Codierung:

Bsp. Musik vom CD Player
- ➔ auf Hi-Fi Verstärker: per Leitung, keine Codierung notwendig
- ➔ ins Radio: hochfrequente Modulation, um per Funk übertragen zu können

Quellencodierung:
- definiert das grundlegende Datenformat (Quelle → Transmitter)
- z.B. PCM, DPCM, DM, $\Sigma\Delta$ - Mod. (Sigma-Delta-Mod., prädiktive Codierung, funktioniert bei Zufallssignalen nicht)

Kanalcodierung:
- erkennt Fehler, unter Umständen aber auch Fehlerkorrektur

Leitungscodierung:
- bereitet das Signal für einen physikalischen Kanal auf (Transmitter → Kanal → Receiver)

➔ Musik und Sprache sind Grenzfälle zwischen deterministischen Signalen und stochastischen Signalen → Algorithmen werden direkt auf bestimmte Musikstile ausgerichtet

Erinnerung: Shannonquader

$$I = B \cdot D \cdot T$$

$$D \approx \frac{SNR}{3}$$

Signal = Entropie → besteht aus relevanter/irrelevanter Information
- Rauschen würde den Quader komplett ausfüllen → enthält keine Redundanz
- Signal sollte möglichst redundanzarm codiert werden → redundanzfrei ist nicht möglich

➔ eine effiziente Quellencodierung ist redundanzarm → Sprache, Musik, Videobilder

➔ Entropie-Codierung: Bsp. ZIP, RAR, Stenografie, Kürzel

Einführungsbeispiel:
- Schwarz/Weiß Bild → Es werden die Zustandsänderungen gespeichert

➔ RLE = Run Length Encoding (TLA = Three Letter Abbreviation)
- funktioniert nicht bei einem Zufallssignal
- Codierung der Anzahl der folgenden gleichen Stufen muss groß sein, um Bits zu sparen

Huffman-Code:
- codiert nach der Symbolwahrscheinlichkeit → gut geeignet für beliebige Texte
- Beispieltext: ANANASAPFELANANASBANANE (23 Buchstaben) →A,N,S,P,F,E,L,B

➔ man muss einen repräsentativen Signalanteil finden

1. Berechnung der Wahrscheinlichkeit der Buchstaben in der Zeichenkette

A	9/23
N	6/23
S	2/23
E	2/23
P	1/23
F	1/23
L	1/23
B	1/23

- ASCII (US-Zeichensatz, 256 Zeichen) codiert würde pro Zeichen 1Byte = 8 Bit gebraucht

➔ man würde 23 Byte benötigen

- 1. Verbesserung → es gibt nur 8 Symbole → 3 Bit pro Symbol würden reichen → 23 Zeichen → 69 Bit = 9 Byte

➔ Sortierung der Zeichen nach Wahrscheinlichkeit (s. Tabelle)

- Codierung ist niemals eindeutig → man kann durch Probieren mehrere „Huffman-Bäume“ zur Codierung finden → Codeschlüssel muss mit gesendet werden, sonst kann nicht übertragen werden

Testlauf mit Huffman A

- N ist häufiger und hat 3-Bit-Code, E ist seltener aber in 2 Bit codiert

➔ Lösung: Tauschen von N und E

➔ durch Rechnung Häufigkeit x Bitlänge ergibt sich eine Länge von 58Bit = 8Byte für Huffman A

Testlauf mit Huffman B

➔ Huffman B ergibt 57 Bit = 8 Byte

LZW (Lempel, Ziv & Welch)

- man braucht keinen Code, sondern nur die Info, dass LZW genutzt wurde, zu übertragen
- Basis von Formaten wie ZIP, RAR, PDF, etc. → überall, wo man Muster finden kann
- Basiscode: z.B. 1Byte = 8Bit → 0…255
- LZW scannt das Signal und erzeugt dabei den Code (das Codebook) → Code hängt direkt mit dem Signal zusammen, wird also durch das Signal gesteuert
- Beispieltext: ANANASAPFELANANASBANANE (23 Buchstaben) →A,N,S,P,F,E,L,B

➔ = A-Z (+ weitere Zeichen) → 0…255

Testlauf mit LZW:

Zeichennummer	Codiertes Signal	Codebook
1	A	AN=256
2	N	NA=257
3	256	ANA=258
4	A	AS=259
5	S	SA=260
6	A	AP=261
7	P	PF=262
8	F	FE=263
9	E	EL=264
10	L	LA=265
11	258	ANAN=266
12	257	NAS=267
13	S	SB=268
14	B	BA=269
15	266	ANANE=270
16	E	

- solange links etwas neues kommt und kein Muster erkannt wird, passiert nichts (Codebook wird immer verlängert)
- ist ein Muster schon vermerkt, wird das Codebook erweitert

Vergleich verschiedener Kompressionsformate:
Unicode 8 → 2300 Byte
Unicode 16 → 4602 Byte
RTF (Richt Text Format, mit Metadaten und Formatierung) → 2637 Byte
DOC → 23.040 Byte (großer Header)
PDF (von Unicode aus) → 12 KB
ZIP (von Unicode aus) → 209 Byte

Bei Audiosignalen (Sinus und weißes Rauschen) sieht man, dass ein Zufallssignal nicht bearbeitet werden kann, ein deterministisches kann redundanzreduziert werden

Kanalcodierung

- Beispiel Paritätsbit: 8Bit → 9Bit

➔ 2^9 Codeworte, davon 2^8 zulässige Datenwerte

Hamming-Distanz (Richard Wesley Hamming)

- Zahl von Bits, die man mindestens verändern muss, um ein gültiges Codewort in ein anderes gültiges Codewort zu überführen
- Am Beispiel Paritätsbit:
    - Ändern von 1 Bit in einem gültigen Codewort → man erhält ungültiges Codewort
    - Ändern von 2 Bit in einem gültigen Codewort → man erhält gültiges Codewort

➔ Hamming-Distanz → H=2

- man kann beliebig viele Fälle konstruieren, in denen der Code versagt
- Paritätsbit ergibt sich entsprechend der Einsen im digitalen Wort (z.B. 0 = geradzahlige Anzahl, 1 = ungeradzahlige Anzahl)

Wiederholungscode (Reception Code) → sehr ineffizient

- man wiederholt z.B. jedes Bit drei Mal (3-fache Sendung)
- $0 = 000, 1 = 111$ → gültig
- $001, 010, 011, 100, 101, 110$ → ungültig
- H=3, man muss drei Werte ändern, um wieder einen gültigen Code zu erhalten
- Redundanz: 66% (bezogen auf absolute Datenmenge)

➔ ein Code mit der Hamming-Distanz H kann…

- $H-1$ Bitfehler erkennen
- $\frac{H-1}{2}$ Fehler korrigieren

Würfelmodell:

- gültige Codeworte haben größtmöglichen Abstand

➔ MAXIMUM LIKELEHOOD