Contents

1 CODES AND CIPHERS. AN INTRODUCTION 2
  1.1 Codes .................................................. 2
  1.2 Ciphers .................................................. 2
  1.3 The basic idea of classical cryptography .................. 3
  1.4 A simple cryptosystem .................................. 3
  1.5 Modular Arithmetic .................................... 4
  1.6 Arithmetic mod $m$ .................................... 4
  1.7 Inverses mod $m$ and GCD’s .......................... 5
  1.8 Shift ciphers .......................................... 6
  1.9 Using a Shift Cipher ................................... 7
  1.10 Substitution ciphers and transposition ciphers ......... 8
  1.11 Polyalphabetic, block and stream ciphers .............. 9
  1.12 Cryptanalysis .......................................... 13
Chapter 1

CODES AND CIPHERS. AN INTRODUCTION

This unit is concerned with two related problems in communication theory, codes and ciphers. Codes are used to deal with errors that may occur in the transmission of information; to detect errors and possibly correct them. Ciphers are used to guard the privacy and security of transmitted information.

1.1 Codes

The main problem of information and coding theory can be described in a simple way as follows. It concerns error detection and correction.

The source data, say a string of zeros and ones, is transmitted along a channel such as a telephone line. The received data may contain errors with some of the zeros having changed to ones, or vice versa.

The main questions are: Have some ‘bits’ (zeros or ones) changed? and If so can we recover the original source data?

This problem occurs in many different situations, for example, in communications from space vehicles, or when we try to recover data from electronic storage.

The aim is for reliability of data transmission across a ‘noisy channel’ that may introduce errors. And we want reliability at a reasonable cost.

1.2 Ciphers

The main issue for cryptography, that is the science of ciphers, is secrecy. A message is sent through an insecure channel, where it may be intercepted by an enemy.

Main questions are: How can the information in the message be concealed from the enemy, even if the message is intercepted? and secondly If the enemy sends a message pretending to come from me, how can
the recipient detect this?

The answers lie in encrypting the message in such a way that the intended recipient can decrypt the message but the enemy cannot. Frameworks for doing this are called cryptosystems.

1.3 The basic idea of classical crypography

Much of the material below is based on Chapter 1 of Cryptography: theory and practice by Douglas R. Stinson.

The aim of cryptography is to enable two people, often called Alice and Bob, to communicate over an insecure channel so that an opponent Oscar cannot understand their messages.

These days the channel might be a telephone line, or a computer network. However cryptography has a long history. In the past the channel may have been a published book or a newspaper article (and the ‘opponent’ would have been an ordinary reader), or the channel may have been a soldier carrying a concealed message with the opponent being the enemy army. Examples can be found in any introduction to cryptography, for example in the Code Book (see recommended reading), or in http://www-math.cudenver.edu/~wcherowi/courses/m5410/m5410f.html Bill Cherowitzo’s notes at the University of Colorado.

We call Alice’s message plaintext. Let’s say her message is \( x \). Alice encrypts the message by applying one of a set of rules. The rule \( e_K \) used depends on her choice of a secret key \( K \). The encrypted message \( y = e_K(x) \), which we call ciphertext, is sent over the insecure channel to Bob. Separately, by a secure channel Alice sends Bob her secret key \( K \), which he will need to decipher Alice’s message. When Bob receives the encrypted message \( y \) he decrypts the message by applying the decryption rule \( d_K \) corresponding to the key \( K \) and retrieves Alice’s plain text message \( x = d_K(y) \). Meanwhile the opponent Oscar has intercepted the ciphertext \( y \), but (ideally) without knowledge of the key \( K \) he is unable to understand it.

1.4 A simple cryptosystem

\[
\begin{align*}
\mathbb{P} & \quad \text{set of all possible plaintexts} \\
\mathbb{C} & \quad \text{set of all possible ciphertexts} \\
\mathbb{K} & \quad \text{set of all possible keys}
\end{align*}
\]

For each key \( K \in \mathbb{K} \), there corresponds an encryption rule \( e_K : \mathbb{P} \to \mathbb{C} \) and a decryption rule \( d_K : \mathbb{C} \to \mathbb{P} \). The rules \( e_K \) and \( d_K \) must have the properties listed in Table 1.1.

Oscar’s job is to find \( x \), knowing \( y \) and without knowing the key \( K \), but possibly knowing the type of encryption system used. This type of decryption is called cryptanalysis. We’ll return to it later.
Property | Reason
---|---
e\_K must be one-to-one | if \( y = e\_K(x_1) = e\_K(x_2) \) with \( x_1 \neq x_2 \) then Bob would be unable to decrypt \( y \)
d\_K(e\_K(x)) = x for each \( x \in \mathbf{P} \) | to enable Bob to decipher all messages from Alice
if \( \mathbf{P} = \mathbf{C} \) then each \( e\_K \) is a permutation and \( d\_K \) is the inverse permutation | if the set of plaintexts and ciphertexts is identical then each encryption rule \( e\_K \) just rearranges (permutes) the elements of this set.

Table 1.1: Properties of cryptosystems

### 1.5 Modular Arithmetic

Some of this material will be already familiar to you from 3P5 or the background notes we handed out at the beginning of the unit. We have also collected together all the sections on the theory of modular arithmetic in a separate chapter on this website, see [http://www.maths.uwa.edu.au/~praeger/WWW/teaching/3CC/lecturenotes.html](http://www.maths.uwa.edu.au/~praeger/WWW/teaching/3CC/lecturenotes.html) this link.

Suppose that \( a \) and \( b \) are integers and \( m \) is a positive integer. Then we write

\[
a \equiv b \pmod{m},
\]

and we read this phrase as \( a \) is congruent to \( b \) modulo \( m \), if \( m \) divides \( a - b \).

Suppose we divide \( a \) and \( b \) by \( m \), obtaining quotients and remainders, where the remainders are between 0 and \( m - 1 \), that is, \( a = q_1 m + r_1 \) and \( b = q_2 m + r_2 \), where \( 0 \leq r_1 \leq m - 1 \) and \( 0 \leq r_2 \leq m - 1 \). Then \( a \equiv b \pmod{m} \iff m \text{ divides } a - b = (q_1 - q_2)m + (r_1 - r_2) \iff m \text{ divides } r_1 - r_2 \iff r_1 = r_2 \).

We use ‘\( a \mod m \)’ (without parentheses) to denote the remainder when \( a \) is divided by \( m \), that is ‘\( a \mod m \)’ = \( r_1 \). Then \( a \equiv b \pmod{m} \) if and only if ‘\( a \mod m \)’ = ‘\( b \mod m \)’. Also, if we replace ‘\( a \)’ by ‘\( a \mod m \)’ then we say that \( a \) is reduced modulo \( m \).

**Warning:** Some computer languages define \( a \mod m \) to be the remainder (after dividing \( a \) by \( m \)) in the range \(-m+1, \ldots, m-1\) having the same sign as \( a \). So, for example, \(-17 \mod 8\) would be \(-1\), whereas we have defined it to be \(7\). For our uses it is more convenient to have \( a \mod m \) non-negative.

### 1.6 Arithmetic mod \( m \)

We define arithmetic modulo \( m \) as follows. Set \( \mathbf{Z}_m = \{0, 1, \ldots, m-1\} \). We define addition and multiplication on \( \mathbf{Z}_m \) to be the same as ordinary addition and multiplication of integers, except that the results are reduced modulo \( m \).

**Example 1.6.1** To compute \( 11 \times 13 \) in \( \mathbf{Z}_{15} \), first work out \( 11 \times 13 = 143 \); then perform ordinary long division: \( 143 = 9 \times 15 + 8 \), so \( 143 \text{ mod } 15 = 8 \) and hence \( 11 \times 13 = 8 \) in \( \mathbf{Z}_{15} \).
addition is \emph{closed} for all \(a, b \in \mathbb{Z}_m\), \(a + b \in \mathbb{Z}_m\)
addition is \emph{commutative} for all \(a, b \in \mathbb{Z}_m\), \(a + b = b + a\)
addition is \emph{associative} for all \(a, b, c \in \mathbb{Z}_m\), \((a + b) + c = a + (b + c)\)

0 is an \emph{additive identity} for all \(a \in \mathbb{Z}_m\), \(a + 0 = 0 + a = a\)
the \emph{additive inverse} of \(a \in \mathbb{Z}_m\) is \(m - a\)
multiplication is \emph{closed} for all \(a, b \in \mathbb{Z}_m\), \(ab \in \mathbb{Z}_m\)
multiplication is \emph{commutative} for all \(a, b \in \mathbb{Z}_m\), \(ab = ba\)
multiplication is \emph{associative} for all \(a, b, c \in \mathbb{Z}_m\), \((ab)c = a(bc)\)
1 is an \emph{multiplicative identity} for all \(a \in \mathbb{Z}_m\), \(a \times 1 = 1 \times a = a\)
multiplicat’n \emph{distributes over addit’n} for all \(a, b, c \in \mathbb{Z}_m\), \((a + b)c = (ac) + (bc)\)

Table 1.2: Properties of + and \(	imes\) for \(\mathbb{Z}_m\).

Addition and multiplication modulo \(m\) satisfy the familiar list of properties in Table 1.2. We do not include proofs. These properties (and more) are proved in the third year algebra units. Ask if you need to know more about them.

### 1.7 Inverses mod \(m\) and GCD’s

Next we consider the problem of \emph{multiplicative inverses modulo} \(m\): for which integers \(a \in \mathbb{Z}_m\) can we find an ‘inverse’ \(x \in \mathbb{Z}_m\), that is \(ax \equiv xa \equiv 1 \pmod{m}\) (equivalently \(ax = xa = 1 \in \mathbb{Z}_m\))?

Obviously \(a\) has to be non-zero but this isn’t sufficient. For example in \(\mathbb{Z}_{26}\), \(a = 2\) has no multiplicative inverse, for if say \(x\) was a multiplicative inverse for 2 then we would have

\[
13 = 13 \cdot 1 = 13 \cdot (2x) = (13 \cdot 2)x = 26x = 0 \pmod{26}
\]

which is not true. We get a similar problem for all the even integers 4, 6, \ldots, 24 and also for 13. (Can you see why?) In fact if \(a\) and \(m\) have a common divisor \(d > 1\), then \(x\) has no inverse in \(\mathbb{Z}_m\). (See Question 2 in the Exercise Set for Chapter 1.) It turns out that all other elements of \(\mathbb{Z}_m\) have multiplicative inverses. We prove this in Lemma 1.7.1 below. The proof is not examinable but you should read it and understand why it is true.

Given two positive integers \(m, n\), there is a unique positive integer \(d\) called the \emph{greatest common divisor} of \(m\) and \(n\) and written \(d = \gcd(m, n)\), such that

(i) \(d\) divides \(m\) and \(d\) divides \(n\), and

(ii) if \(c\) divides \(m\) and \(c\) divides \(n\), then \(c\) divides \(d\).

Condition (i) says that \(d\) is a ‘common divisor’, while condition (ii) says that it is the ‘greatest’ one.
Table 1.3: Inverses in $\mathbb{Z}_{26}$

<table>
<thead>
<tr>
<th>$a$</th>
<th>1 3 5 7 9 11 15 17 19 21 23 25</th>
</tr>
</thead>
<tbody>
<tr>
<td>$a^{-1}$</td>
<td>1 9 21 15 3 19 7 23 11 5 17 25</td>
</tr>
</tbody>
</table>

Lemma 1.7.1 $a \in \mathbb{Z}_m$ has a multiplicative inverse $\iff$ the function $f : \mathbb{Z}_m \to \mathbb{Z}_m$ defined by $f(x) = ax$ is one-to-one $\iff \gcd(a, m) = 1$.

Proof: Suppose $b$ is a multiplicative inverse of $a$. Then for $x_1, x_2 \in \mathbb{Z}_m$, $f(x_1) = f(x_2) \implies ax_1 = ax_2$ in $\mathbb{Z}_m \implies x_1 = 1 \cdot x_1 = (ba)x_1 = b(ax_1) = b(ax_2) = (ba)x_2 = 1 \cdot x_2 = x_2 \bmod m$. Hence $f(x)$ is one-to-one.

Conversely if $f(x)$ is one-to-one then $f(\mathbb{Z}_m) = \{f(x) \mid x \in \mathbb{Z}_m\}$ consists of $m$ distinct elements of $\mathbb{Z}_m$, and since $\mathbb{Z}_m$ only contains $m$ elements, $f(\mathbb{Z}_m) = \mathbb{Z}_m$ and in particular contains 1. So $1 = f(b) = ab$ for some $b \in \mathbb{Z}_m$, and $b$ is a multiplicative inverse of $a$.

Suppose next that $f(x)$ is one-to-one. If $d = \gcd(a, m) > 1$, then $f(0) = 0$ and also $f(\frac{m}{d}) = a \cdot \frac{m}{d} = m \cdot \frac{a}{d} = 0 \bmod m$ but $0 \neq \frac{m}{d} \bmod m$, which contradicts the assumption that $f$ is one-to-one. Hence $\gcd(a, m) = 1$.

Conversely suppose that $\gcd(a, m) = 1$. Then $f(x_1) = f(x_2) \implies ax_1 \equiv ax_2 \pmod m \implies a(x_1 - x_2) \equiv 0 \pmod m \implies m$ divides $a(x_1 - x_2)$. By the property of division, “if $\gcd(a, m) = 1$ and $m$ divides $ab$ then $m$ divides $b$” we deduce that $m$ divides $x_1 - x_2$, so $x_1 = x_2 \bmod m$. That is, $f$ is one-to-one. qed

In Table 1.3 we give for easy reference a list of the multiplicative inverses for the 12 elements $a \in \mathbb{Z}_{26}$ for which $\gcd(a, 26) = 1$.

1.8 Shift ciphers

These form a family of ciphers based on modular arithmetic. One such cipher is reputed to have been used by Julius Caesar and is called the Caesar Cipher. Shift Ciphers, and the more general Affine Ciphers we’ll meet later, are based on modular arithmetic.

Definition 1.8.1 (Shift Cipher) Let $P = C = K = \mathbb{Z}_{26}$. For $0 \leq K \leq 25$, define $e_K : \mathbb{Z}_{26} \to \mathbb{Z}_{26}$ and $d_K : \mathbb{Z}_{26} \to \mathbb{Z}_{26}$ as follows (for $x, y \in \mathbb{Z}_{26}$).

$$e_K(x) = x + K \pmod{26}, \quad d_K(y) = y - K \pmod{26}$$

It is easy to see that each $e_K(x)$ is a one-to-one function, and that $d_K(e_K(x)) = x$ for all $x \in \mathbb{Z}_{26}$. Hence the Shift Cipher has the required properties for a cryptosystem.

The shift cipher with $K = 3$ is the Caesar cipher.

We use $\mathbb{Z}_{26}$ because there are 26 letters in the English alphabet. Shift ciphers can be defined for any modulus $m$. 
1.9 Using a Shift Cipher

For ordinary English text set up a correspondence between the alphabetic characters
and the integers 0, . . . , 25 as in Table 1.4. A shift cipher with key $K$ operates by
‘shifting each letter $K$ places to the left’. We illustrate with a simple example.

Example 1.9.1 Encrypt the following plaintext with a Shift Cipher with key $K = 11$.

agoodproofisonethatmakesuswiser
(This plaintext is a quote from A course in mathematical logic by Yu I. Manin, page 51.)

Alice either constructs and uses a ‘look-up table’ which goes directly from the
plaintext to the ciphertext letters, as in Table 1.5, or she proceeds as follows. She
converts the plaintext to a sequence of integers using Table 1.4, obtaining:

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10|11|12|13|14|15|16|17|18|19|20|21|22|23|24|25|

Table 1.4: Correspondence of alphabet with $\mathbb{Z}_{26}$

Table 1.5: Plaintext to ciphertext with key $K = 11$

| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| L | M | N | O | P | Q | R | S | T | U | V | W | X | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
| Y | Z | A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |

| 0 | 6 | 14 | 14 | 3 | 15 | 17 | 14 | 14 | 5 | 8 | 18 | 14 | 13 | 4 | 19 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 7 | 0 | 19 | 12 | 0 | 10 | 4 | 18 | 20 | 18 | 22 | 8 | 18 | 4 | 17 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 11 | 17 | 25 | 25 | 14 | 0 | 2 | 25 | 25 | 16 | 19 | 3 | 25 | 24 | 15 | 4 |
| 18 | 11 | 4 | 23 | 11 | 21 | 15 | 3 | 5 | 3 | 7 | 19 | 3 | 15 | 2 |

Next she adds 11 to each value, reducing the sum modulo 26.

| 11 | 17 | 25 | 25 | 14 | 0 | 2 | 25 | 25 | 16 | 19 | 3 | 25 | 24 | 15 | 4 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 18 | 11 | 4 | 23 | 11 | 21 | 15 | 3 | 5 | 3 | 7 | 19 | 3 | 15 | 2 |

Finally she converts the sequence of integers to alphabetic characters, obtaining
the ciphertext.

LRZZOACZZQTDZYPESLEXLVPDFDHTDPC

To decrypt this message Bob will first convert the ciphertext to a sequence of in-
tegers, then subtract 11 from each value (reducing modulo 26), and finally he will
convert the sequence of integers to alphabetic characters.

Conventions: We use lower case letters for plaintext and upper case letters for
ciphertext. Often the spaces between words are removed.

Desirable properties for a cryptosystem:
It should be easy to encipher and decipher, that is, $e_K$ and $d_K$ should be efficiently
computable.
It should be secure in the sense that, given the ciphertext, the opponent Oscar
should be unable to find the key or the plaintext.

The Shift Cipher (modulo 26) is **not secure**. Oscar could conduct an exhaustive search by trying out all the possible 26 decryption rules \(d_K\) in turn until he discovers a plaintext that makes sense.

**Example 1.9.2** Given the ciphertext

\[
IUJHKXKGQKXY
\]

simply try out \(d_K\) for all the keys \(K = 0, 1, \ldots\)

\[
iujkhhxkgqkxy
\]
\[
htijgwjfpjwx
\]
\[
gshifvioivw
\]
\[
frgheuhdnhuv
\]
\[
eqfgdgcmtu
\]
\[
dpefsfblfst
\]
\[
codebreakers
\]

until we find the plaintext. Here we find that the key was \(K = 6\) and the plaintext was ‘codebreakers’.

### 1.10 Substitution ciphers and transposition ciphers

For a general **substitution cipher** we require \(P = C\), that is the same ciphertext and plaintext alphabet. In our examples we will take \(P\) to be the 26 letter alphabet and identify it with \(\mathbb{Z}_{26}\). The set of keys \(K\) will be the set of all permutations of \(P\). For a permutation \(\pi \in K\), the rules \(e_{\pi}\) and \(d_{\pi}\) are

\[
e_{\pi}(x) = \pi(x), \quad \text{and} \quad d_{\pi}(y) = \pi^{-1}(y)
\]

where \(\pi^{-1}\) is the inverse permutation of \(\pi\), that is \(\pi^{-1}(y) = x\) if and only if \(\pi(x) = y\).

The number of these permutations is \(26! \approx 10 \times 4.0^{26}\), so an exhaustive key search is infeasible, even using a computer. (However other methods of cryptanalysis can be used to break a substitution cipher, see later.)

A shift cipher is a special type of substitution cipher, and there are only 26 of them. Another special type of substitution cipher is an **Affine Cipher**. This is one for which the encryption rule is

\[
e(x) = ax + b \mod 26
\]

for some \(a, b \in \mathbb{Z}_{26}\). To ensure that decryption is possible this function should be one-to-one (see Table 1.1). This happens if and only if \(\gcd(a, 26) = 1\) (see Lemma 1.7.1, and question 6 in the Exercise Set for Chapter 1). To decrypt the ciphertext \(y = e(x) = ax + b\) we need \(ax = y - b\) and hence \(x = a^{-1}(y - b)\). Thus we must apply the decryption rule

\[
d(y) = a^{-1}(y - b).
\]
**Example 1.10.1** Encrypt the plaintext fun using an affine cipher with key $K = (7, 5)$ (that is, with encryption rule $e_K(x) = 7x + 5$). Convert the letters $f, u, n$ to residues modulo 26 using Table 1.4, and then apply $e_K$ and convert back to letters, again using Table 1.4.

<table>
<thead>
<tr>
<th>Plaintext</th>
<th>$x$</th>
<th>$e_K(x)$</th>
<th>Ciphertext</th>
</tr>
</thead>
<tbody>
<tr>
<td>$f$</td>
<td>5</td>
<td>$7 \cdot 5 + 5 = 40 = 14 \mod 26$</td>
<td>O</td>
</tr>
<tr>
<td>$u$</td>
<td>20</td>
<td>$7 \cdot 20 + 5 = 145 = 15 \mod 26$</td>
<td>P</td>
</tr>
<tr>
<td>$n$</td>
<td>13</td>
<td>$7 \cdot 13 + 5 = 96 = 18 \mod 26$</td>
<td>S</td>
</tr>
</tbody>
</table>

Thus the encrypted message is OPS. Bob, knowing the key $K = (7, 5)$, decrypts this with the decryption rule $d_K(y) = 15(y - 5)$ (since $7^{-1} = 15$).

<table>
<thead>
<tr>
<th>Ciphertext</th>
<th>$y$</th>
<th>$d_K(y)$</th>
<th>Plaintext</th>
</tr>
</thead>
<tbody>
<tr>
<td>O</td>
<td>14</td>
<td>$15(14 - 5) = 135 = 5 \mod 26$</td>
<td>f</td>
</tr>
<tr>
<td>P</td>
<td>15</td>
<td>$15(15 - 5) = 150 = 20 \mod 26$</td>
<td>u</td>
</tr>
<tr>
<td>S</td>
<td>18</td>
<td>$15(18 - 5) = 195 = 13 \mod 26$</td>
<td>n</td>
</tr>
</tbody>
</table>

Note that affine ciphers are not computationally secure as there are only $26 \cdot 12 = 72$ possible keys, and an exhaustive search by computer is feasible.

Another type of cipher is the **transposition cipher**, sometimes also called the **permutation cipher**. For this cipher, the plaintext message is divided into blocks of a certain length $m$, and a permutation $\pi$ is applied to rearrange the letters in each block. For example, if $m = 5$ and

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 \\ 3 & 1 & 4 & 5 & 2 \end{pmatrix}$$

maps 1st letter -> 3rd place, 2nd letter -> 1st place, etc then we encrypt the plaintext ‘This is not secure’ as:

```
Plaintext: this is not secure
Ciphertext: H I T I S N S S O T C E E U R
```

The transposition cipher can be decrypted by applying the inverse permutation $\pi^{-1}$. The transposition cipher is not secure against cryptanalytic attack. However a combination of a substitution cipher followed by a transposition cipher and so on may result in a cipher that is more difficult to break. This approach we will see much mater in the unit when we study **DES**, the Data Encryption Standard.

### 1.11 Polyalphabetic, block and stream ciphers

The ciphers described so far are all **monoalphabetic ciphers**, that is, for each occurrence of a given letter in the plaintext, the same cipher letter is used. A famous cipher that is not monoalphabetic is the **Vigenere cipher** named after Blaise de Vigenere who lived in the 16th century. The key $K$ is a ‘keyword’ of length $m$, for example $K$ might be ‘wombat’ of length $m = 6$. We encrypt the plaintext 6 letters at a time by ‘adding the keyword to the plaintext’. Here is an example. From Table 1.4, the letters of the keyword ‘wombat’ correspond to the numbers 22, 14, 12, 1, 0, 19. We write these out repeatedly under the numbers corresponding
to the plaintext, add the result and convert to ciphertext using the correspondence in Table 1.4. We encrypt ‘Bring me chocolate’ in this way as follows.

<table>
<thead>
<tr>
<th>Plaintext</th>
<th>b r i n g m e c h o c o l a t e</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>1 17 8 13 6 12 4 2 7 14 2 14 11 0 19 4</td>
</tr>
<tr>
<td>keyword</td>
<td>22 14 12 1 0 19 22 14 12 1 0 19 22 14 12 1</td>
</tr>
<tr>
<td>$e_K(x)$</td>
<td>23 5 20 14 6 5 0 16 19 15 2 7 7 14 5 5</td>
</tr>
<tr>
<td>Ciphertext</td>
<td>X F U O G F A Q T P C H H O F F</td>
</tr>
</tbody>
</table>

To decrypt we follow essentially the same procedure, except that we ‘subtract the key word from the ciphertext’ to regain the plaintext.

Using a keyword of length $m$, each plaintext letter $x$ is mapped to one of $m$ possible ciphertext letters. A cipher with this property is called **polyalphabetic**. Usually cryptanalysis is more difficult for polyalphabetic than monoalphabetic ciphers. For the Vigenere cipher, the number of possible keywords of length $m$ is $26^m$, which is quite large even for small values of $m$; for $m = 6$ this is more than $3 \times 10^8$. So for quite moderate keyword lengths, it becomes infeasible to search through all possible keywords.

In all ciphers considered so far, successive plaintext letters, or groups of letters, have been encrypted using the same key $K$, that is the ciphertext string $y$ is obtained as $y = y_1 y_2 \cdots = e_K(x_1) e_K(x_2) \cdots$. This type of cryptosystem is called a **block cipher**. For example, for a substitution cipher, each plaintext letter $x_i$ is encrypted with the single encryption rule $e_K$ to give a ciphertext letter $y_i = e_K(x_i)$. Thus a substitution cipher is a block cipher with blocks of length 1. On the other hand, for a Vigenere cipher with key word $K = (k_1, k_2, \ldots, k_m)$ of length $m$, $m$ different encryption rules $e_{k_1}, \ldots, e_{k_m}$ are used for encrypting successive plaintext letters in each block of $m$ plaintext letters, say $x_1, x_2, \ldots, x_m$ is encrypted as $e_{k_1}(x_1), \ldots, e_{k_m}(x_m)$. We can think of this as a composite encryption rule $e_K$ being applied to the string producing $e_K(x_1 x_2 \cdots x_m)$, a ciphertext string of length $m$. Thus we think of the Vigenere cipher as being a block cipher with blocks of length $m$: we apply the same composite encryption rule $e_K$ to each block of length $m$ of the plaintext.

We often want to send encrypted messages electronically and so we may wish to convert English symbols and numerals (whatever is in our normal plaintext alphabet) into strings of integers modulo 2 (that is, elements of $\mathbb{Z}_2$, also called binary integers or **bits**). For example, since there are $2^6 = 64$ different bit strings of length 6, we would be able to assign a different bit string of length 6 to each of the 26 lower case letters, each of the 26 upper case letters, and each of the ten numerals, and still have two left over. Suppose we decided to group our plaintext into strings with five letters or numerals in each string, and to convert each letter or numeral into a bit string of length 6. Then we would have converted the plaintext into blocks each consisting of 30 bits. A cipher that encrypts these 30-bit strings block by block would be a block cipher with blocks of length 30. As far as the cipher is concerned the encryption key is acting on bit strings of length 30 and the plaintext alphabet $P$ is the set of all 30-bit strings, so $P$ contains $2^{30} \approx 10^9$ different elements (30-bit strings).
Currently the most commonly used block ciphers have blocks consisting of bit strings of length 64. This means that the plaintext alphabet has $2^{64}$ elements! However 64 bit blocks are becoming insecure against modern computer cryptanalytic attack, and block ciphers are moving towards having blocks of 1024 or 2048 bits.

Historically there are several other block ciphers we could mention that use different ideas, and that have influenced future developments. One polyalphabetic cipher that encrypts the plaintext $m$ letters at a time is the **Hill Cipher**, invented in 1929 by Lester S. Hill. It utilises linear algebra over $\mathbb{Z}_{26}$. The key $K$ is an invertible $m \times m$ matrix with entries in $\mathbb{Z}_{26}$, and each string of $m$ characters from the Plaintext is first converted to a sequence $x$ of $m$ numbers in $\mathbb{Z}_{26}$, so $x \in \mathbb{Z}_{26}^m$. Then $x$ is encrypted by multiplying by the matrix $K$: $e_K(x) = xK$. The decryption rule is $d_K(y) = yK^{-1}$.

**Example 1.11.1** Encrypt ‘good luck’, using the Hill Cipher with key

\[
K = \begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 3 \\
3 & 0 & 0 & 0
\end{pmatrix}
\]

First we write the plain text as a string of elements from $\mathbb{Z}_{26}$ using Table 1.4 and break it into blocks of length 4 (since $K$ is a $4 \times 4$ matrix). We find that **good luck** corresponds to $x_1 = (6, 14, 14, 3), x_2 = (11, 20, 2, 10)$. We encipher this in two blocks:

\[
e_K(x_1) = (6, 14, 14, 3) \begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 3 \\
3 & 0 & 0 & 0
\end{pmatrix} = (9, 6, 14, 16)
\]

\[
e_K(x_2) = (11, 20, 2, 10) \begin{pmatrix}
0 & 1 & 0 & 0 \\
0 & 0 & 1 & 0 \\
0 & 0 & 0 & 3 \\
3 & 0 & 0 & 0
\end{pmatrix} = (4, 11, 20, 6)
\]

Finally we convert these strings to ciphertext again using Table 1.4 and obtain **JGOQ ELUG**.

To decipher we reverse this procedure, but we need first to find $K^{-1}$. This can be done using row echelonisation if necessary - but it’s easy to check that $K^{-1}$ is the following matrix by checking that $KK^{-1} = K^{-1}K = I$.

\[
K^{-1} = \begin{pmatrix}
0 & 0 & 0 & 9 \\
1 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 \\
0 & 0 & 9 & 0
\end{pmatrix}
\]

So to deciher **JGOQ ELUG**, we use Table 1.4 to find the corresponding strings $y_1 = (9, 6, 14, 16)$ and $y_2 = (4, 11, 20, 6)$ over $\mathbb{Z}_{26}$, and then use the rule $d_K(y) = yK^{-1}$ to recover the original strings $d_K(y_1) = x_1$ and $d_K(y_2) = x_2$ and we finally convert $x_1, x_2$ back to the plain text **good luck** using Table 1.4.
An alternative to block ciphers is to use what are called **stream ciphers**. A **keystream** is generated $z = z_1z_2\ldots$ and used to encrypt a Plaintext $x = x_1x_2\ldots$ by the rule $y = e_z(x_1)e_z(x_2)\ldots$. The keystream elements $z_i$ usually depend on the original key $K$ and also on the preceding plaintext $x_1, x_2, \ldots, x_{i-1}$. This approach can be used to model mathematically many cryptosystems, including the one-time pad (see below).

Start with a key $K \in K$ and the plaintext string $x = x_1x_2\ldots$. For each $i$ there is a rule (function) $f_i$ for generating the keystream element $z_i$ based on the key $K$; $f_i$ may depend on $K$ and also on $x_1, x_2, \ldots, x_{i-1}$ and $z_1, z_2, \ldots, z_{i-1}$, that is,

$$z_i = f_i(K, x_1, x_2, \ldots, x_{i-1}, z_1, z_2, \ldots, z_{i-1}).$$

We use $z_i$ to generate $y_i = e_z(x_i)$. Thus we generate in order $z_1, y_1, z_2, y_2, \ldots$ ‘Bob’ must know the key $K$ and also the rules $f_i$; in addition he must know the decryption rules $d_z(.)$ to rule to allow him to find the $z_i$ and decrypt $x_i = d_z(y_i)$, for each $i$.

**Example 1.11.2** Encrypt ‘Fine’, using a Stream Cipher with key $z_1 = K = 3$ and $z_i = z_{i-1} + 2x_{i-1} \mod 26$, where $1 \leq i \leq 4$.

<table>
<thead>
<tr>
<th>$i$</th>
<th>plaintext $x_i$</th>
<th>key $z_i = z_{i-1} + 2x_{i-1}$</th>
<th>ciphertext $y_i = x_i + z_i \pmod{26}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>f</td>
<td>5</td>
<td>N</td>
</tr>
<tr>
<td>2</td>
<td>i</td>
<td>8</td>
<td>V</td>
</tr>
<tr>
<td>3</td>
<td>n</td>
<td>13</td>
<td>Q</td>
</tr>
<tr>
<td>4</td>
<td>e</td>
<td>4</td>
<td>H</td>
</tr>
</tbody>
</table>

Now decipher the ciphertext NVQH.

<table>
<thead>
<tr>
<th>$i$</th>
<th>ciphertext $y_i$</th>
<th>key $z_i = z_{i-1} + 2x_{i-1}$</th>
<th>plaintext $x_i = y_i - z_i \pmod{26}$</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>N</td>
<td>8</td>
<td>i</td>
</tr>
<tr>
<td>2</td>
<td>V</td>
<td>21</td>
<td>i</td>
</tr>
<tr>
<td>3</td>
<td>Q</td>
<td>16</td>
<td>n</td>
</tr>
<tr>
<td>4</td>
<td>H</td>
<td>7</td>
<td>e</td>
</tr>
</tbody>
</table>

The Example below is really a collection of important remarks.

**Example 1.11.3** (a) A block cipher can be thought of as a special type of stream cipher by taking a constant keystream $z_i = K$ for all $i$.

(b) A stream cipher is **periodic** with period $d$ if $z_{i+d} = z_i$ for all $i$. The Vigenere cipher with key word of length $m$ can be thought of as a periodic stream cipher with period $m$. In this case the keyword $K = (k_1, k_2, \ldots, k_m)$ gives the first $m$ elements of the keystream $z_1 = k_1, \ldots, z_m = k_m$, and the keystream just repeats itself from this point on. The encryption and decryption rules are simple: $e_z(x) = x + z, d_z(y) = y - z$.

(c) Stream ciphers are often defined with the plaintext, ciphertext, and keystream elements all being integers modulo 2 (bits). In this case

$$e_z(x) = x + z \pmod{2}, \text{ and } d_z(y) = y + z \pmod{2}$$
so encryption and decryption are easy. If we think of 0 as ‘false’ and 1 as ‘true’ then
addition mod 2 corresponds to the ‘exclusive-or’ operation and can be implemented
very efficiently in computer hardware.

(d) The case where the plaintext is a string of \( n \) integers mod 2, say \( x_1x_2\ldots x_n \),
and also the key \( K = (k_1,k_2,\ldots,k_n) \) is a string of the same length \( n \) of integers
mod 2 is called the **one-time pad**. Here

\[
e_K(x_1,x_2,\ldots,x_n) = (x_1 + k_1, x_2 + k_2,\ldots,x_n + k_n) \mod 2,
\]

and

\[
d_K(y_1,y_2,\ldots,y_n) = (y_1 + k_1, y_2 + k_2,\ldots,y_n + k_n) \mod 2.
\]

This was described and patented by Gilbert Vernam in 1917. It is an unconditionally
secure cryptosystem (provided the key is kept secret), and is important in military
and diplomatic contexts. However it has limited use commercially because of key
management problems: the key which must be communicated securely has length
equal to the message length. Historical efforts in cryptography have been to design
cryptosystems where one key can be used for many messages while maintaining (at
least) computational security.

There are many other interesting ‘classical’ ciphers. An excellent account can
be found in the **Code Book** by Simon Singh. This includes stories about many
of the ciphers used during World War II. Many books on classical cryptography
can be found in the library. More recently a cipher called **Solitaire** involving
a pack of cards for both encryption and decryption was suggested in the novel
‘Cryptonomicon’ by Neal Stephenson. This cipher was developed by Bruce Schneier
and a description can be found

http://www.counterpane.com/solitaire.html on the Counterpane Internet Secu-

rity web site.

1.12 Cryptanalysis

The golden rule for designers of cryptosystems is never to underestimate the crypt-
analyst.

**Kerckhoff’s Principle 1883:** Assume that the cryptanalyst (opponent) knows the cryptosystem.

It would be unwise to design a cryptosystem without assuming Kerckhoff’s Principle.
The goal is to achieve security while observing it. There are different levels of attack
on cryptosystems and the most common ones are listed in Table 1.6. The object
is always to determine the key \( K \). The ‘chosen ciphertext attack’ is in particular
relevant to public-key cryptosystems, which we discuss later.

Consider the weakest type of attack, the ‘ciphertext only attack’. Assume that
the plaintext is English or some other structured language. Then a frequency count
will give good guesses for the commonly occurring letters: simply work out the fre-
quency of commonly occurring letters in the ciphertext and compare these against
<table>
<thead>
<tr>
<th>Type</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ciphertext only</td>
<td>The opponent possesses a string of ciphertext $y$.</td>
</tr>
<tr>
<td>Known plaintext</td>
<td>The opponent possesses a string of plaintext $x$ and the corresponding ciphertext $y$.</td>
</tr>
<tr>
<td>Chosen plaintext</td>
<td>The opponent has obtained temporary access to the encryption machinery; has chosen a string of plaintext $x$ and constructed the corresponding ciphertext string $y$.</td>
</tr>
<tr>
<td>Chosen ciphertext</td>
<td>The opponent has obtained temporary access to the decryption machinery; has chosen a string of ciphertext $y$ and constructed the corresponding plaintext string $x$.</td>
</tr>
</tbody>
</table>

Table 1.6: Types of cryptanalytic attack

<table>
<thead>
<tr>
<th>letter</th>
<th>probability</th>
<th>letter</th>
<th>occurrence</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>.082</td>
<td>N</td>
<td>.067</td>
</tr>
<tr>
<td>B</td>
<td>.015</td>
<td>O</td>
<td>.075</td>
</tr>
<tr>
<td>C</td>
<td>.028</td>
<td>P</td>
<td>.019</td>
</tr>
<tr>
<td>D</td>
<td>.043</td>
<td>Q</td>
<td>.001</td>
</tr>
<tr>
<td>E</td>
<td>.127</td>
<td>R</td>
<td>.060</td>
</tr>
<tr>
<td>F</td>
<td>.022</td>
<td>S</td>
<td>.063</td>
</tr>
<tr>
<td>G</td>
<td>.020</td>
<td>T</td>
<td>.091</td>
</tr>
<tr>
<td>H</td>
<td>.061</td>
<td>U</td>
<td>.028</td>
</tr>
<tr>
<td>I</td>
<td>.070</td>
<td>V</td>
<td>.010</td>
</tr>
<tr>
<td>J</td>
<td>.002</td>
<td>W</td>
<td>.023</td>
</tr>
<tr>
<td>K</td>
<td>.008</td>
<td>X</td>
<td>.001</td>
</tr>
<tr>
<td>L</td>
<td>.040</td>
<td>Y</td>
<td>.020</td>
</tr>
<tr>
<td>M</td>
<td>.024</td>
<td>Z</td>
<td>.001</td>
</tr>
</tbody>
</table>

Table 1.7: Probability of occurrence of the 26 letters

the English frequency statistics, see Table 1.7, to guess what they stand for in the plaintext. The rest can be broken by the redundancy naturally present in English. (The data in Table 1.7 is from Beker and Piper, p.397.)

From Table 1.7, the most commonly occurring letters in English are, in order, E T A O I N S R H . . . . Also the most common pairs are TH, HE, IN, ER, AN, RE . . . , and the most common triples are THE, AND, ING, ION. You have an opportunity to try this out in a simple way with the last question in the Exercise Set on Classical Ciphers.