13. POLYNOMIAL CODES

§13.1. Error Detection and Error Correction

Generally when people talk of *codes* they're thinking of cryptography where messages are coded for reasons of secrecy. But there's another reason why we might want to code a message. We might want to transmit across a 'noisy' channel, that is, a channel that could corrupt the text.

A visual example of data being corrupted occurs when a picture is sent. An image is transmitted as a sequence of 1's and 0's and if there is corruption during transmission the received picture is not as clear. Sending it via an error correcting code can ensure that it is received with its original clarity.

Of course we could simply repeat the message. Wherever the two copies differ, the receiver will know that an error has occurred. But they won't know, for each error, which version is correct. We'd need to transmit the message *three* times for the receiver to be able to correct the error – whichever alternative occurs twice will almost

certainly be correct. However, having to transmit each message three times seems rather inefficient.

One very widespread system of error detecting is the ASCII code where every letter, digit and punctuation symbol has a 7 bit code. An 8'th bit (called a **check bit**) is sent after each seven, based on how many 1's there are in the 7 bits. In one such system, if the number of 1's is odd (i.e. 1, 3, 5 or 7) the 8'th bit is '1' and if the number of 1's is even (i.e. 0, 2, 4 or 6) the 8'th bit would be '0'. This rule ensures that the number of 1's in each block of 8 bits is always even.

When the message is received the number of 1's in each block is counted. If there's a single error in a block (that is, a '0' being changed to a '1' or vice-versa) the number of 1's will be odd. The receiver will then know that an error has taken place in that block and, if possible, will ask the transmitter to send that block again. This system is widely used when a computer communicates with another device.

The system is very useful but it has two problems.

(1) If there are 2 errors (or 4 or 6) the errors will go undetected. In practice, if the error rate is low and the blocks are small, the chance of getting 2 errors in a single block is small enough to be ignored.

(2) There needs to be 2-way communication. In many applications this is not feasible.

In a compact disc player, the music is picked up digitally as pulses that represent 0's and 1's. If an error is detected there's no opportunity to say "hey run that by me again!" The laser head has had to move on.

The code that's described here is known as the Reed-Solomon Code. It uses binary polynomials over the field $\mathbb{Z}_2$.

§ 13.2. Polynomials

A **polynomial** is an expression of the form:

$$a(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0.$$

The symbol x is called an **indeterminate** and simply plays the role of a place marker. The role of the x is to provide positions in the expression which can be replaced (substituted) by a value. The numbers a_0, a_1,... are called the **coefficients** of the polynomial.

The coefficients can be rational numbers (from the set $\mathbb{Q}$), real numbers (from the set $\mathbb{R}$), or complex numbers (from the set $\mathbb{C}$). Or they can be integers modulo p, where p is prime, coming from the system $\mathbb{Z}_p$. All these number systems, called **fields**, have the property that x^{-1} exists for every non-zero x. (This somewhat loose description will do for our present purposes. An exact definition comprises 11 separate properties, or axioms.)

The **degree** of a polynomial is the largest power of x that occurs with a non-zero coefficient. That is, if $a(x) = a_n x^n + a_{n-1}x^{n-1} + ... + a_1 x + a_0$, the degree of $a(x)$ is n (provided $a_n \neq 0$). We can write this as **deg** $a(x) = n$.

Polynomials of degree 2 are called **quadratics**, of the form $ax^2 + bx + c$ (where $a \neq 0$).
Polynomials of degree 1 are the **linear polynomials** such as $2x + 3$ and $(½)x - ¼$.
Polynomials of degree 0 are the non-zero **constant polynomials** such as -3 and ¾.

There's one polynomial for which the degree remains undefined. It's the **zero polynomial**, 0.

The coefficient of x^n, for a polynomial of degree n, is called its **leading coefficient**. For example the leading coefficient of $3x^2 - x + 5$ is 3. But beware. The leading coefficient doesn't always come first – the leading coefficient of $1 - x^2$ is -1, not 1.

A **monic** polynomial is one where the leading coefficient is 1. Clearly every non-zero polynomial can be made monic by dividing it by its leading coefficient.

If F is a field (for example F might be $\mathbb{Q}$, the system of rational numbers) we denote the set of all polynomials with coefficients coming from F by the symbol **F[*x*]**.

Example 1: The polynomials in $\mathbb{Z}_2[x]$ are:
the constant polynomials 0, 1 ;
the linear polynomials x, $x + 1$;

the quadratics x^2, $x^2 + 1$, $x^2 + x$, $x^2 + x + 1$;
the cubics x^3, $x^3 + 1$, $x^3 + x$, $x^3 + x + 1$, $x^3 + x^2$, $x^3 + x^2 + 1$, $x^3 + x^2 + x$, $x^3 + x^2 + x + 1$;
and so on for higher degrees.

Polynomials are added, subtracted, and multiplied in the usual way. Just remember to use the appropriate field when doing your arithmetic.

Example 2: In $\mathbb{R}[x]$, $(x + 1)^2 = x^2 + 2x + 1$ but in $\mathbb{Z}_2[x]$ we should write it as $x^2 + 1$ since 2 modulo 2 is zero.

With these operations the system F[x] behaves very much like a field itself, but with one important difference. In a field nearly every number has an inverse under multiplication (in fact 0 is the only exception). Most polynomials, on the other hand, do *not* have inverses. For example, since $\frac{1}{x}$ and $\frac{1}{1-x}$ aren't polynomials, x and $1 - x$ don't have (polynomial) inverses. In fact the *only* polynomials with inverses are the non-zero constant polynomials such as -2 (whose inverse is the constant polynomial $-½$).

Now it *is* possible to write $\frac{1}{1-x}$ as $1 + x + x^2 + ...$ but although this looks like a polynomial it has infinitely many terms while a polynomial, by definition, has only finitely many. An expression like $1 + x + x^2 + ...$ is called a **power series**.

The system F[x] of polynomials over a field behaves much more like the system of integers (where only ±1 have integer inverses).

As mentioned earlier, one polynomial isn't usually exactly divisible by another. Like the system of integers we're left with a **remainder**. We get exact divisibility precisely when the remainder is zero. Furthermore this remainder, when it isn't zero, is in some sense *smaller* than whatever we are dividing by. For polynomials, *smaller* means 'of smaller degree'.

The process of obtaining the remainder on dividing one polynomial by another is very similar to the familiar long-division algorithm.

Example 3:

$$
\begin{array}{r}
 2x + 4 \\
\hline
x^2 - 2x + 7\ )\ 2x^3 + 5x - 3 \\
\underline{2x^3 - 4x^2 + 14x} \\
\ 4x^2 - 9x - 3 \\
\underline{4x^2 - 8x + 28} \\
\ -x - 31
\end{array}
$$

From this calculation we compute the remainder on dividing $2x^3 + 5x - 3$ by $x^2 - 2x + 7$ to be $-x - 31$. Note that the remainder has lower degree than that of $x^2 - 2x + 7$, the polynomial we're dividing by. Note also how we write the terms neatly underneath others of the same degree.

The result of the calculation can also be expressed as:

$$2x^3 + 5x - 3 = (x^2 - 2x + 7)(2x + 4) + (-x - 31).$$

Theorem 1: (Division Algorithm)
If a(x), b(x) are polynomials and b(x) is non-zero then a(x) = b(x)q(x) + r(x) for some polynomials q(x) and r(x) where either r(x) = 0 or deg r(x) < deg b(x).

The polynomial q(x) is called the **quotient** and r(x) is called the **remainder**.

If the remainder on dividing a(x) by b(x) is zero we say that *b*(x) **divides** a(x), or that a(x) is a **multiple** of b(x). If we can't be bothered saying it in words we just write 'b(x) | a(x)' and read it as 'b(x) divides a(x)'.

If f(x) $\in$ F[x], in other words f(x) is a polynomial in x with coefficients coming from the field F, and α is a number from F, or from some larger field that contains F, we define f(α) to be the number that results from replacing, or **substituting**, α for x in the polynomial f(x). For example if f(x) = $x^2 + x - 2$ then f(2) = 4 + 2 – 2 = 4, f(0) = –2 and f(1)=0.

Theorem 2: (Remainder Theorem) The remainder on dividing f(x) by $x - k$ is f(k).

Proof: By the Division Algorithm, f(x) = ($x - k$)q(x) + r(x) for some polynomials q(x), r(x) and the remainder r(x) is either zero or has degree less than 1. In other words r(x) must be a constant polynomial, so we can drop the '(x)' and just call it r. Now substituting $x = k$ into the equation f(x) = ($x - a$) q(x) + r, we get f(k) = r.

Corollary: The polynomial f(x) is divisible by $x - k$ if and only if f(k) = 0.

A **zero** of a polynomial f(x) is a number, α, such that that f (α) = 0. Solving a polynomial equation f(x) = 0 therefore means finding all its zeros.

But where do we look for potential zeros? From the coefficient field. But here we have to be a little careful. Does the polynomial f(x) = x^2 + 1 have any zeros? That depends on the field of coefficients. If f(x) $\in \mathbb{R}[x]$, the answer is "no". If f(x) $\in \mathbb{C}[x]$ the answer is "yes" – they are $\pm$ i.

If f(x) $\in \mathbb{Z}_2[x]$, the answer is again "yes", but this time we have a double zero of 1. If f(x) $\in \mathbb{Z}_3[x]$ the answer is "no" since putting x = 0, 1 and 2 in x^2 + 1 gives us the values 1, 2 and 2 (never zero). The corollary to the Remainder Theorem can be expressed by saying that the number α is a zero of f(α) if and only if $x - \alpha$ is one of its factors.

Now the polynomial $x - \alpha$ has degree 1 and it is a linear polynomial. So there's a connection between linear factors and zeros of a polynomial.

Theorem 3: A polynomial has a zero if and only if it has a linear factor.
Proof: Whenever we have a zero, α, we have a linear factor $x - \alpha$. Conversely having a linear factor $bx + c$ for a polynomial means that we have a zero $\alpha = -c/b$

If we know one zero of a polynomial we can divide by the corresponding linear factor. The other zeros will then be zeros of the quotient.

§13.3. Complete Polynomials

Suppose that $f(x) \in \mathbb{Z}_2[x]$, of degree n. Let t be an indeterminate satisfying f(t) = 0. By this we mean not just that f(t) = 0 but t is not a zero of any polynomial of lower degree. Then f(x) is **complete** if f(t) = 0 implies that every non-zero polynomial of degree less than n can be expressed as a power of t.

Example 4: $f(x) = x^3 + x + 1$ is complete. To see this we put $t^3 + t + 1 = 0$.
This means that $t^3 = -t - 1$, but since the coefficients come from $\mathbb{Z}_2$ where $-1 = +1$, we can write $t^3 = t + 1$. We can use this to express higher degree polynomials in terms of 1, t and t^2:

Multiplying by t we get $t^4 = t^2 + t$, $t^5 = t^3 + t^2 = (t + 1) + t^2 = t^2 + t + 1$ and so on.

1	**t**	**t^2**	**t^3**	**t^4**	**t^5**	**t^6**
1	t	t	t + 1	$t^2 + t$	$t^2 + t + 1$	$t^2 + 1$

The bottom row contains all 7 of the non-zero polynomials of degree at most 2. We could keep the table going but it would simply repeat, since $t^7 = t^3 + t = 1$, and so on.

It's important to remember that t is not an element of $\mathbb{Z}$. In fact, if f(t) = 0 for $t \in \mathbb{Z}_2$ then f(x) is definitely *not* complete. Instead, t is an **indeterminate** that we invent to extend $\mathbb{Z}_2$ to a larger field. It's a similar process to that of the creation of the field of complex numbers from the real numbers. Remember that we had a system $\mathbb{R}$ and a polynomial $f(x) = x^2 + 1$ that has no zeros in $\mathbb{R}$. So what did we do? We invented an 'indeterminate' that we called i that satisfied the equation f(i) = 0.

§13.4. Polynomial Codes: How They Work

The system of error-correcting coding that we'll describe is the Reed-Solomon Code To begin with we need a convention as to how a string of length $n + 1$ can be represented by a polynomial of degree n with the bits representing the coefficients of a polynomial over $\mathbb{Z}_2$.

We could take the first bit to represent the highest power of x, down to the last bit representing the constant

term. Or we could consider the first bit to be the constant term and proceed up through the increasing powers of x.

It doesn't matter which convention is adopted, so long as the transmitter and the receiver use the same one. We shall adopt the convention that the binary string

$$b_0, b_1, b_2, \ldots, b_n$$

is represented by the polynomial

$$b_0\, t^n + b_1\, t^{n-1} + \ldots + b_{n-1}\, t + b_n.$$

Example 5: Using this convention the string **1 0 0 1 1** would be represented by:
$\mathbf{1}.t^4 + \mathbf{0}.t^3 + \mathbf{0}.t^2 + \mathbf{1}.t + \mathbf{1}. = t^4 + t + 1.$

A prime polynomial $f(x)$ is a non-constant polynomial which cannot be factorised (except trivially). More formally $f(x)$ is **prime** if $\deg f(x) \geq 1$ and $f(x) = a(x)b(x)$ implies that either $a(x)$ or $b(x)$ is a constant. Clearly the degree 1 polynomials are prime, as are the quadratics and cubics that have no zeros. But when we get to higher degree polynomials we can get products of prime quadratics that have no zeros.

Example 6: In $\mathbb{Z}_2[x]$ the prime polynomials of degree up to 4 are:
x, $x + 1$, $x^2 + x + 1$, $x^3 + x + 1$, $x^4 + x^3 + 1$, $x^4 + x + 1$ and $x^4 + x^3 + x^2 + x + 1$.

The proof of the following theorem needs some little knowledge of rings and fields.

Theorem 4: If P(x) is a prime polynomial of degree n in $\mathbb{Z}_2[x]$, and $N = 2^{n-1} - 1$ is a prime number, then P(x) is complete.

Proof: Let t be an indeterminate such that P(t) = 0.

Let $F = \{a_{n-1}t^{n-1} + a_{n-2}t^{n-2} + \ldots + a_1t + a_0\}$, where each $a_i \in \mathbb{Z}_2$.

Since P(x) is prime, F is a field, with 2^n elements.

The non-zero elements will form a group under multiplication of order $N = 2^n - 1$.

The order of t in this group must divide N and, since N is prime, t must have order N.

So 1, t, t^2, …, t^{N-1} must be distinct, for if $t^h = t^{h+k}$ for $0 < k < N$ then $t^k = 1$ and so the order of t is less than N, a contradiction. Thus P(x) is complete.

Example 7: The need for N to be prime is shown by the following example.

Let $P(x) = x^6 + x^5 + x^4 + x^3 + x^2 + x + 1$.

It can be shown that P(x) is prime.

Note that $N = 2^6 - 1 = 63$, which is not prime.

Let t be an indeterminate such that:

$$t^6 + t^5 + t^4 + t^3 + t^2 + t + 1 = 0.$$

Now $x^6 - 1 = (x - 1)(x^6 + x^5 + x^4 + x^3 + x^2 + x + 1)$ and hence $t^6 = 1$. So there will be only 6 distinct powers of t, not 63 as we'd need if P(x) was complete.

REED-SOLOMON POLYNOMIAL CODE:

SETUP

* Calculate $N = 2^n - n - 1$.
* Choose a complete polynomial of degree n: $P(x) \in \mathbb{Z}_2[x]$.
* Let t be an indeterminate that satisfies the equation P(t) = 0.
* Construct a table of values of t^k for k up to t^{2^n-2}, expressing each one in terms of powers of t up to t^{n-1}, using the equation P(t) = 0.

ENCODE

* Take a message of N bits and convert it to a polynomial expression $M(x)$
* Calculate S(t) = M(t)P(t).

SEND

* Send the $2^n - 1$ bit string of coefficients of S(t).

RECEIVE

* Convert the received string to a polynomial expression R(t).

DECODE

* Use the table to write R(t) in terms of powers of t up to t^{n-1}.

* If R(t) = 0 there was no error in transmission and S(x) = R(x).
* If R(t) ≠ 0 use the table in reverse to express it as t^k where $0 \le k \le N + n - 1$.
* This indicates that the coefficient of E(x) = x^k was corrupt.
* Correct the error by adding E(x) to R(x). That is, S(x) = R(x) + x^k.
* Divide S(x) by P(x) to get M(x).
* Convert this polynomial to a string of length N.

Example 8:
Use the Reed-Solomon Code, with coding polynomial P(x) = $x^3 + x + 1$, to transmit the message 1100.

SETUP
Here n = deg P(x) = 0.
Suppose P(t) = 0. So $t^3 = t + 1$.
The table of powers is:

1 =	1
t =	t
t^2 =	t^2
t^3 =	t + 1
t^4 =	t^2 + t
t^5 =	t^2 + t + 1
t^6 =	t^2 + 1

Hence P(x) is a complete polynomial.

ENCODING

The message is 1100. Then $M(x) = x^3 + x^2$.
The sent polynomial is thus:

$$S(x) = (x^3 + x^2)(x^3 + x + 1) = x^6 + x^5 + x^4 + x^2.$$

TRANSMISSION

This is transmitted as a string 1 1 1 0 1 0 0.
Suppose during transmission an error occurs in the 3rd bit and that it is received as

1 1 0 0 1 0 0.

Notice that there is an error in the third bit.

DECODING

As a polynomial, the received string is:

$$R(x) = x^6 + x^5 + x^2.$$

Substituting, we get $R(t) = t^6 + t^5 + t^2$, which from the table can be written as

$$(t^2 + 1) + (t^2 + t + 1) + t^2 = t^2 + t.$$

Using the table in reverse we see that $R(t) = t^2 + t = t^4$.
So an error occurred in the bit corresponding to x^4.
That is, $E(x) = x^4$.
Correcting this we get $S(x) = R(x) + E(x)$

$$= x^6 + x^5 + x^4 + x^2.$$

Hence, using polynomial 'long division' the original message was

$$M(x) = \frac{x^6 + x^5 + x^4 + x^2}{x^3 + x + 1} = x^3 + x^2$$

Thus, as a string, the original message was 1 1 0 0.

§13.5. Polynomial Codes: Why They Work

If the original message is represented by the polynomial $M(x)$ the string corresponding to $S(x) = M(x)P(x)$ is what is sent. This is a polynomial of degree at most $2^n - 2$. Assuming at most one bit is altered during transmission, what is received, $R(x)$, differs from $S(x)$ in at most one term. Thus $R(x) = S(x) + E(x)$ where the error polynomial, $E(x)$, is given by $E(x) = 0$ (if there are no errors) or $E(x) = x^k$ for some $k \leq 2^n - 2$ if there is an error.

When we substitute $x = t$ we get $R(t) = S(t) + E(t) = M(t)P(t) + E(t) = E(t)$. This is because t is defined as a zero of the polynomial $P(x)$.

The error-correcting capabilities of this code rely on us being able to recover $E(x)$ from the value of $E(t)$. If $E(t) = 0$ we can infer that there was no error. In all other cases we need to be sure that the $2^n - 1$ powers of t: $1, t, t^2, \ldots, t^{2^n-2}$ are distinct. They all look different, but when reduced to powers of t up to t^{n-1} it could be that there's a repetition.

Now there are precisely 2^n distinct expressions in t up to t^{n-1} and so $2^n - 1$ non-zero ones. So it's just possible for the powers $1, t, t^2, \ldots, t^{2^n-2}$ to be distinct. Every non-zero expression in powers of t up to t^{n-1} would have to be hit exactly once as we go through the powers $1, t, t^2, \ldots, t^{2^n-2}$.

If you examine the table of powers of *t* in the above example you'll see that they *are* all distinct. If we'd used the polynomial $x^3 + x^2 + x + 1$ the table would be:

1 =	1
t =	t
$t2$ =	t^2
t^3 =	$t^2 + t + 1$
t^4 =	1
t^5 =	t
t^6 =	t^2

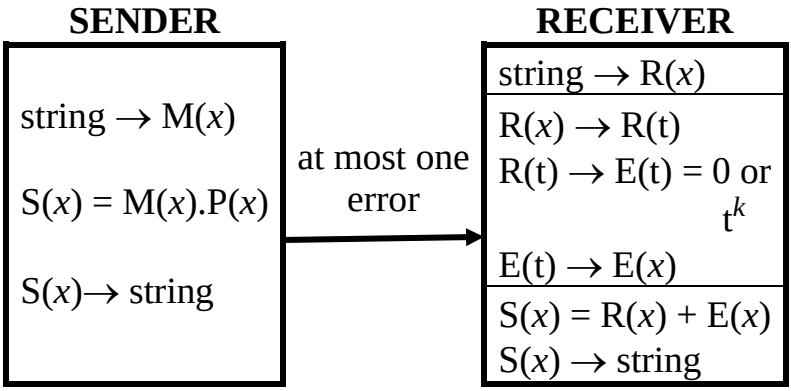
This would be of no use in error correction, because if an error occurred in the bit corresponding to the x^2 term the value of R(t) would be the same as if the error had occurred in the bit corresponding to the x^6 term. This is why P(*x*) needs to be a *complete* polynomial.

Students often get confused between the use of *x* and t in the above discussion. Remember that, although both *x* and t are indeterminates they have different properties, though there is a close connection between them.

The indeterminate *x* satisfies no polynomial equation, other than the trivial 0 = 0. On the other hand, t satisfies the equation P(t) = 0. It's like the difference between *x* and i with complex numbers. We tend to think of i as a number, rather than as an indeterminate, which it is. But it is also an indeterminate satisfying a certain

polynomial equation $i^2 + 1 = 0$. So we can write $i^3 + i = 0$, for example but we can't write $x^3 + x = 0$. Well, we can, but $i^3 + i = 0$ is a *fact*, while $x^3 + x = 0$ is an *equation* that we might want to solve. As polynomials they are quite different.

So we work with x up to the time we transmit a message. It is only when the message is received that we substitute t. This gives an expression in t , which we write as 0 or a power of t. We then take the corresponding polynomial in x to locate the error (or decide that there was none).



Only the receiver uses t, and then only in the central error detecting stage.

§13.6. Analysis of Probabilities

If two or more errors occur, this procedure will fail. We need to ensure that the message string is broken into packets, small enough so that the probability of this happening is acceptably low. On the other hand the smaller the packet size the less efficient is the code (the more bits that need to be sent for the same message). We therefore do a trade-off.

The **efficiency** of a coding process is defined to be the ratio of bits in the message to the total bits sent.

Example 9: For the above code, using a cubic $\mathbf{P}(x)$, we need to transmit 7 bits for a 4 bit message, so the efficiency = 4/7 or 57%.

The **reliability** is defined as the probability of at most one error in transmission.

Example 10: If the probability of an error in a single bit is 1/100, the reliability is

$$0.99^7 + 0.07(0.99)^6 = \approx 0.998 \text{ or } 99.8\%.$$

Theorem 5: For the polynomial code using a complete polynomial of degree n:

$$\text{Efficiency} = \frac{\text{\# bits in message}}{\text{\#bits sent}} = \frac{2^n - n - 1}{2^n - 1};$$

$$\text{Reliability} \approx 1 - (2^n - 1)(2^{n-1} - 1)p^2.$$

n	N	Efficiency	Reliability if p = 0.01	Reliability if p = 0.001
2	3	33%	100%	100%
3	7	57%	100%	100%
4	15	60%	99%	100%
5	31	84%	95%	100%
6	63	90%	80%	100%
7	127	94%	20%	99%
8	255	97%	0%	97%

EXERCISES FOR CHAPTER 13

EXERCISES 13A (Polynomials Mod P(x))

Ex 13A1: (i) Construct the addition and multiplication tables for $\mathbb{Z}_2[t \mid t^2 + 1 = 0]$, the system of the remainders of polynomials in t over $\mathbb{Z}_2$ on division by $t^2 + 1$.
(ii) Which of the four elements of this system have an inverse under addition?
(iii) Which of the four elements of this system have an inverse under multiplication?

Ex 13A2: (i) Construct the addition and multiplication tables for $\mathbb{Z}_2[t \mid t^2 + t + 1 = 0]$,
(ii) Which of the four elements of this system have an inverse under addition?
(iii) Which of the four elements of this system have an inverse under multiplication?

EXERCISES 13B (Polynomial Codes)

Ex 13B1: Use the Reed-Solomon Code, using the polynomial $P(x) = x^3 + x + 1$ to correct a received string 1000101. (We assume that at most one of these 7 bits has been corrupted.)

(i) Show that no error has occurred.
(ii) What was the original message?

Ex 13B2: Use the Reed-Solomon Code, using the polynomial $P(x) = x^3 + x + 1$ to correct a received string

1100100. Assume that at most one of these 7 bits has been corrupted.
(i) Show that indeed an error has occurred in one of the bits.
(ii) Which bit was wrong?
(iii) Correct the error and recover the original message.

Ex 13B3: Find an irreducible polynomial of degree 4 over $\mathbb{Z}_2$ and use it to encode the binary string 11011.

Ex 13B4:
(a) Show that $P(x) = x^3 + x^2 + 1 \in \mathbb{Z}_2[x]$ is a complete polynomial.
(b) (i) Use $P(x)$ to encode the message string 1001.
(ii) Suppose 1011010 is received. Correct the error and work out the original message.

Ex 13B5: Use the Reed-Solomon Code, using the complete polynomial $x^4 + x^3 + 1$ to correct a received string 011101110111001.

SOLUTIONS FOR CHAPTER 13

Exercise 13A1:

+	0	1	t	1+t
0	0	1	t	1+t
1	1	0	1+t	t
t	T	1+t	0	1
1+t	1+t	t	1	0

×	0	1	t	1+t
0	0	0	0	0
1	0	1	t	1+t
t	0	t	1	1+t
1+t	0	1+t	1+t	0

(ii) all four.; (iii) 1 and *t* only.

Exercise 13A2:

+	0	1	t	1+t
0	0	1	t	1+t
1	1	0	1+t	t
t	t	1+t	0	1
1+t	1+t	t	1	0

×	0	1	t	1+t
0	0	0	0	0
1	0	1	t	1+t
t	0	t	1+t	1
1+t	0	1+t	1	t

(ii) all four; (iii) all except 0.

Ex 13B1:
P(x) = $x^3 + x + 1$ is a complete polynomial and in $\mathbb{Z}_2[t \mid t^3 + t + 1 = 0]$ we get:

$t^3 = t + 1$;
$t^4 = t^2 + t$;
$t^5 = t^3 + t^2 = t^2 + t + 1$;
$t^6 = t^3 + t^2 + t = t^2 + 1$;
$t^7 = t^3 + t = 1$.

Converting the received string 1000101 to a polynomial we get: R[x] = $x^6 + x^2 + 1$.
Substituting $x = t$, where $t^3 + t + 1 = 0$ we get:

$$t^6 + t^2 + 1 = (t^2 + 1) + t^2 + 1 = 0.$$

So no error has occurred: E(x) = 0.

(ii) The received polynomial, $x^6 + x^3 + x^2 + 1$, was correct. Dividing this by $x^3 + x + 1$ we get a remainder of zero and a quotient of $x^3 + x + 1$. The original message was thus 1011.

Exercise 13B2:
Let P(x) = $x^3 + x + 1$. Now if P(t) = 0 then:

$t^3 = t + 1$;
$t^4 = t^2 + t$;
$t^5 = t^3 + t^2 = t^2 + t + 1$;
$t^6 = t^3 + t^2 + t = t^2 + 1$;
$t^7 = t^3 + t = 1$.

Converting the received string 1100100 to a polynomial we get $x^6 + x^5 + x^2$. Substituting $x = t$ we get:

$$t^6 + t^5 + t^2 = (t^2 + 1) + (t^2 + t + 1) + t^2 = t^2 + t = t^4.$$

If no errors occurred this would have been 0. Hence an error occurred.

(ii) Since the coefficient of t^4 was incorrect, the third bit was wrongly received.

(iii) The received polynomial should have been:

$$x^6 + x^5 + x^4 + x^2.$$

Dividing this by $x^3 + x + 1$ we get a remainder of zero and a quotient of $x^3 + x^2$.
The corrected message is thus 1100.

Ex 13B3: $t^4 + t^3 + 1$ (or $t^4 + t + 1$ but we'll only give the answers for $t^4 + t^3 + 1$).
Now 11011 becomes $x^4 + x^3 + x + 1$ which is sent as:

$$(x^4 + x^3 + 1)(x^4 + x^3 + x + 1) = x^8 + x^6 + x^5 + x^4 + x^3 + x + 1.$$

This is transmitted as 101111011.

Ex 13B4:
(a) Suppose that $x^3 + x^2 + 1 = 0$.

$\therefore x^3 = x^2 + 1$
$\therefore x^4 = x^3 + x = (x^2 + 1) + x = x^2 + x + 1$
$\therefore x^5 = x^3 + x^2 + x = (x^2 + 1) + x^2 + x = x + 1$
$\therefore x^6 = x^2 + x$
$\therefore x^7 = x^3 + x^2 = 1.$

Since all 7 of the non-zero polynomials of degree less than 3 occur as a power of t this polynomial is complete.

(b) (i) The message 1001 corresponds to the message polynomial $M(x) = x^3 + 1$.
The sent message is $S(x) = M(x)P(x)$
$= (x^3 + 1)(x^3 + x^2 + 1)$
$= x^6 + x^5 + x^3 + x^3 + x^2 + 1$
$= x^6 + x^5 + x^2 + 1$.
This is transmitted as the string 1100101.

(ii) From (a) the table of powers is:

1	1
t	t
t^2	t^2
t^3	$t^2 + 1$
t^4	$t^2 + t + 1$
t^5	$t + 1$
t^6	$t^2 + 1$

The received polynomial is $R(x) = x^6 + x^4 + x^3 + x$.
$R(t) = t^6 + t^4 + t^3 + t$
$= (t^2 + 1) + (t^2 + t + 1) + (t^2 + 1) + t$
$= t^2 + 1 = t^6$.
So an error has occurred and $E(x) = x^6$.
$\therefore M(x)P(x) = S(x) = R(x) + E(x) = x^4 + x^3 + x$.
$\therefore M(x) = (x^4 + x^3 + x)/(x^3 + x^2 + 1) = x$.
So the original message was 0010.

Ex 13B5: Suppose that $t^4 + t^3 + 1 = 0$. The table of powers is:

t^4	$t^3 + 1$
t^5	$t^3 + t + 1$
t^6	$t^3 + t^2 + t + 1$
t^7	$t^2 + t + 1$
t^8	$t^3 + t^2 + t$
t^9	$t^2 + 1$
t^{10}	$t^3 + t$
t^{11}	$t^3 + t^2 + 1$
t^{12}	$t + 1$
t^{13}	$t^2 + t$
t^{14}	$t^3 + t^2$
t^{15}	1 This is a check.

The received polynomial is:

$R(x) = x^{13} + x^{12} + x^{11} + x^9 + x^8 + x^7 + x^5 + x^4 + x^3 + 1$

Thus $R(t) = t^{13} + t^{12} + t^{11} + t^9 + t^8 + t^7 + t^5 + t^4 + t^3 + 1$

$= (t^2 + t) + (t + 1) + (t^3 + t^2 + 1) + (t^2 + 1) + (t^3 + t^2 + t)$

$+ (t^2 + t + 1) + (t^3 + t + 1) + (t^3 + 1) + t^3 + 1$

$= t^3 + t^2 + t + 1 = t^6$, so $E(x) = x^6$.

Therefore the transmitted polynomial must have been:

$R(x) + E(x)$

$= x^{13} + x^{12} + x^{11} + x^9 + x^8 + x^7 + x^6 + x^5 + x^4 + x^3 + 1$

Dividing by $x^4 + x^3 + 1$ we get a remainder of zero (this is a check) and a quotient of $x^9 + x^7 + x^6 + x^5 + x$ and so the original message must have been 01011100010.