DISCRETE MATHEMATICS AND ITS APPLICATIONS
Series editor KENNETH H. ROSEN

CRYPTOGRAPHY
THEORY AND PRACTICE
THIRD EDITION

1 Classical Cryptography

In this chapter, we provide a gentle introduction to cryptography and cryptanalysis. We present several simple systems, and describe how they can be "broken." Along the way, we discuss various mathematical techniques that will be used throughout the book.

1.1 Introduction: Some Simple Cryptosystems

The fundamental objective of cryptography is to enable two people, usually referred to as Alice and Bob, to communicate over an insecure channel in such a way that an opponent, Oscar, cannot understand what is being said. This channel could be a telephone line or computer network, for example. The information that Alice wants to send to Bob, which we call "plaintext," can be English text, numerical data, or anything at all — its structure is completely arbitrary. Alice encrypts the plaintext, using a predetermined key, and sends the resulting ciphertext over the channel. Oscar, upon seeing the ciphertext in the channel by eavesdropping, cannot determine what the plaintext was; but Bob, who knows the encryption key, can decrypt the ciphertext and reconstruct the plaintext.

These ideas are described formally using the following mathematical notation.

Definition 1.1: A *cryptosystem* is a five-tuple $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$, where the following conditions are satisfied:

1. $\mathcal{P}$ is a finite set of possible *plaintexts*;
2. $\mathcal{C}$ is a finite set of possible *ciphertexts*;
3. $\mathcal{K}$, the *keyspace*, is a finite set of possible *keys*;
4. For each $K \in \mathcal{K}$, there is an *encryption rule* $e_K \in \mathcal{E}$ and a corresponding *decryption rule* $d_K \in \mathcal{D}$. Each $e_K : \mathcal{P} \to \mathcal{C}$ and $d_K : \mathcal{C} \to \mathcal{P}$ are functions such that $d_K(e_K(x)) = x$ for every plaintext element $x \in \mathcal{P}$.

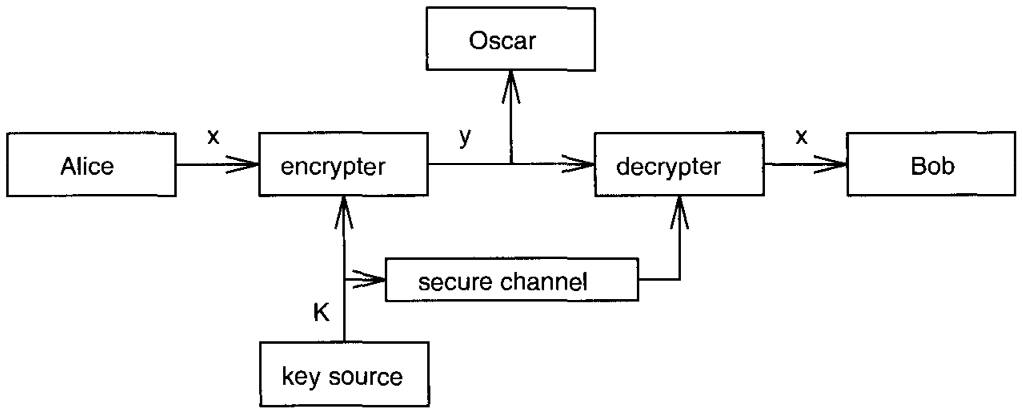

FIGURE 1.1
The communication channel

The main property is property 4. It says that if a plaintext x is encrypted using e_K, and the resulting ciphertext is subsequently decrypted using d_K, then the original plaintext x results.

Alice and Bob will employ the following protocol to use a specific cryptosystem. First, they choose a random key $K \in \mathcal{K}$. This is done when they are in the same place and are not being observed by Oscar, or, alternatively, when they do have access to a secure channel, in which case they can be in different places. At a later time, suppose Alice wants to communicate a message to Bob over an insecure channel. We suppose that this message is a string

$$\mathbf{x} = x_1 x_2 \cdots x_n$$

for some integer $n \geq 1$, where each plaintext symbol $x_i \in \mathcal{P}$, $1 \leq i \leq n$. Each x_i is encrypted using the encryption rule e_K specified by the predetermined key K. Hence, Alice computes $y_i = e_K(x_i)$, $1 \leq i \leq n$, and the resulting ciphertext string

$$\mathbf{y} = y_1 y_2 \cdots y_n$$

is sent over the channel. When Bob receives $y_1 y_2 \cdots y_n$, he decrypts it using the decryption function d_K, obtaining the original plaintext string, $x_1 x_2 \cdots x_n$. See Figure 1.1 for an illustration of the communication channel.

Clearly, it must be the case that each encryption function e_K is an *injective function* (i.e., one-to-one); otherwise, decryption could not be accomplished in an unambiguous manner. For example, if

$$y = e_K(x_1) = e_K(x_2)$$

where $x_1 \neq x_2$, then Bob has no way of knowing whether y should decrypt to x_1 or x_2. Note that if $\mathcal{P} = \mathcal{C}$, it follows that each encryption function is a permutation. That is, if the set of plaintexts and ciphertexts are identical, then each encryption function just rearranges (or permutes) the elements of this set.

1.1.1 The Shift Cipher

In this section, we will describe the *Shift Cipher*, which is based on modular arithmetic. But first we review some basic definitions of modular arithmetic.

Definition 1.2: Suppose a and b are integers, and m is a positive integer. Then we write $a \equiv b \pmod{m}$ if m divides $b-a$. The phrase $a \equiv b \pmod{m}$ is called a *congruence*, and it is read as "a is *congruent* to b modulo m." The integer m is called the *modulus*.

Suppose we divide a and b by m, obtaining integer quotients and remainders, where the remainders are between 0 and $m-1$. That is, $a = q_1m + r_1$ and $b = q_2m + r_2$, where $0 \leq r_1 \leq m-1$ and $0 \leq r_2 \leq m-1$. Then it is not difficult to see that $a \equiv b \pmod{m}$ if and only if $r_1 = r_2$. We will use the notation $a \bmod m$ (without parentheses) to denote the remainder when a is divided by m, i.e., the value r_1 above. Thus $a \equiv b \pmod{m}$ if and only if $a \bmod m = b \bmod m$. If we replace a by $a \bmod m$, we say that a is *reduced* modulo m.

We give a couple of examples. To compute $101 \bmod 7$, we write $101 = 7 \times 14 + 3$. Since $0 \leq 3 \leq 6$, it follows that $101 \bmod 7 = 3$. As another example, suppose we want to compute $(-101) \bmod 7$. In this case, we write $-101 = 7 \times (-15) + 4$. Since $0 \leq 4 \leq 6$, it follows that $(-101) \bmod 7 = 4$.

REMARK Many computer programming languages define $a \bmod m$ to be the remainder in the range $-m+1, \dots, m-1$ having the same sign as a. For example, $(-101) \bmod 7$ would be -3, rather than 4 as we defined it above. But for our purposes, it is much more convenient to define $a \bmod m$ always to be non-negative. ▮

We now define arithmetic modulo m: $\mathbb{Z}_m$ is the set $\{0, \dots, m-1\}$, equipped with two operations, $+$ and $\times$. Addition and multiplication in $\mathbb{Z}_m$ work exactly like real addition and multiplication, except that the results are reduced modulo m.

For example, suppose we want to compute 11×13 in $\mathbb{Z}_{16}$. As integers, we have $11 \times 13 = 143$. Then we reduce 143 modulo 16 as described above: $143 = 8 \times 16 + 15$, so $143 \bmod 16 = 15$, and hence $11 \times 13 = 15$ in $\mathbb{Z}_{16}$.

These definitions of addition and multiplication in $\mathbb{Z}_m$ satisfy most of the familiar rules of arithmetic. We will list these properties now, without proof:

1. addition is *closed*, i.e., for any $a, b \in \mathbb{Z}_m$, $a + b \in \mathbb{Z}_m$
2. addition is *commutative*, i.e., for any $a, b \in \mathbb{Z}_m$, $a + b = b + a$
3. addition is *associative*, i.e., for any $a, b, c \in \mathbb{Z}_m$, $(a + b) + c = a + (b + c)$
4. 0 is an *additive identity*, i.e., for any $a \in \mathbb{Z}_m$, $a + 0 = 0 + a = a$

Cryptosystem 1.1: *Shift Cipher*

Let $\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}$. For $0 \leq K \leq 25$, define

$$e_K(x) = (x + K) \bmod 26$$

and

$$d_K(y) = (y - K) \bmod 26$$

$(x, y \in \mathbb{Z}_{26})$.

5. the *additive inverse* of any $a \in \mathbb{Z}_m$ is $m-a$, i.e., $a+(m-a) = (m-a)+a = 0$ for any $a \in \mathbb{Z}_m$
6. multiplication is *closed*, i.e., for any $a, b \in \mathbb{Z}_m$, $ab \in \mathbb{Z}_m$
7. multiplication is *commutative*, i.e., for any $a, b \in \mathbb{Z}_m$, $ab = ba$
8. multiplication is *associative*, i.e., for any $a, b, c \in \mathbb{Z}_m$, $(ab)c = a(bc)$
9. 1 is a *multiplicative identity*, i.e., for any $a \in \mathbb{Z}_m$, $a \times 1 = 1 \times a = a$
10. the *distributive property* is satisfied, i.e., for any $a, b, c \in \mathbb{Z}_m$, $(a + b)c = (ac) + (bc)$ and $a(b + c) = (ab) + (ac)$.

Properties 1, 3–5 say that $\mathbb{Z}_m$ forms an algebraic structure called a *group* with respect to the addition operation. Since property 2 also holds, the group is said to be an *abelian group*.

Properties 1–10 establish that $\mathbb{Z}_m$ is, in fact, a *ring*. We will see many other examples of groups and rings in this book. Some familiar examples of rings include the integers, $\mathbb{Z}$; the real numbers, $\mathbb{R}$; and the complex numbers, $\mathbb{C}$. However, these are all infinite rings, and our attention will be confined almost exclusively to finite rings.

Since additive inverses exist in $\mathbb{Z}_m$, we can also subtract elements in $\mathbb{Z}_m$. We define $a - b$ in $\mathbb{Z}_m$ to be $(a - b) \bmod m$. That is, we compute the integer $a - b$ and then reduce it modulo m. For example, to compute $11 - 18$ in $\mathbb{Z}_{31}$, we first subtract 18 from 11, obtaining -7, and then compute $(-7) \bmod 31 = 24$.

We present the *Shift Cipher* as Cryptosystem 1.1. It is defined over $\mathbb{Z}_{26}$ since there are 26 letters in the English alphabet, though it could be defined over $\mathbb{Z}_m$ for any modulus m. It is easy to see that the *Shift Cipher* forms a cryptosystem as defined above, i.e., $d_K(e_K(x)) = x$ for every $x \in \mathbb{Z}_{26}$.

REMARK For the particular key $K = 3$, the cryptosystem is often called the *Caesar Cipher*, which was purportedly used by Julius Caesar. ▮

We would use the *Shift Cipher* (with a modulus of 26) to encrypt ordinary English text by setting up a correspondence between alphabetic characters and residues modulo 26 as follows: $A \leftrightarrow 0$, $B \leftrightarrow 1, \ldots, Z \leftrightarrow 25$. Since we will be using this correspondence in several examples, let's record it for future use:

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

A small example will illustrate.

Example 1.1 Suppose the key for a *Shift Cipher* is $K = 11$, and the plaintext is

```
wewillmeetatmidnight.
```

We first convert the plaintext to a sequence of integers using the specified correspondence, obtaining the following:

$$\begin{array}{rrrrrrrrrr} 22 & 4 & 22 & 8 & 11 & 11 & 12 & 4 & 4 & 19 \\ 0 & 19 & 12 & 8 & 3 & 13 & 8 & 6 & 7 & 19 \end{array}$$

Next, we add 11 to each value, reducing each sum modulo 26:

$$\begin{array}{rrrrrrrrrr} 7 & 15 & 7 & 19 & 22 & 22 & 23 & 15 & 15 & 4 \\ 11 & 4 & 23 & 19 & 14 & 24 & 19 & 17 & 18 & 4 \end{array}$$

Finally, we convert the sequence of integers to alphabetic characters, obtaining the ciphertext:

```
HPHTWWXPPELEXTOYTRSE.
```

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

REMARK In the above example we are using upper case letters for ciphertext and lower case letters for plaintext, in order to improve readability. We will do this elsewhere as well. ▌

If a cryptosystem is to be of practical use, it should satisfy certain properties. We informally enumerate two of these properties now.

1. Each encryption function e_K and each decryption function d_K should be efficiently computable.
2. An opponent, upon seeing a ciphertext string $\mathbf{y}$, should be unable to determine the key K that was used, or the plaintext string $\mathbf{x}$.

The second property is defining, in a very vague way, the idea of "security." The process of attempting to compute the key K, given a string of ciphertext $\mathbf{y}$, is called *cryptanalysis*. (We will make these concepts more precise as we proceed.) Note that, if Oscar can determine K, then he can decrypt $\mathbf{y}$ just as Bob would, using d_K. Hence, determining K is at least as difficult as determining the plaintext string $\mathbf{x}$, given the ciphertext string $\mathbf{y}$.

We observe that the *Shift Cipher* (modulo 26) is not secure, since it can be cryptanalyzed by the obvious method of *exhaustive key search*. Since there are only 26 possible keys, it is easy to try every possible decryption rule d_K until a "meaningful" plaintext string is obtained. This is illustrated in the following example.

Example 1.2 Given the ciphertext string

```
JBCRCLQRWCRVNBJENBWRWN,
```

we successively try the decryption keys d_0, d_1, etc. The following is obtained:

```
jbcrclqrwcrvnbjenbwrwn
iabqbkpqvbqumaidmavqvm
hzapajopuaptlzhclzupul
gyzozinotzoskygbkytotk
fxynyhmnsynrjxfajxsnsj
ewxmxglmrxmqiweziwrmri
dvwlwfklqwlphvdyhvqlqh
cuvkvejkpvkogucxgupkpg
btujudijoujnftbwftojof
astitchintimesavesnine
```

At this point, we have determined the plaintext to be the phrase "a stitch in time saves nine," and we can stop. The key is $K = 9$. □

On average, a plaintext will be computed using this method after trying $26/2 = 13$ decryption rules.

As the above example indicates, a necessary condition for a cryptosystem to be secure is that an exhaustive key search should be infeasible; i.e., the keyspace should be very large. As might be expected, however, a large keyspace is not sufficient to guarantee security.

Cryptosystem 1.2: *Substitution Cipher*

Let $\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}$. $\mathcal{K}$ consists of all possible permutations of the 26 symbols $0, 1, \ldots, 25$. For each permutation $\pi \in \mathcal{K}$, define

$$e_\pi(x) = \pi(x),$$

and define

$$d_\pi(y) = \pi^{-1}(y),$$

where π^{-1} is the inverse permutation to π.

1.1.2 The Substitution Cipher

Another well-known cryptosystem is the *Substitution Cipher*, which we define now. This cryptosystem has been used for hundreds of years. Puzzle "cryptograms" in newspapers are examples of *Substitution Ciphers*. This cipher is defined as Cryptosystem 1.2.

Actually, in the case of the *Substitution Cipher*, we might as well take $\mathcal{P}$ and $\mathcal{C}$ both to be the 26-letter English alphabet. We used $\mathbb{Z}_{26}$ in the *Shift Cipher* because encryption and decryption were algebraic operations. But in the *Substitution Cipher*, it is more convenient to think of encryption and decryption as permutations of alphabetic characters.

Here is an example of a "random" permutation, π, which could comprise an encryption function. (As before, plaintext characters are written in lower case and ciphertext characters are written in upper case.)

a	b	c	d	e	f	g	h	i	j	k	l	m
X	N	Y	A	H	P	O	G	Z	Q	W	B	T

n	o	p	q	r	s	t	u	v	w	x	y	z
S	F	L	R	C	V	M	U	E	K	J	D	I

Thus, $e_\pi(a) = X$, $e_\pi(b) = N$, etc. The decryption function is the inverse permutation. This is formed by writing the second lines first, and then sorting in alphabetical order. The following is obtained:

A	B	C	D	E	F	G	H	I	J	K	L	M
d	l	r	y	v	o	h	e	z	x	w	p	t

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
b	g	f	j	q	n	m	u	s	k	a	c	i

Hence, $d_\pi(A) = d$, $d_\pi(B) = l$, etc.

As an exercise, the reader might decrypt the following ciphertext using this decryption function:

```
MGZVYZLGHCMHJMYXSSFMNHAHYCDLMHA.
```

A key for the *Substitution Cipher* just consists of a permutation of the 26 alphabetic characters. The number of possible permutations is 26!, which is more than 4.0×10^{26}, a very large number. Thus, an exhaustive key search is infeasible, even for a computer. However, we shall see later that a *Substitution Cipher* can easily be cryptanalyzed by other methods.

1.1.3 The Affine Cipher

The *Shift Cipher* is a special case of the *Substitution Cipher* which includes only 26 of the 26! possible permutations of 26 elements. Another special case of the *Substitution Cipher* is the *Affine Cipher*, which we describe now. In the *Affine Cipher*, we restrict the encryption functions to functions of the form

$$e(x) = (ax + b) \bmod 26,$$

$a, b \in \mathbb{Z}_{26}$. These functions are called *affine functions*, hence the name *Affine Cipher*. (Observe that when $a = 1$, we have a *Shift Cipher*.)

In order that decryption is possible, it is necessary to ask when an affine function is injective. In other words, for any $y \in \mathbb{Z}_{26}$, we want the congruence

$$ax + b \equiv y \pmod{26}$$

to have a unique solution for x. This congruence is equivalent to

$$ax \equiv y - b \pmod{26}.$$

Now, as y varies over $\mathbb{Z}_{26}$, so, too, does $y - b$ vary over $\mathbb{Z}_{26}$. Hence, it suffices to study the congruence $ax \equiv y \pmod{26}$ ($y \in \mathbb{Z}_{26}$).

We claim that this congruence has a unique solution for every y if and only if $\gcd(a, 26) = 1$ (where the gcd function denotes the greatest common divisor of its arguments). First, suppose that $\gcd(a, 26) = d > 1$. Then the congruence $ax \equiv 0 \pmod{26}$ has (at least) two distinct solutions in $\mathbb{Z}_{26}$, namely $x = 0$ and $x = 26/d$. In this case $e(x) = (ax + b) \bmod 26$ is not an injective function and hence not a valid encryption function.

For example, since $\gcd(4, 26) = 2$, it follows that $4x + 7$ is not a valid encryption function: x and $x + 13$ will encrypt to the same value, for any $x \in \mathbb{Z}_{26}$.

Let's next suppose that $\gcd(a, 26) = 1$. Suppose for some x_1 and x_2 that

$$ax_1 \equiv ax_2 \pmod{26}.$$

Then

$$a(x_1 - x_2) \equiv 0 \pmod{26},$$

and thus

$$26 \mid a(x_1 - x_2).$$

We now make use of a fundamental property of integer division: if $\gcd(a, b) = 1$ and $a \mid bc$, then $a \mid c$. Since $26 \mid a(x_1 - x_2)$ and $\gcd(a, 26) = 1$, we must therefore have that

$$26 \mid (x_1 - x_2),$$

i.e., $x_1 \equiv x_2 \pmod{26}$.

At this point we have shown that, if $\gcd(a, 26) = 1$, then a congruence of the form $ax \equiv y \pmod{26}$ has, at most, one solution in $\mathbb{Z}_{26}$. Hence, if we let x vary over $\mathbb{Z}_{26}$, then $ax \bmod 26$ takes on 26 distinct values modulo 26. That is, it takes on every value exactly once. It follows that, for any $y \in \mathbb{Z}_{26}$, the congruence $ax \equiv y \pmod{26}$ has a unique solution for x.

There is nothing special about the number 26 in this argument. The following result can be proved in an analogous fashion.

THEOREM 1.1 *The congruence* $ax \equiv b \pmod{m}$ *has a unique solution* $x \in \mathbb{Z}_m$ *for every* $b \in \mathbb{Z}_m$ *if and only if* $\gcd(a, m) = 1$.

Since $26 = 2 \times 13$, the values of $a \in \mathbb{Z}_{26}$ such that $\gcd(a, 26) = 1$ are $a = 1$, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23, and 25. The parameter b can be any element in $\mathbb{Z}_{26}$. Hence the *Affine Cipher* has $12 \times 26 = 312$ possible keys. (Of course, this is much too small to be secure.)

Let's now consider the general setting where the modulus is m. We need another definition from number theory.

Definition 1.3: Suppose $a \geq 1$ and $m \geq 2$ are integers. If $\gcd(a, m) = 1$, then we say that a and m are *relatively prime*. The number of integers in $\mathbb{Z}_m$ that are relatively prime to m is often denoted by $\phi(m)$ (this function is called the *Euler phi-function*).

A well-known result from number theory gives the value of $\phi(m)$ in terms of the prime power factorization of m. (An integer $p > 1$ is *prime* if it has no positive divisors other than 1 and p. Every integer $m > 1$ can be factored as a product of powers of primes in a unique way. For example, $60 = 2^2 \times 3 \times 5$ and $98 = 2 \times 7^2$.)

We record the formula for $\phi(m)$ in the following theorem.

THEOREM 1.2 *Suppose*

$$m = \prod_{i=1}^{n} {p_i}^{e_i},$$

where the p_i*'s are distinct primes and* $e_i > 0$, $1 \leq i \leq n$. *Then*

$$\phi(m) = \prod_{i=1}^{n} ({p_i}^{e_i} - p_i^{e_i - 1}).$$

It follows that the number of keys in the *Affine Cipher* over $\mathbb{Z}_m$ is $m\phi(m)$, where $\phi(m)$ is given by the formula above. (The number of choices for b is m, and the number of choices for a is $\phi(m)$, where the encryption function is $e(x) = ax + b$.) For example, suppose $m = 60$. We have

$$60 = 2^2 \times 3^1 \times 5^1$$

and hence

$$\phi(60) = (4-2) \times (3-1) \times (5-1) = 2 \times 2 \times 4 = 16.$$

The number of keys in the *Affine Cipher* is $60 \times 16 = 960$.

Let's now consider the decryption operation in the *Affine Cipher* with modulus $m = 26$. Suppose that $\gcd(a, 26) = 1$. To decrypt, we need to solve the congruence $y \equiv ax + b \pmod{26}$ for x. The discussion above establishes that the congruence will have a unique solution in $\mathbb{Z}_{26}$, but it does not give us an efficient method of finding the solution. What we require is an efficient algorithm to do this. Fortunately, some further results on modular arithmetic will provide us with the efficient decryption algorithm we seek.

We require the idea of a multiplicative inverse.

Definition 1.4: Suppose $a \in \mathbb{Z}_m$. The *multiplicative inverse* of a modulo m, denoted $a^{-1} \bmod m$, is an element $a' \in \mathbb{Z}_m$ such that $aa' \equiv a'a \equiv 1 \pmod{m}$. If m is fixed, we sometimes write a^{-1} for $a^{-1} \bmod m$.

By similar arguments to those used above, it can be shown that a has a multiplicative inverse modulo m if and only if $\gcd(a, m) = 1$; and if a multiplicative inverse exists, it is unique modulo m. Also, observe that if $b = a^{-1}$, then $a = b^{-1}$. If p is prime, then every non-zero element of $\mathbb{Z}_p$ has a multiplicative inverse. A ring in which this is true is called a *field*.

In a later section, we will describe an efficient algorithm for computing multiplicative inverses in $\mathbb{Z}_m$ for any m. However, in $\mathbb{Z}_{26}$, trial and error suffices to find the multiplicative inverses of the elements relatively prime to 26:

$$\begin{aligned}
1^{-1} &= 1,\\
3^{-1} &= 9,\\
5^{-1} &= 21,\\
7^{-1} &= 15,\\
11^{-1} &= 19,\\
17^{-1} &= 23, \text{and}\\
25^{-1} &= 25.
\end{aligned}$$

Cryptosystem 1.3: *Affine Cipher*

Let $\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}$ and let

$$\mathcal{K} = \{(a, b) \in \mathbb{Z}_{26} \times \mathbb{Z}_{26} : \gcd(a, 26) = 1\}.$$

For $K = (a, b) \in \mathcal{K}$, define

$$e_K(x) = (ax + b) \bmod 26$$

and

$$d_K(y) = a^{-1}(y - b) \bmod 26$$

$(x, y \in \mathbb{Z}_{26})$.

(All of these can be verified easily. For example, $7 \times 15 = 105 \equiv 1 \pmod{26}$, so $7^{-1} = 15$ and $15^{-1} = 7$.)

Consider our congruence $y \equiv ax + b \pmod{26}$. This is equivalent to

$$ax \equiv y - b \pmod{26}.$$

Since $\gcd(a, 26) = 1$, a has a multiplicative inverse modulo 26. Multiplying both sides of the congruence by a^{-1}, we obtain

$$a^{-1}(ax) \equiv a^{-1}(y - b) \pmod{26}.$$

By associativity of multiplication modulo 26, we have that

$$a^{-1}(ax) \equiv (a^{-1}a)x \equiv 1x \equiv x \pmod{26}.$$

Consequently, $x = a^{-1}(y - b) \bmod 26$. This is an explicit formula for x, that is, the decryption function is

$$d(y) = a^{-1}(y - b) \bmod 26.$$

So, finally, the complete description of the *Affine Cipher* is given as Cryptosystem 1.3.

Let's do a small example.

Example 1.3 Suppose that $K = (7, 3)$. As noted above, $7^{-1} \bmod 26 = 15$. The encryption function is

$$e_K(x) = 7x + 3,$$

and the corresponding decryption function is

$$d_K(y) = 15(y-3) = 15y - 19,$$

where all operations are performed in $\mathbb{Z}_{26}$. It is a good check to verify that $d_K(e_K(x)) = x$ for all $x \in \mathbb{Z}_{26}$. Computing in $\mathbb{Z}_{26}$, we get

$$\begin{aligned} d_K(e_K(x)) &= d_K(7x+3) \\ &= 15(7x+3) - 19 \\ &= x + 45 - 19 \\ &= x. \end{aligned}$$

To illustrate, let's encrypt the plaintext *hot*. We first convert the letters h, o, t to residues modulo 26. These are respectively 7, 14, and 19. Now, we encrypt:

$$\begin{aligned} (7 \times 7 + 3) \bmod 26 &= 52 \bmod 26 &&= 0 \\ (7 \times 14 + 3) \bmod 26 &= 101 \bmod 26 &&= 23 \\ (7 \times 19 + 3) \bmod 26 &= 136 \bmod 26 &&= 6. \end{aligned}$$

So the three ciphertext characters are $0, 23$, and 6, which corresponds to the alphabetic string AXG. We leave the decryption as an exercise for the reader. □

1.1.4 The Vigenère Cipher

In both the *Shift Cipher* and the *Substitution Cipher*, once a key is chosen, each alphabetic character is mapped to a unique alphabetic character. For this reason, these cryptosystems are called *monoalphabetic cryptosystems*. We now present a cryptosystem which is not monoalphabetic, the well-known *Vigenère Cipher*, as Cryptosystem 1.4. This cipher is named after Blaise de Vigenère, who lived in the sixteenth century.

Using the correspondence $A \leftrightarrow 0$, $B \leftrightarrow 1, \dots, Z \leftrightarrow 25$ described earlier, we can associate each key K with an alphabetic string of length m, called a *keyword*. The *Vigenère Cipher* encrypts m alphabetic characters at a time: each plaintext element is equivalent to m alphabetic characters.

Let's do a small example.

Example 1.4 Suppose $m = 6$ and the keyword is $CIPHER$. This corresponds to the numerical equivalent $K = (2, 8, 15, 7, 4, 17)$. Suppose the plaintext is the string

`thiscryptosystemisnotsecure.`

We convert the plaintext elements to residues modulo 26, write them in groups of six, and then "add" the keyword modulo 26, as follows:

19	7	8	18	2	17	24	15	19	14	18	24
2	8	15	7	4	17	2	8	15	7	4	17
21	15	23	25	6	8	0	23	8	21	22	15

Cryptosystem 1.4: *Vigenère Cipher*

Let m be a positive integer. Define $\mathcal{P} = \mathcal{C} = \mathcal{K} = (\mathbb{Z}_{26})^m$. For a key $K = (k_1, k_2, \ldots, k_m)$, we define

$$e_K(x_1, x_2, \ldots, x_m) = (x_1 + k_1, x_2 + k_2, \ldots, x_m + k_m)$$

and

$$d_K(y_1, y_2, \ldots, y_m) = (y_1 - k_1, y_2 - k_2, \ldots, y_m - k_m),$$

where all operations are performed in $\mathbb{Z}_{26}$.

18	19	4	12	8	18	13	14	19	18	4	2
2	8	15	7	4	17	2	8	15	7	4	17
20	1	19	19	12	9	15	22	8	25	8	19

20	17	4
2	8	15
22	25	19

The alphabetic equivalent of the ciphertext string would thus be:

VPXZGIAXIVWPUBTTMJPWIZITWZT.

To decrypt, we can use the same keyword, but we would subtract it modulo 26 from the ciphertext, instead of adding. □

Observe that the number of possible keywords of length m in a *Vigenère Cipher* is 26^m, so even for relatively small values of m, an exhaustive key search would require a long time. For example, if we take $m = 5$, then the keyspace has size exceeding 1.1×10^7. This is already large enough to preclude exhaustive key search by hand (but not by computer).

In a *Vigenère Cipher* having keyword length m, an alphabetic character can be mapped to one of m possible alphabetic characters (assuming that the keyword contains m distinct characters). Such a cryptosystem is called a *polyalphabetic cryptosystem*. In general, cryptanalysis is more difficult for polyalphabetic than for monoalphabetic cryptosystems.

1.1.5 The Hill Cipher

In this section, we describe another polyalphabetic cryptosystem called the *Hill Cipher*. This cipher was invented in 1929 by Lester S. Hill. Let m be a positive integer, and define $\mathcal{P} = \mathcal{C} = (\mathbb{Z}_{26})^m$. The idea is to take m linear combinations of the m alphabetic characters in one plaintext element, thus producing the m alphabetic characters in one ciphertext element.

For example, if $m = 2$, we could write a plaintext element as $x = (x_1, x_2)$ and a ciphertext element as $y = (y_1, y_2)$. Here, y_1 would be a linear combination of x_1 and x_2, as would y_2. We might take

$$y_1 = (11x_1 + 3x_2) \bmod 26$$
$$y_2 = (8x_1 + 7x_2) \bmod 26.$$

Of course, this can be written more succinctly in matrix notation as follows:

$$(y_1, y_2) = (x_1, x_2)\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix},$$

where all operations are performed in $\mathbb{Z}_{26}$. In general, we will take an $m \times m$ matrix K as our key. If the entry in row i and column j of K is $k_{i,j}$, then we write $K = (k_{i,j})$. For $x = (x_1, \ldots, x_m) \in \mathcal{P}$ and $K \in \mathcal{K}$, we compute $y = e_K(x) = (y_1, \ldots, y_m)$ as follows:

$$(y_1, y_2, \ldots, y_m) = (x_1, x_2, \ldots, x_m)\begin{pmatrix} k_{1,1} & k_{1,2} & \ldots & k_{1,m} \\ k_{2,1} & k_{2,2} & \ldots & k_{2,m} \\ \vdots & \vdots & & \vdots \\ k_{m,1} & k_{m,2} & \ldots & k_{m,m} \end{pmatrix}.$$

In other words, using matrix notation, $y = xK$.

We say that the ciphertext is obtained from the plaintext by means of a *linear transformation*. We have to consider how decryption will work, that is, how x can be computed from y. Readers familiar with linear algebra will realize that we will use the inverse matrix K^{-1} to decrypt. The ciphertext is decrypted using the matrix equation $x = yK^{-1}$.

Here are the definitions of necessary concepts from linear algebra. If $A = (a_{i,j})$ is an $\ell \times m$ matrix and $B = (b_{j,k})$ is an $m \times n$ matrix, then we define the *matrix product* $AB = (c_{i,k})$ by the formula

$$c_{i,k} = \sum_{j=1}^{m} a_{i,j} b_{j,k}$$

for $1 \leq i \leq \ell$ and $1 \leq k \leq n$. That is, the entry in row i and column k of AB is formed by taking the ith row of A and the kth column of B, multiplying corresponding entries together, and summing. Note that AB is an $\ell \times n$ matrix.

Matrix multiplication is associative (that is, $(AB)C = A(BC)$) but not, in general, commutative (it is not always the case that $AB = BA$, even for square matrices A and B).

The $m \times m$ *identity matrix*, denoted by I_m, is the $m \times m$ matrix with 1's on the main diagonal and 0's elsewhere. Thus, the 2×2 identity matrix is

$$I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.$$

I_m is termed an identity matrix since $AI_m = A$ for any $\ell \times m$ matrix A and $I_m B = B$ for any $m \times n$ matrix B. Now, the ***inverse matrix*** of an $m \times m$ matrix A (if it exists) is the matrix A^{-1} such that $AA^{-1} = A^{-1}A = I_m$. Not all matrices have inverses, but if an inverse exists, it is unique.

With these facts at hand, it is easy to derive the decryption formula given above, assuming that K has an inverse matrix K^{-1}. Since $y = xK$, we can multiply both sides of the formula by K^{-1}, obtaining

$$yK^{-1} = (xK)K^{-1} = x(KK^{-1}) = xI_m = x.$$

(Note the use of the associativity property.)

We can verify that the example encryption matrix defined above has an inverse in $\mathbb{Z}_{26}$:

$$\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}^{-1} = \begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}$$

since

$$\begin{aligned}\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}\begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix} &= \begin{pmatrix} 11 \times 7 + 8 \times 23 & 11 \times 18 + 8 \times 11 \\ 3 \times 7 + 7 \times 23 & 3 \times 18 + 7 \times 11 \end{pmatrix} \\ &= \begin{pmatrix} 261 & 286 \\ 182 & 131 \end{pmatrix} \\ &= \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}.\end{aligned}$$

(Remember that all arithmetic operations are done modulo 26.)

Let's now do an example to illustrate encryption and decryption in the *Hill Cipher.*

Example 1.5 Suppose the key is

$$K = \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}.$$

From the computations above, we have that

$$K^{-1} = \begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}.$$

Suppose we want to encrypt the plaintext *july*. We have two elements of plaintext to encrypt: $(9, 20)$ (corresponding to *ju*) and $(11, 24)$ (corresponding to *ly*). We compute as follows:

$$(9, 20)\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix} = (99 + 60, 72 + 140) = (3, 4)$$

and

$$(11,24)\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix} = (121+72, 88+168) = (11,22).$$

Hence, the encryption of *july* is *DELW*. To decrypt, Bob would compute:

$$(3,4)\begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix} = (9,20)$$

and

$$(11,22)\begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix} = (11,24).$$

Hence, the correct plaintext is obtained. □

At this point, we have shown that decryption is possible if K has an inverse. In fact, for decryption to be possible, it is necessary that K has an inverse. (This follows fairly easily from elementary linear algebra, but we will not give a proof here.) So we are interested precisely in those matrices K that are invertible.

The invertibility of a (square) matrix depends on the value of its determinant, which we define now.

Definition 1.5: Suppose that $A = (a_{i,j})$ is an $m \times m$ matrix. For $1 \leq i \leq m$, $1 \leq j \leq m$, define A_{ij} to be the matrix obtained from A by deleting the ith row and the jth column. The *determinant* of A, denoted $\det A$, is the value $a_{1,1}$ if $m = 1$. If $m > 1$, then $\det A$ is computed recursively from the formula

$$\det A = \sum_{j=1}^{m} (-1)^{i+j} a_{i,j} \det A_{ij},$$

where i is any fixed integer between 1 and m.

It is not at all obvious that the value of $\det A$ is independent of the choice of i in the formula given above, but it can be proved that this is indeed the case. It will be useful to write out the formulas for determinants of 2×2 and 3×3 matrices. If $A = (a_{i,j})$ is a 2×2 matrix, then

$$\det A = a_{1,1}a_{2,2} - a_{1,2}a_{2,1}.$$

If $A = (a_{i,j})$ is a 3×3 matrix, then

$$\begin{aligned}\det A = {} & a_{1,1}a_{2,2}a_{3,3} + a_{1,2}a_{2,3}a_{3,1} + a_{1,3}a_{2,1}a_{3,2} \\ & -(a_{1,1}a_{2,3}a_{3,2} + a_{1,2}a_{2,1}a_{3,3} + a_{1,3}a_{2,2}a_{3,1}).\end{aligned}$$

For large m, the recursive formula given in the definition above is not usually a very efficient method of computing the determinant of an $m \times m$ square matrix.

A preferred method is to compute the determinant using so-called "elementary row operations"; see any text on linear algebra.

Two important properties of determinants that we will use are $\det I_m = 1$; and the multiplication rule $\det(AB) = \det A \times \det B$.

A real matrix K has an inverse if and only if its determinant is non-zero. However, it is important to remember that we are working over $\mathbb{Z}_{26}$. The relevant result for our purposes is that a matrix K has an inverse modulo 26 if and only if $\gcd(\det K, 26) = 1$. To see that this condition is necessary, suppose K has an inverse, denoted K^{-1}. By the multiplication rule for determinants, we have

$$1 = \det I = \det(KK^{-1}) = \det K \det K^{-1}.$$

Hence, $\det K$ is invertible in $\mathbb{Z}_{26}$, which is true if and only if $\gcd(\det K, 26) = 1$.

Sufficiency of this condition can be established in several ways. We will give an explicit formula for the inverse of the matrix K. Define a matrix K^* to have as its (i, j)-entry the value $(-1)^{i+j} \det K_{ji}$. (Recall that K_{ji} is obtained from K by deleting the jth row and the ith column.) K^* is called the *adjoint matrix* of K. We state the following theorem, concerning inverses of matrices over $\mathbb{Z}_n$, without proof.

THEOREM 1.3 *Suppose $K = (k_{i,j})$ is an $m \times m$ matrix over $\mathbb{Z}_n$ such that $\det K$ is invertible in $\mathbb{Z}_n$. Then $K^{-1} = (\det K)^{-1}K^*$, where K^* is the adjoint matrix of K.*

REMARK The above formula for K^{-1} is not very efficient computationally, except for small values of m (e.g., $m = 2, 3$). For larger m, the preferred method of computing inverse matrices would involve performing elementary row operations on the matrix K. ▮

In the 2×2 case, we have the following formula, which is an immediate corollary of Theorem 1.3.

COROLLARY 1.4 *Suppose*

$$K = \begin{pmatrix} k_{1,1} & k_{1,2} \\ k_{2,1} & k_{2,2} \end{pmatrix}$$

is a matrix having entries in $\mathbb{Z}_n$, and $\det K = k_{1,1}k_{2,2} - k_{1,2}k_{2,1}$ is invertible in $\mathbb{Z}_n$. Then

$$K^{-1} = (\det K)^{-1} \begin{pmatrix} k_{2,2} & -k_{1,2} \\ -k_{2,1} & k_{1,1} \end{pmatrix}.$$

Let's look again at the example considered earlier. First, we have

$$\begin{aligned}\det\begin{pmatrix}11 & 8\\ 3 & 7\end{pmatrix} &= (11\times 7 - 8\times 3) \bmod 26\\ &= (77-24) \bmod 26\\ &= 53 \bmod 26\\ &= 1.\end{aligned}$$

Now, $1^{-1} \bmod 26 = 1$, so the inverse matrix is

$$\begin{pmatrix}11 & 8\\ 3 & 7\end{pmatrix}^{-1} = \begin{pmatrix}7 & 18\\ 23 & 11\end{pmatrix},$$

as we verified earlier.

Here is another example, using a 3×3 matrix.

Example 1.6 Suppose that

$$K = \begin{pmatrix}10 & 5 & 12\\ 3 & 14 & 21\\ 8 & 9 & 11\end{pmatrix},$$

where all entries are in $\mathbb{Z}_{26}$. The reader can verify that $\det K = 7$. In $\mathbb{Z}_{26}$, we have that $7^{-1} \bmod 26 = 15$. The adjoint matrix is

$$K^* = \begin{pmatrix}17 & 1 & 15\\ 5 & 14 & 8\\ 19 & 2 & 21\end{pmatrix}.$$

Finally, the inverse matrix is

$$K^{-1} = 15K^* = \begin{pmatrix}21 & 15 & 17\\ 23 & 2 & 16\\ 25 & 4 & 3\end{pmatrix}.$$

□

As mentioned above, encryption in the *Hill Cipher* is done by multiplying the plaintext by the matrix K, while decryption multiplies the ciphertext by the inverse matrix K^{-1}. We now give a precise mathematical description of the *Hill Cipher* over $\mathbb{Z}_{26}$; see Cryptosystem 1.5.

Cryptosystem 1.5: *Hill Cipher*

Let $m \geq 2$ be an integer. Let $\mathcal{P} = \mathcal{C} = (\mathbb{Z}_{26})^m$ and let

$$\mathcal{K} = \{m \times m \text{ invertible matrices over } \mathbb{Z}_{26}\}.$$

For a key K, we define

$$e_K(x) = xK$$

and

$$d_K(y) = yK^{-1},$$

where all operations are performed in $\mathbb{Z}_{26}$.

1.1.6 The Permutation Cipher

All of the cryptosystems we have discussed so far involve substitution: plaintext characters are replaced by different ciphertext characters. The idea of a permutation cipher is to keep the plaintext characters unchanged, but to alter their positions by rearranging them using a permutation.

A *permutation* of a finite set X is a bijective function $\pi : X \rightarrow X$. In other words, the function π is one-to-one (injective) and onto (*surjective*). It follows that, for every $x \in X$, there is a unique element $x' \in X$ such that $\pi(x') = x$. This allows us to define the *inverse permutation*, $\pi^{-1} : X \rightarrow X$ by the rule

$$\pi^{-1}(x) = x' \quad \text{if and only if} \quad \pi(x') = x.$$

Then π^{-1} is also a permutation of X.

The *Permutation Cipher* (also known as the *Transposition Cipher*) is defined formally as Cryptosystem 1.6. This cryptosystem has been in use for hundreds of years. In fact, the distinction between the *Permutation Cipher* and the *Substitution Cipher* was pointed out as early as 1563 by Giovanni Porta.

As with the *Substitution Cipher*, it is more convenient to use alphabetic characters as opposed to residues modulo 26, since there are no algebraic operations being performed in encryption or decryption.

Here is an example to illustrate:

Example 1.7 Suppose $m = 6$ and the key is the following permutation π:

x	1	2	3	4	5	6
$\pi(x)$	3	5	1	6	4	2

Note that the first row of the above diagram lists the values of x, $1 \leq x \leq 6$, and the second row lists the corresponding values of $\pi(x)$. Then the inverse permuta-

Cryptosystem 1.6: *Permutation Cipher*

Let m be a positive integer. Let $\mathcal{P} = \mathcal{C} = (\mathbb{Z}_{26})^m$ and let $\mathcal{K}$ consist of all permutations of $\{1, \ldots, m\}$. For a key (i.e., a permutation) π, we define

$$e_\pi(x_1, \ldots, x_m) = (x_{\pi(1)}, \ldots, x_{\pi(m)})$$

and

$$d_\pi(y_1, \ldots, y_m) = (y_{\pi^{-1}(1)}, \ldots, y_{\pi^{-1}(m)}),$$

where π^{-1} is the inverse permutation to π.

tion π^{-1} can be constructed by interchanging the two rows, and rearranging the columns so that the first row is in increasing order. Carrying out these operations, we see that the permutation π^{-1} is the following:

x	1	2	3	4	5	6
$\pi^{-1}(x)$	3	6	1	5	2	4

Now, suppose we are given the plaintext

`shesellsseashellsbytheseashore.`

We first partition the plaintext into groups of six letters:

`shesel | lsseas | hellsb | ythese | ashore`

Now each group of six letters is rearranged according to the permutation π, yielding the following:

`EESLSH | SALSES | LSHBLE | HSYEET | HRAEOS`

So, the ciphertext is:

`EESLSHSALSESLSHBLEHSYEETHRAEOS.`

The ciphertext can be decrypted in a similar fashion, using the inverse permutation π^{-1}. □

We now show that the *Permutation Cipher* is a special case of the *Hill Cipher*. Given a permutation π of the set $\{1, \ldots, m\}$, we can define an associated $m \times m$ permutation matrix $K_\pi = (k_{i,j})$ according to the formula

$$k_{i,j} = \begin{cases} 1 & \text{if } i = \pi(j) \\ 0 & \text{otherwise.} \end{cases}$$

(A *permutation matrix* is a matrix in which every row and column contains exactly one "1," and all other values are "0." A permutation matrix can be obtained from an identity matrix by permuting rows or columns.)

It is not difficult to see that Hill encryption using the matrix K_π is, in fact, equivalent to permutation encryption using the permutation π. Moreover, $K_\pi^{-1} = K_{\pi^{-1}}$, i.e., the inverse matrix to K_π is the permutation matrix defined by the permutation π^{-1}. Thus, Hill decryption is equivalent to permutation decryption.

For the permutation π used in the example above, the associated permutation matrices are

$$K_\pi = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

and

$$K_\pi^{-1} = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}.$$

The reader can verify that the product of these two matrices is the identity matrix.

1.1.7 Stream Ciphers

In the cryptosystems we have studied so far, successive plaintext elements are encrypted using the same key, K. That is, the ciphertext string $\mathbf{y}$ is obtained as follows:

$$\mathbf{y} = y_1 y_2 \cdots = e_K(x_1) e_K(x_2) \cdots .$$

Cryptosystems of this type are often called *block ciphers*.

An alternative approach is to use what are called stream ciphers. The basic idea is to generate a keystream $\mathbf{z} = z_1 z_2 \cdots$, and use it to encrypt a plaintext string $\mathbf{x} = x_1 x_2 \cdots$ according to the rule

$$\mathbf{y} = y_1 y_2 \cdots = e_{z_1}(x_1) e_{z_2}(x_2) \cdots .$$

The simplest type of stream cipher is one in which the keystream is constructed from the key, independent of the plaintext string, using some specified algorithm. This type of stream cipher is called "synchronous" and can be defined formally as follows:

Definition 1.6: A *synchronous stream cipher* is a tuple $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{L}, \mathcal{E}, \mathcal{D})$, together with a function g, such that the following conditions are satisfied:

1. $\mathcal{P}$ is a finite set of possible *plaintexts*
2. $\mathcal{C}$ is a finite set of possible *ciphertexts*
3. $\mathcal{K}$, the *keyspace*, is a finite set of possible *keys*
4. $\mathcal{L}$ is a finite set called the *keystream alphabet*
5. g is the *keystream generator*. g takes a key K as input, and generates an infinite string $z_1 z_2 \cdots$ called the *keystream*, where $z_i \in \mathcal{L}$ for all $i \geq 1$.
6. For each $z \in \mathcal{L}$, there is an *encryption rule* $e_z \in \mathcal{E}$ and a corresponding *decryption rule* $d_z \in \mathcal{D}$. $e_z : \mathcal{P} \rightarrow \mathcal{C}$ and $d_z : \mathcal{C} \rightarrow \mathcal{P}$ are functions such that $d_z(e_z(x)) = x$ for every plaintext element $x \in \mathcal{P}$.

To illustrate this definition, we show how the *Vigenère Cipher* can be defined as a synchronous stream cipher. Suppose that m is the keyword length of a *Vigenère Cipher*. Define $\mathcal{K} = (\mathbb{Z}_{26})^m$ and $\mathcal{P} = \mathcal{C} = \mathcal{L} = \mathbb{Z}_{26}$; and define $e_z(x) = (x + z) \bmod 26$ and $d_z(y) = (y - z) \bmod 26$. Finally, define the keystream $z_1 z_2 \cdots$ as follows:

$$z_i = \begin{cases} k_i & \text{if } 1 \leq i \leq m \\ z_{i-m} & \text{if } i \geq m+1, \end{cases}$$

where $K = (k_1, \ldots, k_m)$. This generates the keystream

$$k_1 k_2 \cdots k_m k_1 k_2 \cdots k_m k_1 k_2 \cdots$$

from the key $K = (k_1, k_2, \ldots, k_m)$.

REMARK We can think of a block cipher as a special case of a stream cipher where the keystream is constant: $z_i = K$ for all $i \geq 1$. ▮

A stream cipher is a *periodic stream cipher* with period d if $z_{i+d} = z_i$ for all integers $i \geq 1$. The *Vigenère Cipher* with keyword length m, as described above, can be thought of as a periodic stream cipher with period m.

Stream ciphers are often described in terms of binary alphabets, i.e., $\mathcal{P} = \mathcal{C} = \mathcal{L} = \mathbb{Z}_2$. In this situation, the encryption and decryption operations are just addition modulo 2:

$$e_z(x) = (x + z) \bmod 2$$

and

$$d_z(y) = (y + z) \bmod 2.$$

If we think of "0" as representing the boolean value "false" and "1" as representing "true," then addition modulo 2 corresponds to the *exclusive-or* operation.

Hence, encryption (and decryption) can be implemented very efficiently in hardware.

Let's look at another method of generating a (synchronous) keystream. We will work over binary alphabets. Suppose we start with a binary m-tuple $(k_1, \ldots, k_m)$ and let $z_i = k_i$, $1 \leq i \leq m$ (as before). Now we generate the keystream using a *linear recurrence* of degree m:

$$z_{i+m} = \sum_{j=0}^{m-1} c_j z_{i+j} \bmod 2,$$

for all $i \geq 1$, where $c_0, \ldots, c_{m-1} \in \mathbb{Z}_2$ are specified constants.

REMARK This recurrence is said to have *degree* m since each term depends on the previous m terms. It is *linear* because z_{i+m} is a linear function of previous terms. Note that we can take $c_0 = 1$ without loss of generality, for otherwise the recurrence will be of degree (at most) $m - 1$. ▌

Here, the key K consists of the $2m$ values $k_1, \ldots, k_m, c_0, \ldots, c_{m-1}$. If

$$(k_1, \ldots, k_m) = (0, \ldots, 0),$$

then the keystream consists entirely of 0's. Of course, this should be avoided, as the ciphertext will then be identical to the plaintext. However, if the constants $c_0, \ldots, c_{m-1}$ are chosen in a suitable way, then any other initialization vector $(k_1, \ldots, k_m)$ will give rise to a periodic keystream having period $2^m - 1$. So a "short" key can give rise to a keystream having a very long period. This is certainly a desirable property: we will see in a later section how the *Vigenère Cipher* can be cryptanalyzed by exploiting the fact that the keystream has a short period.

Here is an example to illustrate.

Example 1.8 Suppose $m = 4$ and the keystream is generated using the linear recurrence

$$z_{i+4} = (z_i + z_{i+1}) \bmod 2,$$

$i \geq 1$. If the keystream is initialized with any vector other than $(0, 0, 0, 0)$, then we obtain a keystream of period 15. For example, starting with $(1, 0, 0, 0)$, the keystream is

$$1\,0\,0\,0\,1\,0\,0\,1\,1\,0\,1\,0\,1\,1\,1 \cdots.$$

Any other non-zero initialization vector will give rise to a cyclic permutation of the same keystream. □

Another appealing aspect of this method of keystream generation is that the keystream can be produced efficiently in hardware using a *linear feedback shift*

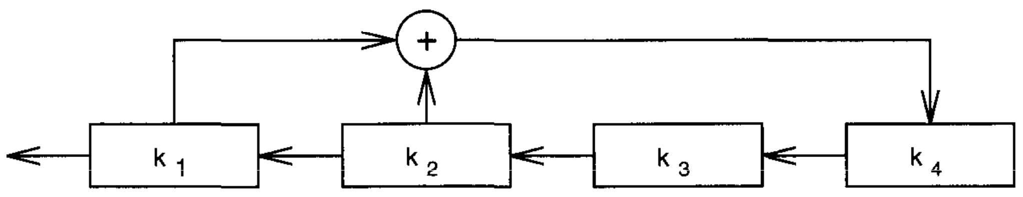

FIGURE 1.2
A linear feedback shift register

register, or LFSR. We would use a shift register with m stages. The vector $(k_1, \ldots, k_m)$ would be used to initialize the shift register. At each time unit, the following operations would be performed concurrently:

1. k_1 would be tapped as the next keystream bit
2. $k_2, \ldots, k_m$ would each be shifted one stage to the left
3. the "new" value of k_m would be computed to be

$$\sum_{j=0}^{m-1} c_j k_{j+1}$$

(this is the "linear feedback").

At any given point in time, the shift register contains m consecutive keystream elements, say $z_i, \ldots, z_{i+m-1}$. After one time unit, the shift register contains $z_{i+1}, \ldots, z_{i+m}$.

Observe that the linear feedback is carried out by tapping certain stages of the register (as specified by the constants c_j having the value "1") and computing a sum modulo 2 (which is an exclusive-or). This is illustrated in Figure 1.2, where we depict the LFSR that will generate the keystream of Example 1.8.

A *non-synchronous stream cipher* is a stream cipher in which each keystream element z_i depends on previous plaintext or ciphertext elements $(x_1, \ldots, x_{i-1}$ and/or $y_1, \ldots, y_{i-1})$ as well as the key K. A simple type of non-synchronous stream cipher, known as the *Autokey Cipher*, is presented as Cryptosystem 1.7. It is apparently due to Vigenère. The reason for the terminology "autokey" is that the plaintext is used to construct the keystream (aside from the initial "priming key" K). Of course, the *Autokey Cipher* is insecure since there are only 26 possible keys.

Here is an example to illustrate:

Example 1.9 Suppose the key is $K = 8$, and the plaintext is

`rendezvous.`

Cryptosystem 1.7: *Autokey Cipher*

Let $\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathcal{L} = \mathbb{Z}_{26}$. Let $z_1 = K$, and define $z_i = x_{i-1}$ for all $i \geq 2$. For $0 \leq z \leq 25$, define

$$e_z(x) = (x + z) \bmod 26$$

and

$$d_z(y) = (y - z) \bmod 26$$

$(x, y \in \mathbb{Z}_{26})$.

We first convert the plaintext to a sequence of integers:

$$17 \quad 4 \quad 13 \quad 3 \quad 4 \quad 25 \quad 21 \quad 14 \quad 20 \quad 18$$

The keystream is as follows:

$$8 \quad 17 \quad 4 \quad 13 \quad 3 \quad 4 \quad 25 \quad 21 \quad 14 \quad 20$$

Now we add corresponding elements, reducing modulo 26:

$$25 \quad 21 \quad 17 \quad 16 \quad 7 \quad 3 \quad 20 \quad 9 \quad 8 \quad 12$$

In alphabetic form, the ciphertext is:

`ZVRQHDUJIM.`

Now let's look at how the ciphertext would be decrypted. First, we convert the alphabetic string to the numeric string

$$25 \quad 21 \quad 17 \quad 16 \quad 7 \quad 3 \quad 20 \quad 9 \quad 8 \quad 12$$

Then we compute

$$x_1 = d_8(25) = (25 - 8) \bmod 26 = 17.$$

Next,

$$x_2 = d_{17}(21) = (21 - 17) \bmod 26 = 4,$$

and so on. Each time we obtain another plaintext character, we also use it as the next keystream element. □

In the next section, we discuss methods that can be used to cryptanalyze the various cryptosystems we have presented.

1.2 Cryptanalysis

In this section, we discuss some techniques of cryptanalysis. The general assumption that is usually made is that the opponent, Oscar, knows the cryptosystem being used. This is usually referred to as *Kerckhoffs' principle*. Of course, if Oscar does not know the cryptosystem being used, that will make his task more difficult. But we do not want to base the security of a cryptosystem on the (possibly shaky) premise that Oscar does not know what system is being employed. Hence, our goal in designing a cryptosystem will be to obtain security while assuming that Kerckhoffs' principle holds.

First, we want to differentiate between different attack models on cryptosystems. The *attack model* specifies the information available to the adversary when he mounts his attack. The most common types of attack models are enumerated as follows.

ciphertext only attack
The opponent possesses a string of ciphertext, $\mathtt{y}$.

known plaintext attack
The opponent possesses a string of plaintext, $\mathbf{x}$, and the corresponding ciphertext, $\mathtt{y}$.

chosen plaintext attack
The opponent has obtained temporary access to the encryption machinery. Hence he can choose a plaintext string, $\mathbf{x}$, and construct the corresponding ciphertext string, $\mathtt{y}$.

chosen ciphertext attack
The opponent has obtained temporary access to the decryption machinery. Hence he can choose a ciphertext string, $\mathtt{y}$, and construct the corresponding plaintext string, $\mathbf{x}$.

In each case, the objective of the adversary is to determine the key that was used. This would allow the opponent to decrypt a specific "target" ciphertext string, and further, to decrypt any additional ciphertext strings that are encrypted using the same key.[1]

We first consider the weakest type of attack, namely a ciphertext-only attack. We also assume that the plaintext string is ordinary English text, without punctuation or "spaces." (This makes cryptanalysis more difficult than if punctuation and spaces were encrypted.)

[1] At first glance, a chosen ciphertext attack may seem to be a bit artificial. For, if there is only one ciphertext string of interest to the opponent, then the opponent can obviously decrypt that ciphertext string if a chosen ciphertext attack is permitted. However, we are suggesting that the opponent's objective normally includes determining the key that is used by Alice and Bob, so that other ciphertext strings can be decrypted (at a later time, perhaps). A chosen ciphertext attack makes sense in this context.

TABLE 1.1
Probabilities of occurrence of the 26 letters

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

Many techniques of cryptanalysis use statistical properties of the English language. Various people have estimated the relative frequencies of the 26 letters by compiling statistics from numerous novels, magazines, and newspapers. The estimates in Table 1.1 were obtained by Beker and Piper. On the basis of these probabilities, Beker and Piper partition the 26 letters into five groups as follows:

1. E, having probability about 0.120
2. T, A, O, I, N, S, H, R, each having probability between 0.06 and 0.09
3. D, L, each having probability around 0.04
4. $C, U, M, W, F, G, Y, P, B$, each having probability between 0.015 and 0.028
5. V, K, J, X, Q, Z, each having probability less than 0.01.

It is also useful to consider sequences of two or three consecutive letters, called *digrams* and *trigrams*, respectively. The 30 most common digrams are (in decreasing order):

$$TH, HE, IN, ER, AN, RE, ED, ON, ES, ST,$$
$$EN, AT, TO, NT, HA, ND, OU, EA, NG, AS,$$
$$OR, TI, IS, ET, IT, AR, TE, SE, HI, OF.$$

The twelve most common trigrams are:

$$THE, ING, AND, HER, ERE, ENT,$$
$$THA, NTH, WAS, ETH, FOR, DTH.$$

1.2.1 Cryptanalysis of the Affine Cipher

As a simple illustration of how cryptanalysis can be performed using statistical data, let's look first at the *Affine Cipher*. Suppose Oscar has intercepted the fol-

TABLE 1.2
Frequency of occurrence of the 26 ciphertext letters

letter	frequency	letter	frequency
A	2	N	1
B	1	O	1
C	0	P	2
D	7	Q	0
E	5	R	8
F	4	S	3
G	0	T	0
H	5	U	2
I	0	V	4
J	0	W	0
K	5	X	2
L	2	Y	1
M	2	Z	0

lowing ciphertext:

Example 1.10 Ciphertext obtained from an *Affine Cipher*

```
FMXVEDKAPHFERBNDKRXRSREFMORUDSDKDVSHVUFEDK
APRKDLYEVLRHHRH
```

The frequency analysis of this ciphertext is given in Table 1.2.

There are only 57 characters of ciphertext, but this is usually sufficient to cryptanalyze an *Affine Cipher*. The most frequent ciphertext characters are: R (8 occurrences), D (7 occurrences), E, H, K (5 occurrences each), and F, S, V (4 occurrences each). As a first guess, we might hypothesize that R is the encryption of e and D is the encryption of t, since e and t are (respectively) the two most common letters. Expressed numerically, we have $e_K(4) = 17$ and $e_K(19) = 3$. Recall that $e_K(x) = ax + b$, where a and b are unknowns. So we get two linear equations in two unknowns:

$$4a + b = 17$$
$$19a + b = 3.$$

This system has the unique solution $a = 6$, $b = 19$ (in $\mathbb{Z}_{26}$). But this is an illegal key, since $\gcd(a, 26) = 2 > 1$. So our hypothesis must be incorrect.

Our next guess might be that R is the encryption of e and E is the encryption of t. Proceeding as above, we obtain $a = 13$, which is again illegal. So we try the next possibility, that R is the encryption of e and H is the encryption of t. This yields $a = 8$, again impossible. Continuing, we suppose that R is the encryption of e and K is the encryption of t. This produces $a = 3$, $b = 5$, which is at least a legal key. It remains to compute the decryption function corresponding to $K = (3, 5)$, and then to decrypt the ciphertext to see if we get a meaningful string of English, or nonsense. This will confirm the validity of $(3, 5)$.

If we perform these operations, we obtain $d_K(y) = 9y - 19$ and the given ciphertext decrypts to yield:

```
algorithmsarequitegeneraldefinitionsofarit
hmeticprocesses
```

We conclude that we have determined the correct key. □

1.2.2 Cryptanalysis of the Substitution Cipher

Here, we look at the more complicated situation, the *Substitution Cipher*. Consider the ciphertext in the following example:

Example 1.11 Ciphertext obtained from a *Substitution Cipher*

```
YIFQFMZRWQFYVECFMDZPCVMRZWNMDZVEJBTXCDDUMJ
NDIFEFMDZCDMQZKCEYFCJMYRNCWJCSZREXCHZUNMXZ
NZUCDRJXYYSMRTMEYIFZWDYVZVYFZUMRZCRWNZDZJJ
XZWGCHSMRNMDHNCMFQCHZJMXJZWIEJYUCFWDJNZDIR
```

The frequency analysis of this ciphertext is given in Table 1.3.

Since Z occurs significantly more often than any other ciphertext character, we might conjecture that $d_K(Z) = e$. The remaining ciphertext characters that occur at least ten times (each) are C, D, F, J, M, R, Y. We might expect that these letters are encryptions of (a subset of) t, a, o, i, n, s, h, r, but the frequencies really do not vary enough to tell us what the correspondence might be.

At this stage we might look at digrams, especially those of the form $-Z$ or $Z-$, since we conjecture that Z decrypts to e. We find that the most common digrams of this type are DZ and ZW (four times each); NZ and ZU (three times each); and RZ, HZ, XZ, FZ, ZR, ZV, ZC, ZD, and ZJ (twice each). Since ZW occurs four times and WZ not at all, and W occurs less often than many other characters, we might guess that $d_K(W) = d$. Since DZ occurs four times and

TABLE 1.3
Frequency of occurrence of the 26 ciphertext letters

letter	frequency	letter	frequency
A	0	N	9
B	1	O	0
C	15	P	1
D	13	Q	4
E	7	R	10
F	11	S	3
G	1	T	2
H	4	U	5
I	5	V	5
J	11	W	8
K	1	X	6
L	0	Y	10
M	16	Z	20

ZD occurs twice, we would think that $d_K(D) \in \{r, s, t\}$, but it is not clear which of the three possibilities is the correct one.

If we proceed on the assumption that $d_K(Z) = e$ and $d_K(W) = d$, we might look back at the ciphertext and notice that we have ZRW occurring near the beginning of the ciphertext, and RW occurs again later on. Since R occurs frequently in the ciphertext and nd is a common digram, we might try $d_K(R) = n$ as the most likely possibility.

At this point, we have the following:

```
------end---------e----ned---e------------
YIFQFMZRWQFYVECFMDZPCVMRZWNMDZVEJBTXCDDUMJ

--------e----e---------n--d---en----e----e
NDIFEFMDZCDMQZKCEYFCJMYRNCWJCSZREXCHZUNMXZ

-e---n------n------ed---e---e--ne-nd-e-e--
NZUCDRJXYYSMRTMEYIFZWDYVZVYFZUMRZCRWNZDZJJ

-ed-----n-----------e----ed-------d---e--n
XZWGCHSMRNMDHNCMFQCHZJMXJZWIEJYUCFWDJNZDIR
```

Our next step might be to try $d_K(N) = h$, since NZ is a common digram and ZN is not. If this is correct, then the segment of plaintext $ne - ndhe$ suggests that $d_K(C) = a$. Incorporating these guesses, we have:

```
------end-----a---e-a--nedh--e------a-----
YIFQFMZRWQFYVECFMDZPCVMRZWNMDZVEJBTXCDDUMJ

h-------ea---e-a---a---nhad-a-en--a-e-h--e
NDIFEFMDZCDMQZKCEYFCJMYRNCWJCSZREXCHZUNMXZ

he-a-n------n------ed---e---e--neandhe-e--
NZUCDRJXYYSMRTMEYIFZWDYVZVYFZUMRZCRWNZDZJJ

-ed-a---nh---ha---a-e----ed-----a-d--he--n
XZWGCHSMRNMDHNCMFQCHZJMXJZWIEJYUCFWDJNZDIR
```

Now, we might consider M, the second most common ciphertext character. The ciphertext segment RNM, which we believe decrypts to $nh-$, suggests that $h-$ begins a word, so M probably represents a vowel. We have already accounted for a and e, so we expect that $d_K(M) = i$ or o. Since ai is a much more likely digram than ao, the ciphertext digram CM suggests that we try $d_K(M) = i$ first. Then we have:

```
-----iend-----a-i-e-a-inedhi-e------a---i-
YIFQFMZRWQFYVECFMDZPCVMRZWNMDZVEJBTXCDDUMJ

h-----i-ea-i-e-a---a-i-nhad-a-en--a-e-hi-e
NDIFEFMDZCDMQZKCEYFCJMYRNCWJCSZREXCHZUNMXZ

he-a-n-----in-i----ed---e---e-ineandhe-e--
NZUCDRJXYYSMRTMEYIFZWDYVZVYFZUMRZCRWNZDZJJ

-ed-a--inhi--hai--a-e-i--ed-----a-d--he--n
XZWGCHSMRNMDHNCMFQCHZJMXJZWIEJYUCFWDJNZDIR
```

Next, we might try to determine which letter is the encryption of o. Since o is a common plaintext character, we guess that the corresponding ciphertext character is one of D, F, J, Y. Y seems to be the most likely possibility; otherwise, we would get long strings of vowels, namely aoi from CFM or CJM. Hence, let's suppose $d_K(Y) = o$.

The three most frequent remaining ciphertext letters are D, F, J, which we conjecture could decrypt to r, s, t in some order. Two occurrences of the trigram NMD suggest that $d_K(D) = s$, giving the trigram his in the plaintext (this is consistent with our earlier hypothesis that $d_K(D) \in \{r, s, t\}$). The segment $HNCMF$ could be an encryption of $chair$, which would give $d_K(F) = r$ (and $d_K(H) = c$) and so we would then have $d_K(J) = t$ by process of elimination. Now, we have:

```
o-r-riend-ro--arise-a-inedhise--t---ass-it
YIFQFMZRWQFYVECFMDZPCVMRZWNMDZVEJBTXCDDUMJ

hs-r-riseasi-e-a-orationhadta-en--ace-hi-e
NDIFEFMDZCDMQZKCEYFCJMYRNCWJCSZREXCHZUNMXZ

he-asnt-oo-in-i-o-redso-e-ore-ineandhesett
NZUCDRJXYYSMRTMEYIFZWDYVZVYFZUMRZCRWNZDZJJ

-ed-ac-inhischair-aceti-ted--to-ardsthes-n
XZWGCHSMRNMDHNCMFQCHZJMXJZWIEJYUCFWDJNZDIR
```

It is now very easy to determine the plaintext and the key for Example 1.11. The complete decryption is the following:

> Our friend from Paris examined his empty glass with surprise, as if evaporation had taken place while he wasn't looking. I poured some more wine and he settled back in his chair, face tilted up towards the sun.[2]

□

1.2.3 Cryptanalysis of the Vigenère Cipher

In this section we describe some methods for cryptanalyzing the *Vigenère Cipher*. The first step is to determine the keyword length, which we denote by m. There are a couple of techniques that can be employed. The first of these is the so-called Kasiski test and the second uses the index of coincidence.

The *Kasiski test* was described by Friedrich Kasiski in 1863; however, it was apparently discovered earlier, around 1854, by Charles Babbage. It is based on the observation that two identical segments of plaintext will be encrypted to the same ciphertext whenever their occurrence in the plaintext is δ positions apart, where $\delta \equiv 0 \pmod{m}$. Conversely, if we observe two identical segments of ciphertext, each of length at least three, say, then there is a good chance that they correspond to identical segments of plaintext.

The Kasiski test works as follows. We search the ciphertext for pairs of identical segments of length at least three, and record the distance between the starting positions of the two segments. If we obtain several such distances, say $\delta_1, \delta_2, \ldots$, then we would conjecture that m divides all of the δ_i's, and hence m divides the greatest common divisor of the δ_i's.

Further evidence for the value of m can be obtained by the index of coincidence. This concept was defined by William Friedman in 1920, as follows.

[2]P. Mayle, *A Year in Provence*, A. Knopf, Inc., 1989.

Definition 1.7: Suppose $\mathbf{x} = x_1 x_2 \cdots x_n$ is a string of n alphabetic characters. The *index of coincidence* of $\mathbf{x}$, denoted $I_c(\mathbf{x})$, is defined to be the probability that two random elements of $\mathbf{x}$ are identical.

Suppose we denote the frequencies of $A, B, C, \ldots, Z$ in $\mathbf{x}$ by $f_0, f_1, \ldots, f_{25}$ (respectively). We can choose two elements of $\mathbf{x}$ in $\binom{n}{2}$ ways.[3] For each i, $0 \leq i \leq 25$, there are $\binom{f_i}{2}$ ways of choosing both elements to be i. Hence, we have the formula

$$I_c(\mathbf{x}) = \frac{\sum_{i=0}^{25} \binom{f_i}{2}}{\binom{n}{2}} = \frac{\sum_{i=0}^{25} f_i(f_i - 1)}{n(n-1)}.$$

Suppose $\mathbf{x}$ is a string of English language text. Denote the expected probabilities of occurrence of the letters $A, B, \ldots, Z$ in Table 1.1 by $p_0, \ldots, p_{25}$, respectively. Then, we would expect that

$$I_c(\mathbf{x}) \approx \sum_{i=0}^{25} {p_i}^2 = 0.065,$$

since the probability that two random elements both are A is ${p_0}^2$, the probability that both are B is ${p_1}^2$, etc. The same reasoning applies if $\mathbf{x}$ is a ciphertext string obtained using any monoalphabetic cipher. In this case, the individual probabilities will be permuted, but the quantity $\sum {p_i}^2$ will be unchanged.

Now, suppose we start with a ciphertext string $\mathbf{y} = y_1 y_2 \cdots y_n$ that has been constructed by using a *Vigenère Cipher*. Define m substrings of $\mathbf{y}$, denoted $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m$, by writing out the ciphertext, in columns, in a rectangular array of dimensions $m \times (n/m)$. The rows of this matrix are the substrings $\mathbf{y}_i$, $1 \leq i \leq m$. In other words, we have that

$$\begin{aligned} \mathbf{y}_1 &= y_1 y_{m+1} y_{2m+1} \cdots, \\ \mathbf{y}_2 &= y_2 y_{m+2} y_{2m+2} \cdots, \\ \vdots \;\; &\vdots \;\; \vdots \\ \mathbf{y}_m &= y_m y_{2m} y_{3m} \cdots. \end{aligned}$$

If $\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_m$ are constructed in this way, and m is indeed the keyword length, then each value $I_c(\mathbf{y}_i)$ should be roughly equal to 0.065. On the other hand, if m is not the keyword length, then the substrings $\mathbf{y}_i$ will look much more random, since they will have been obtained by shift encryption with different keys. Observe that a completely random string will have

$$I_c \approx 26\left(\frac{1}{26}\right)^2 = \frac{1}{26} = 0.038.$$

[3]The *binomial coefficient* $\binom{n}{k} = n!/(k!(n-k)!)$ denotes the number of ways of choosing a subset of k objects from a set of n objects.

The two values 0.065 and 0.038 are sufficiently far apart that we will often be able to determine the correct keyword length by this method (or confirm a guess that has already been made using the Kasiski test).

Let us illustrate these two techniques with an example.

Example 1.12 Ciphertext obtained from a *Vigenère Cipher*

```
CHREEVOAHMAERATBIAXXWTNXBEEOPHBSBQMQEQERBW
RVXUOAKXAOSXXWEAHBWGJMMQMNKGRFVGXWTRZXWIAK
LXFPSKAUTEMNDCMGTSXMXBTUIADNGMGPSRELXNJELX
VRVPRTULHDNQWTWDTYGBPHXTFALJHASVBFXNGLLCHR
ZBWELEKMSJIKNBHWRJGNMGJSGLXFEYPHAGNRBIEQJT
AMRVLCRREMNDGLXRRIMGNSNRWCHRQHAEYEVTAQEBBI
PEEWEVKAKOEWADREMXMTBHHCHRTKDNVRZCHRCLQOHP
WQAIIWXNRMGWOIIFKEE
```

First, let's try the Kasiski test. The ciphertext string CHR occurs in five places in the ciphertext, beginning at positions 1, 166, 236, 276 and 286. The distances from the first occurrence to the other four occurrences are (respectively) 165, 235, 275 and 285. The greatest common divisor of these four integers is 5, so that is very likely the keyword length.

Let's see if computation of indices of coincidence gives the same conclusion. With $m = 1$, the index of coincidence is 0.045. With $m = 2$, the two indices are 0.046 and 0.041. With $m = 3$, we get 0.043, 0.050, 0.047. With $m = 4$, we have indices 0.042, 0.039, 0.045, 0.040. Then, trying $m = 5$, we obtain the values 0.063, 0.068, 0.069, 0.061 and 0.072. This also provides strong evidence that the keyword length is five. □

Assuming that we have determined the correct value of m, how do we determine the actual key, $K = (k_1, k_2, \ldots, k_m)$? We describe a simple and effective method now. Let $1 \leq i \leq m$, and let $f_0, \ldots, f_{25}$ denote the frequencies of $A, B, \ldots, Z$, respectively, in the string $\mathbf{y}_i$. Also, let $n' = n/m$ denote the length of the string $\mathbf{y}_i$. Then the probability distribution of the 26 letters in $\mathbf{y}_i$ is

$$\frac{f_0}{n'}, \ldots, \frac{f_{25}}{n'}.$$

Now, recall that the substring $\mathbf{y}_i$ is obtained by shift encryption of a subset of the plaintext elements using a shift k_i. Therefore, we would hope that the shifted probability distribution

$$\frac{f_{k_i}}{n'}, \ldots, \frac{f_{25+k_i}}{n'}$$

would be "close to" the ideal probability distribution $p_0, \ldots, p_{25}$ tabulated in Table 1.1, where subscripts in the above formula are evaluated modulo 26.

TABLE 1.4
Values of M_g

i	value of $M_g(\mathbf{y}_i)$								
1	.035	.031	.036	.037	.035	.039	.028	.028	.048
	$\boxed{.061}$	.039	.032	.040	.038	.038	.045	.036	.030
	.042	.043	.036	.033	.049	.043	.042	.036	
2	$\boxed{.069}$	.044	.032	.035	.044	.034	.036	.033	.029
	.031	.042	.045	.040	.045	.046	.042	.037	.032
	.034	.037	.032	.034	.043	.032	.026	.047	
3	.048	.029	.042	.043	.044	.034	.038	.035	.032
	.049	.035	.031	.035	$\boxed{.066}$	.035	.038	.036	.045
	.027	.035	.034	.034	.036	.035	.046	.040	
4	.045	.032	.033	.038	$\boxed{.060}$	.034	.034	.034	.050
	.033	.033	.043	.040	.033	.029	.036	.040	.044
	.037	.050	.034	.034	.039	.044	.038	.035	
5	.034	.031	.035	.044	.047	.037	.043	.038	.042
	.037	.033	.032	.036	.037	.036	.045	.032	.029
	.044	$\boxed{.072}$	.037	.027	.031	.048	.036	.037	

Suppose that $0 \leq g \leq 25$, and define the quantity

$$M_g = \sum_{i=0}^{25} \frac{p_i f_{i+g}}{n'}. \tag{1.1}$$

If $g = k_i$, then we would expect that

$$M_g \approx \sum_{i=0}^{25} {p_i}^2 = 0.065,$$

as in the consideration of the index of coincidence. If $g \neq k_i$, then M_g will usually be significantly smaller than 0.065 (see the Exercises for a justification of this statement). Hopefully this technique will allow us to determine the correct value of k_i for each value of i, $1 \leq i \leq m$.

Let us illustrate by returning to Example 1.12.

Example 1.12 (Cont.) We have hypothesized that the keyword length is 5. We now compute the values M_g as described above, for $1 \leq i \leq 5$. These values are tabulated in Table 1.4. For each i, we look for a value of M_g that is close to 0.065. These g's determine the shifts $k_1, \ldots, k_5$.

From the data in Table 1.4, we see that the key is likely to be $K = (9, 0, 13, 4, 19)$, and hence the keyword likely is *JANET*. This is correct, and the complete decryption of the ciphertext is the following:

The almond tree was in tentative blossom. The days were longer, often ending with magnificent evenings of corrugated pink skies. The hunting season was over, with hounds and guns put away for six months. The vineyards were busy again as the well-organized farmers treated their vines and the more lackadaisical neighbors hurried to do the pruning they should have done in November.[4]

□

1.2.4 Cryptanalysis of the Hill Cipher

The *Hill Cipher* can be difficult to break with a ciphertext-only attack, but it succumbs easily to a known plaintext attack. Let us first assume that the opponent has determined the value of m being used. Suppose he has at least m distinct plaintext-ciphertext pairs, say

$$x_j = (x_{1,j}, x_{2,j}, \ldots, x_{m,j})$$

and

$$y_j = (y_{1,j}, y_{2,j}, \ldots, y_{m,j}),$$

for $1 \leq j \leq m$, such that $y_j = e_K(x_j)$, $1 \leq j \leq m$. If we define two $m \times m$ matrices $X = (x_{i,j})$ and $Y = (y_{i,j})$, then we have the matrix equation $Y = XK$, where the $m \times m$ matrix K is the unknown key. Provided that the matrix X is invertible, Oscar can compute $K = X^{-1}Y$ and thereby break the system. (If X is not invertible, then it will be necessary to try other sets of m plaintext-ciphertext pairs.)

Let's look at a simple example.

Example 1.13 Suppose the plaintext *friday* is encrypted using a *Hill Cipher* with $m = 2$, to give the ciphertext *PQCFKU*.

We have that $e_K(5,17) = (15,16)$, $e_K(8,3) = (2,5)$ and $e_K(0,24) = (10,20)$. From the first two plaintext-ciphertext pairs, we get the matrix equation

$$\begin{pmatrix} 15 & 16 \\ 2 & 5 \end{pmatrix} = \begin{pmatrix} 5 & 17 \\ 8 & 3 \end{pmatrix} K.$$

Using Corollary 1.4, it is easy to compute

$$\begin{pmatrix} 5 & 17 \\ 8 & 3 \end{pmatrix}^{-1} = \begin{pmatrix} 9 & 1 \\ 2 & 15 \end{pmatrix},$$

so

$$K = \begin{pmatrix} 9 & 1 \\ 2 & 15 \end{pmatrix} \begin{pmatrix} 15 & 16 \\ 2 & 5 \end{pmatrix} = \begin{pmatrix} 7 & 19 \\ 8 & 3 \end{pmatrix}.$$

This can be verified by using the third plaintext-ciphertext pair. □

[4]P. Mayle, *A Year in Provence*, A. Knopf, Inc., 1989.

What would the opponent do if he does not know m? Assuming that m is not too big, he could simply try $m = 2, 3, \ldots$, until the key is found. If a guessed value of m is incorrect, then an $m \times m$ matrix found by using the algorithm described above will not agree with further plaintext-ciphertext pairs. In this way, the value of m can be determined if it is not known ahead of time.

1.2.5 Cryptanalysis of the LFSR Stream Cipher

Recall that the ciphertext is the sum modulo 2 of the plaintext and the keystream, i.e., $y_i = (x_i + z_i) \bmod 2$. The keystream is produced from an initial m-tuple, $(z_1, \ldots, z_m) = (k_1, \ldots, k_m)$, using the linear recurrence

$$z_{m+i} = \sum_{j=0}^{m-1} c_j z_{i+j} \bmod 2,$$

$i \geq 1$, where $c_0, \ldots, c_{m-1} \in \mathbb{Z}_2$.

Since all operations in this cryptosystem are linear, we might suspect that the cryptosystem is vulnerable to a known-plaintext attack, as is the case with the *Hill Cipher*. Suppose Oscar has a plaintext string $x_1 x_2 \cdots x_n$ and the corresponding ciphertext string $y_1 y_2 \cdots y_n$. Then he can compute the keystream bits $z_i = (x_i + y_i) \bmod 2$, $1 \leq i \leq n$. Let us also suppose that Oscar knows the value of m. Then Oscar needs only to compute $c_0, \ldots, c_{m-1}$ in order to be able to reconstruct the entire keystream. In other words, he needs to be able to determine the values of m unknowns.

Now, for any $i \geq 1$, we have

$$z_{m+i} = \sum_{j=0}^{m-1} c_j z_{i+j} \bmod 2,$$

which is a linear equation in the m unknowns. If $n \geq 2m$, then there are m linear equations in m unknowns, which can subsequently be solved.

The system of m linear equations can be written in matrix form as follows:

$$(z_{m+1}, z_{m+2}, \ldots, z_{2m}) = (c_0, c_1, \ldots, c_{m-1}) \begin{pmatrix} z_1 & z_2 & \cdots & z_m \\ z_2 & z_3 & \cdots & z_{m+1} \\ \vdots & \vdots & & \vdots \\ z_m & z_{m+1} & \cdots & z_{2m-1} \end{pmatrix}.$$

If the coefficient matrix has an inverse (modulo 2), we obtain the solution

$$(c_0, c_1, \ldots, c_{m-1}) = (z_{m+1}, z_{m+2}, \ldots, z_{2m}) \begin{pmatrix} z_1 & z_2 & \cdots & z_m \\ z_2 & z_3 & \cdots & z_{m+1} \\ \vdots & \vdots & & \vdots \\ z_m & z_{m+1} & \cdots & z_{2m-1} \end{pmatrix}^{-1}.$$

In fact, the matrix will have an inverse if m is the degree of the recurrence used to generate the keystream (see the Exercises for a proof).

Let's illustrate with an example.

Example 1.14 Suppose Oscar obtains the ciphertext string

$$101101011110010$$

corresponding to the plaintext string

$$011001111111000.$$

Then he can compute the keystream bits:

$$110100100001010.$$

Suppose also that Oscar knows that the keystream was generated using a 5-stage LFSR. Then he would solve the following matrix equation, which is obtained from the first 10 keystream bits:

$$(0,1,0,0,0) = (c_0,c_1,c_2,c_3,c_4)\begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}.$$

It can be verified that

$$\begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}^{-1} = \begin{pmatrix} 0 & 1 & 0 & 0 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 \\ 1 & 0 & 1 & 1 & 0 \end{pmatrix},$$

by checking that the product of the two matrices, computed modulo 2, is the identity matrix. This yields

$$\begin{aligned}(c_0,c_1,c_2,c_3,c_4) &= (0,1,0,0,0)\begin{pmatrix} 0&1&0&0&1 \\ 1&0&0&1&0 \\ 0&0&0&0&1 \\ 0&1&0&1&1 \\ 1&0&1&1&0 \end{pmatrix} \\ &= (1,0,0,1,0).\end{aligned}$$

Thus the recurrence used to generate the keystream is

$$z_{i+5} = (z_i + z_{i+3}) \bmod 2.$$

$\square$

1.3 Notes

Material on classical cryptography is covered in various textbooks and monographs, such as "Decrypted Secrets, Methods and Maxims of Cryptology" by Bauer [9]; "Cipher Systems, The Protection of Communications" by Beker and Piper [13]; "Cryptology" by Beutelspacher [32]; "Cryptography and Data Security" by Denning [109]; "Code Breaking, A History and Exploration" by Kippenhahn [192]; "Cryptography, A Primer" by Konheim [203]; and "Basic Methods of Cryptography" by van der Lubbe [222].

We have used the statistical data on frequency of English letters that is reported in Beker and Piper [13].

A good reference for elementary number theory is "Elementary Number Theory and its Applications" by Rosen [284]. Background in linear algebra can be found in "Elementary Linear Algebra" by Anton [4].

Two very enjoyable and readable books that provide interesting histories of cryptography are "The Codebreakers" by Kahn [185] and "The Code Book" by Singh [307].

Exercises

1.1 Evaluate the following:
   (a) $7503 \bmod 81$
   (b) $(-7503) \bmod 81$
   (c) $81 \bmod 7503$
   (d) $(-81) \bmod 7503$.

1.2 Suppose that $a, m > 0$, and $a \not\equiv 0 \pmod{m}$. Prove that

$$(-a) \bmod m = m - (a \bmod m).$$

1.3 Prove that $a \bmod m = b \bmod m$ if and only if $a \equiv b \pmod{m}$.

1.4 Prove that $a \bmod m = a - \lfloor \frac{a}{m} \rfloor m$, where $\lfloor x \rfloor = \max\{y \in \mathbb{Z} : y \leq x\}$.

1.5 Use exhaustive key search to decrypt the following ciphertext, which was encrypted using a *Shift Cipher*:

BEEAKFYDJXUQYHYJIQRYHTYJIQFBQDUYJIIKFUHCQD.

1.6 If an encryption function e_K is identical to the decryption function d_K, then the key K is said to be an *involutory key*. Find all the involutory keys in the *Shift Cipher* over $\mathbb{Z}_{26}$.

1.7 Determine the number of keys in an *Affine Cipher* over $\mathbb{Z}_m$ for $m = 30, 100$ and 1225.

1.8 List all the invertible elements in $\mathbb{Z}_m$ for $m = 28, 33$ and 35.

1.9 For $1 \leq a \leq 28$, determine $a^{-1} \bmod 29$ by trial and error.

1.10 Suppose that $K = (5, 21)$ is a key in an *Affine Cipher* over $\mathbb{Z}_{29}$.
   (a) Express the decryption function $d_K(y)$ in the form $d_K(y) = a'y + b'$, where $a', b' \in \mathbb{Z}_{29}$.

(b) Prove that $d_K(e_K(x)) = x$ for all $x \in \mathbb{Z}_{29}$.

1.11 (a) Suppose that $K = (a, b)$ is a key in an *Affine Cipher* over $\mathbb{Z}_n$. Prove that K is an involutory key if and only if $a^{-1} \bmod n = a$ and $b(a+1) \equiv 0 \pmod{n}$.

(b) Determine all the involutory keys in the *Affine Cipher* over $\mathbb{Z}_{15}$.

(c) Suppose that $n = pq$, where p and q are distinct odd primes. Prove that the number of involutory keys in the *Affine Cipher* over $\mathbb{Z}_n$ is $n + p + q + 1$.

1.12 (a) Let p be prime. Prove that the number of 2×2 matrices that are invertible over $\mathbb{Z}_p$ is $(p^2 - 1)(p^2 - p)$.

HINT Since p is prime, $\mathbb{Z}_p$ is a field. Use the fact that a matrix over a field is invertible if and only if its rows are linearly independent vectors (i.e., there does not exist a non-zero linear combination of the rows whose sum is the vector of all 0's).

(b) For p prime and $m \geq 2$ an integer, find a formula for the number of $m \times m$ matrices that are invertible over $\mathbb{Z}_p$.

1.13 For $n = 6, 9$ and 26, how many 2×2 matrices are there that are invertible over $\mathbb{Z}_n$?

1.14 (a) Prove that $\det A \equiv \pm 1 \pmod{26}$ if A is a matrix over $\mathbb{Z}_{26}$ such that $A = A^{-1}$.

(b) Use the formula given in Corollary 1.4 to determine the number of involutory keys in the *Hill Cipher* (over $\mathbb{Z}_{26}$) in the case $m = 2$.

1.15 Determine the inverses of the following matrices over $\mathbb{Z}_{26}$:

(a) $\begin{pmatrix} 2 & 5 \\ 9 & 5 \end{pmatrix}$

(b) $\begin{pmatrix} 1 & 11 & 12 \\ 4 & 23 & 2 \\ 17 & 15 & 9 \end{pmatrix}$

1.16 (a) Suppose that π is the following permutation of $\{1, \ldots, 8\}$:

x	1	2	3	4	5	6	7	8
$\pi(x)$	4	1	6	2	7	3	8	5

.

Compute the permutation π^{-1}.

(b) Decrypt the following ciphertext, for a *Permutation Cipher* with $m = 8$, which was encrypted using the key π:

TGEEMNELNNTDROEOAAHDOETCSHAEIRLM.

1.17 (a) Prove that a permutation π in the *Permutation Cipher* is an involutory key if and only if $\pi(i) = j$ implies $\pi(j) = i$, for all $i, j \in \{1, \ldots, m\}$.

(b) Determine the number of involutory keys in the *Permutation Cipher* for $m = 2, 3, 4, 5$ and 6.

1.18 Consider the following linear recurrence over $\mathbb{Z}_2$ of degree four:

$$z_{i+4} = (z_i + z_{i+1} + z_{i+2} + z_{i+3}) \bmod 2,$$

$i \geq 0$. For each of the 16 possible initialization vectors $(z_0, z_1, z_2, z_3) \in (\mathbb{Z}_2)^4$, determine the period of the resulting keystream.

1.19 Redo the preceding question, using the recurrence

$$z_{i+4} = (z_i + z_{i+3}) \bmod 2,$$

$i \geq 0$.

1.20 Suppose we construct a keystream in a synchronous stream cipher using the following method. Let $K \in \mathcal{K}$ be the key, let $\mathcal{L}$ be the keystream alphabet, and let Σ be a finite set of *states*. First, an *initial state* $\sigma_0 \in \Sigma$ is determined from K by some method. For all $i \geq 1$, the state σ_i is computed from the previous state σ_{i-1} according to the following rule:

$$\sigma_i = f(\sigma_{i-1}, K),$$

where $f : \Sigma \times \mathcal{K} \to \Sigma$. Also, for all $i \geq 1$, the keystream element z_i is computed using the following rule:

$$z_i = g(\sigma_i, K),$$

where $g : \Sigma \times \mathcal{K} \to \mathcal{L}$. Prove that any keystream produced by this method has period at most $|\Sigma|$.

1.21 Below are given four examples of ciphertext, one obtained from a *Substitution Cipher*, one from a *Vigenère Cipher*, one from an *Affine Cipher*, and one unspecified. In each case, the task is to determine the plaintext.

Give a clearly written description of the steps you followed to decrypt each ciphertext. This should include all statistical analysis and computations you performed.

The first two plaintexts were taken from "The Diary of Samuel Marchbanks," by Robertson Davies, Clarke Irwin, 1947; the fourth was taken from "Lake Wobegon Days," by Garrison Keillor, Viking Penguin, Inc., 1985.

(a) *Substitution Cipher*:

```
EMGLOSUDCGDNCUSWYSFHNSFCYKDPUMLWGYICOXYSIPJCK
QPKUGKMGOLICGINCGACKSNISACYKZSCKXECJCKSHYSXCG
OIDPKZCNKSHICGIWYGKKGKGOLDSILKGOIUSIGLEDSPWZU
GFZCCNDGYYSFUSZCNXEOJNCGYEOWEUPXEZGACGNFGLKNS
ACIGOIYCKXCJUCIUZCFZCCNDGYYSFEUEKUZCSOCFZCCNC
IACZEJNCSHFZEJZEGMXCYHCJUMGKUCY
```

HINT F decrypts to w.

(b) *Vigenère Cipher*:

```
KCCPKBGUFDPHQTYAVINRRTMVGRKDNBVFDETDGILTXRGUD
DKOTFMBPVGEGLTGCKQRACQCWDNAWCRXIZAKFTLEWRPTYC
QKYVXCHKFTPONCQQRHJVAJUWETMCMSPKQDYHJVDAHCTRL
SVSKCGCZQQDZXGSFRLSWCWSJTBHAFSIASPRJAHKJRJUMV
GKMITZHFPDISPZLVLGWTFPLKKEBDPGCEBSHCTJRWXBAFS
PEZQNRWXCVYCGAONWDDKACKAWBBIKFTIOVKCGGHJVLNHI
FFSQESVYCLACNVRWBBIREPBBVFEXOSCDYGZWPFDTKFQIY
CWHJVLNHIQIBTKHJVNPIST
```

(c) *Affine Cipher*:

```
KQEREJEBCPPCJCRKIEACUZBKRVPKRBCIBQCARBJCVFCUP
KRIOFKPACUZQEPBKRXPEIIEABDKPBCPFCDCCAFIEABDKP
BCPFEQPKAZBKRHAIBKAPCCIBURCCDKDCCJCIDFUIXPAFF
ERBICZDFKABICBBENEFCUPJCVKABPCYDCCDPKBCOCPERK
IVKSCPICBRKIJPKABI
```

(d) unspecified cipher:

```
BNVSNSIHQCEELSSKKYERIFJKXUMBGYKAMQLJTYAVFBKVT
DVBPVVRJYYLAOKYMPQSCGDLFSRLLPROYGESEBUUALRWXM
MASAZLGLEDFJBZAVVPXWICGJXASCBYEHOSNMULKCEAHTQ
OKMFLEBKFXLRRFDTZXCIWBJSICBGAWDVYDHAVFJXZIBKC
GJIWEAHTTOEWTUHKRQVVRGZBXYIREMMASCSPBNLHJMBLR
FFJELHWEYLWISTFVVYFJCMHYUYRUFSFMGESIGRLWALSWM
NUHSIMYYITCCQPZSICEHBCCMZFEGVJYOCDEMMPGHVAAUM
ELCMOEHVLTIPSUYILVGFLMVWDVYDBTHFRAYISYSGKVSUU
HYHGGCKTMBLRX
```

1.22 (a) Suppose that $p_1, \ldots, p_n$ and $q_1, \ldots, q_n$ are both probability distributions, and $p_1 \geq \cdots \geq p_n$. Let $q'_1, \ldots, q'_n$ be any permutation of $q_1, \ldots, q_n$. Prove that the quantity

$$\sum_{i=1}^{n} p_i q'_i$$

is maximized when $q'_1 \geq \cdots \geq q'_n$.

(b) Explain why the expression in Equation (1.1) is likely to be maximized when $g = k_i$.

1.23 Suppose we are told that the plaintext

`breathtaking`

yields the ciphertext

RUPOTENTOIFV

where the *Hill Cipher* is used (but m is not specified). Determine the encryption matrix.

1.24 An *Affine-Hill Cipher* is the following modification of a *Hill Cipher*: Let m be a positive integer, and define $\mathcal{P} = \mathcal{C} = (\mathbb{Z}_{26})^m$. In this cryptosystem, a key K consists of a pair (L, b), where L is an $m \times m$ invertible matrix over $\mathbb{Z}_{26}$, and $b \in (\mathbb{Z}_{26})^m$. For $x = (x_1, \ldots, x_m) \in \mathcal{P}$ and $K = (L, b) \in \mathcal{K}$, we compute $y = e_K(x) = (y_1, \ldots, y_m)$ by means of the formula $y = xL + b$. Hence, if $L = (\ell_{i,j})$ and $b = (b_1, \ldots, b_m)$, then

$$(y_1, \ldots, y_m) = (x_1, \ldots, x_m) \begin{pmatrix} \ell_{1,1} & \ell_{1,2} & \ldots & \ell_{1,m} \\ \ell_{2,1} & \ell_{2,2} & \ldots & \ell_{2,m} \\ \vdots & \vdots & & \vdots \\ \ell_{m,1} & \ell_{m,2} & \ldots & \ell_{m,m} \end{pmatrix} + (b_1, \ldots, b_m).$$

Suppose Oscar has learned that the plaintext

`adisplayedequation`

is encrypted to give the ciphertext

DSRMSIOPLXLJBZULLM

and Oscar also knows that $m = 3$. Determine the key, showing all computations.

1.25 Here is how we might cryptanalyze the *Hill Cipher* using a ciphertext-only attack. Suppose that we know that $m = 2$. Break the ciphertext into blocks of length two letters (digrams). Each such digram is the encryption of a plaintext digram using the unknown encryption matrix. Pick out the most frequent ciphertext digram and assume it is the encryption of a common digram in the list following Table 1.1 (for example, TH or ST). For each such guess, proceed as in the known-plaintext attack, until the correct encryption matrix is found.

Here is a sample of ciphertext for you to decrypt using this method:

```
LMQETXYEAGTXCTUIEWNCTXLZEWUAISPZYVAPEWLMGQWYA
XFTCJMSQCADAGTXLMDXNXSNPJQSYVAPRIQSMHNOCVAXFV
```

1.26 We describe a special case of a *Permutation Cipher*. Let m, n be positive integers. Write out the plaintext, by rows, in $m \times n$ rectangles. Then form the ciphertext by taking the columns of these rectangles. For example, if $m = 3$, $n = 4$, then we would encrypt the plaintext "*cryptography*" by forming the following rectangle:

```
cryp
togr
aphy
```

The ciphertext would be "*CTAROPYGHPRY*."

(a) Describe how Bob would decrypt a ciphertext string (given values for m and n).

(b) Decrypt the following ciphertext, which was obtained by using this method of encryption:

```
MYAMRARUYIQTENCTORAHROYWDSOYEOUARRGDERNOGW
```

1.27 The purpose of this exercise is to prove the statement made in Section 1.2.5 that the $m \times m$ coefficient matrix is invertible. This is equivalent to saying that the rows of this matrix are linearly independent vectors over $\mathbb{Z}_2$.

Suppose that the recurrence has the form

$$z_{m+i} = \sum_{j=0}^{m-1} c_j z_{i+j} \bmod 2,$$

where $(z_1, \dots, z_m)$ comprises the initialization vector. For $i \geq 1$, define

$$v_i = (z_i, \dots, z_{i+m-1}).$$

Note that the coefficient matrix has the vectors $v_1, \dots, v_m$ as its rows, so our objective is to prove that these m vectors are linearly independent.

Prove the following assertions:

(a) For any $i \geq 1$,

$$v_{m+i} = \sum_{j=0}^{m-1} c_j v_{i+j} \bmod 2.$$

(b) Choose h to be the minimum integer such that there exists a non-trivial linear combination of the vectors $v_1, \dots, v_h$ which sums to the vector $(0, \dots, 0)$ modulo 2. Then

$$v_h = \sum_{j=0}^{h-2} \alpha_j v_{j+1} \bmod 2,$$

and not all the α_j's are zero. Observe that $h \leq m + 1$, since any $m + 1$ vectors in an m-dimensional vector space are dependent.

(c) Prove that the keystream must satisfy the recurrence

$$z_{h-1+i} = \sum_{j=0}^{h-2} \alpha_j z_{j+i} \bmod 2$$

for any $i \geq 1$.

(d) If $h \leq m$, then the keystream satisfies a linear recurrence of degree less than m. Show that this is impossible, by considering the initialization vector $(0, \ldots, 0, 1)$. Hence, conclude that $h = m + 1$, and therefore the matrix must be invertible.

1.28 Decrypt the following ciphertext, obtained from the *Autokey Cipher*, by using exhaustive key search:

```
MALVVMAFBHBUQPTSOXALTGVWWRG
```

1.29 We describe a stream cipher that is a modification of the *Vigenère Cipher*. Given a keyword $(K_1, \ldots, K_m)$ of length m, construct a keystream by the rule $z_i = K_i$ $(1 \leq i \leq m)$, $z_{i+m} = (z_i + 1) \bmod 26$ $(i \geq 1)$. In other words, each time we use the keyword, we replace each letter by its successor modulo 26. For example, if $SUMMER$ is the keyword, we use $SUMMER$ to encrypt the first six letters, we use $TVNNFS$ for the next six letters, and so on.

(a) Describe how you can use the concept of index of coincidence to first determine the length of the keyword, and then actually find the keyword.

(b) Test your method by cryptanalyzing the following ciphertext:

```
IYMYSILONRFNCQXQJEDSHBUIBCJUZBOLFQYSCHATPEQGQ
JEJNGNXZWHHGWFSUKULJQACZKKJOAAHGKEMTAFGMKVRDO
PXNEHEKZNKFSKIFRQVHHOVXINPHMRTJPYWQGJWPUUVKFP
OAWPMRKKQZWLQDYAZDRMLPBJKJOBWIWPSEPVVQMBCRYVC
RUZAAOUMBCHDAGDIEMSZFZHALIGKEMJJFPCIWKRMLMPIN
AYOFIREAOLDTHITDVRMSE
```

The plaintext was taken from "The Codebreakers," by D. Kahn, Scribner, 1996.

1.30 We describe another stream cipher, which incorporates one of the ideas from the "Enigma" system used by Germany in World War II. Suppose that π is a fixed permutation of $\mathbb{Z}_{26}$. The key is an element $K \in \mathbb{Z}_{26}$. For all integers $i \geq 1$, the keystream element $z_i \in \mathbb{Z}_{26}$ is defined according to the rule $z_i = (K + i - 1) \bmod 26$. Encryption and decryption are performed using the permutations π and π^{-1}, respectively, as follows:

$$e_z(x) = \pi(x) + z \bmod 26$$

and

$$d_z(y) = \pi^{-1}(y - z \bmod 26),$$

where $z \in \mathbb{Z}_{26}$.

Suppose that π is the following permutation of $\mathbb{Z}_{26}$:

x	0	1	2	3	4	5	6	7	8	9	10	11	12
$\pi(x)$	23	13	24	0	7	15	14	6	25	16	22	1	19

x	13	14	15	16	17	18	19	20	21	22	23	24	25
$\pi(x)$	18	5	11	17	2	21	12	20	4	10	9	3	8

The following ciphertext has been encrypted using this stream cipher; use exhaustive key search to decrypt it:

```
WRTCNRLDSAFARWKXFTXCZRNHNYPDTZUUKMPLUSOXNEUDO
KLXRMCBKGRCCURR
```