Vorlesung Kryptologie:
Kapitel 1: Historische Chiffren

von
Peter Hellekalek

Fachbereich Mathematik, Universität Salzburg

Tel: +43-(0)662-8044-5310
Fax: +43-(0)662-8044-137
e-mail: peter.hellekalek@sbg.ac.at
web: http://random.mat.sbg.ac.at/

Salzburg, 19. März 2014

Inhaltsverzeichnis

Kapitel 1

Grundlagen und historische Verfahren

▷ **Inhalt**
In diesem Kapitel klären wir zunächst einige Begriffe und lernen dann berühmte klassische Verfahren der Kryptographie kennen.

▷ **Ziel**
Wir stellen zentrale Konzepte der Kryptographie an Hand klassischer Verfahren vor. Auf den Erfahrungen, die man mit diesen Methoden im Laufe der Geschichte machte, bauen die modernen kryptographischen Algorithmen auf.

▷ **Stichwörter**
Die Stichwörter zu diesem Kapitel lauten

- Verschlüsselungsverfahren, Chiffre
- Cäsarchiffre und Verschiebechiffre
- Substitutionschiffre
- Häufigkeitsanalyse
- Vigenére- und Hillchiffre
- one-time pad
- Zufallszahlengeneratoren

▷ **Literatur**
Die folgenden Bücher sind besonders empfehlenswert.

Buchmann[Buc01], Ertel[Ert01], Stinson[Sti06], Trappe und Washington[TW06], Vaudenay[Vau06]

▷ **Links**
`http://people.csail.mit.edu/rivest/crypto-security.html`
(Linksammlung von Ron Rivest; als Starthilfe gedacht)

1.1 Einleitung

Seit der Antike ist die sichere Übermittlung sensibler Daten eine Grundaufgabe der Gesellschaft. Bereits damals wurden Techniken ersonnen, um Nachrichten zu verschlüsseln und so deren Inhalt vor dem Gegner zu schützen.

In der modernen Informationsgesellschaft besteht ein enormer Bedarf an sicherem Datenaustausch, zum Beispiel beim Homebanking oder beim Online-Zahlungsverkehr, sowie bei der Gewährleistung der Datenintegrität, etwa in Zusammenhang mit der digitalen Signatur oder bei der elektronischen Übermittlung von Gesundheitsdaten.

Hinter dem kleinen Vorhängeschloss, das Sie in Ihrem Browser beim Aufruf von bestimmten Seiten sehen, sowie bei den Zertifikaten zu diverser Software, die Sie in manchen Situationen überprüfen sollen, stehen mathematische Konzepte, die teilweise nicht älter als dreißig Jahre sind.

Von diesen modernen Konzepten und ihrer Vorgeschichte handelt diese Vorlesung.

Bei diesen kryptographischen Vorgängen sind meist mehrere (Kommunikations-) Partner beteiligt, wie zum Beispiel der Sender und der Empfänger einer Nachricht sowie ein fiktiver Angreifer. Es ist in der Kryptographie üblich, bei der Beschreibung derartiger Abläufe den einzelnen Instanzen Namen zu geben.

Mit *Alice* und *Bob* werden die beiden Kommunikationspartner bezeichnet, die verschlüsselte und/oder signierte Nachrichten austauschen möchten. *Eve* bezeichnet die dritte Partei, die dieses System attackiert und den Geheimtext oder die digitale Signatur knacken möchte. Manchmal wird dieser Gegner auch mit *Mallory* bezeichnet.

Eve besitzt mehrere Möglichkeiten, sich an der Kommunikation zwischen Alice und Bob zu beteiligen:

1. unbefugtes Entschlüsseln des Geheimtextes, z.B. durch Herausfinden des Schlüssels,

2. unbefugtes Verändern des Geheimtextes auf eine Art, die Bob nicht erkennen kann, und anschließendes Weitersenden an Bob,

3. Eve gibt sich Bob gegenüber als Alice aus und kommuniziert mit ihm, ohne dass er diesen Betrug merkt.

Es ist seit der Antike klar, dass die Schlüssel für die Ver- und Entschüsselung geheimer Botschaften möglichst sicher verwahrt werden müssen. Wie macht man dies auf einem Computer, angesichts von Bedrohungen wie Trojanern?

Wie übermittelt man einen geheimen Schlüssel über das Internet, angesichts der vielen Möglichkeiten des Mitlauschens?

Wie verwaltet man große Netzwerke zum Austausch sicherer Nachrichten zwischen tausenden von Benutzern?

Dies sind einige der Aufgaben, die man mit der modernen Kryptographie zu lösen versucht.

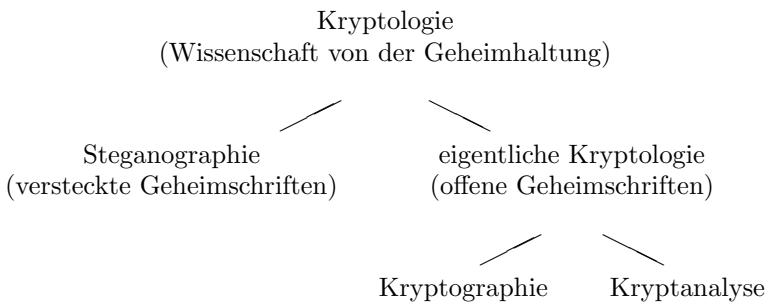
Im folgenden Abschnitt legen wir die mathematischen Begriffe fest, mit denen wir kryptologische Prozesse beschreiben und analysieren werden.

1.2 Definitionen

1.1 Bemerkung (Ziele der Lehrveranstaltung)
Die Ziele der Lehrveranstaltung lauten:

- Diskussion der grundlegenden kryptographischen Verfahren:
  - historische Verfahren
  - moderne symmetrische Verfahren (DES, AES)
  - moderne asymmetrische Verfahren (RSA, DH, ...)
  - Hashfunktionen
- Hinweise auf das gesellschaftliche und technologische Umfeld:
  - digitale Signatur
  - e-commerce
  - internet security
  - cyber-warfare

1.2 Definition (Kryptologie)
Kryptologie ist die Wissenschaft vom Geheimhalten von Information. Sie bietet Methoden an, sogenannte Geheimschriften, um die Vertraulichkeit von Daten sicherzustellen. Wir unterscheiden zwischen den folgenden Teilgebieten der Kryptologie:



1.3 Bemerkung
In dieser Vorlesung geht es fast ausschließlich um die eigentliche Kryptologie, und hier vor allem um die Kryptographie. Beispiele für steganographische Verfahren gibt es seit der Antike (rasierte Schädel von Sklaven, Wachstafeln, etc. Genaueres in der Vorlesung). Effizientere Methoden sind unsichtbare Tinten, der Mikropunkt (Stichwort 2. Weltkrieg), sowie die modernen Verfahren, die Daten in Bildern oder Sound- und Video-Dateien verstecken.
▷ Link: `http://www.jjtc.com/Steganography`

1.4 Definition (Verschlüsselungsverfahren, engl.: cryptosystem)
Unter einem Verschlüsselungsverfahren oder Kryptosystem (oft auch kurz "Chiffre") verstehen wir ein Quintupel $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ mit den folgenden Eigenschaften:

1. $\mathcal{P}$ ist eine nichtleere endliche Menge und wird der Klartextraum genannt. Die Elemente von $\mathcal{P}$ heißen Klartextelemente (engl.: plaintext elements).

2. $\mathcal{C}$ ist eine nichtleere endliche Menge und wird der Chiffretextraum genannt. Die Elemente von $\mathcal{C}$ heißen Chiffretextelemente (engl.: ciphertext elements). Oft sagt man auch: Geheimtextraum bzw. Geheimtextelemente

3. $\mathcal{K}$ ist eine nichtleere endliche Menge und wird der Schlüsselraum genannt. Die Elemente von $\mathcal{K}$ heißen Schlüssel (engl.: keys).

4. $\mathcal{E} = \{e_k : k \in \mathcal{K}\}$ ist eine Familie von Funktionen mit der Eigenschaft:

$$\forall\, k \in \mathcal{K}: \quad e_k : \mathcal{P} \mapsto \mathcal{C} \quad \text{injektiv.}$$

   Die Elemente von $\mathcal{E}$ heißen Verschlüsselungsfunktionen (engl.: encryption functions).

5. $\mathcal{D} = \{d_k : k \in \mathcal{K}\}$ ist eine Familie von Funktionen mit der Eigenschaft:

$$\exists \mathcal{C}' \subseteq \mathcal{C} : \forall\, k \in \mathcal{K}: \quad d_k : \mathcal{C}' \mapsto \mathcal{P} \quad \text{injektiv}$$

   Die Elemente von $\mathcal{D}$ heißen Entschlüsselungsfunktionen (engl.: decryption functions).

6. Es gilt folgende Beziehung zwischen den Elementen von $\mathcal{E}$ und $\mathcal{D}$:

$$\forall\, k \in \mathcal{K} \quad \exists\, k' \in \mathcal{K}\ \forall\, c \in \mathcal{C}': \quad d_{k'} \circ e_k = id_{\mathcal{P}}, \quad e_k \circ d_{k'} = id_{\mathcal{C}'}.$$

1.5 Bemerkung
Ein Klartextelement beziehungsweise ein Chiffretextelement kann aus einem einzelnen Zeichen oder einem Zeichenblock bestehen, zum Beispiel:

1|0|1|0|0|1|1|0|1|1	... einzelne Zeichen
01101010|0110110	... Zeichenblöcke

Je nachdem, ob der Klartext nun Zeichen für Zeichen oder blockweise verschlüsselt wird, unterscheidet man zwischen *Stromchiffren* und *Blockchiffren*.

Bei der Klassifikation der Attacken auf ein kryptographisches System unterscheidet man vier Grundformen:

1. Attacke nur mit Geheimtext: Eve besitzt nur einen Geheimtext und versucht diesen zu entschlüsseln.

2. Attacke mit bekanntem Klartext: Eve besitzt ein Stück Geheimtext und den zugehörigen Klartext.

3. Attacke mittels wählbarem Klartext: Eve erhält für begrenzte Zeit Zugang zum Verschlüsselungsmechanismus. Sie kennt zwar den geheimen Schlüssel nicht, kann aber die Klartexte frei wählen. Aus den erhaltenen Klartext-Geheimtext-Paaren wird sie versuchen, den geheimen Schlüssel herauszufinden.

4. Attacke mittels wählbarem Geheimtext: Eve erhält für begrenzte Zeit Zugang zum Entschlüsselungsmechanismus. Sie kennt zwar den geheimen Schlüssel nicht, kann aber die Geheimtexte frei wählen. Aus den erhaltenen Geheimtext-Klartext-Paaren wird sie versuchen, den geheimen Schlüssel herauszufinden.

1.3 Verschiebeschiffre

1.6 Beispiel (Verschiebechiffre)
Wir identifizieren das Alphabet $\Sigma = \{a, b, c, \ldots, x, y, z\}$ mit der Menge der Restklassen modulo 26, $\mathbb{Z}_{26} = \mathbb{Z}/26\mathbb{Z} = \{\overline{0}, \overline{1}, \ldots, \overline{25}\}$:

$$\begin{aligned} a &\leftrightarrow \overline{0} \\ b &\leftrightarrow \overline{1} \\ &\vdots \\ z &\leftrightarrow \overline{25} \end{aligned}$$

Wir wählen $\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}$ und definieren für $k \in \mathcal{K}$ die folgenden Ver- und Entschlüsselungsfunktionen:

$$\begin{aligned} e_k(p) &= p + k \\ d_k(c) &= c - k \end{aligned}$$

Hinweis: Beachten Sie, dass hier im Restklassenring $(\mathbb{Z}_{26}, +, \cdot)$ gerechnet wird!

Das Quintupel $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ ist ein Kryptosystem. Dazu ist nur zu zeigen:

$$\forall\, p \in \mathcal{P} = \mathbb{Z}_{26},\ \forall\, k \in \mathcal{K} = \mathbb{Z}_{26} : \quad d_k(e_k(p)) = (p + k) - k = p.$$

1.7 Definition (Verschiebechiffre)
Das Kryptosystem von Beispiel 1.6 heißt eine *Verschiebechiffre*.

1.8 Bemerkung
Wir werden im Folgenden die Restklassen $\overline{0}, \overline{1}, \overline{2}, \ldots$ durch die Repräsentanten $0, 1, 2, \ldots$ ersetzen und an Stelle von Operationen wie

$$\overline{7} \cdot \overline{3} + \overline{9} = \overline{4}$$

im Ring $(\mathbb{Z}_{26}, +, \cdot)$ einfach modulare Arithmetik verwenden:

$$7 \cdot 3 + 9 \equiv 4 \pmod{26}$$

Also: Wir lassen die Querstriche weg, schreiben $\equiv$ an Stelle von $=$ und fügen (mod 26) an. Wenn dies zu umständlich wird, Manchmal schreiben wir sogar nur $7 \cdot 3 + 9 \equiv 4\ (26)$.

1.9 Bemerkung (Cäsar-Chiffre)
Die spezielle Verschiebechiffre mit dem Schlüssel $k = 3$ wird auf Grund ihrer Geschichte Cäsar-Chiffre genannt:

Klartext	a	v	e	c	a	e	s	a	r	...
Zahlencode	0	21	4	2	0	4	18	0	17	...
Schlüssel	3	3	3	3	3	3	3	3	3	...
Geheimtextcode	3	24	7	5	3	7	21	3	20	...
Geheimtext	D	Y	H	F	D	H	V	D	U	...

1.10 Bemerkung (Sicherheit der Verschiebechiffre)
Mit der Sicherheit der Verschiebechiffre ist es nicht weit her. Die Verschiebechiffre ist nicht sicher: durch Ausprobieren aller möglichen Schlüssel können wir leicht auf den Klartext kommen.

1.11 Bemerkung (Allgemeine Sicherheitsaspekte)
Bereits das einfache Beispiel der Verschiebechiffre zeigt einige grundlegende Tatsachen auf:

1. Wenn der Schlüsselraum zu klein ist, dann läßt sich das Verfahren schon mit Durchprobieren aller möglichen Schlüssel knacken. Eine derartige brutale Methode heißt eine *brute-force-Attacke*.

   ▷ **Erkenntnis:** Wir müssen den Schlüsselraum $\mathcal{K}$ möglichst groß machen.

2. Sender und Empfänger der Nachricht müssen bei derartigen Kryptosystemen den gleichen Schlüssel besitzen.

   ▷ **Erkenntnis:** Wir müssen den Schlüssel auf einem sicheren Weg übermitteln (Kurier,...) übermitteln. Dies ist aufwändig.

3. Die Hintereinanderausführung zweier Verschlüsselungsfunktionen ergibt wieder eine Verschiebechiffre:

   $$d_k \circ d_l \;=\; d_{k+l} \pmod{26}.$$

   Man spricht in einem solchen Fall von der *Gruppeneigenschaft* des Chiffrierverfahrens.

   ▷ **Erkenntnis** Nochmalige Verschlüsselung des Geheimtextes mit einer Verschiebechiffre erhöht die Sicherheit nicht.

1.4 Substitutionschiffre

1.12 Bemerkung (Permutationen)
Sei $S \neq \emptyset$ eine beliebige endliche Menge und sei $n = |S|$ die Anzahl der Elemente in S. Eine Permutation von S ist eine bijektive Abbildung $\pi : S \mapsto S$.

Jede Permutation von S kann in der Form

$$\pi = \begin{pmatrix} 1 & 2 & \dots & n \\ \pi(1) & \pi(2) & \dots & \pi(n) \end{pmatrix}$$

geschrieben werden. Bezeichne S_n die Menge der Permutationen von S. Dann heißt S_n die symmetrische Gruppe (über S), da –wie man leicht überprüft– das Paar $(S_n, \circ)$ eine Gruppe bildet. Mit dem Symbol $\circ$ wird die Hintereinanderausführung von Funktionen bezeichnet. $(S_n, \circ)$ ist für $n \geq 3$ nicht kommutativ. Dies ist am Beispiel der S_3 (und damit für alle $n \geq 3$) leicht einzusehen.

Aus der diskreten Mathematik wissen wir, dass $|S_n| = n!$ gilt.

1.13 Beispiel
Wir betrachten den Fall $n = 3$ etwas genauer:

$$S_3 = \left\{ \underbrace{\begin{pmatrix} 1 & 2 & 3 \\ 1 & 2 & 3 \end{pmatrix}}_{\substack{\text{Identität} \\ \text{auf } \{1,2,3\}}}, \underbrace{\begin{pmatrix} 1 & 2 & 3 \\ 2 & 1 & 3 \end{pmatrix}}_{=:f}, \underbrace{\begin{pmatrix} 1 & 2 & 3 \\ 2 & 3 & 1 \end{pmatrix}}_{=:g}, \dots \right\}$$

$$f \circ g = \begin{pmatrix} 1 & 2 & 3 \\ 1 & 3 & 2 \end{pmatrix} \neq \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix} = g \circ f$$

$$g^2 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 1 & 2 \end{pmatrix}$$

$$f \circ g^2 = \begin{pmatrix} 1 & 2 & 3 \\ 3 & 2 & 1 \end{pmatrix} = g \circ f$$

Es gilt (nachrechnen!): $S_3 = \{e, f, g, g^2, f \circ g, f \circ g^2\}$

Die Permutation $\pi \in S_n$ heißt zyklisch, wenn es eine Teilmenge $\{i_1, i_2, \dots, i_k\}$ von S gibt mit

$$\pi(i_1) = i_2, \quad \pi(i_2) = i_3, \quad \dots \quad \pi(i_k) = i_1$$

und die übrigen Elemente von S fest bleiben. Man schreibt dann in der sogenannten Zyklenschreibweise

$$\pi = \begin{pmatrix} i_1 & i_2 & \dots & i_k \end{pmatrix}.$$

1.14 Beispiel
Wir machen uns mit der Zyklenschreibweise vertraut:

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

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

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

Am Beispiel sehen wir, daß manche Permutationen ein *Produkt von Zyklen* sind, wobei diese Zyklen paarweise disjunkt sind. Diese Eigenschaft gilt ganz allgemein, also für beliebige Permutationen $\pi \in S_n$, $n \in \mathbb{N}$.

1.15 Definition (Substitutionschiffre, Giovanni Battista Argenti, um 1580)
Unter einer Substitutionschiffre versteht man ein Chiffriersystem der folgenden Gestalt: sei $\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}$, sei $\mathcal{K} = S_{26}$ die Menge der Permutationen von 26 Elementen, und seien die Verschlüsselungs- und Entschlüsselungsfunktionen definiert durch

$$\begin{aligned} e_\pi(p) &:= \pi(p) \\ d_\pi(c) &:= \pi^{-1}(c) \qquad\qquad (\pi \in S_{26}) \end{aligned}$$

1.16 Bemerkung (Kryptoanalyse der Substitutionschiffre)

1. Jedem Klartextelement p ist stets dasselbe Geheimtextelement c zugeordnet.
2. Gleiche Klartextblöcke werden in gleiche Geheimtextblöcke übergeführt.
3. Die Hintereinanderausführung von zwei Substitutionen steigert die Sicherheit nicht. Sie ist gleichwertig zu einer einzigen Substitution, da $(S_n, \circ)$ eine Gruppe bildet.
4. Auch wenn wir nur den Geheimtext kennen, also bei einer sogenannten *ciphertext-only*-Attacke, können wir uns bei ausreichend langen Texten auf statistische Methoden stützen: Jedem Klartextelement ist ja stets das gleiche Geheimtextelement zugeordnet.

   Wir können daher in deutschen, englischen, ... Texten rasch das Bild von –zum Beispiel– dem Buchstaben e unter der Substitution herausfinden und dann diese *statistische Analyse* fortsetzen, durch eine Häufigkeitsanalyse der Buchstaben, der Buchstabenpaare und der Buchstabentripel. Die statistische Analyse längerer Tupel ist selten sinnvoll.
5. Wenn wir vermuten, daß im Klartext bestimmte Wörter vorkommen (wie z.B. das Wort `communication`), dann führen wir eine sogenannte *Mustersuche* durch (siehe unten sowie die Bücher [Bau97, Sti06] für ausführliche Beispiele).
6. Der große Schlüsselraum $\mathcal{K}$ mit $|\mathcal{K}| = 26! \sim 4 \cdot 10^{26}$ bietet keinen ausreichenden Schutz gegen das Brechen der Chiffre. Die Substitutionschiffre kann durch die beschriebene Häufigkeitsanalyse, verbunden mit einer Mustersuche, gebrochen werden.

   ▷ **Erkenntnis:** Große Schlüsselräume sind für die Sicherheit eines Chiffrierverfahrens notwendig, aber nicht hinreichend!

1.17 Beispiel Gegeben sei der folgende Geheimtext, von dem bekannt ist, dass der englische Klartext mit einer uns unbekannten Substitution erzeugt wurde (siehe Abbildung 1.1).

Durch die einfache Frequenzanalyse (Abbildung 1.2) erhalten wir eine erste Vermutung, wie der Buchstabe e verschlüsselt wurde.

```
zhjeo ndize hicle osiol digic lmhzq zolyi zehdp zhjeo ndize
hycdh hlpvs uczyc dhzhj eondi zehge moylk zhjpm lhylg gidiz
gizyd ppsdo lylzr losye nnmhz ydize hicle osceu lrloq lgyoz
vlgic lneol flhlo dpydg lzhuc zyciu eeone olzhj eondi zehge
moylg zhjpm lhyll dycei clogi dizgi zydpp siclq zolyi zehej
iczgz hjpml hylzg lkaol gglqv sqzol yilqi odhgj eondi zehxm
dhizi zlguc zycyd hehps vlqlo zrlqz jiclp duejy dmgdp ziszg
evglo rlqqz gizhf mzgcz hficl ldopz loydm gljoe niclp dilol
jjlyi zhvze pefsd hqgey zepef syenn mhzyd izehi cleos gllng
iecdr luzql daapz ydize hgqml ieicl jdyii cdipz rzhfv lzhfg
dolvs iclzo dyize hggem oylge jzhje ondiz ehucz yczhj pmlhy
lldyc eiclo zhdpp aeggz vplqz olyiz ehgic laolg lhiad aloql
gyzvl gicly dglej vzqzo lyize hdpye nnmhz ydize hicle osdaa
pzlqi eiclg eyzdp vlcdr zemoe jneht lsg…
```

Abbildung 1.1: Der Geheimtext.

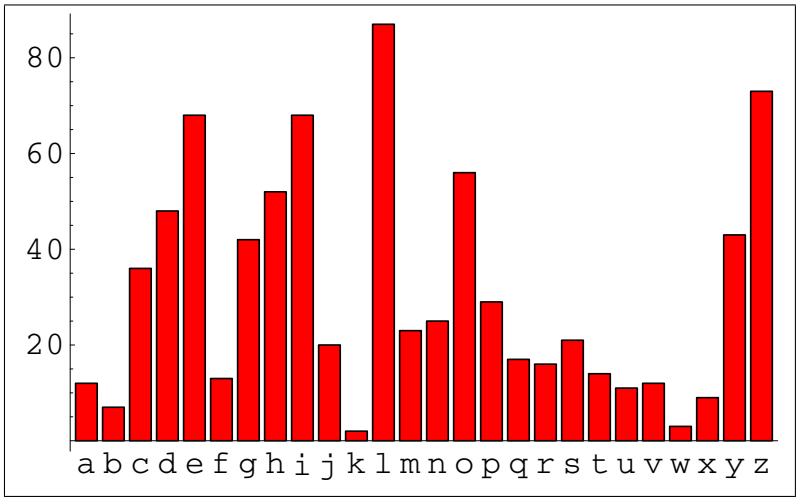


Abbildung 1.2: Häufigkeiten im Geheimtext.

$\mathbf{p}$	e
$\mathbf{e}_\pi(\mathbf{p})$	l

Wir haben Grund zur Annahme, dass in diesem englischen Klartext das Wort "`communication`" vorkommt. Diese Information wird uns eine wichtige Hilfe beim Entschlüsseln sein, wie sich gleich zeigen wird.

"`communication`" ist ein String der Länge 13 mit folgenden Eigenschaften:

1. = 8. Buchstabe
2. = 12. Buchstabe
3. = 4. Buchstabe
6. = 13. Buchstabe
7. = 11. Buchstabe

Wir führen nun im Geheimtext eine Mustersuche durch, indem wir alle Strings mit dieser kombinatorischen Eigenschaft (1.=8. Buchstabe, 2.=12. Buchstabe, usw.) suchen. Durch Vergleich mit den entsprechenden Geheimtextstellen erkennen wir:

$\mathbf{p}$	c	o	m	u	n	i	a	t
$\mathbf{e}_\pi(\mathbf{p})$	y	e	n	m	h	z	d	i

Wir erhalten damit folgenden partiell entschlüsselten Text

```
I N   O   M A T I O N T   E O     T   E A T   T   E
U N I   I   E C T I O N A   I N   O   M A T I O N
C       A N N E         I C   A N I N   O   M A T I
O N       O U   C E   I N     U E N C E     T A T I
T I C     A       A   E C E I   E     C O M M U N I
C A T I O     N T   E O       O   E   E     E   C   I
E   T   E     M O   E   E N E   A   C A   E I N     I
C   T   O O       M O   E I N   O   M A T I O N   O
U   C E   I N         U E N C E E A C   O T   E T A
T I   T I C A           T   E   I   E C T I O N O   T
I   I N     U E     N C E I   E       E     E I   E C T
E   T       A N     O   M A T I O N   U A N T I T I
E       I C       C A N O N       E   E   I E   I   T E
A   O       C A U   A   I T   I   O E     E     I T I N
U I     I N   T   E E A I E   C A U   E O     M T   E
A T E   E     E C T I N   I O   O A N         O C I O
O     C O M M U N I C A T I O N T   E     O       E
E M   T O A   E   I   E A       I C A T     I O N
U E T O T   E   A C T T   A T I   I N         E I N
A E     T   E I   A C T I O N     O U       C E   O
I N   O   M A T I O N     I C   I N         U E N C E
E A C   O T   E   I N A O     I         E   I   E C T
I O N   T   E     E   E N T   A   E     E   C I E   T
E C A   E O     I   I E C T I     O N A C O M M U N
I C A T I O N T E O     A           I E   T O T   E
O C I A     E   A I O U   O       M O N
```

Mit ein bisschen zusätzlichem Aufwand gelingt es schließlich, den Geheimtext komplett zu entschlüsseln. Wir geben den Schlüssel an:

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

Abbildung 1.3: Der Schlüssel.

1.5 Häufigkeitsanalyse

Wir haben bereits in Zusammenhang mit der Substitutionschiffre gesehen, dass eine Häufigkeitsanalyse sehr wirkungsvoll sein kann. In diesem Abschnitt geben wir einige Beispiele für diese Anwendung statistischer Methoden.

In den ersten Graphiken präsentieren wir die Buchstabenhäufigkeiten der deutschen Sprache. Natürlich kann in einzelnen Texten die Verteilung der Häufigkeiten von diesen gemittelten Werten abweichen.

a	0.0647	n	0.0984
b	0.0193	o	0.0298
c	0.0268	p	0.0096
d	0.0483	q	0.0002
e	0.1748	r	0.0754
f	0.0165	s	0.0683
g	0.0306	t	0.0613
h	0.0423	u	0.0417
i	0.0773	v	0.0094
j	0.0027	w	0.0148
k	0.0146	x	0.0004
l	0.0349	y	0.0008
m	0.0258	z	0.0114

Abbildung 1.4: Buchstabenhäufigkeiten im Deutschen, in Zahlen

Wir ordnen die Buchstaben nach ihrer Häufigkeit, beginnend mit dem häufigsten Buchstaben:

Wenn wir die Häufigkeiten alphabetisch ordnen, dann erhalten wir die folgende Graphik:

Eine wichtige Rolle spielt die Analyse der Paarhäufigkeiten, denn schließlich tritt nicht jedes mögliche Paar von Buchstaben auf und unter jenen Paaren, die im Deutschen vorkommen, sind manche häufig anzutreffen und manche sind sehr selten. Wir geben die zehn häufigsten Paare an:

Dieselbe Art von Häufigkeitsanalyse kann man auch für die Tripel durchführen:

Zwei Beispiele aus der Literatur (J. W. von Goethe: Iphigenie auf Tauris und Hugo von Hofmannsthal: Reitergeschichte):

Die kleinen Unterschiede in den Häufigkeitsverteilungen sind deutlich zu erkennen. Ganz ähnliche Aussagen erhält man für englische Texte und viele andere

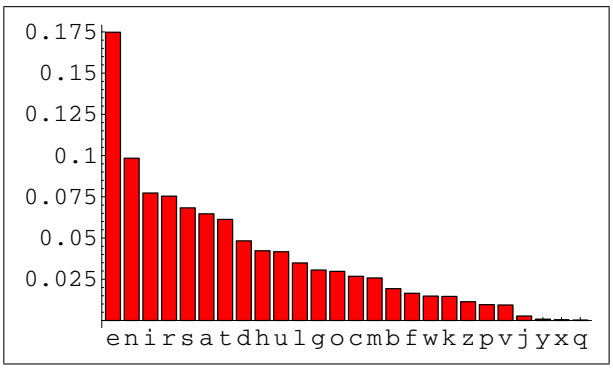


Abbildung 1.5: Buchstabenhäufigkeiten im Deutschen

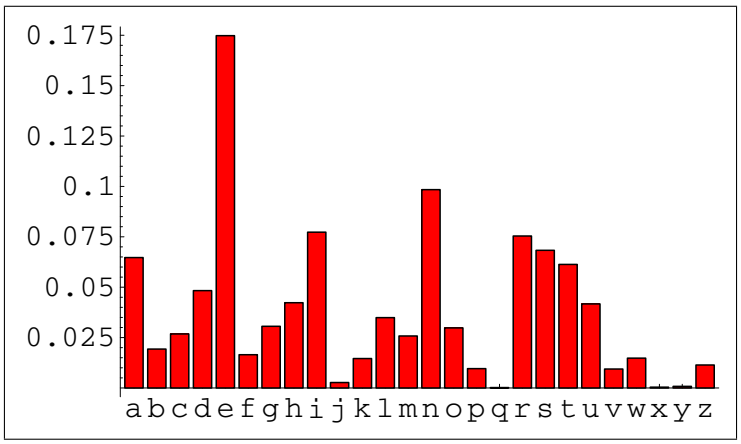


Abbildung 1.6: Buchstabenhäufigkeiten im Deutschen

Sprachen.

In der Kryptanalyse geht man wesentlich mehr auf die Details einer Sprache ein, als wir es hier getan haben. Es wird dann zwischen verschiedenen Texttypen unterschieden, wie etwa militärischen Nachrichten, wissenschaftlichen Texten oder diplomatischen Mitteilungen, da sich zwischen all diesen Typen von Texten die Häufigkeitsverteilungen etwas ändern. Geheimdienste verfügen seit langem über umfangreiches Datenmaterial, um diese Art von Analyse durchführen zu können.

1.18 Bemerkung (Project Gutenberg)
Project Gutenberg ist die erste und umfassendste Sammlung freier elektronischer Bücher. Ähnlich wie bei Wikipedia, so haben auch hier viele freiwillige Helfer eine wichtige Informationsquelle für die Allgemeinheit geschaffen.

Mit Hilfe der im Project Gutenberg verfügbaren Literatur können Sie selbst umfangreiche Textanalysen durchführen.

▷ Link: `http://www.gutenberg.org`

er	0.0409
en	0.0400
ch	0.0242
de	0.0227
ei	0.0193
nd	0.0187
te	0.0185
in	0.0168
ie	0.0163
ge	0.0147

Abbildung 1.7: Die zehn häufigsten Paare im Deutschen

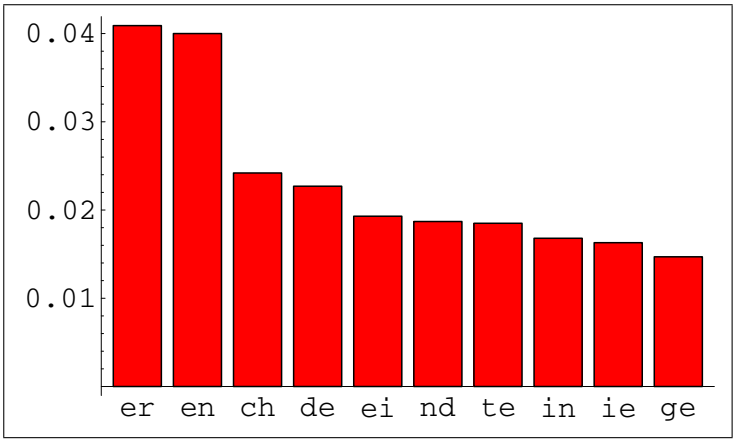


Abbildung 1.8: Die zehn häufigsten Paare im Deutschen

1.6 Affine Chiffre

Die Verschlüsselungsfunktionen $x \mapsto x + \overline{b}$ der Verschiebechiffre sind spezielle affine Funktionen auf $\mathbb{Z}_{26}$. Die allgemeine Form einer affinen Funktion von $\mathbb{Z}_{26}$ in sich ist $x \mapsto \overline{a} \cdot x + \overline{b}$, $\overline{a}, \overline{b} \in \mathbb{Z}_{26}$.

1.19 Beispiel
Sei $\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}$. Für welche Elemente $\overline{a}, \overline{b} \in \mathbb{Z}_{26}$ ist die Abbildung

$$x \mapsto \overline{a} \cdot x + \overline{b} \qquad (x \in \mathbb{Z}_{26})$$

eine injektive Abbildung?

Wir merken an, dass für Abbildungen einer endliche Menge in sich die folgende Beziehung gilt:

$$\text{bijektiv} \Leftrightarrow \text{injektiv} \Leftrightarrow \text{surjektiv}.$$

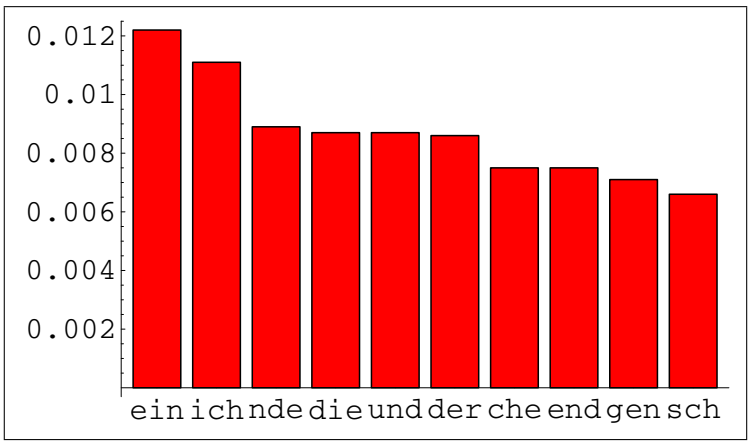


Abbildung 1.9: Die zehn häufigsten Tripel im Deutschen

Daher ist klar, dass unsere Frage eine Frage nach den Bedingungen für die Bijektivität ist. Die Abbildung

$$y \mapsto y + \overline{b} \qquad (y \in \mathbb{Z}_{26})$$

ist offensichtlich eine bijektive Abbildung auf $\mathbb{Z}_{26}$. Daher müssen wir nur noch untersuchen, wann

$$x \mapsto \overline{a} \cdot x \qquad (x \in \mathbb{Z}_{26})$$

bijektiv auf $\mathbb{Z}_{26}$ ist.

Wegen der oben angesprochenen Äquivalenz zwischen injektiv, surjektiv und bijektiv untersuchen wir die Surjektivität von $x \mapsto \overline{a} \cdot x$:

$x \mapsto \overline{a} \cdot x$ ist surjektiv

$\Leftrightarrow$ $\forall\, \overline{c} \in \mathbb{Z}_{26} : \exists\, x$ mit $\overline{a} \cdot x = \overline{c}$

$\Leftrightarrow$ $\forall\, c \in \mathbb{Z} : \exists\, x \in \mathbb{Z}$ mit $a \cdot x \equiv c \pmod{26}$

$\Leftrightarrow$ die Lösbarkeitsbedingung muss für alle c gelten

$\Leftrightarrow$ $(a, 26) = 1$

Die Abbildung $x \mapsto \overline{a} \cdot x + \overline{b}$, $x \in \mathbb{Z}_{26}$, ist eine Permutation von $\mathbb{Z}_{26}$ genau dann, wenn $(a, 26) = 1$. Sei nun

$$\mathcal{K} = \{(\overline{a}, \overline{b}) : (a, 26) = 1,\ b \text{ beliebig}\},$$

sei $e_{(\overline{a},\overline{b})} : \mathcal{P} \to \mathcal{C}$ die Verschlüsselungsfunktion zum Schlüssel $k = (\overline{a}, \overline{b})$,

$$e_{(\overline{a},\overline{b})}(p) = \overline{a} \cdot p + \overline{b} \qquad (\overline{a}, \overline{b}) \in \mathcal{K}.$$

Die zugehörige Dechiffrierfunktion $d_{(\overline{a},\overline{b})} : \mathcal{C} \to \mathcal{P}$ lautet:

$$d_{(\overline{a},\overline{b})}(c) := \overline{a}^{\,-1} \cdot (c - \overline{b}).$$

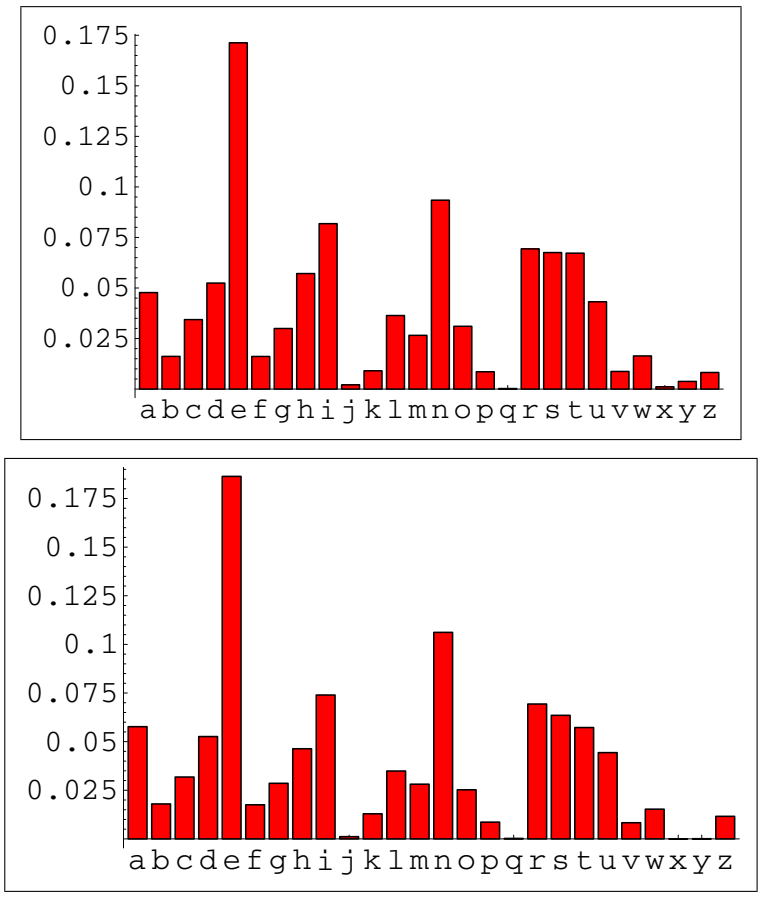


Abbildung 1.10: J. W. von Goethe und Hugo von Hofmannsthal

1.20 Definition (Affine Chiffre)
Diese Chiffre heißt eine affine Chiffre.

1.21 Bemerkung

1. Die affine Chiffre ist ein Spezialfall der Substitutionschiffre.
2. Für den Schlüsselraum gilt:

$$|\{\overline{a} \in \mathbb{Z}_{26} : (a, 26) = 1\}| = \varphi(26) = 12.$$

   Es folgt:

$$|\mathcal{K}| = \varphi(26) \cdot 26 = 12 \cdot 26 = 312.$$

3. Die Verschiebechiffre ist ein Spezialfall der affinen Chiffre, hier ist $\overline{a} = 1$.

1.22 Beispiel
Wir betrachten das Beispiel $(\overline{a}, \overline{b}) = (\overline{15}, \overline{3})$. Die Verschlüsselungsfunktion $e_{(\overline{15},\overline{3})}$

zum Schlüssel $k = (\overline{15}, \overline{3})$ lautet:

$$e_{(\overline{15},\overline{3})} : \; p \mapsto \overline{15} \cdot p + \overline{3}$$

Die Entschlüsselungsfunktion $d_{(\overline{15},\overline{3})}$ lautet:

$$d_{(\overline{15},\overline{3})}(c) = \overline{7} \cdot (c - \overline{3})$$

Diese Entschlüsselungsfunktion wurde auf folgende Weise gefunden. Wir betrachten als Erstes

$$\begin{aligned} \overline{15} \cdot \overline{x} &= \overline{1} \\ 15 \cdot x &\equiv 1 \pmod{26}. \end{aligned}$$

Für die Lösung dieser linearen Kongruenz untersuchen wir die linear diophantische Gleichung

$$15 \cdot x + 26 \cdot y = 1.$$

Die Lösbarkeitsbedingung dieser Gleichung lautet

$$(15, 26) = (3 \cdot 5, 2 \cdot 13) = 1.$$

Wir wenden den Euklidschen Algorithmus an, um eine Lösung dieser Gleichung zu finden:

$$\begin{array}{ll} 26 = 1 \cdot 15 + 11 & 1 = 4 - 3 \\ 15 = 1 \cdot 11 + 4 & \quad = 4 - (11 - 2 \cdot 4) \\ 11 = 2 \cdot 4 + 3 & \quad = 3 \cdot 4 - 11 \\ 4 = 1 \cdot 3 + 1 & \quad = 3 \cdot (15 - 11) - 11 \\ 3 = 3 \cdot 1 & \quad = 15 \cdot 3 - 4 \cdot 11 \\ & \quad = 15 \cdot 3 - 4 \cdot (26 - 15) \\ & \quad = 15 \cdot 7 + 26 \cdot (-4) \end{array}$$

Die modulo 26 eindeutige Lösung der linearen Kongruenz $15 \cdot x \equiv 1 \pmod{26}$ lautet daher

$$x \equiv 7 \pmod{26}.$$

1.23 Bemerkung (Kryptanalyse zur affinen Chiffre)
Wir wissen, daß es sich um eine affine Chiffre handelt und müssen aus dem Geheimtext den Schlüssel $(\overline{a}, \overline{b})$ bestimmen. Auf Grund des Studiums der Häufigkeitsverteilungen vermuten wir

$$\begin{array}{ccc} e \mapsto L & \text{und} & i \mapsto T \\ (\overline{4} \mapsto \overline{11}) & & (\overline{8} \mapsto \overline{19}) \end{array}$$

Wir erhalten zwei Gleichungen:

$$\begin{aligned} \overline{a} \cdot \overline{4} + \overline{b} &= \overline{11} \\ \overline{a} \cdot \overline{8} + \overline{b} &= \overline{19} \end{aligned}$$

Als Kongruenzen geschrieben:

$$\begin{aligned} a \cdot 4 + b &\equiv 11 \pmod{26} \\ a \cdot 8 + b &\equiv 19 \pmod{26} \end{aligned}$$

$$\begin{aligned} &\Rightarrow && 4 \cdot a \equiv 8 \pmod{26} \\ &\Rightarrow && 2 \cdot a \equiv 4 \pmod{13} \\ &\Rightarrow && a \equiv 2 \pmod{13} \end{aligned}$$

Kandidaten für $\overline{a} \in \mathbb{Z}_{26}$ sind daher die Restklassen $\overline{2}$ und $\overline{15}$. Wegen $(2, 26) = 2 \neq 1$ fällt der Kandidat $\overline{a} = \overline{2}$ weg, wegen $(15, 26) = 1$ folgern wir, dass $\overline{a} = \overline{15}$ und somit $\overline{b} = \overline{3}$ gilt. Damit lautet der gesuchte Schlüssel

$$(\overline{a}, \overline{b}) = (\overline{15}, \overline{3}).$$

1.24 Bemerkung (Sicherheitsaspekte)
Die Linearität der affinen Chiffre hat es uns ermöglicht, durch einfaches Gleichungslösen das System zu knacken. Allgemein gilt in der Kryptographie das Prinzip, dass *lineare* Verfahren durch das Lösen *linearer Gleichungssysteme* gebrochen werden können. Kennen wir das Bild von n linear unabhängigen Elementen unter der Verschlüsselungsfunktion, so kennen wir die Funktion zur Gänze (wenn die lineare Abbildung auf einem Vektorraum der Dimension n wirkt).

▷ **Erkenntnis:** Linearität der Chiffre ist gefährlich, wir benötigen *nichtlineare* Verfahren.

1.7 Vigenèrechiffre

Das Grundübel der Substitutionschiffre ist die Tatsache, dass jedem Klartextelement genau ein Geheimtextelement zugeordnet wird und diese Zuordnung während der Verschlüsselung konstant bleibt. Daher reicht eine recht einfache statistische Analyse aus, um solche Chiffren zu knacken.

Einen großen Fortschritt bei diesem Problem bedeutete die Vigenere-Chiffre, bei der die Häufigkeiten *verschleiert* werden.

1.25 Definition (Vigenère-Chiffre, ca. 16. Jahrhundert)
Für ein festes $m \in \mathbb{N}$ sei $\mathcal{P} = \mathcal{C} = \mathcal{K} = \mathbb{Z}_{26}^m$. Die Ver- und Entschlüsselungsfunktionen seien wie folgt definiert:

$$\begin{aligned} e_k(p) &= p + k = (p_1 + k_1, p_2 + k_2, \ldots, p_m + k_m), \\ d_k(c) &= c - k = (c_1 - k_1, c_2 - k_2, \ldots, c_m - k_m), \end{aligned}$$

wobei $p = (p_1, \ldots, p_m) \in \mathcal{P}$, $c = (c_1, \ldots, c_m) \in \mathcal{C}$ und $k = (k_1, \ldots, k_m) \in \mathcal{K}$. Das Tupel $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ heißt eine Vigenère-Chiffre.

Die Verschlüsselung wird in der Praxis mit Hilfe des Vigenère-Quadrates durchgeführt.

Dabei geht man wie folgt vor. Man schreibt als Erstes unter den Klartext den Schlüssel. Der Schlüssel wird sooft wiederholt, wie der Klartext lang ist.

a	n	d	e	n	v	o	r	s	t	a	n	d	d	e	r	d	e	u	...
m	o	n	e	y	m	o	n	e	y	m	o	n	e	y	m	o	n	e	...

0	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	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a
2	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b
3	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c
4	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d
5	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e
6	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f
7	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g
8	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h
9	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i
10	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j
11	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k
12	m	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l
13	n	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m
14	o	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n
15	p	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o
16	q	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p
17	r	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q
18	s	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r
19	t	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s
20	u	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t
21	v	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u
22	w	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v
23	x	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w
24	y	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x
25	z	a	b	c	d	e	f	g	h	i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y

Abbildung 1.11: Das Vigenère-Quadrat

Als nächstes wird der Klartext Zeichen für Zeichen verschlüsselt. Man sucht im Vigenère-Quadrat jene Spalte auf, die mit dem Klartextbuchstaben beginnt. Dann sucht man jene Zeile auf, die mit dem Schlüsselbuchstaben beginnt. Am Kreuzungspunkt von Zeile und Spalte findet man den zugeordneten Geheimtextbuchstaben.

In unserem Beispiel ergibt sich der folgende Geheimtext:

a	n	d	e	n	v	o	r	s	t	a	n	d	d	e	r	d	e	u	...
m	o	n	e	y	m	o	n	e	y	m	o	n	e	y	m	o	n	e	...
M	B	Q	I	L	H	C	E	W	R	M	B	Q	H	C	D	R	R	Y	...

1.26 Bemerkung

1. Die Vigenere-Chiffre ist eine **polyalphabetische** Chiffre: ein Buchstabe des Klartextes wird in verschiedene Geheimtextbuchstaben übergeführt. Damit wird die Häufigkeitsanalyse wesentlich erschwert.

   (In Gegensatz dazu wird bei einer **monoalphabetischen** Chiffre jedem Klartextbuchstaben ist genau ein Geheimtextbuchstabe zugeordnet. Ein typisches Beispiel für monoalphabetische Chiffren ist die Substitutionschiffre)

2. Wie wir an den Beispielen zur Vigenère-Chiffre sehen, werden die Strukturen im Klartext (wie Häufigkeiten, Muster, ...) umso besser zerstört, je

länger die Schlüsselphrase ist. In Zusammenhang mit Stromchiffren werden wir auf den Einfluß des Schlüsselwortes auf die Häufigkeitsverteilungen im Klartext zurückkommen (siehe Kapitel 1.10).

Wir untersuchen nun an einem Beispiel, wie sich die Häufigkeitsverteilung ändert, wenn wir mit dem (relativ kurzen aber doch recht "zufälligen") Schlüsselwort `hellekalek` einen längeren Text verschlüsseln.

Als Erstes die Häufigkeitsverteilung des Klartextes. Wir haben die Häufigkeiten alphabetisch geordnet, es ergibt sich folgende Graphik (Text: J. W. von Goethe, Iphigenie auf Tauris; siehe auch Abbildung 1.10):

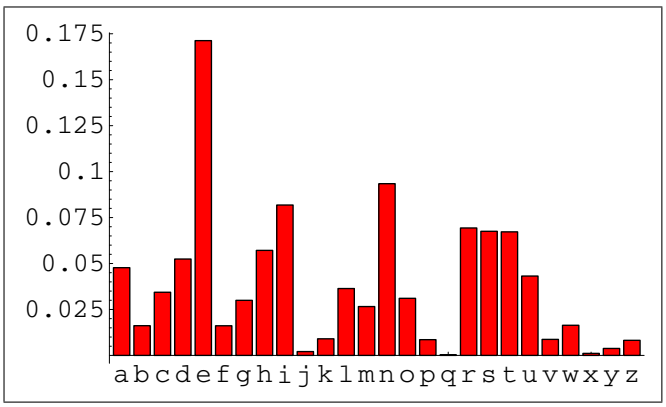


Abbildung 1.12: Buchstabenhäufigkeiten im Klartext

Im Vergleich dazu ändert sich die Häufigkeitsverteilung im Geheimtext dramatisch, siehe Abbildung 1.13.

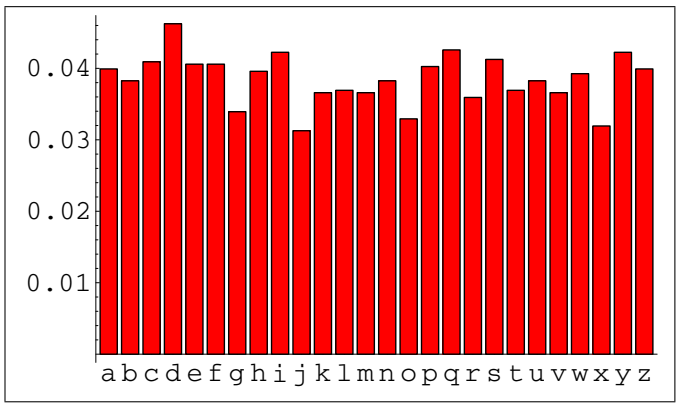


Abbildung 1.13: Buchstabenhäufigkeiten im Geheimtext

Die Kryptanalyse mittels Häufigkeitsverteilungen, die bei der Substitutionschiffre so hilfreich war, versagt hier. Dieser Effekt ist aber beabsichtigt! Nicht nur bei den einzelnen Geheimtextbuchstaben wurde die Häufigkeitsverteilung deutlich

sichtbar eingeebnet, sondern auch bei den Paaren und Tripeln im Geheimtext, siehe Abbildung 1.14.

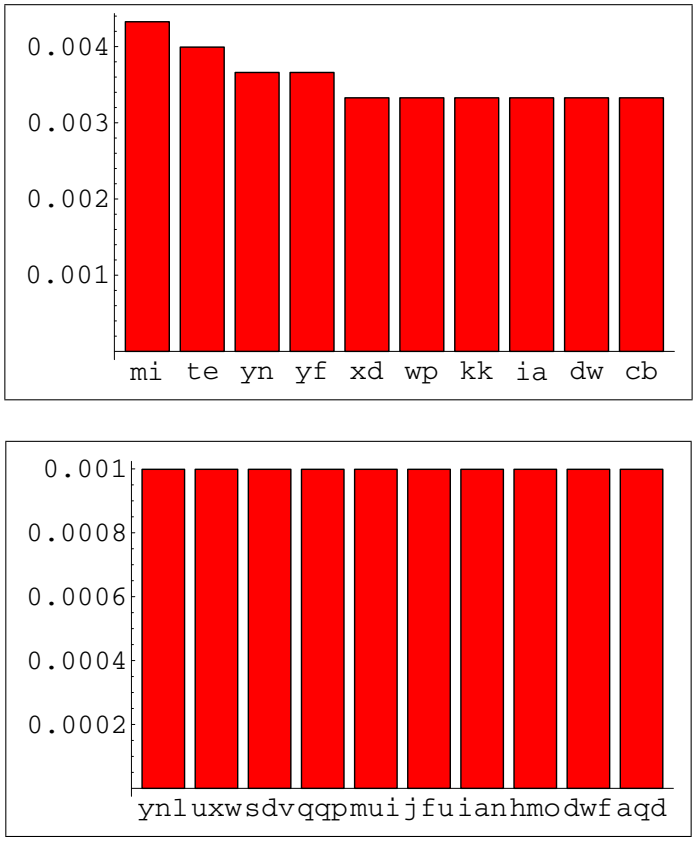


Abbildung 1.14: Paare und Tripel im Geheimtext

Es ist damit klar, dass die Vigenèrechiffre ihre Aufgabe, nämlich die Häufigkeiten einzuebnen, sehr gut erfüllt (siehe Abbildung 1.8 und 1.9 für die Verteilungen bei Klartexten).

In der Kryptoanalyse der Vigenèrechiffre gab es zwei wichtige Schritte, die Idee von Kasiski (1863), eine kombinatorische Überlegung, und das Konzept des Koinzidenzindex von Friedman (1920), ein sehr weitreichender statistischer Ansatz.

1.7.1 Der Kasiski-Test

Die Idee von Kasiski beruht auf der folgenden Beobachtung. Bei längeren Klartexten werden sich manche Buchstabenblöcke wiederholen. Wird ein kurzes Schlüsselwort verwendet, so kann es relativ leicht passieren, dass unter dem gleichen Buchstabenblock der gleiche Block des Schlüsselwortes zu liegen kommt. Man spricht in einem solchen Fall von *Parallelstellen* im Geheimtext. In einem solchen Fall gilt daher:

- die entsprechenden Geheimtextblöcke sind gleich.
- der Abstand zwischen diesen Parallelstellen ist ein Vielfaches der Länge des Schlüsselwortes.

In unserem Beispiel tritt dieses Phänomen ebenfalls auf:

a	n	d	e	n	v	o	r	s	t	a	n	d	d	e	r	d	e	u	...
m	o	n	e	y	m	o	n	e	y	m	o	n	e	y	m	o	n	e	...
M	B	Q	I	L	H	C	E	W	R	M	B	Q	H	C	D	R	R	Y	...

Der Klartextblock "`a n`" tritt im Abstand von 10 Zeichen wieder auf. Dies ist noch nichts Ungewöhnliches. Was aber die Attacke erleichtert, ist die Tatsache, dass sich der Schlüsselblock "`m o`" ebenfalls im Abstand von 10 Zeichen wiederholt. Daher sind die so entstehenden Geheimtextblöcke gleich.

Beim sogenannten Kasiskitest geht man wie folgt vor:

1. Man sucht mehrere Parallelstellen im Geheimtext und bestimmt deren Abstände. Auf diese Weise erhält man mehrere natürliche Zahlen.
2. Man bestimmt für jede dieser Zahlen die Primfaktorzerlegung.
3. Man berechnet den größten gemeinsamen Teiler d dieser Zahlen.

Mit etwas Glück ist die Länge des Schlüsselwortes ein Teiler von d. Erschwert wird diese Schätzung der Schlüsselwortlänge durch die Tatsache, dass Parallelstellen im Geheimtext auch zufällig entstehen und daher ihr Abstand nichts mit der Schlüsselwortlänge zu tun haben muss. Manchmal benötigt man also viel Glück für den Erfolg dieser Attacke (siehe dazu Bauer[Bau97] für illustrative Beispiele zum Kasiskitest).

Was ist damit gewonnen, wenn wir die Länge des Schlüsselwortes herausgefunden haben? Sehr viel, denn wenn L die Schlüsselwortlänge ist, dann sind die Buchstaben p_1, p_{L+1}, p_{2L+1}, ... des Klartextes alle mit demselben Buchstaben des Schlüsselwortes verschlüsselt worden. Mit anderen Worten, die Teilfolge c_1, c_{L+1}, c_{2L+1}, ... des Geheimtextes ist das Ergebnis einer Verschiebechiffre. Mit Hilfe einer Häufigkeitsanalyse ist der unbekannte Schlüsselbuchstabe dann zu bestimmen.

Mit Hilfe der Teilfolge c_2, c_{L+2}, c_{2L+2}, ... bestimmen wir dann den zweiten Buchstaben des Schlüssels, und so weiter, bis alle L Buchstaben des Schlüsselwortes bekannt sind. Damit können wir dann den Geheimtext leicht entschlüsseln.

1.7.2 Der Friedman-Test

Der Friedman-Test ist ein wesentlich grösserer Schritt in der Entwicklung der Kryptoanalyse als der Kasiski-Test. Er geht auf einen der berühmtesten Kryptoanalytiker der Geschichte zurück, den Amerikaner William F. Friedman.

1.27 Definition (Kappa-Wert)
Seien $a_0, \ldots, a_{m-1}$ die m Buchstaben des Alphabets einer bestimmten Sprache (Deutsch, Englisch, ...) und seien p_0, ..., p_{m-1} die zugehörigen relativen Häufigkeiten der m Buchstaben in dieser Sprache.

Dann versteht man unter dem Kappa-Wert (in Symbolen: κ) dieser Sprache die Zahl

$$\kappa \;=\; \sum_{i=0}^{m-1} p_i^2. \tag{1.1}$$

1.28 Beispiel
Die Kappa-Werte einiger Sprachen:

Sprache	♯ Buchstaben	κ
Arabisch	28	0.076
Deutsch	26	0.076
Englisch	26	0.066
Französisch	26	0.076
Russisch	33	0.056
Schwedisch	29	0.064

Interessanterweise variieren die in der Literatur angegebenen κ-Werte. Die Unterschiede beschränken sich auf die dritte Dezimalstelle. So wird in einigen Büchern der κ-Wert des Russischen mit 0.056, in anderen mit 0.053 angegeben.

1.29 Bemerkung
Wenn $m = 26$ Buchstaben gleichverteilt sind, so gilt für den zugehörigen κ-Wert:

$$\kappa \;=\; \sum_{i=0}^{25} \left(\frac{1}{26}\right)^2 \;=\; \frac{1}{26} \;\approx\; 0.038$$

1.30 Bemerkung
Es ist offensichtlich, dass der κ-Wert für m gleichverteilte Buchstaben gleich $1/m$ ist. Wir interessieren uns nun dafür, welche Werte der κ-Wert überhaupt annehmen kann.

1.31 Bemerkung (Interpretation des κ-Wertes)
Wir können uns den κ-Wert einer Sprache an Hand eines Urnenmodells veranschaulichen. Der κ-Wert gibt jene Wahrscheinlichkeit an, mit welcher wir beim zweimaligen Ziehen *mit Zurücklegen* zwei gleichfarbige Kugeln (d.h. zwei gleiche Buchstaben) aus einer *großen* Grundgesamtheit ziehen.

In Zusammenhang mit einer Quadratsumme der Gestalt, wie sie in der Definition des κ-Wertes auftritt, fragen wir nach oberen und unteren Schranken. Wenn wir eine obere und eine untere Schranke angeben können, dann interessieren wir uns für jene Fälle, in denen diese Schranken angenommen werden. Es gilt folgender Satz.

1.32 Satz
Sei $m \in \mathbb{N}$, $m \geq 2$, und sei $(p_0, p_1, \ldots, p_{m-1})$ eine Wahrscheinlichkeitsverteilung. Dann gilt:

1. Eine allgemeine Ungleichung

$$\frac{1}{m} \leq \sum_{i=0}^{m-1} p_i^2 \leq 1.$$

2. Die linke Schranke wird genau im Fall der Gleichverteilung $(1/m, \ldots, 1/m)$ angenommen.

3. Die rechte Schranke wird genau dann angenommen, wenn es sich um eine Punktverteilung $(0, \ldots, 0, 1, 0, \ldots, 0)$ handelt.

Beweis. Die Behauptungen folgen aus den nachstehenden Darstellungen für die Quadratsumme:

$$\begin{aligned}\sum_{i=0}^{m-1} p_i^2 &= \frac{1}{m} + \sum_{i=0}^{m-1} \left(p_i - \frac{1}{m}\right)^2 \\ &= \sum_{i=0}^{m-1} p_i - \sum_{i=0}^{m-1} p_i(1-p_i) = 1 - \sum_{i=0}^{m-1} p_i(1-p_i)\end{aligned}$$

□

Für einen gegebenen Geheimtext stellen sich zwei Fragen,

- Handelt es sich um eine monoalphabetische oder um eine polyalphabetische Verschlüsselung nach Vigenère?
- Wenn es sich um eine polyalphabetische Verschlüsselung nach Vigenère handelt, wie lange ist das Schlüsselwort?

Bei der Beantwortung beider Fragen kann uns der κ-Wert helfen. Sei der Geheimtext c gegeben und sei die Sprache, in welcher der Klartext p geschrieben wurde, bekannt.

Wenn es sich um eine monoalphabetische Verschlüsselung handelt, dann wird die Summe der relativen Häufigkeiten, mit denen die 26 Buchstabenpaare aa, bb, $\ldots zz$ im Geheimtext c auftreten, nahe beim κ-Wert der Sprache liegen.

Wenn es sich um eine polyalphabetische Verschlüsselung (nach Vigenère) handelt, dann wird die Summe der relativen Häufigkeiten, mit denen diese 26 Buchstabenpaare in c auftreten, vermutlich deutlich geringer als der κ-Wert der Sprache sein, da die polyalphabetische Verschlüsselung die Häufigkeitsverteilung der Buchstaben im Geheimtext c einebnet und wir daher in die Nähe der Gleichverteilung kommen.

Wir beachten bei dieser Argumentation, dass der κ-Wert der deutschen Sprache 0.076 beträgt und der κ-Wert der Gleichverteilung mit 0.038 wesentlich kleiner ist. Dieser deutliche Unterschied in den beiden κ-Werten ist die Grundlage für den Erfolg dieser Methode.

In der Sprache der mathematischen Statistik handelt es sich hier um einen Hypothesentest. Die Nullhypothese H_0 besagt, dass der Geheimtext mit einem

monoalphabetischen Verfahren erzeugt wurde. Die Alternativhypothese H_1 ist, dass ein polyalphabetisches Verfahren die Quelle des Geheimtextes ist. Wir berechnen aus dem Geheimtext einen Schätzwert. Je nach Größe des Schätzwertes akzeptieren wir H_0 oder lehnen H_0 ab.

Zur Durchführung dieses Hypothesentests benötigen wir eine geeignete Schätzfunktion. Dieser Punktschätzer soll ein erwartungstreuer Schätzer für den unbekannten Parameter κ sein. Weiters interessiert uns die Varianz dieses Schätzers.

Dieser Ansatz wurde von Friedman wie folgt präzisiert.

1.33 Definition (Koinzidenzindex)
Sei ein Text der Länge N über dem Alphabet $a_0, \ldots, a_{m-1}$ gegeben. Für jeden der m Buchstaben a_i des Alphabets bezeichne f_i, $0 \leq i \leq m-1$, die absolute Häufigkeit des i-ten Buchstabens im Text. Dann heißt die folgende Zahl der (Friedmansche-) Koinzidenzindex, auf Englisch "index of coincidence".

$$IC \;=\; \sum_{i=0}^{m-1} \frac{f_i}{N} \frac{f_i - 1}{N-1}. \tag{1.2}$$

Der folgende Satz gibt den Zusammenhang zwischen IC und κ an. Wir formulieren ihn für den allgemeinen Fall mit m Buchstaben.

1.34 Satz
Sei eine Sprache mit κ-Wert κ gegeben und sei der Vektor $(f_0, f_1, \ldots, f_{m-1})$ der absoluten Häufigkeiten der m Buchstaben in einem Text der Länge N multinomialverteilt,

$$(f_0, f_1, \ldots, f_{m-1}) \;\sim\; M_{N,(p_0,\ldots,p_{m-1})}$$

Unter diesen Voraussetzungen ist IC ein erwartungstreuer Schätzer für κ,

$$E[IC] \;=\; \kappa.$$

Beweis. Siehe die Diplomarbeit von Fischer[Fis96a]. □

Aus Zeitgründen werden wir nicht näher auf die konkrete Anwendung dieses Zusammenhangs eingehen. Nur so viel sei hier erwähnt: da sich auch die Varianz von IC angeben läßt, kann man zu vorgegebener Irrtumswahrscheinlichkeit den kritischen Bereich angeben und so eine numerische Bedingung für die Ablehnung der Nullhypothese formulieren.

1.35 Bemerkung Für die Bestimmung der Schlüsselwortlänge kann man wie folgt vorgehen. Man testet die Nullhypothese H_0, dass die Schlüsselwortlänge gleich 2 ist. Dazu berechnet man den IC für die Teilfolge $c^{(1)} = c_1, c_3, c_5, \ldots$ und für die zweite Teilfolge $c^{(2)} = c_2, c_4, c_6, \ldots$ des Geheimtextes. Wenn das Schlüsselwort tatsächlich Länge 2 hatte, dann werden diese beiden IC-Werte nahe beim κ-Wert der Sprache liegen. Wenn diese Nullhypothese verworfen werden muss, da die beiden κ-Werte zu klein sind, dann testet man für Länge 3. Man hat nun 3 κ-Werte zu berechnen. Liegen diese drei Werte nahe beim κ-Wert der Sprache, dann ist die Schlüsselwortlänge vermutlich 3 und man erhält mittels einfacher Frequenzanalyse der 3 Teilfolgen den jeweiligen Schlüsselwortbuchstaben

(siehe dazu den Abschnitt über den Kasiski-Test). Sind die 3 Werte zu klein, um die Nullhypothese "Schlüsselwortlänge=3" zu akzeptieren, dann führt man diese Analyse für die Länge 4 durch, und so fort, bis sich ein aktzeptabler Wert von IC für die Teilfolgen ergibt.

1.36 Bemerkung (Erinnerung: Multinomialverteilung)
Die Multinomialverteilung ist die Verallgemeinerung der Binomialverteilung. Anstelle von zwei möglichen Ergebnissen wie bei der Binomialverteilung werden in einem Zufallsexperiment m mögliche Ergebnisse betrachtet.

Denken Sie dabei an ein Urnenmodell. In der Urne seien m Sorten von Kugeln, der Anteil der Kugeln des Typs i sei p_i, $1 \leq i \leq m$.

Wir ziehen N-mal *mit Zurücklegen*. Der Urne wird also N-mal jeweils eine Kugel entnommen, deren Typ festgestellt und anschließend die Kugel wieder in die Urne zurückgelegt.

Wir interessieren uns für die Anzahl X_i der Kugeln eines jeden Typs i in dieser Stichprobe der Größe N. Die Zufallsvariablen X_i genügen der Multinomialverteilung. Für die Wahrscheinlichkeiten gilt:

$$P(X_1 = x_1 \wedge X_2 = x_2 \wedge \ldots \wedge X_m = x_m) = \frac{N!}{x_1! \cdot x_2! \cdot \ldots \cdot x_m!} p_1^{x_1} \cdot p_2^{x_2} \cdot \ldots \cdot p_m^{x_m}.$$

1.8 Hillchiffre

In diesem Kapitel besprechen wir eine Chiffre, die für die kryptographische Praxis zwar bedeutungslos war, jedoch viele klassischen Verfahren zu einem mathematisch sehr klaren und schönen Konzept zusammenfasst.

Dieses Verfahren geht auf Lester S. Hill (1929) zurück.

1.37 Bemerkung (Matrizen über Ringen)
Sei $(R, +, \cdot)$ ein kommutativer Ring mit Einselement. Typische Beispiele für solche Ringe sind der Restklassenring $\mathbb{Z}_m$ oder der Ring $\mathbb{Z}$ der ganzen Zahlen.

Sei $A = (a_{ij})$ eine $n \times n$-Matrix über R. Sei A_{ij} jene $(n-1) \times (n-1)$- Teilmatrix, die entsteht, wenn in A die i-te Zeile und die j-te Spalte gestrichen wird.

1.38 Bemerkung (Identität von Lagrange)
Für jedes $i \in \{1, \ldots, n\}$ gilt die folgende Identität:

$$\det A = \sum_{j=1}^{n} (-1)^{i+j} \cdot a_{ij} \cdot \det A_{ij}.$$

Diese Identität gilt in analoger Weise auch für jedes $j \in \{1, \ldots, n\}$.

Nach der Lagrangeschen Identität gilt insbesondere die bekannte Regel

$$\det \begin{pmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{pmatrix} = a_{11} \cdot a_{22} - a_{12} \cdot a_{21}.$$

Aus diesem Beispiel und aus der vorangegangenen Identität folgt:

$$\begin{aligned} &A \text{ eine } 2\times 2\text{-Matrix über } R \Rightarrow \det A \in R \\ &\vdots \\ &A \text{ eine } n\times n\text{-Matrix über } R \Rightarrow \det A \in R \end{aligned}$$

Bezeichne $M(n,R)$ die Menge der $n \times n$-Matrizen über dem Ring R und sei I die $n \times n$-Einheitsmatrix. Wir nennen $A \in M(n,R)$ invertierbar über R, wenn es in der Menge $M(n,R)$ eine Inverse zu A gibt:

$$A \cdot A^{-1} = A^{-1} \cdot A = I.$$

1.39 Bemerkung Aus der Identität von Lagrange (siehe Bemerkung 1.38) folgt

$$A \in M(n,R) \;\Rightarrow\; \det A \in R.$$

1.40 Bemerkung (Teilbarkeit in Ringen)
In einem Ring $(R,+,\cdot)$ wird die Teilbarkeit von Elementen wie im Ring $(\mathbb{Z},+,\cdot)$ definiert:

$$b \text{ teilt } a \Leftrightarrow \exists\, c \in R : a = b \cdot c \qquad (a,b,c \in R)$$

Bezeichnung: $b \mid a$
Das Ringelement a heißt eine Einheit, wenn $a \mid 1$. Dazu muß im Ring $(R,+,\cdot)$ natürlich ein Einselement existieren!

1.41 Proposition
Sei $(R,+,\cdot)$ ein kommutativer Ring mit Einselement und sei $A \in M(n,R)$. Dann gilt:

$$A \text{ invertierbar über } R \Leftrightarrow \det A \text{ ist eine Einheit von } R.$$

Beweis.
Sei A invertierbar über R.

$$\begin{aligned} &\Rightarrow \exists\, A^{-1} \in M(n,R) \text{ mit } A \cdot A^{-1} = I \\ &\Rightarrow \det A \cdot \underbrace{\det A^{-1}}_{\in R} = \det(A \cdot A^{-1}) = \det I = 1 \\ &\Rightarrow \det A \mid 1 \end{aligned}$$

Umkehrung: es sei nun $\det A \mid 1$ vorausgesetzt.
Die Adjunkte A^* (engl.: adjoint) zu A ist eine Matrix in $M(n,R)$, die folgendermaßen definiert ist:

$$A^* = \left((-1)^{i+j} \cdot \det A_{ji}\right).$$

Wegen der Voraussetzung $\det A \mid 1$ folgt $\det A \neq 0$. Für die Matrix

$$B := \frac{1}{\det A} \cdot A^*$$

gilt wegen $\det A \mid 1$, dass $B \in M(n,R)$. Weiters ist aus der linearen Algebra bekannt, dass die Matrix B die inverse Matrix zu A ist, $B = A^{-1}$. Somit liegt die Matrix A^{-1} in der Menge $M(n,R)$. Daher ist A invertierbar über R. $\square$

Wir haben in dieser Vorlesung mit dem speziellen Ring $(\mathbb{Z}_{26}, +, \cdot)$ gearbeitet. Wie sehen die Einheiten von $\mathbb{Z}_{26}$ aus?

1.42 Lemma Sei $A \in M(n, \mathbb{Z}_{26})$. Die Matrix A ist invertierbar über $\mathbb{Z}_{26}$ genau dann, wenn $\det A$ eine prime Restklasse ist, d.h. wenn $\det A \in \mathbb{Z}_{26}^*$.

(Zur Erinnerung: eine prime Restklasse modulo 26 ist eine Restklassen $\overline{a}$ mit der Eigenschaft $(a, 26) = 1$. Allgemein ist die prime Restklassengruppe modulo m definiert als die multiplikative Gruppe $\mathbb{Z}_m^* = \{\overline{a} : (a, m) = 1\}$. Sie besitzt $\varphi(m)$ Elemente.)

Beweis.

$\overline{a}$ Einheit von $\mathbb{Z}_{26}$
$\Leftrightarrow \overline{a} \mid \overline{1}$
$\Leftrightarrow$ die lineare Kongruenz $a \cdot x \equiv 1 \pmod{26}$ ist lösbar
$\Leftrightarrow$ (Lösbarkeitsbedingung!) $(a, 26) = 1$
$\Leftrightarrow \overline{a} \in \mathbb{Z}_{26}^*$.

Die Einheiten des Ringes $(\mathbb{Z}_{26}, +, \cdot)$ sind also genau die primen Restklassen modulo 26. □

1.43 Beispiel

$\begin{pmatrix} \overline{1} & \overline{0} \\ \overline{0} & \overline{2} \end{pmatrix}$ ist nicht invertierbar über $\mathbb{Z}_{26}$ (denn: $\det A = \overline{2}$, $(2, 26) \neq 1$)

$\begin{pmatrix} \overline{11} & \overline{0} \\ \overline{0} & \overline{3} \end{pmatrix}$ ist invertierbar über $\mathbb{Z}_{26}$ (denn: $\det A = \overline{33} = \overline{7}$, $(7, 26) = 1$)

1.44 Bemerkung (Matrizen modulo 26)
In der Praxis rechnen wir nicht mit Restklassen, sondern mit ganzen Zahlen modulo 26. Wann ist eine $n \times n$-Matrix A mit Elementen aus $\mathbb{Z}$ invertierbar modulo 26? Die Antwort lautet: wenn $(\det A, 26) = 1$. Die Begrünung ist einfach: anstatt mit A modulo 26 zu arbeiten, verwenden wir die zugehörige Matrix über $\mathbb{Z}_{26}$. So ist zum Beispiel

$$A = \begin{pmatrix} 12 & 3 \\ 3 & 2 \end{pmatrix}$$

invertierbar modulo 26 (da $\det A = 15$, $(15, 26) = 1$), aber die Matrix

$$B = \begin{pmatrix} 11 & 3 \\ 3 & 2 \end{pmatrix}$$

ist nicht invertierbar modulo 26 (da $\det B = 13$, $(13, 26) \neq 1$).

1.45 Definition (Hillchiffre, Lester S. Hill, 1929)
Sei $m \in \mathbb{Z}$, $m \geq 2$. Sei $\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}^m$ und sei

$$\mathcal{K} = \{A \in M(m, \mathbb{Z}_{26}) : A \text{ invertierbar über } \mathbb{Z}_{26}\} \; = \; GL(m, \mathbb{Z}_{26}).$$

Wir definieren die Verschlüsselungsfunktionen:

$$e_k(p) = k \cdot p = \begin{pmatrix} k_{11} & \dots & k_{1m} \\ \vdots & & \vdots \\ k_{m1} & \dots & k_{mm} \end{pmatrix} \cdot \begin{pmatrix} p_1 \\ \vdots \\ p_m \end{pmatrix},$$

die Entschlüsselungsfunktionen sind definiert als

$$d_k(c) = k^{-1} \cdot c = \begin{pmatrix} k_{11} & \dots & k_{1m} \\ \vdots & & \vdots \\ k_{m1} & \dots & k_{mm} \end{pmatrix}^{-1} \cdot \begin{pmatrix} c_1 \\ \vdots \\ c_m \end{pmatrix},$$

wobei k^{-1} die Inverse zu k in $\mathcal{K}$ ist.

1.46 Beispiel
Um ein Beispiel zu geben:

$$k = \begin{pmatrix} 12 & 3 \\ 3 & 2 \end{pmatrix} \quad \Rightarrow \quad k^{-1} = 15^{-1} \cdot \begin{pmatrix} 2 & -3 \\ -3 & 12 \end{pmatrix} \pmod{26}$$

Was ist das Inverse zu 15 modulo 26?

$$15 \cdot x \equiv 1 \pmod{26} \quad \text{bzw.} \quad 15 \cdot x + 26 \cdot y = 1$$

$$\begin{aligned} 26 &= 1 \cdot 15 + 11 & 1 &= 4 - 3 \\ 15 &= 1 \cdot 11 + 4 & &= 4 - (11 - 2 \cdot 4) \\ 11 &= 2 \cdot 4 + 3 & &= 4 \cdot 3 + 11 \cdot (-1) \\ 4 &= 1 \cdot 3 + 1 & &= (15 - 11) \cdot 3 + 11 \cdot (-1) \\ 3 &= 3 \cdot 1 & &= 15 \cdot 3 + 11 \cdot (-4) \\ & & &= 15 \cdot 3 + (26 - 15) \cdot (-4) \\ & & &= 15 \cdot 7 + 26 \cdot (-4) \end{aligned}$$

$$\Rightarrow x \equiv 7 \pmod{26}$$

$$\Rightarrow k^{-1} = 7 \cdot \begin{pmatrix} 2 & 23 \\ 23 & 12 \end{pmatrix} = \begin{pmatrix} 14 & 5 \\ 5 & 6 \end{pmatrix} \pmod{26}$$

1.47 Bemerkung (Schlüsselraum der Hill-Chiffre)
Die Frage nach der Größe des Schlüsselraums der Hill-Chiffre ist mathematisch anspruchsvoll. Offensichtlich gilt

$$|\mathcal{K}| = |GL(m, \mathbb{Z}_{26})| \; = \; \text{Anzahl der regulären } m \times m\text{-Matrizen über } \mathbb{Z}_{26}.$$

Für die Anzahl $g(N, m)$ der regulären $m \times m$-Matrizen über $\mathbb{Z}_N$ existieren folgende Resultate (siehe dazu Bauer[Bau97, Kap. 5.2]):

$N = p$ prim:

$$\begin{aligned} g(p, m) &= (p^m - 1) \cdot (p^m - p) \cdot (p^m - p^2) \cdot \ldots \cdot (p^m - p^{m-1}) \\ &= p^{m^2} \cdot \rho(p, m), \end{aligned}$$

mit

$$\rho(p,m) = \prod_{k=1}^{m} \left(1 - \frac{1}{p^k}\right).$$

Wir beachten in diesem Zusammenhang, dass die Zahl p^{m^2} die Anzahl aller möglichen $m \times m$-Matrizen über $\mathbb{Z}_p$ angibt.

Speziell gilt für $p = 2$:

$$g(2,1) = 1, \quad g(2,2) = 6, \quad g(2,3) = 168, \quad g(2,4) = 20160.$$

Sei nun $N = p^\alpha$ eine Primzahlpotenz. Dann gilt die Beziehung

$$g(p^\alpha, m) = g(p,m) \cdot (p^{\alpha-1})^{m^2} = (p^\alpha)^{m^2} \cdot \rho(p,m) = N^{m^2} \cdot \rho(p,m).$$

Sei nun $N = p_1^{\alpha_1} \cdot \ldots \cdot p_r^{\alpha_r}$ eine beliebige natürliche Zahl. Dann gilt

$$g(N,m) = N^{m^2} \cdot \rho(p_1,m) \cdot \rho(p_2,m) \cdot \ldots \cdot \rho(p_r,m).$$

Die Zahl $\rho(p,m)$ kann näherungsweise über eine lakunäre Potenzreihe berechnet werden (Euler 1760):

$$\lim_{m\to\infty} \rho(p,m) = h\left(\frac{1}{p}\right),$$

wobei

$$\begin{aligned} h(x) &= 1 + \sum_{k=1}^{\infty} (-1)^k \cdot \left(x^{(3k^2-k)/2} + x^{(3k^2+k)/2}\right) \\ &= 1 - x - x^2 + x^5 + x^7 - x^{12} - x^{-15} + x^{22} + x^{26} \ldots \end{aligned}$$

Für $p > 10$ ist bereits $1 - \frac{1}{p} - \frac{1}{p^2}$ eine gute Näherung für $h\left(\frac{1}{p}\right)$. Details zu dieser Darstellung sowie weiterführende Literaturhinweise finden sich in [Bau97, Kap. 5.2].

Für uns ist der Fall $N = 26$ interessant:

$$\begin{aligned} g(26,1) &= 12 \\ g(26,2) &= 157\ 248 \\ g(26,3) &= 1\ 634\ 038\ 189\ 056 \end{aligned}$$

Bauer[Bau97, Kap. 5.2] enthält eine Diskussion zur Konstruktion regulärer Matrizen über $\mathbb{Z}_N$. Der Autor merkt an, dass selbstreziproke Matrizen über $\mathbb{Z}_N$ (d.h. $A \cdot A = I$) nicht so einfach abzählbar sind wie die regulären Matrizen, für ihre Anzahl ist keine Formel bekannt wie im Fall der regulären Matrizen!

1.9 Transpositionschiffren

Wir stellen eine weitere Idee der klassischen Kryptographie vor: die Klartextzeichen werden nicht verändert, sie tauschen aber ihre Plätze.

1.48 Definition (Permutationschiffre, Transpositionschiffre)
Sei $\mathcal{P} = \mathcal{C} = Z_{26}^m$ und sei

$$\mathcal{K} = S_m = \text{Menge der Permutationen von } \{1, 2, \ldots, m\}.$$

Die Verschlüsselungsfunktionen seien definiert durch

$$e_\pi(p_1, \ldots, p_m) = (p_{\pi(1)}, \ldots, p_{\pi(m)}), \quad \pi \in S_m,$$

die Entschlüsselungsfunktion seien festgelegt durch

$$d_\pi(c_1, \ldots, c_m) = (c_{\pi^{-1}(1)}, \ldots, c_{\pi^{-1}(m)}), \quad \pi \in S_m.$$

Dann heißt das Chiffriersystem $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ eine Permutations- oder Transpositionschiffre.

1.49 Beispiel
Konkretes Beispiel: mit $m = 6$

$$\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 6 & 1 & 2 & 5 \end{pmatrix} \quad \Rightarrow \quad \pi^{-1} = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 4 & 5 & 1 & 2 & 6 & 3 \end{pmatrix}$$

Klartext	a	n	d	i	e	f	r	a	u	b	u	n	...
Geheimtext	D	I	F	A	N	E	U	B	N	R	A	U	...

1.50 Bemerkung
Die Permutationschiffre- bzw. Transpositions-Chiffre ist ein Spezialfall der Hillchiffre. Sei $\pi \in S_m$ der Schlüssel der Permutationschiffre mit $\mathcal{P} = \mathcal{C} = \mathbb{Z}_{26}^m$. Dann können wir π eine eindeutig bestimmte Matrix $A(\pi) \in M(m, \mathbb{Z}_2)$ zuordnen, bei der in jeder Zeile und jeder Spalte genau eine Eins steht (und sonst lauter Nullen):

$$A(\pi) := (a_{i,j})_{i,j=1}^m,$$

wobei $a_{i,j} = 1$ falls $\pi(j) = i$ und 0 sonst.

Zur Illustration ein konkretes Beispiel:

$$m = 6 \text{ und } \pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 \\ 3 & 4 & 6 & 1 & 2 & 5 \end{pmatrix}: \quad A(\pi) = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \end{pmatrix}$$

Allgemein gilt: die Matrix $A(\pi)$ entsteht aus der Einheitsmatrix $E \in M(m, \mathbb{Z}_2)$ durch Zeilen- und Spalten-Vertauschungen

$$\Rightarrow \det A(\pi) \in \{-1, 1\}$$
$$\Rightarrow A(\pi) \text{ ist regulär modulo } 26$$
$$\Rightarrow A(\pi) \text{ definiert eine Hill-Chiffre}$$

1.51 Definition (Permutationsmatrix)
Sei $A \in M(m, \mathbb{Z}_2)$ eine Matrix, die in jeder Zeile und jeder Spalte genau eine Eins enthält. Dann heißt A eine Permutationsmatrix.

Somit: Jedem Element $\pi \in S_m$ ist genau eine Permutationsmatrix zugeordnet. Umgekehrt ist leicht zu sehen, daß jeder Permutationsmatrix $A \in M(m, \mathbb{Z}_2)$ genau eine Permutation $\pi \in S_m$ mit $A(\pi) = A$ zugeordnet ist.

1.52 Lemma Es gilt folgender Zusammenhang:

$$\forall\, \pi \in S_m : A(\pi^{-1}) = A(\pi)^{-1}.$$

Beweis. Für den Zusammenhang mit Ver- und Entschlüsselungsfunktionen gilt nach Definition:

$$\begin{aligned} d_\pi(c_1, \ldots, c_m) &= e_{\pi^{-1}}(c_1, \ldots, c_m) \\ &= e_{\pi^{-1}}(e_\pi(p_1, \ldots, p_m)) \\ &= (p_1, \ldots, p_m) \end{aligned}$$

D.h.:

$$(p_1, \ldots, p_m) = e_{\pi^{-1}}(e_\pi(p_1, \ldots, p_m))$$

Weiters gilt:

$$e_\pi(p_1, \ldots, p_m)^T = A(\pi) \cdot (p_1, \ldots, p_m)^T$$

Aus diesen letzten beiden Identitäten schließen wir nun:

$$A(\pi^{-1}) \cdot A(\pi) = A(\pi) \cdot A(\pi^{-1}) = E$$

$$\Rightarrow \quad A(\pi^{-1}) = A(\pi)^{-1}.$$

Insbesondere ist $A(\pi)^{-1}$ wieder eine Permutationsmatrix. □

1.53 Bemerkung (Produktchiffren)
Die Arbeit eines Kryptanalytikers wird durch die *Kombination* verschiedenartiger Chiffren erschwert. Die Kombination von Chiffren nennt man eine **Produktchiffre**.

Sei zum Beispiel $\mathcal{P} = \mathcal{C} = \mathcal{K}_1 = \mathbb{Z}_2^8$ und $\mathcal{K}_2 = S_8$. Wir verschlüsseln zuerst mittels

$$e_k^{(1)}(p) = p + k \quad k \in \mathcal{K}_1$$

und dann mittels

$$e_\sigma^{(2)}(p'_1, \ldots, p'_8) = (p_{\sigma(1)}, \ldots, p_{\sigma(8)}) \quad \sigma \in \mathcal{K}_2$$

Sei zum Beispiel $k = (1\ 0\ 0\ 1\ 0\ 1\ 1\ 0)$ und $\pi = \begin{pmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 \\ 3 & 7 & 4 & 1 & 8 & 5 & 2 & 6 \end{pmatrix}$.
Für $p = (0\ 0\ 0\ 0\ 1\ 1\ 1\ 0)$ erhalten wir

$$e_k^{(1)}(p) = (1\ 0\ 0\ 1\ 1\ 0\ 0\ 0)$$

und

$$e_\pi^{(2)}(1\ 0\ 0\ 1\ 1\ 0\ 0\ 0) = (1\ 0\ 1\ 0\ 0\ 0\ 0\ 1).$$

1.10 Stromchiffren

1.54 Bemerkung (Block- und Stromchiffren)
Die Chiffrierverfahren werden in zwei Typen aufgeteilt, je nachdem, wie sie den Klartext verschlüsseln:

- *Blockchiffren*
  Sie fassen den Klartext zu Blöcken zusammen und verschlüsseln dann der Reihe nach Block für Block.
- *Stromchiffren*
  Sie verschlüsseln jedes Klartextelement einzeln, üblicherweise mit jeweils verschiedenen Schlüsseln.

Diese Unterscheidung ist oberflächlich und dient nur dazu, einen ersten Überblick zu schaffen. Tatsächlich ist es in der Praxis so, dass Blockchiffren auch als Stromchiffren eingesetzt werden können, je nach Betriebsmodus.

1.55 Definition (Stromchiffre)
Für eine endliche, nichtleere Menge A bezeichne A^* die Menge der endlichen Folgen mit Elementen aus A. Eine Stromchiffre ist ein Chiffriersystem $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ mit folgenden Eigenschaften:

1. $\mathcal{P} = \mathcal{C} = \mathcal{K} = A^*$
2. Für jedes $k \in A$ existieren zwei eindeutig bestimmte bijektive Funktionen $e_k : A \mapsto A$ und $d_k : A \mapsto A$ mit der Eigenschaft
$$d_k \circ e_k = e_k \circ d_k = \text{Identität}$$
3. Verschlüsselung
   Sei $p = (p_i)_{i\geq 1} \in \mathcal{P}$ und sei $k = (k_i)_{i\geq 1} \in \mathcal{K}$ mindestens so lange wie p. Dann definieren wir die Verschlüsselungsfunktion durch
$$e_k(p) = (e_{k_i}(p_i))_{i\geq 1}\,.$$
   Sei $c = (c_i)_{i\geq 1} \in \mathcal{C}$ und sei $k = (k_i)_{i\geq 1} \in \mathcal{K}$ mindestens so lange wie c. Dann definieren wir die Entschlüsselungsfunktion durch
$$d_k(c) = (d_{k_i}(c_i))_{i\geq 1}\,.$$

Die Folge $p = (p_i)_{i\geq 1}$ heißt der Klartextstrom, $k = (k_i)_{i\geq 1}$ heißt der Schlüsselstrom und $c = (c_i)_{i\geq 1}$ nennt man den Geheimtextstrom.

1.56 Definition (Synchrone Stromchiffre)
Eine Stromchiffre heißt synchron, wenn der Schlüsselstrom $(k_i)_{i\geq 1}$ unabhängig ist vom Klartextstrom. Eine Stromchiffre heißt periodisch, wenn der Schlüsselstrom $(k_i)_{i\geq 1}$ eine periodische Folge ist, wenn also eine Zahl $d \in \mathbb{N}$ existiert mit $k_{i+d} = k_i \quad \forall\, i \in \mathbb{N}$. Die kleinste natürliche Zahl d mit dieser Eigenschaft heißt die Periode des Schlüsselstroms.

1.57 Bemerkung
Wir merken an:

1. Wir können die Vigenèrechiffre als eine periodische Stromchiffre auffassen: Sei $k = (k_1, \ldots, k_m) \in \mathbb{Z}_{26}^m$ das Schlüsselwort. Dann setzen wir $A = \mathbb{Z}_{26}$ und betrachten als Schlüsselstrom die periodische Folge

$$k_1, k_2, \ldots, k_m, k_1, k_2, \ldots, k_m, k_1, k_2, \ldots$$

2. Für moderne Stromchiffren verwendet man üblicherweise $A = \mathbb{Z}_2$.

Wir untersuchen nun den Einfluß, den die Länge und die Art des Schlüsselwortes auf die Häufigkeitsverteilungen im Vigenèreverfahren hat.

Zunächst verschlüsseln wir mit einem kurzen Schlüsselwort (siehe Abbildungen 1.15 und 1.16). Als Klartext wurde "Der zerbrochene Krug" von Kleist verwendet.

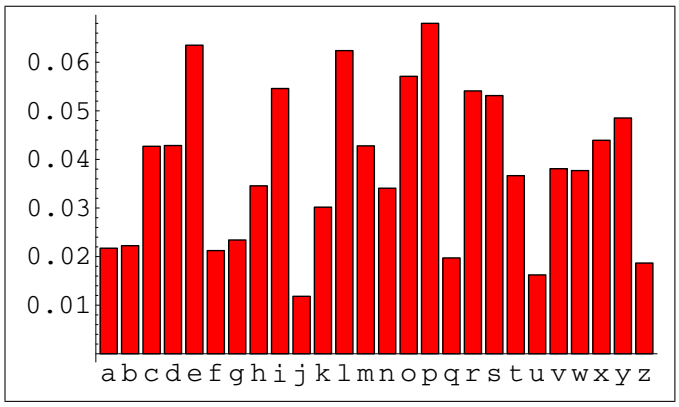


Abbildung 1.15: Vigenèrechiffre, Schlüsselwort: `hellekalek`

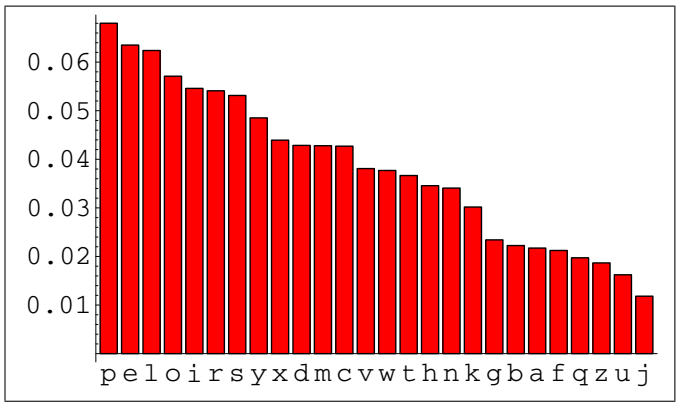


Abbildung 1.16: Vigenèrechiffre, Schlüsselwort: `hellekalek`

Der gleiche Text wird dann mit einem langen Schlüsselwort (50 zufällige Zeichen) verschlüsselt. Die Verteilung ändert sich auffallend (siehe Abbildungen 1.17 und 1.18).

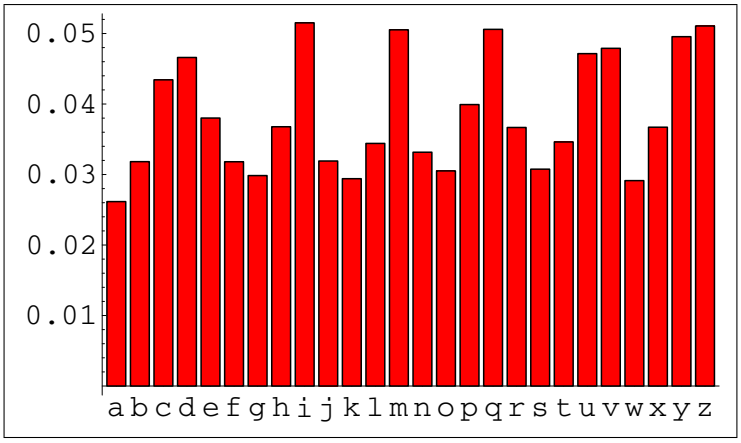


Abbildung 1.17: Vigenèrechiffre, Schlüsselwort mit 50 Zeichen

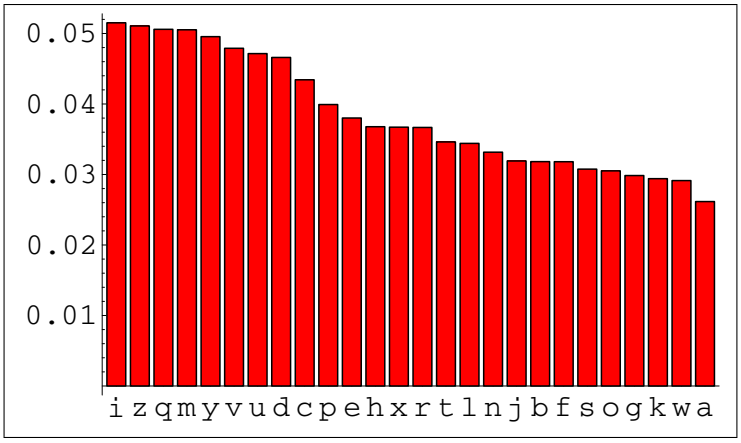


Abbildung 1.18: Vigenèrechiffre, Schlüsselwort mit 50 Zeichen

1.58 Bemerkung
Wir haben das folgende Phänomen beobachtet: je länger und je "zufälliger" das Schlüsselwort ist, desto mehr nähert sich die Verteilung der Häufigkeiten der Gleichverteilung an. Umso schwieriger wird es dann, mit der Methode der Häufigkeitsanalyse dieses Chiffrierverfahren zu attackieren.

Im Folgenden bezeichne $\oplus$ die Addition auf $\mathbb{Z}_2$, also das wohlbekannte `XOR` in der Informatik.

1.59 Definition (Vernam-Chiffre, 1917)
Sei $A = \mathbb{Z}_2$. Die Vernam-Chiffre ist eine Stromchiffre mit der Verschlüsselungsfunktion

$$e_k(p) = (p_i \oplus k_i)_{i \geq 1}$$

und der Entschlüsselungsfunktion

$$d_k(c) = (c_i \oplus k_i)_{i \geq 1},$$

mit $p = (p_i)_{i \geq 1} \in \mathcal{P}$, $c = (c_i)_{i \geq 1} \in \mathcal{C}$ und $k = (k_i)_{i \geq 1} \in \mathcal{K}$.

1.60 Definition (one-time pad)
Ein one-time pad ist eine Vernamchiffre, in der der Schlüsselstrom eine Zufallsbitfolge ist.

1.61 Bemerkung
Eine Zufallsbitfolge ist die Realisierung einer Folge unabhängiger, auf $\mathbb{Z}_2$ identisch gleichverteilter Zufallsvariablen. Denken Sie dabei an das Ergebnis einer langen Reihe von Münzwürfen mit einer gerechten Münze.

1.62 Proposition
Seien P und K zwei diskrete Zufallsvariable mit Werten in $\{0,1\}$. Sei $\mathbf{Pr}(P = 0) = \alpha$, $\mathbf{Pr}(P = 1) = 1 - \alpha$, $0 \leq \alpha \leq 1$, die Verteilung von P und sei K gleichverteilt auf $\{0,1\}$, in Symbolen: $K \sim U(\{0,1\})$.

Wenn P und K unabhängig sind, dann gilt

$$P \ \oplus \ K \ \sim \ U(\{0,1\}).$$

Beweis. Es gilt wegen der Unabhängigkeit:

$$\begin{aligned} \mathbf{Pr}(P \oplus K = 0) \ &= \ \mathbf{Pr}(P = 0, K = 0) \ + \ \mathbf{Pr}(P = 1, K = 1) \\ &= \ \alpha\frac{1}{2} \ + \ (1 - \alpha)\frac{1}{2} \\ &= \ \frac{1}{2}. \end{aligned}$$

□

1.63 Korollar
Das one-time pad gleicht ein Bias des Klartextes aus.

Wie arbeitet man mit one-time pads in der Praxis? Prinzipiell gibt es zwei Möglichkeiten, Zufallsbits und (über die Binärentwicklung) Zufallszahlen zu erzeugen:

- Hardwaregeneratoren
  Verwenden elektronisches Rauschen, radioaktive Zerfallsprozesse, ...
  ▷ Link: `www.mils.com`
  ▷ Link: `www.protego.com`

- Softwaregeneratoren
  Sie verwenden deterministische Algorithmen, um aus einer kurzen Folge (dem seed) eine lange Folge von deterministisch erzeugten (Zufalls-)Zahlen zu produzieren. Man spricht in diesem Fall von Pseudozufallszahlengeneratoren.
  ▷ Link: `random.mat.sbg.ac.at`

Ein typisches Beispiel eines Hardwaregenerators ist der SG100, der elektronisches Rauschen verwendet, um Zufallsbits zu erzeugen (siehe Abbildung 1.19).

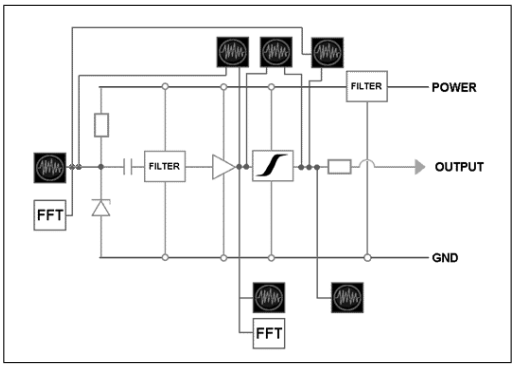


Abbildung 1.19: SG100 (Protego SA, Sweden)

Hardwaregeneratoren sind langsam, schwer portierbar, theoretisch nicht analysierbar und ihr Ausgabestrom ist unvorhersagbar. In Gegensatz dazu sind Softwaregeneratoren (in den meisten Fällen) sehr schnell, portierbar, in manchen Fällen theoretisch analysierbar und der Ausgabestrom ist reproduzierbar. Je nach Anwendung kann die Unvorhersagbarkeit oder die Reproduzierbarkeit ein Vor- oder ein Nachteil sein.

1.11 Einige Konzepte von Shannon

1.64 Bemerkung (Shannons Forderungen)
Der berühmte Mathematiker Claude E. Shannon, einer der Begründer der Informationstheorie, hat sich eingehend mit der theoretischen und praktischen Analyse von Chiffriersystemen beschäftigt (siehe [Sha49]). Shannon hat für ein Chiffriersystem die folgenden Forderungen aufgestellt. Sie haben bis heute ihre Bedeutung behalten:

1. *Sicherheit*
   Ein Chiffriersystem wäre perfekt, wenn auch noch so viele abgefangene Geheimtexte keinen Aufschluß über den verwendeten Schlüssel geben.
   (In der Praxis können wir diesen theoretischen Grad von Sicherheit nicht erreichen. Wir werden aber auf jeden Fall anstreben, daß zu einem abgefangenen Geheimtext der Aufwand zur Bestimmung des verwendeten Schlüssels und damit des Klartextes möglichst hoch ist.)

2. *Schlüssellänge*
   Der verwendete Schlüssel sollte möglichst kurz sein, damit er leicht zu übersenden ist.
   (Hinweis: Durch die Public-Key Verfahren hat sich die Situation grundlegend verändert)

3. *Arbeitsaufwand für Ver- und Entschlüsselung*
   Die Komplexität dieser Operationen (entspricht der Rechenzeit) sollte möglichst gering sein.

4. *Fehlerfortpflanzung*
   Sie sollte möglichst gering sein.
   (Heutzutage löst man diese Aufgabe mit speziellen Betriebsmodi, wie etwa bei den Blockchiffren, und auch mit Methoden der Kodierungstheorie, siehe [Hil86, TW06, Wel88].)

5. *Expansion der Nachrichten*
   Sie sollte möglichst gering sein.
   (Um die Häufigkeitsverteilungen im Klartext zu verschleiern, wird oft auf Füllzeichen zurückgegriffen. Diese blähen die Nachrichten jedoch auf. Je länger die Nachricht ist, desto aufwändiger wird dann die Übertragung.)

In der Praxis ist es nicht möglich, alle diese Forderungen gleichzeitig zu erfüllen. Ein in diesem Sinne perfektes System gibt es nicht.

1.65 Bemerkung (Konfusion und Diffusion)
Ebenfalls von Claude Shannon stammen zwei berühmte Konzepte, die bis heute die wesentlichen Designkriterien für Chiffrierverfahren darstellen:

- *Konfusion*
  Der Zusammenhang zwischen dem Klartext, dem Schlüssel und dem Geheimtext sei so komplex wie nur irgendwie möglich beschaffen.

- *Diffusion*
  Jedes Geheimtextzeichen soll von möglichst vielen Klartextzeichen und möglichst vom gesamten Schlüssel abhängen.

1.11.1 Perfekte Sicherheit

Sei $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ ein gegebenes Verschlüsselungsverfahren.

Das Sicherheitskonzept von Shannon beruht auf folgender, leicht verständlicher Forderung für eine Chiffre: die Kenntnis des Geheimtextes soll keinen Aufschluss über den dahinterliegenden Klartext liefern. Die *Ungewissheit* für den Gegner, welcher Klartext P verwendet wurde, soll sich nicht vermindern, wenn der Geheimtext C bekannt ist.

Zum Studium dieser Frage verwendete Shannon die Sprache der Stochastik. Seien im Folgenden P, C und K Zufallsvariable. Die Werte von P sollen in der Menge $\mathcal{P}$ liegen, die Werte von C in $\mathcal{C}$, und die Werte von K in $\mathcal{K}$. Gegeben sei weiters eine Wahrscheinlichkeitsverteilung $\{\mathbf{Pr}(P = p) : \ p \in \mathcal{P}\}$ auf dem Klartextraum $\mathcal{P}$. Diese Verteilung heißt die *a-priori* Verteilung der Klartexte.

Weiters sei eine Wahrscheinlichkeitsverteilung $\{\mathbf{Pr}(K = k) : \ k \in \mathcal{K}\}$ auf dem Schlüsselraum $\mathcal{K}$ gegeben.

1.66 Bemerkung (Annahme I)
Wir nehmen im Folgenden an, dass die Zufallsvariablen P und K *unabhängig* sind. Es gilt also:

$$\mathbf{Pr}(P = p, \ K = k) \ = \ \mathbf{Pr}(P = p) \cdot \mathbf{Pr}(K = k) \quad \forall\, p \in \mathcal{P}, \forall\, k \in \mathcal{K}.$$

Diese Annahme ist durch die kryptographische Praxis gerechtfertigt, da zuerst der geheime Schlüssel k vereinbart wird und dann erst, zu einem späteren Zeitpunkt, die Klartexte damit verschlüsselt werden.

Die Wahrscheinlichkeitsverteilungen auf $\mathcal{P}$ und auf $\mathcal{K}$ induzieren eine Wahrscheinlichkeitsverteilung auf dem Geheimtextraum $\mathcal{C}$:

$$\begin{aligned} \mathbf{Pr}(C = c) \ &= \sum_{(p,k):\, e_k(p)=c} \mathbf{Pr}(P = p, \ K = k) \qquad (1.3) \\ &= \sum_{(p,k):\, e_k(p)=c} \mathbf{Pr}(P = p) \cdot \mathbf{Pr}(K = k). \end{aligned}$$

1.67 Bemerkung (Annahme II)
Wir entfernen aus dem Mengen $\mathcal{P}, \mathcal{C}$ und $\mathcal{K}$ alle Elemente mit Wahrscheinlichkeit Null und setzen daher im Folgenden die Bedingungen

$$\begin{aligned} \mathbf{Pr}(P = p) \ &> \ 0 \quad \forall\, p \in \mathcal{P}, \\ \mathbf{Pr}(C = c) \ &> \ 0 \quad \forall\, c \in \mathcal{C}, \\ \mathbf{Pr}(K = k) \ &> \ 0 \quad \forall\, k \in \mathcal{K}, \end{aligned}$$

voraus. Diese Bedingungen sind gerechtfertigt. Es macht keinen Sinn, Klartexte p oder Schlüssel k zu betrachten, die mit Wahrscheinlichkeit 0 auftreten. Ebenso unnötig ist es Geheimtexte c zu betrachten, für welche die Menge $\{(p, k) : \ e_k(p) = c\}$ leer ist, für die daher wegen Identität 1.3 $\mathbf{Pr}(C = c) = 0$ gilt.

1.68 Definition (Perfekte Sicherheit)
Die Chiffre $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ heißt perfekt sicher, wenn Annahme I und II erfüllt sind und wenn gilt:

$$\mathbf{Pr}(P = p \mid C = c) \ = \ \mathbf{Pr}(P = p) \qquad \forall\, p \in \mathcal{P}, \ \forall\, c \in \mathcal{C}. \qquad (1.4)$$

Die perfekte Sicherheit eines Kryptosystems bedeutet, dass die Kenntnis des Geheimtextes keinerlei Information über den möglichen Klartext liefert. Nach Kenntnis des Geheimtextes sind wir so "klug" wie zuvor, es gab keinen Informationsgewinn. Mit anderen Worten, die *a-posteriori* Wahrscheinlichkeiten $\mathbf{Pr}(P = p \mid C = c)$ sind gleich den a-priori-Wahrscheinlichkeiten.

1.69 Bemerkung (Entropie und bedingte Entropie)
Die *Entropie* ist ein Maß für die Ungewissheit des Ausgangs eines Zufallsexperiments. Mit Hilfe des Begriffs der *bedingten Entropie* kann man ebenfalls

perfekte Sicherheit definieren. Es läßt sich leicht zeigen, dass beide Definitionen äquivalent sind.

1.70 Lemma
Für die bedingte Wahrscheinlichkeit $\mathbf{Pr}(P = p \mid C = c)$ gilt folgende Identität:

$$\mathbf{Pr}(P = p \mid C = c) \;=\; \frac{\mathbf{Pr}(P = p) \cdot \sum_{k:\, e_k(p)=c} \mathbf{Pr}(K = k)}{\sum_{(q,k):\, e_k(q)=c} \mathbf{Pr}(P = q)\, \mathbf{Pr}(K = k)} \tag{1.5}$$

Beweis. Wir schreiben die bedingte Wahrscheinlichkeit $\mathbf{Pr}(P = p \mid C = c)$ wie folgt um:

$$\begin{aligned} \mathbf{Pr}(P = p \mid C = c) \;&=\; \frac{\mathbf{Pr}(P = p,\; C = c)}{\mathbf{Pr}(C = c)} \\ &=\; \frac{\mathbf{Pr}(C = c \mid P = p) \cdot \mathbf{Pr}(P = p)}{\mathbf{Pr}(C = c)} \end{aligned}$$

Die bedingte Wahrscheinlichkeit $\mathbf{Pr}(C = c \mid P = p)$ gibt die Wahrscheinlichkeit an, mit der p in c verschlüsselt wird. Da der "Startpunkt" p und der "Endpunkt" c der Verschlüsselung gegeben sind, können wir nur den Schlüssel variieren. Daher gilt:

$$\mathbf{Pr}(C = c \mid P = p) \;=\; \sum_{k:\, e_k(p)=c} \mathbf{Pr}(K = k). \tag{1.6}$$

Daraus und mit Hilfe von (1.3) folgt die behauptete Identität (1.5). □

1.71 Lemma
Sei die Chiffre $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ perfekt sicher. Dann gilt:

$$\forall\, p \in \mathcal{P},\; \forall\, c \in \mathcal{C}: \quad \exists k \in \mathcal{K}:\; e_k(p) = c \text{ (und } \mathbf{Pr}(K = k) > 0).$$

Beweis. Nach Definition der bedingten Wahrscheinlichkeit gilt:

$$\begin{aligned} \mathbf{Pr}(P = p \mid C = c) \cdot \mathbf{Pr}(C = c) \;&=\; \mathbf{Pr}(P = p,\; C = c) \\ &=\; \mathbf{Pr}(C = c \mid P = p) \cdot \mathbf{Pr}(P = p). \end{aligned}$$

Aus der perfekten Sicherheit folgt für alle $p \in \mathcal{P}$ und für alle $c \in \mathcal{C}$ die Gleichung

$$\mathbf{Pr}(P = p \mid C = c) \cdot \mathbf{Pr}(C = c) \;=\; \mathbf{Pr}(P = p) \cdot \mathbf{Pr}(C = c).$$

Daher gilt die folgende Äquivalenz:

$$\begin{aligned} \mathbf{Pr}(P = p \mid C = c) \;&=\; \mathbf{Pr}(P = p) \\ \Leftrightarrow \mathbf{Pr}(C = c \mid P = p) \;&=\; \mathbf{Pr}(C = c). \end{aligned}$$

Wegen $\mathbf{Pr}(C = c) > 0$ gilt daher $\mathbf{Pr}(C = c \mid P = p) > 0$. Aus der Gleichung (1.6) folgt die Existenz eines $k \in \mathcal{K}$ mit den Eigenschaften $e_k(p) = c$ und $\mathbf{Pr}(K = k) > 0$. □

1.72 Lemma
Es gelten die folgenden Beziehungen zwischen der Größe von Klartext-, Geheimtext- und Schlüsselraum.

1. Für jede Chiffre $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ gilt die Ungleichung
$$|\mathcal{P}| \ \leq \ |\mathcal{C}|.$$
2. Wenn die Chiffre $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ perfekt sicher ist, dann gilt *zusätzlich* die Ungleichung
$$|\mathcal{C}| \ \leq \ |\mathcal{K}|.$$

Beweis. Zu 1.
Für jedes $k \in \mathcal{K}$ ist die Abbildung $e_k : \mathcal{P} \to \mathcal{C}$, $p \mapsto e_k(p)$ nach Voraussetzung injektiv (siehe Definition 1.4). Aus der Injektivität von e_k folgt $|\mathcal{P}| = |e_k(\mathcal{P})|$. Aus $e_k(\mathcal{P}) \subseteq \mathcal{C}$ folgt $|e_k(\mathcal{P})| \leq |\mathcal{C}|$ und daraus die Behauptung.

Zu 2.
Wir erinnern an die Voraussetzung $\mathbf{Pr}(C = c) > 0 \ \forall \ c \in \mathcal{C}$. Aus Lemma 1.71 wissen wir, dass zu beliebigem p und c ein $k \in \mathcal{K}$ existiert mit $e_k(p) = c$. Somit gilt
$$\mathcal{K}(p, c) \ := \ \{k \in \mathcal{K} : \ e_k(p) = c\} \ \neq \ \emptyset. \tag{1.7}$$
Mit anderen Worten,
$$|\mathcal{K}(p, c)| \ \geq \ 1.$$
Weiters gilt, dass bei festem $p \in \mathcal{P}$ und $c, c' \in \mathcal{C}$, $c \neq c'$, die beiden zugehörigen Schlüsselmengen $\mathcal{K}(p, c)$ und $\mathcal{K}(p, c')$ disjunkt sind,
$$\mathcal{K}(p, c) \cap \mathcal{K}(p, c') \ = \ \emptyset, \quad \text{falls } c \neq c'. \tag{1.8}$$
Diese Behauptung ist leicht einzusehen, mittels eines indirekten Beweises. Angenommen, $k \in \mathcal{K}(p, c) \cap \mathcal{K}(p, c')$. Dann gilt nach Definition der beiden Schlüsselmengen $c \ = \ e_k(p) \ = \ c'$, also ein Widerspruch zur Voraussetzung $c \neq c'$.

Wir halten nun $p \in \mathcal{P}$ fest und lassen c die Menge $\mathcal{C}$ durchlaufen. Wegen $|\mathcal{K}(p, c)| \geq 1$ und der paarweisen Disjunktheit dieser Mengen folgt
$$|\mathcal{C}| \ \leq \ \sum_{c \in \mathcal{C}} |\mathcal{K}(p, c)| \ = \ |\bigcup_{c \in \mathcal{C}} \mathcal{K}(p, c)|. \tag{1.9}$$
Wegen $\bigcup_{c \in \mathcal{C}} \mathcal{K}(p, c) \subseteq \mathcal{K}$ folgt
$$|\mathcal{C}| \ \leq \ |\mathcal{K}|.$$
□

1.73 Korollar
Für eine perfekt sichere Chiffre $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ gilt
$$|\mathcal{P}| \ \leq \ |\mathcal{C}| \ \leq \ |\mathcal{K}|.$$

1.74 Satz
Sei $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ eine Chiffre mit der Eigenschaft $|\mathcal{K}| = |\mathcal{C}| = |\mathcal{P}|$. Dann ist diese Chiffre genau dann perfekt sicher, wenn die folgenden beiden Bedingungen erfüllt sind:

1. $\forall\, p \in \mathcal{P}$, $\forall\, c \in \mathcal{C}$ gilt:

$$\exists\, k \in \mathcal{K},\ k \text{ eindeutig:} \quad e_k(p) \ = \ c.$$

   (Zu jedem Paar $(p, c) \in \mathcal{P} \times \mathcal{C}$ existiert ein eindeutig bestimmter Schlüssel k, der p in c überführt.)

2. $\forall\, k \in \mathcal{K}$:

$$\mathbf{Pr}(K = k) \ = \ \frac{1}{|\mathcal{K}|}.$$

   (Jeder Schlüssel ist gleich wahrscheinlich.)

Beweis. Sei $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ perfekt. Nach Lemma 1.71 existiert zu jedem $p \in \mathcal{P}$ und jedem $c \in \mathcal{C}$ ein Schlüssel $k \in \mathcal{K}$ mit $e_k(p) = c$. Wir behaupten, dass dieser Schlüssel *eindeutig bestimmt* ist. In der Bezeichnung von Lemma 1.72 heißt dies, dass wir die Behauptung

$$|\mathcal{K}(p, c)| \ = \ 1$$

zeigen müssen.

Sei $p \in \mathcal{P}$ fest. Wenn ein $c^* \in \mathcal{C}$ existiert mit $|\mathcal{K}(p, c^*)| > 1$, dann folgt aus Ungleichung (1.9), dass gilt:

$$|\mathcal{C}| \ < \ \sum_{c \in \mathcal{C}} |\mathcal{K}(p, c)| \ = \ |\bigcup_{c \in \mathcal{C}} \mathcal{K}(p, c)|. \tag{1.10}$$

Wegen der trivialen Ungleichung $|\bigcup_{c \in \mathcal{C}} \mathcal{K}(p, c)| \leq |\mathcal{K}|$ folgt $|\mathcal{C}| \ < \ |\mathcal{K}|$. Dies ist ein Widerspruch zur Voraussetzung.

Somit gilt $|\mathcal{K}(p, c)| = 1$, $\forall\, p \in \mathcal{P}$, $\forall\, c \in \mathcal{C}$. Wir bezeichnen den durch $p \in \mathcal{P}$ und $c \in \mathcal{C}$ eindeutig bestimmten Schlüssel $k \in \mathcal{K}(p, c)$ mit $k = \psi_p(c)$.

Sei $p \in \mathcal{P}$ fest. Dann ist die Abbildung

$$\begin{aligned} \psi_p : \ \mathcal{C} \ &\to \ \mathcal{K}, \\ c \ &\mapsto \ \psi_p(c) \end{aligned} \tag{1.11}$$

bijektiv.

Aus der perfekten Sicherheit folgt, wie wir im Beweis von Lemma 1.71 gesehen haben, die Beziehung

$$\mathbf{Pr}(C = c \mid P = p) \ = \ \mathbf{Pr}(C = c).$$

Aus der Identität (1.6) und der soeben bewiesenen Tatsache, dass zu gegebenem p und c genau ein Schlüssel k mit $e_k(p) = c$ existiert, nämlich $k = \psi_p(c)$, folgt

$$\begin{aligned} \mathbf{Pr}(C = c \mid P = p) \ &= \ \sum_{k:\, e_k(p) = c} \mathbf{Pr}(K = k) \\ &= \ \mathbf{Pr}(K = \psi_p(c)). \end{aligned}$$

Somit gilt:

$$\forall\, p \in \mathcal{P}, \forall\, c \in \mathcal{C}: \quad \mathbf{Pr}(K = \psi_p(c)) \;=\; \mathbf{Pr}(C = c). \tag{1.12}$$

Wir behaupten: wenn wir c festhalten und wenn p die Menge $\mathcal{P}$ durchläuft, dann durchläuft $\psi_p(c)$ den gesamten Schlüsselraum $\mathcal{K}$.

Dies ist leicht einzusehen: zu gegebenem c und p existiert genau ein $k \in \mathcal{K}$ mit $e_k(p) = c$. Wenn $p, p' \in \mathcal{P}$, $p \neq p'$, dann sind die zugehörigen Schlüssel k und k' mit $e_k(p) = c = e_{k'}(p')$ verschieden, denn die Funktionen e_k, $k \in \mathcal{K}$, sind nach Definition injektiv. Daher ist der Fall $e_k(p) = e_k(p')$, $p \neq p'$, unmöglich und es muss $k = \psi_p(c) \neq k' = \psi_{p'}(c)$ gelten.

Gleichung (1.12) besagt somit, dass gilt:

$$\forall\, k \in \mathcal{K}: \quad \mathbf{Pr}(K = k) \;=\; \mathbf{Pr}(C = c). \tag{1.13}$$

Daraus folgt zunächst, dass zwei beliebige Schlüssel k und k' gleich wahrscheinlich sind, denn $\mathbf{Pr}(C = c)$ ist eine Konstante.

Es folgt aber noch mehr. Die Verteilung $\{\mathbf{Pr}(K = k):\ k \in \mathcal{K}\}$ ist eine Wahrscheinlichkeitsverteilung. Daher muss gelten

$$\mathbf{Pr}(K = k) \;=\; \frac{1}{|\mathcal{K}|}.$$

Somit ist eine Richtung des Satzes bewiesen.

Die Umkehrung: seien die beiden Bedingungen 1. und 2. erfüllt. Nach Identität (1.6) folgt

$$\forall\, p \in \mathcal{P},\ \forall\, c \in \mathcal{C}: \quad \mathbf{Pr}(C = c \mid P = p) \;=\; \mathbf{Pr}(K = \psi_p(c)) \;=\; \frac{1}{|\mathcal{K}|},$$

wobei wir den durch p und c eindeutig bestimmten Schlüssel k mit der Eigenschaft $e_k(p) = c$ wieder mit dem Symbol $\psi_p(c)$ bezeichnet haben. Weiters gilt in der Identität (1.3) wegen der Eindeutigkeit von k mit $e_k(p) = c$ und der Gleichverteilung $\mathbf{Pr}(K = k) = 1/|\mathcal{K}|$ folgende Gleichungskette:

$$\begin{aligned} \mathbf{Pr}(C = c) \;&=\; \sum_{(p,k):\, e_k(p) = c} \mathbf{Pr}(P = p) \cdot \mathbf{Pr}(K = k) \\ &=\; \frac{1}{|\mathcal{K}|} \cdot \sum_{p \in \mathcal{P}} \mathbf{Pr}(P = p) \\ &=\; \frac{1}{|\mathcal{K}|}, \qquad \forall\, c \in \mathcal{C}. \end{aligned} \tag{1.14}$$

Wir haben hier verwendet, dass $\{\mathbf{Pr}(P = p):\ p \in \mathcal{P}\}$ eine Wahrscheinlichkeitsverteilung ist, dass also $\sum_{p \in \mathcal{P}} \mathbf{Pr}(P = p) = 1$ gilt.

Aus der Gleichung

$$\begin{aligned} \mathbf{Pr}(P = p \mid C = c) \;&=\; \frac{\mathbf{Pr}(P = p,\ C = c)}{\mathbf{Pr}(C = c)} \\ &=\; \frac{\mathbf{Pr}(C = c \mid P = p) \cdot \mathbf{Pr}(P = p)}{\mathbf{Pr}(C = c)} \\ &=\; \frac{1/|\mathcal{K}| \cdot \mathbf{Pr}(P = p)}{1/|\mathcal{K}|} \;=\; \mathbf{Pr}(P = p). \end{aligned}$$

folgt die gesuchte Gleichheit $\mathbf{Pr}(P = p \mid C = c) = \mathbf{Pr}(P = p)$, also die perfekte Sicherheit. □

1.75 Korollar
Sei $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ eine perfekt sichere Chiffre mit der Eigenschaft $|\mathcal{K}| = |\mathcal{C}| = |\mathcal{P}|$. Aus der Identität (1.13) folgt, dass nicht nur die Schlüssel k, sondern auch die Geheimtexte c gleichverteilt sind,

$$\mathbf{Pr}(C = c) \;=\; \frac{1}{|\mathcal{C}|}.$$

1.76 Bemerkung (Erinnerung)
Sei $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ ein one-time pad, also eine Stromchiffre mit Alphabet $A = \mathbb{Z}_2 = \{0, 1\}$, $\mathcal{P} = \mathcal{C} = \mathcal{K} = A^*$ und der Eigenschaft, dass jeder Schlüsselstrom $k \in \mathcal{K}$ eine Zufallsbitfolge ist.

1.77 Definition
Unter einem one-time pad der Länge n verstehen wir ein one-time pad mit $|p| = |k| = n$ für alle $p \in \mathcal{P}$ und alle $k \in \mathcal{K}$.

1.78 Satz
Sei $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ ein one-time pad der Länge n. Dann ist dieses one-time pad perfekt sicher:

$$\mathbf{Pr}(P = p \mid C = c) \;=\; \mathbf{Pr}(P = p) \qquad \forall\, p \in \mathcal{P}, \forall\, c \in \mathcal{C}.$$

Beweis. Für jedes one-time pad der Länge n gilt die Eigenschaft $\mathcal{P} = \mathcal{K} = \mathcal{C} = \mathbb{Z}_2^n$. Es folgt $|\mathcal{P}| = |\mathcal{K}| = |\mathcal{C}| = 2^n$. Weiters sind die folgenden Bedingungen erfüllt:

1. Zu jedem $p \in \mathcal{P}$ und jedem $c \in \mathcal{C}$ existiert genau ein Schlüsselstrom $k \in \mathcal{K}$ mit $e_k(p) = c$.
2. Die Schlüsselströme sind gleichverteilt,

$$P(K = k) \;=\; \frac{1}{2^n}.$$

Nach Satz 1.74 folgt aus diesen Eigenschaften die perfekte Sicherheit dieser Chiffre. □

1.79 Bemerkung
Wir sollten uns angesichts dieses "Sicherheitsbeweises" zwei Aspekte des one-time pad vor Augen halten:

1. Der Nachweis der perfekten Sicherheit beruht auf der Annahme, dass die Schlüsselströme Zufallsbitfolgen sind. Diese Annahme ist in der kryptographischen Praxis nicht gerechtfertigt. Es existieren zwar theoretische Konzepte, um den Begriff einer "Zufallsbitfolge der Länge n" zu definieren,

diese Konzepte sind für die kryptographische Praxis aber leider nicht umsetzbar (Stichwort: *Kolmogoroff-Komplexität*, siehe [Cal94]). Unabhängig von der Art der Erzeugung ist es für eine gegebene Bitfolge nicht möglich nachzuweisen, dass es sich um eine Zufallsbitfolge handelt. Der Sicherheitsbeweis für das one-time pad ist daher *rein theoretischer Natur*.

2. Das one-time pad erfordert Schlüssel k, die gleich lang sind wie die Nachrichten p, die verschickt werden sollen. Für die kryptographische Praxis bedeutet dies einen sehr großen Aufwand bei der *Schlüsselverwaltung*, also beim sicheren Austausch der Schlüssel zwischen berechtigten Benutzern und bei der sicheren Verwahrung der Schlüssel beim jeweiligen Benutzer.

1.80 Bemerkung
Durch die Quantenkryptographie könnten one-time pads stark an Bedeutung gewinnen, siehe für weitere Informationen

▷ Link: `http://www.quantiki.org/`

1.12 Zufallszahlen

Ob bei Lehman Bros. in London die Preisentwicklung von Optionen, am CERN in Genf Modelle der Atomphysik oder an der TH Darmstadt Kläranlagen simuliert werden, ob diverse diplomatische Dienste den Schlüsselvorrat für hochgeheime Nachrichten mittels der Vernam-Chiffre erzeugen oder Sie bei PGP Ihren privaten und Ihren öffentlichen Schlüssel generieren, stets ist ein scheinbar recht simples und harmloses mathematisches Werkzeug im Einsatz, sogenannte *Zufallszahlengeneratoren*.

Es gibt zwei große Einsatzgebiete für Zufallszahlengeneratoren, erstens die *Monte Carlo Methode*, sie wird zum Gebiet der *stochastischen Simulation* gezählt, und zweitens die Kryptographie.

Standardwerke zur Monte Carlo Methode sind die Monographien von Sobol' [Sob83] sowie Fishman [Fis96b]. Online-Dokumente finden sich unter

▷ Link: `http://csep1.phy.ornl.gov/CSEP/TEXTOC.html`

sowie

▷ Link: `http://www.ncsa.uiuc.edu/Apps/SPRNG/`

Für Zufallszahlen in der Kryptographie ist die Webadresse

▷ Link: `http://random.mat.sbg.ac.at/links/crypto.html`

ein geeigneter Startpunkt.

Das Ziel für den Einsatz von Zufallszahlen in der Monte Carlo Methode lautet, "möglichst gute Näherungen" für den unbekannten Parameter zu finden, also möglichst gute "Schätzwerte" im Sinne der Statistik.

In der Kryptographie geht es darum, sogenannte "sichere" Bitströme zu erzeugen, zum Beispiel für one-time pads.

In beiden Anwendungsfällen müssen wir klären, wie wir die vagen Bezeichnungen "möglichst gute Näherung" oder "sicherer Bitstrom" mathematisch definieren können.

Was sind nun Zufallszahlen oder Zufallsbits? In beiden Fällen meint man damit, dass die generierten Folgen $(x_n)_{n\geq 0}$ in $[0,1[$ (beziehungsweise $\{0,1\}$) als Realisierungen einer Folge von unabhängigen, identisch gleichverteilten Zufallsvariablen angesehen werden können.

Wann sieht man eine Folge $(x_n)_{n\geq 0}$ als solch eine Realisierung an? Sicherlich dann, wenn sie zum Beispiel durch das (sehr häufige) Werfen einer fairen Münze entstanden ist. Aus Zeitgründen wirft man in der Praxis nicht Münzen, sondern man startet entweder einen Hardware- oder Softwaregenerator. Bei den Hardwaregeneratoren verwendet man einen Zufallsprozess, um sogenannte "echte" Zufallszahlen zu erzeugen. Beispiele sind das Diodenrauschen oder radioaktive Zerfallsprozesse. Der Begriff *echter* Zufallszahlen ist natürlich schwammig, aber sehr weit verbreitet. Wir werden später noch darauf eingehen.

Bei Softwaregeneratoren, den sogenannten *Pseudozufallszahlengeneratoren*, handelt es sich um deterministische Algorithmen, die auf dem Computer, also einer Maschine mit endlich vielen Zuständen, ablaufen und deren Ausgabe die gewünschten Zufallszahlen sind. Sie erzeugen aus einem kurzen Bitstring, dem sogenannten Startwert (E: seed, initial value) eine lange Folge von Zufallszahlen oder Zufallsbits.

1.81 Definition (Pseudozufallszahlengenerator)
Unter einem Pseudozufallszahlengenerator verstehen wir ein Tupel

$$\mathcal{G} = (\mathcal{S}, T, \mathcal{O}, g, s_0)\,,$$

wobei $\mathcal{S}$ und $\mathcal{O}$ endliche Mengen sind, $T : \mathcal{S} \to \mathcal{S}$ und $g : \mathcal{S} \to \mathcal{O}$ zwei Funktionen und $s_0 \in \mathcal{S}$ ein festes Element ist. Die Menge $\mathcal{S}$ heißt der Zustandsraum, die Menge $\mathcal{O}$ heißt der Ausgaberaum, die Funktion T heißt die Übergangsfunktion und g nennt man die Ausgabefunktion des Generators $\mathcal{G}$.

Der Generator $\mathcal{G}$ erzeugt Zufallszahlen beziehungsweise Zufallsbits nach dem folgenden Schema:

1. Wir wählen den Startwert s_0 im Zustandsraum $\mathcal{S}$.
   In der Kryptographie ist es üblich, den Startwert zufällig zu wählen. In der Monte Carlo Methode arbeitet man mit fixen Startwerten, um die Reproduzierbarkeit der Zufallszahlen zu gewährleisten. Man will auf diese Weise eine zusätzliche Kontrolle über die Berechnungen erhalten.

2. Wir erzeugen durch Iteration der Übergangsfunktion T eine Folge $(s_n)_{n\geq 0}$ von Zuständen:
   $$s_n \;=\; T^n(s_0), \qquad n \geq 0.$$

3. Wir erzeugen die Zufallszahlen mit Hilfe der Ausgabefunktion g:
   $$x_n \;=\; g(s_n), \qquad n \geq 0.$$

Es ist zu beachten, dass die Folge $(x_n)_{n\geq 0}$ vom gewählten Startwert s_0 abhängt. Die Elemente dieser Folge heißen (Pseudo-) Zufallszahlen zum Pseudozufallszahlengenerator $\mathcal{G} = (\mathcal{S}, T, \mathcal{O}, g, s_0)$.

1.82 Beispiel
Wir betrachten ein einfaches Beispiel zur Illustration der Begriffe. Die dabei verwendeten Parameterwerte sind für die Praxis viel zu klein, es geht hier nur um die Erläuterung des Konzeptes.

Sei $\mathbb{Z}_7$ der Ring der Restklassen modulo 7. Wir identifizieren $\mathbb{Z}_7$ mit der Menge $\{0, 1, 2, 3, 4, 5, 6\}$, wobei diese Elemente nach den Regeln von $\mathbb{Z}_7$ addiert und multipliziert werden.

$\mathcal{S}$	$\mathbb{Z}_7 = \{0, 1, 2, 3, 4, 5, 6\}$
T	$T(s) = 3 \cdot s$
$\mathcal{O}$	$[0, 1[$
g	$g(s) = s/7$
s_0	$s_0 = 2$

Der Generator $\mathcal{G} = (\mathcal{S}, T, \mathcal{O}, g, s_0)$ heißt ein multiplikativer linearer Kongruenzgenerator mit Modul 7, Multiplikator 3 und Startwert $s_0 = 2$.

Wir untersuchen nun die Folge $(s_n)_{n\geq 0}$ der Zustände zum Startwert $s_0 = 2$:

$$\boxed{s_0 = 2}\,,\ s_1 = 6,\ s_2 = 4,\ s_3 = 5,\ s_4 = 1,\ s_5 = 3,\ \boxed{s_6 = 2}\,,\ \ldots$$

Die Folge der Zustände ist offensichtlich periodisch mit Periode 6.

Wenn wir als Multiplikator das Element 2 wählen, dann ergibt sich eine andere Folge von Zuständen zum gleichen Startwert s_0:

$$\boxed{s_0 = 2}\,,\ s_1 = 4,\ s_2 = 1,\ \boxed{s_3 = 2}\,,\ldots$$

Wieder ist die Folge der Zustände periodisch, aber mit einer wesentlich kürzeren Periode, nämlich 3. Dahinter steckt natürlich ein zahlentheoretisches Prinzip. Sei a der Multiplikator des Generators. Dann gilt

$$s_n = T^n(s_0) \ = \ a \cdot T^{n-1}(s_0) \ = \ a^n \cdot s_0, \qquad n \geq 0.$$

Die Ordnung des Elementes a in der multiplikativen Gruppe $\mathbb{Z}_7^*$ bestimmt offensichtlich die Länge der Periode der Folge $(s_n)_{n\geq 0}$.

Dies war ein Baby-Generator. Wir haben aber mit seiner Hilfe ein Prinzip erkannt und verallgemeinern nun.

1.83 Beispiel
Sei $m \geq 2$ eine (sehr große) ganze Zahl. Typische Werte für m sind in der Praxis $m = 2^{16}, 2^{31} - 1, 2^{48}, \ldots$ Sei $\mathbb{Z}_m$ der Ring der Restklassen modulo m. Wir identifizieren $\mathbb{Z}_m$ mit der Menge $\{0, 1, \ldots, m-1\}$, wobei diese Elemente nach den Regeln von $\mathbb{Z}_m$ addiert und multipliziert werden. Seien $a \not\equiv 0 \pmod{m}$ und b zwei ganze Zahlen.

$\mathcal{S}$	$\mathbb{Z}_m = \{0, 1, \ldots, m-1\}$
T	$T(s) = a \cdot s + b$
$\mathcal{O}$	$[0, 1[$
g	$g(s) = s/m$
s_0	$s_0 \in \mathbb{Z}_m$, fest

Der Generator $\mathcal{G} = (\mathcal{S}, T, \mathcal{O}, g, s_0)$ heißt ein **linearer Kongruenzgenerator** mit Modul m, Multiplikator 2, additivem Term b und Startwert s_0. Er wird in der Monte Carlo-Literatur meist in der Form LCG(m, a, b, s_0) geschrieben.

Wie im Trivial-Beispiel von $\mathbb{Z}_7$, so entscheidet auch hier die Ordnung von a über die Periodenlänge, siehe Niederreiter [Nie92] für eine genaue Analyse der zahlentheoretischen Bedingungen.

Der LCG ist *der klassische* Pseudozufallszahlengenerator. Wie steht es ganz allgemein um die Periodenlänge eines beliebigen Pseudozufallszahlengenerators $\mathcal{G} = (\mathcal{S}, T, \mathcal{O}, g, s_0)$?

Wir weisen an dieser Stelle auf die Tatsache hin, dass es nur bei relativ einfachen Übergangsfunktionen T möglich ist, konkrete Aussagen über die Periodenlänge für alle möglichen Startwerte s_0 zu machen. Gerade bei kryptographischen Generatoren ist die exakte Analyse der Periodenlänge meist nicht möglich. Dies wird im nächsten Kapitel am Beispiel von AES deutlich werden.

1.84 Satz Sei $\mathcal{G} = (\mathcal{S}, T, \mathcal{O}, g, s_0)$ ein Pseudozufallszahlengenerator. Dann ist für jeden Startwert $s_0 \in \mathcal{S}$ die Zufallszahlenfolge $(x_n)_{n \geq 0}$ eine periodische Folge.

Beweis. Die Menge $\mathcal{S}$ ist endlich. Nach dem Schubfachprinzip existieren zwei Zahlen n_0 und t, $n_0 \geq 0$, $t > 0$, in der Weise, dass

$$T^n(s_0) \;=\; T^{n+t}(s_0) \qquad \forall\, n \geq n_0.$$

Daher ist die Folge $(s_n)_{n \geq 0}$ periodisch. Als Konsequenz ist auch die Folge $(x_n)_{n \geq 0}$, $x_n = g(s_n)$, periodisch. □

Diese unumgängliche Tatsache scheint es auf den ersten Blick aussichtslos zu machen, auf dem Computer brauchbare Zufallszahlen zu erzeugen. Die Lösung des Problemes mit der Periodizität ist aber einfach. Man verwendet Generatoren, bei denen die Periodenlänge so groß ist, dass auch die anspruchvollsten Anwendungen nur einen kleinen Bruchteil der Periode ausschöpfen. Die derzeit besten Generatoren für die Monte Carlo Methode haben eine Periodenlänge über 2^{128}.

Wie erzeugt man nun konkret Zufallszahlen auf dem Computer? Alle Programmierumgebungen und Programmbibliotheken bieten dafür Routinen an, zum Beispiel die NAG Library der Numerical Algorithms Group in Oxford:

▷ Link: `http://www.nag.co.uk/numeric.html`

die CERNLIB des Cern in Genf:

▷ Link: `http://wwwinfo.cern.ch/asd/index.html`

das RANPACK des Pittsburgh Supercomputing Centre:

▷ Link: `http://www.psc.edu/general/software/packages/ranpack/`

und die Linksammlung von David Wagner (Berkeley):

▷ Link: `http://www.cs.berkeley.edu/$\sim$daw/rnd/index.html`

Hinter diesen Programmen zur Erzeugung von Zufallszahlen verbergen sich wie bereits gesagt nicht physikalische Prozesse, sondern deterministische Algorithmen.

Wie erzeugt man nun *unvorhersagbare* Zahlen mit einem deterministischen Algorithmus, bei dem die Startwerte bereits eindeutig alle weiteren Zufallszahlen festlegen? Bei Kenntnis der Startwerte und Kenntnis des Algorithmus sind die so erzeugten Zufallszahlen ja jederzeit reproduzierbar.

Eine Lösung, die zum Beispiel im GSM-Bereich gewählt wurde, besteht in der *Geheimhaltung des Verfahrens*. Derartige geheime Algorithmen bleiben der Öffentlichkeit bestenfalls einige Monate verborgen. Diese Vorgangsweise ist in der Kryptographie seit dem 19. Jahrhundert als völlig sinnlos bekannt und es ist erstaunlich, wie oft dieser alte Fehler immer wieder neu begangen wird, selbst von sehr großen Firmen oder Organisationen. Es muß nämlich immer davon ausgegangen werden, dass der Feind das verwendete Verfahren kennt. Die Sicherheit darf nicht auf der Geheimhaltung des Verfahrens beruhen, sondern auf der Geheimhaltung des Schlüssels. Dies ist das sogenannte **Kerkhoffsche Prinzip**.

Wir müssen also realistischerweise voraussetzen, dass der Feind den Algorithmus kennt, in unserem Fall also zumindest den Typ unseres Zufallszahlengenerators. Damit bleibt uns nur mehr die *Geheimhaltung der Startwerte*.

Zur Sicherheitsanalyse von Zufallszahlengeneratoren stellt sich sofort eine Frage: warum schränkt man sich hier auf die Praxis ein und verzichtet auf eine theoretische Analyse? Eine Bedingung wie " Zufallszahlen sollten weder theoretisch noch praktisch von Realisierungen ... (siehe oben) zu unterscheiden sein" wäre doch naheliegend. Derartige "Definitionen" von kryptographisch sicheren Zufallszahlen finden sich in zahlreichen Publikationen zu diesem Thema. Tatsache ist, dass für die in der Praxis verwendeten Generatoren ein theoretischer Nachweis der "Zufälligkeit" der erzeugten Zahlen nicht durchführbar ist. Leider zählen fast alle kryptographischen Zufallszahlengeneratoren zu dieser Gruppe. Wir verfügen heute nicht über die Methoden, bestimmte Algorithmen (wie zum Beispiel AES) bezüglich Zufälligkeit theoretisch zu untersuchen. Außerdem ist anzumerken, dass ungeklärt ist, was man unter "zufällig" verstehen soll. Ein geflügeltes Wort in diesem Gebiet lautet

"Randomness is in the eye of the beholder".

Die Situation ist zwar ernst, aber nicht hoffnungslos. Sie gleicht der Situation in allen anderen angewandten Disziplinen. Selbst wenn eine theoretische (Korrelations-)Analyse für manche Generatoren nicht möglich ist, so bleibt stets noch die empirische Korrelationsanalyse, das *statistische Testen*. Dieser empirischen Analyse liegt die berechtigte Hoffnung zugrunde, dass (wenn es um stochastische Simulation geht) gut gewählte Tests viele Simulationsprobleme der

numerischen Praxis repräsentieren, und andererseits (wenn es um die kryptographische Sicherheit geht) diese Tests sehr zuverlässig Alarm geben werden, wenn statistisch aufällige Strukturen innerhalb der erzeugten Zufallszahlen vorliegen. Das Erkennen von (mathematischen) Strukturen in den Daten bedeutet ja das Erkennen von Korrelationen. Dies wäre bereits ein möglicher Ansatzpunkt für das Brechen des Chiffrierverfahrens, das diese Zufallszahlen verwendet. Eine der früheren PGP-Versionen hatte das Problem, dass bei für die Erzeugung der Primzahlen, die man zur Generierung des Schlüsselpaares benötigt, ein schlechter Zufallszahlengenerator verwendet wurde. Stellen Sie sich das Problem vor, das sich ergibt, wenn unter den vielen möglichen Primzahlen mit L Stellen (z.B. $L = 500$) bei dieser (Pseudo-) Zufallsauswahl nur immer einige wenige ausgewählt werden. Derart katastrophal war das Problem bei PGP zwar nicht, aber es schränkte die Auswahl der Kandidaten doch so stark ein, dass es ein Sicherheitsproblem bedeutete.

Wir kehren nun zum linearen Kongruenzgenerator zurück und führen an ihm einige Phänomene vor. Der LCG ist das wichtigste Beispiel eines Zufallszahlengenerators für die Monte Carlo Methode, also für die stochastische Simulation auf dem Computer. An ihm läßt sich sehr gut demonstrieren, wie sich auf deterministische Weise sehr gute und, falls man nicht vorsichtig ist, sehr schlechte Zufallszahlen erzeugen lassen.

Die übliche Art, einen LCG zu definieren, sieht wie folgt aus. Ein LCG wird durch eine Rekurrenz erster Ordnung definiert:

$$y_{n+1} \equiv ay_n + b \pmod{m}, \qquad n \geq 0,$$

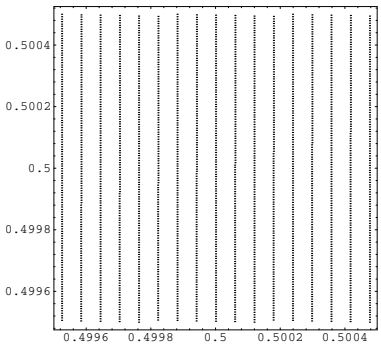
wobei die Parameter m, a, b, und y_0 nichtnegative ganze Zahlen sind. Dieser Generator wird dann mit LCG(m, a, b, y_0) bezeichnet. Die zahlentheoretische Beziehung zwischen dem Modul m, dem Multiplikator a, dem additiven Term b und dem Startwert y_0 entscheidet über die Länge der Periode der Folge $(y_n)_{n\geq 0}$ ganzer Zahlen und über die statistische Qualität der Zufallszahlen $x_n := y_n/m \in [0, 1[$. LCGs verhalten sich extrem sensibel gegenüber der Wahl der Parameter a und m. Typisch für diesen *linearen* Typ von Generator sind die *linearen* Punktstrukturen, die erzeugt werden. Wenn man zum Beispiel alle möglichen Vektoren der Gestalt

$$\mathbf{x}_n := (x_n, x_{n+1}, \ldots, x_{n+d-1}) \in [0, 1[^d, \quad n \geq 0, \tag{1.15}$$

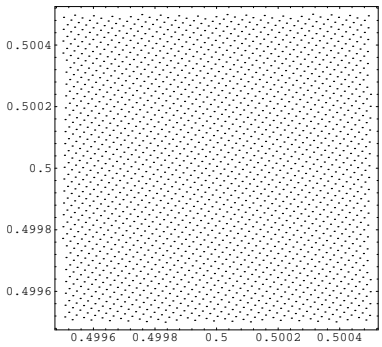
betrachtet ("überlappende" d-Tupel), dann erhält man auf diese Weise sehr regelmäßig verteilte Punkte im d-dimensionalen Einheitswürfel, wie die folgenden beiden Beispiele weit verbreiteter Generatoren in Dimension $d = 2$ zeigen (siehe Abbildung 1.20). Deutlich erkennbar ist der große Einfluß der Parameter auf die Punktverteilung.

Im nächsten Beispiel studieren wir die Verteilung einer Stichprobe von $N = 2^{15}$ Punkten von "Randu", dem Generator LCG(2^{31},65539,0,1) in Dimension zwei und drei, wobei die d-Tupel dieses Mal nach der Vorschrift $\mathbf{x}_n := (x_{nd}, x_{nd+1}, \ldots, x_{nd+d-1}) \in [0, 1[^d, n \geq 0$, ("nichtüberlappende" Tupel) gebildet werden (siehe Abbildung 1.21). Während die Stichprobe in Dimension zwei zumindest für das Auge unauffällig erscheint, enthüllt die Darstellung in Dimension drei die starken Korrelationen zwischen den Zufallszahlen[1].

[1] Diese beiden Graphiken stammen aus Hellekalek [Hel98a]



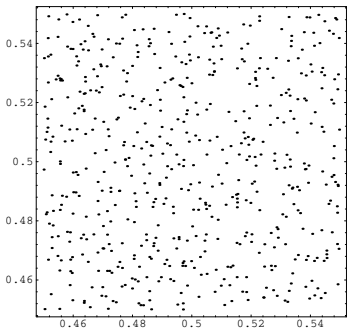
LCG($2^{31}-1$,16807,0,1)
Minimal Standard



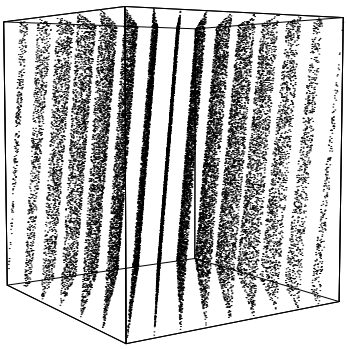
LCG(2^{31},1103515245,12345,1)
ANSI C

Abbildung 1.20: Zwei LCGs

Trotz eindrücklicher Warnungen, zum Beispiel in der "Bibel" der Zufallszahlen Knuth[Knu98], hält sich dieser Generator weiterhin hartnäckig in der Lehrbuchliteratur und damit auch in der Praxis.



LCG(2^{31},65539,0,1)
Zoom in das Einheitsintervall



LCG(2^{31},65539,0,1)
Die 15 Ebenen von Randu in $[0,1[^3$

Abbildung 1.21: Randu im Dimension 2 und 3

Die vorhin skizzierten Regelmäßigkeiten in der erzeugten Punktstruktur sind typisch für den LCG und seine Verallgemeinerungen, die sogenannten mehrfachrekursiven Generatoren, siehe L'Ecuyer et al. [L'E98].

Jeder Standardgenerator erzeugt also *periodische* Folgen von Zufallszahlen. Je nach dem verwendeten Algorithmus treten die Regelmäßigkeiten bereits bei kleineren oder größeren Stichprobengrößen N auf, siehe dazu L'Ecuyer und Hellekalek [LH98]. Eine gängige "Daumenregel" besagt, daß bei linearen Generatoren wie dem LCG oder MRG ab etwa $N = \sqrt{T}$, T die Periode des Generators, zahlreiche Verteilungen nicht mehr genügend genau simuliert werden können. Die Regelmäßigkeiten dieser Generatortypen beginnen sich in vielen Simulationen ab etwa dieser Stichprobengröße ungünstig auf die Ergebnisse auszuwirken. Dieses Problem läßt sich auf drei Arten lösen. Zum einen kann man lineare Generato-

ren mit extrem großen Perioden verwenden, siehe dazu L'Ecuyer [L'E98, LA97] und Matsumoto und Nishimura [MN98],

▷ Link: `http://www.math.keio.ac.jp/$\sim$matumoto/emt.html`

für konkrete Beispiele. Diese Generatoren sind trotz ihrer extremen Perioden sehr schnell. *Nichtlineare*, insbesondere *inverse* Zufallszahlengeneratoren bieten eine andere, zahlentheoretisch besonders reizvolle Alternative. Eichenauer-Herrmann und Niederreiter haben dieses Thema sehr umfassend behandelt, siehe die Überblicksartikel [EHHW97, Nie95, Hel95].

Wie beurteilt man die Qualität eines Zufallszahlengenerators? Hier gibt es mehrere Ebenen, als erste die theoretische Analyse des Generators, dann das empirische (statistische) Testen und schließlich die Bewertung praktisch relevanter Eigenschaften wie numerische Effizienz, Verhalten bei Parallelisierungsstrategien oder Portabilität.

Die Grundidee des theoretischen Testens ist einfach. Zur Überprüfung von Korrelationen zwischen den Zufallszahlen $x_n \in [0,1[$ bildet man auf verschiedene Weise d-Tupel $\mathbf{x}_n \in [0,1[^d$, zum Beispiel wie in (1.15) angegeben. Man erhält auf diese Weise –wegen der Periodizität des Generators– endliche Punktfolgen $\omega_d = (\mathbf{x}_n)_{n=0}^{N-1}$ im d-dimensionalen Einheitswürfel $[0,1[^d$, die annähernd gleichverteilt sein sollten. Darüber, wie gut die Gleichverteilung angenähert werden soll, gibt es zwei verschiedene Ansichten, siehe dazu die Darstellung in Hellekalek [Hel98a]. Im Fall linearer Generatoren kann man auf Grund ihrer Gitterstruktur sehr interessante Methoden der Geometrie der Zahlen verwenden, um zu einem gegebenen Generator und gegebener Dimension d ein konkretes Gütemaß zu berechnen. Der wichtigste Gütebegriff in diesem Zusammenhang heißt *Spektraltest*, siehe Hellekalek [Hel98b] für einen Überblick.

Während sich mit dem Spektraltest nur Generatoren mit Gitterstruktur beurteilen lassen, erlaubt die aus der Theorie der Gleichverteilung von Folgen bekannte *Diskrepanz* zumindest theoretisch die Bewertung beliebiger Punktmengen ω_d und damit beliebiger Generatoren. Es ist damit möglich, fast alle derzeit bekannten Generatortypen im Monte Carlo Bereich sowie einige eher theoretisch als praktisch interessante kryptographische Generatoren zu analysieren. Leider ist diese Art von theoretischer Korrelationsanalyse für keinen der derzeit in der Praxis üblichen kryptographischen Generatoren bekannt.

Die theoretische Analyse der Monte Carlo Generatoren ist nämlich deswegen erfolgreich, da diese Algorithmen auf sehr einfachen Rekursionen beruhen und damit zahlreiche zahlentheoretische Methoden anwendbar sind. Die Algorithmen der Monte Carlo Methoden sind zwar raffiniert genug, um statistische Tests zu täuschen, die innere Struktur dieser Zufallszahlen ist aber zu simpel, um kryptanalytischen Methoden zu widerstehen.

Kryptographische Generatoren werden so konstruiert, dass sie nicht nur die statistischen Tests überlisten, sondern dass die erzeugten Zufallszahlen möglichst keine Struktur besitzen. Da also keine mathematisch einfach beschreibbaren Strukturen (wie Punkte auf Hyperebenen etc.) vorliegen, wird die theoretische Analyse unmöglich. Wir haben keine Regelmäßigkeiten in der Hand, mit denen wir mathematische Beweise führen könnten. Umgekehrt lassen sich aus genau diesem Grund keine Beweise für die praktische Sicherheit dieser Generatoren

angeben. Ein Restrisiko bleibt trotz der ausgefeilten Konstruktionsprinzipien bestehen. Irgendwann in der Zukunft wird es möglich sein, einen neuen Typ von mathematischer Struktur in den Zufallszahlen zu entdecken, den wir heute noch nicht kennen. Er könnte dann ein geeigneter Angriffspunkt sein.

Der Generator von Blum, Blum und Shub [BBS86] (BBS-Generator) gilt als einer der wichtigsten kryptographischen Generatoren. Er ist aber vor allem von rein theoretischem Interesse und für die Praxis kaum relevant. Er ist wie folgt definiert. Seien p und q zwei verschiedene Primzahlen, beide kongruent 3 modulo n, wobei $n = pq$. Für eine nichtnegative ganze Zahl a bezeichne $LSB(a)$ das *least significant bit* in der Binärentwicklung von a, also das Bit a_0, falls $a = \sum_{j\geq 0} a_j\ 2^j$. Die Komponenten des BBS-Generators lauten:

$\mathcal{S}$	$\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$
T	$T(s) = s^2$
$\mathcal{O}$	$\{0, 1\}$
g	$g(s) = LSB(s)$
s_0	$s_0 \in \mathbb{Z}_m \setminus \{0\}$, zufällig

Die Periodenlänge konkreter Ausgabeströme ist nicht bekannt, allerdings existieren probabilistische Resultate.

Der RSA-Generator oder Power-Generator ist am RSA-Verfahren angelehnt. Seien p und q zwei verschiedene Primzahlen und sei $n = pq$. Sei e eine feste ganze Zahl mit $1 < e < \varphi(n)$ und $(e, \varphi(n)) = 1$. Die Komponenten des RSA-Generators lauten:

$\mathcal{S}$	$\mathbb{Z}_n = \{0, 1, \ldots, n-1\}$
T	$T(s) = s^e$
$\mathcal{O}$	$\{0, 1\}$
g	$g(s) = LSB(s)$
s_0	$s_0 \in \mathbb{Z}_m \setminus \{0\}$, zufällig

Auch für diesen Generator kennt man keine exakten Resultate zur Periodenlänge konkreter Ausgabeströme, jedoch konnte eine Reihe von schönen zahlentheoretischen Aussagen bewiesen werden (siehe [NS02]).

Unter der Adresse `http://www.ecrypt.eu.org/stream/index.html` finden Sie folgenden Text:

> *This is the home page for eSTREAM, the ECRYPT Stream Cipher Project. This is a multi-year effort to identify new stream ciphers that might become suitable for widespread adoption. All information on the stream cipher project can be found on this site.*

Mit anderen Worten, eSTREAM ist ein europäisches Projekt, in dem Vorschläge für neue Stromchiffren diskutiert werden. Einerseits wurden aus der ganzen Welt diverse Stromchiffren vorgeschlagen, andererseits treffen aus der ganzen Welt Analysen und manchmal sogar Attacken auf diese Algorithmen ein.

Wir werden aus den Vorschlägen für Stromchiffren den Algorithmus CryptMT in der Version 2 vorstellen. Diese Stromchiffre stammt von Matsumoto, Saito,

Nishimura und Hagita [MSNH06]. Die Autoren stellen im Abstract zur Arbeit fest:

> *As a pseudorandom number generator (PRNG) for a stream cipher, we propose a combination of (1) an $\mathbb{F}_2$-linear generator of a wordsize integer sequence with huge state space, and (2) a filter with one wordsize memory, based on the accumulative integer multiplication and extracting some most significant bits from the memory. We proposed CryptMT as an example. Merits of this type of generators are (1) the strength against various attacks assured by the huge state, (2) assurance on the period and the distribution, and (3) high algebraic degree and nonlinearity obtained by the integer multiplication.*

In ihrer Sicherheitsanalyse des eigenen Generators CryptMT behaupten die Autoren (siehe [MSNH05]):

> *CryptMT (Cryptographic Mersenne Twister) is an 8-bit pseudorandom integer generator for a stream cipher. It combines an $\mathbb{F}_2$-linear generator of period $2^{19937-1}$ and a multiplicative filter with 32-bit memory. We analyze its security against some standard cryptanalytic attacks for filter generators. It is proved that CryptMT has strong resistance against them: CryptMT has a period of $2^{19937-1}$, the correlations among the consecutive 624-bytes of outputs are of order 2^{-19937}, the algebraic degree of the output bits with respect to the bits in Key and IV is expected to near to the size of Key and IV. The Key size and IV size are variable, up to 2048-bit for each. We claim that CryptMT has the same security level with the minimum of the key size and the IV size. CryptMT is 1.52.0 times faster than the optimized AES CTR mode with 256-bit security level.*

Wir beschreiben nun den Algorithmus, der CryptMT zugrunde liegt.

1.85 Definition (CryptMT)
Das Konzept:
Sei S ein Vektorraum und sei $T : S \to S$ eine lineare Abbildung und sei $s_0, s_1, \ldots$ die Folge $s_{i+1} = T(s_i)$, $i \geq 0$. Sei Y eine endliche Menge. Y sei die Menge der möglichen Zustände des Speichers des Filters. Sei y_0 ein Startzustand im Speicher und sei f folgende Übergangsfunktion für Y:

$$\begin{aligned} f : Y \times X &\to Y, \\ y_{i+1} &= f(y_i, s_i). \end{aligned}$$

Die Ausgabefunktion g operiere auf Y, $g : Y \to \mathcal{O}$.

Die Details:

$$\begin{aligned} S &= \mathbb{F}_2^{32} \\ T &= \text{Mersenne Twister} \\ f(y, x) &= y \cdot (x|1) \pmod{2^{32}} \\ g(y) &= 8 \text{ MSB von } y, \end{aligned}$$

wobei $(x|1)$ jene ganze Zahl bezeichnet, die aus x hervorgeht, wenn das LSB gleich 1 gesetzt wurde, und "8 MSB von y" die acht MSB der ganzen Zahl y bezeichnet. Zum Start wird der IV (initial value) y_0 als eine beliebige ungerade Zahl gewählt.

Für die weitere Diskussion von CryptMT wird auf die eSTREAM-Webseite verwiesen:

▷ Link: `http://www.ecrypt.eu.org/stream/index.html`

Für jene Leser, denen (Pseudo-)Zufallszahlen weiterhin suspekt sind und die "echte" Zufallszahlen bevorzugen (Was ist hier mit "echt" eigentlich gemeint? Siehe dazu Kac [Kac83]), empfehle ich einen Blick auf die "HOT BITS" von John Walker. Es handelt sich hier um Bits (und Bytes), die aus einem radioaktiven Zerfallsprozeß stammen. Sie sind von der folgenden Seite zu beziehen:

▷ Link: `http://www.fourmilab.ch/hotbits/`

Literaturverzeichnis

[Bau97] F. L. Bauer. *Decrypted Secrets*. Springer, Berlin, 1997.

[BBS86] L. Blum, M. Blum, and M. Shub. A simple unpredictable pseudo-random number generator. *SIAM J. Comput.*, **15**:364–383, 1986.

[Buc01] J. Buchmann. *Einführung in die Kryptographie*. Springer Verlag, 2nd edition, 2001.

[Cal94] C. Calude. *Information and Randomness*. Springer Verlag, 1994.

[EHHW97] J. Eichenauer-Herrmann, E. Herrmann, and S. Wegenkittl. A survey of quadratic and inversive congruential pseudorandom numbers. volume 127 of *Springer Lecture Notes in Statistics*, pages 66–97. Springer, New York, 1997.

[Ert01] W. Ertel. *Angewandte Kryptographie*. Vieweg, Braunschweig, 2001.

[Fis96a] G. Fischer. Cäsarcode, Vigenèrecode und Shannons Zugang zur Kryptographie. Master's thesis, University of Salzburg, 1996. [Sehr empfehlenswerte Arbeit!].

[Fis96b] G.S. Fishman. *Monte Carlo: Concepts, Algorithms, and Applications*. Springer, New York, 1996.

[Hel95] P. Hellekalek. Inversive pseudorandom number generators: concepts, results, and links. In C. Alexopoulos, K. Kang, W.R. Lilegdon, and D. Goldsman, editors, *Proceedings of the 1995 Winter Simulation Conference*, pages 255–262, 1995.

[Hel98a] P. Hellekalek. Good random number generators are (not so) easy to find. *Mathematics and Computers in Simulation*, **46**:485–505, 1998.

[Hel98b] P. Hellekalek. On the assessment of random and quasi-random point sets. In Hellekalek and Larcher [HL98], pages 49–108.

[Hil86] R. Hill. *A First Course in Coding Theory*. Clarendon Press, Oxford, 1986.

[HL98] P. Hellekalek and G. Larcher, editors. *Random and Quasi-Random Point Sets*, volume 138 of *Springer Lecture Notes in Statistics*. Springer, New York, 1998.

[Kac83] M. Kac. What is random? *American Scientist*, **71**:405–406, 1983.

[Knu98] D.E. Knuth. *The Art of Computer Programming, Vol. 2*. Addison-Wesley, Reading, Mass., third edition, 1998.

[LA97] P. L'Ecuyer and T.H. Andres. A random number generator based on the combination of four LCGs. *Mathematics and Computers in Simulation*, **44**:99–107, 1997.

[L'E98] P. L'Ecuyer. Random number generation. In Jerry Banks, editor, *The Handbook of Simulation*, pages 93–137. Wiley, New York, 1998.

[LH98] P. L'Ecuyer and P. Hellekalek. Testing random number generators. In Hellekalek and Larcher [HL98], pages 223–265.

[MN98] M. Matsumoto and T. Nishimura. Mersenne Twister: A 623-dimensionally equidistributed uniform pseudorandom number generator. *ACM Trans. Modeling and Computer Simulation*, **8**:3–30, 1998.

[MSNH05] M. Matsumoto, M. Saito, T. Nishimura, and M. Hagita. Cryptanalysis of CryptMT: effect of huge prime period and multiplicative filter. eSTREAM, ECRYPT Stream Cipher Project, Report 2005/083, 2005. `http://www.ecrypt.eu.org/stream`.

[MSNH06] M. Matsumoto, M. Saito, T. Nishimura, and M. Hagita. CryptMT Version 2.0: A large state generator with faster initialization. eSTREAM, ECRYPT Stream Cipher Project, Report 2006/023, 2006. `http://www.ecrypt.eu.org/stream`.

[Nie92] H. Niederreiter. *Random Number Generation and Quasi-Monte Carlo Methods*. SIAM, Philadelphia, 1992.

[Nie95] H. Niederreiter. New developments in uniform pseudorandom number and vector generation. In H. Niederreiter and P.J.-S. Shiue, editors, *Monte Carlo and Quasi-Monte Carlo Methods in Scientific Computing*, volume 106 of *Lecture Notes in Statistics*, pages 87–120. Springer, New York, 1995.

[NS02] H. Niederreiter and I. E. Shparlinski. Recent advances in the theory of nonlinear pseudorandom number generators. In K.-T. Fang, F.J. Hickernell, and H. Niederreiter, editors, *Monte Carlo and Quasi-Monte Carlo Methods 2000*, pages 86–102. Springer, New York, 2002.

[Sha49] C. E. Shannon. Communication theory of secrecy systems. *Bell Sys. Tech. J.*, 28:656–715, 1949.

[Sob83] I.M. Sobol. *Die Monte-Carlo-Methode*. VEB Deutscher Verlag der Wissenschaften, 1983.

[Sti06] D. R. Stinson. *Cryptography*. Chapman and Hall/CRC Press, Boca Raton, 3rd edition, 2006.

[TW06] W. Trappe and L. Washington. *Introduction to Cryptography with Coding Theory*. Pearson Prentice Hall, 2nd edition, 2006.

[Vau06] S. Vaudenay. *A Classical Introduction to Cryptography.* Springer, 2006.

[Wel88] D. Welsh. *Codes and Cryptography.* Oxford Science Publ., 1988.