Error correcting codes: The classical and the quantum

Siddhant Midha
Department of Electrical Engineering
Indian Institute of Technology Bombay
siddhant-midha.github.io/

Autumn 2023

Last Updated: December 7, 2023

Abstract

This is a set of notes and reviews resulting from my study of the theory of classical and quantum error correcting codes. The idea is to compile my understanding in a coherent manner, as well as to create a document suitable for young researchers in quantum with the purpose of gaining a perspective on quantum error correction while reviewing the quintessential classical background. Further, this will also consist of reviews of contemporary research in quantum error correction broken down into suitable sections.

The theoretical formulations will stay the same after a while of making the notes, but this will be a running document wherein I update the research part as the said research progresses.

Contents

§0. Error correction – wer, was, warum und wie?

§1. Classical Error Correcting Codes

§§1.1. Linear Codes

§§1.2. Bounds on classical codes

Definition 1.1. Fix n and d. Define $A_q(n,d)$ as the maximum value of M such that a q-ary $[n,M,d]$ code exists.

Theorem 1.2 (The Singleton Bound)**.** For a $[n,k,d]$ linear code, one has that $d \leq n-k+1$. Thereby, $k \leq n-d+1$, and $A_q(n,d) \leq q^{n-d+1}$.

Proof. We have previously shown that for a $[n,k,d]$ linear code with the parity check matrix H of size $(n-k)\times n$, every $d-1$ columns of H are linearly independent, and there exists d columns which are linearly dependent. Thus, we must have that $d-1$ is equal to the column rank of H. Since we must have $\text{rank}(H) \leq n-k, n$. Thus, $d-1 \leq n-k$ and $d \leq n-k+1$. Now, note that the number of codewords $M = q^k$, from where $A_q(n,d) \leq q^{n-d+1}$. □

Interestingly, we can give a simpler proof without assuming a linear structure of the code. Considering a (n,M,d) code, we have that

$$d \leq n - log_q M + 1 \tag{1.1}$$

Proof. Let $l := \lceil \log_q M \rceil - 1$. Since $q^l < M$, there must be atleast two codewords that agree on the first l coordinates, as in total we have M codewords, but the spacae of the first l-coordinates is q^l dimensional. Thus, $n-d \geq l$ and we have that $d \leq n-l \leq n - \log_q M + 1$. □

Codes that achieve the singleton bound are called *Maximal Distance Separable* (MDS) codes. Some examples include,

1. The $(n,1,n)$ repetition code.
2. The $(n,n+1,2)$ simple parity check code.
3. **Reed-Solomon codes** over $\mathbb{F}_q$ with $q > n$. These are constructed as follows. Choose n *distibct* elements $\alpha_1,\ldots,\alpha_n \in \mathbb{F}_q$ (this is possible, why?), and construct the p.c. matrix as,

$$H := \begin{pmatrix} 1 & 1 & \cdots & 1 \\ \alpha_1 & \alpha_2 & \cdots & \alpha_n \\ \alpha_1^2 & \alpha_2^2 & \cdots & \alpha_n^2 \\ \vdots & \vdots & \vdots & \vdots \\ \alpha_1^{n-k-1} & \alpha_2^{n-k-1} & \cdots & \alpha_n^{n-k-1} \end{pmatrix}_{(n-k)\times n} \tag{1.2}$$

It is easy to prove that $d = n - k + 1$ here: We need to show that any $n - k$ columns are linearly independent, and there exists a set of $n - k + 1$ linearly dependent columns.

Proposition 1.3. Every RS code is MDS.

Proof. Notice that every $n-k$ by $n-k$ submatrix of H_{RS} is a Vandermonde matrix, which is non-singular. Thus, every set of $n - k$ columns of H is linearly independent. Moreover, since $rank(H) \leq \min\{n-k, n\}$ we have that there exists a set of $n - k$ independent columns. Thus, the RS code has distance $d = n - k + 1$, and is MDS. □

We will next talk about the Hamming bound, which is essentially a clever sphere backing bound on the space of received vectors.

Theorem 1.4 (The Hamming Bound)**.** We have,

$$A_q(n, d) \leq \frac{q^n}{\sum_{l=0}^{t} \binom{n}{l}(q-1)^l} \tag{1.3}$$

Proof. Let $c_1, \ldots, c_M$ denote the codewords. For all $i \in [M]$, define

$$B_i(r) := \{c \in \mathbb{F}_q^n \mid d(c_i, c) \leq r\} \tag{1.4}$$

Let $t := \lfloor (d-1)/2 \rfloor$. Note that, $B_i(t) \cap B_j(t) = \emptyset$ for $i \neq j$. If we let $B := \bigcup_{i \in [M]} B_i(t)$ then $c \in B \implies c \in \mathbb{F}^n$. Also, $|B| = \sum_i |B_i(t)|$, and $|B_i(t)| = \sum_{l=0}^{t} \binom{n}{l}(q-1)^l$ by simple counting arguments. Thus, we have $|\mathcal{C}| \times \sum_{l=0}^{t} \binom{n}{l}(q-1)^l \leq q^n$, thereby establishing the bound. □

A code which attains the Hamming bound is called a *perfect* code. An exhaustive list of perfect codes is,

1. The $[n, 1, n]$ repetition code over $\mathbb{F}_2$.
2. The $[n, n-m, 3]$ Hamming codes over $\mathbb{F}_q$.
3. THe $[23, 12, 7]$ Golay code over $\mathbb{F}_2$.
4. The $[11, 6, 5]$ Golay code over $\mathbb{F}_3$.

Let us now talk about a lower bound on $A_q(n, d)$. This is derived in a very similar way to the Hamming bound, but with a covering argument instead of a packing argument.

Theorem 1.5 (The Gilbert-Varshamov Bound)**.** We have,

$$A_q(n, d) \geq \frac{q^n}{\sum_{l=0}^{d-1} \binom{n}{l}(q-1)^l} \tag{1.5}$$

Proof. Fix n, d and let $\mathcal{C}_0$ be the code with the maximum M. We postulate that for every $x \in \mathbb{F}^n$, there must exist a $c \in \mathcal{C}_0$ s.t. $x \in B_c(d-1)$. This is easy to establish, as if there were a x not lying in any $B_c(d-1)$, we could just append it to the codespace and get a bigger set of codewords, thereby violating the assumption we started with. Now, if $B := \bigcup_c B_c(d-1)$ then $x \in \mathbb{F}^n \implies x \in B$. Thus, $|\mathbb{F}^n| = q^n \leq |B|$. Use a similar counting argument as the previous proof and the fact that $|\mathcal{C}_0| = A_q(n,d)$ and conclude. □

Combining the lower and upper bounds on $A_q(n,d)$, we have,

$$\boxed{\frac{q^n}{\sum_{l=0}^{d-1}\binom{n}{l}(q-1)^l} \leq A_q(n,d) \leq \frac{q^n}{\sum_{l=0}^{t}\binom{n}{l}(q-1)^l}} \tag{1.6}$$

In fact, we can also discuss a constructive version of the GV bound, which I call the Constructive GV.

Theorem 1.6 (Constructive GV)**.** If there exist $n, k, d, q \in$ satisfying

$$q^k < \frac{q^n}{\mathrm{Vol}_q(n-1, d-2)} \tag{1.7}$$

then there exists a $[n,k,d]$ linear code.

Proof. We attempt to construct a $(n-k) \times n$ p.c. matrix for the $[n,k,d]$ linear code. Start by the $I_{n-k \times n-k}$ identity matrix, and keep adding columns h_i. Suppose we have added the columns $h_1, h_2, \ldots, h_{l-1}$ and we need to add $h_l \in \mathbb{F}_q^{n-k}$ *such that* h_l is *not* a linear combination of any of $d-2$ columns from $h_1, \ldots, h_{l-1}$. Phrased differently, this condition reads,

$$h_l \notin \{[h_1, h_2, \ldots, h_{l-1}]x : x \in \mathbb{F}_q^{l-1},\ \mathrm{wt}(x) \leq d-2\} \tag{1.8}$$

The number of such columns, which is the number of ineligible vectors, is less than or equal to $\mathrm{Vol}_q(l-1, d-2)$[1]. As long as the set of all columns is bigger than the set of ineligible vectors at each stage l, we can continue the construction. This is guaranteed if $q^{n-k} > \mathrm{Vol}_q(n-1, d-2)$, which is equivalent to

$$q^k < \frac{q^n}{\mathrm{Vol}_q(n-1, d-2)} \tag{1.9}$$

as postulated. □

Let us now talk about an interesting bound derived specifically for linear codes, which deals with the probability of error over a BSC channel instead of the

[1] inequality instead of equality as some linear combinations might be the same vector

number of codewords. Recall that a linear code can be fully specified by the generator matrix, which in the standard form can be written as $G = [I|A]$ where I is $k \times k$ and A is $k \times (n-k)$. Now, working over $q = 2$, all entries of A are either 0 or 1. Thus, there are $2^{k(n-k)}$ linear codes, say given by $\mathcal{C}_i$. Let A_{ij} be the number of codewords in $\mathcal{C}_j$ with weight i. Then, if I choose the codewords randomly and send it over a BSC channel with p, the probability of error is

$$P(E) = \sum_j P(E|\mathcal{C}_j)P(\mathcal{C}_j) = 2^{-k(n-k)} \sum_j \sum_{i=1}^{n} A_{ij} p^i (1-p)^{n-i} = 2^{-k(n-k)} \sum_{i=1}^{n} p^i (1-p)^{n-i} \sum_j A_{ij} \tag{1.10}$$

Now, a non-zero codeword is contained in exactly $2^{(k-1)(n-k)}$ codes in this ensemble of codes, or in none of them. Thus, $\sum_j A_{ij} \leq 2^{(k-1)(n-k)} \times \binom{n}{i}$, and

$$P(E) \leq 2^{-(n-k)} \sum_i p^i (1-p)^{n-i} \binom{n}{i} = 2^{-(n-k)} \left(1 - (1-p)^n\right) \tag{1.11}$$

Thus we have the following results.

Theorem 1.7 (*Good linear codes exist*)**.** There exists a (n, k) linear code $\mathcal{C}$ over $\mathbb{F}_2$ such that the probability of error over a uncorrelated BSC(p) channel is upper bounded as,

$$\boxed{P_{\mathcal{C}}(E) \leq 2^{-(n-k)} \left(1 - (1-p)^n\right) \leq 2^{-(n-k)}} \tag{1.12}$$

This is *nice*, take $n - k \sim 20$ and see.

We will give another bound, without proof.

Theorem 1.8 (The Plotkin Bound)**.** We have that for an $[n, M, d]$ binary code

$$A_2(n, d) \leq \frac{2d}{2d - n} \tag{1.13}$$

for all $d > n/2$.

Proof. Arrange the M codewords row-wise to form a $M \times n$ matrix. Let this matrix be,

$$\text{MAT} = [X_1, X_2, \cdots, X_n] \tag{1.14}$$

where $X_i \in \mathbb{F}_q^M$. Denote the number of 1's in X_i as x_i. Then,

$$\sum_{i \neq j} d(c_i, c_j) = \sum_{k=1}^{n} \binom{x_i}{1} \binom{M - x_i}{1} = \sum_{k=1}^{n} x_k (M - x_k) \tag{1.15}$$

as we will have a unit contribution to the distance for every $1 - 0$ pair in every column which would represent two codewords being different at the index corr.

to the column index. But,

$$\binom{M}{2} d \leq \sum_{i \neq j} d(c_i, c_j) = \sum_{k=1}^{n} x_k(M - x_k) \leq \frac{nM^2}{4} \tag{1.16}$$

This gives us,

$$M \leq \frac{2d}{2d - n} \tag{1.17}$$

□

In fact,

Lemma 1.9. For a $[n, M, d]$ code over $\mathbb{F}_q$, if we let $\theta := 1 - 1/q$, then $d > \theta n$ implies,

$$A_q(n, d) \leq \frac{d}{d - \theta n} \tag{1.18}$$

Theorem 1.10 (The Johnson Bound)**.** Let $A(n, d, w)$ denote the maximum size of the n-length code with min. dist. d and *constant weight* w.

$$A_2(n, d, w) \leq \frac{2nd}{(n - 2w)^2 - n(n - 2d)} \tag{1.19}$$

Proof.

$$\binom{M}{2} d = \frac{M(M-1)}{2} \leq \sum_i x_i(M - x_i) = M^2 w - \sum_i x_i^2 \tag{1.20}$$

Also,

$$n \frac{1}{n} \sum_i x_i^2 \geq n(\frac{1}{n} \sum_i x_i)^2 = \frac{M^2 w^2}{n} \tag{1.21}$$

Leading to

$$\frac{M(M-1)}{2} \leq \frac{M^2 w(n-w)}{n} \implies M \leq \frac{nd}{nd - 2wn + 2w^2} = \frac{2nd}{(n-2w)^2 - n(n-2d)} \tag{1.22}$$

□

This is interesting because, the denomenator of the bound is positive when,

$$w < w^* := \frac{n - \sqrt{n^2 - 2nd}}{2} \tag{1.23}$$

When errors lie in radius $t \leq (d-1)/2$ we can do exact decoding. But if we choose radius $t = w^*$, the number of codewords in this ball would be $O(n)$, and not $O(\exp\{n\})$, so we can generalize unique decoding to this setting, wherein

we will have a polynomially sized list to decode in; this is called list decoding, and is used when unique decoding is not possible/feasible.

There can be other ways of characterizing good codes, for instance, in a more information theoretic fashion. Recall that the (information) rate of a code is $R := k/n$ and define the relative minimum distance of a code as $\delta := d/n$. We have that $R, \delta \in [0, 1]$ and a code is 'good' if both $R, \delta \approx 1$. With this in mind, we define a quantity,

$$\alpha_q(\delta) := \limsup_{n\to\infty} \frac{\log_q A_q(n, \lfloor \delta n \rfloor)}{n} \tag{1.24}$$

called the *asymptotic information rate*, which would be the highest information rate to which a sequence of codes with relative distance δ can converge. There exist asymptotic bounds concerning this quantity, which are sometimes easier to prove. Formally, we have,

Definition 1.11 (Asymptotically good codes)**.** An *asymptotically good code* is a code family that has an asymptotically positive normalized distance

$$\lim_{n\to\infty} \delta = \lim_{n\to\infty} \frac{d}{n} > 0 \tag{1.25}$$

and a positive rate

$$\lim_{n\to\infty} R(\delta) = \lim_{n\to\infty} \frac{log_q A_q(n, \delta n)}{n} \geq 0 \tag{1.26}$$

For the rest of the discussion in this lecture, we will take the case of $\mathbb{F}_2$ and find the relation between δ and $R(\delta)$.

Theorem 1.12 (Asymptotic singleton bound)**.** For any binary (n, d) code, the asymptotic rate $R(\delta)$ of the code is bounded by

$$\boxed{R(\delta) \leq 1 - \delta} \tag{1.27}$$

Proof. We start with the original singleton bound that gives us

$$log_2 A_2(n, \delta n) \leq n - d + 1,$$

on dividing by n and applying the asymptotic condition, we get

$$\lim_{n\to\infty} \frac{log_2 A_2(n, \delta n)}{n} \leq \lim_{n\to\infty} \frac{n - d + 1}{n},$$

which corresponds to the equation

$$R \leq 1 - \delta + o(1)$$

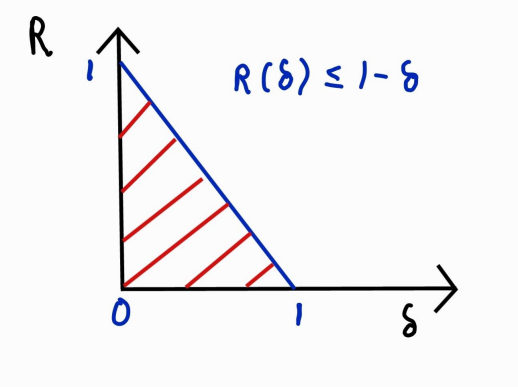


Figure 1: Rate vs δ for asymptotic singleton bound

□

Now, we look into the sphere packing (Hamming) bound for the asymptotic case. First, we start with the following definition-

Definition 1.13 (q-ary entropy)**.** The entropy $H_q(p)$ for any $p \in [0,1]$ is defined as

$$H_q(p) = -p\log_q p - (1-p)\log_q(1-p) + p\log_q(q-1) \tag{1.28}$$

with $H(0) = H(1) = 0$.

The common case for $q = 2$ reads,

$$H(p) = -p\log p - (1-p)\log(1-p) \tag{1.29}$$

We know that volume of sphere $\mathrm{Vol}_2(n, pn)$ is given by

$$\mathrm{Vol}_2(n, pn) = \sum_{i=0}^{pn} \binom{n}{i}, \quad p < \frac{1}{2}. \tag{1.30}$$

Also, note that for the case of asymptotic conditions (check out *Roth* Lemma 4.7 and *Stirling approximation* for further details)-

$$\lim_{n\to\infty} \mathrm{Vol}_2(n, pn) = \lim_{n\to\infty} \sum_{i=0}^{pn} \binom{n}{i} \approx 2^{nH(p)} \tag{1.31}$$

In general, we have,

Theorem 1.14. For $0 \le t/n \le 1 - (1/q)$

$$\mathrm{Vol}_q(n, t) = q^{nH_q(t/n)+o(1)} \tag{1.32}$$

Theorem 1.15 (Asymptotic Hamming bound)**.** For any binary (n, d) code, the asymptotic rate $R(\delta)$ of the code is bounded by

$$\boxed{R(\delta) \leq 1 - H_q(\frac{\delta}{2})} \tag{1.33}$$

Proof. We with the Hamming bound expression,

$$A_q(n, d) \leq \frac{q^n}{\text{Vol}_q(n, \lceil \frac{n\delta - 1}{2} \rceil).} \tag{1.34}$$

Taking the logarithm,

$$\log_q A_q(n, d) \leq n - \log_q \text{Vol}_q(n, \lceil \frac{n\delta - 1}{2} \rceil) \leq n - nH_q(\delta/2 - 1/n) + o(1) \tag{1.35}$$

Thus,

$$R(\delta) \leq 1 - H_q(\delta/2 - 1/n) + o(1) \tag{1.36}$$

Continuity of $H_q(\cdot)$ stitches up the proof. □

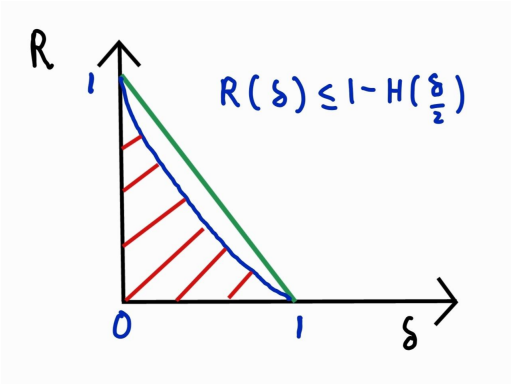


Figure 2: Rate vs δ for asymptotic Hamming bound

Theorem 1.16 (Asymptotic GV Bound)**.** For $[n, nR, \delta n]$ codes over $\mathbb{F}_q$ we have,

$$R(\delta) \geq 1 - H_q(\delta) \tag{1.37}$$

Proof. We have the GV bound as,

$$A_q(n, d) \geq \frac{q^n}{\text{Vol}_q(n, n\delta - 1)} \tag{1.38}$$

Npw,

$$\log_q A_q(n, d) \geq n - \log_q \text{Vol}_q(n, n\delta - 1) \tag{1.39}$$

Since $\text{Vol}_q(n, d-1) = q^{nH_q(\delta - 1/n)}$ we have,

$$R(\delta) \geq 1 - H(\delta - 1/n) + o(1) \tag{1.40}$$

By continuity, the result follows. □

We again have the constructive result as well:

Theorem 1.17. Let n, nR be positive integers and let $\delta \in (0, 1 - 1/q]$ which satisfies,

$$R \leq 1 - H_q(\delta) \tag{1.41}$$

Then there exists a linear $[n, nR, \geq n\delta]$ code over $\mathbb{F}_q$.

Now, we have the asymptotic Plotkin bound,

Theorem 1.18. For a $[n, M, d]$ binary code, we have,

$$R(\delta) \leq 1 - 2\delta \tag{1.42}$$

for $\delta < 1/2$.

Proof. Define $f : C \to \{0,1\}^{n-2d+1}$ as

$$f(c = (c_1, c_2, \ldots, c_n)) = (c_1, c_2, \ldots, c_{n-2d+1}) \tag{1.43}$$

Define for any $x \in \{0,1\}^{n-2d+1}$

$$C_x := \{(c_{n-2d+2}, c_{n-2d+3}, \ldots, c_n) \in \{0,1\}^{2d-1} : f(c) = x\} \tag{1.44}$$

That is, the set of all suffixes for the prefix x. Now, the minimum distance for the elements in C_x is greater than or equal to d. Thus, C_x is a code with length $2d-1$ and minimum distance $\geq d$. Thus, the minimum distance is greater than one half of the length. The Plotkin bound can thus be applied, which tells us,

$$|C_x| \leq \frac{2d}{2d - (2d-1)} = 2d \tag{1.45}$$

For any given $(n - 2d + 1)$-length prefix, there are at-most $2d$ codewords. Thus, we can conclude that,

$$M \leq 2d \times 2^{n-2d+1} \tag{1.46}$$

Further,

$$\log_2 M \leq \log 2d + (n - 2d + 1) \tag{1.47}$$

Dividing by n,

$$R(\delta) \leq 1 - 2\delta + o(1) \tag{1.48}$$

□

Lemma 1.19. For any vector x, define

$$x + C := \{x + c : c \in C\} \tag{1.49}$$

For any two sets $C, A \subseteq \{0,1\}^n$ we have,

$$\sum_x \mid (x + X) \cap A \mid = \mid C \mid \cdot \mid A \mid \tag{1.50}$$

Proof.

$$\sum_x \mid (x + X) \cap A \mid = \sum_x \sum_{c \in C} \sum_{a \in A} 1(x + c = a) = \sum_{c \in A} \sum_{a \in A} \sum_x 1(x = c - a) \tag{1.51}$$

The innermost sum is 1, which proves the lemma. □

Corollary 1.20. $\exists x \in \{0,1\}^n$ s.t.

$$\mid (x + C) \cap A \mid \geq \frac{1}{2^n} \mid C \mid \cdot \mid A \mid \tag{1.52}$$

Theorem 1.21 (Elias)**.** We have,

$$R(\delta) \leq 1 - H\left[\frac{1 - \sqrt{1 - 2\delta}}{2}\right] \tag{1.53}$$

Proof. Let A be the set of all vectors of length n and wt. w, see that $(x + C) \cap A$ also has wt. w. Let C be a code with min. dist. d, then $(x + C) \cap A$ is again a code of min. dist. d, but with constant weight w. Further, by the corollary above,

$$|(x + C) \cap A| \geq \frac{1}{2^n} |C| \binom{n}{w} \tag{1.54}$$

But, by the Johnson bound,

$$|(x + C) \cap A| \leq \frac{2nd}{(n - 2w)^2 - n(n - 2d)} \tag{1.55}$$

Chaining these two, we get,

$$\frac{1}{2^n} |C| \binom{n}{w} \leq \frac{2nd}{(n - 2w)^2 - n(n - 2d)} \tag{1.56}$$

Singleton	$R(\delta) \leq 1 - \delta$
Hamming	$R(\delta) \leq 1 - H(\delta/2)$
Plotkin	$R(\delta) \leq 1 - 2\delta$
E-B	$R(\delta) \leq 1 - H\left[\frac{1-\sqrt{1-2\delta}}{2}\right]$
GV	$R(\delta) \geq 1 - H(\delta)$

Table 1: Summary of bounds

Thus,

$$|C| \leq \frac{2nd \cdot 2^n}{\binom{n}{w} \cdot (n-2w)^2 - n(n-2d)} \tag{1.57}$$

Choose $w = (n - \sqrt{n^2 - 2nd + 1})/2$ to get,

$$|C| \leq \frac{2^n \cdot 2nd}{\binom{n}{(n-\sqrt{n^2-2nd+1})/2}} \tag{1.58}$$

This leads to

$$R(\delta) \leq 1 - H\left[\frac{1-\sqrt{1-2\delta}}{2}\right] \tag{1.59}$$

□

Note that these bounds are of merit when we work with smaller alphabets. With bigger alphabets, we can just create Reed-Solomon codes with good properties easily. A nice question is,

Are there explicit **binary** *codes which match the GV lower bound?* (1.60)

Existence of such codes is known, but not explicity constructions. Moreover, the general idea is to find/construct asymptotically good codes.

§§1.3. Reed-Solomon codes

With RS codes, we will introduce the more general notion of constructing codes using polynomials over finite fields. Given the field $\mathbb{F}_q$, such a polynomial looks likee,

$$f(x) = a_0 + a_1 x + \cdots + a_d x^d \tag{1.61}$$

for $a_i, x \in \mathbb{F}_q$. We denote,

$$\mathbb{F}_q[X] := \{f(\cdot) : f \text{ is a polynomial evaluated for} x \text{ over } \mathbb{F}_q\} \tag{1.62}$$

We do away with a lemma.

Lemma 1.22. We have,

1. A non-zero polynomial of degree d over $\mathbb{F}_q$ has atmost d roots.
2. Given $\{(\gamma_i, y_i) \in \mathbb{F}_q \times \mathbb{F}_q : i \in \{1, 2, \cdots, d+1\}\}$ there exists a unique polynomial with degree $\leq d$ such that $f(\gamma_i) = y_i \forall i \in \{1, 2, \cdots, d+1\}$.

The proofs are easy and we skip them. Let us define RS codes.

Definition 1.23. Given $q > n \geq k$, and $\gamma_1, \gamma_2, \ldots, \gamma_n \in \mathbb{F}_q$ s.t. $\gamma_i \neq \gamma_j$ for $i \neq j$ define,

$$RS_q(\vec{\gamma}, n, k) := \{(f(\gamma_1), f(\gamma_2), \ldots, f(\gamma_n)) : f \in \mathbb{F}_q[X], deg(f) \leq k-1\} \quad (1.63)$$

This is a $[n, k]$ linear code.

Note that there are q^k polynomials over $\mathbb{F}_q$ with $def(f) \leq k-1$. Thus, formally, we are mapping the q^k possible input messages to polynomials, and evaluating them at n distinct evaluation points to form the codewords. One way to do this is,

$$\mathbf{u} = (u_0, \ldots, u_{k-1}) \mapsto f_u(x) := \sum_{i=0}^{k-1} u_i x^i \quad (1.64)$$

$$u \xrightarrow{\mathcal{C}} (f_u(\gamma_1), f_u(\gamma_2), \cdots, f_u(\gamma_n)) \quad (1.65)$$

It is easy to see that the generator matrix is,

$$G = \begin{bmatrix} 1 & 1 & \cdots & 1 \\ \gamma_1 & \gamma_2 & \cdots & \gamma_n \\ \gamma_1^2 & \gamma_2^2 & \cdots & \gamma_n^2 \\ \vdots & \vdots & \cdots & \vdots \\ \gamma_1^{k-1} & \gamma_2^{k-1} & \cdots & \gamma_n^{k-1} \end{bmatrix} \quad (1.66)$$

Further, as we claimed before, RS codes are MDS. Since the number of roots of every polynomial is bounded by $k-1$, we have that $d \geq n - (k-1)$. Coupling this with the singleton bound,

$$\boxed{d_{RS} = n - k + 1} \quad (1.67)$$

So, if $d - 1 = n - k$ bits drop out of our n-bit RS code, we can still recover the message. As long as k-bits are intact in the n-bit message, the information is preserved (the reduced $k \times k$ generator matrix is invertible – just a linear map!).

What is the p.c. matrix of RS codes, you ask? We give two lemmas to answer this.

Lemma 1.24. Let $\mathbb{F}_q^*$ be the set of all non-zero elements of $\mathbb{F}_q$. Then, $\mathbb{F}_q^*$ is a cyclic group, generated by some $\alpha \in \mathbb{F}_q^*$.

$$\mathbb{F}_q^* = \langle \alpha \rangle \quad (1.68)$$

Lemma 1.25. Given the field $\mathbb{F}_q$,

$$\sum_{\gamma \in \mathbb{F}_q} \gamma^l \quad \forall 0 < l < q-1 \tag{1.69}$$

Proof. Reduce the sum to over $\mathbb{F}_q^*$ (why?), use the previous lemma and write $\gamma_i = \alpha^i$, write the geometric series, and conclude. □

With these lemmas, we have the following.

Proposition 1.26. Let $q = n+1$. Let $\mathbb{F}_q^* = \langle \alpha \rangle$. Then,

$$RS(n,k,(\alpha^0,\alpha^1,\ldots,\alpha^{n-1})) = \{(c_0,c_1,\ldots,c_{n-1}) \in \mathbb{F}_q^n : c(x) := \sum_i c_i x^i \\ \text{has } c(\alpha^j) = 0 \text{ for all } j = 1,2,\ldots,d-1\} \tag{1.70}$$

Now this is a *dual* interpretation of RS codes, wherein the codewords are polynomials instead of the input messages.

Proof. This property is equivalent to the fact that we can write the p.c. matrix as,

$$H = \begin{bmatrix} 1 & \alpha & \alpha^2 & \cdots & \alpha^{n-1} \\ 1 & \alpha^2 & \alpha^4 & \cdots & \alpha^{2(n-1)} \\ 1 & \vdots & \vdots & \cdots & \vdots \\ 1 & \alpha^{n-k} & \alpha^{2(n-k)} & \cdots & \alpha^{2(n-1)(n-k)} \end{bmatrix} \tag{1.71}$$

Now, note that the $(i,j)^{th}$ entry of GH^T is given as,

$$\sum_{l=0}^{n-1} \alpha^{(i+j)l} = \sum_{a \in \mathbb{F}_q^*} a^{(i+j)} \tag{1.72}$$

But $0 < i+j \leq k-1+n-k < q-1$. By the lemmas above, this is zero. Thus, H is the p.c. matrix, and the dual interpretation holds true. □

How do we perform decoding of the class of RS codes? Since they are linear codes, we can always follow coset decompositions and NN decoding, but the central question is, can we use the underlying polynomial structure of the RS code to perform efficient decoding? Let us discuss one such algorithm using the polynomial structure.

Berlekamp-Welch Algorithm Let us have an $RS(n,k,\vec{\gamma})$ code with $e \leq t = \lfloor (n-k)/2 \rfloor$ errors. Let the received codeword be $\vec{r}$, which, for some polynomial

$f \in \mathbb{F}_q[X]$ with $deg(f) \leq k-1$ differs from $f(\vec{\gamma})$ at e locations. How do we find f? Define the error locator polynomial,

$$E(x) := \Pi_{i:r_i \neq f(\gamma_i)}(x - r_i) \tag{1.73}$$

Note that $deg(E) \leq t$. Then we *will* have

$$r_i E(\gamma_i) = f(\gamma_i)E(\gamma_i) \tag{1.74}$$

Why? Well, if $\vec{r}$ and f agree on, γ_i then $f(\gamma_i) = r_i$ and it follows. If not, then $E(\gamma_i) = 0$, and it follows again. If we think of $f(\gamma_i)E(\gamma_i)$ as evaluations of a polynomial $Q(x)$, then $f \equiv Q/E$. Thus we describe the algorithm as follows.

Algorithm 1 BW algorithm for RS codes

Require: $\vec{r}, \vec{\gamma}, n, k$
  Find
    1. a monic polynomial $E(\cdot)$ with $deg(E) = t$
    2. a polynomial $Q(\cdot)$ of $deg(Q) \leq t + k - 1$ s.t. $r_i E(\gamma_i) = Q(\gamma_i) \forall i$
  if no such polynomials E, Q exist **then**
    declare error
  end if
  $\hat{f}(x) \leftarrow Q(x)/E(x)$
  if $\hat{f}(\gamma_i) \neq r_i$ at more than t locations **then**
    declare error
  end if
  output $\hat{f}$

It is not difficult to prove that if there exists f with $deg(f) \leq k-1$ and $\leq t$ error occur, such E, Q exist and $f = Q/E$. Further, one can show that if multiple such (Q, E) exist, then Q/E remains same. Moreover, the complexity of this algorithm including the solution of the systems of linear equations and polynomial divisons is $\mathcal{O}(n^3)$, which is fairly nice. But, better decoders for the RS codes exist.

Alright. But what about the question we asked before – asymptotically good codes over small fields? Okay, so what if we just 'binarize' this RS code? That is, given this $[n, k, d = n - k + 1]$ RS code over $\mathbb{F}_q$ with $n = \amalg$ with codewords $c = (c_1, \ldots, c_n)$, we just map each c_i to a binary representation of $\log q$ bits? If this binarized code is $[n', k', d']$, then we would have

$$n' = n \log q; \;\; k' = k \log q \tag{1.75}$$

Note that, in this binarization, if one element $c_i \neq 0$, then it will contribute atleast one non-zero element (i.e., 1) in the binarization. Thus,

$$d' \geq n - k + 1 \tag{1.76}$$

Then,

$$\delta := \frac{d'}{n'} \sim \frac{n-k+1}{n \log q} \sim \frac{n-k+1}{n \log n} \tag{1.77}$$

This is,

$$\delta \sim \frac{1-R}{\log n} + o(1) \tag{1.78}$$

Owing to the logarithm term, we have $\delta = o(1)$! Thus, this code is *not* asymptotically good! That is unfortunate, but does not defeat theoretical efforts. We will next discuss methods to beat this, including a new concept called *code concatenation*, which is a valuable tool in quantum error correction as well.

§§1.4. Reed Müller Codes

We now discuss the simplest 'extension' of RS codes: the Reed-Müller (RM) codes. They are defined by using multivariate polynomials, instead of polynomials of one variable.

$$\mathcal{C}(r,m) := \{(f(\alpha_1),\ldots,f(\alpha_{2^m}) : f \in \mathbb{F}_2[X_1,\ldots,X_m] \text{ with } deg(f) \leq r, \alpha_i \in \mathbb{F}_2^m\} \tag{1.79}$$

Note that this is a linear code (why?). The length of a codeword is 2^m. The number of codewords, is basically the number of multinomials satisfying the degree constraint. This would lead to,

$$|\mathcal{C}| = \binom{m}{0} + \binom{m}{1} + \cdots + \binom{m}{r} \tag{1.80}$$

Why? Note that for $x \in \mathbb{F}_2$, $x^2 = x$. Further, we adopt the following notation for the multinomials,

$$f(X) := \sum_{S\subseteq[m],\ |S|\leq r} C_S \Pi_{i\in S} X_i \tag{1.81}$$

How do we characterize the distance properties? For RM codes, we used facts about the number of roots of polynomials over finite fields. Now we give a lemma which does that (sort of) for multinomials.

Lemma 1.27 (Schwartz-Zippel Lemma)**.** Let $f \in \mathbb{F}_2[X_1,\ldots,X_m] \neq 0$ with $deg(f) \leq r$, then,

$$\sum_{\alpha\in\mathbb{F}_2^m} \mathbf{1}[f(\alpha) \neq 0] \geq 2^{m-r} \tag{1.82}$$

This lemma says that the number of *non*-roots is greater than or equal to 2^{m-r}. Further, a multinomial like $f \equiv X_1X_2\ldots X_r$ saturates this inequality with number of non-roots exactly 2^{m-r}. This means that all codewords will have weight atleast 2^{m-r} and there is a codeword with weight 2^{m-r}. Thus, the code distance is 2^{m-r}. Now, let us discuss a decoding algorithm for such codes.

Decoding multinomial codes. Suppose the multinomial generating the true codeword is

$$f(X) := \sum_{S\subseteq[m],\ |S|\leq r} C_S \Pi_{i\in S} X_i \tag{1.83}$$

and the received word is $r = (r_1, \ldots, r_{2^m})$. We are perhaps guaranteed that $(f(\alpha_1), \ldots, f(\alpha_{2^m})$ differs from $r = (r_1, \ldots, r_{2^m})$ at atmost $(d-1)/2 = 2^{m-r-1} - 1$ locations. Define,

$$R_S(X) := \Pi_{i \in S} X_i \tag{1.84}$$

Then f becomes

$$f(X) = \sum_S C_S R_S(X) \tag{1.85}$$

We will use this notation and outline an algorithm to decode each C_S. We start with the following lemmas

Lemma 1.28. For any $S \subseteq [m]$ we have

1. $\sum_{a \in \mathbb{F}_2^{|S|}} R_S(a) = 1$
2. For any $T \subset S$,
$$\sum_{a \in \mathbb{F}_2^{|S|}} R_T(a) = 0 \tag{1.86}$$

Proof. Part 1. is easy to see, there will be only one a such that $R_S(a) = 1$. Part 2. can be justified from the fact that for every bit in a which is outside of T, the sum will have the same value when that bit is zero and one, so when summed up, the identical values add up to zero. □

Now the idea is, given $S \subseteq [m]$ with $|S| = r$, we can devise 2^{m-r} parity checks for the value of C_S and choose $\hat{C}_S$ as the majority value in those set of parity checks. Formally,

$$\hat{C}_S := \text{MAJORITY DECODE}_{b \in \mathbb{F}_2^{m-r}} \left(\sum_{a \in \mathbb{F}_2^m : a_{\bar{S}} = b} f(a) \right) \tag{1.87}$$

This works because,

$$f_b(X) = C_S R_S(X) + \sum_{T \subset S} \zeta_T R_T(X) \tag{1.88}$$

The first part of the lemma applies to the first term in the LHS, and the second to the second, when we sum up over X s.t. $X_{\bar{S}} = b$. This is done recursively, once we do this for $|S| = r$, we keep those decoded codewords aside, and proceed to decode the set of S with $|S| = m - 1$.

§§1.5. Making bigger codes: concatenated codes

Consider two codes, one which is $\mathcal{C}_{out} = [n_{out}, k_{out}, d_{out}]$ over Σ_{out}, and one which is, well, $\mathcal{C}_{in} = [n_{in}, k_{in}, d_{in}]$ over Σ_{in} with $|\Sigma_{in}| \leq |\Sigma_{out}|$. The idea, with our extremely non-chalant notation, is that, we want to take input messages,

encode using $\mathcal{C}_{out}$ and *then* using $\mathcal{C}_{in}$ by encoding each code symbol of the $\mathcal{C}_{out}$ codeword into $\mathcal{C}_{in}$. This would require,

$$|\Sigma_{out}| = |\Sigma_{in}|^{k_{in}} \tag{1.89}$$

Thus we would finally have a $[n, k, d]$ code **over**[2] Σ_{in} with

1. $n = n_{in}n_{out}$
2. $k = k_{in}k_{out}$
3. $d \geq d_{in}d_{out}$
4. $\delta \geq \delta_1\delta_2$
5. $R = R_1R_2$.

The first two are straightforward, the third follows by noting the fact that a non-zero symbol in the outer word will lead to atleast d_{in} non-zero symbols in the total codeword, leading to atleast $d_{in}d_{out}$ non-zero symbols in total. Thus,

$$\begin{aligned} m \in \Sigma_{in}^{k_{in}k_{out}} \sim m \in \Sigma_{out}^{k_{out}} \xrightarrow{\mathcal{C}_{out}} x = (x_1, x_2, \ldots, x_{n_{out}}) \in \Sigma_{out}^{n_{out}} = \Sigma_{in}^{n_{out}k_{in}} \\ \xrightarrow{\mathcal{C}_{in}} (\mathcal{C}_{in}(x_1), \ldots, \mathcal{C}_{in}(x_{n_{out}})) \in \Sigma_{in}^{n_{out}n_{in}} \end{aligned} \tag{1.90}$$

One example could be taking RS-code over $\mathbb{F}_{2^3}$ with $n_{out} = 6, k_{out} = 2$. We could concatenate this with a Hamming code, which is $[7, 4, 3]$. Check. We will in general take RS codes as the outer codes, and codes with nice asymptotic properties as inner codes, in the hope that we will end up getting codes with good rate and distance asymptotically.

Theorem 1.29. Zyablov Bound For all $\epsilon > 0$ there exists a family of explicit and binary codes with asymptotic rate R and relative distance δ such that

$$R \geq \sup_{0<r<1-H(\delta)-\epsilon} \left(r \left[1 - \frac{\delta}{H^{-1}(1-r-\epsilon)} \right] \right) \tag{1.91}$$

The proof is as follows, via concatenation.

Proof. Take a RS code with $\delta_{out} = 1 - R_{out}$ and $R_{out} \in (0, 1)$ as the outer code, and take a binary linear code which lies on the GV curve $r \geq 1 - H(\delta_{in}) - \epsilon$. We will have

$$\delta \geq (1 - R_{out})H^{-1}(1 - r - \epsilon) \tag{1.92}$$

and

$$R \geq r \left[1 - \frac{\delta}{H^{-1}(1-r-\epsilon)} \right] \tag{1.93}$$

Since r is a free parameter, we can take a supremum accordingly. □

[2] if we forget this, things will not make sense!

The idea is that, if the RS code has evaluation points $\mathbb{F}_q^*$, then we have $q = 2^{k_{in}}$ and $n_{out} = q - 1 = 2^{k_{in}} - 1$. Leading to $k_{in} \approx \log n_{out}$ and

$$n_{in} = \frac{k_{in}}{r} = \frac{\log n_{out}}{r} \leq \frac{\log n}{r} \tag{1.94}$$

We can just find $\mathcal{C}_{in}$ by a brute force search over all $k_{in} \times n_{in}$ generating matrices, which still has complexity

$$2^{n_{in} \times k_{in}} \leq 2^{\log^2 n/r} = n^{\mathcal{O}(\log n)} \tag{1.95}$$

Which is O.K. Further, if we construct the p.c. matrix iteratively by adding columns, the complexity of finding one column is $2^{\mathcal{O}(\log n)} = \mathcal{O}(\text{POLY}(n))$.

We now describe another way of using concatenated codes for getting good codes over the binary fields. This is called the Justesen construction, and is based on two ideas,

1. We don't need all the inner codes to be the same
2. We can do away with a large fraction of inner codes being good

We let the outer code be $\text{RS}(\mathbb{F}_q^*, n_{out}, k_{out})$ with $q = 2^{k_{in}}$ and $n_{out} = q - 1$ and $R_{out} = k_{out}/n_{out}$. We fix the rate of any inner code to be $r_{in} = 1/2$ and assume that $1 - \epsilon$ are *good*, that is, they have $\delta \geq H^{-1}(1 - r - \epsilon)$. We have the following result.

Proposition 1.30 (Justesen Construction)**.** For $R_{out} \in (0, 1)$ the Justesen construction is asymptotically good with rate $R = R_{out}/2$.

Proof. Since the outer code is RS, a fraction of $1 - R_{out} = \delta_{out}$ bits of the outer codewords is atleast non-zero. Now, if atmost a fraction ϵ of the n_{out} inner codes is bad, then a fraction of

$$1 - R_{out} - \epsilon \tag{1.96}$$

of outer code symbols will map via good inner codes. Thus,

$$\delta \geq \frac{(1 - R_{out} - \epsilon) n_{out} \times (2k_{in} H^{-1}(1/2 - \epsilon))}{2k_{in} k_{out}} \tag{1.97}$$

leading to $\delta \geq (1 - R_{out} - \epsilon) H^{-1}(1/2 - \epsilon)$. We can choose $1 - R_{out} > 2\epsilon$ and get,

$$\delta \geq \epsilon H^{-1}(1/2 - \epsilon) > 0 \tag{1.98}$$

□

Well that's nice, but what did we gain from such an argument? The actually nice fact is that such an inner code family can be found *explicitly*! This turns us over to the *Wozencraft* ensemble. Working over $\mathbb{F}_2$, for $k > 1$ let $\alpha \in \mathbb{F}_2^*$ with $\alpha \neq 0$, and define $C^\alpha : \mathbb{F}_2^k \to \mathbb{F}_2^{2k}$ as

$$C^\alpha(x) := (x, \alpha x) \ \ \forall x \in \mathbb{F}_q^k \tag{1.99}$$

Note that the rate of any such code is $1/2$, and, if $\alpha_i \neq \alpha_j$, the corresponding codes cannot share a non-zero codeword (why?). Now, define $P \subset \mathbb{F}_2^k$ to be set of α's for which $\delta < H^{-1}(1/2 - \epsilon)$. That is, some $\alpha \in P$ if there exists a $c \in C^\alpha$ with $wt(c) < (2k)H^{-1}(1/2 - \epsilon)$. Now, since the codes do not share non-zero codewords, the number of bad codes, $|P|$ is upper bounded by the number of vectors in $\mathbb{F}_q^{2k}$ with $wt(\cdot) < (2k)H^{-1}(1/2 - \epsilon)$. This number is $Vol_2((2k)H^{-1}(1/2 - \epsilon), 2k)$, and thus,

$$|P| < 2^{2kH(H^{-1}(1/2-\epsilon))} = 2^{k-\epsilon/2} < \epsilon 2^k \tag{1.100}$$

for small enough ϵ. Note that the number of codes in this family is $2^k - 1 \sim 2^k$, so the *fraction* of bad codes is bounded by ϵ, just as we assumed in the Justesen construction. Neat, isn't it?

§2. Quantum Error Correcting Codes

§§2.1. Introduction: correcting quantum errors

Cutting short the introduction too soon, the idea is as follows: today's quantum computers fall in the so call noise intermediate scale quantum computing (NISQ) era, wherein, *noise*, is a big problem. Generically, quantum states are not stable. There is an arrow of time, and a decay to the lowest energy state everpresent. For instance, in a superconducting qubit, there are (effictively) two levels, $|e\rangle$ and $|g\rangle$, where $|g\rangle$ is the ground state of the qubit. Any such qubit is characterized by a *relaxation* time T_1, which is a measure of time taken for the qubit to relax into the ground state if it were prepared in the excited state. This, amongst other issues, make the world of quantum errors.

Let us jump into the central problem without any delay. Consider a single qubit state $|\psi\rangle = a|0\rangle + b|1\rangle$ and consider errors on this state. Apriori, there exists a *continuum* of errors, which worries us. But let's start simple. Consider single qubit X errors, the so-called bit-flips in analogy to the classical case. That is, one could have,

$$a|0\rangle + b|1\rangle \to a|1\rangle + b|0\rangle \tag{2.1}$$

How do we protect our information against such errors? Let's follow the classical steps, and start with the simplest error-correcting code, the three *qubit* bit-flip code. Wherein, we do,

$$|0\rangle \mapsto |000\rangle\,; \;\; |1\rangle \mapsto |111\rangle \tag{2.2}$$

Note that this does not violate the no-cloning theorem (why?). This is shown in the circuit Fig.3.

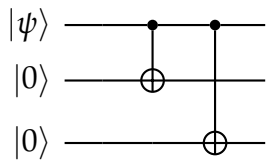


Figure 3: Encoding for the three qubit bit-flip code

Now, in the classical case, we simply defined *parity checks* to assist in decoding. How do we talk about quantum parity checks? Motivated by the formalism of general and projective measurements, we can define the following prjoectors,

$$P_0 := |000\rangle\langle 000| + |111\rangle\langle 111| \tag{2.3}$$
$$P_1 := |100\rangle\langle 100| + |011\rangle\langle 011| \tag{2.4}$$
$$P_2 := |010\rangle\langle 010| + |101\rangle\langle 101| \tag{2.5}$$
$$P_3 := |001\rangle\langle 001| + |110\rangle\langle 110| \tag{2.6}$$

What did we acheive? Consider an X error on the second qubit on the encoded state $a|000\rangle + b|111\rangle \mapsto a|010\rangle + b|101\rangle$. Note that $\langle\psi|P_2|\psi\rangle = 1$ and

all $\langle\psi|P_i|\psi\rangle = 0 \forall i \neq 2$. That is, the projector P_2 projected the state onto the subspace with a bit-flip on the second qubit. Thus, if we assume that only a bit flip error occurs on our intial encoded state $a|000\rangle + b|111\rangle$ we can detect the location of the error using these projectors. What next? Correction! Given that there is an error on the second qubit, we can swiftly correct it by applying a X to the second qubit. Note that under the assumption of only single qubit X errors, we never perturb the state while doing these proejctions. Neat, no?

But wait – in QM1 we learned that measurements in quantum mechanics are facilitated by hermitian observibles. Where did those go? Consider the two observables Z_1Z_2 and Z_2Z_3. What happens when we measure these two on our errorful state? We get a vector $v \in \mathbb{Z}_2^2$. Why is this useful? Note,

$$Z_1Z_2 = (|00\rangle\langle 00| + |11\rangle\langle 11|) \otimes I - (|01\rangle\langle 01| + |10\rangle\langle 10|) \otimes I \tag{2.7}$$

which upon measurement gives us ± 1 depending on the *similarity* of the first and second qubits. Similarly, Z_2Z_3 gives us this for the next two qubits. If we get $v = (1,1)$, this means that all the three qubits are in the same bit-state, which under the assumption of error probability $< 1/2$ implies the maximum likelihood scenario of no error occuring. Similarly, $v = (-1,1)$ means the first two qubits are different, but the next two are the same, hence localizing the error on the first qubit. This way, we can do the projections discussed above using measurable operators.

But this is quantum! Why must thou correct bit-errors solely? Convince yourself that single qubit phase errors (Z errors) can be corrected using a very similar encoding where $|0\rangle \mapsto |+++\rangle$ and $|1\rangle \mapsto |---\rangle$, as the encoding circuit in Fig.4.

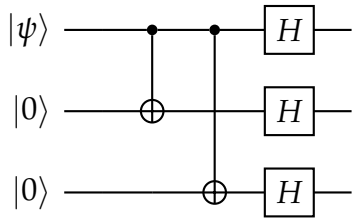


Figure 4: Encoding for the three qubit phase-flip code

So far, we only corrected for either bit flips or phase flips errors. What if we have both? We use the concatenation technique from classical ECC and apply it here, and see what we get if we concatenate the two, by applying the phase-flip code first, followed by the bit-flip code. Recall that if we wish to concatenate $\mathcal{C}_1 = [n_1, k_1, d_1]$ code over Σ_1 with a $\mathcal{C}_2 = [n_2, k_2, d_2]$ code over Σ_2 with $|\Sigma_1| = |\Sigma_2|^{k_2}$, we first encode with $\mathcal{C}_1$ to get a n_1-length vector over Σ_1, and express each element of that vector as a k_2 length vector over Σ_2 and subsequently encode each k_2-length vector using $\mathcal{C}_2$ to get a n_2 length vector over Σ_2; and combining all these n_2 length vectors lead to final codewords of length

$n_1 n_2$ over the smaller alphabet. In our case, the qubit space is always 'binary' (two-dimensional), so $\Sigma_1 = \Sigma_2$

$$|0\rangle \mapsto |+++\rangle \mapsto (|+\rangle \to ((|000\rangle + |111\rangle)/\sqrt{2})^{\otimes 3} \tag{2.8}$$

This can be seen as the following circuit

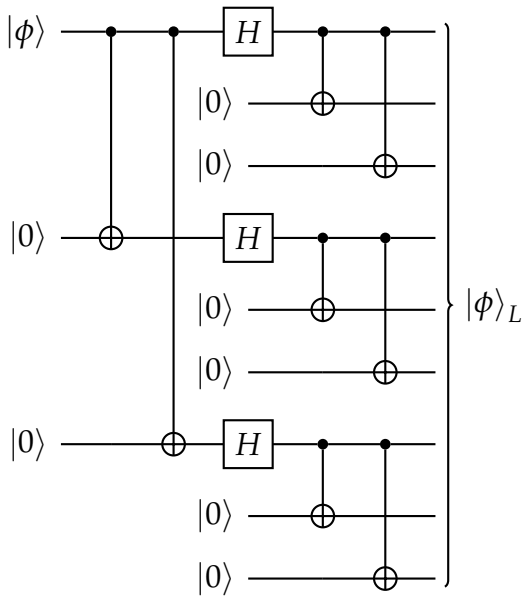


Figure 5: The 9-qubit Shor code

This code is capable of correcting any single qubit X or Z error. In fact, it can correct *arbitrary* single qubit errors, which we will see later. We have,

$$|0\rangle \mapsto \frac{(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)(|000\rangle + |111\rangle)}{2\sqrt{2}} \tag{2.9}$$

If we have a X error on some qubit, we can correct it by the properties of the local 3-qubit bit-flip code. If we have a phase error, we need to check parities in the three rows of qubits. For instance, a phase error in the first qubit for the logical-zero state will result in the first three qubits having a minus sign, and the rest of the rows having the plus signs. We can detect these, and correct for them. The corresponding measurement operators are,

Bit-flips $\{Z_1Z_2, Z_2Z_3, Z_4Z_5, Z_5Z_6, Z_7Z_8, Z_8Z_9\}$

Phase-flips $\{X_1X_2X_3X_4X_5X_6, X_4X_5X_6X_7X_8X_9\}$

Interestingly, note that phase errors on qubits in the same row have the same effect of changing the sign in that row. But, flips have very different effects. Thus the Shor code is said to be highly degenerate to phase errors, and not bit flips.

Thus the takeaway from this discussion is that, *quantum* error correcting codes should work in a fashion wherein the correctable errors map the codespace into

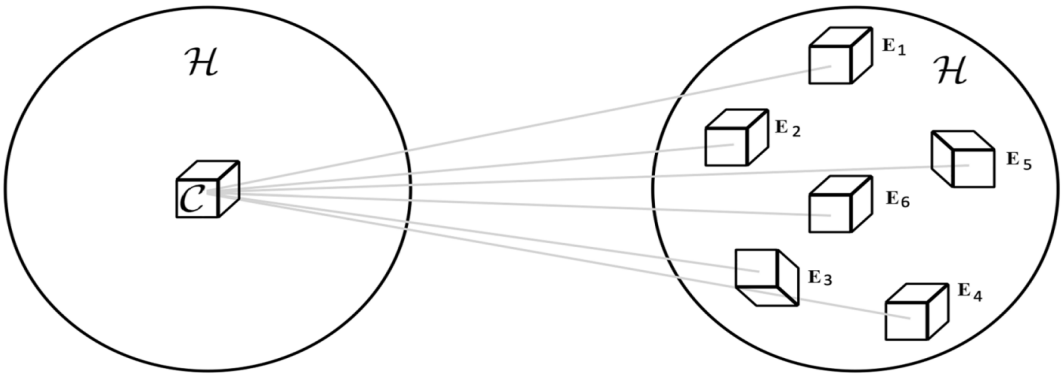


Figure 6: A good QECC, figure from [JM23]

orthogonal Hilbert spaces sitting inside the big Hilbert space (cleverly called 'pizza slices' by Serge Rosenblum at a colloquia in ICTS), as shown in Fig.6.

Now, let us address the question regarding a possible continuum of errors that we asked before. We know that, a generic single qubit operation can be written as,

$$\mathcal{E}(\rho) = \sum_i E_i \rho E_i^\dagger \tag{2.10}$$

wherein ρ is a single qubit density matrix, and E_i are Kraus operators, and $\mathcal{E}(\cdot)$ describes a generic CPTP map. Simple linear algbra tells us that,

$$E_i = c_{i0} I + c_{iX} X + c_{iZ} Z + c_{iXZ} XZ \tag{2.11}$$

using that fact that $Y \sim XZ$. If we allow such errors to act and measure our measurement operators, we will always collapse to either one of $|\psi\rangle, X|\psi\rangle, Z|\psi\rangle, XZ|\psi\rangle$ and we will get the corresponding syndromes. But, if we collapsed to the right subspace, and got the right syndrome, and applied the right correction, is this any different than only X, Z errors occuring? No! Thus, we have effectively *discretized* our problem– if we can correct for X and Z errors, we can correct for *any* single-qubit errors! What about multi-qubit errors, you ask? Well, keep on reading, I'm not gonna give it away for free.

§§2.2. Theory of quantum error correction

Let us proceed with a more formal theory.

What do we do? We have to start somewhere. Let's take a first look. Consider a quantum system in a state $|\psi\rangle$ of k qubits, and a model for the error a quantum channel $\mathcal{E}$ described by the Kraus operators E_a. Let's follow the footsteps of classical computer scientists, and encode our state into $n > k$ qubits by adding redundancy suitably. What we will be left with, is the familiar notion of a codespace,

$$\mathcal{C} := \{|c_i\rangle : i \in [M], |c_i\rangle \in \mathcal{H}_2^n\} \tag{2.12}$$

where $M = 2^k$. Now, say we have a codeword state $|\psi\rangle$ and an error occurs,

$$\rho_1 := \mathcal{E}(|c\rangle) = \sum_a E_a |c\rangle \langle c| E_a^\dagger \tag{2.13}$$

What we hope to do now, is to apply a *recovery operation*, which formally, is another CPTP map $\mathcal{R} = \{R_r\}$. Then, we have,

$$\rho_2 := \mathcal{R}(\mathcal{E}(|c\rangle)) = \sum_{r,a} R_r E_a |c\rangle \langle c| E_a^\dagger R_r^\dagger \tag{2.14}$$

What now? We *hope* that $\mathcal{R}(\mathcal{E}(|c\rangle)) \approx |c\rangle$, right? Let's try to quantify this. Remember fidelity? Define,

$$\mathfrak{F}(|c\rangle, \mathcal{R}, \mathcal{E}) := Tr(|c\rangle \langle c| \rho_2) = \sum_{r,a} |\langle c|R_r E_a|c\rangle|^2 \tag{2.15}$$

The error, is then,

$$\mathfrak{E}(|c\rangle, \mathcal{R}, \mathcal{E}) := 1 - \mathfrak{F}(|c\rangle, \mathcal{R}, \mathcal{E}) = \sum_{r,a} |(R_r E_a - \langle c|R_r E_a|c\rangle|^2 \tag{2.16}$$

But wait – why are we fixating over a single ket? We want to protect the whole codespace, yes? Let us do so. We can then define,

$$\mathfrak{E}(|c\rangle, \mathcal{R}, \mathcal{E}) := \min_{|c\rangle \in \mathcal{C}} \sum_{r,a} |(R_r E_a - \langle c|R_r E_a|c\rangle) |c\rangle|^2 \tag{2.17}$$

The solution is to make sure all the terms inside are zero. That is,

$$|(R_r E_a - \langle c|R_r E_a|c\rangle) |c\rangle| = 0; \ \forall |c\rangle \in \mathcal{C} \tag{2.18}$$

Thus, such a condition must hold for all E_a. We can concretify this statement while giving an equivalence relation for the code $(\mathcal{C}, \mathcal{R})$ to correct E_a. We denote the set of errors correctible by such a code as $\mathfrak{E}(\mathcal{C}, \mathcal{R})$.

Theorem 2.1. The Kraus operator E_a is in $\mathfrak{E}(\mathcal{C}, \mathcal{R})$ iff, when restricted to the codespace we have that $R_r E_A = \gamma_{r,a} 1$ for all R_r. Further, $\mathfrak{E}(\mathcal{C}, \mathcal{R})$ is linearly closed.

First of all: the linear closure of the correctable family is nice, yes? It takes away the stress of having to deal with a continuum of errors as one would expect – as long as one can correct for a set of basic ('basis'? Pauli?) We will see this later) errors, one can correct for arbitrary linear combinations. Second of all, the theorem above consists of the correction operator. Can we come up with a correction condition without explicitly involving the correction? Let us reformulate this with the aid of the following definition.

Definition 2.2. For the code $\mathcal{C} = \{|c_i\rangle\}$, define the projector onto the codespace,

$$P_{\mathcal{C}} := \sum_i |c_i\rangle \langle c_i| \tag{2.19}$$

We then have,

Theorem 2.3 (Knill-Laflamme Conditions)**.** The code is $\mathcal{E}$-correctable iff

$$P_{\mathcal{C}} E_a^\dagger E_b P_{\mathcal{C}} = \lambda_{ab} P_{\mathcal{C}} \tag{2.20}$$

Now the KL equations are the compact statement that we were looking for. If we write this using the states explicitly, we have,

$$\boxed{\langle c_i | E_a^\dagger E_b | c_j \rangle = \delta_{ij} \lambda_{ab}} \tag{2.21}$$

Intuitively, this says the following two things:

1. If errors act on initially orthogonal states $i \neq j$, then the overlap in the error-ful states should be zero.
2. If different errors act on the same state i, then the overlap between the corrupted state should not depend on i itself, only the errors as λ_{ab}.

We now give the proof for the KL conditions. This can be skipped if the intuition is well understood, but the explicit error correction operator constructed here is useful in a bunch of other formal stuff.

Proof. $\implies$ Recall that we have unitary equivalence in the action of Kraus operators. With this we claim we can find unitarily equivalent Kraus operators $\{F_k\}$ with the same action as $\{E_k\}$ such that the coefficient matrix λ becomes diagonal. Note that λ is a hermitian matrix, and thus, we can form a diagonal matrix $d = u^\dagger \lambda u$, where u is unitary. Define,

$$F_k := \sum_i u_{ik} E_i \tag{2.22}$$

Check that $P_{\mathcal{C}} F_k^\dagger F_l P_{\mathcal{C}} = d_{kl} P_{\mathcal{C}}$. We now apply the polar decomposition,

$$F_k P = U_k \sqrt{P F_k^\dagger F_k P} = \sqrt{d_{kk}} U_k P \tag{2.23}$$

Thus, when F_k acts on the codespace, it rotates it into a subspace defined by the projector

$$P_k := U_k P U_k^\dagger = F_k P U_k^\dagger / \sqrt{d_{kk}} \tag{2.24}$$

Now, one can see that $P_l P_k = 0$ for $l \neq k$ by the fact that d is diagonal. Thus, these subspaces are orthogonal, as needed for a good code! Now, one can easily measure the syndromes with the projectors P_k and recover by applying $U_k^\dagger$ when projected. Formally, if we define,

$$\mathcal{R}(\sigma) := \sum_k U_k^\dagger P_k \sigma P_k U_k \tag{2.25}$$

one sees that $\mathcal{R}(\mathcal{E}(\rho)) \propto \rho$ Thus, given that the KL conditions are satisfied for $(\mathcal{C}, \mathcal{E})$, we have shown that there exists a correction operator $\mathcal{R}$.

$\Longleftarrow$ If $\mathcal{R}$ corrects for $\mathcal{E}$ on $\mathcal{C}$, then we must have,

$$\mathcal{R}(\mathcal{E}(P\rho P)) \propto P\rho P \tag{2.26}$$

for all states ρ. Naturally, the constant of proportionality c should not depend on the state. We then have,

$$\sum_{ik} R_k E_i P \rho P E_i^\dagger R_k^\dagger = cP\rho P \tag{2.27}$$

This mains that, the set of Kraus operators $\{R_k E_i P\}$ is equivalent to the set of Kraus operators $\{\sqrt{c}P\}$. By unitary equivalence, we then have that there must exist a c_{ki} s.t.

$$R_k E_i P = c_{ki} P \implies P E_i^\dagger R_k^\dagger R_k E_j P = c_{ki}^* c_{kj} P \tag{2.28}$$

Sum over k and use $\sum_k R_k^\dagger R_k = 1$ to get,

$$P E_i^\dagger E_j P = \lambda_{ij} = \sum_k c_{ki}^* c_{kj} \tag{2.29}$$

Thus, we have showed that if there exists a correction channel, the codespace and error operators must satisfy the KL conditions.

□

The error correction operation constructed in this proof, as adapted from [NC10], can be used to show that they correct errors whose Kraus operators are linear combinations of E_i's in a very similar fashion. This is the same statement as the fact that $\mathfrak{E}(\mathcal{C}, \mathcal{E})$ is linearly closed. Moreover, we can do away with quantum channels in some sense, and just talk about a set of correctable error operators $\{E_i\}$ and use this linearity condition suitably.

One could note that we explicitly talked about *correcting* errors till yet. But in the classical case, we had separate notions of error correction and detection, which had intimate connections with the way the code distance is defined. Which brings us to another problem – how do we define the distance of a quantum

code? We cannot take Hamming distance here (why?). Let us first talk about error detection.

Consider a Pauli operator O. For instance, O could be $X \otimes Y \otimes I \otimes I \otimes Z$ for five qubits. We make away with some definitions.

Definition 2.4 (A-Weight of an operator)**.** The A-weight, for $A \in \{X, Y, Z\}$ of a Pauli operator O is defined as the number of qubit locations wherein O acts as A. This is denoted $wt_A(O)$.

Definition 2.5 (Weight of an operator)**.** Define the weight of a Pauli operator O as

$$wt(O) := wt_X(O) + wt_Y(O) + wt_Z(O) \tag{2.30}$$

Now, let us take a step back and think about what it means for an error to be detectable. If an error E occurs on a codeket $|c\rangle$, it would be detectible if $E|c\rangle$ had no overlap with any other codeket. Formally, we have that,

Theorem 2.6. The codespace $\mathcal{C}$ is E_μ-detectable if

$$P_\mathcal{C} E_\mu P_\mathcal{C} = \lambda_\mu P_\mathcal{C} \tag{2.31}$$

Now, recall that in the classical case, a code of distance d can detect $d-1$ errors. One can guess what we do next.

Definition 2.7 (Code distance)**.** We say that $\mathcal{C}$ is a d-distance code if d is the smallest integer such that

$$P_\mathcal{C} O_\mu P_\mathcal{C} = \lambda_\mu P_\mathcal{C} \tag{2.32}$$

holds true for all Pauli errors with $wt(O_\mu) < d$ but not for $wt(O_\mu) = d$.

If one relates the detection conditions to the correction conditions, one makes the identification $E_\mu \sim E_a E_b$. By noting that the weight of two operator multiplied is bounded above by the sum of their weights, we come back to the classical statement that

$$\boxed{\textit{A code capable of detecting } d-1 \textit{ errors can correct for } t := \lfloor (d-1)/2 \rfloor \textit{ errors.}} \tag{2.33}$$

§§2.3. Quantum Stabilizer Codes

Consider the n-qubit Pauli group, $\mathcal{P}_n := \{\alpha \cdot A_1 A_2 \cdots A_n : A_i \in \{I, X, Y, Z\}, \alpha \in \{\pm 1, \pm\iota\}\}$ We make another definition for convenience. [Normalizer] Given a subgroup H of a group G, the normalizer $\mathcal{N}(H)$ of H is defined as

$$\mathcal{N}(H) := \{g \in G : gh = hg \forall h \in H\} \tag{2.34}$$

Consider an Abelian subgroup of the Pauli group, $\mathcal{S} \subseteq \mathcal{P}_n$. Let $\mathcal{S}$ be generated by $S_1, S_2, \cdots, S_r$, that is $\mathcal{S} = \langle S_1, S_2, \cdots, S_r \rangle$. Consider the subspace (prove!) of the Hilbert space which is *stabilized* by this subgroup, i.e.,

$$\mathcal{C} := \{|\psi\rangle : S_i |\psi\rangle = +1 |\psi\rangle \, \forall i\} \tag{2.35}$$

What if we want to use this as our codespace? Note that, we must not have $-I \in \mathcal{S}$ as $(-I)|\psi\rangle = |\psi\rangle \implies |\psi\rangle = 0$. What would be the dimension of the codespace? Each stabilizer generator is a Pauli element, with eigenvalues in $\{\pm 1\}$. Thus, each S_i partitions $\mathcal{H}$ into two pieces, labelled by the eigenvalue. If one carries on this partitioning, one finds that $\mathcal{C}$ is of size 2^{n-r}. Thus, such a code is encoding $k := n - r$ qubits. In general, X-flavoured stabilizer generators detect phase-flips, and Z-flavoured generators detect bit-flips.

Error syndrome: As discussed in classical ECC, an error syndrome is a glorified parity check, which we performed using a coset decomposition in the classical case. Here, in the case of stabilizer codes, we see that the syndrome of a state $|\psi\rangle$ is nothing but a r-dimensional vector over $\{\pm 1\}$ which is attained by measuring S_i in the state $|\psi\rangle$. If the state is in one of the cosets, this measurement will be deterministic. If the state is in a superposition over the coset subspaces, then such a measurement will collapse the state into one of the cosets.

Effect of errors: Let us start by considering the effect of Pauli errors on our stabilizer states. Note that, in the Pauli algebra, two operators either commute or anticommute. Consider error $E \in \mathcal{P}_n$; we have three cases,

1. $E \in \mathcal{S}$: Then nothing happens! We are good to go.
2. $[E, S_j]_+ = 0$ atleast for one j. Then, this error takes us to another coset, and we flag a error through the syndrome being -1 atleast in one (the j^{th}) place.
3. $E \in \mathcal{N}(\mathcal{S}) - \mathcal{S}$: Such an error takes the state to *another* state in the codespace. Thus, this is a non-detectable error. Viewed another way, this could be a *logical operator* on the codespace.

With this, we define the distance of a code as,

$$d := \min_{E \in \mathcal{N}(\mathcal{S}) - \mathcal{S}} wt(E) \tag{2.36}$$

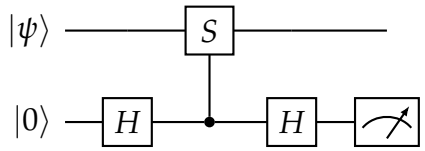
where $wt(E)$ is the number of non-identity single qubit errors in E. For instance, $wt(XXIIZ) = 3$.

2.3.1 Knill-Laflamme Conditions

2.3.2 Operational treatment

Given the stabilizer formalism and the general procedure of error correction, what steps do we take in one cycle of quantum error correction?

1. *Prepare codespace* using the stabilizer formalism
2. *Allow errors to act*
3. *Diagnose errors* by measuring all stabilizer generators to get the syndrome of the error. For one generator, this is done by adding an ancilla qubit in an interferometer, so as to measure without collaps, in a setup as follows:



   where $|0\rangle$ is the ancillla state.
4. *Decoding* is the procedure of determining which was the original state, or equivalently, which error happened. Two strategies:
   - Strategy A: Find the most likely error given the syndrome data s

$$E^* := \arg\max_E \mathbb{P}(E \mid s) \tag{2.37}$$

   - Strategy B: Find the most likely degenerate error. The idea is that two different errors, if composing to identity $E_1E_2 = 1$ on the codespace can be corrected by applying the other error instead of necessarily searching for the same error. This can happen if $E_1E_2 = S$ or equivalently $E_2 = E_1S$. Thus given the syndrome s we have,

$$E^* := \arg\max_{E\in\mathcal{P}_n} \mathbb{P}(ES \mid s : S \text{ is a stabilizer generator}) \tag{2.38}$$

   - *Apply the correction* and go home.

§§2.4. Examples of quantum codes

2.4.1 The 3-qubit bit-flip code

Consider the encoding,

$$|0\rangle_L := |000\rangle\,,\ |1\rangle_L := |111\rangle \tag{2.39}$$

This is a $[n = 3, k = 1, d = 3]$ code. This can correct $t = \lfloor (d-1)/2 \rfloor = 1$ error. Specifically, the code space $\mathcal{C} = \text{span}\{|000\rangle\,, |111\rangle\}$ is invariant under Z errors on any qubit, but X errors take $\mathcal{C}$ to another coset in the Hilbert space. Thus, this code can detect any X error on a *single* qubit.

2.4.2 The 3-qubit phase-flip code

Consider the encoding,

$$|0\rangle_L := |+++\rangle\,,\ |1\rangle_L := |---\rangle \tag{2.40}$$

This is a $[n = 3, k = 1, d = 3]$ code. This can correct $t = \lfloor (d-1)/2 \rfloor = 1$ error. Just similar albeit contrast to the bit-flip code, this code can correct *phase flips*, that is Z errors, but not X errors on any single qubit.

2.4.3 The Shor Code

This is the simplest example of a *concatenated* code. The bit-flip code can be concatenated with the phase flip code to form the codewords,

$$|0\rangle_L := (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \otimes (|000\rangle + |111\rangle) \tag{2.41}$$
$$|1\rangle_L := (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \otimes (|000\rangle - |111\rangle) \tag{2.42}$$

This is a $[9, 1, 3]$ code, capable of detecting *any* error on a single qubit.

2.4.4 Four qubit detection code

Consider,

$$|0\rangle_L := |0000\rangle - |1111\rangle \tag{2.43}$$
$$|1\rangle_L : + |1100\rangle + |0011\rangle \tag{2.44}$$

This is a $[4, 1, 2]$ code, thus cannot correct any errors. But, it is capable of detecting $d - 1 = 1$ single qubit errors. See that this is the stabilizer code generated by $\langle XXXX, ZIZI, IZIZ\rangle$.

§3. Operator Quantum Error Correction

References

[CPG23] Avimita Chatterjee, Koustubh Phalak, and Swaroop Ghosh. *Quantum Error Correction For Dummies*. 2023. arXiv: 2304.08678 [quant-ph].

[Gai13] Frank Gaitan. *Quantum Error Correction and Fault Tolerant Quantum Computing (1st ed.)* 2013. DOI: https://doi.org/10.1201/b15868.

[Got09] Daniel Gottesman. *An Introduction to Quantum Error Correction and Fault-Tolerant Quantum Computation*. 2009. arXiv: 0904.2557 [quant-ph].

[JM23] A. Jayashankar and P. Mandayam. "Quantum Error Correction: Noise-Adapted Techniques and Applications". In: *J Indian Inst Sci 103, 497–512* (2023). DOI: https://doi.org/10.1007/s41745-022-00332-x.

[NC10] Michael A. Nielsen and Isaac L. Chuang. *Quantum Computation and Quantum Information: 10th Anniversary Edition*. Cambridge University Press, 2010. DOI: 10.1017/CBO9780511976667.

[Rof19] Joschka Roffe. "Quantum error correction: an introductory guide". In: *Contemporary Physics* 60.3 (2019), pp. 226–245. DOI: 10.1080/00107514.2019.1667078. URL: https://doi.org/10.1080%2F00107514.2019.1667078.