \notin B(E(\mu),(p+\varepsilon')n)) \leq e^{-\frac{n(\varepsilon')^2}{2}}$$

where the last inequality follows by Theorem A.3.
Thus

$$\Pr_{e\sim \mathrm{BSC}_p}\left(D(E(\mu)+e)\neq \mu\right) \leq \sum_{y\in B(E(\mu),(p+\varepsilon')n)} \Pr(y_\mu = y)\mathbb{1}_{D(y)\neq\mu} + e^{-\frac{n(\varepsilon')^2}{2}}$$

$$\implies \mathbb{E}_E\left[\Pr_{e\sim \mathrm{BSC}_p}\left(D(E(\mu)+e)\neq \mu\right)\right] \leq \mathbb{E}_E\left[\sum_{y\in B(E(\mu),(p+\varepsilon')n)} \Pr(y_\mu = y)\mathbb{1}_{D(y)\neq\mu}\right] + e^{-\frac{n(\varepsilon')^2}{2}}$$

By linearity of expectation,

$$\mathbb{E}_E\left[\sum_{y\in B(E(\mu),(p+\varepsilon')n)} \Pr(y_\mu = y)\mathbb{1}_{D(y)\neq\mu}\right] = \sum_{y\in B(E(\mu),(p+\varepsilon')n)} \mathbb{E}_E\left[\Pr(y_\mu = y)\mathbb{1}_{D(y)\neq\mu}\right]$$

$$= \sum_{y\in B(E(\mu),(p+\varepsilon')n)} \Pr_{e\sim \mathrm{BSC}_p}(y_\mu = y)\mathbb{E}_E\left[\mathbb{1}_{D(y)\neq\mu}\right]$$

Note that

$$\mathbb{E}_E\left[\mathbb{1}_{D(y)\neq\mu}\right] = \Pr_E\left(D(y)\neq\mu \middle| y_\mu = y\right) \leq \sum_{\mu'\neq\mu} \Pr\left(\Delta(E(\mu'),y) = \Delta(E(\mu),y)\middle| y_\mu = y\right)$$

where the last inequality is quite apparent when we recall that D was the MLD decoder: Thus D can fail to decode a message only when there are other, equidistant messages [11]. Now, if $y \in B(E(\mu),(p+\varepsilon')n)$, then $\Delta(E(\mu),y) \leq (p+\varepsilon')n$. Consequently, $\Delta(E(\mu'),y) \leq (p+\varepsilon')n \iff E(\mu') \in B(y,(p+\varepsilon')n)$, where μ,μ' are as in the summation above.
Thus

$$\mathbb{E}_E\left[\mathbb{1}_{D(y)\neq\mu}\right] \leq \sum_{\mu'\neq\mu} \Pr\left(\Delta(E(\mu'),y) = \Delta(E(\mu),y)\middle| y_\mu = y\right) = \sum_{\mu'\neq\mu} \frac{|B(y,(p+\varepsilon')n)|}{2^n} \leq \sum_{\mu'\neq\mu} \frac{2^{nH(p+\varepsilon')}}{2^n}$$

$$= (2^k-1)\frac{2^{nH(p+\varepsilon')}}{2^n} < 2^{k-n(1-H(p+\varepsilon'))}$$

Since $k \leq n(1-H(p+\varepsilon))$,

$$\mathbb{E}_E\left[\mathbb{1}_{D(y)\neq\mu}\right] \leq 2^{-n(H(p+\varepsilon)-H(p+\varepsilon'))}$$

Thus

$$\mathbb{E}_E\left[\Pr_{e\sim \mathrm{BSC}_p}\left(D(E(\mu)+e)\neq \mu\right)\right] \leq \sum_{y\in B(E(\mu),(p+\varepsilon')n)} \Pr_{e\sim \mathrm{BSC}_p}(y_\mu = y)\cdot 2^{-n(H(p+\varepsilon)-H(p+\varepsilon'))} + e^{-\frac{n(\varepsilon')^2}{2}}$$

Finally,

$$\sum_{y\in B(E(\mu),(p+\varepsilon')n)} \Pr_{e\sim \mathrm{BSC}_p}(y_\mu = y) \leq \sum_{y\in\{0,1\}^n} \Pr_{e\sim \mathrm{BSC}_p}(y_\mu = y) = 1$$

[11] in which case our MLD scheme assigns an output randomly

Thus

$$\mathbb{E}_E\left[\Pr_{e\sim \mathrm{BSC}_p}\left(D(E(\mu)+e)\neq \mu\right)\right] \leq 2^{-n(H(p+\varepsilon)-H(p+\varepsilon'))} + e^{-\frac{n(\varepsilon')^2}{2}} = 2^{-n(H(p+\varepsilon)-H(p+\frac{\varepsilon}{2}))} + 2^{-\frac{n\varepsilon^2}{8\ln 2}}$$

where we recall that $\varepsilon' = \frac{\varepsilon}{2}$.
The above expression can be upper bounded by $2^{-\delta' n}$ for some small enough $\delta' > 0$, for large enough n. We have thus proved our desired result. ■

Thus, for every $\mu \in \{0,1\}^k$, $\mathbb{E}_E\left[\Pr_{e\sim \mathrm{BSC}_p}\left(D(E(\mu)+e)\neq \mu\right)\right] \leq 2^{-\delta' n}$. Consequently

$$\mathbb{E}_\mu\left[\mathbb{E}_E\left[\Pr_{e\sim \mathrm{BSC}_p}\left(D(E(\mu)+e)\neq \mu\right)\right]\right] \leq 2^{-\delta' n}$$

But

$$\mathbb{E}_\mu\left[\mathbb{E}_E\left[\Pr_{e\sim \mathrm{BSC}_p}\left(D(E(\mu)+e)\neq \mu\right)\right]\right] = \mathbb{E}_E\left[\mathbb{E}_\mu\left[\Pr_{e\sim \mathrm{BSC}_p}\left(D(E(\mu)+e)\neq \mu\right)\right]\right]$$

Consequently, there exists some E^* such that $\mathbb{E}_\mu\left[\Pr_{e\sim \mathrm{BSC}_p}\left(D(E^*(\mu)+e)\neq \mu\right)\right] \leq 2^{-\delta' n}$.
Thus for E^*, the *average* decoding error is upper bounded. We now show that by throwing away half of the codewords, the *maximum* decoding error is also upper bounded. This process of *throwing away* codewords to *convert an average-case bound to a worst-case bound* is also known as *expurgation* [12].

Lemma 6.3. Let $\mu_1, \mu_2, \ldots, \mu_{2^k}$ be an ordering of $\{0,1\}^k$ such that for

$$p_i := \Pr_{e\sim \mathrm{BSC}_p}\left(D(E^*(\mu_i)+e)\neq \mu_i\right)$$

we have $p_1 \leq p_2 \leq \ldots \leq p_{2^k}$.
Then $p_{2^{k-1}} \leq 2^{1-n\delta'}$.

Proof. Note that

$$\frac{1}{2^k}\sum_{i=1}^{2^k} p_i = \mathbb{E}_\mu\left[\Pr_{e\sim \mathrm{BSC}_p}\left(D(E^*(\mu)+e)\neq \mu\right)\right] \leq 2^{-\delta' n}$$

Now, assume for the sake of contradiction that $p_{2^{k-1}} > 2^{1-n\delta'}$. Then

$$\frac{1}{2^k}\sum_{i=1}^{2^k} p_i \geq \frac{1}{2^k}\sum_{i=2^{k-1}}^{2^k} p_i > \frac{1}{2^k} 2^{k-1} 2^{1-\delta' n} = 2^{-\delta' n}$$

leading to a contradiction. ■

Proof of the first part. Following the groundwork laid above, the proof is quite straightforward: Define $\delta = \delta' + \frac{1}{n}$. In Lemma 6.2, choose some $k \leq (1-H(p+\varepsilon))n$, and conclude that there is some E^* such that

$$\mathbb{E}_\mu\left[\Pr_{e\sim \mathrm{BSC}_p}\left(D(E^*(\mu)+e)\neq \mu\right)\right] \leq 2^{-\delta' n}$$

Apply Lemma 6.3 to get that if we reduce our dimension by 1, ie:- set $k' = k-1$ (note that $k' \leq \lfloor (1-H(p+\varepsilon))n \rfloor$), then our maximum decoding error is bounded above by $2^{1-n\delta'} = 2^{-n\delta}$, as desired. ■

Note that even though we presented Shannon's theorem, we haven't given any explicit construction of a code that achieves the optimal rate predicted by Shannon's theorem. Such codes do exist, but we shall not have the occasion to see them here.

[12] The technique of *expurgation* is pretty common across computer science: For example, in these notes [Pan17], one may see expurgation in action to establish the Goldreich-Levin theorem.

6.2. Shannon vs. Hamming

Note that in the Shannon model, the errors are stochastic. Furthermore, for BSC_p, the errors in different bits are independent too. This makes the probabilistic error model much weaker than the Hamming error model, where various errors can be chosen adversarially to thwart the decodability of the transmitted codeword.
Stated equivalently, given a particular code, and a particular parameter denoting the fraction of errors, we can achieve higher rates in the Shannon model than in the Hamming model.
One can argue about that as follows:

Lemma 6.4. Consider $p \in \left(0, \frac{1}{2}\right), \varepsilon \in \left(0, \frac{1}{2} - p\right]$. If some algorithm $\mathcal{A}$ can handle $(p + \varepsilon)$ fraction of errors in the transmitted codewords, then BSC_p can be used for reliable communication.

Proof. Indeed, by Theorem A.3, with probability $\geq 1 - e^{-n\varepsilon^2/2}$, the fraction of errors is $\leq p + \varepsilon$. We can then invoke $\mathcal{A}$ to recover the transmitted codeword. ■

Consequently, if the relative distance of our code is $> 2p + \varepsilon$, then reliable decoding is possible over BSC_p.
Although we won't cover it in this chapter, an analogous statement to Theorem 6.1 holds for qSC_p channels too, ie:- consider a channel transmitting codewords in $\mathbb{F}_q^n$, and every entry of the codeword can get changed to some different element of $\mathbb{F}_q$ with probability p. Then if $p < 1 - \frac{1}{q}$, then reliable communication is possible for $R \lessapprox 1 - H_q(p)$. Furthermore, one can show, using the tools developed in Appendix A.2, that $1 - H_q(p) \in [1 - p - \varepsilon, 1 - p]$ for $\varepsilon = \mathcal{O}\left(\frac{1}{\log_2(q)}\right)$. Consequently, for large enough q, one can have reliable communication with $R \approx 1 - H_q(p) \approx 1 - p$ in the Shannon model. Stated differently, a Shannon channel with rate R can tolerate $\sim 1 - R$ fraction of errors.
On the other hand, for the Hamming model, decodability is possible only if the number of errors made is at most $\frac{1}{2}$ the distance, ie:- $e < \frac{d}{2}$.
For the sake of comparison with Shannon's theorem, assume that the fraction of errors is p. Then $p < \frac{\delta}{2}$, where δ is the relative distance.
But by Theorem 4.5, $\delta \leq 1 - R$, where R is the rate of the code. Consequently, for the Hamming model, we need to have $p < \frac{1-R}{2}$ if we want decodability.
Thus, we see a quantitative verification of our intuition: The Shannon model can tolerate *twice* as many errors as the Hamming model. We shall later see a method of "bridging" this gap.

§7. List Decoding

As noted in the last chapter, there is a factor of about 2 in the fraction of errors that can be tolerated in the Shannon model, vs. in the Hamming model.
Now, suppose we give up on our insistence on *unique* decoding, ie:- given a transmitted codeword, we no longer insist that there be some algorithm that tells us what the original codeword was. We instead demand only a *list* of possible codewords as candidates from our decoding algorithm.
But then how do we deal with the real-life problem of assigning a unique codeword as the decoded result of a transmitted codeword? We go about it as follows: Suppose our list is of size L. Then we include some "metadata" within codewords which helps us in choosing the correct codeword from the list of L codewords.
Note that the above process effectively reduces the dimension of our code by $\log_2 L$, since you need that many bits to encode the metadata. But if L is not too large, then this slight decrease in the dimension of the code is permissible: After all, it helps us tolerate more errors in our channel.
We shall now make the aforementioned notions precise.

7.1. List Decoding

Definition 7.1. Given $\rho \in [0,1], L \geq 1$, we say that a code $C \subseteq \Sigma^n$ is (ρ, L)-list decodable if for every received word $y \in \Sigma^n$,

$$|\{c \in C : \Delta(c,y) \leq \rho n\}| \leq L$$

Note that since we will eventually be concerned with efficient (read poly-time) algorithms for list decoding, we will generally consider L to be $\text{poly}(n)$, because if L is superpolynomial, then any list decoding algorithm will have to output a list of length L, which itself will not be a poly-time task.
Now, having set up this notion of list decoding, we want to explore how many errors we can tolerate in this paradigm. To do that, we recall Lemma 4.17, which we restate here for the reader's convenience:

Definition 7.2. We define the q-ary Johnson function $J_q : \left[0, 1 - \frac{1}{q}\right] \mapsto \mathbb{R}$ as

$$J_q(x) := \left(1 - \frac{1}{q}\right)\left(1 - \sqrt{1 - \frac{qx}{q-1}}\right)$$

Lemma 7.1 (Johnson Bound)**.** Consider any code $C = (n,k,d)_q$, and let $e < J_q(\delta)n$, where $\delta = \frac{d}{n} \leq 1 - \frac{1}{q}$. Let $y \in \mathbb{F}_q^n$ be any arbitrary vector. Then $|B(y,e) \cap C| \leq qdn = q\delta n^2$. In other words, C is $(\rho, q\delta n^2)$-list decodable, for any $\rho < J_q(\delta)$.

Finally, for the sake of comparison, we state an inequality for $J_q(\cdot)$.

Lemma 7.2. Let $q \geq 2$ be an integer, and let $x \in \left(0, 1 - \frac{1}{q}\right)$. Then

$$J_q(x) \geq 1 - \sqrt{1-x} > \frac{x}{2}$$

This result already shows that list decoding can outperform unique decoding: Indeed, as argued in the last chapter, unique decoding is possible only if $p < \frac{\delta}{2} \leq \frac{1-R}{2}$.
However, we can tolerate up to $J_q(\delta) > \frac{\delta}{2}$ fraction of errors in the list-decoding paradigm [13], and thus we have outperformed the unique decoding bound already!
Now that we have outperformed the unique decoding barrier, we would like to know how far we can go. Fortunately, an analog of Shannon's theorem tells us that.

Theorem 7.3 (List Decoding Capacity)**.** Let $q \geq 2$ be an integer, let $\rho \in \left(0, 1 - \frac{1}{q}\right)$, and let $\varepsilon > 0$. Then for large enough $n \in \mathbb{N}$, we have that:

1. For any $L \in \mathbb{N}$, if $R \leq 1 - H_q(\rho) - \frac{1}{L}$, there exists a (ρ, L)-list decodable code with rate R.
2. If $R \leq 1 - H_q(\rho) + \varepsilon$, every (ρ, L)-list decodable code with rate R has $L \geq q^{\Omega(\varepsilon n)}$.

Perhaps unsurprisingly, the proof of this theorem exploits the probabilistic method, in a manner not dissimilar to the technique used in the proof of Shannon's theorem.

Proof. The proofs for both parts go as follows:

1. Consider any $k \in \mathbb{N}$ such that $k \leq \left(1 - H_q(\rho) - \frac{1}{L}\right) n$. Let $C \subseteq \mathbb{F}_q^n$ be a random code of dimension k, ie:- pick q^k elements from $\mathbb{F}_q^n$, uniformly, without replacement.
   Now, given any $y \in \mathbb{F}_q^n$, and any $c_0, \ldots, c_L \in \mathbb{F}_q^n$, we say that the tuple $(y, c_0, \ldots, c_L)$ constitutes a "bad event" if $c_i \in B(y, \rho n)$ for all $i \in \{0, \ldots, L\}$. Note that C is (ρ, L)-list decodable if and only if there are no bad events.
   Now,
   $$\Pr\left(\bigwedge_{i=0}^{L} (c_i \in B(y, \rho n))\right) = \frac{\binom{|B(y,\rho n)|}{L+1}}{\binom{q^n}{L+1}} \leq \left(\frac{|B(y, \rho n)|}{q^n}\right)^{L+1}$$
   where the last inequality follows from the fact that $\frac{\binom{a}{k}}{\binom{b}{k}} = \frac{a(a-1)\cdots(a-k+1)}{b(b-1)\cdots(b-k+1)} \leq \left(\frac{a}{b}\right)^k$ if $a \leq b$, since if $a \leq b$, then $\frac{a-r}{b-r} < \frac{a}{b}$ for any $0 < r < a$.
   But by Theorem A.10,
   $$\left(\frac{|B(y, \rho n)|}{q^n}\right) \leq q^{-n(1-H_q(\rho))}$$
   Thus,
   $$\Pr(\text{A bad event occurs}) = \Pr\left(\bigcup_{\substack{\{c_0,\ldots,c_L\} \subseteq C \\ y \in \mathbb{F}_q^n}} \bigwedge_{i=0}^{L} (c_i \in B(y, \rho n))\right) \leq \sum_{\substack{\{c_0,\ldots,c_L\} \subseteq C \\ y \in \mathbb{F}_q^n}} \Pr\left(\bigwedge_{i=0}^{L} (c_i \in B(y, \rho n))\right)$$
   $$\leq q^n \binom{q^k}{L+1} \left(\frac{|B(y, \rho n)|}{q^n}\right)^{L+1} \leq q^n \binom{q^k}{L+1} \left(q^{-n(1-H_q(\rho))}\right)^{L+1} \leq q^n q^{k(L+1)} \left(q^{-n(1-H_q(\rho))}\right)^{L+1}$$
   $$= q^{n + nR(L+1) - n(L+1)(1-H_q(\rho))} < 1$$
   where the last inequality follows from the fact that $R \leq 1 - H_q(\rho) - \frac{1}{L}$.
   Since for a random code, the probability of a bad event occurring is less than 1, we get that there must be some code for which the bad event doesn't occur, as desired.

[13] a small technicality here: As we remarked earlier, we require L to be polynomial, when we say that some code is (ρ, L)-decodable. Now, the Johnson bound gives us $L = q\delta n^2$. Thus, $L = \text{poly}(n)$ if and only if $q = \text{poly}(n)$. However, that is hardly a concern, as a majority of codes studied in theory and used in practice satisfy $q = \text{poly}(n)$.

2. Let $C = (n, k)_q$ be *any* code. Pick $y \in \mathbb{F}_q^n$ uniformly randomly. Fix any $c \in C$. Then

$$\Pr(c \in B(y, \rho n)) = \Pr(y \in B(c, \rho n)) = \frac{|B(c, \rho n)|}{q^n} \geq q^{-n(1-H_q(\rho))-o(n)}$$

where the second equality follows from the fact that y is uniformly random on $\mathbb{F}_q^n$, and the last inequality follows from Theorem A.10.
Thus

$$\mathbb{E}[|C \cap B(y, \rho n)|] = \sum_{c \in C} \Pr(c \in B(y, \rho n)) \geq |C| \cdot q^{-n(1-H_q(\rho))-o(n)} = q^{n(H_q(\rho)-(1-R)-o(1))}$$

Substituting $R = 1 - H_q(\rho) + \varepsilon$, yields $q^{n(H_q(\rho)-(1-R)-o(1))} = q^{n(\varepsilon - o(1))} = q^{\Omega(\varepsilon n)}$.
Since $\mathbb{E}[|C \cap B(y, \rho n)|] \geq q^{\Omega(\varepsilon n)}$, there exists some $y^* \in \mathbb{F}_q^n$ such that $|C \cap B(y^*, \rho n)| \geq q^{\Omega(\varepsilon n)}$.
Consequently, if C is (ρ, L)-list decodable, then $L \geq q^{\Omega(\varepsilon n)}$, as demonstrated by y^*.

■

7.2. Conclusion

We thus got a brief overview of the notion of list decoding and saw that it outperforms unique decoding. We also saw what the limits of list decoding were: Once again, note that we didn't give explicit codes satisfying that bound (though there do exist such codes).
Finally, note that we haven't seen any explicit list decoding algorithms. We shall see some soon for the specific case of Reed-Solomon codes.

§8. Efficient Decoding of Reed Solomon Codes

Reed-Solomon codes are one of the crown jewels of coding theory: They pop up in unexpected places and have a wide variety of applications.
It is thus worth exploring the decoding of Reed Solomon codes in more detail, and not just as a part of decoding in general.

8.1. Unique Decoding: Berlekamp-Welch Algorithm

Notation alert: Unlike in the chapter for Shannon's theorem, for this chapter e shall denote an integer, not an element of $\{0,1\}^n$.
Consider a $[n, k, n-k+1]_q$ RS code with evaluation points $(\alpha_1, \ldots, \alpha_n)$.
Now, suppose we receive some $y \in \mathbb{F}_q^n$ which is guaranteed to be within $\frac{n-k}{2}$ distance of some RS codeword. Then by Theorem 2.1, y can be decoded uniquely.
By the naive MLD algorithm, we would have to test q^k codewords to find the nearest one, which is an exponential time algorithm.
We shall now focus on finding a polynomial time algorithm for error correction of RS codes.
Note that finding the codeword '$\mathbf{m}$' nearest to y is equivalent to finding the corresponding polynomial $f_{\mathbf{m}}(X)$, so we shall focus on that now.
Thus our problem is:

Problem. Given $y \in \mathbb{F}_q^n$, a $[n,k]_q$ RS code with n distinct evaluation points $(\alpha_1, \ldots, \alpha_n)$, and an $e \leq \frac{n-k}{2}$, calculate a polynomial $p(X)$ of degree $\leq k-1$ such that the distance between the message encoded by $p(X)$ and some RS codeword is $\leq e$.

Remark. The fact that the number of errors has been given to us may trouble some readers, since in real life the exact number of errors in the received codeword is usually not known.
However, if we can find a polynomial (in n) time algorithm for any permissible e, we can simply run that algorithm over all possible values of e in $\{1, \ldots, \lfloor \frac{d-k}{2} \rfloor\}$.
Also, note that the polynomial $p(X)$, if it exists, is unique, so we needn't worry about getting different answers for different values of e. When e is smaller than the actual number of errors, our algorithm must not give any answer (ie:- our algorithm should throw an error), while for any e greater than or equal to the actual number of errors, our algorithm should give us the correct answer.
Thus, if we start iterating for e in $\{1, \ldots, \lfloor \frac{d-k}{2} \rfloor\}$, we will get errors for initial values of e, and then when we arrive at the correct e, we will receive our answer $p(X)$, at which point we can stop.
Thus WLOG we can assume that we know e.

8.1.1. Motivation for the Berlekamp-Welch Algorithm

Before describing the Berlekamp-Welch algorithm for decoding noisy vectors, we first set up some motivation for it.
Suppose we knew the polynomial $p(X)$: Then we could calculate for which $i \in [n]$, $y_i \neq p(\alpha_i)$. These are precisely the indices for which an error has occurred. In particular, we could encode these indices into an *error locator* polynomial 'E', ie:-

$$E(X) := \prod_{i \in [n] : y_i \neq p(\alpha_i)} (X - \alpha_i)$$

Now, note that $y_i E(\alpha_i) = p(\alpha_i) E(\alpha_i)$ for *every* $i \in [n]$: Indeed, if $y_i = p(\alpha_i)$, then the equality holds trivially. If $y_i \neq p(\alpha_i)$, then $E(\alpha_i) = 0$, and then also equality holds.
Thus, if we define the polynomial $N(X) := p(X)E(X)$, then $N(\alpha_i) = y_i p(\alpha_i)$ for every $i \in [n]$.
Consequently, if we knew $N(X)$ and $E(X)$, where $N(X)$ satisfies the aforementioned constraint, then we could calculate $p(X)$ by $\frac{N(X)}{E(X)}$.
Now, the sketch above can become a proper algorithm only if two of our wishes are satisfied: We somehow find a way of calculating $N(X)$ and $E(X)$ given just the data in Section 8.1. We then certify that these $N(X)$ and $E(X)$ will give

rise to a unique $p(X)$, ie:- we won't have a situation where we have (N_1, E_1) and (N_2, E_2) where $N_j(\alpha_i) = y_i E_j(\alpha_i)$ for all $i \in [n], j \in [2]$, yet $\frac{N_1}{E_1} \neq \frac{N_2}{E_2}$.
We now prove that our wishes are indeed true.

Lemma 8.1. Suppose we have polynomials $(N_1(X), E_1(X))$ and $(N_2(X), E_2(X))$ where $N_j(\alpha_i) = y_i E_j(\alpha_i)$ for all $i \in [n], j \in [2]$, and where $\deg(E_1) = \deg(E_2) = e$, and $\deg(N_1), \deg(N_2) \leq e + k - 1$.
Then $N_1(X)E_2(X) = N_2(X)E_1(X)$.

Proof. Consider the polynomial $R(X) := N_1(X)E_2(X) - N_2(X)E_1(X)$. Then $\deg(R) \leq \max(\deg(N_1) + \deg(E_2), \deg(N_2) + \deg(E_1)) \leq 2e + k - 1$. Since $e \leq \frac{n-k}{2}$, $\deg(R) \leq n - 1$.
But $R(\alpha_i) = 0$ for every $i \in [n]$. Since a non-zero polynomial of degree t over a field can have at most t roots, we get that R must be the zero polynomial, implying that $N_1(X)E_2(X) = N_2(X)E_1(X)$, as desired. ■

Now, note that $N(X), E(X)$ are our creations: Thus one may worry that there can be a situation where there exists some $p(X)$ corresponding to a message $\mathbf{m}$ at a distance of at most $\frac{n-k}{2}$ from our received codeword y, yet there don't exist N, E satisfying the requisite constraints, in which case our algorithm may fail to find p.
Fortunately, we show that that is not the case.

Lemma 8.2. Suppose y is the received word at a distance $e \leq \frac{n-k}{2}$ away from some codeword. Then there exist $N^*(X), E^*(X)$ satisfying $N^*(X) = E^*(X)p(X)$, $\deg(E^*) = e, \deg(N^*) \leq e + k - 1$, and $N^*(\alpha_i) = y_i E^*(\alpha_i)$ for every $i \in [n]$.

Proof. One can verify that

$$E^*(X) := \prod_{i : y_i \neq p(\alpha_i)} (X - \alpha_i)$$

$$N^*(X) := E^*(X)p(X)$$

work. ■

We are now ready to state the Berlekamp-Welch algorithm. We explain below how the constraints $y_i E(\alpha_i) = N(\alpha_i)$

Algorithm 2: Berlekamp-Welch algorithm

Data: $n \geq k \geq 1, e \leq \frac{n-k}{2}, y, (\alpha_1, \ldots, \alpha_n) \in \mathbb{F}_q^n$,
Result: $p(X)$ of degree $\leq k - 1$ or FAIL
Compute $E(X), N(X)$ such that $y_i E(\alpha_i) = N(\alpha_i)$ for every $i \in [n]$;
if such E, N don't exist **then**
  return FAIL
if $E(X) \nmid N(X)$ **then**
  return FAIL
$p(X) \leftarrow \frac{N(X)}{E(X)}$;
Let $\mathbf{m}$ be message corresponding to $p(X)$;
if $\Delta(\mathbf{m}, y) > e$ **then**
  return FAIL
return $p(X)$

for every $i \in [n]$ are satisfied.
Now, WLOG E can be assumed to be a monic polynomial, and thus $E(X) = X^e + \lambda_1 X^{e-1} + \ldots + \lambda_e$. Similarly, set

$N(X) = \mu_0 X^{e+k-1} + \mu_1 X^{e+k-2} + \ldots + \mu_{e+k-1}$.
Then $y_i E(\alpha_i) = N(\alpha_i)$ is a set of n linear equations on $2e + k - 1 \leq n - 1$ variables.
We can solve this system of linear equations by Gaussian elimination in $\mathcal{O}(n^3)$ time: If there is no solution to this system, that means that the number of errors is greater than e, and our algorithm returns 'FAIL'.
If there is any solution, then (N^*, E^*) (as defined in Lemma 8.2) are a solution, and their ratio is the correct polynomial $p(X)$. Thus any solution of our linear system yields the same $p(X)$ by Lemma 8.1.
Furthermore, note that this algorithm is polynomial time since both Gaussian elimination and polynomial division can be done in $\text{poly}(n)$ time (In fact, $\mathcal{O}(n^3)$ time).

Theorem 8.3. Error correction (if possible) of a $[n, k]$ RS code can be done in $\mathcal{O}(en^3)$ time, where $e \leq \frac{n-k}{2}$ is the number of errors in the transmitted codeword.

8.2. List Decoding of Reed-Solomon Codes

We present an algorithm here for list decoding Reed-Solomon codes, up to an extent allowed by the Johnson bound (Lemma 7.1).
By Lemma 7.2, $J_q(\delta) \geq 1 - \sqrt{1-\delta} \geq 1 - \sqrt{R}$. The algorithm described below list-decodes RS codes up to $1 - \sqrt{R}$-fraction of errors.
Before that, we introduce some mathematical notions.

Lemma 8.4 (Bivariate Polynomials can be efficiently factorized)**.** Given a bivariate polynomial $Q(X, Y) \in \mathbb{F}_q[X, Y]$ of degree n, $Q(X, Y)$ can be factorized in $\text{poly}(n)$ time.

Proof. Refer [Kal85]. ■

Thus, from now on, we shall assume that we have all our bivariate polynomials in factorized form.
We also generalize our notion of degree.

Definition 8.1. Consider a bivariate polynomial $Q(X, Y) \in \mathbb{F}_q[X, Y]$. The (a, b)-degree of Q is said to be the degree of the (univariate) polynomial $Q(X^a, X^b)$. We shall denote the (a, b)-degree of a polynomial as $\deg_{(a,b)}(Q)$.

Remark. Note that $\deg(Q) = \deg_{(1,1)}(Q)$, $\deg_X(Q) = \deg(Q(X, 1)) = \deg_{(1,0)}(Q)$.
A very easy lemma which follows directly from the definition goes as follows.

Lemma 8.5. Let $Q(X, Y)$ be a bivariate polynomial such that $\deg_{(1,w)}(Q) = D$. Let $P(X)$ be a polynomial of degree $\leq w$. Then $\deg(Q(X, P(X))) \leq D$.

Finally, we define the notion of multiplicity for bivariate polynomials.

Definition 8.2. Note that $Q(X, Y)$ has r roots at $(0, 0)$ if $Q(X, Y)$ doesn't have any monomial with degree $\leq r - 1$.
We alternatively say that $(0, 0)$ is a root of Q with multiplicity r.
If (α, β) is a root of some $Q(X, Y)$, the multiplicity of (α, β) is defined to be the multiplicity of $(0, 0)$ as a root of $Q(X - \alpha, Y - \beta)$.

Before we formally state our list decoding algorithm, we first give an overview of it's structure:

1. *Interpolation step*: In this step, we calculate a bivariate polynomial $Q(X,Y)$ such that $Q(\alpha_i, y_i) = 0$ for all $i \in [n]$. In the Berlekamp-Welch algorithm, we had $Q(X,Y) = YE(X) - N(X)$. As in the Berlekamp-Welch algorithm, this Q will be calculated by solving a system of linear equations (through Gaussian elimination). As it turns out, if the number of variables in this linear system is greater than the number of constraints, then we are able to find a non-trivial Q [14]. Note that in the Berlekamp-Welch algorithm, the situation was opposite: There we had more constraints than variables.

2. *Root Finding Step*: In this step, we calculate all factors of $Q(X,Y)$ (thanks to Lemma 8.4) of the form $Y - P(X)$, and if $P(X)$ is close enough to our message, we output it. Now, in order to ensure that all eligible P's show up in the factors of Q, we will need Q to have some special properties. For example, in the Berlekamp-Welch algorithm, we required that $\deg(E) = e, \deg(N) \leq e + k - 1$. We shall see soon what impositions on Q are required in order to enable list decoding.

We are now ready to state our list decoding algorithm.
There is a lot to unpack in this algorithm. We go about it one by one.

Algorithm 3: List-decoding algorithm for RS codes

Data: $n \geq k \geq 1, e = n - t, y = (y_1, \ldots, y_n), (\alpha_1, \ldots, \alpha_n) \in \mathbb{F}_q^n$,
Result: (Possibly empty) list of polynomials $p(X)$ of degree $\leq k-1$ or FAIL

1 $r \leftarrow 2(k-1)n$;
2 $D \leftarrow \left\lceil \sqrt{(k-1)nr(r-1)} \right\rceil$;
3 Compute non-zero $Q(X,Y)$ such that $\deg_{(1,k-1)}(Q) \leq D$ and $Q(\alpha_i, y_i) = 0$ with multiplicity r for every $i \in [n]$;
4 **if** such Q doesn't exist **then**
5 **return** FAIL
6 $\mathbb{L} \leftarrow \varnothing$;
7 **for** every factor $Y - P(X)$ of $Q(X,Y)$ **do**
8 Let $\mathbf{m}$ be message corresponding to $p(X)$;
9 **if** $\Delta(\mathbf{m}, y) \leq e$ and $\deg(p) \leq k-1$ **then**
10 Add $p(X)$ to $\mathbb{L}$
11 **return** $\mathbb{L}$

8.2.1. Computation of bivariate polynomials with given degree and multiplicity conditions

Here we will justify Line 3 of Algorithm 3.

Lemma 8.6. Let $\mathscr{N} := \left|\{(i,j) : i,j \geq 0, i + (k-1)j \leq D\}\right|$. Then $\mathscr{N} \geq \frac{D(D+2)}{2(k-1)}$.

Proof. Set $\ell := \left\lfloor \frac{D}{k-1} \right\rfloor$.
Note that

$$\mathscr{N} := \sum_{j=0}^{\ell} \sum_{i=0}^{D-(k-1)j} 1 = \sum_{j=0}^{\ell} (D - (k-1)j + 1) = \frac{1+\ell}{2}(2D + 2 - (k-1)\ell) \geq \frac{1+\ell}{2}(D+2) \geq \frac{D(D+2)}{2(k-1)}$$

as desired. ∎

[14] this fact is not immediately apparent; it requires proof, which we won't give

Lemma 8.7. We can compute a non-zero bivariate polynomial $Q(X,Y) \in \mathbb{F}_q[X,Y]$ such that $\deg_{(1,k-1)}(Q) \leq D$ and $Q(\alpha_t, y_t) = 0$ with multiplicity r for every $t \in [n]$, in $\text{poly}(n)$ time, provided such a polynomial exists.

Proof. Let

$$Q(X,Y) = \sum_{i+(k-1)j \leq D} c_{i,j} X^i Y^j$$

By Lemma 8.6, Q has $\geq \frac{D(D+2)}{2(k-1)}$ coefficients.
Define

$$Q_{\alpha,\beta}(X,Y) = Q(X+\alpha, Y+\beta) = \sum_{i+(k-1)j \leq D} c_{i,j}^{\alpha,\beta} X^i Y^j$$

Now, by comparing coefficients we get that

$$c_{i,j}^{\alpha,\beta} = \sum_{i' \geq i, j' \geq j} c_{i',j'} \binom{i'}{i} \binom{j'}{j} \alpha^i \beta^j$$

Now, fix $\alpha = \alpha_t, \beta = y_t$ for some $t \in [n]$. Then $Q(X+\alpha, Y+\beta)$ has a root at $(0,0)$ of multiplicity r, implying that it has no monomials of degree $\leq r-1$, further implying that $c_{i,j}^{\alpha,\beta} = 0$ for all $(i,j) \in \mathcal{I} := \{(i,j) : i,j \geq 0, i+j \leq r-1\}$. Now, note that the equation $c_{i,j}^{\alpha,\beta} = 0$ translates into a linear equation on the coefficients of Q. Also, note that $|\mathcal{I}| = \binom{r+1}{2}$. Thus, collecting all the equations of the form $c_{i,j}^{\alpha,\beta} = 0$ for every permissible i,j, and every $t \in [n]$, we have a total of $n\binom{r+1}{2}$ different linear equations on $c_{\cdot,\cdot}$'s.
Finally, note that for $D = \left\lceil \sqrt{(k-1)nr(r-1)} \right\rceil$, $\frac{D(D+2)}{2(k-1)} > n\binom{r+1}{2}$. Thus variables outnumber the constraints, and thus we can find our desired Q in polynomial time, by the discussion above for the "interpolation step". ■

8.2.2. Correctness of Root Finding Step

Suppose we have found a Q. We now have to argue that Lines 7-10 will give us the correct list $\mathbb{L}$. To do so, we go through the following chain of lemmata first.

Lemma 8.8. Let Q be a non-zero bivariate polynomial satisfying the conditions in Line 3 of Algorithm 3. Suppose $(Y - P(X)) \Big| Q(X,Y)$. Define $R(X) := Q(X, P(X))$. Suppose for some i we have $y_i = P(\alpha_i)$. Then $(X - \alpha_i)^r \Big| R(X)$.

Proof. Define $P_{\alpha_i,y_i}(X) := P(X+\alpha_i) - y_i, Q_{\alpha_i,y_i}(X,Y) := Q(X+\alpha_i, Y+y_i), R_{\alpha_i}(X) := R(X+\alpha_i)$. Then

$$R_{\alpha_i}(X) = R(X+\alpha_i) = Q(X+\alpha_i, P(X+\alpha_i)) = Q(X+\alpha_i, P_{\alpha_i,y_i}(X) + y_i) = Q_{\alpha_i,y_i}(X, P_{\alpha_i,y_i}(X))$$

Now, since $P(\alpha_i) = y_i$, $P_{\alpha_i,y_i}(0) = 0$, and thus $P_{\alpha_i,y_i}(X) = Xg(X)$, where g is a polynomial of degree at most $k-2$ [15].
Thus we can write

$$R_{\alpha_i}(X) = Q_{\alpha_i,y_i}(X, P_{\alpha_i,y_i}(X)) = \sum_{i',j'} c_{i',j'}^{\alpha_i,y_i} X^{i'} P_{\alpha_i,y_i}(X)^{j'} = \sum_{i',j'} c_{i',j'}^{\alpha_i,y_i} X^{i'+j'} g(X)^{j'}$$

[15] note that in the algorithm we only care about those polynomials p which are of degree at most $k-1$, that's why this assumption is justified

Now, recall that $Q_{\alpha_i,y_i}(X,Y)$ had no monomial of degree $< r$ since (α_i, y_i) was a root of $Q(X,Y)$ with multiplicity r. Consequently, for every i', j' such that $c_{i',j'}^{\alpha_i,y_i} \neq 0$, we must have $i' + j' \geq r$. But that implies that $R_{\alpha_i}(X)$ has no monomial of degree $< r$, and consequently, $X^r \Big| R_{\alpha_i}(X)$. Now, it is easy to see that $(X-\alpha_i)^r \Big| R(X) \iff X^r \Big| R_{\alpha_i}(X)$, from which the lemma follows. ■

Lemma 8.9. Let Q be a non-zero bivariate polynomial satisfying the conditions in Line 3 of Algorithm 3. Let $P(X)$ be a polynomial of degree $\leq k-1$ such that $P(\alpha_i, y_i) = 0$ for $\geq t$ values of i in $[n]$. If $t > \frac{D}{r}$, then $(Y - P(X)) \Big| Q(X,Y)$.

Proof. Let $R(X) := Q(X, P(X))$ as usual. By Lemma 8.5, $\deg(R) \leq D$. By Lemma 8.8, $R(X)$ has atleast $t \cdot r$ roots (counted with multiplicity). But $t \cdot r > D$. Since a non-zero polynomial of degree w over a field can't have more than w roots, we get that $R(X) \equiv 0$, and the lemma follows. ■

Now, note that to ensure the correctness of the root finding step, we have to ensure that if $P(X)$ is a polynomial which agrees with y in $\geq n - e = t$ entries, then $Y - P(X)$ divides $Q(X,Y)$. But this is precisely what Lemma 8.9 shows, and thus, every "correct" polynomial appears as a factor of Q and thus our list $\mathbb{L}$ is correct.
Now, note that we have already chosen $D = \left\lceil \sqrt{(k-1)nr(r-1)} \right\rceil$. Furthermore, recall that we wanted to this list decoding algorithm to work for $1 - \sqrt{R}$ fraction of errors: Since the number of errors $e = n - t$, we want $t \sim \sqrt{kn}$. Finally, by the lemma above, we also want $t > \frac{D}{r}$ so that our root finding step correctly calculates $\mathbb{L}$.
One can thus verify that choosing $t = \left\lceil \sqrt{(k-1)n\left(1 - \frac{1}{r}\right)} \right\rceil$ makes $t > \frac{D}{r}$. Finally, in order to ensure t is as close to $\sqrt{kn}$ as possible, we set $r = 2(k-1)n$. Then note that

$$t = \left\lceil \sqrt{(k-1)n\left(1 - \frac{1}{r}\right)} \right\rceil = \left\lceil \sqrt{(k-1)n - \frac{1}{2}} \right\rceil > \left\lceil \sqrt{(k-1)n} \right\rceil$$

where the last inequality follows since t is an integer.

8.3. A Recapitulation of the List-Decoding algorithm

We first calculate a bivariate polynomial Q interpolating the points (α_i, y_i). We also require Q to satisfy some bounded degree requirements, and some multiplicty requirements. While this may look strange at first, by now, I hope the reader realizes that these properties were required to ensure the correctness of the following "root-finding" part of the algorithm.
Indeed, once we find such a Q, it can be shown that every desired message occurs as a factor of Q, and consequently, by Lemma 8.4, all elements of our list can be calculated in polynomial time, as desired.

8.4. Conclusion

As promised, we saw decoding and list-decoding algorithms for Reed-Solomon codes. Although at first these algorithms might seem very narrow in their utility, they pop up widely in different areas and contexts: For example, [BRSV17] use the Berlekamp-Welch algorithm in a crucial way to prove some very important results in fine-grained cryptography. This shouldn't be too surprising given the fact that Reed-Solomon codes occur widely in theoretical science, and thus algorithms to decode it can never be too far away.

References

[ABI86] Noga Alon, László Babai, and Alon Itai. A fast and simple randomized parallel algorithm for the maximal independent set problem. *J. Algorithms*, 7(4):567–583, dec 1986. doi:10.1016/0196-6774(86)90019-2.

[AS98] Sanjeev Arora and Shmuel Safra. Probabilistic checking of proofs: A new characterization of np. *J. ACM*, 45(1):70–122, jan 1998. doi:10.1145/273865.273901.

[BRSV17] Marshall Ball, Alon Rosen, Manuel Sabin, and Prashant Nalini Vasudevan. Average-case fine-grained hardness. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 483–496, New York, NY, USA, 2017. Association for Computing Machinery. doi:10.1145/3055399.3055466.

[For] David Forney. Introduction to finite fields. URL: http://web.stanford.edu/~marykw/classes/CS250_W19/readings/Forney_Introduction_to_Finite_Fields.pdf.

[GRS22] Venkatesan Guruswami, Atri Rudra, and Madhu Sudan. *Essential Coding Theory*. 2022. URL: https://cse.buffalo.edu/faculty/atri/courses/coding-theory/book/.

[Gur10] Venkatesan Guruswami. Introduction to coding theory, spring 2010. online, 2010. URL: http://www.cs.cmu.edu/~venkatg/teaching/codingtheory/.

[Had23] Hadamard code. Hadamard code, 2023. URL: https://en.wikipedia.org/wiki/Hadamard_code.

[Kal85] Erich Kaltofen. Polynomial-time reductions from multivariate to bi- and univariate integral polynomial factorization. *SIAM Journal on Computing*, 14(2):469–489, 1985. arXiv:https://doi.org/10.1137/0214035, doi:10.1137/0214035.

[Pan17] Omkant Pandey. Proof of gl theorem, 2017. URL: https://www3.cs.stonybrook.edu/~omkant/L06.pdf.

§A. Appendix

A.1. Some elementary facts about probability

We state here, without proof some elementary facts about probability theory, which nevertheless can be quite useful, especially in the analysis of randomized algorithms.

Lemma A.1 (The Union Bound). Let $\mathcal{E}_1, \mathcal{E}_2, \ldots, \mathcal{E}_n$ be events such that $\Pr(\overline{\mathcal{E}_i}) \leq p_i$ for every $i \in [n]$, where $p_i \in [0,1]$. Then

$$\Pr\left(\bigwedge_{i=1}^{n} \mathcal{E}_i\right) \geq 1 - \sum_{i=1}^{n} p_i$$

Lemma A.2. Let $\mathbb{F}$ be a finite field.
Let $G \in \mathbb{F}^{a \times b}$ be a uniformly randomly chosen matrix, ie:- the probability that any matrix in $\mathbb{F}^{a \times b}$ gets chosen is $\frac{1}{|\mathbb{F}|^{ab}}$. Then for any $x \in \mathbb{F}^b \setminus \{0^b\}$, Gx is uniformly distributed in $\mathbb{F}^a$.

Theorem A.3 (Chernoff Bound). Let $X_1, \ldots, X_m$ be i.i.d Bernoulli random variables. Let $X = \sum_{i=1}^{m} X_i$. Then

$$\Pr(|X - \mathbb{E}[X]| > \varepsilon \mathbb{E}[X]) < 2e^{-\frac{\varepsilon^2 \mathbb{E}[X]}{3}}$$

$$\Pr(|X - \mathbb{E}[X]| > \varepsilon m) < 2e^{-\frac{\varepsilon^2 m}{2}}$$

A.2. The Entropy Function

Definition A.1. Let $q \geq 2$ be an integer, and let $x \in [0,1]$ be a real number. Then the q-ary function is defined as

$$H_q(x) := x \log_q(q-1) - x \log_q(x) - (1-x) \log_q(1-x)$$

Remark. A few points are in order:

1. When we use the above definition with $q = 2$, we shall just write $H(x)$ instead of $H_2(x)$ for notational clarity. Furthermore, note that $H(x) = -(x \log_2(x) + (1-x) \log_2(1-x))$.
2. For evaluating H_q at 0 and 1, we replace the functions $x \log_q(x)$ and $(1-x) \log_q(1-x)$ by their respective limits, ie:- for evaluating H_q at 0 we replace "$0 \cdot \log_q(0)$" by $\lim_{x \searrow 0} x \log_q(x) = 0$, and for evaluating H_q at 1 we replace "$(1-1) \cdot \log_q(1-1)$" by $\lim_{x \nearrow 1} (1-x) \log_q(1-x) = 0$.
3. Note that $H_q(x) \in [0,1]$ for every $x \in [0,1]$.
4. $H_q(x)$ is an increasing function on $\left(0, 1 - \frac{1}{q}\right)$, attains maxima at $1 - \frac{1}{q}$, and is decreasing after that. Furthermore, $H_q(0) = 0, H_q\left(1 - \frac{1}{q}\right) = 1$. Consequently, $H_q(x)$ is a bijective function on $\left[0, 1 - \frac{1}{q}\right]$, which allows us to define its inverse $H_q^{-1} : [0,1] \mapsto \left[0, 1 - \frac{1}{q}\right]$.

The entropy function pops up quite often in various locations, which makes it necessary to have various mathematical inequalities and estimates to deal with it.
However, most of the proofs of the lemmata given below are somewhat tedious, yet a fairly routine application of basic analysis. Consequently, we just state them without proof, for reference.

Lemma A.4. For small enough $\varepsilon > 0$,

$$H_q(p) \leq p + \varepsilon, \ \forall p \in \left(0, 1 - \frac{1}{q}\right] \iff q = 2^{\Omega(\frac{1}{\varepsilon})}$$

Lemma A.5. Let $q \leq 2$, and let $p \in \left[0, 1 - \frac{1}{q}\right]$. Then $H_q(p) \geq H_{q^m}(p)$ for every

$$m \geq 1 + (q-1)\log_q\left(\frac{q}{q-1}\right)$$

Corollary A.6. For every $q \geq 3, m \geq 2$ and $p \in \left[0, 1 - \frac{1}{q}\right]$ we have $H_q(p) \geq H_{q^m}(p)$.

Lemma A.7. For any $q \geq 2$, we have that for small enough $\varepsilon \geq 0$,

$$H_q\left(1 - \frac{1}{q} - \varepsilon\right) \leq 1 - \frac{q^2}{4(q-1)\ln q}\varepsilon^2$$

Remark. For $q \leq 9$, the above inequality is true for every $\varepsilon \in \left[0, 1 - \frac{1}{q}\right]$. For $q \geq 10$, if ε gets very close to $1 - \frac{1}{q}$, the inequality fails.

Lemma A.8. For small enough $\varepsilon > 0$,

$$H_q(\varepsilon) = \Theta\left(\varepsilon \log_q\left(\frac{1}{\varepsilon}\right)\right)$$

Remark. In fact, much more is true: $H_q(x) \geq x\log_q\left(\frac{1}{x}\right)$ for every $x \in [0,1]$. Furthermore, $\frac{H_q(x)}{x\log_q(\frac{1}{x})}$ is an increasing function of x such that $\lim_{x \searrow 0} \frac{H_q(x)}{x\log_q(\frac{1}{x})} = 1$.

Lemma A.9. For every $y \in \left[0, 1 - \frac{1}{q}\right]$, and for every small enough $\varepsilon > 0$, there exists a constant c_q that depends only on q, for which we have

$$H_q^{-1}(y - c_q\varepsilon^2) \geq H_q^{-1}(y) - \varepsilon$$

Remark. At the risk of repetition, note that c_q doesn't depend on y, or in other words, the same c_q works for every $y \in \left[0, 1 - \frac{1}{q}\right]$.

One of the most important reasons why the notion of q-ary entropy is important is because it is involved in very important bounds for the *volume of the Hamming ball*.

A.2.1. Volume of a Hamming Ball

Recall that the volume of a Hamming Ball of radius r, denoted $B_q(0^n, r)$ [16], was $\sum_{i=0}^{r} \binom{n}{i}(q-1)^i$. Then

Theorem A.10. Let $q \geq 2$ be an integer, and let $p \in [0, 1 - \frac{1}{q}]$ be a real number. Denote by $\mathrm{vol}_{q,n}(r)$ the volume of $B_q(0^n, r)$. Then $q^{H_q(p)n - o(n)} \leq \mathrm{vol}_{q,n}(pn) \leq q^{H_q(p)n}$.

Proof. Note that

$$1 = (p + (1-p))^n = \sum_{i=0}^{n} \binom{n}{i} p^i (1-p)^{n-i} \geq \sum_{i=0}^{pn} \binom{n}{i} p^i (1-p)^{n-i} = \sum_{i=0}^{pn} \binom{n}{i} (q-1)^i \left(\frac{p}{q-1}\right)^i (1-p)^{n-i} =$$
$$\sum_{i=0}^{pn} \binom{n}{i} (q-1)^i \left(\frac{p}{(q-1)(1-p)}\right)^i (1-p)^n \geq \sum_{i=0}^{pn} \binom{n}{i} (q-1)^i \left(\frac{p}{(q-1)(1-p)}\right)^{pn} (1-p)^n$$

where the last inequality follows from the fact that $\frac{p}{(q-1)(1-p)} < 1$, since $p \in [0, 1 - \frac{1}{q}]$.
Simplifying further yields

$$\sum_{i=0}^{pn} \binom{n}{i} (q-1)^i \left(\frac{p}{(q-1)(1-p)}\right)^{pn} (1-p)^n = \underbrace{\left(\sum_{i=0}^{pn} \binom{n}{i} (q-1)^i\right)}_{=\mathrm{vol}_{q,n}(pn)} \underbrace{\left(\frac{p}{q-1}\right)^{pn} (1-p)^{(1-p)n}}_{=q^{-H_q(p)n}}$$

Consequently $\mathrm{vol}_{q,n}(pn) \leq q^{H_q(p)n}$.
Finally, note that

$$\mathrm{vol}_{q,n}(pn) \geq \binom{n}{pn} (q-1)^{pn}$$

Now, note that, $\sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n+1}} < n! < \sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n}}$ for *every* $n \in \mathbb{N}$, and consequently,

$$\binom{n}{pn} = \frac{n!}{(pn)!((1-p)n)!} > \frac{\sqrt{2\pi n}\left(\frac{n}{e}\right)^n e^{\frac{1}{12n+1}}}{\sqrt{2\pi pn}\left(\frac{pn}{e}\right)^{pn} e^{\frac{1}{12pn}} \cdot \sqrt{2\pi(1-p)n}\left(\frac{(1-p)n}{e}\right)^{(1-p)n} e^{\frac{1}{12(1-p)n}}}$$
$$= \frac{1}{p^{pn}(1-p)^{(1-p)n}} \cdot \underbrace{\frac{1}{\sqrt{2\pi p(1-p)n}} \cdot \exp\left(\frac{1}{12n+1} - \frac{1}{12pn} - \frac{1}{12(1-p)n}\right)}_{=\ell(n)}$$

[16] strictly speaking $B_q(0^n, r)$ denotes the ball of radius r centered at the origin, but from the point of view of calculating the volume, the location of the center is unimportant

Note that $\ell(n) = e^{-o(n)} = q^{-o(n)}$, and consequently,

$$\mathrm{vol}_{q,n}(pn) \geq \underbrace{\frac{(q-1)^{pn}}{p^{pn}(1-p)^{(1-p)n}}}_{=q^{H_q(p)n}} q^{-o(n)} = q^{H_q(p)n-o(n)}$$

■

A.3. Miscellaneous

Lemma A.11. Let $v_1, v_2, \ldots, v_m \in \mathbb{R}^N \setminus \{0^N\}$.

1. If $\langle v_i, v_j \rangle \leq 0$ for every $1 \leq i < j \leq m$, then $m \leq 2N$.
2. If $v_i, i \in [m]$ are all unit vectors, such that $\langle v_i, v_j \rangle \leq -\varepsilon < 0$ for every $1 \leq i < j \leq m$, then $m \leq 1 + \frac{1}{\varepsilon}$.

Proof. The proofs go as follows:

1. We prove the statement by induction on N: The statement is trivially true for $N = 1$. Thus assume that the statement is true upto some $N < N_0 \geq 2$, and assume for the sake of contradiction that the statement fails for $N = N_0$, ie:- there exist $m = 2N_0 + 1$ vectors $v_1, v_2, \ldots, v_m$ such that $\langle v_i, v_j \rangle \leq 0$ for every $1 \leq i < j \leq m$.
   Note that rotating and scaling the vectors doesn't change their inner products, and thus WLOG assume that $v_m = (1, 0, \ldots, 0)$. Since $\langle v_i, v_m \rangle \leq 0$ for every $1 \leq i \leq m-1$, we get that the first coordinates of all vectors $v_i, 1 \leq i \leq m-1$, is ≤ 0. Further note that there can be at most one vector of the form $(-\alpha, 0, \ldots, 0), \alpha > 0$, among $\{v_1, \ldots, v_{m-1}\}$, since two such vectors will have a positive dot product.
   Now consider the last $N_0 - 1$ coordinates of $v_2, \ldots, v_{m-1}$ to get $m-2$ vectors $\{v_2', \ldots, v_{m-1}'\}$ in $\mathbb{R}^{N_0-1}$. Note that
   $$\langle v_i', v_j' \rangle = \langle v_i, v_j \rangle - v_{i1}v_{j1} \leq \langle v_i, v_j \rangle \leq 0, \ \forall i, j \in \{2, \ldots, m-1\}, i \neq j$$
   Thus we obtain $\geq 2N_0 + 1 - 2 = 2N_0 - 1$ vectors in $\mathbb{R}^{N_0-1}$, all at obtuse angles with each other, thus violating the induction hypothesis.
2. Note that
   $$0 \leq \left\| \sum_{i=1}^{m} v_i \right\|_2^2 = \sum_{i=1}^{m} \|v_i\|_2^2 + 2 \sum_{1 \leq i < j \leq m} \langle v_i, v_j \rangle \leq m - 2\binom{m}{2}\varepsilon \implies m - 2\binom{m}{2}\varepsilon \geq 0 \implies m \leq 1 + \frac{1}{\varepsilon}$$

■

Lemma A.12. For any non-empty $C \subseteq [q]^n$, there exists a function $f : C \mapsto \mathbb{R}^{nq}$ such that:

1. For every $c \in C$, $\|f(c)\| = 1$.
2. For every $c_1, c_2 \in C, c_1 \neq c_2$, $\langle f(c_1), f(c_2) \rangle = 1 - \frac{q\Delta(c_1,c_2)}{n(q-1)}$.

Proof. Define a function $\phi : [q] \mapsto \mathbb{R}^q$ where

$$\phi(i) := \left(\frac{1}{q}, \ldots, \underbrace{\frac{1-q}{q}}_{i^{\text{th}} \text{ position}}, \ldots, \frac{1}{q} \right)$$

We now define $f : [q]^n \mapsto \mathbb{R}^{qn}$, where

$$f(c) = f((c_1, c_2, \ldots, c_n)) := \sqrt{\frac{q}{n(q-1)}}(\phi(c_1), \ldots, \phi(c_n)) \in \mathbb{R}^{nq}$$

The properties mentioned can be now verified easily. ■