26. Welch-Berlekamp decoding algorithm

26.1 Introduction

In today's lecture, we look at the unique decoding problem for Reed Solomon codes, and look at the Welch-Berlekamp decoder.

26.2 Decoding Reed-Solomon codes

The decoding problem for Reed-Solomon codes is the following. Given a message $\mathbf{y} = (\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n)$ such that $\Delta(\mathbf{y}, \mathcal{C}) < \frac{n-k+1}{2}$ where $\mathcal{C}$ is an $[n, k, n-k+1]_q$ RS code, find the polynomial $P(x)$ of degree at most $k-1$ such that $P(\alpha_i) \neq \mathbf{y}_i$ for at most $e = \lfloor \frac{n-k+1}{2} \rfloor$ many values of α_i.

To this end we define an error locator polynomial $E(x)$ such that $E(\alpha_i) = 0$ whenever $P(\alpha_i) \neq \mathbf{y}_i$. In fact, polynomial

$$E(x) = \prod_{\alpha_i | \mathbf{y}_i \neq P(\alpha_i)} (x - \alpha_i)$$

is such a polynomial, and it has degree at most e. But, if we don't have P, we also don't have E and this definition seems useless at the moment. Also observe that $\mathbf{y}_i E(\alpha_i) = P(\alpha_i)E(\alpha_i)$ for all $i \in [n]$(why?). If we were given the polynomial E, then if we take the coefficients of $P(x)$ as unknowns, we get n linear equations over k unknowns where $k \leq n$. This would give us the corrected message we want. But what if we think of the coefficients of $P(x)$ and $E(X)$ as unknowns. Then we have n equations over $k+e+1$ variables, but the equations are no longer linear. Define $N(x) = P(x)E(x)$. The way this is written, all the polynomials are unknown to us and it seems as though this will not give us anything. The Welch-Berlekamp decoder that we describe next shows that we can forget about $P(x)$ and just try to find two polynomials $N(x)$ and $E(x)$ with certain properties and then obtain $P(x) = N(x)/E(x)$.

26.2.1 The Welch-Berlekamp decoder

The algorithm can be explained simply as follows: Given a vector $\mathbf{y} \in \mathbb{F}_q^n$, find polynomials $E(x)$ of degree $e < (n-k+1)/2$ and $N(x)$ of degree at most $e+k-1$ such that for all $i \in \{1, 2, \ldots, n\}$, $N(\alpha_i) = \mathbf{y}_i E(\alpha_i)$, degree of E is e and degree of $N(\alpha_i)$ is at most $e+k-1$. If we find such polyomials $N(x)$ and $E(x)$, return $P(x) = N(x)/E(x)$. Else the algorithm fails. Observe that finding the coefficients of $N(x)$ and $E(x)$ is solving n linear equations with $(e+1) + (e+k)$ unknowns.

Now, we need to prove the correctness of this procedure. First observe the following simple lemma

Lemma 26.1. *There exists polynomials $E^\star(x)$ and $N^\star(x)$ such that $\mathbf{y}_i E^\star(\alpha_i) = N^\star(\alpha_i)$ for all $i \in [n]$, and $P(x) = N^\star(x)/E^\star(x)$.*

Proof. Choose the following polynomials.

$$E^\star(x) = x^{e-\Delta(\mathbf{y},\mathcal{C})} \prod_{\mathbf{y}_i \neq P(\alpha_i)} (x - \alpha_i), \text{ and}$$
$$N^\star(x) = P(x)E^\star(x).$$

Observe that $E^\star$ is a non-zero polynomial of degree e, and degree of $N^\star$ is at most $e+k-1$. Furthermore, for every $\mathbf{y_i}$, one of the two happens:

- If $\mathbf{y}_i \neq P(\alpha_i)$, then $E^\star(\alpha_i) = 0$, and therefore $N^\star(\alpha_i) = \mathbf{y}_i E^\star(\alpha_i)$.
- If $\mathbf{y}_i = P(\alpha_i)$, then $N^\star(\alpha_i) = P(\alpha_i)E^\star(\alpha_i) = \mathbf{y}_i E^\star(\alpha_i)$.

This shows that there is some solution for $E(x)$ and $N(x)$ that gives the correct $P(x)$. □

To complete the proof of correctness of the decoder, it is sufficient to show the following.

Lemma 26.2. *Let (E_1, N_1) and (E_2, N_2) be two different solutions such that $N_1(\alpha_i) = \mathbf{y}_i E_1(\alpha_i)$ and $N_2(\alpha_i) = \mathbf{y}_i E(\alpha_i)$. Then $N_1/E_1 = N_2/E_2$.*

Proof. Let $R(x) = N_1(x)E_1(x) - N_2(x)E_1(x)$. We will show that $R(x)$ is the zero polynomial. For any α_i, $N_1(\alpha_i)E_1(\alpha_i) = \mathbf{y}_i E_1(\alpha_i)E_2(\alpha_i)$ and $N_2(\alpha_i)E_1(\alpha_i) = \mathbf{y}_i E_2(\alpha_i)E_1(\alpha_i)$. Therefore $R(x)$ has at least n roots. But, notice that the degree of $R(x)$ is at most $e + k - 1 + e = 2e + k - 1 < n$. But this is not possible unless $R(x)$ is the identically zero polynomial. □

Implementing the algorithm

Notice that to implement the algorithm, we need to enforce that degree of E is exactly 1. This can be done by adding a new constraint that the coefficient corresponding to x^e is 1. The set of equations has at most $n + 1$ unknowns and $n + 1$ equations. The standard Guassian elimination algorithm gives an $O(n^3)$ algorithm to solve for $N(x)$ and $E(x)$. To compute $P(x)$, we need to perform long division. So the total running time of this algorithm is $O(n^3)$.