Reed–Solomon codes and list decoding

Last modified: 2026-11-12 01:15:28 (PST)

A Reed–Solomon code recovers the message when fewer than $d/2$ symbols are corrupted, where d is its minimum distance. Past that point a received word can be near several codewords, and a decoder can return all of them as a list.

List decoding

Definition 0.1 (List-decodable code)**.** Let C be a block code of length n over the alphabet Σ, and let $e \geq 0$ and $L \geq 1$ be integers. Then C is (e, L)**-list-decodable** if every received word $r \in \Sigma^n$ has at most L codewords c with $\Delta(c, r) \leq e$. To **list decode** a received word r is to find all of those codewords.

Example 0.1 (A received word with two codewords nearby)**.**
Take the Reed–Solomon code with $q = n = 5$ and $k = 2$ from the small example, whose codewords are the values of the 25 lines $m_0 + m_1 t$ at the points $0, 1, 2, 3, 4$. Its minimum distance is 4, so every received word has at most one codeword within distance 1. Within distance 2 it can have more. The R chunk below lists the lines within distance 2 of the received word $r = (1, 3, 1, 0, 4)$.

```
points <- 0:4
lines <- expand.grid(m0 = 0:4, m1 = 0:4)
codeword <- function(m0, m1) (m0 + m1 * points) %% 5
received <- c(1, 3, 1, 0, 4)
distance <- function(m0, m1) sum(codeword(m0, m1) != received)
lines$distance <- mapply(distance, lines$m0, lines$m1)
lines[lines$distance <= 2, ]
#>    m0 m1 distance
#> 12  1  2        2
#> 24  3  4        2
```

Two lines are within distance 2 of r, so this received word has a list of size 2 at distance 2. A decoder that outputs one codeword must choose between them.

Theorem 0.1 (A bound on the list size of a Reed–Solomon code)**.** *Let C be the Reed–Solomon code with length n and k message symbols, and let $e \in \{0, 1, \ldots, n-k\}$. Then every received word has at most*

$$\frac{\binom{n}{k}}{\binom{n-e}{k}}$$

codewords within Hamming distance e. So C is (e, L)-list-decodable for every L at least the value above.

Proof

Proof. Fix a received word r and let S be the codewords c with $\Delta(c, r) \leq e$. Count the pairs (c, T), where $c \in S$ and T is a choice of k positions at which c and r agree.
Each $c \in S$ differs from r in at most e positions, so it agrees with r in at least $n - e$ positions. Since $n - e \geq k$, it has at least $\binom{n-e}{k}$ choices of T. So there are at least $|S|\binom{n-e}{k}$ pairs.
Now fix a choice T of k positions. Two distinct codewords that both agree with r on T agree with each other on T. By the distance theorem, distinct codewords differ in at least $n - k + 1$ positions, so they agree in at most $k - 1$ positions, which is fewer than k. So at most one codeword c agrees with r on T. There are $\binom{n}{k}$ choices of T, so there are at most $\binom{n}{k}$ pairs.
Combine the two counts:

$$|S| \binom{n-e}{k} \leq \binom{n}{k}.$$

Divide both sides by $\binom{n-e}{k}$, which is nonzero because $n - e \geq k$. □

Example 0.2 (The list size for the length-5 Reed–Solomon code)**.**
Take $n = 5$, $k = 2$ and $e = 2$, as in Example 0.1. Theorem 0.1 gives $\binom{5}{2}/\binom{3}{2}$ as the largest list size.

```
choose(5, 2) / choose(3, 2)
#> [1] 3.333333
```

The list size is a whole number, so it is at most 3. The received word of Example 0.1 has 2.

> *Remark* 0.1 (Efficient list decoding of Reed–Solomon codes). Theorem 0.1 says how long the list can be, but it does not say how to find the list quickly. Guruswami and Sudan (1999) gave a polynomial-time algorithm that finds the list for a Reed–Solomon code.

References

Guruswami, Venkatesan, and Madhu Sudan. 1999. "Improved Decoding of Reed–Solomon and Algebraic-Geometry Codes." *IEEE Transactions on Information Theory* 45 (6): 1757–67. https://doi.org/10.1109/18.782097.