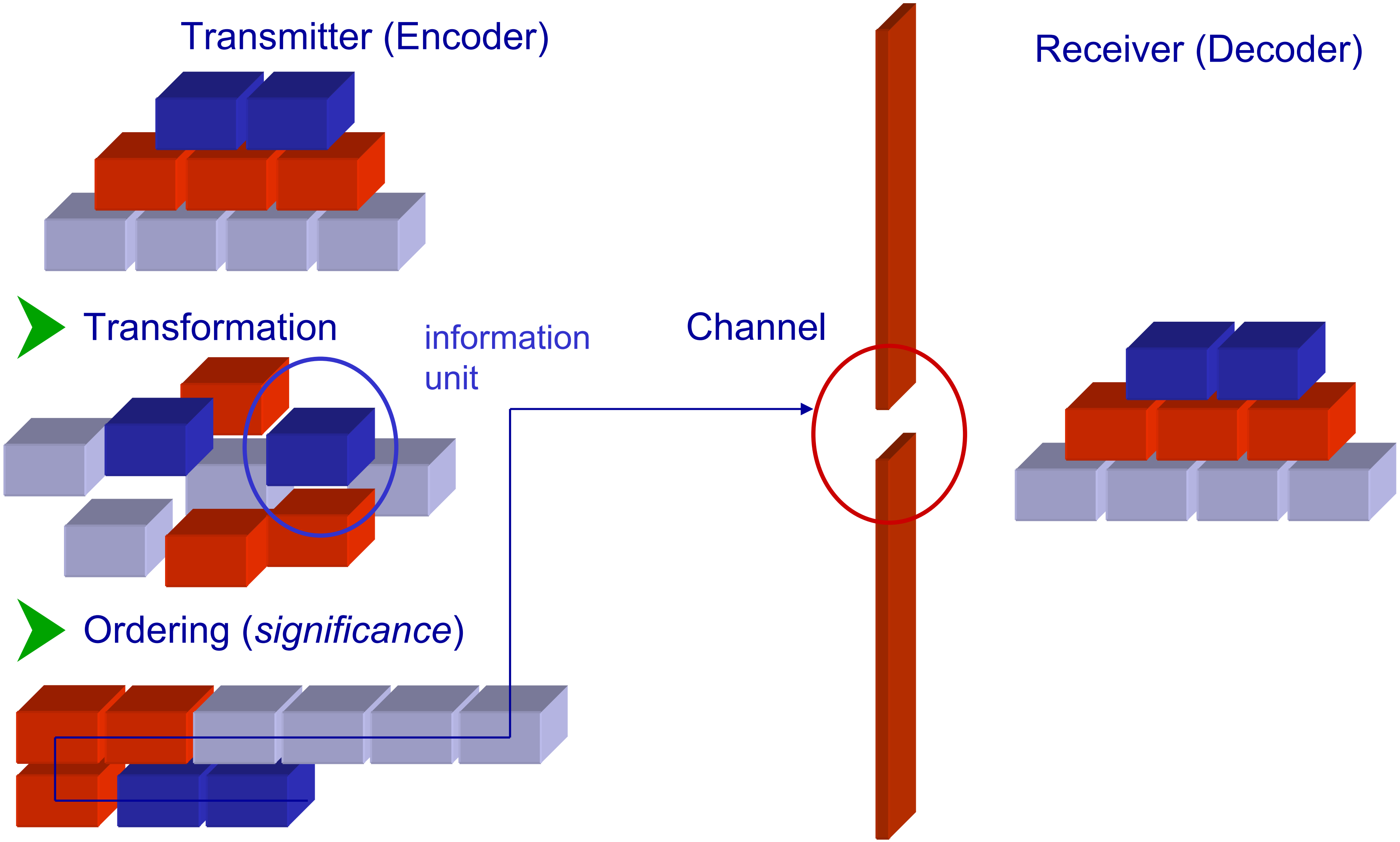
Compression and Coding

Theory and Applications

Part 1: Fundamentals

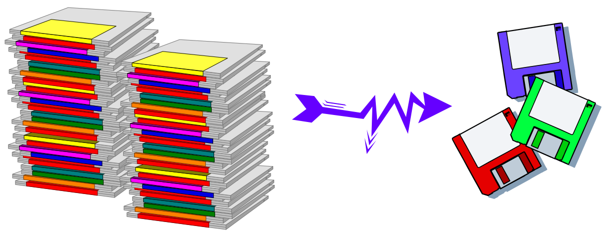
What is the problem?



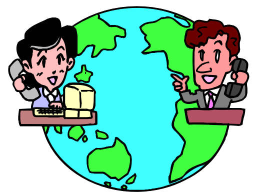
Why is it important?

- The available resources for signal communication and archiving are limited

Compression

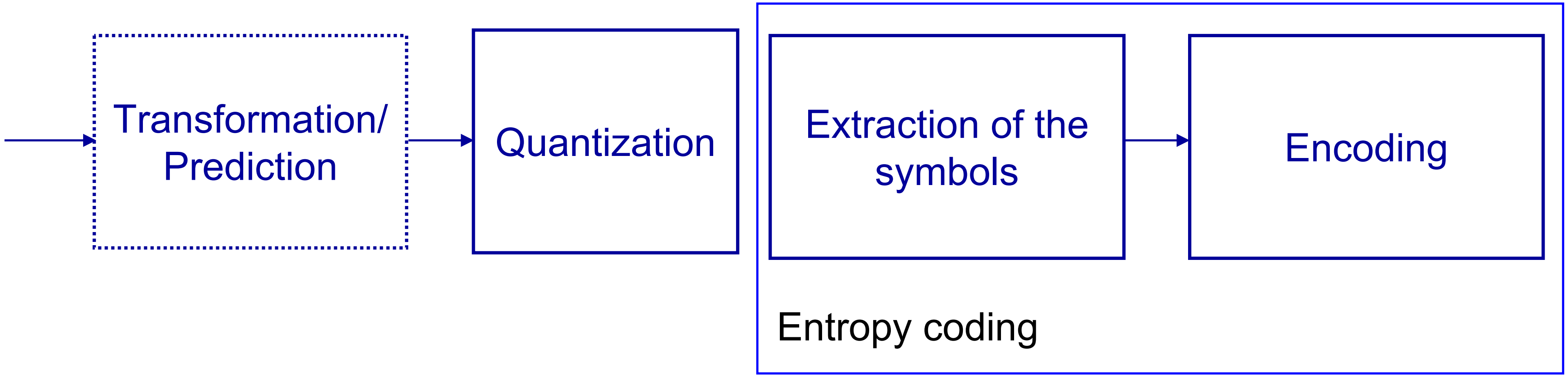


Standardization



Basic steps

- Goal: minimize the amount of resources needed to transmit a source signal from the transmitter to the receiver
- Basic steps:
  - Reduction of the redundancy in the data
    - Transform-based coding
    - Prediction-based coding
  - Translate the resulting information from to a sequence of *symbols* suitable for encoding
  - *Entropy coding* of the sequence of symbols



Basic idea

- Exploit the redundancy among the data samples for an *effective* representation of the data
- Classical coding schemes
  - Look at the data as to set of numbers and reduce the mathematical and/or statistical redundancy among the samples
    - JPEG, MPEG
- Second generation coding schemes
  - Adapt the coding scheme to the different image regions featuring some omogeneity for optimizing the coding gain given the data
    - ROI based coding, JPEG2000
- Model-based coding
  - Look at the data as to perceptual information and exploit the way such information is processed by the sensory system to improve compression

Compression modes

- Lossless
  - The original information can be recovered without loss from the compressed data
  - Low compression factors
    - Less than a factor 3 for natural images
- Lossy
  - The compression process implies the loss of information that cannot be recovered at the decoding
  - Basically due to quantization
  - Very high compression factors
  - Degradation of the perceived quality

$\Rightarrow$ Key point: rate/distortion tradeoff

Information theoretical limits

- Noisy channel coding theorem
  - Information can be transmitted reliably (i.e. without error) over a noisy channel at any source rate, R, below a so-called *capacity C of the channel*

$$R<C \text{ for reliable transmission}$$

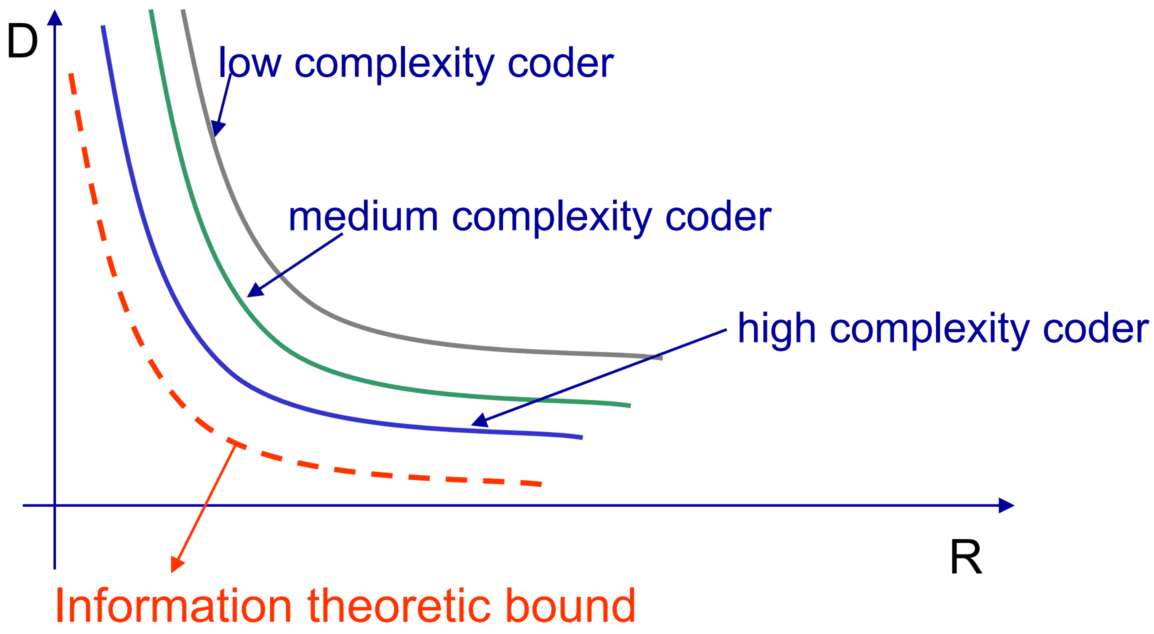
- Source coding theorem
  - There exists a map from the source waveform to the codewords such that for a given distortion D, $R(D)$ bits (per source sample) are sufficient to enable waveform reconstruction with an average distortion that is arbitrarily close to D. Therefore, the actual rate R has to obey:

$$R \geq R(D) \text{ for fidelity given by } D$$

R(D): rate distortion function

Qualitative *R(D)* curves

- *R(D)* curves are monotonically no-increasing
  - Noteworthy points
    - R(0): rate needed for exact reproduction of the source⇔ *entropy* of the source
    - Ropt, Dopt: minimum rate for a given distortion / minimum distortion at a given rate



Entropy Coding

Fundamentals

Information

- Information

  Let X be a Random Variable (RV) and s be a realization of X. Then, the information hold by symbol s can be written as

$$I(s) = -\log_2 p(s)$$

  where $p(s)$ is the probability of the symbol s.

  – *$I(s)$ represents the amount of information carried by the symbol s.*
    - *p(s)=1 → There is no uncertainty on the expectation on value taken by the RV → no information is conveyed by the knowledge of the actual value of the RV (current realization). This is expressed by the corresponding information being zero → I(s)=0*
    - *p(s)<< (very small) → the value s is highly improbable → it corresponds to a rare event → knowing that the current realization of the RV is equal to s is highly informative, as an indication of a rare event. This is expressed by the corresponding information being very high in value I(s) → infinity*
    - *Summary: symbols that are **certain** convey **no information**, while **very improbable** symbols are **highly informative***

Information

- Discrete time sources
  - Let X be a discrete time ergodic source generating the sequences $\{x_k\}_{\{k=1,K\}}$ of *source symbols*.
    - The sequences are realizations of the RV $\{X\}$
    - The source is *memoryless* if successive samples are *statistically independent*
  - *Information*

$$I_k = -\log_2 p_k = -\log_2 p(x_k)$$
$$p(x_k) = 1 \rightarrow I_k = 0$$
$$p(x_k) << 1 \rightarrow I_k \rightarrow \infty$$

Information

- Relation to uncertainty

  If the K symbols have the same probability $$p_k = \frac{1}{K}$$

  Then the information is

$$I_k = -\log_2 \frac{1}{K} = \log_2 K$$

In this case, the *uncertainty* on the expectation is *maximized*, because all the symbols are equally probable.

The amount of information is the *same* for all symbols

Same probability ↔ Maximum uncertainty

Entropy

- Entropy

  Let X be a discrete RV: $\{x_k\}_{\{k=1,K\}}$. Then, the ***entropy*** is defined as

$$H(X) = \sum_{k=1}^{K} p_k I_k = -\sum_{k=1}^{K} p_k \log_2 p_k$$

$$p_k = p(x_k)$$

  – H(X) represents the ***average information content of the source*** (or the average information conveyed by the RV)
  – Symbols with **same probability** (<u>maximum uncertainty</u>)

$$H(X) = \sum_{k=1}^{K} p_k I_k = \sum_{k=1}^{K} \frac{1}{K} \log_2 K = K \frac{1}{K} \log_2 K = \log_2 K$$

  – It can be shown that this corresponds to the **upper bound**

$$0 \le H(X) \le \log_2 K$$

Entropy

- Summary
  - The entropy represents the average information conveyed by the source RV
    - H(X) is the average information received if one is informed about the value of the RV X has taken
  - The entropy *increases with the degree of uncertainty* on the expectation of the realizations of the RV
    - Equivalently: it is the uncertainty about the source output before one is informed about it
  - All the discrete sources with a ***finite*** number K of possible amplitudes have a finite informational entropy that is no greater than $\log_2 K$ **bits/symbol**

$$0 \leq H(X) \leq \log_2 K$$

    - The right side equality holds if and only if all probabilities are equal (most unpredictable source)
    - **Due to unequal symbol probabilities and inter-symbol dependencies $H(X)$ will in general be lower than the bound value**

- **Entropy coding** exploits unequal symbol probabilities as well as source memory to realize average bit rates approaching $H(X)$ bits/symbol

Entropy coding

- Goal: Minimize the number of bits needed to represent the values of X.
  - We consider the codes that **associate** to each **symbol** x_k a **binary word** w_k of **length** l_k.
  - A sequence of values produced by the source is coded by aggregating the corresponding binary words.
- Bit-rate
  - The *average* bit-rate to code each symbol emitted by the source is

$$R_X = -\sum_k l_k \log_2 p_k$$

  - Goal: optimize the codewords to **minimize R_X**

Shannon theorem

- The Shanno theorem proves that the entropy is a *lower bound* for the average bitrate R_X of a prefix code
- The *average rate* of a prefix code satisfies

$$R_X \geq H(X) = -\sum_k p_k log_2 p_k$$

Moreover, there exists a prefix code such that

$$R_X \leq H(X)+1$$

  - The lower bound is set by the entropy of the source
  - We cannot do better than reaching the entropy of the source
- Redundancy:

$$R(X)=log_2 K-H(X)$$

Entropy coding policies

- Fix and variable length codes
  - Fix length codes: If $\log_2 K$ is an integer, all symbols could be coded with words of the same length $l_k = \log_2 K$ bits.
  - Variable length codes: the average code length can be reduced by using *shorter* binary codewords for symbols that occur *frequently*.

$$p_k \text{ large} \rightarrow \text{short codewords}$$
$$p_k \text{ small} \rightarrow \text{long codewords}$$

- Variable Length Codes (VLCs)
  - Prefix codes
    - Huffman coding
    - Arithmetic coding

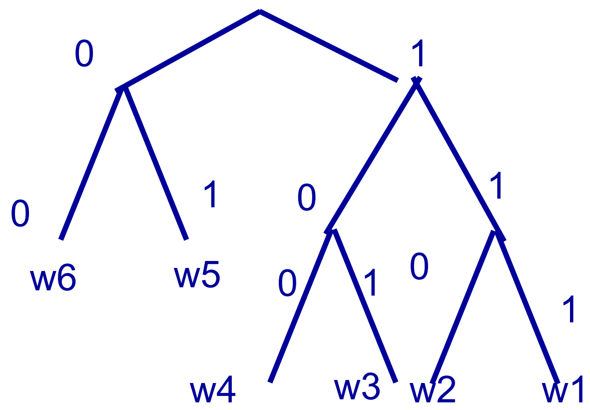
Prefix codes

- To guarantee that any aggregation of codewords is *uniquely* decodable the *prefix condition* imposes that *no codeword may be the prefix (beginning) of another one*
- Example

  {w1=0, w2=10, w3=110, w4=101}

  → 1010 can be read as both w2w2 and w4w1: ambiguous!

→ Prefix codes are constructed by building binary trees



Huffman code

- Optimal prefix code tree
  - rate approaching the lower bound
- Each symbol is represented by a codeword whose length gets longer as the probability of the symbol gets smaller
- Dynamic programming rule that constructs a binary tree from bottom up by successively aggregating low probability symbols

Let us consider K symbols with their probability of occurrence sorted by **increasing** order $p_k \leq p_{k+1}$

$$\{(x_1,p_1),(x_2,p_2),\ldots,(x_K,p_K)\}$$

we aggregate x_1 and x_2 in a single symbol of probability $p_{12}=p_1+p_2$.

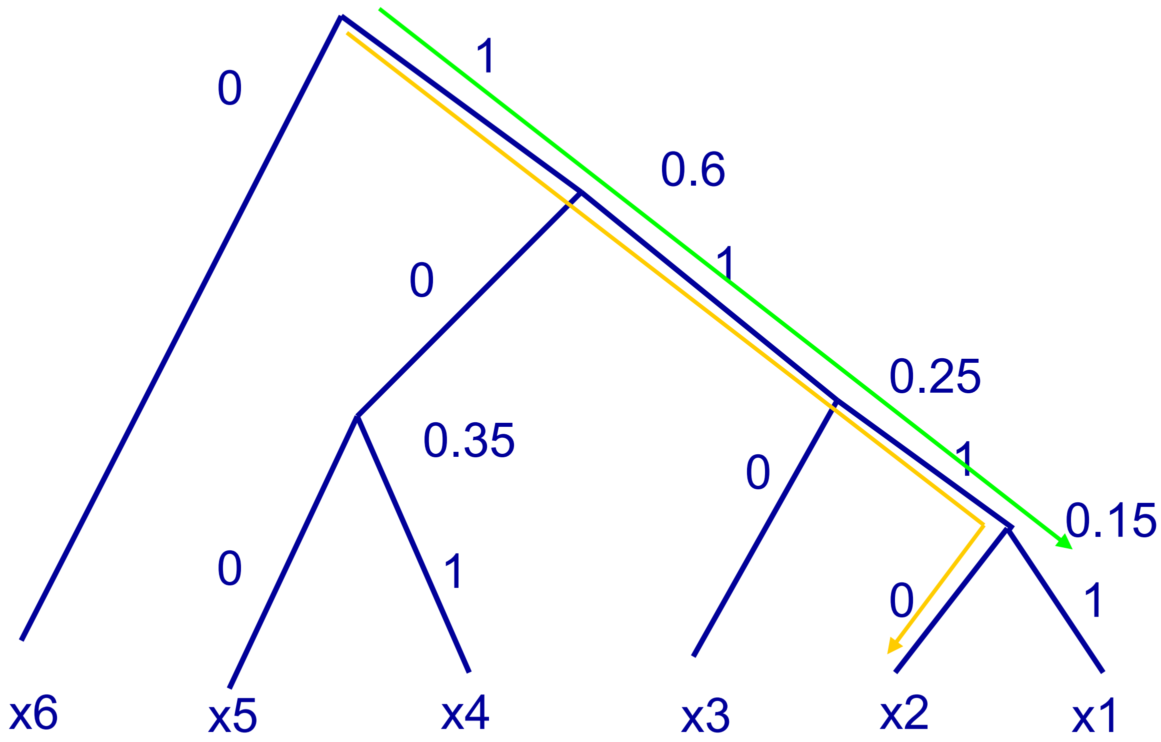
Recursivity: An optimal prefix tree for K symbols can be obtained by constructing an optimal prefix tree for the K-1 symbols

$$\{(x_{12},p_{12}),(x_2,p_2),\ldots,(x_K,p_K)\}$$

and by dividing the leafs of p_{12} in two children corresponding to x_1 and x_2

Huffman code

- Example
  - {p1=0.05, p2=0.1, p3=0.1, p4=0.15, p5=0.2, p6=0.4}



x1	1111
x2	1110
x3	110
x4	101
x5	100
x6	0

Arithmetic coding

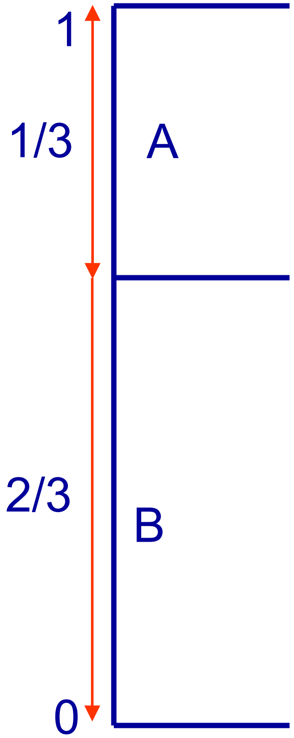
- The symbols are on the number line in the probability interval 0 to 1 in a sequence that is known to both encoder and decoder
- Each symbol is assigned a sub-interval equal to its probability
- Goal: create a codeword that is a *binary fraction* pointing to the interval for the symbol being encoded
- Coding additional symbols is a matter of subdividing the probability interval into smaller and smaller sub-intervals, always in proportion to the probability of the particular symbol sequence

Arithmetic coding

- Example

  p(A)=1/3

  p(B)=2/3

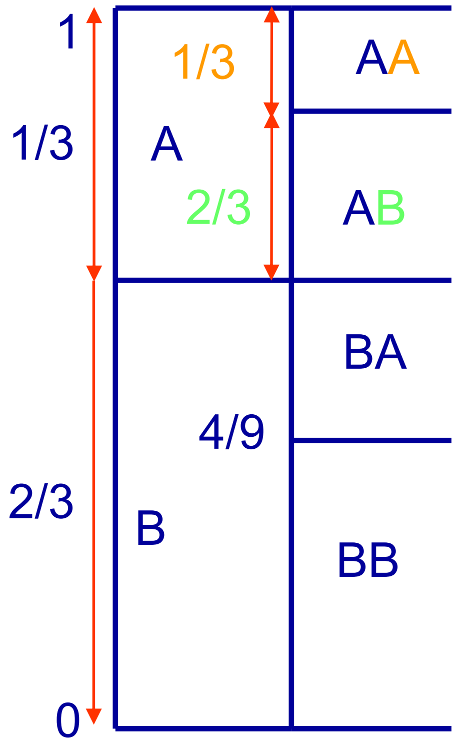


Arithmetic coding

- Example

  p(AA)=1/3*1/3=1/9 p(BA)==1/3*2/3=2/9

  p(AB)=2/9 p(BB)=4/9



Arithmetic coding

- After encoding many symbols
  - the final interval *width* P is the *product* of the probabilities of all symbols coded;
  - the interval *precision*, the number of bits required to express an interval of that size, is given approximately by $-log_2(P)$.

  Therefore, since

$$P=p_1*p_2*....*p_N$$

  the number of bits of precision is approximately

$$-\ log_2(P)= -(log_2(p_1)+log_2(p_2)+....+\ log_2(p_N))$$

  thus *the codestream length will be very nearly equal to the information for the individual symbol probabilities*, and the average number of bits/symbol will be very close to the bound computed from the entropy.

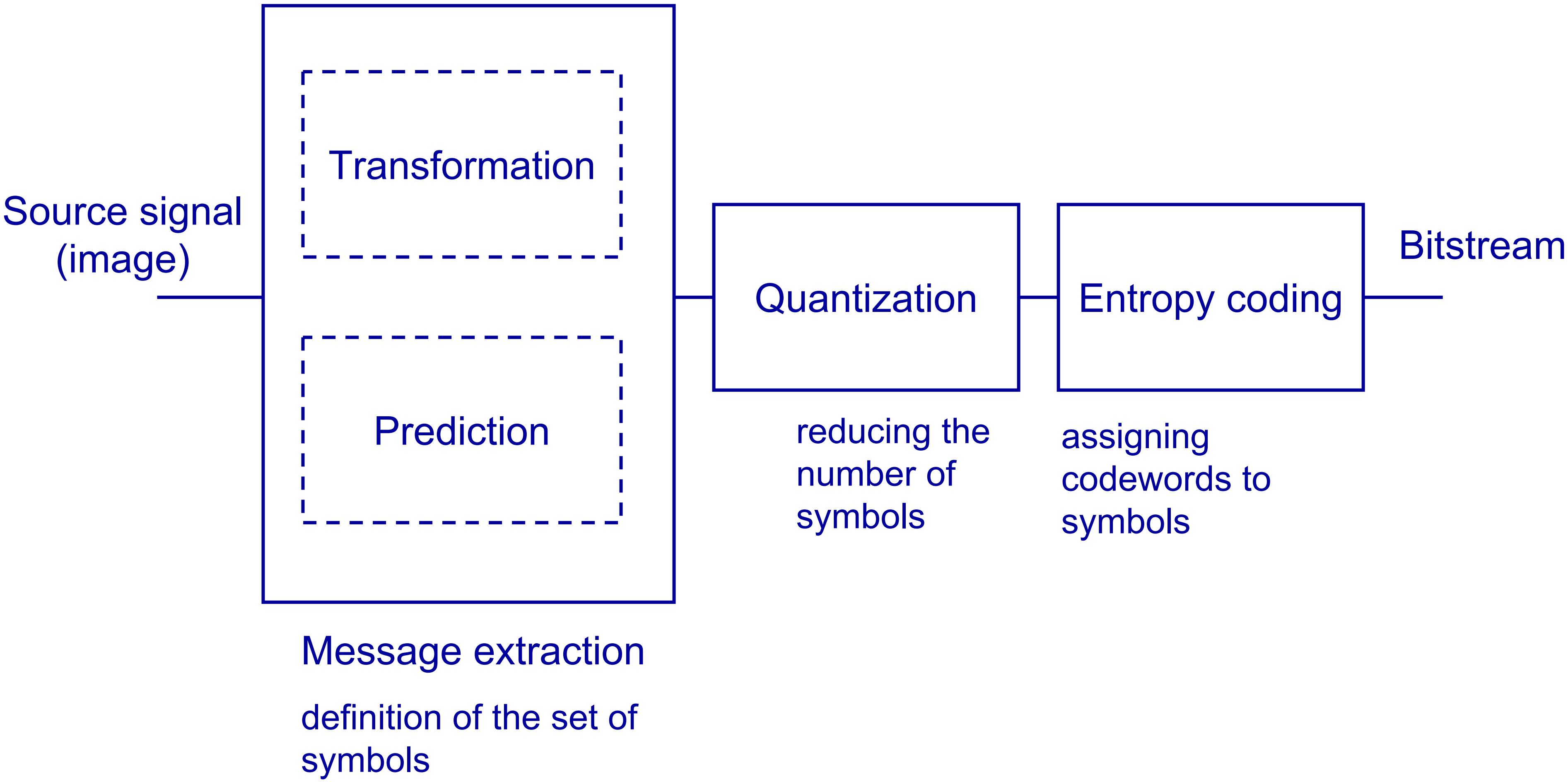
- *Adaptive* arithmetic coding
  - The probability tables for the different symbols can be made adaptive to the source statistics and updated during encoding

Arithmetic coding

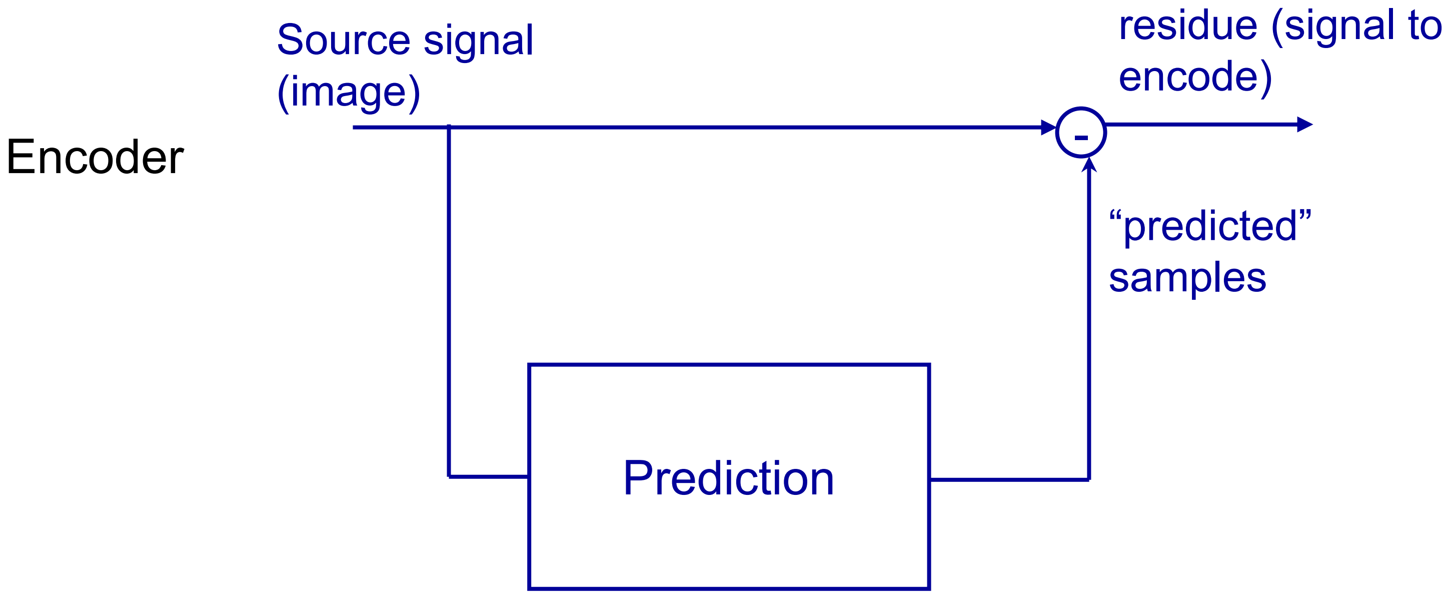
- Features
  - Does not require integer length codes
  - Encodes sequences of symbols
  - Each sequence is represented as an interval included in [0,1]
  - The longer the sequence, the smaller the interval and the larger the number of bits needed to specify the interval
  - The average bit rate asymptotically tends to the entropy lower bound when the sequence length increases

  - On average, performs better than Huffman coding
  - Moderate complexity
  - Used in JPEG2000

Coding systems



Prediction based coding

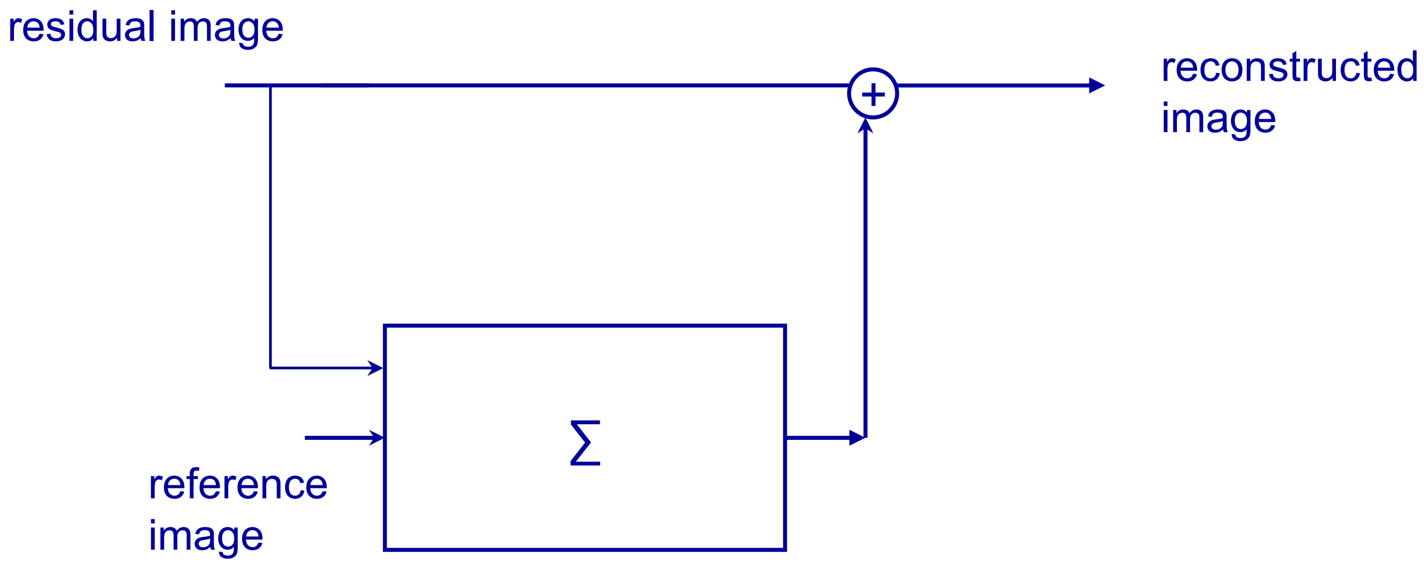


The value of the samples are estimated according to a predefined rule and the resulting values are **subtracted** from the corresponding ones in the original image to obtain the **residual** (or error) image. This last one is then quantized and entropy coded.

- Still images → spatial (intra-frame) prediction
- Image sequences → temporal (inter-frame) prediction

Prediction based coding

Decoder



- Still images (JPEG lossless)
- Image sequences : motion compensation (MPEG4)

Intra-frame *linear* prediction

12	34	27	42
21	3	44	1
12	34	27	42

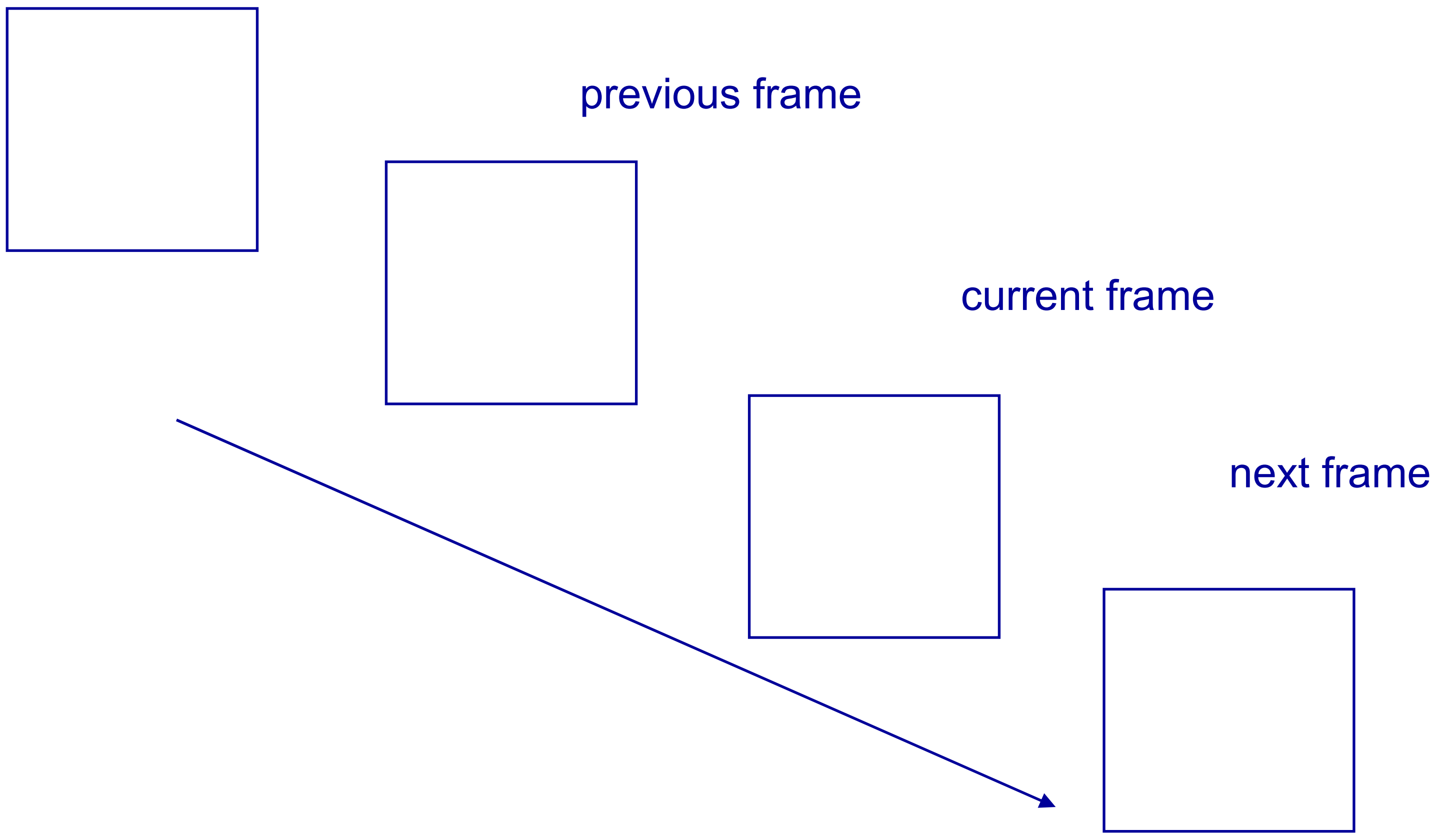
A	B	C
D	X	

$X_{est} = aA+bB+cC+dD$
$E = X - X_{est}$

symbol to predict or estimate

The error image is quantized and entropy encoded. At the receiver, it is decoded and used to recover the original image.

Inter-frame prediction



Temporal redundancy

Transform based coding

- Given the source signal, il can be convenient to project the data to a different domain to improve compression $\Rightarrow$ transformation
  - Discrete Cosine Transform (DCT), used in JPEG
  - Discrete Wavelet Transform (DWT), used in JPEG2000
- The transformed coefficients are then to be quantized for mapping to a finite set of symbols
- Such symbols can also be mapped to another set of symbols to further improve compression performance
  - Embedded Zerotree Wavelet based coding (EZW)
  - Layered Zero Coding (LZC)
  - Multidimensional LZC (for volumetric data, after a 3D DWT)

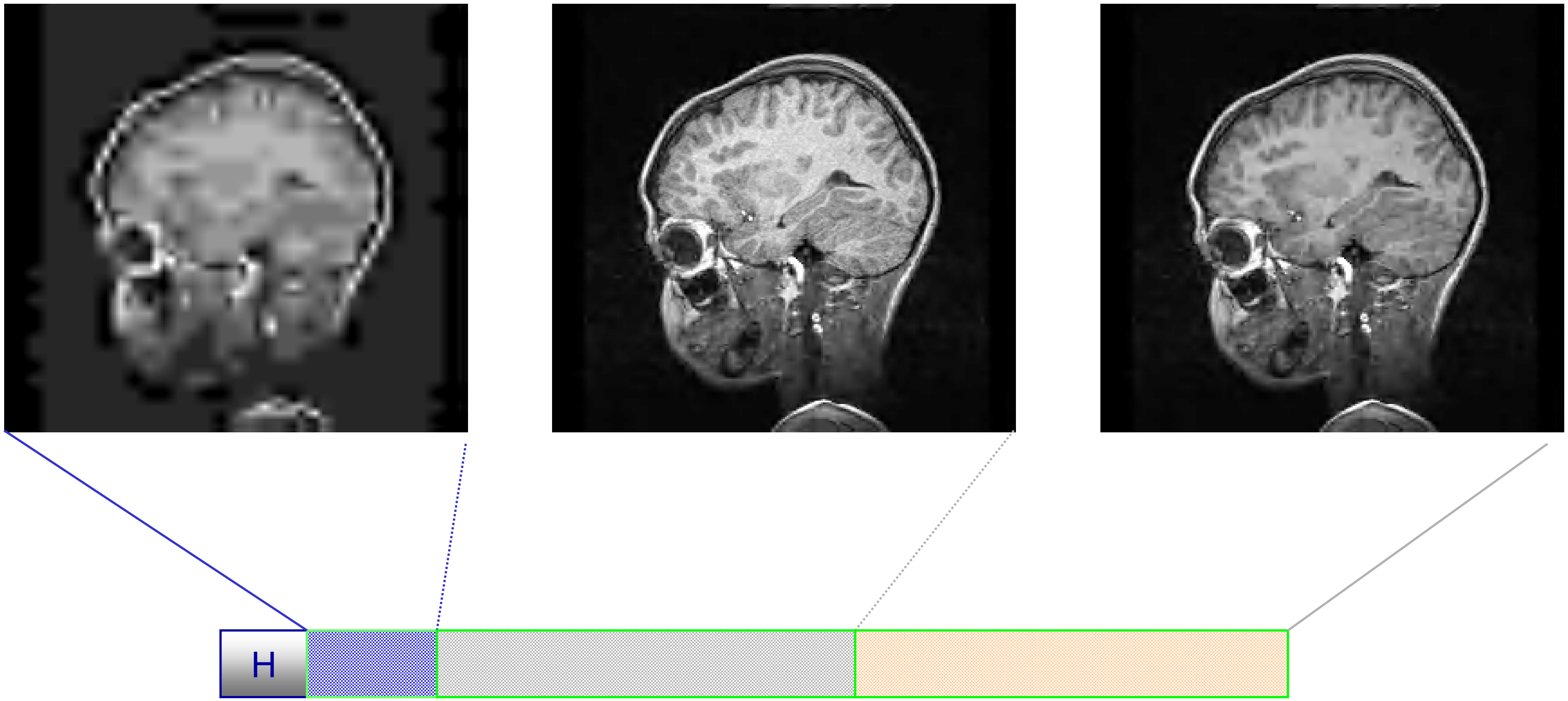
Coding artifacts at low rates

Original

JPEG

Wavelets

Scalability by quality



Scalability by resolution

Object-based processing