1 The Goal

Recall our purpose - building a binary code with constant relative rate and distance. The Reed-Solomon code achieves our goal modulo the binary requirement. In order to fix this issue, we will use a concatenation technique where the outer coding is done via RS\[n = q - 1, k = rm, d = \delta n\] with some constant relative weight and distance with an inner code which we now present.

2 The Inner Code

Consider the following family of codes Cα[2m, m]² for α ∈ \( F_q^* \) and let \( \{\alpha_i\}_{i=1}^{q-1} \) be some enumeration of \( F_q^* \). For a code \( C_\alpha \) given an input \( x \in \{0, 1\}^m \) we consider \( x \) as an element in \( F_q \) and we output \( C_\alpha(x) = (x, \alpha x) \) where multiplication is done in \( F_q \) and the representation of the output is given in binary form. We use this family of inner codes for concatenation by taking \( C_\alpha \) as the inner code for the \( i \)th block of our output. Note that this technique differs from concatenation techniques we’ve seen before, as we use a different inner code for each block. Formally, for an input \( x \in F_q^m \) let \( p_x \) be the RS polynomial \( p_x(\alpha) = \sum_{i=0}^{k-1} x_i \alpha^i \) and our concatenated code outputs:

\[
JUS(x) = C_{\alpha_1}(RS(x)_1) \circ ... \circ C_{\alpha_n}(RS(x)_n) = ((p_x(\alpha_1), \alpha_1 p_x(\alpha_1)), ..., (p_x(\alpha_n), \alpha_n p_x(\alpha_n)))
\]

Note: in the code we use \( \alpha_i \) for both the \( i \)th coordinate of the outer RS code and the \( i \)th inner block coordinate, though this is not mandatory. We can pick any two enumerations of \( F_q^* \) and use one in the outer code and the other in the inner code. We now want to show that for most \( \alpha \), \( C_\alpha \) achieves a constant relative distance. To show this, we fix \( \delta_0 \) s.t. \( 2H(\delta_0) < 1 \) (one can verify that any \( \delta_0 \leq 0.1 \) works) and perform some computations. We begin with a definition that will characterise a "bad" encoded block.

**Definition 1.** Fix \( m \) and let \( q = 2^m \) and \( \delta_0 \) as above. We call \( \alpha \in F_q^* \) **bad** if there exists \( x, y \in F_q^* \) s.t. \( x, y \in B(0, \delta_0 m) \) and \( y = \alpha x \). Note: Each such pair \( x, y \) define a single \( \alpha \), as \( \alpha = xy^{-1} \)

Next, we show that a random \( \alpha \) is not bad WHP:

**Claim 2.** For \( \delta_0 \) s.t. \( 2H(\delta_0) < 1 \) we have \( \varepsilon \equiv \Pr[\alpha \text{ is bad}] \leq 2^{-\Omega(m)} \)

**Proof** As each such \( \alpha \) is given by a unique pair \( x, y \) we can bound the probability by choosing pairs from the hamming ball \( B(0, \delta_0 m) \), thus:

\[
\Pr[\alpha \text{ is bad}] = \frac{|B(0, \delta_0 m)|^2}{q - 1} \leq \frac{(2^{H(\delta_0)m})^2}{2^m - 1} \approx \frac{2^{2H(\delta_0)m}}{2^m} = 2^{(2H(\delta_0) - 1)m} = 2^{-\Omega(m)}
\]

Next, we show that if \( \alpha \) is not bad, then our inner code achieves the desired distance:

**Claim 3.** If \( \alpha \) is not bad, then \( \text{wt}(x, \alpha x) \geq \delta_0 m \)
Proof Assume that \(\text{wt}(x, \alpha x) < \delta_0 m\), then it follows that \(\text{wt}(x, \alpha(x)) < \delta_0 m\) and therefore \(x, \alpha x \in B(0, \delta_0 m)\) which implies that \(\alpha\) is bad by definition.

Corollary 4. If \(\alpha\) is not bad then \(C_\alpha\) is a \([2m, m, \delta_0 m]_2\) code.

We note that by this corollary, if we could deterministically find such an \(\alpha\) then we will have achieved our goal - a binary code with constant relative distance and rate. Alas, though these \(\alpha\)'s are abundant, deterministically pointing at one is hard. By using all possible \(\alpha\)'s in our inner coding we ensure that in most cases the inner blocks have good properties.

Corollary 5. There is at most a fraction \(\varepsilon\) of elements \(y \in \mathbb{F}_q^*\) s.t. \(\text{wt}(y, \alpha y) < \delta_0 m\)

3 The Justensen code parameters

All we have left is the computation of the new code parameters. Let \(\text{JUS}[N, K, D]_2\) denote our new code, which has an outer RS\([n, rn, \delta n]_q\) code and inner \(C_\alpha\) which is a \([2m, m, \frac{3m}{2}]_2\) code for most blocks, and we observe:

- As each block \(y_i = p_x(\alpha_i) \in \mathbb{F}_q\) is encoded by \(m\) bits and is mapped to \((y_i, y_i \alpha_i)\), clearly \(N = n \cdot 2m\)
- As the relative rate of \(C_\alpha\) is \(1/2\), we have \(K = \frac{\varepsilon}{2} N\)
- Finally, for the distance, due to the RS properties, there are at least \(\delta n\) blocks \(y_i\) s.t. \(y_i \neq 0\). Out of these blocks, a fraction of at most \(\varepsilon = 2^{-\Omega(m)}\) give an encoded block with \(\text{wt}(y_i, \alpha_i y_i) < \delta_0 m\), thus:

\[
D \geq (\delta - \varepsilon)n \cdot \delta_0 m = \frac{N}{2} (\delta - \varepsilon) \delta_0
\]

Note that \(2^{-\Omega(m)} = o(1)\) as \(m \to \infty\), so we get:

\[
D \geq \frac{N}{2} (\delta - o(1)) \delta_0
\]

And so, picking for example \(\delta_0 = 0.1\) we get an \([N = 2nm, K = N \frac{\varepsilon}{2}, D = N \frac{\delta - o(1)}{20}]_2\) code, with constant relative distance and rate as required.

Lastly, we recall that in the outer RS code we have \(d = n - k + 1\) and so \(\delta = 1 - r + \frac{1}{2} = 1 - r + o(1)\), and so we can rewrite our code parameters as \(\text{JUS} \left[ N = 2nm, K = N \frac{\varepsilon}{2}, D = N \frac{1-r-o(1)}{20} \right]_2\)