Compact Lecture

Multimedia Coding: Methods & Applications

Part 1.2: Introducing Coding Fundamentals

Dr. Klaus Illgner

Dr. Uwe Rauschenbach

What is „Probability?“

Several concepts to approach „probability“

Empirical concept → „Relative Frequency“

Let‘s start with a finite set of events, e.g. a die

$\Omega = \{1, 2, 3, 4, 5, 6\}$ are elementary events

Ω is the probability space

Tossing the dice once → called an experiment with a certain unknown outcome

The outcome is an event $\mathcal{A}$ → element of the probability space $\mathcal{A} \subset \Omega$

Event is any subset of the probability space, e.g. $\mathcal{A} = \{2, 4, 6\}$

Repeating the experiment N times

Relative frequency of event $\mathcal{A}$ $\qquad H(\mathcal{A}) = \frac{n_{\mathcal{A}}}{n}$

Probability: $\qquad P(\mathcal{A}) := \lim_{n \to \infty} H(\mathcal{A})$

Note:
This it NOT a definition!

Concept of Random Variables

Discrete Random Variable (RV): $X : \Omega \to \mathbb{Z}$

Is a function to map the outcome of an experiment to a number $\mathbf{x}(\mathcal{A})$

$$\Omega = \{\mathrm{KOPF}, \mathrm{ZAHL}\} \qquad X(KOPF) = 1,\ p_X(1) = p_1 = p_{KOPF}$$
$$x_1 = \mathrm{KOPF},\ x_2 = \mathrm{ZAHL} \qquad X(ZAHL) = 2,\ p_X(2) = p_2 = p_{ZAHL}$$

Discrete Random Process:

Sequence of random variables $\mathbf{X} = \{X_i\}$

→ used to model signal sources

Probability of a random variable:

event $\mathcal{A} = \{\mathbf{x} \leq x_0\}$

with probability $P(\mathcal{A}) = P\{\mathbf{x} \leq x_0\}$

Distribution and Density Functions

Distribution function:

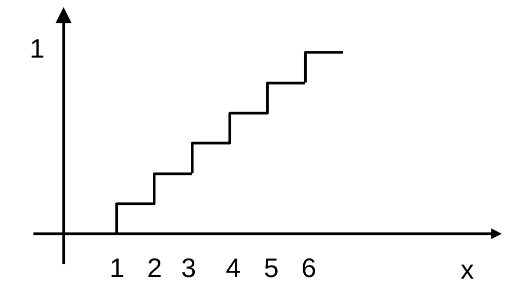
$$P_X(x) = P\{X \leq x\}$$

Characteristics:

$$P_X(-\infty) = 0 \qquad P_X(\infty) = 1$$

$$\text{if} \quad x_1 < x_2 \quad \text{then} \quad P_X(x_1) < P_X(x_2)$$

Example: PF of a faire die



Probability density function (pdf):

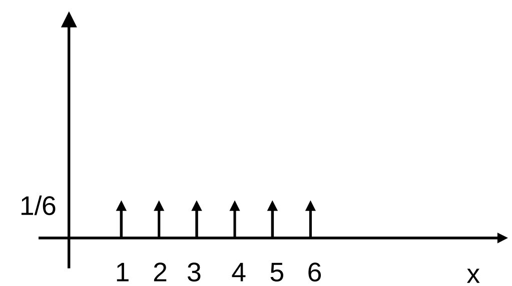
$$p_X(x) := \frac{d}{dx} P_X(x)$$

Characteristics:

$$p_x(i) \geq 0; \quad \sum_{i=1}^{N} p_x(i) = 1$$

$$P(x_n) = \sum_{i=-\infty}^{n} p(x_n), \quad n \leq N$$

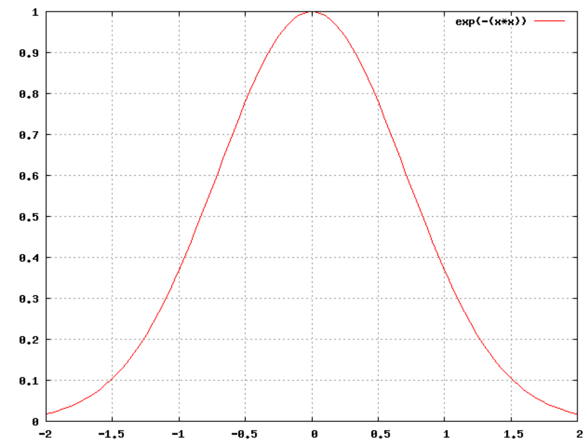
Example: PDF of a faire die



Specific Probability Density Functions

Normal standard distribution (Gaussian Distribution):

$$p_N(x) = \frac{1}{\sigma_N \sqrt{2\pi}} e^{-x^2/2\sigma_N^2} \qquad \bar{x}_N = 0, \quad \sigma_N^2 = 1$$



Other frequently applicable probability density functions:

- Gamma distribution
- Binomial distribution
- Laplace distribution
- Poisson distribution

Characterizing RV:

Expectation:

$$E\{\mathbf{X}\} = \sum_i x_i p(x_i), \quad x_i \in \Omega$$

Moment of k^th order
(Mean = 1^st order)

$$\bar{X} = E\{\mathbf{X}^k\}$$

Central moment of k^th order
(variance = 1^st order)

$$\sigma^k = E\{(\mathbf{X} - \bar{x})^k\} = \sum_i (x_i - \bar{x})^k p(x_i)$$

Joint Distribution Functions

Joint density distribution $p(x_i, y_j), \quad \mathrm{RV}\mathbf{X}, \mathbf{Y}$

Statistical independence: $p(x_i, y_j) = p(x_i)p(y_j)$

Conditional probability: $p(x|y) := \frac{p(x,y)}{p(y)}$

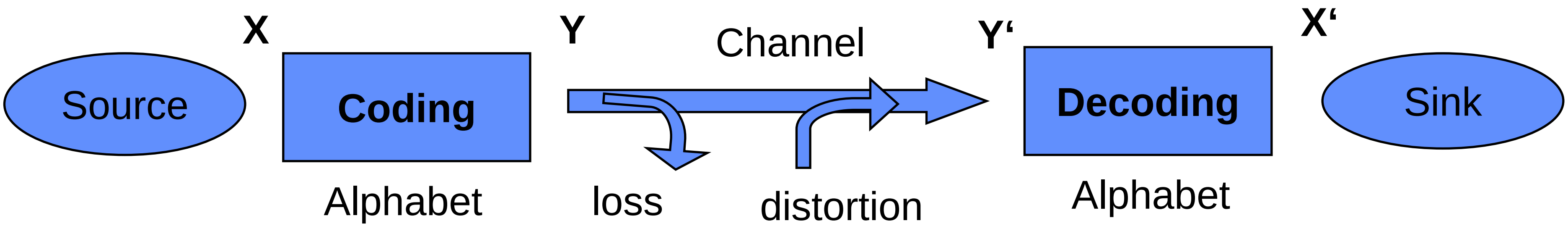
Bayes‘ rule: $p(x|y) = \frac{p(y|x)}{p(y)}p(x)$

These concepts can be generalized for random vectors.

Information

Example: Message

- Transmitter has an information (in mind) and wants to communicate it (semantics)
- He formulates sentences – by using a character set (coding of the information)
- Receiver regenerates sentences from the characters received (decoding)
- sink reconstructs semantics -- „understands“ the meaning of the sentence (hopefully correctly)



What is information?

- Information theory defines the information content of a message as “amount of information” – independent of the actual meaning (semantics).
- Entropy refers to the average information content of a source

What does „Coding“ really mean?

Characteristics:

- Separation of transmitted information from its meaning
- the message itself as well as the transmission point of time is unknown
  → information content is the “uncertainty” / information not known to the receiver

Coding: one-to-one mapping of source alphabet to code alphabet

Goal:

representing „Information“ with the least number of “bits”

… without changing / limiting the information

→ **redundancy reduction,** also termed **lossless** coding

→ in contrast to **lossy** coding, which modifies the source signal

Which is the most “efficient” code?

→ transmitting the most frequently symbols with the shortest code words

→ selecting the right symbols (correlation / higher order statistics)

→ requires to model the signal statistics of the source

Types of Codes

Source alphabet $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$

Code alphabet $\mathcal{Y} = \{y_1, y_2, \ldots, y_M\}$

Set of representable messages: $\mathcal{Y}^* = U_{i \in \mathbb{N}} \mathcal{Y}^i$

Code $\mathcal{C} : \mathcal{X} \to \mathcal{Y}^*$

Block codes → „fixed length“ codes, e.g. ASCII, HEX, …

not applicable to source coding (redundancy reduction)

Variable length codes (VLC): frequent symbols → short code words

seldom symbols → long code words

Example: Morse-Code

- Mapping of characters, digits, and 3 control characters to binary symbols

A $\bullet\ -$

E $\bullet$

Q $-\ -\ \bullet\ -$

$\mathcal{X} = \{A, B, \ldots, Z, 0, \ldots, 9\}$

$\mathcal{Y} = \{\bullet, -\}$

- Length of binary character string depends on frequency of characters

Information Content and Entropy

Source signal and random variables: $\mathbf{X} : x \in \mathcal{X}, \quad \mathbf{Y} : y \in \mathcal{Y}$

Information content of symbols x_i

$$I(x_i) := -\log_2 p(x_i), \quad \text{(in bit)}$$

Entropy represents the **average** Information content / uncertainty of a random variable

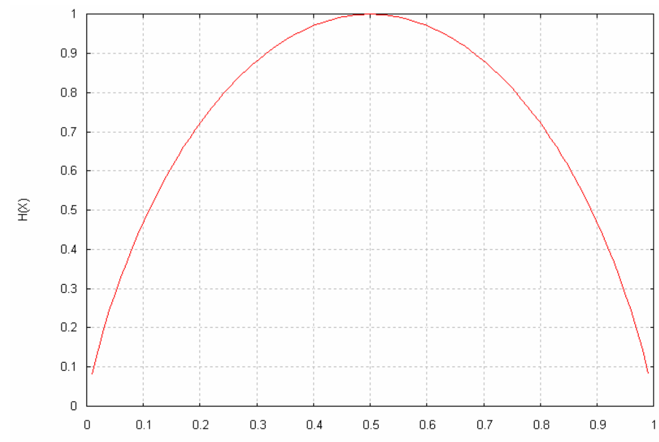
$$H(\mathbf{X}) = E\{I(\mathbf{X})\} = -\sum_{x \in \mathcal{X}} p_x log_2(p_x)$$

Code word length (bits)

$$l(x), \quad x \in \mathcal{X}$$

Expected code word length

$$L(\mathcal{C}) = \sum_{x \in \mathcal{X}} l(x) p_x$$

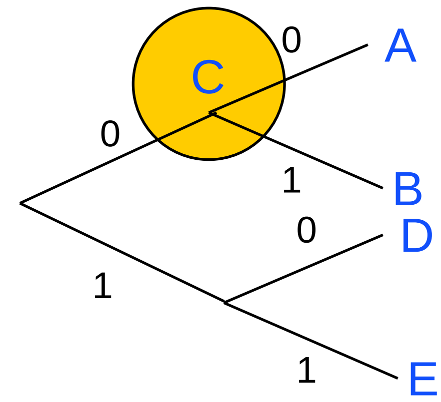


Entropy of a binary source

Unique Decodability

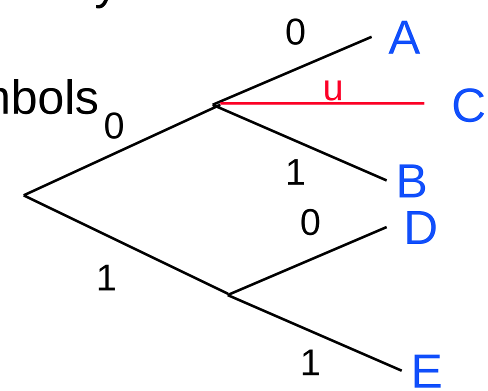
Example:

x	A	B	C	D	E
$\mathcal{C}(x)$	00	01	0	10	11



Code is <u>uniquely</u> decodable,
no codeword is a combination of 2 other codewords

Code is <u>NOT instantaneously</u> decodable,
Since a “comma” is needed for instantaneously decode received symbols
(without knowing the other coded symbols)
(compare Morse-Code → comma inserts breaks between symbols



x	A	B	C	D	E
$\mathcal{C}(x)$	00	01	0u	10	11

Continuous Decodability

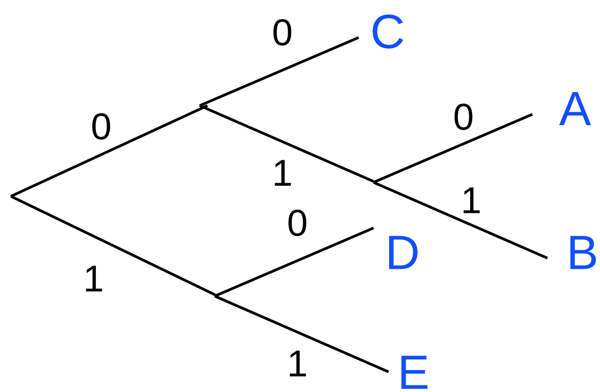
Definition:

A Code, which can be continuously and instantaneously decoded without coma is called a Prefix-Code. No code word is a prefix of any other code word.

A prefix code is characterized by the fact, that only leaves are valid code words.

Example:

x	A	B	C	D	E
$\mathcal{C}(x)$	010	011	00	10	11



So called (3 3 2 2 2) Code, and (2 2 3 3 3) respectively

Kraft Inequality

Theorem:

a prefix code exists if and only if:

$$K = \sum_{i=1}^{N} \frac{1}{M^{l(x_i)}} \leq 1$$

$$\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$$

$$\mathcal{Y} = \{y_1, y_2, \ldots, y_M\}$$

M: alphabt size; binary M=2
N: number of code words

Example:

For (2 2 3 3 3) code :

$$K = \tfrac{1}{2^2} + \tfrac{1}{2^2} + \tfrac{1}{2^3} + \tfrac{1}{2^3} + \tfrac{1}{2^3} = 0,875 < 1$$

What is a „good“ Code? – Optimal Codes

Minimizing the mean expected code length

$$L(\mathcal{C}) = \sum_{x \in \mathcal{X}} l(x) p_x \quad \rightarrow \min$$

Optimal code length:

$$l(x_i)^* = -\log_D x_i$$

Binary codes: D = 2

Entropy is the lower bound of the expected code length!

Characteristics of optimal codes

$$p_i > p_j \rightarrow l_i \leq l_j \quad \forall i, j$$

- The longest code words (minimum 2) have the same length
- They differ only on the least significant bit

Huffman Code

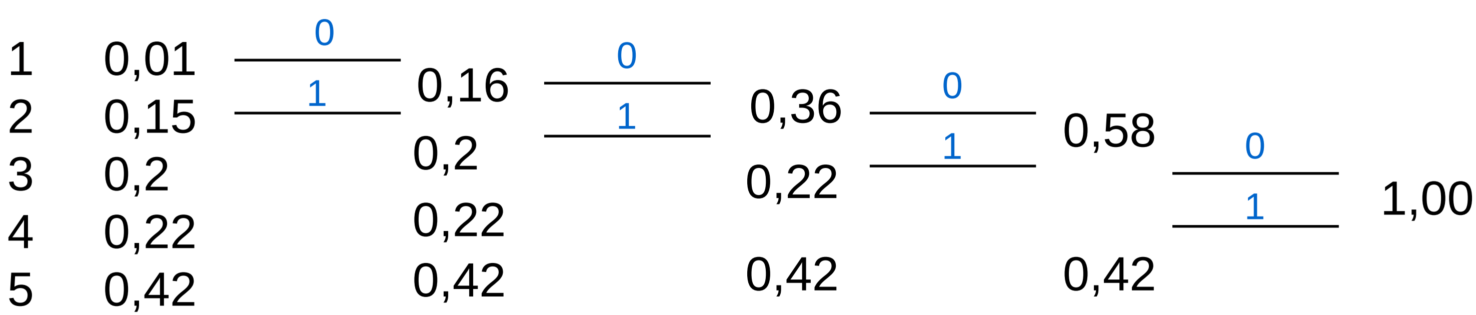
Optimal code satisfies: $$l(x_i) = I(x_i) = -\log_2(p_i)$$

BUT: l is not an integer

<u>Shannon Code:</u> $$l(x_i) = \lceil -\log_2(p_i) \rceil$$

BUT: Shannon Code is in general NOT optimal

Huffman Code is just one possible optimal code



Code Extensions (1)

Problem:

requirement to assign at least 1 bit to the most likely symbol
→ Huffman Code is inefficient for unequal distributions

For optimal codes the equation holds (e.g. Huffman Code)

$$H(X) \leq L(X) < H(X) + 1$$

Optimization Approach:

Handling sequences of symbols as new symbols (new alphabet)
→ Enlarging the alphabet size

Original source: $\mathcal{X} = \{x_1, x_2, \ldots, x_N\}$

New source: merging of m symbols → new alphabet with N^m symbols

$$\mathcal{X} = \{x_1x_1, x_2x_1, \ldots, x_Nx_1, x_1x_2, x_2x_2, \ldots, x_Nx_2, \ldots x_Nx_N\} \quad m = 2$$

Such a code extension is termed <u>product code</u>

$$C^m(x_1, x_2, \ldots, x_m) = C(x_1)C(x_2)\ldots C(x_m)$$

Code Extensions (2)

Alternative:
Calculating a “new” optimal code based on product probabilities $p_{x_1 x_2} = p_{x_1} p_{x_2}$

Assuming statistical independence:

$$H(X_1, X_2, \ldots, X_n) = \sum H(X_i) = nH(X)$$

$$H(X) \leq L(X) < H(X) + \frac{1}{n}$$

→ Lower bound given by entropy can be reached in principle
→ implementation cost increases exponentially

Disadvantage:
Size of code table increases exponentially
→ Practically not manageable anymore
→ gain shrinks asymptotically

Code Extensions: An Example

Original source:

S1: (black) $p_S = 0.3$ → $l(S1) = 1$ bit
S2: (white) $p_w = 0.7$ → $l(S2) = 1$ bit

H = 0,88 bit / symbol
L = 1 bit / symbol

C_R = H – L = 0,12 bit / symbol

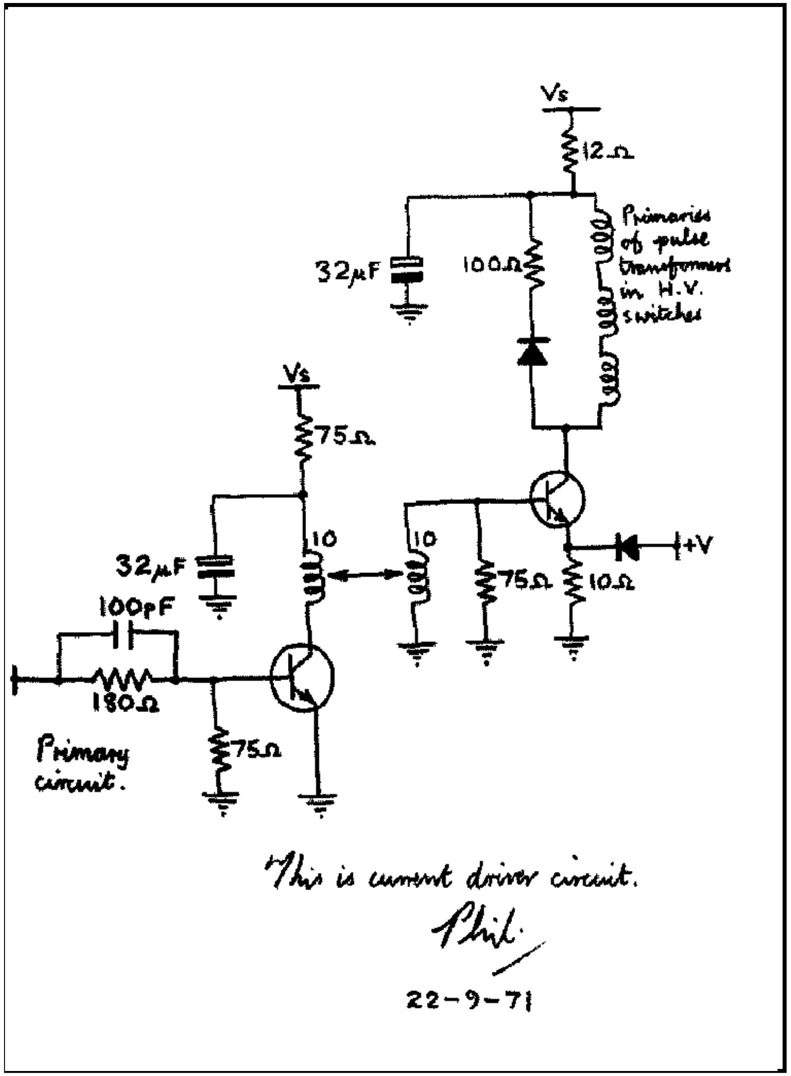
New source: (assuming <u>statistically independent</u> symbols)

S1: (black, black) $p_{SS} = 0.09$ → $l(S1) = 3$ bit
S2: (black, white) $p_{SW} = 0.21$ → $l(S2) = 3$ bit
S3: (white, black) $p_{WS} = 0.21$ → $l(S3) = 2$ bit
S4: (white, white) $p_{WW} = 0.49$ → $l(S4) = 1$ bit

H = 1,76 bit / 2 symbole = 0,88 bit / symbol
L = 1,81 bit / 2 symbole = 0,91 bit / symbol

C_R = H – L = 0,03 bit / symbol

Example FAX



Assuming statistical independence
the average code length gets smaller
by treating several (two) pixels as one symbol:

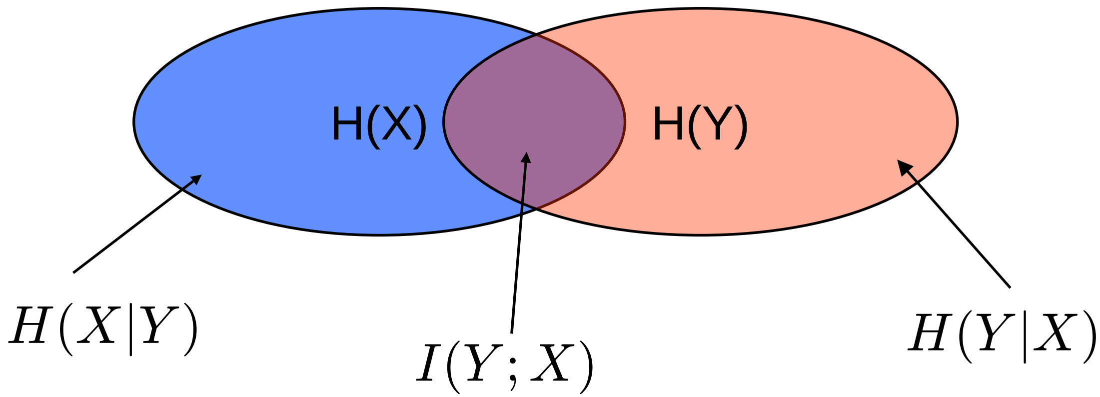
p(1) = 0.95232	H = 0.19475 bit
p(0) = 0.04768	L = 1 bit

1. Code extension:

p(0) * p(0) = 0,002273 3 bit
p(0) * p(1) = 0,045407 3 bit
p(0) * p(1) = 0,045407 2 bit
p(1) * p(1) = 0,906913 1 bit

L = 1.14 bit / 2 symbols = 0.57038 bit / symbol

Conditional Entropy



Chain property:

$$H(X,Y) = H(X) + H(Y/X) = H(Y) + H(X|Y)$$

Mutual information:

$$I(Y;X) = H(Y)H(Y|X) = H(X)H(X|Y) = I(X;Y)$$

Statistical dependency:

Knowing the previously decoded symbol allows to deduce information about following symbol

$$H(Y|X) = H(Y) - I(Y;X) \leq H(Y)$$

Symbols Sequences – Source with Memory

Example:

Source transmits words (ordered set of symbols)

frequency of appearance depends on joint probability

$$p(x_1, x_2, x_3, \dots, x_K)$$

Modeling the words a **random vector**: $\mathbf{X} = \{X_1, X_2, \dots, X_K\}$

Joint entropy (K=2)

$$H(X,Y) = H(X) + H(Y) - I(X;Y)$$

Statistical dependency of symbols → source with memory $I(X;Y) > 0$

→ Coding taking into account neighbored symbols (in time / in space)
→ taking into account the **context**

Example: FAX

Replacing the mean code length with actually measured joint probabilities

factor 17!

P(0,0) = 0.039972	p(0) * p(0) = 0,002273
P(0,1) = 0.007617	p(0) * p(1) = 0,045407
P(1,0) = 0.007798	p(0) * p(1) = 0,045407
P(1,1) = 0.944613	p(1) * p(1) = 0,906913

Neighbored signal values are ***statistically dependent***

H = 0,18576 bit /sample H = 0.19475 bit/sample

P(0,0) = 2 bit
P(1,0) = 3 bit
P(0,1) = 3 bit → L = 1,0708 bit / 2 sample
P (1,1) = 1bit = 0,5354 bit / sample

Predictive Coding

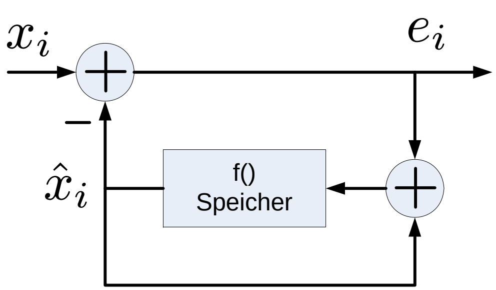
different approach (traditional way) to take into account statistical dependency of random vectors for coding

Principle:

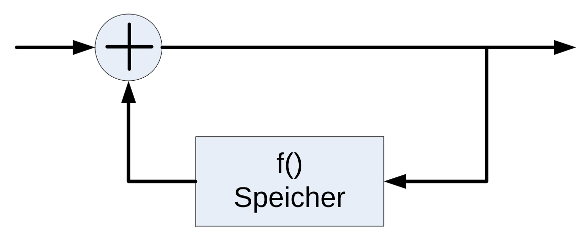
Predict symbol to be coded exploiting the knowledge of neighbored symbols

causal predictor: → Prediction is based on already coded (and known to the receiver) neighbored symbols.

Approach: $$e_i = x_i - \hat{x}_i, \quad \hat{x}_i = h(x_{i-1}, x_{i-2}, x_{i-3}, \ldots)$$



Encoder



Decoder

Linear Prediction

General linear filter h:

$$e[n] = \sum_{i=1}^{p} h_i x[n-i]$$

Optimization criteria:

→ goal: minimizing the prediction error

→ criteria: MSE

$$\mathrm{E}(e^2) \rightarrow \min$$

Solving an equation system (normal equations / Yule-Walker equations)

$$\sum_{i=1}^{p} h_i \varphi_{xx}(i-j) = \varphi_{xx}(j) \quad j = 1 \ldots p$$

Approach to solve the equations → Levinson-Durban algorithm

DPCM (Differential Pulse Code Modulation)

Simplest form of prediction

Linear prediction:

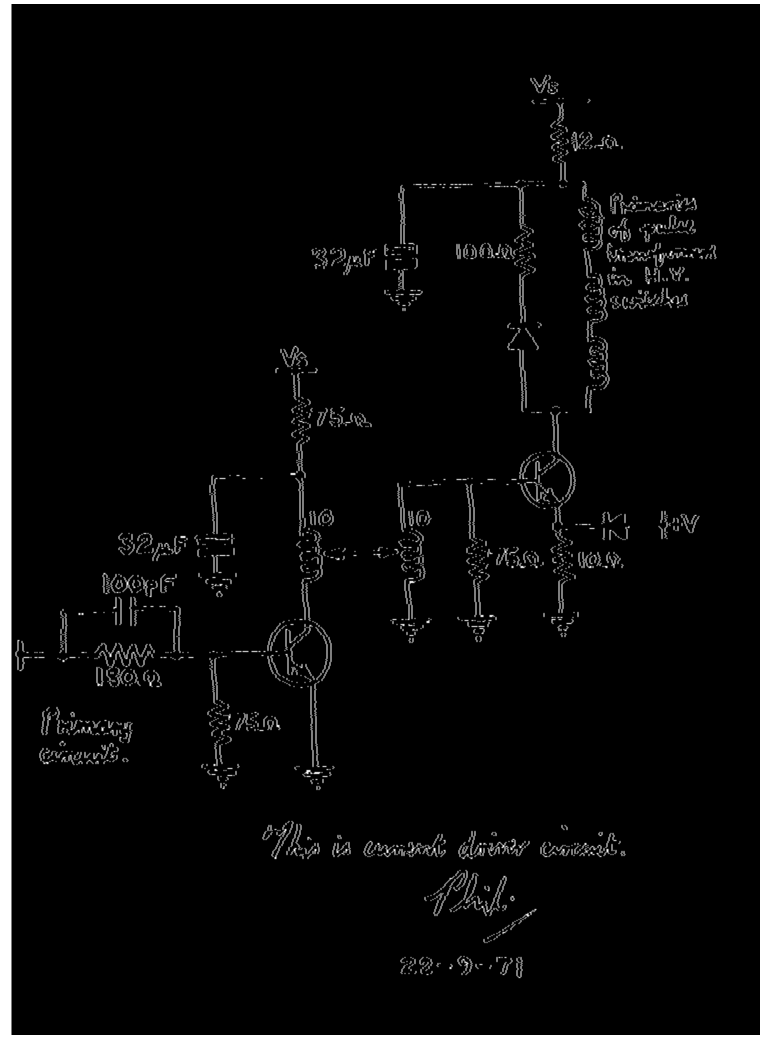
$$e[n] = x[n] - x[n-1]$$

Problem:
Extending the range of the differential signal e

But:
With knowing the range of the original signal

$$x[n] = (0,1)$$

$\rightarrow$ $e[n] = (0,1)$ $\rightarrow$ (equal, unequal)



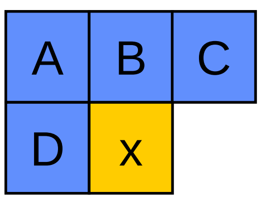
P(0) = 0,983811
P(1) = 0,016189

H = 0,11946   (PCM: H= 0,18576)

Context Dependent Coding

- **Precondition: source symbols are statistically dependent**
- **Based on already sent symbols (thus known to the receiver) a „context“ is created**
- **Coding of a symbol depending on its context**
  - Selecting different code tables dependent on the symbols in the context
  - Code tables are matching the different symbol probabilities

Example: coding a FAX-image



$$H(X|A,B,C,D)$$

(B,D)	p(x=0)
(0,0)	
(1,0)	
(0,1)	
(1,1)	

Dynamic Statistics

Statistics changes during the encoding process

→ dynamically modifying the code being used for encoding

- Modifed table must be transmitted (reduces the gain of modified codes)
- decoder must be able to reconstruct the code based on received symbols

→ adaptive approaches

Huffman Code

- **Permanently updating the symbol frequency in a table**
- **Recalculating the code table based on the symbol frequencies**
  - After each symbol
  - After N symbols
  - . . . . .

Advantage:

higher compression efficiency

Disadvantage:

higher processing power and memory demand for recalculating the code tables in the encoder as well as in the decoder

Arithmetic Coding

Approach:

each (infinitely long) sequence of binary digits represents a number $y \in [0, 1)$

example: 0,101101 = ½ + 0 + 1/8 + 1/16 + 0 + 1/64 = 45 / 64

→ relative frequency sums up to 1
→ complete and non-overlapping partitioning of the interval [0,1)

Principle of intervals → interval partitioning

$$\lceil y \rceil_N = \lfloor y \rfloor_N + 2^{-N}$$

$$\lfloor y \rfloor_N \leq y < \lfloor y \rfloor_N + 2^{-N}$$

If the lower bound of a number (integer) is known,
the upper bound of that number is also known.

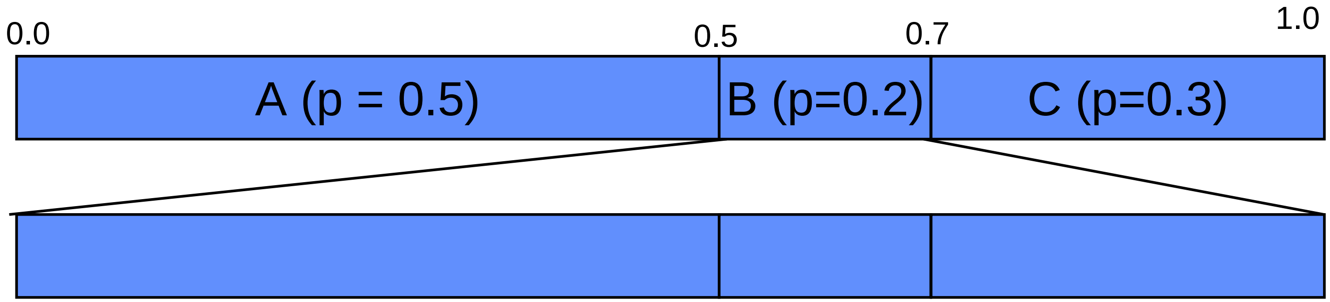
→ every finite sequence of N binary digits uniquely defines an interval of size 2^{-N}

Arithmetic Encoding Procedure

Encoder working principle:

- partition the interval [0,1) according to the probabilities of the symbols
- generate an interval partition using the incoming binary symnols
- the binray sequence of coded symbols representd to LOWER probability bound

Example:



<u>Important</u>: accumulated probabilities

Decoder

- reconstructs the lower and upper interval boundaries

Algorithm

Encoder:

```
Low = 0
High = 1.0
While input symbols available
   get (symbol)
   range = high – low;
   high = low + range * high_range (symbol);
   low = low + range * low_range (symbol)
End of while;
Output low
```

Decoder:

```
Do
   identify interval of present code number
   output corresponding symbol
   range = symbol_high – symbol_low
   number -= symbol_low
   number /= range
While number == 0;
```

Example:

Symbol	Range	Low	High	Channel	Low	High	Range	Number	Symbol
--	--	0	1	--	0	1	1	0,554151	--
B	1	0,5	0,7	01	0.5		0,2	0,054151	B
A	0,2	0,5	0,6	1			0,5		A
C	0,1	0,507	0,6	00			0,3		
A	0,093	0,507	0,554 151	1			0,5		

Advantage of Arithmetic Coding

- Implementation complexity significantly higher than for Huffman-Codes
- No limitations with respect to the length of sequences to be encoded (instantaneous coding)
- Context dependent and adaptive coding extensions do not increase the computational complexity (except for calculating the context and dynamically adapt the context)
- Low coding latency
- High efficiency

$$H \leq R < H + 2/m$$

M: sequence length

Different Coding Formats

- FAX group 3
- FAX group 4
- JBIG
- JPEG lossless
- Lempel-Ziv
- Zero-Tree Coding
- Context adaptive arithmetic coding
  - MQ-Coder (used in JPEG2000)

MQ-Coder

Context dependent binary arithmetic codec

- Coding of binary symbols
- Based on their respective frequency in the coding process
- For each context probability table is tracked
- Updating the frequency using a finite-state engine

Peculiarity:

- Fast adaptation of symbol probabilities employed in coding

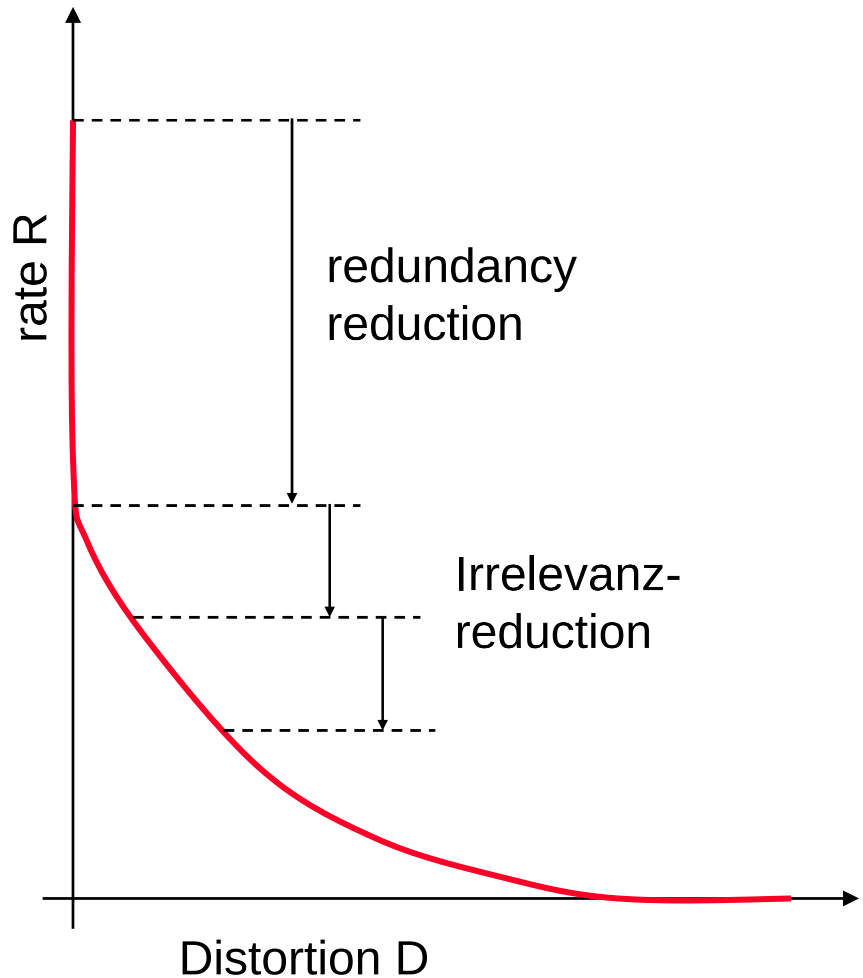
Approach:

- Assigns a context to each individual symbol
- adapt probability with each coded symbol

$$P_1(ctx_i) = (n_1 + 1)/(N + 2) \qquad P_1(ctx_1) = \frac{n_1+1}{N+2}$$

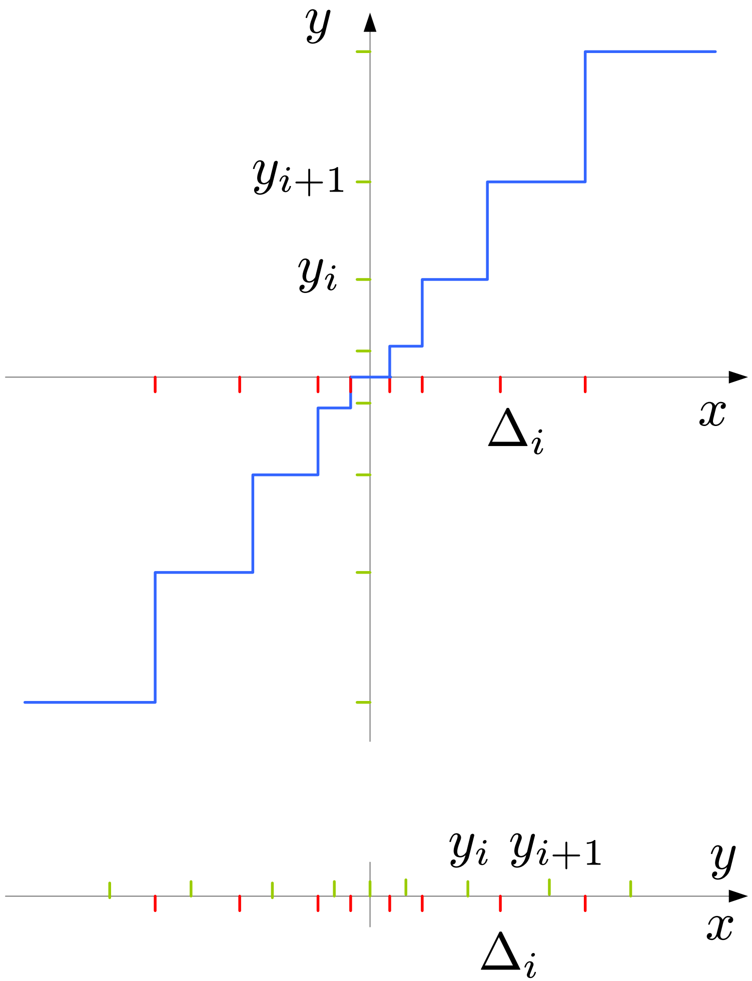
Part: 1.4
Introduction to Quantization

Redundanz und Irrelevanz



- **Analogues audio / video signals**
  Information content → infinite
- **Digital signals**
  – Finite information content
  – Representation > entropy
- **Redundancy reduction**
  – representation ~ entropy
  → Lossless coding
- **Irrelevance reduction**
  – Irrelevance reduction; subjectively not or barely noticeable distortions
  – e.g. by quantization
  → Lossy coding
- **Significant distortions**
  – Determined by application

Scalar Quantization



Principle:

Assign an interval of signal values to a single (replacement) value

Design of a quantizer:

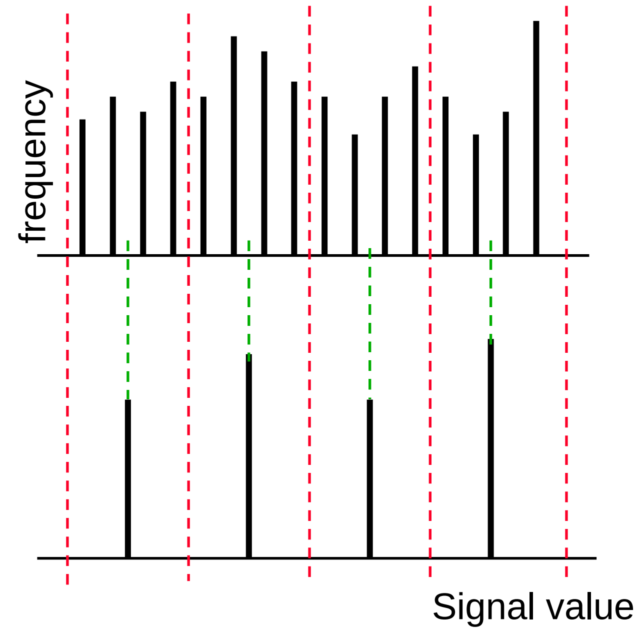
- Split up signal range into intervals
- Determine the replacement value

… under certain design criteria

Methods to visualize quantizers

- 1D along a numbering beam
- 2D as step curve

Consequences of Quantization



PDF – Entropy reduction

- Reduction of possible signal values
- Reduction of entropy
- Quantization results in <u>loss of information</u>

  → Information is not any more contained in the signal

Increase the signal distortion

Distortion: $D = \mathrm{E}\{f(X - Y)\}$

Mean Square Error: $D = \mathrm{E}\{(X - Y)^2\}$

SNR: $\mathrm{SNR} = \log_{10} \frac{P_x}{D} \quad [dB]$

PSNR: $\mathrm{PSNR} = \log_{10} \frac{\max P_x}{D} \quad [dB]$

Quantizer Design

Distortion depends on PDF of quantization errors signal:

$$D = \sum_i p_i e_i^2$$

Selecting the "optimal" replacement value

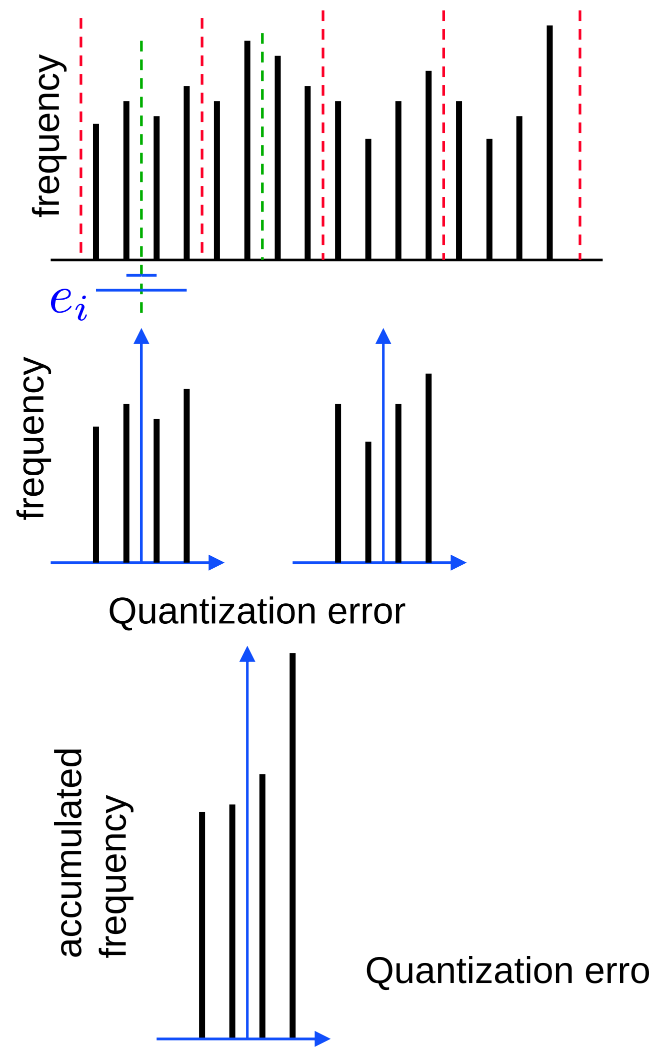
(using an uniform interval partition)

criteria: minimize the MSE

→ Llyod-Max Quantizer

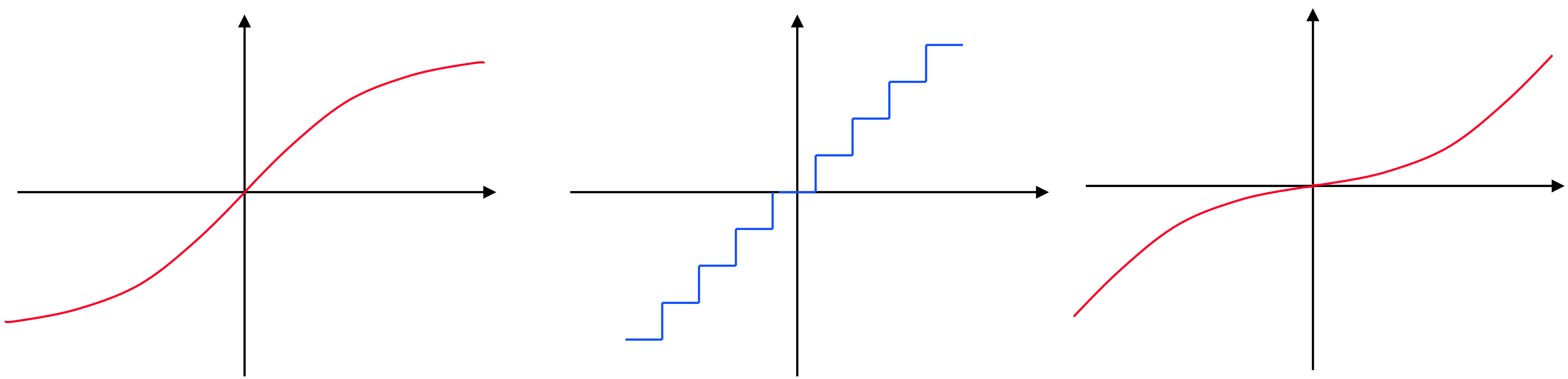
$$y_k = \frac{\sum_{i \in \Delta_k} x_i p_i}{\sum_{i \in \Delta_k} p_i}$$

Replacement value in the center of mass of p(x) . of the area spanned over the interval



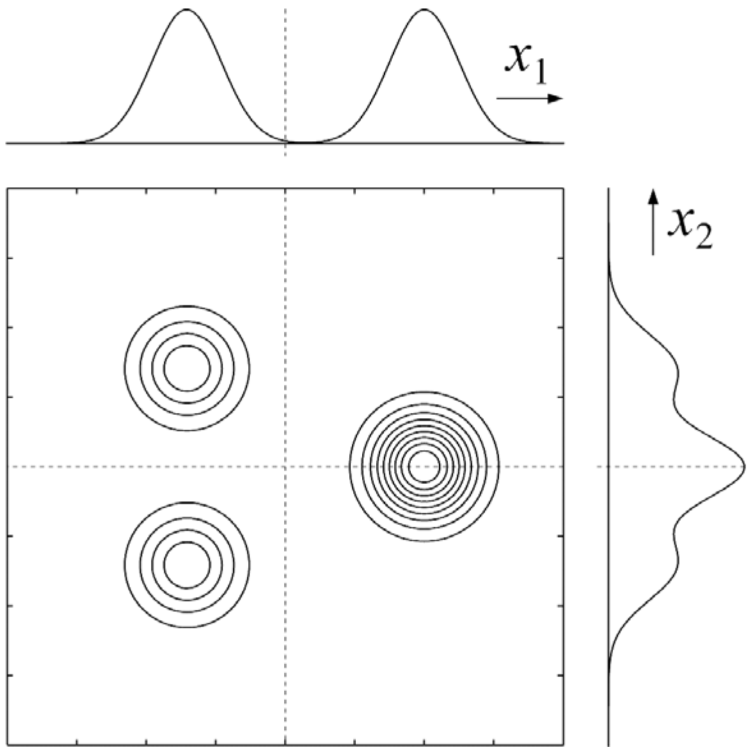
Quantiser Types

- **Mid-thread**
  - Replacement value for value 0
- **Mid raise**
  - No replacement value for 0
- **Linear**
  - equally sized interval partitioning for the entire range
- **Non-linear → utilizing a compander**
  - Implementing a non-linear quantizer
  - Optimal NL-Quantiser: each interval contributes approximately the same amont to the overal distrotion Di = 1/N D



Statistical Dependency and Joint Probability Distribution

Source: Stefan Simon, IENT RWTH Aachen 2000

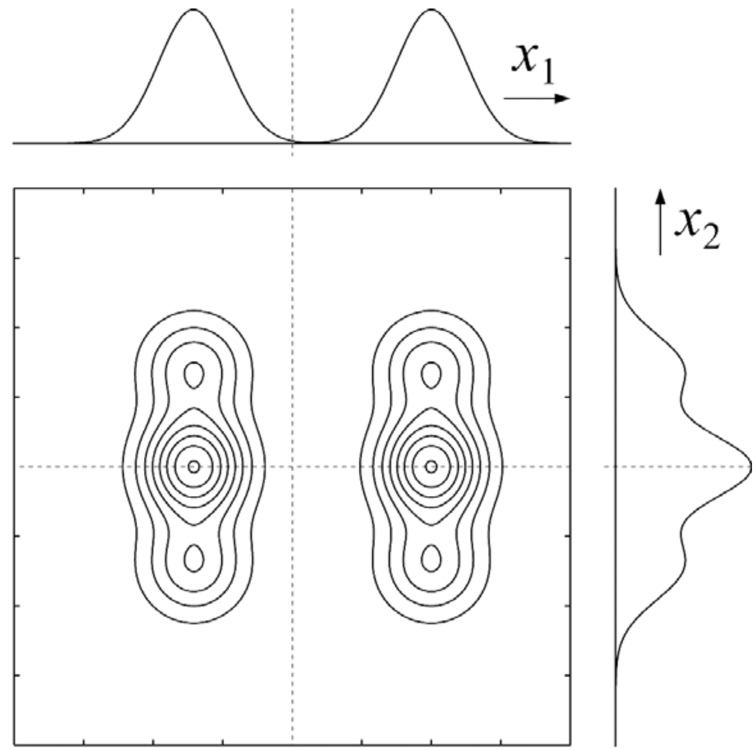


$X_1, X2$ are uncorrelated

$\mathrm{E}\{X_1, X_2\} = m_1 \cdot m_2 = 0$

$X_1, X2$ Is statistically dependent

$p(x_1, x_2) \neq p_{X_1}(x_1) p_{X_2}(x_2)$



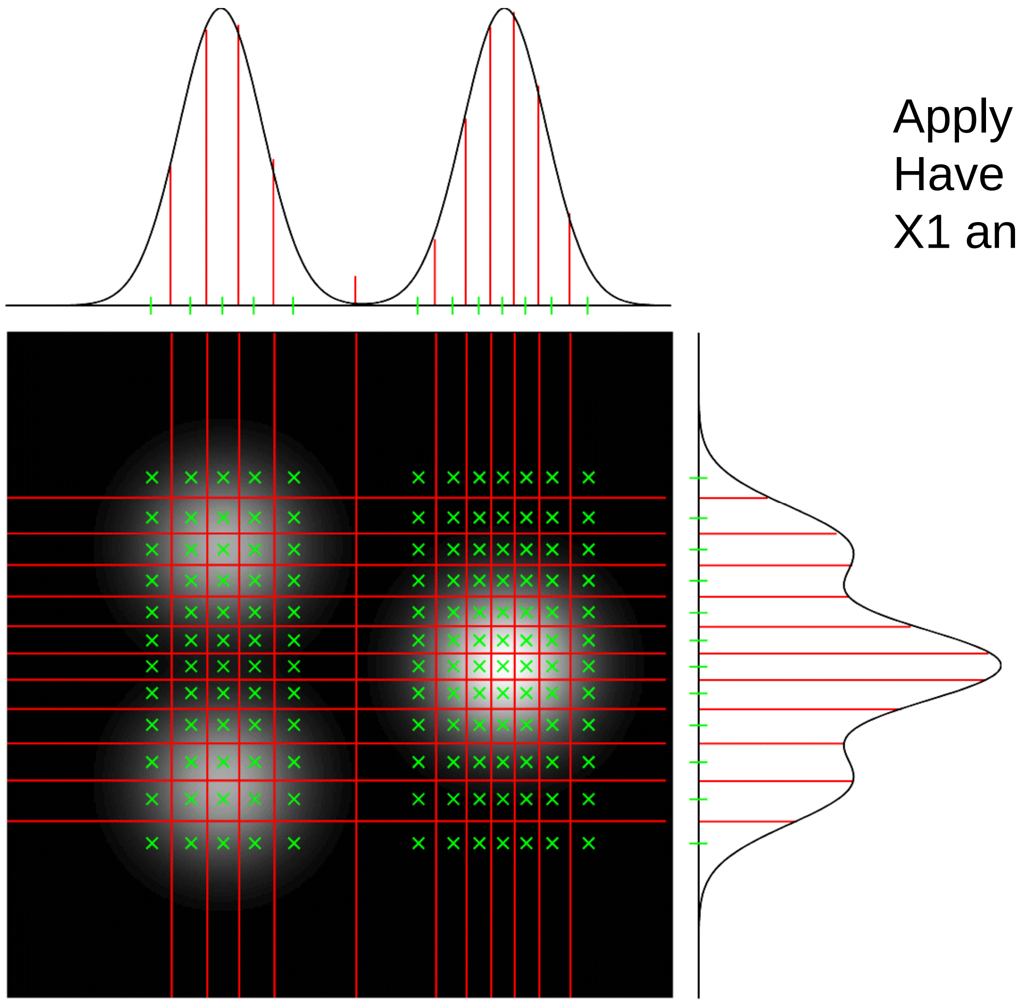
$X_1, X2$ are uncorrelated

$\mathrm{E}\{X_1, X_2\} = m_1 \cdot m_2 = 0$

$X_1, X2$ are statistically independent

$p(x_1, x_2) = p_{X_1}(x_1) p_{X_2}(x_2)$

Vector Quantization (1)



Apply the scalar Llyod quantizers which Have individually beein optimized for X1 and X2

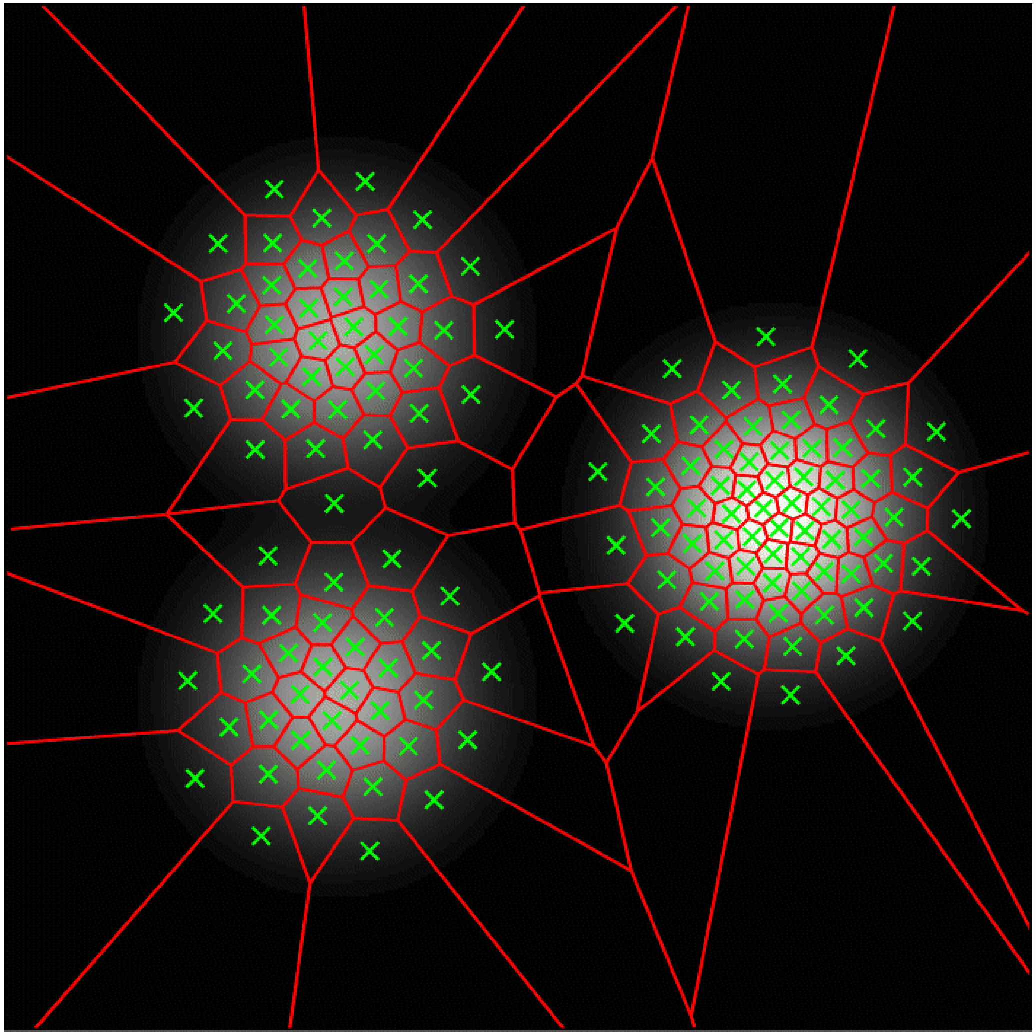
cross: replacement vectors

lines: partitions

→ bad adaptation to joint density

Source: Stefan Simon, IENT RWTH Aachen 2000

Vector Quantization (2)



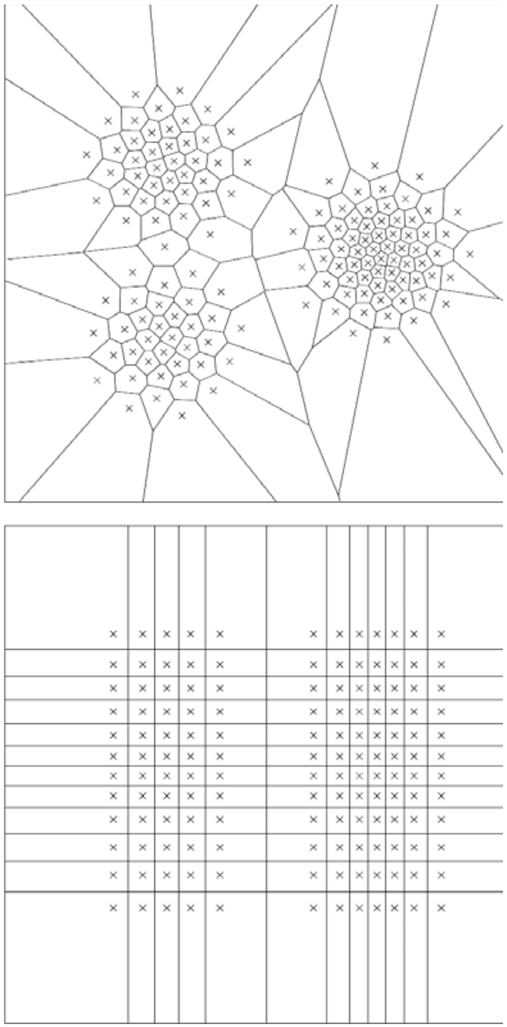
Quelle: Stefan Simon, IENT RWTH Aachen 2000

Partition optimized with
LBG-Algorithm for
144 replacement vectors
(criterion: MSE)

→ cells adjust to joint density distribution

→ improving the SNR by 1.96 dB with identical code book size

Aspects of Vector Quantization



Vector quantization assigns

- Individual regions,
- Represented by a single replacement vector,
- A binary code word.

All regions together assemble a non-uniform partition.

The code book design determines the optimal partition and identifies the optimal replacement vectors for each cell.

The vectors are collected in code books.

Summary

The basic for coding of 1D / 2D signals have been introduced and the following processing seps have been explained:

Sampling and aliasing
Quantization
Filtering
Transformation
Prediction
Entropy coding

The following sections introduce concrete coding concepts and algorithms for speech, audio, still images and video.