1 Berlekamp-Welch Warm Up

(a) When does $r_i = P(i)$?

(b) When does r_i not equal $P(i)$?

(c) If you want to send a length-n message, what should the degree of $P(x)$ be? Why?

(d) If there are at most k erasure errors, how many packets should you send?

(e) If there are at most k general errors, how many packets should you send? (We will see the reason for this later.) Now we will only consider general errors.

(f) What do the roots of the error polynomial $E(x)$ tell you? Does the receiver know the roots of $E(x)$?

(g) If there are at most k errors, what is the maximum degree of $E(x)$?

(h) Using the information about the degree of $P(x)$ and $E(x)$, what can you conclude about the degree of $Q(x) = P(x)E(x)$?

(i) Why is the equation $Q(i) = P(i)E(i) = r_iE(i)$ always true? (Consider what happens when $P(i) = r_i$, and what happens when $P(i)$ does not equal r_i.)

(j) In the polynomials $Q(x)$ and $E(x)$, how many total unknown coefficients are there? (These are the variables you must solve for. Think about the degree of the polynomials.)

(k) When you receive packets, how many equations do you have? Do you have enough equations to solve for all of the unknowns? (Think about the answer to the earlier question - does it make sense now why we send as many packets as we do?)

(l) If you have $Q(x)$ and $E(x)$, how does one recover $P(x)$?

(m) If you know $P(x)$, how can you recover the original message?

2 Berlekamp-Welch for General Errors

Suppose that Hector wants to send you a length $n = 3$ message, m_0, m_1, m_2, with the possibility for $k = 1$ error. For all parts of this problem, we will work mod 11, so we can encode 11 letters as shown below:

A	B	C	D	E	F	G	H	I	J	K
0	1	2	3	4	5	6	7	8	9	10

Hector encodes the message by finding the degree ≤ 2 polynomial $P(x)$ that passes through $(0, m_0)$, $(1, m_1)$, and $(2, m_2)$, and then sends you the five packets $P(0), P(1), P(2), P(3), P(4)$ over a noisy channel. The message you receive is

$$\text{DHACK} \Rightarrow 3, 7, 0, 2, 10 = r_0, r_1, r_2, r_3, r_4$$

which could have up to 1 error.

(a) First, let's locate the error, using an error-locating polynomial $E(x)$. Let $Q(x) = P(x)E(x)$. Recall that

$$Q(i) = P(i)E(i) = r_i E(i), \quad \text{for} \quad 0 \leq i < n + 2k.$$

What is the degree of $E(x)$? What is the degree of $Q(x)$? Using the relation above, write out the form of $E(x)$ and $Q(x)$ in terms of the unknown coefficients, and then a system of equations to find both these polynomials.

(b) Solve for $Q(x)$ and $E(x)$. Where is the error located?

(c) Finally, what is $P(x)$? Use $P(x)$ to determine the original message that Hector wanted to send.

Hint: The message refers to a US federal agency.

3 List Decoding

(a) Consider an n character message encoded into m characters over the field GF(p) using polynomials. Suppose that one receives $n-1$ of the m packets. Give a method to find a list of size at most p of all possible messages.

(b) Consider an n character message encoded into $m = n + 2k$ characters over the field GF(p) using polynomials. Suppose that $k+1$ of the m received packets are corrupted. Give a method to find a list of all possible messages which contain the original message. What is the size of the list for your scheme?

(c) Consider the protocol in (b) where we are working in GF(7). Let the original message have $n = 1$ and $k = 2$, so there are 5 symbols. Now suppose that there are 3 errors, but these three errors all landed on different values. Assume that we received: $0, 0, 1, 2, 3$. How does your list-decoding strategy perform?