In this lecture, we will study the information theoretic aspects of Reed-Solomon codes and concatenated codes. One should always remember the aim of constructing good codes:

*Thou shalt strive for constant rate and constant relative distance codes.*

## 1 Review of Reed-Solomon Codes

Reed-Solomon (RS) Codes are an example of linear codes. If we fix the dimension $k$ and a field $\mathbb{F}_q$ as the alphabet, where $q$ is a prime power (usually a power of 2), then the RS encoding maps a $q$-ary message $m = (m_0, \ldots, m_{k-1})$ of length $k$ to a $q$-ary codeword $c = (c_0, \ldots, c_{n-1})$ of length $n$ ($\leq q$). The RS codeword corresponds to some bounded degree polynomial $p : \mathbb{F}_q \rightarrow \mathbb{F}_q$ being evaluated at field elements. The set of RS-codewords can be written as $C_{RS} = \{ p(\mathbb{F}_q^* ) \mid \deg(p) \leq k - 1 \}$. The polynomial $p$ is usually defined with respect to the message $m$ as

$$ p_m(x) = \sum_{i=0}^{k-1} m_i x^i, $$

with the codeword then being $c_m = \{ p_m(\alpha) \mid \alpha \in \mathbb{F}_q^* \}$. In general, some subset of the field is sufficient for evaluation. In this case, the codeword becomes $c_m = \{ p_m(\alpha) \mid \alpha \in A \}$, where $A = \{ \alpha_0, \ldots, \alpha_{n-1} \} \subseteq \mathbb{F}_q^*$, and the set of RS-codewords can be rewritten as $C_{RS} = \{ (p(\alpha_0), \ldots, p(\alpha_{n-1})) \mid \deg(p) \leq k - 1 \}$.

One can also define other types of polynomials corresponding to messages, thus giving rise to different types of RS codes. For example, we could have $\forall i \ p_m'(\alpha_i) = m_i$ as our polynomial. Defining it this way has efficiency advantages in some situations.

### 1.1 Parameters of RS Codes

Let us take a closer look at the parameters and the properties of Reed-Solomon codes.

**Alphabet Size** $= q$.

**Block Length** $= n$ ($\leq q$).

**Dimension** $= k$.

*Theorem 1.1.* RS code is a linear code.

*Proof.* Consider $p(x), p'(x) \in C_{RS}$ and $a, b \in \mathbb{F}_q$. Then $ap(x) + bp'(x)$ still has degree $\leq k - 1$ and is thus in $C_{RS}$. \qed
We can define the generating matrix of RS codes, i.e., \( G_m \). Here, \( G \) takes the form of an \( n \times k \) Vandermonde matrix.

\[
\begin{pmatrix}
\alpha_0^0 & \alpha_1^0 & \cdots & \alpha_{k-1}^0 \\
\alpha_0^1 & \alpha_1^1 & \cdots & \alpha_{k-1}^1 \\
\vdots & \vdots & \ddots & \vdots \\
\alpha_0^{n-1} & \alpha_1^{n-1} & \cdots & \alpha_{k-1}^{n-1}
\end{pmatrix}
\begin{pmatrix}
m_0 \\
m_1 \\
\vdots \\
m_{k-1}
\end{pmatrix}
= \begin{pmatrix}
\sum_{i=0}^{k-1} m_i \alpha_i^0 \\
\sum_{i=0}^{k-1} m_i \alpha_i^1 \\
\vdots \\
\sum_{i=0}^{k-1} m_i \alpha_i^{n-1}
\end{pmatrix}
\]

Rate \( R = \frac{\# \text{ information bits}}{\# \text{ bits in codeword}} = \frac{k \log q}{n \log q} = \frac{k}{n} \).

Distance \( d = n - (k - 1) \). This is because two distinct polynomials of degree \( \leq k - 1 \) can agree on at most \( k - 1 \) points (see notes of previous lecture).

Recall that this distance exactly matches the Singleton bound (\( k \leq n - d + 1 \) or \( d \leq n - k + 1 \)).

**Definition 1.2.** Maximum Distance Separable (MDS) codes are codes matching the Singleton bound.

Then the following claim is straightforward.

**Claim 1.3.** Reed-Solomon codes are MDS codes.

Reed-Solomon codes are optimal between rate and distance, but we pay for that by a large alphabet size. This leads us to a question. Can we have MDS codes with a binary or a constant size alphabet? The following claim rules out this possibility.

**Claim 1.4.** MDS codes must have a superconstant alphabet size.

**Proof.** By the Plotkin bound we have

\[
k \leq n - \frac{q}{q - 1} d + O(\log d),
\]

and by the definition of MDS codes, we have

\[
k = n - d + 1.
\]

Combining 1 and 2 gives

\[
\frac{1}{q - 1} \leq \frac{O(\log d) - 1}{d}
\]

\[
q \geq 1 + \frac{d}{O(\log d) - 1} = \omega(1). \quad \square
\]

So then what is the minimal \( q \) that we can hope to achieve? By the previous proof, we have \( q = \Omega\left(\frac{d}{\log q}\right) + 1 \). We want \( d \) to be a constant fraction of the length for a good code. If \( d = \Omega(n) \) then we get \( q = \Omega\left(\frac{n}{\log n}\right) \). But for RS codes, \( q \geq n \), i.e., the best we can do is to shave some \( \log n \) factor off the alphabet size.
1.2 Alphabet size does matter

A large alphabet size is undesirable because it abuses the noise model. If we have $\sigma \in \mathbb{F}_q$ to send, we can think of $\sigma$ as (say) some binary string of length $\log q$: $(\sigma_1, \ldots, \sigma_{\log q})$. Then the number of information bits is $k \log q$ and the number of bits per codeword is $n \log q$. But then changing the codeword into another one might require the manipulation of just one bit per symbol, making the distance $n - k + 1$. The relative distance of the code now becomes

$$\delta = \frac{n - k + 1}{n \log q} = \Theta\left(\frac{n}{\log q}\right) \rightarrow 0.$$ 

We thus have a vanishing distance, which is horrible!

One can try to get around this problem by changing the noise model, but even this will fail. Say now the noise model is that one can only corrupt some number of whole symbols. Then we do not need the mechanism of RS codes at all! To correct one error, we just implement the naive solution of repeating each symbol three times.

$$c_{m^i} = (m^i, m^i, m^i) \xrightarrow{\text{channel}} (m^i, *, m^i) \xrightarrow{\text{decode}} m^i$$

Thus all the work gets pushed into having a large alphabet.

Even in this form, Reed-Solomon codes are useful for secret sharing. Also for noise models such as Burst Noise, many whole symbols might be ruined and we can correct a lot of these errors. RS codes are used on numerous digital media, like CDs.

2 Concatenated Codes

Our goal is to reduce the alphabet size for RS codes. We have already seen one trick, binary representation, which did not give a good relative distance. In this section, we will study the idea of concatenation of codes.

Concatenation uses two codes to perform the encoding. Say the output of the first code (let us call this the outer code) applied to the message is the codeword $(\sigma_0, \ldots, \sigma_{n - 1})$. Now each $\sigma_i$ can be interpreted as a string in a smaller alphabet. Then we can use the second code (inner code) to encode each $\sigma_i$ using a smaller alphabet. The final codeword is the concatenation of all these inner codewords. Typically, the outer code is the RS code, which encodes over a large alphabet, and the inner code is some binary encoding procedure.

**Definition 2.1** (Concatenated Codes). The concatenation $C$ of two codes $C_{\text{out}} : [Q]^K \rightarrow [Q]^N$ and $C_{\text{in}} : [q]^k \rightarrow [q]^n$, where $Q = q^K$, is the code $C_{\text{out}} \circ C_{\text{in}}$ defined as

$$C(m) = (C_{\text{in}}((C_{\text{out}}(m))_0), \ldots, C_{\text{in}}((C_{\text{out}}(m))_{N - 1}))$$

where each $(C_{\text{out}}(m))_i$ is mapped to a $q$-ary string of length $k$.

The next claim is easy to prove.

**Claim 2.2.** If $C_{\text{out}}$ and $C_{\text{in}}$ are linear and we use a linear mapping between the two alphabets, then $C = C_{\text{out}} \circ C_{\text{in}}$ is also linear.

We can derive the parameters of the concatenated code from the parameters of the constituent codes.

**Theorem 2.3.**

2.1 Ideas for a good $C_{\text{in}}$

2.2 Justensen Codes