

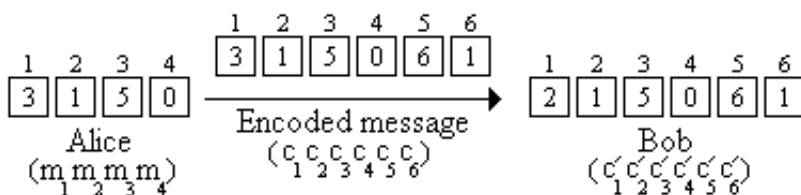
Error Correcting Codes

General Errors

We previously considered erasure resistant coded, we now consider a much more challenging scenario. Now Alice wishes to communicate with Bob over a noisy channel (say via a modem). Her message is m_1, \dots, m_n , where we will think of the m_i 's as characters (either bytes or characters in the English alphabet). The problem now is that some of the characters are corrupted during transmission due to channel noise. So Bob receives exactly as many characters as Alice transmits. However, k of them are corrupted, and Bob has no idea which k . Recovering from such general errors is much more challenging than erasure errors, though once again polynomials hold the key.

Let us again think of each character as a number modulo q for some prime q (for the English alphabet q is some prime larger than 26, say $q = 29$). As before, we can describe the message by a polynomial $P(x)$ of degree $n - 1$ over $GF(q)$, such that $P(1) = m_1, \dots, P(n) = m_n$. As before, to cope with the transmission errors Alice will transmit additional characters obtained by evaluating $P(x)$ at additional points. As we shall see shortly, in order to guard against k general errors, Alice must transmit $2k$ additional characters (as opposed to just k additional packets as was the case with erasure errors). Thus the encoded message is c_1, \dots, c_{n+2k} where $c_j = P(j)$ for $1 \leq j \leq n + 2k$, and $n + k$ of these characters that Bob receives are uncorrupted. As before, we must put the mild constraint on q that it be large enough so that $q \geq n + 2k$.

For example, if Alice wishes to send $n = 4$ characters to Bob via a modem in which $k = 1$ of the characters is corrupted, she must redundantly send an encoded message consisting of 6 characters. Suppose she wants to transmit the same message as above, and that c_1 is corrupted and changed to $c'_1 = 2$. This scenario can be visualized in the following figure:

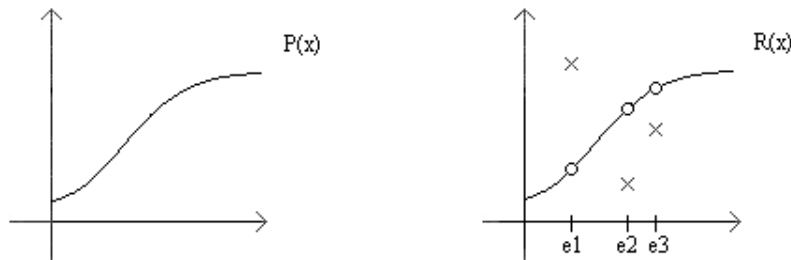


From Bob's viewpoint, the problem of reconstructing Alice's message is the same as reconstructing the polynomial $P(x)$ from the $n + 2k$ received characters $R(1), R(2), \dots, R(n + 2k)$. In other words, Bob is given $n + 2k$ values modulo q , $R(1), R(2), \dots, R(n + 2k)$, with the promise that there is a polynomial $P(x)$ of degree $n - 1$ over $GF(q)$ such that $R(i) = P(i)$ for $n + k$ distinct values of i between 1 and $n + 2k$. Bob must reconstruct $P(x)$ from this data (in the above example, $n + k = 5$ and $R(2) = P(2) = 1$, $R(3) = P(3) = 5$, $R(4) = P(4) = 0$, $R(5) = P(5) = 6$, and $R(6) = P(6) = 1$). Note, however, that Bob does not know *which* of the $n + k$ values are correct!

Does Bob even have sufficient information to reconstruct $P(x)$? Our first observation shows that the answer is yes: there is a unique polynomial that agrees with $R(x)$ at $n + k$ points. Suppose that $P'(x)$ is any polynomial of degree $n - 1$ that agrees with $R(x)$ at $n + k$ points. Then among these $n + k$ points there are at most

k errors, and therefore on at least n points x_i we must have $P'(x_i) = P(x_i)$. But a polynomial of degree $n - 1$ is uniquely defined by its values at n points, and therefore $P(x) = P'(x)$ (for all x).

But how can Bob quickly find such a polynomial? The issue at hand is the locations of the k errors. Let e_1, \dots, e_k be the k locations at which errors occurred. Note that $P(e_i) \neq R(e_i)$ for $1 \leq i \leq k$:



We could try to guess where the k errors lie, but this would take too long (it would take exponential time, in fact). Consider the so-called *error-locator polynomial* $E(x) = (x - e_1)(x - e_2) \cdots (x - e_k)$, which has degree k (since x appears k times).

Let us make a simple but crucial observation: $P(i)E(i) = R(i)E(i)$ for $1 \leq i \leq n + 2k$ (this is true at points i at which no error occurred since $P(i) = R(i)$, and trivially true at points i at which an error occurred since $E(i) = 0$). This observation forms the basis of a very clever algorithm invented by Berlekamp and Welch. Looking more closely at these equalities, we will show that they are $n + 2k$ linear equations in $n + 2k$ unknowns, from which the locations of the errors and coefficients of $P(x)$ can be easily deduced.

Let $P(x)E(x) = Q(x)$, which is a polynomial of degree $n + k - 1$, and is therefore described by $n + k$ coefficients. The error-locator polynomial $E(x) = (x - e_1) \cdots (x - e_k)$ has degree k and is described by $k + 1$ coefficients, but the leading coefficient (coefficient of x^k) is always 1. So we have:

$$Q(x) = a_{n+k-1}x^{n+k-1} + \cdots + a_1x + a_0$$

$$E(x) = x^k + b_{k-1}x^{k-1} + \cdots + b_1x + b_0$$

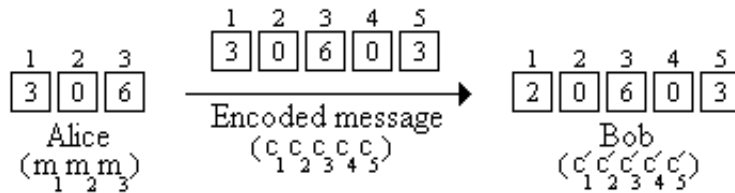
Once we fix a value i for x , the received value $R(i)$ is fixed. Also, $Q(i)$ is now a linear function of the $n + k$ coefficients $a_{n+k-1} \dots a_0$, and $E(i)$ is a linear function of the k coefficients $b_{k-1} \dots b_0$. Therefore the equation $Q(i) = R(i)E(i)$ is a linear equation in the $n + 2k$ unknowns a_{n+k-1}, \dots, a_0 and b_{k-1}, \dots, b_0 . We thus have $n + 2k$ linear equations, one for each value of i , and $n + 2k$ unknowns. We can solve these equations and get $E(x)$ and $Q(x)$. We can then compute the ratio $\frac{Q(x)}{E(x)}$ to obtain $P(x)$.

Example. Suppose we are working over $GF(7)$ and Alice wants to send Bob the $n = 3$ characters “3,” “0,” and “6” over a modem. Turning to the analogy of the English alphabet, this is equivalent to using only the first 7 letters of the alphabet, where $a = 0, \dots, g = 6$. So the message which Alice wishes for Bob to receive is “dag”. Then Alice interpolates to find the polynomial

$$P(x) = x^2 + x + 1,$$

which is the unique polynomial of degree 2 such that $P(1) = 3$, $P(2) = 0$, and $P(3) = 6$.

Suppose that $k = 1$ character is corrupted, so she needs to transmit the $n + 2k = 5$ characters $P(1) = 3$, $P(2) = 0$, $P(3) = 6$, $P(4) = 0$, and $P(5) = 3$ to Bob. Suppose $P(1)$ is corrupted, so he receives 2 instead of 3 (i.e., Alice sends the encoded message “dagad” but Bob instead receives “cagad”). Summarizing, we have the following situation:



Let $E(x) = (x - e_1)$ be the error-locator polynomial—remember, Bob doesn't know what e_1 is yet since he doesn't know where the error occurred—and let the degree 3 polynomial $Q(x) = R(x)E(x)$ for $x = 1$ to 5 . Now Bob just substitutes $x = 1, x = 2, \dots, x = 5$ to get five linear equations in five unknowns (recall that we are working modulo 7 and that $R(i) = c'_i$ is the value Bob received for the i -th character):

$$\begin{aligned} a_3 + a_2 + a_1 + a_0 + 5b_0 &= 2 \\ a_3 + 4a_2 + 2a_1 + a_0 &= 0 \\ 6a_3 + 2a_2 + 3a_1 + a_0 + b_0 &= 4 \\ a_3 + 2a_2 + 4a_1 + a_0 &= 0 \\ 6a_3 + 4a_2 + 5a_1 + a_0 + 4b_0 &= 1 \end{aligned}$$

Bob then solves this linear system and finds that $a_3 = 1, a_2 = 0, a_1 = 0, a_0 = 6$, and $b_0 = 6$ (all mod 7). (As a check, this implies that $E(x) = x + 6 = x - 1$, so the location of the error is position $e_1 = 1$, which is correct since the first character was corrupted from a “d” to a “c”.) This gives him the polynomials $Q(x) = x^3 + 6$ and $E(x) = x - 1$. He can then find $P(x)$ by computing the quotient $P(x) = \frac{Q(x)}{E(x)} = \frac{x^3+6}{x-1} = x^2 + x + 1$. Bob notices that the first character was corrupted (since $e_1 = 1$), so now that he has $P(x)$, he just computes $P(1) = 3 = \text{“d”}$ and obtains the original, uncorrupted message “dag”.

Finer Points

Two points need further discussion. How do we know that the $n + 2k$ equations are consistent? What if they have no solution? This is simple. The equations must be consistent since $Q(x) = P(x)E(x)$ together with the error locator polynomial $E(x)$ gives a solution.

A more interesting question is this: how do we know that the $n + 2k$ equations are independent, i.e., how do we know that there aren't other spurious solutions in addition to the real solution that we are looking for? Put more mathematically, how do we know that the solution $Q'(x)$ and $E'(x)$ that we reconstruct satisfies the property that $E'(x)$ divides $Q'(x)$ and that $\frac{Q'(x)}{E'(x)} = \frac{Q(x)}{E(x)} = P(x)$? To see this notice that $Q(x)E'(x) = Q'(x)E(x)$ for $1 \leq x \leq n + 2k$. This holds trivially whenever $E(x)$ or $E'(x)$ is 0, and otherwise it follows from the fact that $\frac{Q'(x)}{E'(x)} = \frac{Q(x)}{E(x)} = R(x)$. But the degree of $Q(x)E'(x)$ and $Q'(x)E(x)$ is $n + 2k - 1$. Since these two polynomials are equal at $n + 2k$ points, it follows that they are the same polynomial, and thus rearranging we get that $\frac{Q'(x)}{E'(x)} = \frac{Q(x)}{E(x)} = P(x)$.