Chapter 2

Introduction to classical cryptography

2.1 Introduction

Already more than 2000 years ago people had realized the need to protect written messages against falling into the wrong hands. Powerful people of this time distributed messages by sending couriers who carried the messages. There was of course a great uncertainty whether the courier would actually reach its destination and deliver the message or would get caught by some enemy. For this type of communication one developed secret writing, i.e., writing the message in such a way that only the true receiver can understand the meaning.

Throughout history, secret writing became an established problem and the area was named cryptology. Up to very recently, cryptology was primarily concerned with military and diplomatic applications and distinguished between two disciplines, the discipline of *cryptography* which deals with development of systems for secret writing, and the discipline of *cryptanalysis* which analyze existing systems in order to break them. It is for example a well known fact that the effords in breaking some of the Japanese and German ciphers during World War II were major contributions to the victory for the allied.

Some years after the World War II a strong technical development started and it build what many people nowadays refer to as the ”information society“. The storage and transmission of information became cheap and simple. Huge amounts of information were suddenly transmitted in such a way that almost anyone could take part of the content. This increased the research in cryptology and it was apparent that it was no longer the small group of military people and diplomats but more or less any company who needed cryptographic equipment to protect their information.

The increasing efforts of open research in this area during the last 25 years has resulted in a well established theory on cryptography.

2.2 Cryptographic primitives

Nowadays, the word “cryptography” is used to name the research area that has arised from the ancient secret writing. Secret writing is only one out of many different problems

that has found a solution within cryptography. From the introduction of public-key cryptography, many seemingly impossible problems could suddenly be solved.

The solutions to different cryptographic problems are refered to as *cryptographic primitives*. For example, the primitive "symmetric encryption scheme" refers to a cipher where the sender and the receiver share a common secret key, which enables them to communicate some message in such a way that an enemy cannot get any information about the message even though he observes the communication, provided that he is not in posession of the secret key.

The primitive "symmetric encryption scheme" can then be divided into "stream ciphers" and "block ciphers", which are two different types of symmetric ciphers. Other types of general primitives are for example: asymmetric encryption schemes, digital signatures, secret sharing scheme, etc.

Cryptographic primitives are never isolated occurrences, but are implemented in a larger system. The cryptographic primitives are used to provide security in the system. The area of designing secure systems is called *Information security* (or Data security).

The role of cryptographic primitives in information security can be viewed as the role tools and building material for a carpenter. In information security, we use different cryptographic primitives to build a whole system that provides security. Each primitive provides a certain service in the system. It is worth pointing out that failure in the security of a system is almost always due to a badly designed system or a erronous use of a primitive, and almost never due to a bad primitive. For system designers, it is of outmost importance to learn how different cryptographic primitives work. Only through such knowledge can erronous use of primitives be avoided.

2.3 A first model of a cryptosystem

In order to describe a cryptosystem, we need som kind of model of our communication system. The traditional and most straight-forward way to do this is to use the model of a symmetric cryptosystem. The mathematical foundation of the underlying theory of this model is due to Shannon, and the model is sometimes refered to as the Shannon model of secrecy.

Let $\mathcal{P}$ be a finite set which is called the *alphabet*. With $|\mathcal{P}|$ we denote the cardinality of the alphabet $\mathcal{P}$. In this section we often use the english letters as our alphabet, and to allow arithmetics we number them as $\mathtt{a} = 0$, $\mathtt{b} = 1$, $\ldots$, $\mathtt{z} = 26$. This gives $\mathcal{P} = \mathbb{Z}_{26}$, and $|\mathcal{P}| = 26$.

It is a cryptographic convention not to use anonomous words like sender, receiver, but to use the names Alice, Bob, Caesar, and Eve (Eve is considered to be the enemy).

In the Shannon model, as given in Figure 2.1, Alice has a *source* producing symbols from the alphabet $\mathcal{P}$. The *plaintext* $\mathbf{m}$ consists of the produced symbols in vector form, i.e.,

$$\mathbf{m} = (m_1, m_2, \ldots, m_n),$$

where $m_i \in \mathcal{P}$. The number of plaintext symbols n is the length of the plaintext.

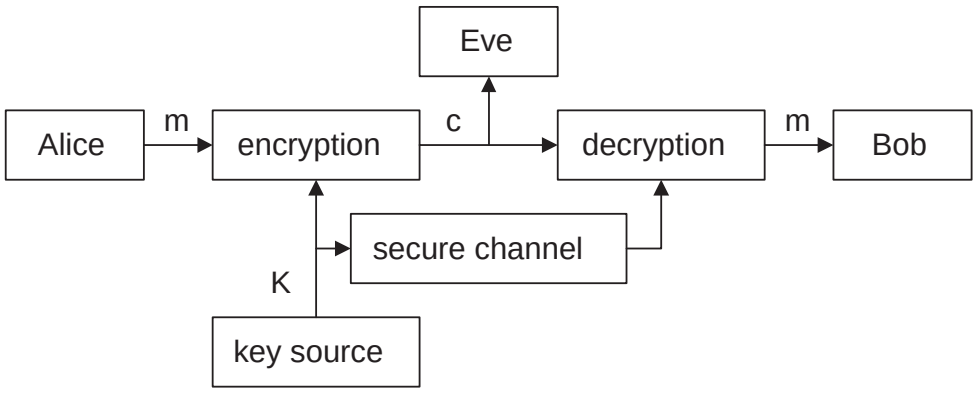

Figure 2.1: The Shannon model for symmetric encryption

Alice applies the *encryption function* $E_k()$ to the plaintext, obtaining the ciphertext $\mathbf{c}$ as $\mathbf{c} = E_k(\mathbf{m})$. The encryption function is indexed by a secret key k, shared between Alice and Bob. The key is taken from a set $\mathcal{K}$ of possible keys. Each key describes a certain function $E_k : \mathcal{P}^n \to \mathcal{C}^{n'}$, where $\mathcal{C}$ is called the ciphertext alphabet. Usually, we consider the case $n = n'$.

The set of keys can equivalently be seen as a set of $|\mathcal{K}|$ different functions mapping $\mathcal{P}^n \to \mathcal{C}^n$. With the knowledge of the secret key k, Bob is supposed to be able to recover the plaintext. This means that each function $E_k()$ must be **invertible** (*injective*), and that the set of functions cannot be selected arbitrarily. To his disposal, Bob has a *decryption function* $D_k()$ that decrypts encrypted messages back to its original form. In a more formal language,

$$D_k(E_k(\mathbf{m})) = \mathbf{m}, \quad \forall \mathbf{m} \in \mathcal{P}^n. \tag{2.1}$$

Usually, a cryptosystem will be used extensively, by many people and for a long time. Hence, it is natural to adopt assumptions regarding the knowledge of various details concerning the crypto system. Such assumptions were formulated already in the 19th century by A. Kerckhoff as follows. The enemy is always supposed to have access to all information regarding a cryptosystem except the actual choice of secret key k. In particular, this means that the set of transformations $\{E_k()|k \in \mathcal{K}\}$ is known to the enemy, as well as the probability distribution for the selected key and the plaintext. This is generally known as *Kerckhoff's principle*.

This assumption is of outmost importance, because without it we can not provide analysis of the system and then we cannot make claims regarding security. The assumption also follows very well the current situation for cryptosystems. Most cryptographic primitives that are used are formulated as standards and are available in full description to everyone.

Finally, we must also consider the different kind of attacks that can be applied. In the treatment of classical ciphers we mainly consider *ciphertext only attacks*. A ciphertext only attack is an attack in which the enemy has access to a ciphertext $\mathbf{c}$ and tries to recover either the plaintext $\mathbf{m}$ or the secret key k. This is the weakest form of attack (from the enemy's point of view). A modern cipher must be able to protect against other and stronger attacks as well. We will come back to such attacks when we discuss symmetric encryption schemes.

2.4 A first cryptosystem - The Caesar cipher

One of the simplest and also one of the oldest cryptosystems is due to Julius Caesar. He used a transformation on the plaintext that cyclically shifted each letter of the plaintext 3 steps. The letter `a` became the letter `d`, `b` became `e`, etc., in the ciphertext. A natural generalization is to shift not three but k positions, where $k \in \{0, 1, \ldots, 25\}$ is the secret key. This cryptosystem is usually called the *Caesar cipher*. The encryption function $E_k()$ now operates on individual characters and is given by

$$E_k(m) = m + k \pmod{26}^1.$$

Recall that we assumed `a` $= 0$, `b` $= 1$, ... We can check the fact that $E_k()$ is injective, i.e., there do not exist two plaintext letters mapping to the same ciphertext letter. The decryption function is given by

$$D_k(m) = m - k.$$

We can also verify that $D_k(E_k(m)) = m, \forall m \in \mathcal{P}$.

Example 2.1. *The plaintext* `thisisamessagetoyou` *would under the key* $k = 3$ *be mapped to the ciphertext* `wklvlvdphvvdjhwrbrx`.

Cryptanalysis of the Caesar cipher

Since the number of keys in the Caesar cipher is limited to 26, the easiest way to perform a ciphertext-only attack is through *exhaustive key search*. We simply try all possible keys, decrypt the ciphertext using the selected key and check if we get something meaningful. The following example illustrates this.

Example 2.2. *Assume that we have received the ciphertext* `wklvlvdphvvdjhwrbrx`. *We try all keys* $k \in \mathbb{Z}_{26}$, *and decrypt using the decryption function* $D_k()$. *Then we get the following table of posssible plaintexts.*

[1] The abuse of notation is just a reminder to the reader that arithmetics take place in $\mathbb{Z}_{26}$. Continuing, this will not be explicitly written

k	*plaintext*
0	wklvlvdphvvdjhwrbrx
1	vjkukucoguucigvqaqw
2	uijtjtbnfttbhfupzpv
3	thisisamessagetoyou
4	sghrhrzldrrzfdsnxnt
5	rfgqgqykcqqyecrmwms
6	qefpfpxjbppxdbqlvlr
7	pdeoeowiaoowcapkukq
8	ocdndnvhznnvbzojtjp
9	nbcmcmugymmuaynisio
10	mablbltfxlltzxmhrhn
11	lzakaksewkksywlgqgm
12	kyzjzjrdvjjrxvkfpfl
13	jxyiyiqcuiiqwujeoek
14	iwxhxhpbthhpvtidndj
15	hvwgwgoasggoushcmci
16	guvfvfnzrffntrgblbh
17	ftueuemyqeemsqfakag
18	estdtdlxpddlrpezjzf
19	drscsckwocckqodyiye
20	cqrbrbjvnbbjpncxhxd
21	bpqaqaiumaaiombwgwc
22	aopzpzhtlzzhnlavfvb
23	znoyoygskyygmkzueua
24	ymnxnxfrjxxfljytdtz
25	xlmwmweqiwwekixscsy

2.5 The simple substitution cipher

The simple (or mono-alphabetic) substitution cipher is a generalization of the Caesar cipher. The encryption function $E_k()$ operates on individual characters, but now we consider an arbitrary permutation of the alphabet as the key.

Consider the set of all permutations on $\mathbb{Z}_{26}$, written as

$$\{\pi_1, \pi_2, \ldots, \pi_{|\mathcal{K}|}\}.$$

The number of different permutations is $|\mathcal{K}| = 26!$. The encryption function is defined by

$$E_K(m) = \pi_k(m). \tag{2.2}$$

Since π_k is a permutation it is invertible. The decryption function is then given by $D_k(c) = \pi_k^{-1}(c)$.

Example 2.3. *Let π_k be the permutation described by*

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
h	y	n	u	j	m	i	k	o	l	p	t	g	b	f	r	v	c	d	e	s	x	w	a	z	q

a	0.0804	j	0.0016	s	0.0654
b	0.0154	k	0.0067	t	0.0925
c	0.0306	l	0.0414	u	0.0271
d	0.0399	m	0.0253	v	0.0099
e	0.1251	n	0.0709	w	0.0192
f	0.0230	o	0.0760	x	0.0019
g	0.0196	p	0.0200	y	0.0173
h	0.0549	q	0.0011	z	0.0009
i	0.0726	r	0.0612		

Table 2.1: The probability distribution of single letters in English text

The plaintext `thisisamessagetoyou` *would under this key be mapped to the ciphertext* `ekododhgjddhijefzfs`.

Example 2.4. *Assume that the ciphertext* `djbuekjecfsrdefboike` *has been encrypted by a simple substitution cipher using the permutation* π_k *from the previous example. The decryption function is then given by* $D_k(c) = \pi_k^{-1}(c)$*, which is described as follows,*

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
x	n	r	s	t	o	m	a	g	e	h	j	f	c	i	k	z	p	u	l	d	q	w	v	b	y

The ciphertext `djbuekjecfsrdefboike` *would under this key be mapped to the ciphertext* `sendthetroupstonight`.

Cryptanalysis of the simple substitution cipher

We first note that the number of possible keys, i.e., the number of different permutations on $\mathbb{Z}_{26}$ is $26! > 4 \cdot 10^{26}$. This is a much too large number of keys to search by exhaustive key search, even if we had access to all existing computing power.

Nevertheless, it is a trivial task to force such a cipher. We expoit the statistical nature of the plaintext source. Let us assume that the plaintext source is written English. There is a lot that can be said about the statistical nature of such a language, but we just mention a few basic facts that will enable us to do cryptanalysis.

Let us consider different letter combinations, called *r-grams*, $(m_i, m_{i+1}, \ldots, m_{i+r})$ from the sequence of letters $m_1, m_2, \ldots$ produced by the source. For single letters, 1-grams, the distribution is given in Table 2.1. Note that in the simple substitution cipher, the chosen permutation does not change the (set of) values in the probability distribution, it just permutes them. So the ciphertext will have the same distribution as the plaintext, the difference is that the letters have been permuted.

To force the cipher, the first approach that comes to mind is to simply count the number of occurrences of the different letters in the ciphertext. Clearly, the most common letter in the ciphertext `c` is likely to correspond to the letter `e` in the plaintext, if the ciphertext is long enough. The second most common letter in `c` is likely to correspond to `t`, etc. If we can get a few letters correct (possibly after trying some different possibilities), we might be able to correctly guess the remaining letters.

th	0.0270	on	0.0154	ed	0.0111
he	0.0257	an	0.0152	te	0.0109
in	0.0194	en	0.0129	ti	0.0108
er	0.0180	at	0.0127	or	0.0108
re	0.0160	es	0.0115	st	0.0103
the	0.0215	for	0.0036	ere	0.0027
and	0.0060	tha	0.0032	con	0.0026
tio	0.0048	ter	0.0029	ted	0.0023
ati	0.0036	res	0.0027		

Table 2.2: The most common 2-grams and 3-grams and the corresponding probabilities.

Our method can get stronger if we instead of 1-grams make use of 2-grams or 3-grams. Our source is not *memoryless*. In a memoryless source, the probability distribution for the letter m_{i+1} is independent of the letter m_i. This is not at all the case for the English language (or any other language). Instead, we have a large dependence between consecutive letters in a produced sequence. For example, if the source has generated the two letters `wa`, the next letter can be for example `f`, `g`, `i`, etc., but the otherwise so common letter `e` can not occur.

The most common 2-grams and 3-grams and their corresponding probabilities are given in Table 2.2.

The way to proceed in cryptanalysis using 2-grams and 3-grams should now be clear. If we have a long enough ciphertext, we determine the most frequent 2-grams and 3-grams. The most frequent 3-gram is likely to correspond to the plaintext letters `the`. We make this guess, and based on that we try to determine more about the permutation defining the encryption function. The procedure is illustrated in the following example.

Example 2.5. *Assume that we have received the following ciphertext:*

```
xsftbiwmqooxwdxssfmtxmaibcsfiisctfzsfmibcixiiczc
awbxosbxamqiwqsaxjfqscixssaxnxiibxijcznlqlcmixnq
emtfmiczmxiufinbqvwfixsoxwlxrfmtxmflvqzixmiafwuq
jcznibclnwiczfqewvxifcmifmzqqlioqqmcibzccbxadfmx
ssnxoxrcmcawbclqxmcawqdisnuqesafigcibxiwbcoxwibc
wfwiczqdibcgqnfmrxmwxwobqsqjcaibctfzsofibibcixii
czcawbxosobqoxwibcaxetbiczqdibclxfaobqbxacwuxvca
         dzqlibcvfzxicwibcfmiczmdzqomca
```

Counting the number of occurences of different letters, the most frequent ones are `i` *46 times,* `c` *40 times,* `x` *37 times, etc. We might think that these letters will correspond to some of the most frequent plaintext letters* `e`*,* `t`*,* `a`*,* `o`*,* `i`*,* `n`*. However, the frequencies of the letters are quite close to each other so we cannot be sure that* `i` *corresponds to the most frequent plaintext letter* `e`*.*

Instead we do a frequency count on 3-grams, which is much more powerful. The most frequent 3-grams in the ciphertext are `ibc` *occuring 11 times and* `icz` *occuring 7 times.*

We now adopt the hypothesis that `ibc` *corresponds to the most common 3-gram being* `the`*. This gives us* $\pi($`t`$) =$ `i`*,* $\pi($`h`$) =$ `b`*, and* $\pi($`e`$) =$ `c`*. This means that* `ibc` *corresponds to a plaintext 3-gram of the form* `te*` *(the* `*` *means an unknown character). Looking at the most common 3-grams we come to the conclusion that* $\pi($`r`$) =$ `z`*.*

If we use the partial information we have and start decrypting the ciphertext we get `a***ht*****a**a*****a**the**tt*e**r***thetattere**ha**ha***t****`*... The unknown last letter in the sequence* `*thetattere*` *is a* `d` *(tattered). This gives* $\pi($`d`$) =$ `a`*. This gives us some additional information and if we continue to do a "partial" decryption we find later in the plaintext the sequence* `**a*theda**hter**the*`*... . We guess that the word* `daughter` *is present, giving* $\pi($`o`$) =$ `q`*,* $\pi($`f`$) =$ `d`*.*

We leave it to the reader to complete the analysis.

2.6 Polyalphabetic ciphers

The ciphers we have seen sofar have all operated on single letters of the plaintext, and are named mono-alphabetic ciphers. *Polyalphabetic ciphers*, on the contrary, operate on various portions of the plaintext. In the simplest case, a number of mono-alphabetic ciphers with different keys are used sequentially and then cyclically repeated. This makes statistical attacks harder, but not at all impossible.

The most well known polyalphabetic cipher is the *Vigenére cipher*. It cyclically uses t Caesar ciphers, where t is called the *period* of the cipher. The encryption function maps the plaintext $\mathbf{m} = m_1, m_2, \ldots$ to the ciphertext $\mathbf{c} = c_1, c_2, \ldots$, through

$$\mathbf{c} = E_{\mathbf{k}}(m_1, m_2, \ldots, m_t), E_{\mathbf{k}}(m_{t+1}, m_{t+2}, \ldots, m_{t+t}), \ldots,$$

where

$$E_{\mathbf{k}}(m_1, m_2, \ldots, m_t) = (m_1 + k_1, m_2 + k_2, \ldots, m_t + k_t), \tag{2.3}$$

and the key $\mathbf{k}$ consists of t characters $\mathbf{k} = (k_1, k_2, \ldots, k_t)$. In the encryption, the first plaintext letter is encrypted with the first Caesar cipher, the second plaintext letter with the second Caesar cipher, etc. After having used t Caesar ciphers, the $(t+1)$st plaintext letter is again encrypted with the first Caesar cipher, and so on. The key $\mathbf{k}$ is often chosen as a word (*keyword*).

Here is an example demonstrating the Vigenére cipher.

Example 2.6. *Assume that the plaintext* `youmustvisitmetonight` *is to be encrypted using a Vigenére cipher with period* 4*, where the key has been chosen to be* $\mathbf{k} =$ `lucy`*. Then the encryption can be illustrated as follows:*

y	o	u	m	u	s	t	v	i	s	i	t	m	e	t	o	...
+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	+	
l	u	c	y	l	u	c	y	l	u	c	y	l	u	c	y	...
j	i	w	k	f	m	v	t	t	m	k	r	x	y	v	m	...

The resulting ciphertext is `jiwkfmvttmkrxyvm`*....*

Cryptanalysis of the Vigenére cipher

The methods discussed here actually apply to any polyalphabetic substitution cipher. Cryptanalysis of a polyalphabetic substitution cipher is split into two different problems, the first problem is to determine the period of the cipher, the second to reconstruct the t different substitution alphabets that have been used.

The first method for determining the period that is discussed is called *Kasiski's method*. It is based on the following observation: repeated portions of plaintext encrypted with the same portion of the keyword results in identical ciphertext segments. Consequenctly, one expects the number of characters between the beginning of repeated ciphertext segments to be a multiple of the keyword length. Ideally, it then suffices to compute the greatest common factor of all such distances between identified repeated segments. However, coincidental repeated ciphertexts may also occur. So, an analysis of common factors among all distances should be done, and the largest factor that occurs most commonly is likely to be the period.

Example 2.7. *Assume that we have the following ciphertext*

vyckbygecgxukgftkmuzlvtjgcyibngcwhagtkntkaqughbfuvajkwdlrqrm
uzlvtjcebdfcgprtqlqirwxhuvsshjcebdfcgprtqlqwntkaqujtgyvyegxs
vvpiaspkftfwujyvxshrggequvajkwupqixedvadfwurtpbdcsjtmzgkfgxw
nflvkwrvyixvuvojxfevqxgljzqekgdccbtjgkwebuccmumzgncpdfgjqdyl
jvndeqccnwttgkgrthrimpvzyzrwtkorjadwanmgwp

We have underlined long sequences that repeat in the ciphertext. The distance between the two occurences of muzlvtj *is 42. The other two distances are 96 and 24 respectively. The greatest common divisor between these three numbers is 6. The assumed period is then set to 6.*

After the period t has been determined, we know that the letters $m_i, m_{i+t}, m_{i+2t}, \ldots$ all have been encrypted with the same substitution cipher (or Caesar cipher in the case of a Vigenére cipher). With the period given, we can split the ciphertext characters into t different sets, each set containing the letters encrypted by a mono-alphabetic substitution cipher. The key of each mono-alphabetic substitution cipher can finally be determined through the statistics of 1-grams together with some trial and error.

Example 2.8. *(cont') The text from the previous example can be split into six different multisets, each corresponding to the letters encrypted by the same Caesar cipher. Taking every* 6^{th} *letter gives multisets*

$$S_1 = \{v, g, k, u, g, g, t, \ldots\},$$

$$S_2 = \{y, e, g, z, c, c, k, \ldots\},$$

$$\vdots$$

$$S_6 = \{y, u, m, j, n, g, a, \ldots\},$$

To find the key used in the first Caesar cipher, exhaustive key search does not provide a direct solution since we only get to know every 6th letter when guessing a key. This makes it impossible to identify the correctly guessed key. Instead we match a frequency count of the multiset S_1 with the known probability distribution of single letters in English.

Counting the different letters in S_1 we get

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
1	0	5	3	1	0	10	1	0	2	1	0	0	1	1	1	4	2	0	1	9	4	0	0	1	0

By cyclically shifting this frequency count and looking for the best match compared to the known distribution we find it at $k_1 = 2$ (or $k_1 =$ `c`).

Doing the same procedure for all six multisets we finally reach the key $\mathbf{k} =$ `crypts` *and the ciphertext can be decrypted as* `the vigenere cipher using a relatively short period is insecure but by using ...`.

The last part of the example, matching a received frequency count to different distributions (keys) can of course be done in a much more formal way, but we skip that for now. Instead, we examine a more systematic approach for determining the period of a polyalphabetic substitution cipher, called *index of coincidence*, IC. The index of coincidence is a measure of the relative frequency of letters in a ciphertext sample, which allow us to determine the period t.

Let the ciphertext alphabet be $\mathbb{Z}_{26}$, and let p_i be the unknown probability of the character i in the ciphertext, $i =$`a,b,...,z`. We start by introducing the *measure of roughness*, MR, which measures the deviation of the distribution of ciphertext characters from a flat frequency distribution as follows:

$$MR = \sum_{i=\mathtt{a}}^{\mathtt{z}} (p_i - \frac{1}{26})^2 = \sum_{i=\mathtt{a}}^{\mathtt{z}} p_i^2 - \frac{1}{26}. \tag{2.4}$$

Clearly, the minimum value of MR is $MR_{\min} = 0$, corresponding to a flat distribution. The maximum value will occur when $\sum_{i=\mathtt{a}}^{\mathtt{z}} p_i^2$ is maximized, which in our case corresponds to a mono-alphabetic cipher. The maximum value can be calculated by calculating (2.4) over the plaintext distribution to be

$$MR_{\max} = 0.0667 - 0.0385 = 0.0282, \tag{2.5}$$

(these numbers may slightly vary depending on the source of statistics for English). Of cause, MR will vary with the period t, reaching $MR_{\max}$ for $t = 1$ and MR_{min} for $t = \infty$. The MR cannot be computed directly, because the distribution (and t) is unknown, but it can be estimated through the frequency of letters in the ciphertext as follows.

Let f_i denote the number of appearances of letter i, $i =$`a,b,..., z`, in a ciphertext of length n. The index of coincidence is defined as the probability that two arbitrary chosen character from the *given* ciphertext are the same, i.e.,

$$IC = \frac{\sum_{i=\mathtt{a}}^{\mathtt{z}} f_i(f_i - 1)}{n(n-1)}. \tag{2.6}$$

In general, the probability that two randomly chosen ciphertext characters are the same equals $\sum_{i=\mathtt{a}}^{\mathtt{z}} p_i^2$. Thus IC is an *estimate* of $\sum_{i=\mathtt{a}}^{\mathtt{z}} p_i^2$, and from (2.4) it will also provide an estimate of $MR + 1/26$. We can express this as $E(IC) = MR + 1/26$, where $E(IC)$ is the expectation of (2.6). By calculating IC for the given ciphertext we then get a rough estimate of the period t, as given in the table.

t	1	2	3	4	5	7	∞
E(IC)	0.066	0.052	0.047	0.044	0.042	0.041	0.038

Example 2.9. *Calculating IC for the ciphertext in the previous example gives $IC = 0.042$, indicating a period of 5. As we know that the period is 6, we come to the conclusion that the method of calculating IC does not always give a correct period.*

Unfortunately, the obtained estimate of the period t is usually too rough. The best way of determining the period t is instead by *ciphertext autocorrelation*. Given a ciphertext with unknown period, we examine the number of coincidences when the ciphertext is auto-correlated. This means that with a given ciphertext $\mathbf{c} = c_1, c_2, \ldots, c_n$, we count the number of occurrences $c_i = c_{i+t^*}$ in the interval $i = 1, \ldots, n - t^*$, for different values of t^*. The lowest value of t^* with number of occurrences around $0.066n$ is with high probability the period t.

The argument explaining this is the following. When t^* is a multiple of the period t, the ciphertext letters c_i and c_{i+t^*} are both from the same mono-alphabetic substitution cipher, and the probability that $c_i = c_{i+t^*}$ is then around 0.066. If t^* is not a multiple of the period t, the ciphertext letters c_i and c_{i+t^*} are from different mono-alphabetic substitution ciphers. Then the probability that $c_i = c_{i+t^*}$ is much smaller.

Finally, the method of ciphertext autocorrelation is improved if we divide the letters in t^* multisets by selecting every $t^{*\text{th}}$ letter and consider all possible pairs of letters within each multiset.

Example 2.10. *Consider again the ciphertext in previous examples. Pick the first and then every t^*thth letter and form multiset S_1. Pick the second and then every t^{*th} letter and form multiset S_2, and so on until we have formed t^* multisets. Calculate IC for multisets $S_1, S_2, \ldots, S_{t^*}$. The t^* value giving IC values around 0.066 is a likely period (or multiple of).*

2.7 Other important classical ciphers

The Vernam cipher: (or the one-time-pad) Consider a ciphertext of fixed length n encrypted by a Vigenére cipher, where the period of the key is chosen to be $t = n$. It can then be checked that all methods of crypanalysis based on statistics from the plaintext now fails.

In fact, in the next chapter we prove that the Vernam cipher is totally secure (i.e. unbreakable) if the key is chosen uniformly at random among all $\mathbb{Z}_{26}^n$ possible values. The price we pay for having an unbreakable system is that the length of the key must be at

least as long as the plaintext. This is of course totally unacceptable in most practical situations. However, there has been a few classical "communication channels" where the Vernam cipher has been used, one was the "hot-line" between Washington and Moscow.

The Vernam cipher is also important because it forms the basis for the class of symmetric encryption schemes called *stream ciphers*. Basically, a stream cipher is a Vernam cipher, but instead of using a completely random key of length n, it uses a much smaller key (by a smaller key we mean that the number of possible keys is smaller), which is transformed into a length n *pseudo-random sequence* (keystream).

The simple transposition cipher: (or permutation cipher) A simple transposition cipher with a period t involves grouping the plaintext into blocks of t consecutive characters. The key is a permutation π on the positions within the block, which means that there are $t!$ possible keys. The encryption function maps the plaintext $\mathbf{m} = m_1, m_2, \ldots$ to the ciphertext $\mathbf{c} = c_1, c_2, \ldots$, through

$$c = E_k(m_1, m_2, \ldots, m_t), E_k(m_{t+1}, m_{t+2}, \ldots, m_{t+t}), \ldots,$$

where

$$E_k(m_1, m_2, \ldots, m_t) = m_{\pi(1)}, m_{\pi(2)}, \ldots, m_{\pi(t)}.$$

Decryption is done through the inverse permutation,

$$D_k(c_1, c_2, \ldots, c_t) = c_{\pi^{-1}(1)}, c_{\pi^{-1}(2)}, \ldots, c_{\pi^{-1}(t)}.$$

Example 2.11. *Let the key be* $\pi = (25134)$*, meaning that* $\pi(1) = 2, \pi(2) = 5, \ldots$*. Then* $E_k(m_1, m_2, \ldots, m_5) = (m_2, m_5, m_1, m_3, m_4)$*.*

Assume that $\mathbf{m} =$ `findingthetreasurecanonlybedoneby ....` *Then* $E_k($`findi`$) =$ `iifnd`*,* $E_k($`ngthe`$) =$ `genth`*, etc., and the ciphertext is*

$$\mathbf{c} = \texttt{iifndgenthrstearauecoynnl}\ldots$$

Decryption is done by the inverse permutation $\pi^{-1} = (31452)$*, and* $D_k(c_1, c_2, \ldots, c_5) = (c_3, c_1, c_4, c_5, c_2)$*. Decrypting gives* $D_k($`iifnd`$) =$ `findi`*, etc.*

Since a simple transposition cipher does not change the statistics, it is an easy task to force it.

Example 2.12. *Assume we have observed*

$$c = \texttt{iifndgenthrstearauecoynnl}\ldots$$

Knowing that a permutation of each block of some t letters should result in readable text, there are not many options for the first letters `iifndgen...`*. A suitable choice would be that the message starts as* `find...`*.*

Next we describe another historical cipher, the *Hill cipher*. The Hill cipher acts on t-grams from $\mathbb{Z}_{26}$ through a key which is an invertible $t \times t$ matrix $A = [a_{ij}]$. The encryption

function maps the plaintext $\mathbf{m} = m_1, m_2, \ldots$ to the ciphertext $\mathbf{c} = c_1, c_2, \ldots$, through $c = E_k(m_1, m_2, \ldots, m_t), E_k(m_{t+1}, m_{t+2}, \ldots, m_{t+t}), \ldots$, where

$$E_k(m_1, m_2, \ldots, m_t) = (m_1, m_2, \ldots, m_t)A.$$

Decryption involves using A^{-1}.

Although a ciphertext-only attack can be quite tricky against a Hill cipher, its linearity makes it suseptible to a *known plaintext attack*. In a known plaintext attack, a part of the plaintext is known to the attacker, and the task is to recover the key, or the unknown part of the plaintext.

Rotor-based machines: Polyalphabetic substitution ciphers implemented by a class of rotor-based machines were the dominant cryptographic tool in World War II. A *rotor* is a mechanical wheel designed to control other rotors or devices inside the machine. A rotor machine has a mechanical key board and a series of rotors and other physical devices that usually implement some version of a polyalphabetic substitution cipher. It is interesting to note that rotor-based machines are purely mechanical devices without power supply, making them very robust (like a typewriter).

The most well known rotor-based machine is the *Enigma* (used by the Germans in World War II). A Swede named Boris Hagelin also made several designs of rotor-based machines. One of the, called M-209, was extensively used by the US army during the 1940s.

2.8 Problems

Problem 2.1. The *affine cipher* is defined by the encryption function

$$E_k(m) = am + b, \quad a, b, m \in \mathbb{Z}_{26},$$

where the key is $k = (a, b)$. However, not all pairs (a, b) are possible keys.

a) Determine the decryption function.

b) Determine the number of possible keys.

Problem 2.2. The affine cipher (se Problem 1.1) has been used to encrypt the following ciphertext (from D. Stinson: *Cryptography: Theory and Practice*):

```
fmxvedkaphferbndkrxrsrefmorudsdkdvshvufedkaprkdlyevlrhhrh
```

Determine the plaintext.

Problem 2.3.

A Vigenére cipher has been used to encrypt the following ciphertext (repeated textblocks are underlined).

```
aukhy jamki zymwm jmigx nfmlx etimi zhbhr
aymzm ilvme jkutg dpvxk qukhq lhvrm jazng
gzvxe nlufm pzjnv chuas hkqgk iplwp ajzxi
gumtv dptej ecmys qybav alahy poexw pvnye
eyxee udpxr bvzvi ziivo spteg kubbr qllxp
wfqgk nllle ptikw djzxi goioi zlamv kfmwf
nplzi ovvfm zktxg nlmdf aaexi jlufm pzjnv
     caigi uawpr nviwe jkzas zlafm hs
```

Find the key and the plaintext.

Problem 2.4. Show that the transposition cipher is a special case of the Hill cipher.

Problem 2.5. In a Hill cipher the key is chosen as any invertible $t \times t$ matrix $A = [a_{ij}]$. Determine the number of possible keys if $t = 2$ and $t = 3$ (and the alphabet size is 26).