Spoken Language Translation
Research Laboratories

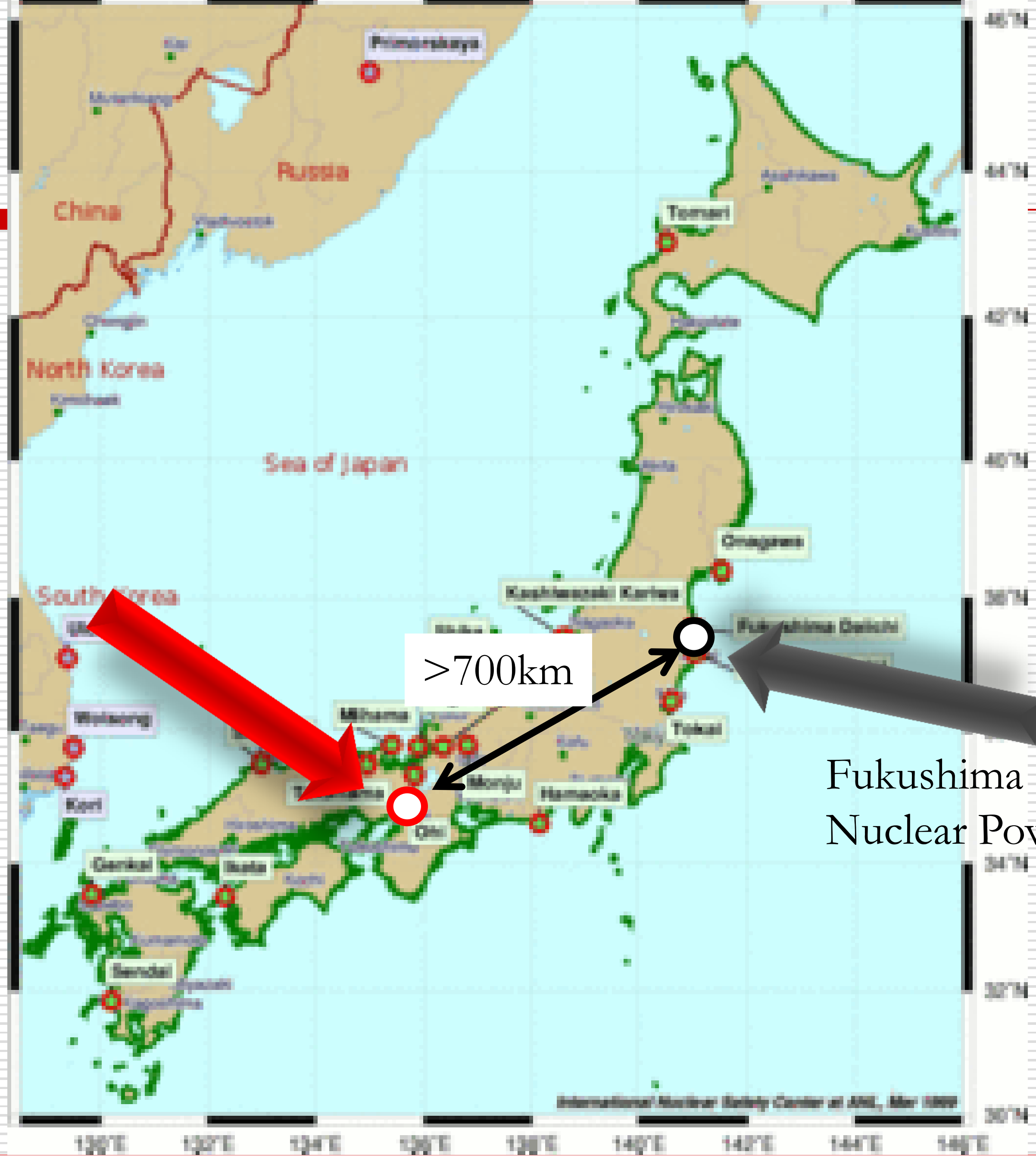
Information Theory: Source Coding

Dr. Satoshi Nakamura

Honorarprofessor, Karlsruhe Institute of Technology, Germany
Professor, Graduate School of Information Science, Nara Institute of Science and Technology, Japan
Invited Advisor, National Institute of Information and Communications Technology, Japan
Fellow, Advanced Telecommunication Research Institute International, Kyoto, Japan

s-nakamura@is.naist.jp

NAIST?



Kenhanna Science City
(NAIST, ATR, NICT,
NTT, NEC…)

Fukushima
Nuclear Power Plant

Acknowledgements of appreciation to the help and supports.

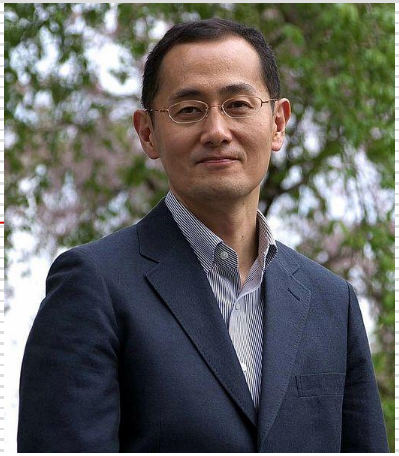
43 rescue members and 4 rescue dogs

Daimler Special Purpose Car
(Total 4M Euro)

Example: Deutsche Bank $2640,000 and more.

Deeply thank you !

Congratulations, Shinya Yamanaka on **Nobel Prize in Physiology or Medicine**



- Education
- He received his M.D. at Kobe University in 1987 and his Ph.D. at Osaka City University Graduate School in 1993.

- Professional career
  - Between 1987 and 1989, Yamanaka was a Resident in orthopedic surgery at the National Osaka Hospital.
  - During 1993–1995, he was a Postdoctoral Fellow at the Gladstone Institute of Cardiovascular Disease, which is affiliated with the University of California, San Francisco.
  - During 1995–1996, he was a staff research investigator at the UCSF-affililated Gladstone Institute of Cardiovascular Disease.
  - Between 1996 and 1999, he was an assistant professor at Osaka City University Medical School.
  - During 1999–2003, he was an associate professor at the Nara Institute of Science and Technology. During 2003–2005, he was a professor at the Nara Institute of Science and Technology. Between 2004 and 2010, Yamanaka was a professor at the Institute for Frontier Medical Sciences.[9]
  - Currently Yamanaka is the director and a professor at the Center for iPS Cell Research and Application in Kyoto University, Japan.
- In 2006, he and his team generated Induced Pluripotent Stem Cells – pluripotent stem cells from adult mouse fibroblasts. In 2007, he and his team were able to generate Induced Pluripotent Stem Cells from human adult fibroblasts.[1][2][3]

NAIST?

Kenhanna Science City (NAIST, ATR, NICT, NTT, NEC…)

About NAIST ?

- Nara Institute of Science and Technology, Japan established 1991.
  - Japanese national university for basic research and higher education.
  - 1st rank research evaluation among Japanese universities in #papers, #grand per faculty.
  - Three graduate schools (No undergraduate school)
    - Information Science
    - Biological Science: Prof. Yamanaka IPS Cell.
    - Material Science
  - Sister school: JAIST, Japan Advanced Institute of Science and Technology

- Graduate School of Information Science
  - 20 laboratories
  - 10 collaborative laboratories
    (ATR, AIST, NEC, Panasonic, NTT, NICT, Fujitsu, Docomo, OMRON)

NAIST Ranking

- **Overall : Ranked 1st**
  Highest Evaluated University in Japan
  based on data in Thomson Reuters"“Essential Science Indicators”and published in“University Ranking 2010”by the leading Japanese newspaper“Asahi Shimbun”
- in the top **5% A＋**
  Three research areas in the Graduate School of Information Science received top scores in a survey conducted by the Ministry of Economy, Trade and Industry
- **Ranked 1st** in “Research”and“Education”among all national universities in Japan published in the weekly magazine “Toyo Keizai”.
- Number of Grants-in-Aid for scientific research **Ranked 1st** per faculty member*
- Grants-in-Aid for scientific research **Ranked 1st** per faculty member*

National Institute of Information and Communications Technology, NICT

About NICT ?

National Laboratory

1952
Radio Research Lab

1988
Communications
Research Lab

2001
Communications
Research Laboratory

Incorporated Administrative Agency

1979
Telecommunications
and Broadcasting
Satellite Organization

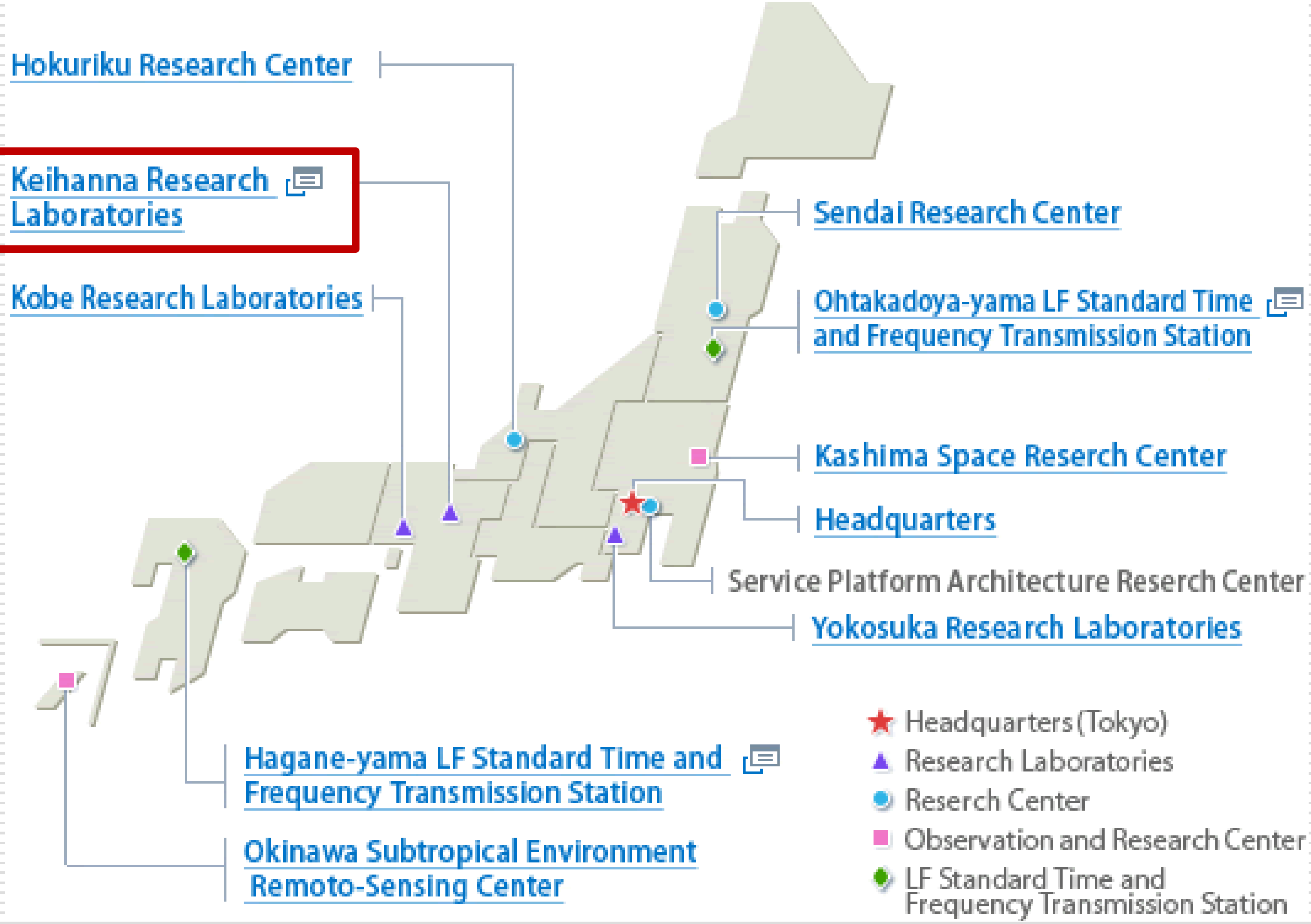
1992
Telecommunications
Advancement
Organization

Certified Institution

2004

National Institute of Information and Communications Technology

Locations



Hokuriku Research Center
Keihanna Research Laboratories
Kobe Research Laboratories
Sendai Research Center
Ohtakadoya-yama LF Standard Time and Frequency Transmission Station
Kashima Space Reserch Center
Headquarters
Service Platform Architecture Reserch Center
Yokosuka Research Laboratories
Hagane-yama LF Standard Time and Frequency Transmission Station
Okinawa Subtropical Environment Remoto-Sensing Center
Headquarters (Tokyo)
Research Laboratories
Reserch Center
Observation and Research Center
LF Standard Time and Frequency Transmission Station

NICT Keihanna Research Laboratories

(Open) since 1. April, 2008

(Location) Kansai Science City

(Number of Staffs) about 160

A part of ATR

Overcome the barriers in ICT society

(Ⅰ) Barriers of language

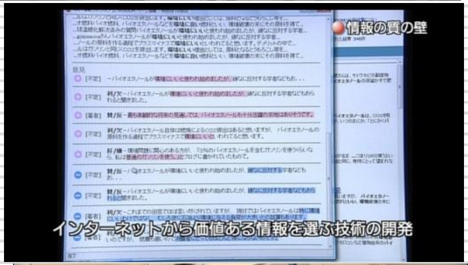
R&D on the multi-lingual technology

(Ⅱ) Barriers of ability

spoken language and nonverbal interaction technology

(Ⅲ) Barriers of information quality

Information analysis with information credibility criteria



(Ⅳ) Barriers between the real and the cyber world

Natual, real-time connections between the two worlds

(Ⅴ) Barries of distance

Ultra-realistic communications to provide the feeling of "being there" via all five senses, etc

About ATR?

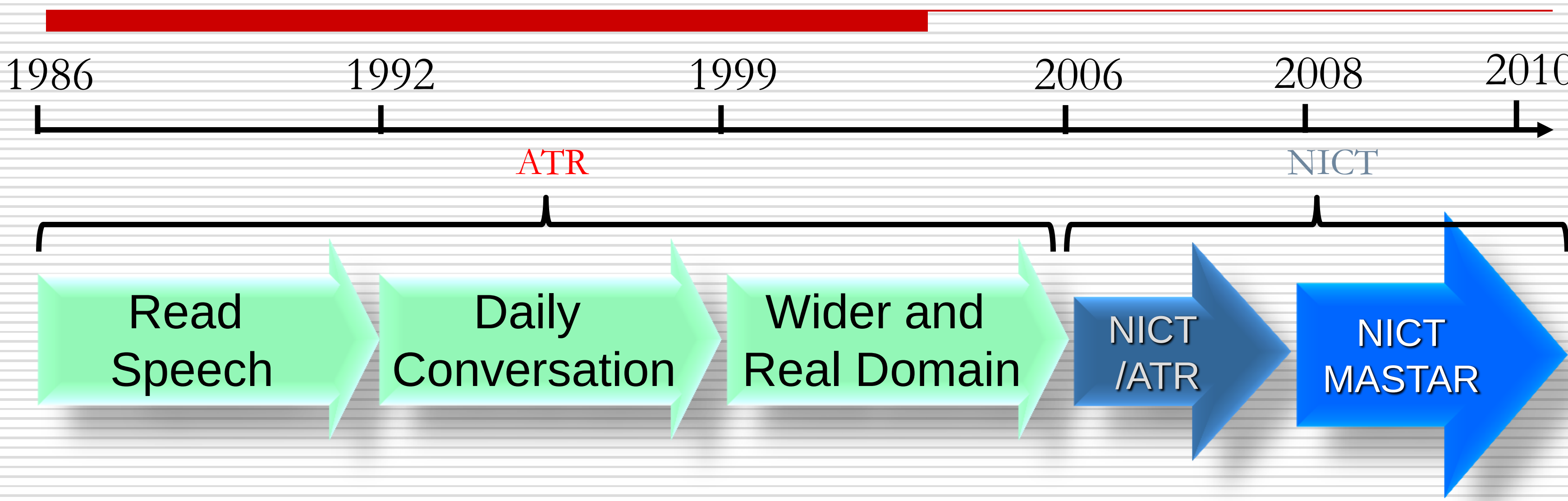
- ATR: Advanced Telecommunication Research Institute International

ATR was founded in March 1986.

ATR Laboratories

- Brain Information Communication Research Labs Group
  - Computational Neuroscience Lab.
  - Cognitive Mechanisms Lab.
  - Neural Information Analysis Lab.
- Social Media Research Labs. Group
  - Intelligent Robotics and Communication Lab.
  - Hiroshi Ishiguro Lab.
  - Adaptive Communications Research Lab.
  - Wave Engineering Lab.
- Spoken Language Communication Research Labs.
- Media Information Science Labs

History of Speech Translation Research



- Syntactically correct
- Clear utterance
- Limited domain
  "Conference Registration"

- Standard expression
- Unclear utterance
- Limited domain
  "Hotel Reservation"

- Wider and real domain
  "International Travel"
- Realistic expressions
- Noisy speech
- J-E, J-C speech translation

MIC & NICT
& CSTP PJ

C-STAR
(ATR,CMU, UKA, CLIPS, IRST, ETRI, CAS)

A-STAR (→ U-STAR)

Source Coding

- Contents of the lecture

Information Theory:

Source Coding + Channel Coding + Encryption

- Goal:
  - Understanding of Source Coding by theory and application

- Contents:
  - Amount of information, modeling of information source
  - Zero-memory source, Markov source, hidden Markov source
  - Source coding theorem, compact codes
  - Universal coding, rate distortion theory
  - Source coding of analog signal, vector quantization
  - Modeling and coding of language and speech

Text book and references

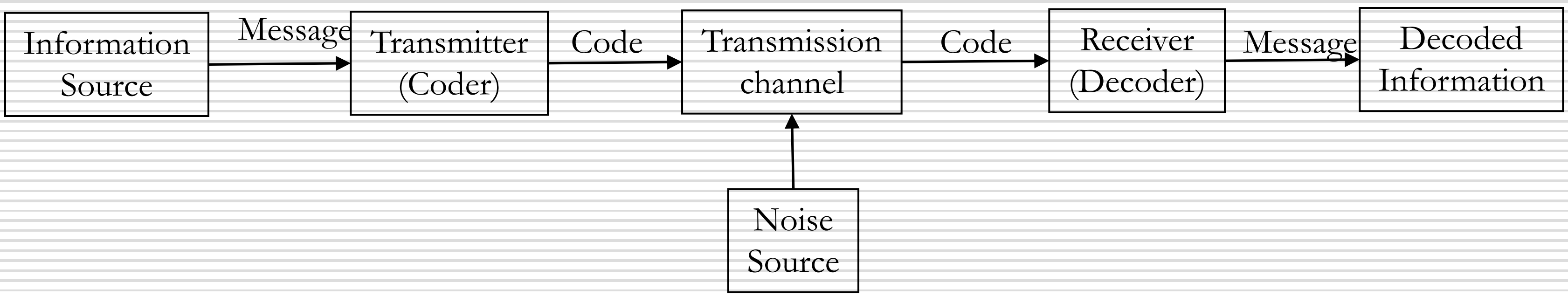
- Norman Abramson: “Information Theory and Coding”, McGraw-Hill, 1963
- A.Gersho, R.M.Gray: “Vector Quantization and Signal Compression”, Kluwer Academic Publisher
- T.C.Bell, J.G.Cleary, I.H.Witten： “Text Compression”, Prentice Hall

Role of information theory

- Information Theory:
  Measure for Information Amount, Modeling of Information Source

- Claude Shannon:
  ``Mathematical Theory of Communication" (1948),
  Bell System Technical Journal
  - "Shannon entitles his theory a mathematical theory of communication: Theory of carriers of information."
  - "Theory about carriers of information-symbols and not with information itself."
  - "The semantic aspects of communication are irrelevant to the engineering problems."

Transmission model



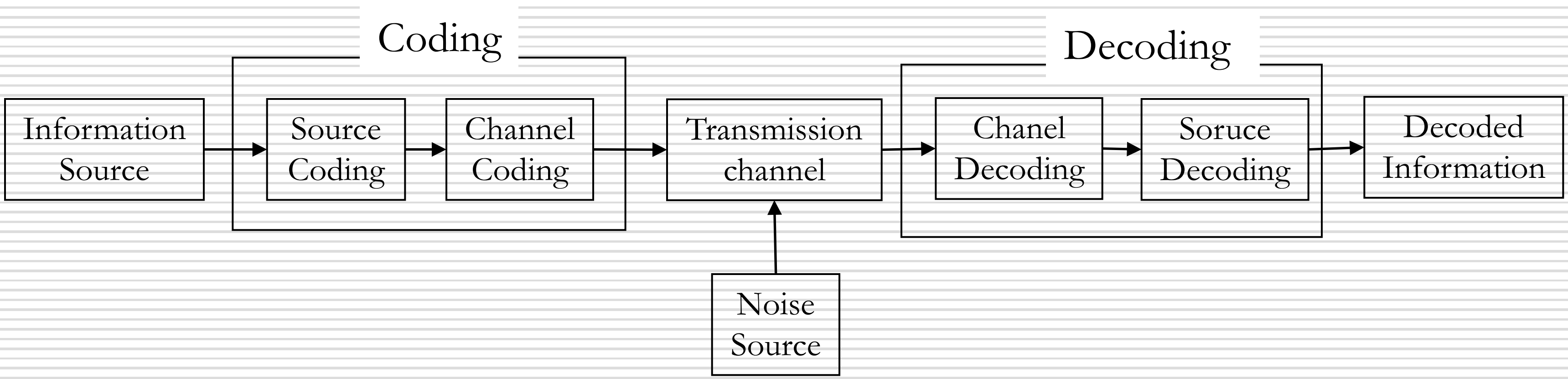
Efficient usage of transmission channel

- Digital channel: Reduction of transmission codes
- Analog channel: Reduction of transmission time and frequency bands

Improve reliability

- Digital channel: Reduction of transmission errors
- Analog channel: Improve Signal to Noise Ratio

Separate modeling



- Separate optimization:
  Source coding + Channel coding

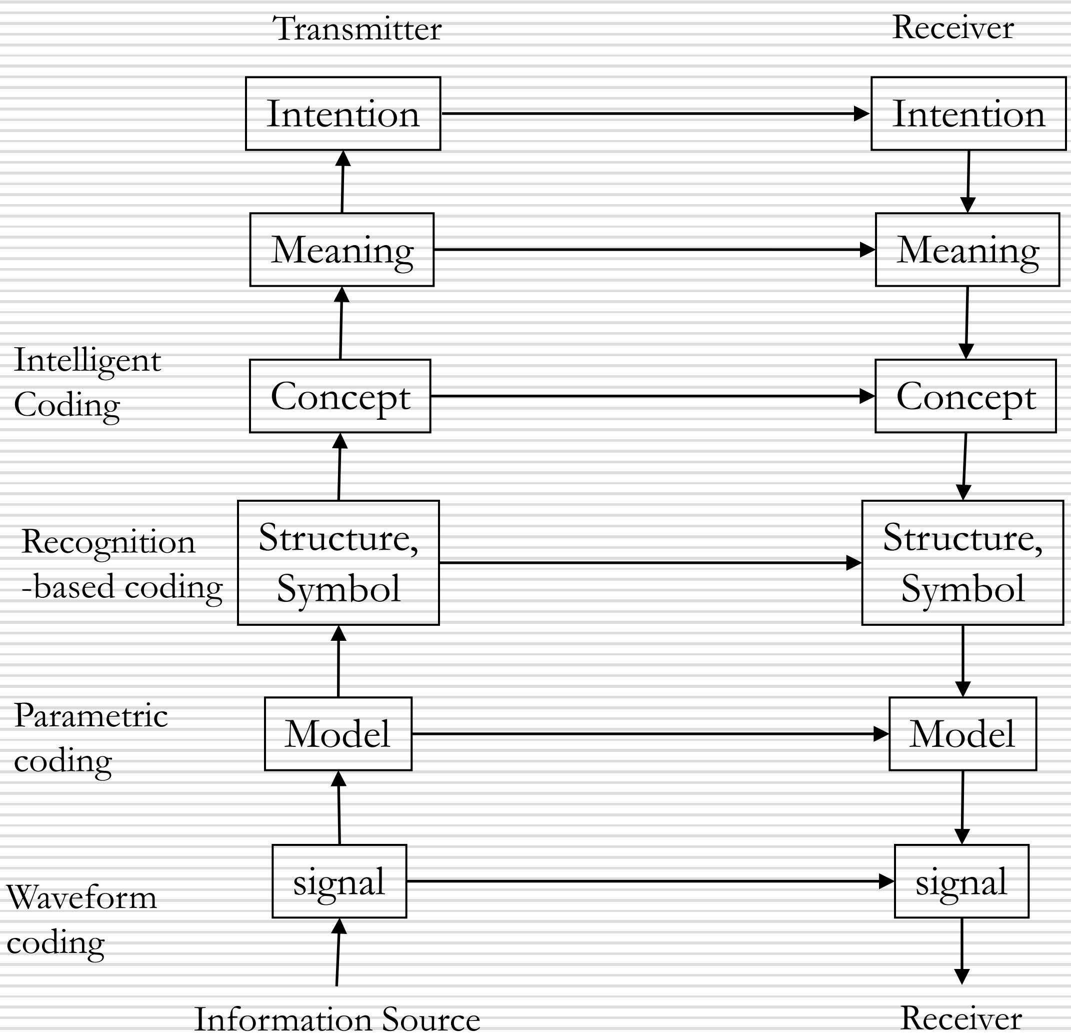
Amount of information

- Amount of Information:
  Defined by statistical property of an overall set not by individual events.

- Statistical Structure
  - Statistically definable Sets
    => Memoryless source, Markov source
  - Non-statistical sets and unknown-structured sets

- Unknown-structured information sets
  - Universal Coding
  - Lempel Zip Coding
  - Arithmetic Coding

Hierarchical model of codes



What is information

- Messages which reduce uncertainty
  - Measurement of body temperature
    Prediction whether he caught cold or not is possible.
  - Weather forecast
    Prediction of tomorrows weather is possible.

- Information theory:
  - Measurement of information
  - Higher efficiency and reliability of transmission

Properties of information

- Non-negativity:
  Information amount is non-negative. If probability of the event equals to *0* or *1*, amount of information becomes *0*.
  - Events which does surely happen or doesn't happen, don't have any additional information. The amount of information of these events is *0*.
  - To know the events whose probabilities are $0<p<1$ bring certain amount of information since it reduces ambiguity.

- Monotonic decreasing:
  The more amount of information the less probability the event has.
  - Amount of information is bigger if the event is unexpected.

Amount of information: Additivity

How much is the amount of information, $I(pq)$, of an joint event with probability p and q ?,

where,

$I(p)$: amount of information of an event with probability p

$I(q)$: amount of information of an event with probability q

$$I(pq) = I(p) + I(q)$$

means,

amount of information is same if given once or one by one.

Amount of information

- Only function form which satisfies the above three properties is,

$$I(p) = -\log(p).$$

  Now, *I(P)* is defined as amount of information.

- Units of amount of information
  - $-\log_2(p)$ [bit]
  - $-\log_e(p)$ [nat]
  - $-\log_{10}(p)$ [dit] or [Hartley]
- If p=0.5, $I(p)$ is maximum. -> only valid for the average case!
- Amount of information by [bit] represents average number of [yes/no] questions to know what event has happened.

What is coding?

Decimal number	Binary representation
0	0000
1	0001
2	0010
3	0011
4	0100
5	0101
6	0110
7	0111
8	1000
9	1001

- Binary coding of the decimal digits.
  - Message Symbols:
    0,1,…,9
  - Code word:
    0000,0001,0010,…
  - Backward decoding is straight-forward in this example.

What is coding ?

Message Symbols	Binary representation
s1	0
s2	01
s3	001
s4	111

- A binary code.
  - Backward decoding is NOT straightforward.
  - 111001 can be generated by "S_4S_3" and "$S_4S_1S_2$"

What is coding ?

Message Symbols	Binary representation
s1	0
s2	10
s3	110
s4	1110

- Another binary code
  - Use "0" as a separator.
  - Backward decoding is unique and straightforward.

One problem in coding

Message Symbols	Binary representation
Sunny	1/4
Cloudy	1/4
Rainy	1/4
Foggy	1/4

Message Symbols	Codes
Sunny	00
Cloudy	01
Rainy	10
Foggy	11

- Weather in San Francisco
- Code alpha:
  - Two binary digits are used.
  - "Sunny, Foggy, Foggy, Cloudy," comes to "00111101".
  - Two binary digits are necessary to backward decoding.

One problem in coding

Message Symbols	Binary representation
Sunny	1/4
Cloudy	1/8
Rainy	1/8
Smoggy	1/2

Message Symbols	Codes
Sunny	10
Cloudy	110
Rainy	1110
Smoggy	0

- Weather in Los Angels
- Code beta:
  - Two binary digits are used.
  - Probabilities are non-uniform
  - "Sunny,Smoggy,Smoggy, Cloudy" comes to "1000110".
  - Waiting for 0 is necessary to backward decoding.
  - Average code length = 1 7/8 < 2 binit.

$$L = 2\Pr(Sunny) + 3\Pr(Cloudy) + 4\Pr(Rainy) + 1\Pr(Smoggy)$$
$$= 2\frac{1}{4} + 3\frac{1}{8} + 4\frac{1}{8} + 1\frac{1}{2}$$
$$= 1\frac{7}{8} \quad binits\,/\,message$$

Amount of information

- TV: Black, white, and gray dots, with roughly 500 rows and 600 columns. Namely 500x600=300,000 dots may take on any one of 10 distinguishable brightness levels. (p= 1/10 ^{300,000})

$$I(E) = 300{,}000 \log 10 \approx 10^6 \, bits$$

- Radio: 10,000 words vocabulary announcer selects 1,000 words randomly.

  (p= 1/10,000 ^{1,000})

$$I(E) = 1{,}000 \log 10{,}000 \approx 1.3 \times 10^4 \, bits$$

- TV picture is worth more 1,000 words.

Average amount of information

- Amount of information is defined by,

$$I(p) = -\log(p).$$

- Average amount of information of the information source *A* is defined by,

$$H(A) = \sum_{i=1}^{n} P(e_i) I(e_i) = -\sum_{i=1}^{n} P(e_i) \log_2 P(e_i)$$

and, *H(A)* satisfies,

$$0 \le H(A) \le \log_2 n$$

Entropy = Average amount of information (bit).

Entropy

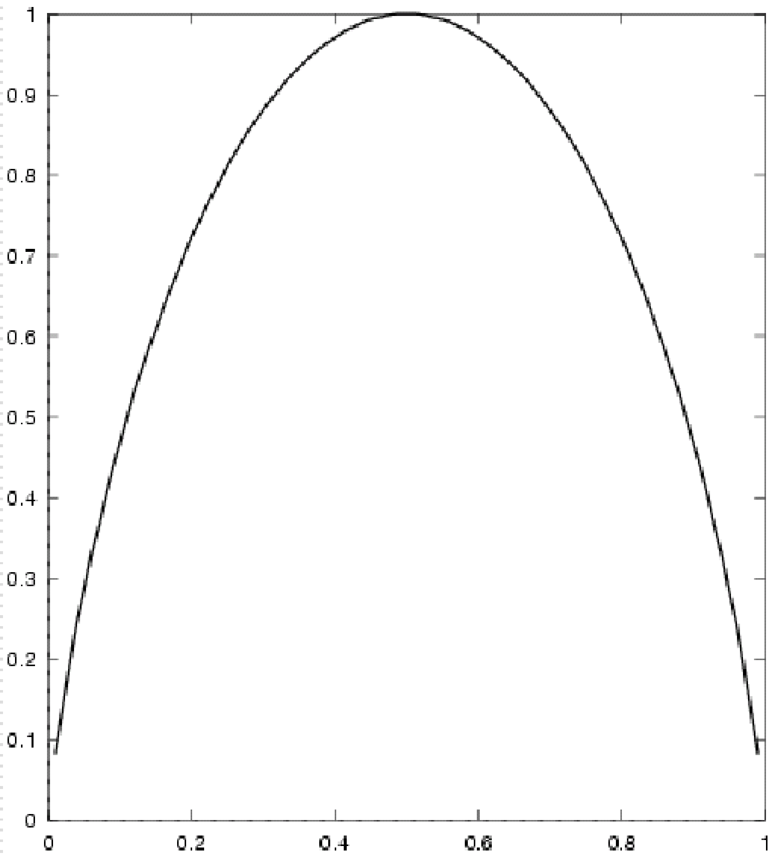
- Entropy represents ambiguity of the information source.
  When one message e_i is received, ambiguity of the information H(A) is decreased.
  This amount of decrease is equivalent to the amount of information of the message e_i.

Properties of Entropy

- Now we have source alphabet {0,1}, and

$$P(0)=\omega,\ P(1)=1-\omega=\overline{\omega}.$$

Entropy function is like,



Amount of information for multiple events

- Amount of information for multiple events can be defined by the decrease of the Entropy.
  Now let $P(a_i)$ be a prior probability of a message a_i, and $P(a_i|b_j)$ be a posterior probability of a_i given a message b_i. A prior Entropy of information source A is defined by,

$$H(A) = \sum_{A} P(a_i) \log \frac{1}{P(a_i)}$$

and, a posterior Entropy of information source A given a message b_j is defined by,

$$H(A/b_j) = \sum_{A} P(a_i/b_j) \log \frac{1}{P(a_i/b_j)}$$

Therefore,

Amount of information of multiple events

$$= H(A) - H(A/b_j)$$

Conditional Entropy

- Conditional Entropy is expectation of $H(A|b_j)$.

$$H(A|B) = -\sum_{i=1}^{n}\sum_{j=1}^{m} P(b_j)P(a_i|b_j)\log P(a_i|b_j)$$
$$= -\sum_{i=1}^{n}\sum_{j=1}^{m} P(a_i, b_j)\log P(a_i|b_j)$$

- And following inequality holds,

$$0 \le H(A|B) \le H(AB) \le H(A) + H(B)$$

Mutual Information

- Amount of information of multiple events

$$H(A) - H(A/b_j)$$

- What is an amount of information if we know information source B not a single message of b_j of B.

$$\begin{aligned} I(A;B) &= H(A) - H(A/B) \\ &= \sum_{A} P(a_i) \log \frac{1}{P(a_i)} - \sum_{A,B} P(a_i, b_j) \log \frac{1}{P(a_i / b_j)} \\ &= \sum_{A,B} P(a_i, b_j) \log \frac{P(a_i, b_j)}{P(a_i) P(b_j)} \end{aligned}$$

I(A;B) is called "Mutual Information".

Joint Entropy

- Entropy of joint information source A and B is defined by,

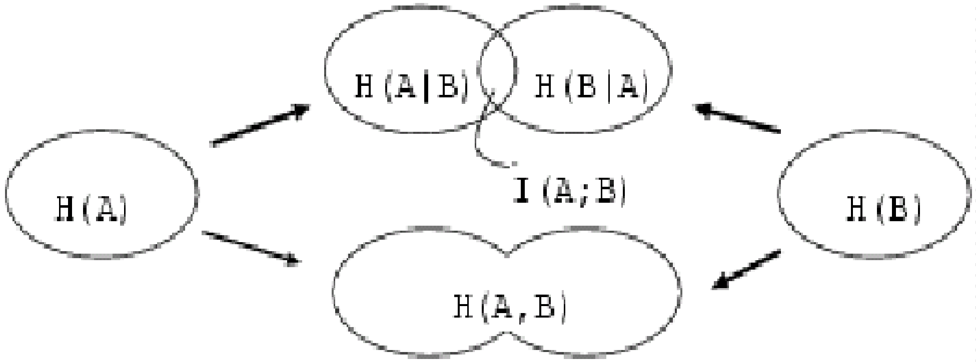
$$H(A,B) = -\sum_{i=1}^{n}\sum_{j=1}^{m} P(a_i, b_j)\log P(a_i, b_j)$$

Mutual Information

- Mutual Information I(A;B) holds,

$$0 \le I(A;B) \le H(A)$$

$$\begin{aligned} I(A;B) &= I(B;A) \\ &= H(B) - H(B|A) \\ &= H(A) + H(B) - H(A,B) \end{aligned}$$



Mutual Information (example)

A　B	Play (b_1)	Not Play(b_2)	P(a_i)
Win(a_1)	0.42(0.6)	0.28(0.93)	0.7
Lose(a_2)	0.28(0.4)	0.02(0.07)	0.3
P(b_i)	0.7	0.3	1.0

- Initial Entropy of A, H(A) is,

$$H(A) = -0.7\log 0.7 - 0.3\log 0.3 = 0.88$$

The winning rate after we know he plays a game becomes 0.6. The Entropy H(A|b_i) is,

$$H(A|b_1) = -0.6\log 0.6 - 0.4\log 0.4 = 0.97$$

Entropy increases by knowing the information of bi.

If we know he doesn't play, the winning rate is 0.93. This time, Entropy decreses.

$$H(A|b_2) = -0.93\log 0.93 - 0.07\log 0.07 = 0.17$$

Now mutual information is,

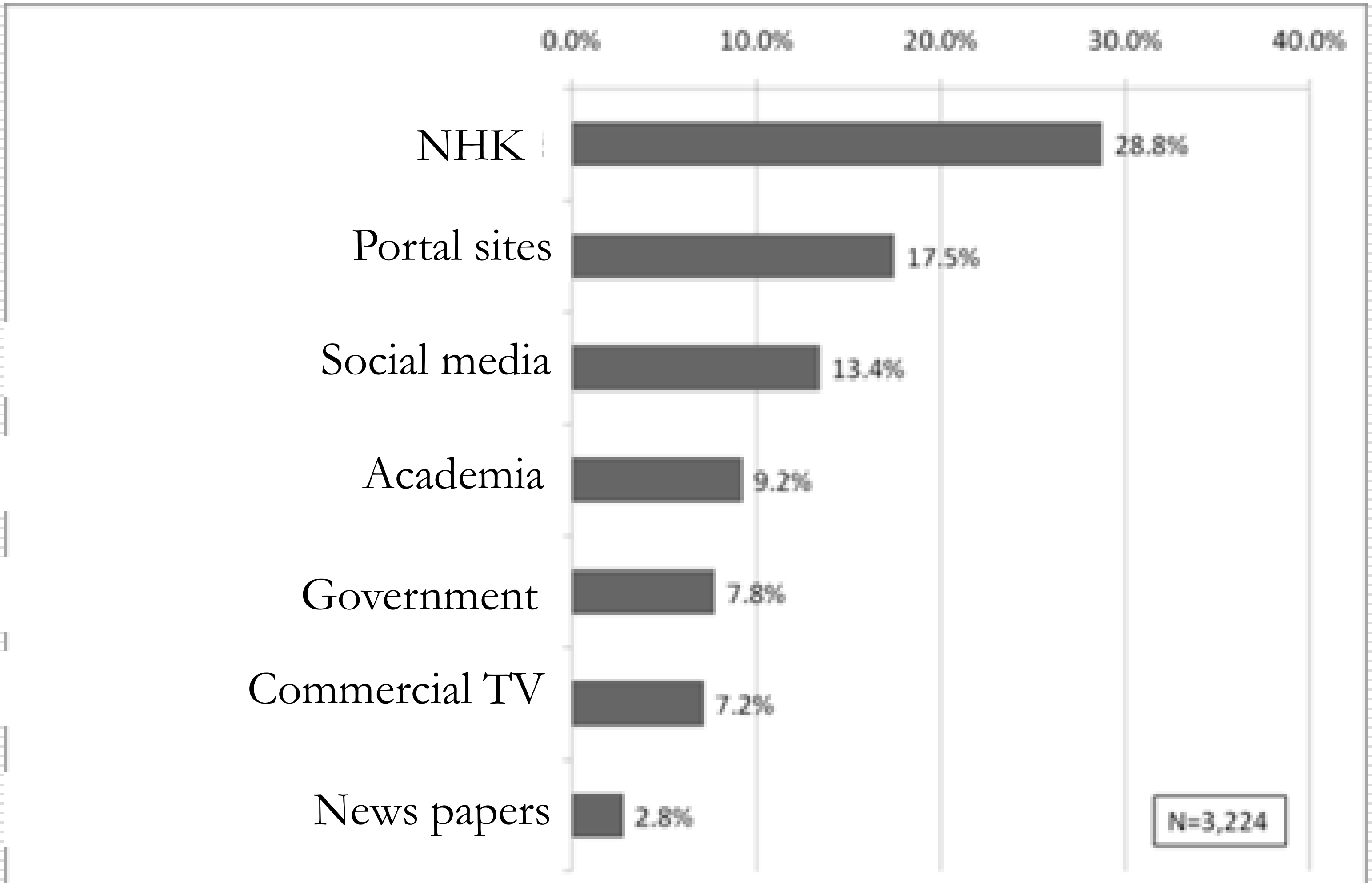
$$I(A;B) = H(A) - H(A|B)$$

$$= 0.88 - (0.7\times 0.97 + 0.3\times 0.17) = 0.88 - 0.79 = 0.09 \geq 0$$

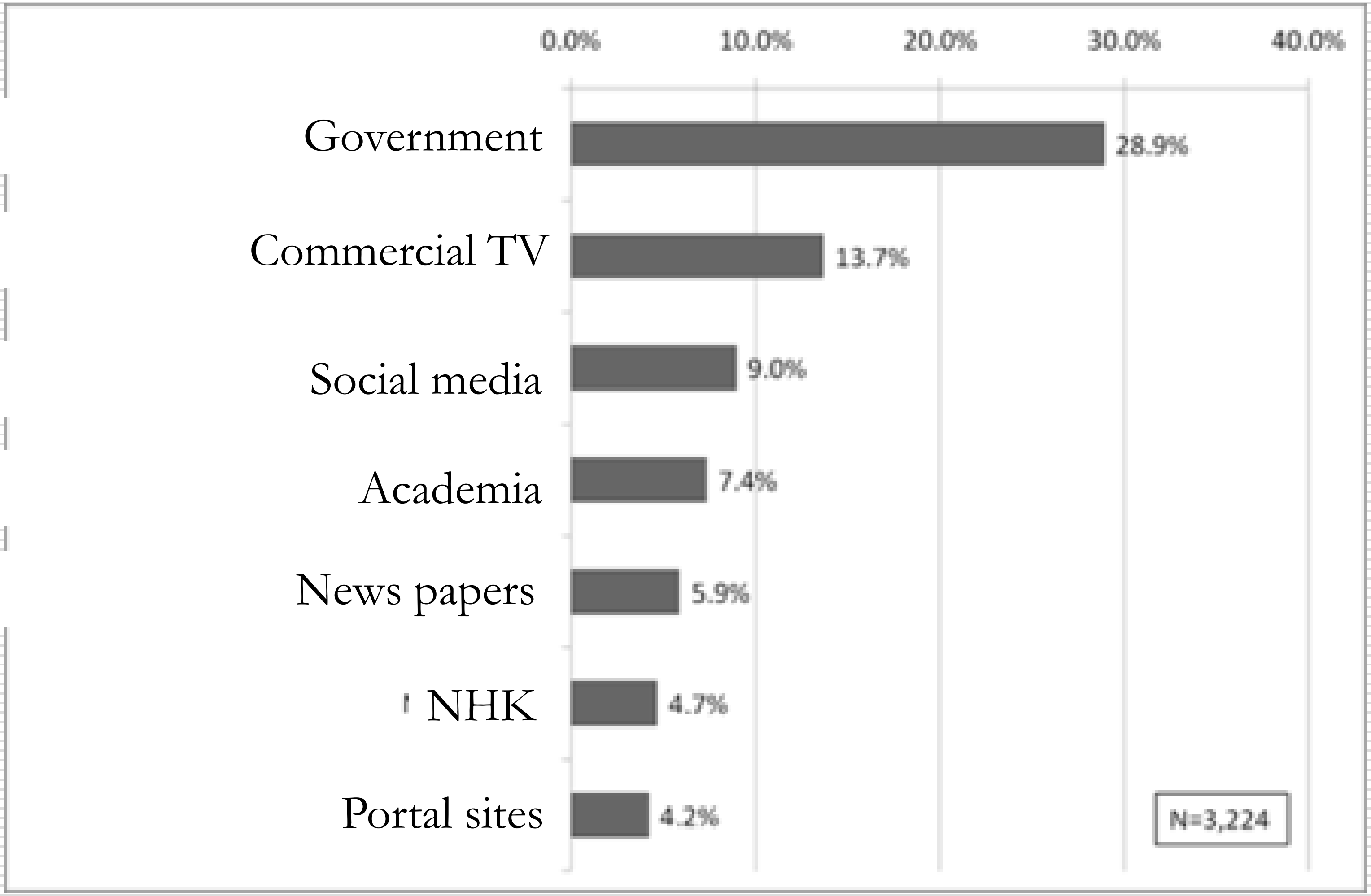
Goal of 1st day

ROLE OF SOCIAL MEDIA

Credibility Increased information Source

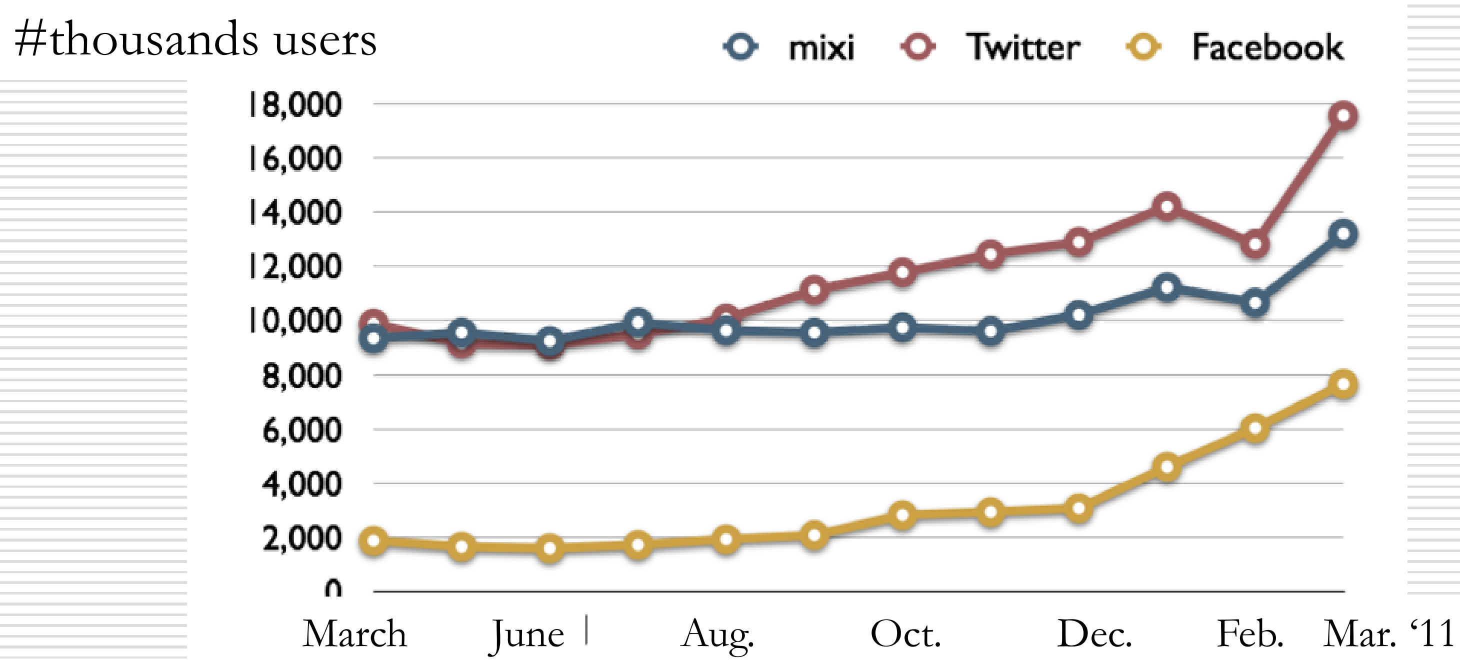


Credibility Decreased Information Source



Trends of Social Media Users

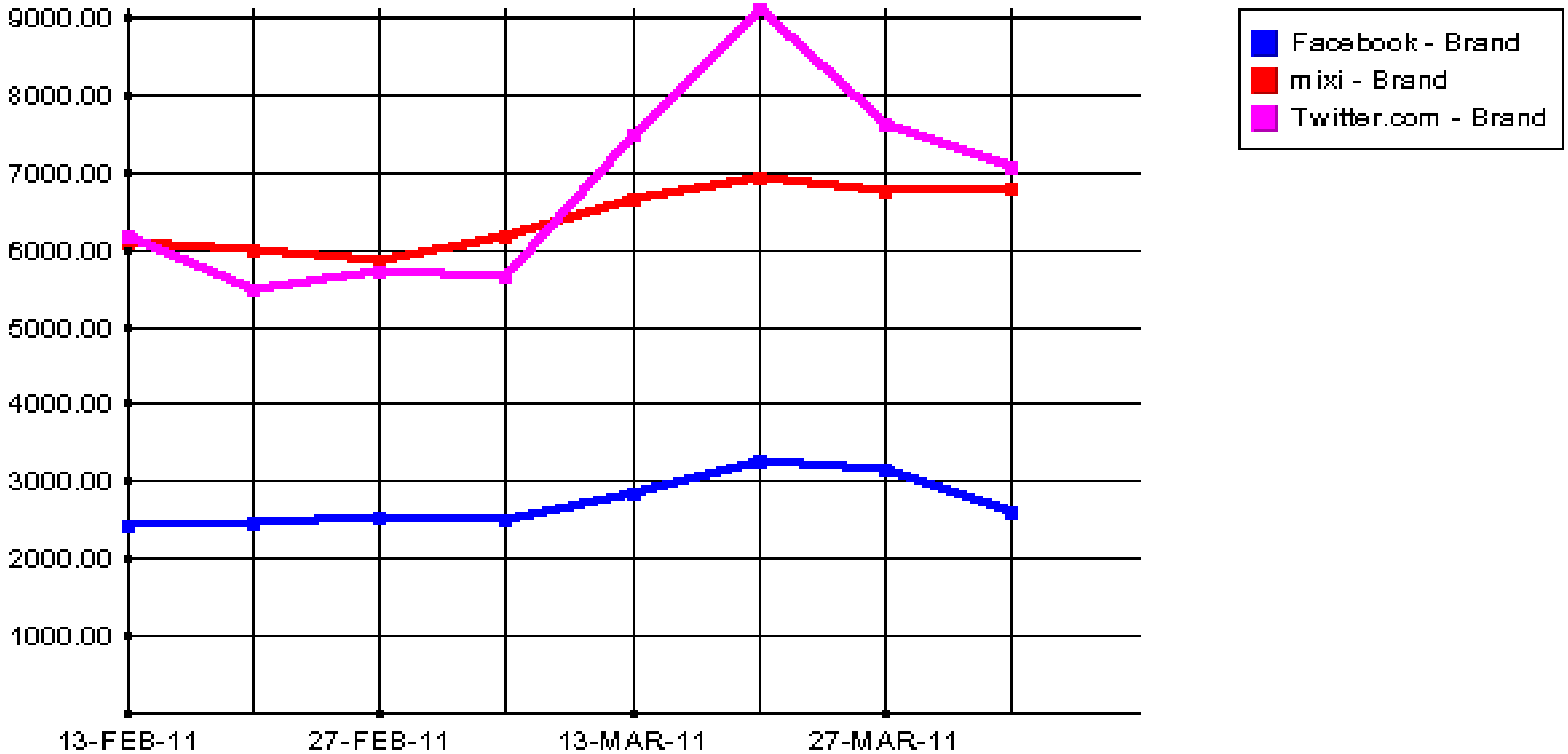
by Neilsen Netview



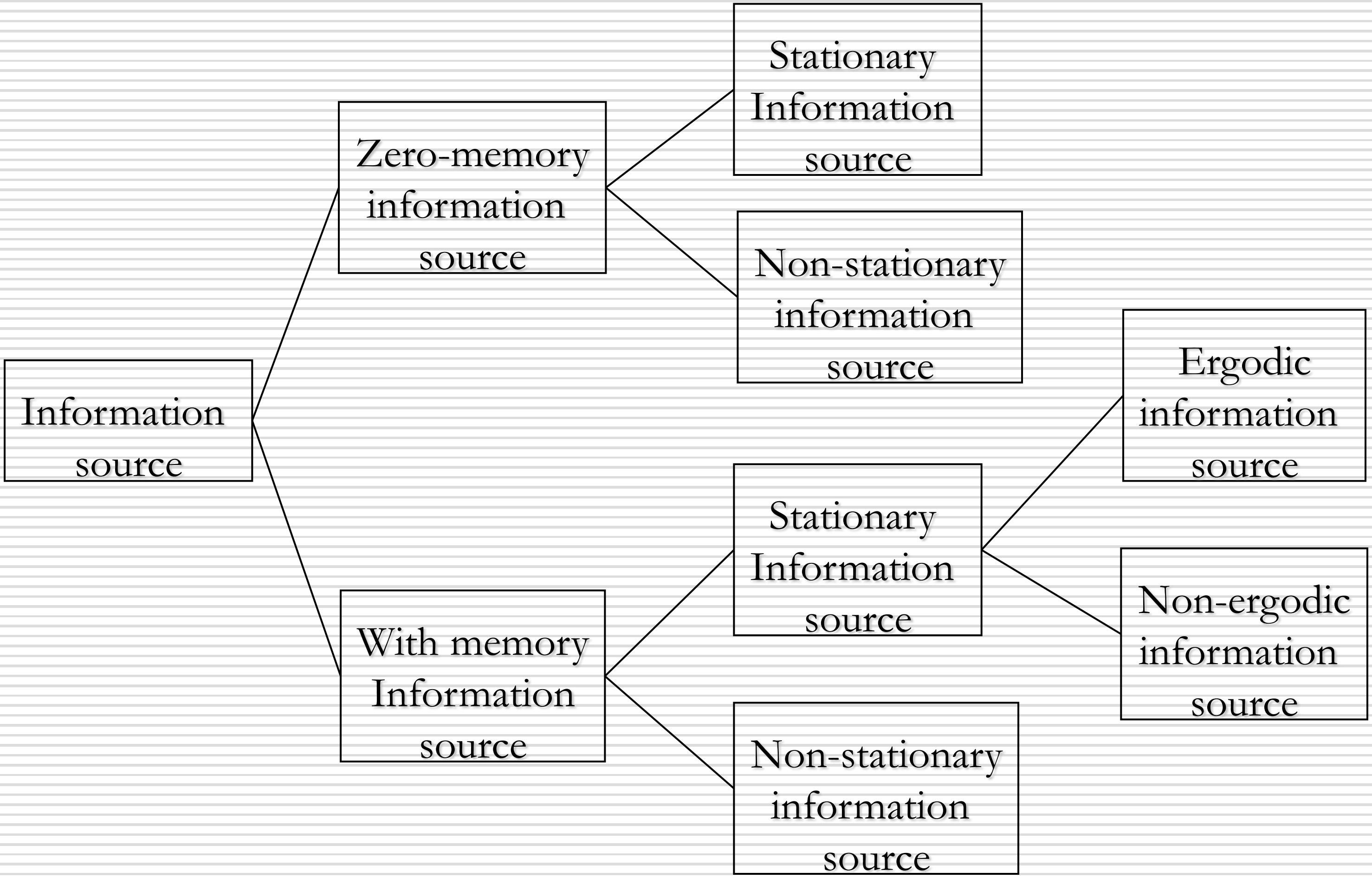
(単位: 千人)

	4月	5月	6月	7月	8月	9月	10月	11月	12月	1月	2月	3月
mixi	9,344	9,559	9,242	9,930	9,632	9,557	9,744	9,608	10,214	11,228	10,659	13,211
Twitter	9,882	9,157	9,100	9,496	10,069	11,129	11,778	12,444	12,901	14,211	12,824	17,571
Facebook	1,873	1,653	1,596	1,719	1,928	2,080	2,819	2,934	3,077	4,598	6,030	7,659

Weekly Trends of #users



Models for information sources



Models for information sources

- Zero-memory information source:
  Source alphabets in $S=\{s_1, s_2, s_3,\ldots,s_q\}$ are mutually independent and independent to alphabets in history. Zero-memory information source is completely described by the source alphabet S and their probabilities, $P(s_1),P(s_2),..,P(s_q)$.

- Markov information source:
  Probability of the source alphabet S_i is described by previous m alphabets. If $m=1$, it is called 1st order Markov Model. Probabilities of the alphabet is described by, $P(s_i \mid s_{j1},s_{j2},..,s_{jm})\ i=1,2,..q; j_q=1,2,..,q$.

Models of information source

- Stationary information source:
  Probabilities of the specific source alphabets are invariant to time shift.

- Ergodic information source:
  Observed source alphabet sequence becomes same as a representative one with probability 1, when we observe the source alphabet sequence for long time.

Zero-memory information source

$$\boxed{\text{Source}} \longrightarrow s_i,\ s_j,\ \ldots$$

- Zero-memory information source:
  Successive symbols emitted from the source are statistically independent, which is described by source alphabet S and the probabilities with which the symbols occur:

$$P(s_1), P(s_2), \cdots, P(s_q)$$

- An amount of information for one symbol s_i is,

$$I(s_i) = \log \frac{1}{P(s_i)} \, bits$$

- An average amount of information for information source S is,

$$\sum_S P(s_i) I(s_i)$$

- Entropy H(S) of zero-information information source is,

$$H(s) \equiv \sum_S P(s_i) \log \frac{1}{P(s_i)}$$

Examples

- Source S; $$S = \{s_1, s_2, s_3\}, P(s_1) = \frac{1}{2}, P(s_2) = P(s_3) = \frac{1}{4}$$
- $$H(S) = \frac{1}{2}\log 2 + \frac{1}{4}\log 4 + \frac{1}{4}\log 4 = \frac{3}{2} bits$$
- If I(s_i) is measured in r-ary units, we have

$$H(S) = \sum_S P(s_i) \log_r \frac{1}{P(s_i)} \quad r - ary\ units$$

$$H_r(S) = \frac{H(S)}{\log r}$$

Some properties of Entropy

$y = x - 1$ lies above $y = \ln x$

$\ln x \leq x - 1$ with equality if, and only if x=1

$$\ln \frac{1}{x} \geq 1 - x$$

$$x_i \geq 0,\ y \geq 0, \quad for\ i\ and\ \ j,$$

$$\sum_{i=1}^{q} x_i = \sum_{j=1}^{q} y_j = 1$$

$$\sum_{i=1}^{q} x_i \log \frac{y_i}{x_i} = \frac{1}{\ln 2} \sum_{i=1}^{q} x_i \ln \frac{y_i}{x_i}$$

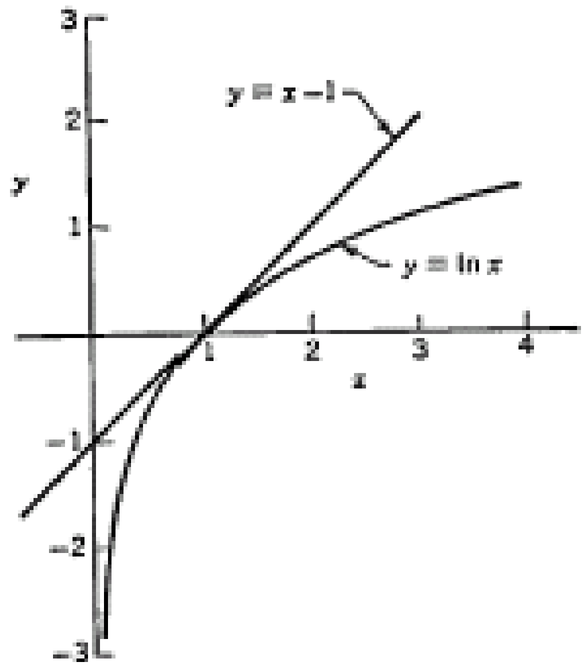


FIGURE 2-2. The natural logarithm of x, and $x - 1$.

Some properties of Entropy

$$\sum_{i=1}^{q} x_i \log \frac{y_i}{x_i} \le \frac{1}{\ln 2} \sum_{i=1}^{q} x_i \left(\frac{y_i}{x_i} - 1\right)$$

$$\le \frac{1}{\ln 2} \left(\sum_{i=1}^{q} y_i - \sum_{i=1}^{q} x_i\right)$$

$$\le 0 \quad or,$$

$$\sum_{i=1}^{q} x_i \log \frac{1}{x_i} \le \sum_{i=1}^{q} x_i \log \frac{1}{y_i}$$

with equality if, and only if, $x_i = y_i$ for all i.

This is called Jensen's inequality.

Some properties of Entropy

$$H(s) \equiv \sum_{i=1}^{q} P_i \log \frac{1}{P_i}$$

$$\log q - H(s) = \sum_{i=1}^{q} P_i \log q - \sum_{i=1}^{q} P_i \log \frac{1}{P_i}$$

$$= \sum_{i=1}^{q} P_i \log qP_i$$

$$= -\log e \sum_{i=1}^{q} P_i \ln qP_i$$

$$\log q - H(s) \geq \log e \sum_{i=1}^{q} P_i \left(1 - \frac{1}{qP_i}\right)$$

$$= \log e \left(\sum_{i=1}^{q} P_i - \frac{1}{q} \sum_{i=1}^{q} \frac{P_i}{P_i}\right)$$

$$= 0$$

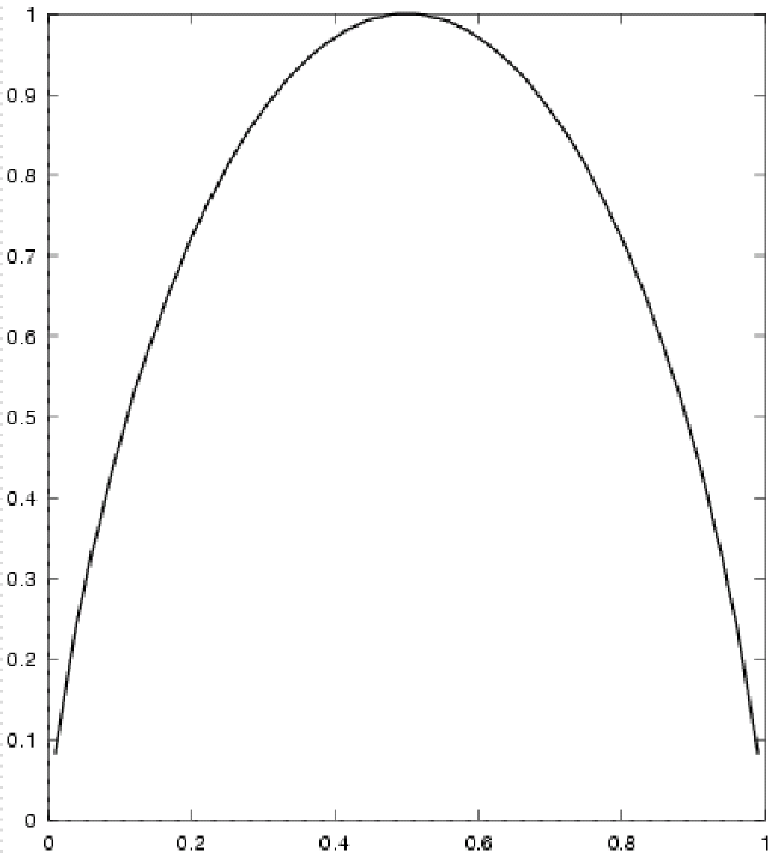
H(s) always less than or equal to, *log q*. Equality holds if, and only if, *Pi=1/q* for all *i*.

Properties of Entropy

- Again, we have source alphabet {0,1}, and

$$P(0)=\omega,\ P(1)=1-\omega=\overline{\omega}.$$

Entropy function is like,



Extensions of a zero-memory source

- Extensions to blocks of symbols.
  For instance, suppose two binary source alphabet case, 00, 01, 10, and 11.

- Definition:
  Let S be a zero-memory information source with source alphabet $\{s_1, s_2, \ldots, s_q\}$ and with the probability of s_i equal to P_i. Then the n-th extension of S, S_n, is a zero-memory source with q^n symbols $\sigma_1, \sigma_2, \ldots, \sigma_{q^n}$.

  Each σ_i corresponds to some sequence of n of the s_i. $P(\sigma_i)$, the probability of σ_i , is just the probability of the corresponding sequence of s_i's. That is, if σ_i corresponds to $(s_{i1}, s_{i2}, .., s_{in})$, then $P(\sigma_i) = P_{i1}, P_{i2}, \ldots, P_{in}$.

Extension of zero-memory source

- Entropy: $$H(s^n) = \sum_{s^n} P(\sigma_i) \log \frac{1}{P(\sigma_i)}$$

$$H(s^n) = \sum_{s^n} P(\sigma_i) \log \frac{1}{P_{i_1} P_{i_2} \cdots P_{i_n}}$$

$$= \sum_{s^n} P(\sigma_i) \log \frac{1}{P_{i_1}} + \sum_{s^n} P(\sigma_i) \log \frac{1}{P_{i_2}} + \cdots + \sum_{s^n} P(\sigma_i) \log \frac{1}{P_{i_n}}$$

$$\sum_{s^n} P(\sigma_i) \log \frac{1}{P_{i_1}} = \sum_{s^n} P_{i_1} P_{i_2} \cdots P_{i_n} \log \frac{1}{P_{i_1}}$$

$$= \sum_{i_1=1}^{q} P_{i_1} \log \frac{1}{P_{i_1}} \sum_{i_2=1}^{q} P_{i_2} \sum_{i_3=1}^{q} P_{i_3} \cdots \sum_{i_n}^{q} P_{i_n}$$

$$= \sum_{i_1=1}^{q} P_{i_1} \log \frac{1}{P_{i_1}} = \sum_{S}^{q} P_{i_1} \log \frac{1}{P_{i_1}}$$

$$= H(s)$$

$$H(s^n) = nH(s)$$

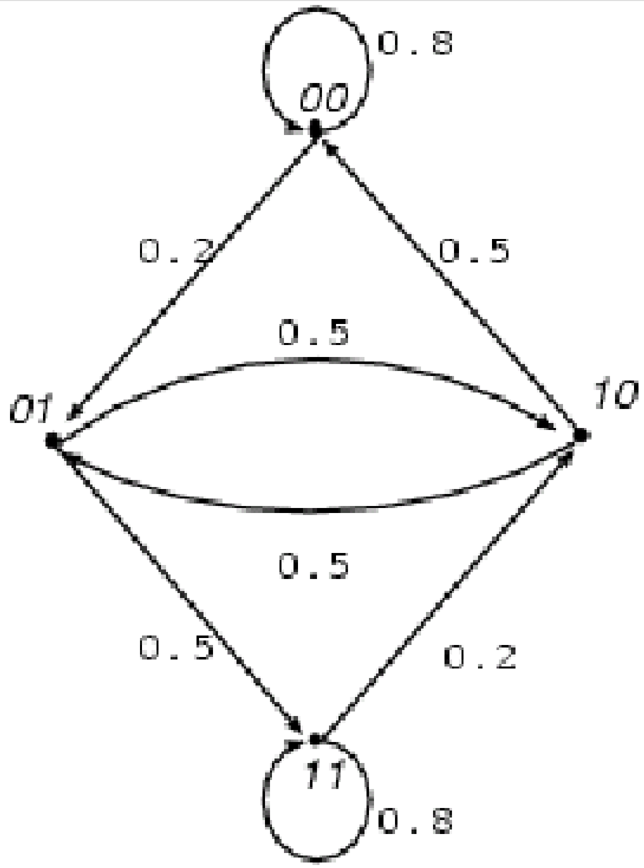
Markov Information Source

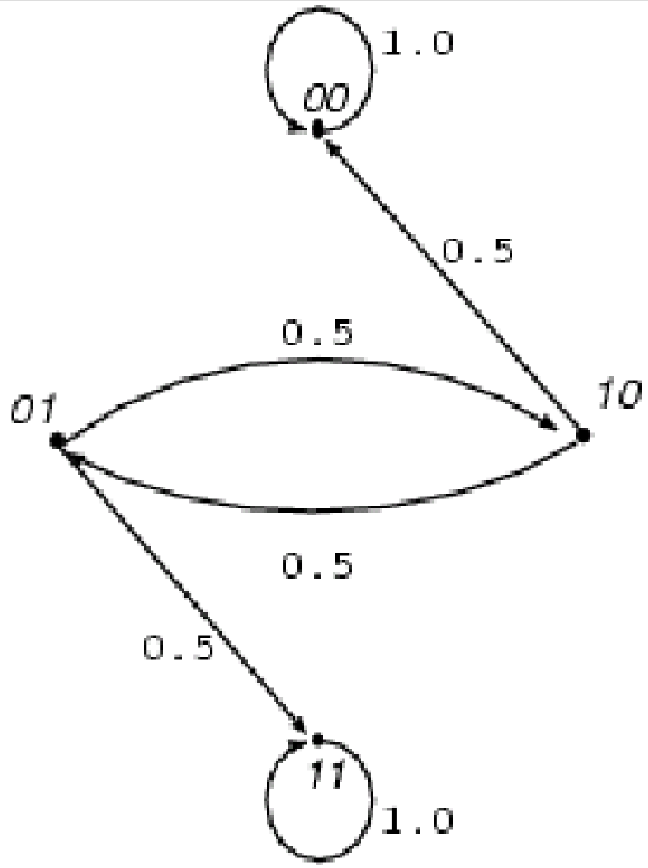
- A more general type of information source with q symbols than the zero-memory source is one in which the occurrence of a source symbol s_i may depend on a finite number m of preceding symbols. Such a source, *m*th-order Markov source, is defined by giving the source alphabet S and the set of conditional probabilities.

$$P(s_i / s_{j_1}, s_{j_2}, \ldots, s_{j_m}) \quad for \quad i = 1,2,\ldots,q;\ j_p = 1,2,\ldots,q$$

- State: the probability of emitting a given symbol is known if we know the m preceding symbols. We call the *m* preceding symbols as a state of the *m*th-order Markov source.

Markov information source





Non-Stationary

Ergodic Markov source

Non-ergodic Markov source

Entropy for Markov source

- If we are in the state specified by $(s_{j1},s_{j2},\ldots,s_{jm})$, then the conditional probability of receiving symbol s_i is $P(s_i/s_{j1},s_{j2},..,s_{jm})$. The information we obtain if s_i occurs while we are in state $(s_{j1},s_{j2},..,s_{jm})$ is,

$$I(s_i/s_{j_1},s_{j_2},\ldots,s_{j_m}) = \log\frac{1}{P(s_i/s_{j_1},s_{j_2},\ldots,s_{j_m})}$$

- amount of information per symbol while we are in state $(s_{j1},s_{j2},\ldots, s_{jm})$ is given by,

$$H(S/s_{j_1},s_{j_2},\ldots,s_{j_m}) = \sum_{S} P(s_i/s_{j_1},s_{j_2},\ldots,s_{j_m}) I(s_i/s_{j_1},s_{j_2},\ldots,s_{j_m})$$

- If we average this quantity over the q_m possible states, we obtain the average amount of information by a product of the above entropy and steady state probability , namely the entropy of the mth-order Markov source S.

$$H(S) = \sum_{s^m} P(s_{j_1},s_{j_2},\ldots,s_{j_m}) H(S/s_{j_1},s_{j_2},\ldots,s_{j_m})$$

Entropy of Markov source

- Entropy of mth-order Markov source is given by,

$$H(S)=\sum_{S^m} P(s_{j_1},s_{j_2},\ldots,s_{j_m})\sum_{S} P(s_i/s_{j_1},s_{j_2},\ldots,s_{j_m})\log\frac{1}{P(s_i/s_{j_1},s_{j_2},\ldots,s_{j_m})}$$

$$=\sum_{S^{m+1}} P(s_{j_1},s_{j_2},\ldots,s_{j_m})P(s_i/s_{j_1},s_{j_2},\ldots,s_{j_m})\log\frac{1}{P(s_i/s_{j_1},s_{j_2},\ldots,s_{j_m})}$$

$$=\sum_{S^{m+1}} P(s_{j_1},s_{j_2},\ldots,s_{j_m},s_i)\log\frac{1}{P(s_i/s_{j_1},s_{j_2},\ldots,s_{j_m})}$$

- If S is zero-memory source,

$$P(s_i/s_{j_1},s_{j_2},\ldots,s_{j_m})=P(s_i)$$

Example

Probabilities for the Markov source

s_j, s_k, s_i	$P(s_i/s_j, s_k)$	$P(s_j, s_k)$	$P(s_j, s_k, s_i)$
000	0.8	5/14	4/14
001	0.2	5/14	1/14
010	0.5	1/14	1/14
011	0.5	1/14	1/14
100	0.5	1/14	1/14
101	0.5	1/14	1/14
110	0.2	5/14	1/14
111	0.8	5/14	4/14

Then the entropy is calculated using (2-24b):

$$H(S) = \sum_{S^3} P(s_j, s_k, s_i) \log \frac{1}{P(s_i/s_j, s_k)}$$

$$= 2 \times \tfrac{4}{14} \log \frac{1}{0.8} + 2 \times \tfrac{1}{14} \log \frac{1}{0.2} + 4 \times \tfrac{1}{14} \log \frac{1}{0.5}$$

$$= 0.81 \text{ bit/binit}$$

Adjoint source

- Definition: Let $S=\{s_1,s_2,\ldots,s_q\}$ be the source alphabet of an mth-order Markov source, and let $P_1,P_2,\ldots,P_q$ be the first-order symbol probabilities of the source. The adjoint source to S , written $\bar{S}$, is the zero-memory information source with source alphabet identical with that of S, and with symbol probabilities $P_1,P_2,\ldots,P_q$,
  here the following relationship holds,

$$H(S) \leq H(\bar{S}).$$

Adjoint source

- Let S be a 1st order Markov source,

$$\sum_{S^2} P(s_j, s_i) \log \frac{P(s_j)P(s_i)}{P(s_j, s_i)}$$

By applying Jensen's inequation,

$$\sum_{i=1}^{q} x_i \log \frac{y_i}{x_i} \le 0 \quad or,$$

$$\sum_{i=1}^{q} x_i \log \frac{1}{x_i} \le \sum_{i=1}^{q} x_i \log \frac{1}{y_i}$$

$$\sum_{S^2} P(s_j, s_i) \log \frac{P(s_j)P(s_i)}{P(s_j, s_i)} = \sum_{S^2} P(s_j, s_i) \log \frac{P(s_i)}{P(s_j \mid s_i)} \le 0$$

$$H(S) = \sum_{S^2} P(s_j, s_i) \log \frac{1}{P(s_j \mid s_i)} \le \sum_{S^2} P(s_j, s_i) \log \frac{1}{P(s_i)}$$

$$= \sum_{s_i} \sum_{s_j} P(s_j \mid s_i) P(s_i) \log \frac{1}{P(s_i)}$$

$$= \sum_{s_i} P(s_i) \log \frac{1}{P(s_i)}$$

$$= H(\overline{S})$$

Extension of a Markov source

- Definition: Let S be an mth-order Markov information source with source alphabet $(s_1,s_2,\ldots,s_q)$ and conditional symbol probabilities $P(s_i/s_{j1},s_{j2},\ldots,s_{jm})$. Then the nth extension of S, S^n, is a μ th-order Markov source with q^n symbols, $\sigma_1,\sigma_2,\ldots,\sigma_{q^n}$. Each σ_i corresponds to some sequence of n of the s_i, and the conditional symbol probabilities of σ_i are $P(\sigma_i \mid \sigma_{j_1},\sigma_{j_2},\ldots,\sigma_{j_\mu})$. μ is given by $\mu = [m/n]$, here [ ] is a minimum integer number bigger than m/n. Entropy is given by,

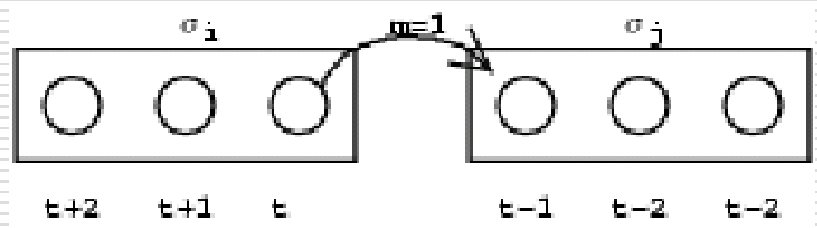
$$H(S^n) = \sum_{S^n}\sum_{S^n} P(\sigma_j,\sigma_i)\log\frac{1}{P(\sigma_i \mid \sigma_j)}$$

$$H(S^n) = nH(S).$$

Extension of a Markov source

- Example:

$$m=1, n=3, \mu=[m/n]=1$$



$$P(\sigma_i \mid \sigma_j) = P(s_{t+2}, s_{t+1}, s_t \mid s_{t-3}, s_{t-2}, s_{t-1})$$

$$= P(s_{t+2}, s_{t+1}, s_t \mid s_{t-1})$$

$$H(S^n) = \sum_{S^n}\sum_{S^n} P(\sigma_j, \sigma_i)\log\frac{1}{P(\sigma_i/\sigma_j)}$$

$$= \sum_{S^{2n}} P(\sigma_j, \sigma_i)\log\frac{1}{P(\sigma_i/\sigma_j)}$$

$$P(\sigma_i \mid \sigma_j) = P(s_{i_1}, s_{i_2}, \ldots, s_{i_n} \mid s_j)$$

$$= P(s_{i_1}, s_{i_2}, \ldots, s_{i_n}, s_j) / P(s_j)$$

$$= \frac{1}{P(s_j)} P(s_{i_n} \mid s_{i_{n-1}}, \ldots, s_{i_1}, s_j) P(s_{i_{n-1}}, \ldots, s_{i_1}, s_j)$$

$$= P(s_{i_n} \mid s_{i_{n-1}}) P(s_{i_{n-1}} \mid s_{i_{n-2}}) \cdots P(s_{i_2} \mid s_{i_1}) P(s_{i_1} \mid s_j)$$

$$H(S^n) = \sum_{S^{2n}} P(\sigma_j, \sigma_i)\log\frac{1}{P(s_{i_1}/s_j)}$$

$$+ \sum_{S^{2n}} P(\sigma_j, \sigma_i)\log\frac{1}{P(s_{i_1}/s_{i_2})} +$$

$$\cdots + \sum_{S^{2n}} P(\sigma_j, \sigma_i)\log\frac{1}{P(s_{i_n}/s_{i_{n-1}})}$$

$$= nH(S)$$

Adjoint source of extended Markov source

- Adjoint source of extended Markov source, $\overline{S^n}$.

  Let $P(\sigma_1), P(\sigma_2), \ldots, P(\sigma_{q^n})$ be the first-order symbol probabilities of the σ_i symbols of the *n*th extension of the first-order Markov source. Since σ_i corresponds to the sequence $(s_{i_1}, s_{i_2}, \ldots, s_{i_n})$, we see that $P(\sigma_i)$ may also be thought of as the nth-order joint probability of the S_{i_k}.

$$H(\overline{S^n}) = \sum_{S^n} P(\sigma_i) \log \frac{1}{P(\sigma_i)}$$

$$= \sum_{S^n} P(s_{i_1}, s_{i_2}, \ldots, s_{i_n}) \log \frac{1}{P(s_{i_1}, s_{i_2}, \ldots, s_{i_n})}$$

If S is a first-order Markov source.

$$P(s_{i_1}, s_{i_2}, \ldots, s_{i_n}) = P(s_{i_1}) P(s_{i_2}/s_{i_1}) P(s_{i_3}/s_{i_2}) \cdots P(s_{i_n}/s_{i_{n-1}})$$

$$H(\overline{S^n}) = \sum_{S^n} P(s_{i_1}, s_{i_2}, \ldots, s_{i_n}) \left[\log \frac{1}{P(s_{i_1})} + \log \frac{1}{P(s_{i_2}/s_{i_1})} + \cdots + \log \frac{1}{P(s_{i_n}/s_{i_{n-1}})} \right]$$

$$= H(\overline{S}) + (n-1) H(S) \quad or$$

$$H(\overline{S^n}) = nH(S) + [H(\overline{S}) - H(S)]$$

Adjoint source of extended Markov source

$$H(\overline{S^n}) = nH(S) + \varepsilon_m$$

$$\frac{H(\overline{S^n})}{n} = H(S) + \frac{\varepsilon_m}{n} \qquad\qquad H(\overline{S^n}) \geq H(S^n) = nH(S)$$

This inequality becomes less important as n becomes larger.

$$\lim_{n\to\infty} \frac{H(\overline{S^n})}{n} = H(S)$$

For larger *n*, the Markov constraints on the symbols from S^n becomes less and less important. The adjoint of the nth extension of *S* is not the same as the *n*th extension of the adjoint of *S*.

$$H(\overline{S^n}) \neq H(\overline{S}^n)$$

If $\overline{S}$ is a zero-memory source,

$$H(\overline{S^n}) = nH(\overline{S})$$

Example

Probabilities for the Markov source

s_j, s_k, s_i	$P(s_i/s_j, s_k)$	$P(s_j, s_k)$	$P(s_j, s_k, s_i)$
000	0.8	5/14	4/14
001	0.2	5/14	1/14
010	0.5	1/14	1/14
011	0.5	1/14	1/14
100	0.5	1/14	1/14
101	0.5	1/14	1/14
110	0.2	5/14	1/14
111	0.8	5/14	4/14

Then the entropy is calculated using (2-24b):

$$H(S) = \sum_{S^3} P(s_j, s_k, s_i) \log \frac{1}{P(s_i/s_j, s_k)}$$

$$= 2 \times \tfrac{4}{14} \log \frac{1}{0.8} + 2 \times \tfrac{1}{14} \log \frac{1}{0.2} + 4 \times \tfrac{1}{14} \log \frac{1}{0.5}$$

$$= 0.81 \text{ bit/binit}$$

Examples

$$H(S) = 0.81\, bit$$

$$H(\overline{S}) = 1.00\; bit$$

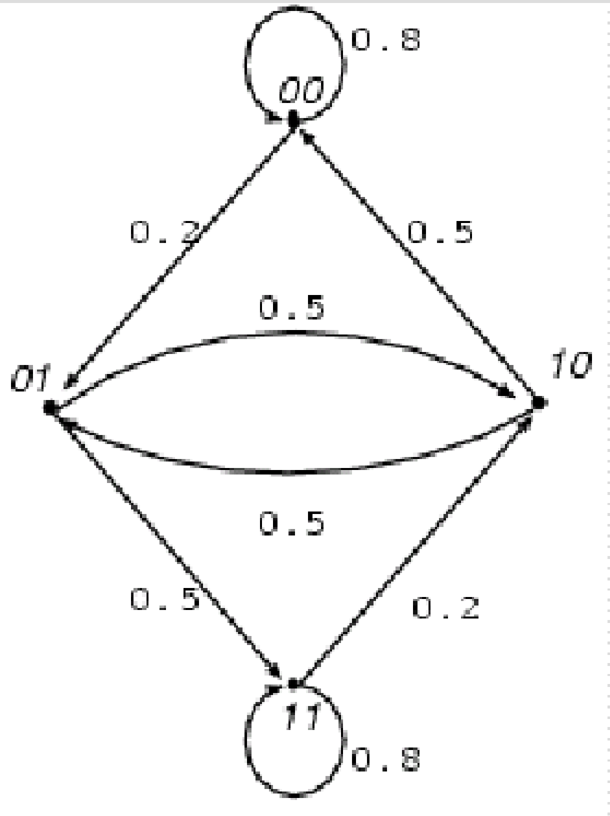
$$H(S^2) = 2H(S) = 1.62\; bits$$

$$H(\overline{S^2}) = \sum_{S^2} P(s_j, s_i) \log \frac{1}{P(s_j, s_k)} =$$

$$= 1.86\; bits$$

$$H(\overline{S^3}) = 2.66\; bits$$

$$H(\overline{S^3}) = 3.47\; bits$$



Note how the sequence approaches *H(S)*.

$$H(\overline{S}) = 1.00\; bit, \quad \frac{H(\overline{S^2})}{2} = 0.93\, bit$$

$$\frac{H(\overline{S^3})}{3} = 0.89\; bit, \quad \frac{H(\overline{S^4})}{4} = 0.87\; bit$$

Example: English

- 27 symbols: 26 alphabets + space

$$H(S) = \log 27$$
$$= 4.75 \quad bits / symbol$$

ZEWRTZYNSADXESYJRQY_WGECIJJ_OBVKRBQPOZB
YMBUAWVLBTQCNIKFMP_KMVUUGBSAXHLHSIE_M

FIGURE 2-6. Zeroth approximation to English.

TABLE 2-2. PROBABILITIES OF SYMBOLS IN ENGLISH (REZA, 1961)

Symbol	*Probability*	*Symbol*	*Probability*
Space	0.1859	N	0.0574
A	0.0642	O	0.0632
B	0.0127	P	0.0152
C	0.0218	Q	0.0008
D	0.0317	R	0.0484
E	0.1031	S	0.0514
F	0.0208	T	0.0796
G	0.0152	U	0.0228
H	0.0467	V	0.0083
I	0.0575	W	0.0175
J	0.0008	X	0.0013
K	0.0049	Y	0.0164
L	0.0321	Z	0.0005
M	0.0198		

$$H(S) = \sum_{S} P_i \log \frac{1}{P_i}$$
$$= 4.03 \quad bits / symbol$$

AI_NGAE_ _ITF_NNR_ASAEV_OIE_BAINTHA_HYR
OO_POER_SETRYGAIETRWCO_ _EHDUARU_EU_C_F
T_NSREM_DIY_EESE_ _F_O_SRIS_R_ _UNNASHOR

FIGURE 2-7. First approximation to English.

Example: English

- 1^{st} order Markov source:

$$H(S) = \sum_{S^2} P(i,j) \log \frac{1}{P(i/j)} = 3.32 \quad bits / symbol$$

URTESHETHING_AD_E_AT_FOULE_ITHALIORT_W
ACT_D_STE_MINTSAN_OLINS_TWID_OULY_TE_T
HIGHE_CO_YS_TH_HR_UPAVIDE_PAD_CTAVED

FIGURE 2-8. Second approximation to English.

- 2^{nd} order Markov source

IANKS_CAN_OU_ANG_RLER_THATTED_OF_TO_S
HOR_OF_TO_HAVEMEM_A_I_MAND_AND_BUT_
WHISSITABLY_THERVEREER_EIGHTS_TAKILLIS_TA

FIGURE 2-9. Third approximation to English.

Example: English

- Word-based zero-memory source

REPRESENTING AND SPEEDILY IS AN GOOD APT OR COME CAN DIFFERENT NATURAL HERE HE THE A IN CAME THE TO OF TO EXPERT GRAY COME TO FURNISHES THE LINE MESSAGE HAD BE THESE

FIGURE 2-10. Fourth approximation to English.

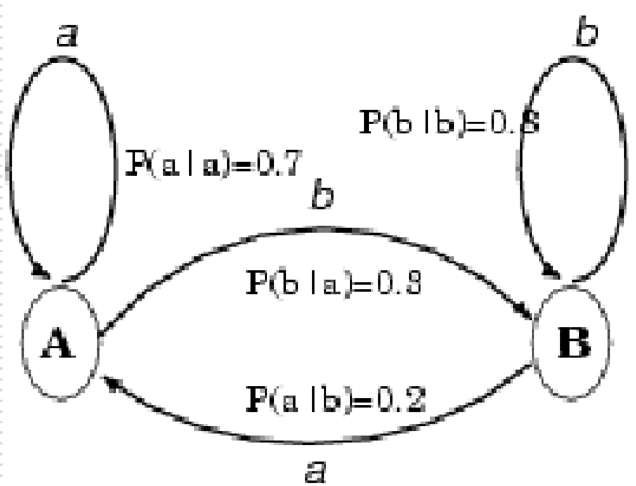
- Word-based 1st order Markov source

THE HEAD AND IN FRONTAL ATTACK ON AN ENGLISH WRITER THAT THE CHARACTER OF THIS POINT IS THEREFORE ANOTHER METHOD FOR THE LETTERS THAT THE TIME OF WHO EVER TOLD THE PROBLEM FOR AN UNEXPECTED

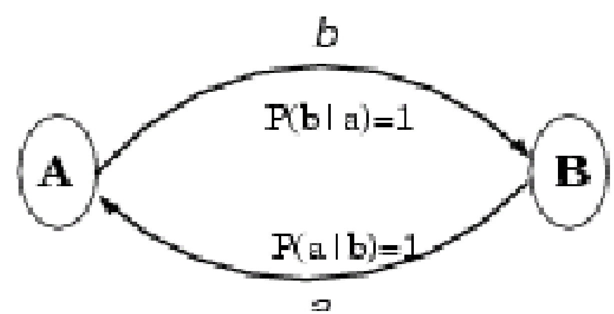
FIGURE 2-11. Fifth approximation to English.

Estimation of parameters of Markov source

- Estimation of $P(s_i/s_j)$ from samples emitted from the information source.



Regular 1st order Markov source



Non-regular 1st order Markov source

Estimation of parameters of Markov source

- The state transition sequence of the Markov source associated the emitted output symbols is uniquely determined. We maximize the following probability *P*, if *P* is a joint probability of *N* observed samples.

$$P = W_0 P_A(a)^{c_1} P_A(b)^{c_2} P_B(a)^{c_3} P_B(b)^{c_4} F$$

,where W_0, F are initial and final state probabilities, respectively. $P_A(a)=P(a|a)$ is conditional probability of state transition.
Now find conditional probabilities which maximize $\log P$ under the following constraints by the Lagrangean method.

$$\sum c_i = N,\ P_A(a) + P_A(b) = 1,\ P_B(a) + P_B(b) = 1$$

The optimal conditional probabilities are given by,

$$P_A(a) = \frac{c_1}{c_1 + c_2}, P_A(b) = \frac{c_2}{c_1 + c_2} \qquad P_B(a) = \frac{c_3}{c_3 + c_4}, P_B(b) = \frac{c_4}{c_3 + c_4}$$

Estimation of parameters of Markov source

- The Lagrangean Method:

$$P = W_0 P_A(a)^{C1} \cdot P_A(b)^{C2} \cdot P_B(a)^{C3} \cdot P_B(b)^{C4} \cdot F$$

Our aim is to maximize the above objective function under constraints of $P_A(a)+P_A(b)=1, P_B(a)+P_B(b)=1$. For simplicity, we maximize the $Q=log\ P$ function instead.

$$Q = \log P + \lambda_1 (P_A(a)+P_A(b)-1) + \lambda_2 (P_B(a)+P_B(b)-1)$$

By taking derivative for each parameter, now we have,

$$\frac{\partial Q}{\partial P_A(a)} = \frac{C_1 \cdot P}{P_A(a)} + \lambda_1 = 0, \frac{\partial Q}{\partial P_B(a)} = \frac{C_3 \cdot P}{P_B(a)} + \lambda_2 = 0,$$

$$\frac{\partial Q}{\partial P_A(b)} = \frac{C_2 \cdot P}{P_A(b)} + \lambda_1 = 0, \frac{\partial Q}{\partial P_B(b)} = \frac{C_4 \cdot P}{P_B(b)} + \lambda_2 = 0,$$

$$\frac{\partial Q}{\partial \lambda_1} = P_A(a)+P_A(b)-1 = 0,$$

$$\frac{\partial Q}{\partial \lambda_2} = P_B(a)+P_B(b)-1 = 0$$

Finally, we obtain,

$$P_A(a) = \frac{c_1}{c_1+c_2}, P_B(a) = \frac{c_3}{c_3+c_4},$$

$$P_A(b) = \frac{c_2}{c_1+c_2}, P_B(b) = \frac{c_4}{c_3+c_4}$$

Estimation of parameters of Markov source

$$P_A(a)=\frac{c_1}{c_1+c_2},P_B(a)=\frac{c_3}{c_3+c_4},P_A(b)=\frac{c_2}{c_1+c_2},P_B(b)=\frac{c_4}{c_3+c_4}$$

- These are nothing but a relative frequencies of symbols sequences observed through state sequences. Now let N_A be a frequency of state *A* and $N_A(b)$ a frequency of the symbol b produced at state *A*.
  $p(b|a)$ can be calculated by,

$$P_A(b)=p(b|a)=\frac{N(a,b)}{N(a)}=\frac{N_A(b)}{N_A}.$$

- Let $P(A,a)$ be a joint probability of symbol a produced at the state *A*, and $P(A)$ be a probability of state *A*.

$$P_A(a)=p(a|a)=\frac{P(A,a)}{P(A)}$$

State transition matrix

- Definition: Matrix representation of conditional probabilities.

$$P = \begin{bmatrix} p(a|a) & p(b|a) \\ p(a|b) & p(b|b) \end{bmatrix}$$

- Let *P(a),P(b)* be state transition matrices for symbol *a* and *b*, and let *A*, W_0=[1,0], W_F=[1,1] be an initial state, an initial state probability and a final state probability.
- $$P(a) = \begin{bmatrix} p(a|a) & 0 \\ p(a|b) & 0 \end{bmatrix} \qquad P(b) = \begin{bmatrix} 0 & p(b|a) \\ 0 & p(b|b) \end{bmatrix}$$
- Now we can calculate a probability for the observed symbols with arbitrary length.

$$M = W_0 P(S_1) P(S_2),\ldots, P(S_m) W_F^t, \quad S_i = \{a,b\}$$

State transition matrix

- Limit distribution: Let W_0 be an initial state probability vector with an initial probability π_i at state i, time $n=0$, and let P and $W_n = [\pi_1^{(n)}, \pi_2^{(n)}, \ldots, \pi_k^{(n)}]$ be a state probability vector at state j, time n.
  Limit distribution is given by,

$$\overline{W} = \lim_{n\to\infty} W_n = \lim_{n\to\infty} W_0 P^n$$

Regular Markov source

- Definition:
  - P^n converges to an unique matrix P^∞ as n becomes large.
  - Each column vector converges to an unique state probability vector W^∞, where each element is positive.
  - Steady state distribution exists uniquely and is equal to W^∞.
- Steady state distribution is $Z=(z_1,z_2,\ldots,z_k)$, which satisfies,

$$ZP = Z, \quad \sum_{i=1}^{k} Z_i = 1.$$

- Example:
  The steady state vector is,

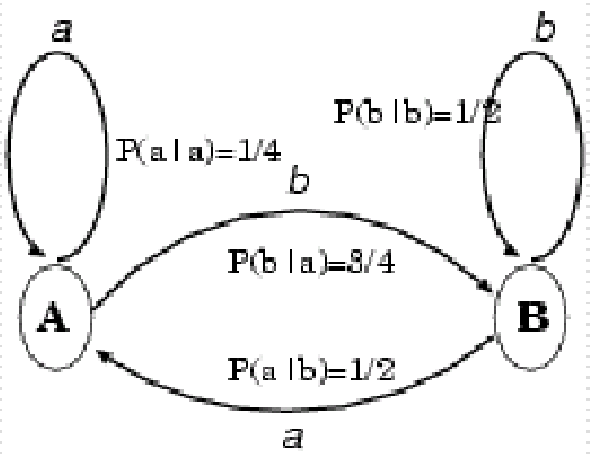
$$P = \begin{bmatrix} 0.7 & 0.3 \\ 0.2 & 0.8 \end{bmatrix}$$

$$0.7z_1 + 0.2z_2 = z_1$$
$$0.3z_1 + 0.8z_2 = z_2$$

$$Z_1 = 0.4, \; Z_2 = 0.6$$

Example

$$P(a|a) = P(a|A) = 1/4, P(b|a) = P(b|A) = 3/4,$$
$$P(a|b) = P(a|B) = 1/2, P(b|b) = P(b|B) = 1/2$$



$$P = \begin{bmatrix} P(a|A) & P(b|A) \\ P(a|B) & P(b|B) \end{bmatrix} = \begin{bmatrix} 1/4 & 3/4 \\ 1/2 & 1/2 \end{bmatrix}$$

$$(Z_a, Z_b) = (Z_a, Z_b)\begin{bmatrix} 1/4 & 3/4 \\ 1/2 & 1/2 \end{bmatrix}$$

$$Z_a = \frac{1}{4}Z_a + \frac{1}{2}Z_b, Z_a + Z_b = 1$$

Now we have, $Z_a = P(A) = \dfrac{2}{5}, Z_b = P(B) = \dfrac{3}{5}.$

Entropy is given by, $H(s) = \sum_{s^2} P(S_i|S_j)P(S_j)\log\dfrac{1}{P(S_i|S_j)}$

$$= P(a|A)P(A)\log\frac{1}{P(a|A)} + P(b|A)P(A)\log\frac{1}{P(b|A)}$$
$$+ P(a|B)P(B)\log\frac{1}{P(a|B)} + P(b|B)P(B)\log\frac{1}{P(b|B)} = 0.92$$

Example

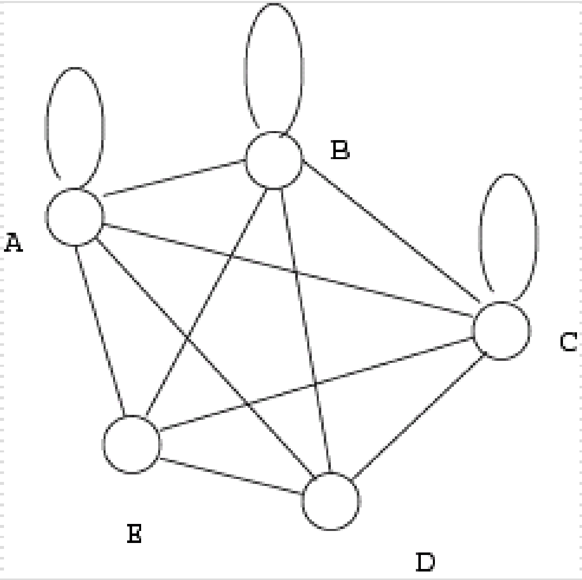
- Entropy of the extended Markov source is,

$$H(\bar{S}) = \frac{2}{5}\log\frac{1}{2/5} + \frac{3}{5}\log\frac{1}{3/5} = 0.97$$

Goal of 2^{nd} day

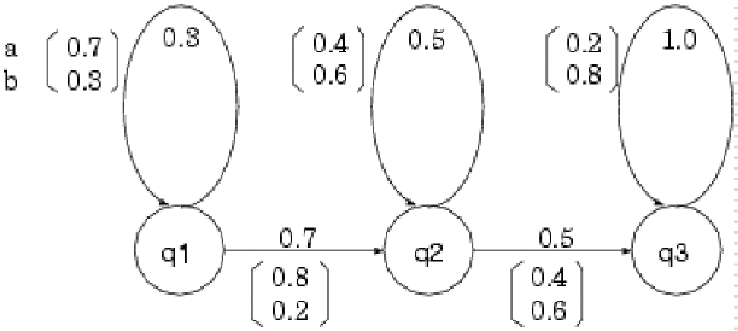
Hidden Markov information source

- Information source with k symbols can be represented by nth Markov source with k^n states.
  If we merge states which have similar behavior, we can have a non-deterministic automata. This is called a hidden Markov source model.
  The hidden Markov source model doesn't have unique state sequence for the observed symbol sequence.



Hidden Markov source

- Definition: Non-deterministic probabilistic automata or Markov source model. The unique state sequence cannot be obtained by observed symbol sequences.



- If we let an initial state be q_1, an final state be q_3, the symbol sequence *abab* can be produced by the following state sequences.

$$Q_1 = q_1 \oplus q_1 \oplus q_1 \oplus q_2 \oplus q_3,\ Q_2 = q_1 \oplus q_1 \oplus q_2 \oplus q_2 \oplus q_3$$
$$Q_3 = q_1 \oplus q_1 \oplus q_2 \oplus q_3 \oplus q_3,\ Q_4 = q_1 \oplus q_2 \oplus q_2 \oplus q_2 \oplus q_3$$
$$Q_5 = q_1 \oplus q_2 \oplus q_2 \oplus q_3 \oplus q_3,\ Q_6 = q_1 \oplus q_2 \oplus q_3 \oplus q_3 \oplus q_3$$

Hidden Markov source

P(Q_1) can be calculated by

$$P_{Q_1} = 0.3*0.7*0.3*0.3*0.7*0.8*0.5*0.6 = 0.0031752$$

$$\vdots$$

Now we have,

$$P = \sum P_{Q_1} = 0.734832$$

- [Forward calculation]: Now let the observed symbol sequence for the source,

$$X = x_1, x_2, \ldots, x_I.$$

- We try to estimate probability of P(X|M) assuming a hidden Markov information source. An initial and final probabilities holds,

$$q_0^{(k)} \in I, q_I^{(k)} \in F$$

- ,where I and F are an initial and final state set.

Probability of observed symbol sequence

- The probability of the observed symbol sequence x on the model M is given by,

$$P(X \mid M) = \sum_{Q_k} \pi_0^{(k)} \prod_{i=1}^{I} a_{q_{i-1},q_i}^{(k)} \cdot b_{q_{i-1},q_i}^{(k)}(x_i)$$

$$P(X \mid M) = \sum_{Q_k} P(X, Q_k \mid M) = \sum_{Q_k} P(X \mid Q_k) P(Q_k)$$

Now we apply 1st order Markov assumption,

$$P(Q_k) = \prod_i \underline{P(q_i^{(k)} \mid q_{i-1}^{(k)})} \quad \text{State transition probability}$$

Now,

$$P(X \mid Q_k) = \prod_i \underline{P(X_i \mid q_{i-1}^{(k)} \to q_i^{(k)})} \quad \text{Emission probability}$$

$$P(X \mid M) = \sum_{Q_k} \prod_i P(X_i \mid q_{i-1}^{(k)} \to q_i^{(k)}) \prod_i P(q_i^{(k)} \mid q_{i-1}^{(k)})$$

$$= \sum_{Q_k} \prod_i P(q_i^{(k)} \mid q_{i-1}^{(k)}) P(X_i \mid q_{i-1}^{(k)} \to q_i^{(k)})$$

$$= \sum_{Q_k} \prod_i a_{q_{i-1}q_i}^{(k)} b_{q_{i-1}q_i}^{(k)}(x_i)$$

Hidden Markov source

- Now let $\pi_i : \ (\sum_{q_i \in L} \pi_i = 1)$ be probabilities of the initial state, the probability of observed symbol sequence given the model is,

$$P(X \mid M) = \sum_{allQ_k} \pi_0{}^{(k)} \prod_{i=1}^{I} a^{(k)}{}_{q_{i-1},q_i} \cdot b^{(k)}{}_{q_{i-1},q_i}(x_i)$$

and, let forward probabilities in the following,

$$\alpha(i,0) = \pi_i \quad for \quad i = 1,2 \ldots S$$

we get

$$\alpha(i,t) = \sum_j \alpha(j,t-1) \cdot a_{ji} \cdot b_{ji}(x_t)$$

$$P(X \mid M) = \sum_{i,i \in F} \alpha(i,I)$$

Probability of observed symbol sequence

- If we apply a forward probability α ,

$$\alpha(i,t) = \sum_j \alpha(i,t-1) a_{ji} b_{ji}(x_i)$$

$$P(X \mid M) = \sum_{i \in F} \alpha(i,I)$$

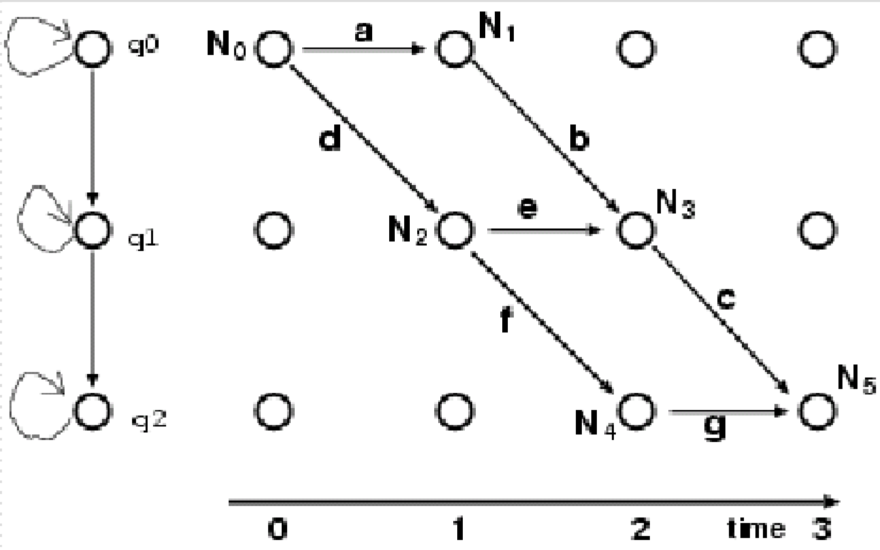
and if we apply a backward probability β ,

$$\beta(i,t) = \sum_j a_{ij} b_{ij}(x_t) \beta(j,t+1)$$

$$P(X \mid M) = \sum_i \beta(i,0) \pi_i$$

Trellis calculation

- Three paths:
  abc+dec+dfg



α (state No. time)	Node No.	path
α (0,1)	N1	a
α (1,1)	N2	d
α (1,2)	N3	ab+de
α (2,2)	N4	df
α (2,3)	N5	(ab+de)c+dfg =abc+dec+dfg

Parameter estimation of HMM source

- State transition sequence cannot be determined uniquely in the HMM while the symbol sequence is observed. Once number of transitions between states is obtained, state transition probabilities and emission probabilities can be estimated easily.
- EM (Expectation and Maximization) algorithm:
  Iterative algorithm for parameter estimation.
  - Expectation Step:
    Find state sequence to observed sequence based on the assumed HMM model parameters.
  - Maximization Step:
    Estimate HMM parameters along the state sequences, which maximize the probability to observed symbol sequence.

,here HMM parameters include state transition parameters and emission parameters.

EM algorithm

- Leonard Baum proved the following important inequation.

$$P_{\hat{\theta}}(X) \geq P_{\theta}(X) \qquad \text{with equality if, and only if} \quad \hat{\theta} = \theta. \cdots (1)$$

,where θ is an assumed HMM parameter set, $\hat{\theta}$ is an estimated HMM parameter set by EM algorithm.

- Let $A=\{a_i\}$ be a state sequence estimated by the observed symbol sequence. We modify the objective function as follows,

$$P_{\hat{\theta}}(X) = P_{\hat{\theta}}(X)\frac{P_{\hat{\theta}}(A, X)}{P_{\hat{\theta}}(A, X)} = \frac{P_{\hat{\theta}}(A, X)}{P_{\hat{\theta}}(A|X)}. \cdots (2)$$

by taking logarithm,

$$\log P_{\hat{\theta}}(X) = \log P_{\hat{\theta}}(A, X) - \log P_{\hat{\theta}}(A|X). \cdots (3)$$

Now we take expectation, $E_{\theta}[]_{A|X}$ over estimated state sequences,

$$E_{\theta}[\log P_{\hat{\theta}}(X)]_{A|X} = \sum_{a_i} P_{\theta}(a_i \mid X) \log P_{\hat{\theta}}(X) = \log P_{\hat{\theta}}(X). \cdots (4)$$

EM algorithm

If we substitute (4) with (3),

$$\begin{aligned}\log P_{\hat{\theta}}(X) &= E_{\theta}[\log P_{\hat{\theta}}(X)]_{A|X} \\ &= E_{\theta}[\log P_{\hat{\theta}}(A,X)]_{A|X} - E_{\theta}[\log P_{\hat{\theta}}(A|X)]_{A|X} \\ &= \sum_{a_j} P_{\theta}(a_i|X)\log P_{\hat{\theta}}(a_i,X) - \sum_{a_i} P_{\theta}(a_i|X)\log P_{\hat{\theta}}(a_i|X) \cdots (5)\end{aligned}$$

Now we recall Jensen's inequality.

$$\int_R f(x)\log f(x)dx \geq \int_R f(x)\log g(x)dx$$ with equality if, and only if $f(x)=g(x)$

,where $f(x)$,$g(x)$ are probability density function.

Apply Jensen's inequality to the second term in the right side .

$$\sum_{a_i} P_{\theta}(a_i|X)\log P_{\hat{\theta}}(a_i|X) \leq \sum_{a_i} P_{\theta}(a_i|X)\log P_{\theta}(a_i|X) \cdots (6)$$

,with equality if, and only if

$$P_{\hat{\theta}}(a_i|X) = P_{\theta}(a_i|X), \quad \textit{that is,} \quad \hat{\theta}=\theta.$$

EM algorithm

Now we have,

$$\log P_{\hat{\theta}}(X) \geq \sum_{a_i} P_\theta(a_i \mid X) \log P_{\hat{\theta}}(a_i, X) - \sum_{a_i} P_\theta(a_i \mid X) \log P_\theta(a_i \mid X).$$

If we set 1st term in right side to be as follows,

Namely,

$$\sum_{a_i} P_\theta(a_i \mid X) \log P_{\hat{\theta}}(a_i, X) \geq \sum_{a_i} P_\theta(a_i \mid X) \log P_\theta(a_i, X) \cdots (7)$$

$$E_\theta[\log P_{\hat{\theta}}(A, X)]_{A|X} \geq E_\theta[\log P_\theta(A, X)]_{A|X}, \cdots (8)$$

Equation (5) holds,

$$\log P_{\hat{\theta}}(X) \geq \sum_{a_i} P_\theta(a_i \mid X) \log P_\theta(a_i, X) - \sum_{a_i} P_\theta(a_i \mid X) \log P_\theta(a_i \mid X) \cdots (9)$$

EM algorithm

- In summary,

$$
\begin{aligned}
&\log P_{\hat{\theta}}(X) \\
&\quad = \sum_{a_i} P_{\theta}(a_i \mid X) \log P_{\hat{\theta}}(a_i, X) - \sum_{a_i} P_{\theta}(a_i \mid X) \log P_{\hat{\theta}}(a_i \mid X) \\
&\quad \geq \sum_{a_i} P_{\theta}(a_i \mid X) \log P_{\hat{\theta}}(a_i, X) - \sum_{a_i} P_{\theta}(a_i \mid X) \log P_{\theta}(a_i \mid X) \\
&\quad \geq \sum_{a_i} P_{\theta}(a_i \mid X) \log P_{\theta}(a_i, X) - \sum_{a_i} P_{\theta}(a_i \mid X) \log P_{\theta}(a_i \mid X) \\
&\quad = \log P_{\theta}(X).
\end{aligned}
$$

- If equation (7) holds, we obtain parameters which satisfy,

$$\log P_{\hat{\theta}}(X) \geq \log P_{\theta}(X).$$

Parameter estimation by EM algorithm

- As in the previous slides, parameter estimation can be achieved by maximizing $E_\theta\left[\log P_{\hat{\theta}}(A, X)\right]_{A|X}$.

$$E \equiv \sum_{a_k} P(a_k \mid X) \log P_{\hat{\theta}}(a_k, X)$$
$$= \sum_{a_k} \frac{P(a_k, X)}{P(X)} \log P_{\hat{\theta}}(a_k, X)$$

, where $\frac{P(a_k, X)}{P(X)}$ can be calculated using parameter θ .

  - Numerator of $\frac{P(a_k, X)}{P(X)}$ is a joint probability of events of observing X and state sequence a_k.
  - Denominator of $\frac{P(a_k, X)}{P(X)}$ is a probability of observing X based on the HMM.

Parameter estimation by EM algorithm

- Now, we have $P_{\hat{\theta}}(a_k, X)$ by counting state transitions along the state sequence a_k.

$$P_{\hat{\theta}}(a_k, X) = \pi_0^{(k)} \prod_{i=I}^{I} a_{q_i-1q_i}^{(k)} b_{q_i-1q_i}^{(k)}(x_i)$$

$$= \pi_0^{(k)} a_{ijq_i}^{(k)c_{ij}} b_{ij}^{(k)d_{ij}}(x_i)$$

,where c_{ij} and d_{ij} are counts of state transition a_{ij} and $b_{ij}(x_t)$, respectively.
Then E can be re-written by,

$$E = \sum_{a_k} \frac{P(a_k, X)}{P(X)} \log \pi_0^{(k)} a_{ij}^{(k)c_{ij}} b_{ij}^{(k)d_{ij}}(x_i)$$

$$= \log \pi_0^{(k)\sum_{a_k}\frac{P(a_k,X)}{P(X)}} a_{ij}^{(k)\sum_{a_k}\frac{P(a_k,X)}{P(X)}c_{ij}} b_{ij}^{(k)\sum_{a_k}\frac{P(a_k,X)}{P(X)}d_{ij}}(x_i)$$

if we let $c'_{ij} = \sum_{a_k} \frac{P(a_k, X) c_{ij}^{(k)}}{P(X)}, d'_{ij} = \sum_{a_k} \frac{P(a_k, X) d_{ij}^{(k)}}{P(X)}$, we have E as follows.

$$E = \log \pi'^{(k)}_0 a_{ij}^{c'_{ij}} b_{ij}^{d'_{ij}}(x_i)$$

Parameter estimation by EM algorithm

- This is nothing but a probability function of a Markov source. Thus we can obtain parameters by maximization of E, with $\frac{\partial E}{\partial a_{ij}} = 0$.

  For a_{ij}, $c'_{ij} = \sum_{a_k} \frac{P(a_k, X) c_{ij}^{(k)}}{P(X)}$ can be thought as a relative counts of the state transition from state i to state j. Thereby, we have,

$$\hat{a}_{ij} = \frac{c'_{ij}}{\sum_j c'_{ij}}.$$

  If use $\gamma(i,j,t) = \frac{\alpha_i(t) a_{ij} b_{ij}(x_t) \beta_j(t+1)}{\sum_t \alpha_i(t) \beta_i(t)}$, we have,

$$\hat{a}_{ij} = \frac{c'_{ij}}{\sum_j c'_{ij}} = \frac{\sum_t \gamma(i,j,t)}{\sum_{t,j} \gamma(i,j,t)}$$

Parameter estimation of HMM source

- First we define backward probability $\beta(i,t)$, which is a probability at state q_i, $time=t$ emitting $x_i, x_{i+1}, x_{t+2}, \ldots, x_I$.. This probability can be efficiently calculated from the final symbol.
- Initial setting:

$$\begin{aligned} &for\ \ q=1, Q_n \\ &\beta(q,0)=1.0 : if\ \ q \in F \\ &\beta(q,0)=0.0 : otherwise \end{aligned}$$

- Iteration of backward path:

$$\begin{aligned} &for\ \ t=I-1, i-2, \ldots, 1, 0 \\ &\quad for\ \ q=1,2,\ldots,Q_n \end{aligned}$$

$$\beta(q,t)=\sum\nolimits_{\{j \in \{j: a_{qj} \neq 0\}} \beta(j,t+1) \cdot a_{qj} \cdot b_{qj}(x_{t+1})$$

- The following relationship holds.

$$\sum_{i \in F} \alpha(q_i, I) = \sum_{q_i \in S} \beta(q_i, 0) \cdot \pi_i$$

Parameter estimation of HMM source

- Let $\gamma(i, j, t)$ be an emission probability producing x_t during a transition from state q_i to q_j. Now $\gamma(i, j, t)$ can be calculated using $\alpha(i, t-1)$ and $\beta(j, t)$.

$$\gamma(i, j, t) = \frac{\alpha(i, t-1) \cdot a_{ij} \cdot b_{ij}(x_t) \cdot \beta(j, t)}{P(x \mid M)}$$

Here, $\gamma(i, j, t)$ represents a probability (relative transition counts) producing x_t during a transition from state q_i to state q_j assuming an HMM $\theta = \{a_{ij}, b_{ij}(x_t), \pi_i\}$.

Parameter estimation of HMM source

- Now we have the following estimation formulae.

$$\pi_i \Leftarrow \frac{\sum_j \gamma(i,j,l)}{\sum_i \sum_j \gamma(i,j,l)}$$

$$a_{ij} \Leftarrow \frac{\sum_t \gamma(i,j,t)}{\sum_t \sum_j \gamma(i,j,t)} = \frac{\sum_t \alpha(i,t-1)\cdot a_{ij}\cdot b_{ij}(x_t)\cdot \beta(j,t)}{\sum_t \alpha(i,t)\cdot \beta(i,t)}$$

$$b_{ij}(k) \Leftarrow \frac{\sum_{t:x_t=k} \gamma(i,j,t)}{\sum_t \gamma(i,j,t)} = \frac{\sum_{t:x_t=k} \alpha(i,t-1)\cdot a_{ij}\cdot b_{ij}(x_t)\cdot \beta(j,t)}{\sum_t \alpha(i,t-1)\cdot a_{ij}\cdot b_{ij}(x_t)\cdot \beta(j,t)}$$

Parameter estimation of HMM

- The calculations above will be iterated until its convergence. Also parameter estimation will be applied not to a single observation but to many symbol observations like,

$$a_{ij} \Leftarrow \frac{\sum_{n=1}^{N} \sum_t \gamma^{(n)}(i,j,t)}{\sum_{n=1}^{N} \sum_t \sum_j \gamma^{(n)}(i,j,t)}.$$

- γ represents a probability that information source produces a symbol x_i during a state transitions from state q_i to q_j, assuming the symbol sequence x is observed regardless to the state sequences.
  At least the calculation for α is the same as that of the Markov source model.

Entropy for HMM source

- Let Entropy per one symbol at a state q_j is given by,

$$H(X \mid q_j) = -\sum_x p(x \mid q_j) \log p(x \mid q_j)$$

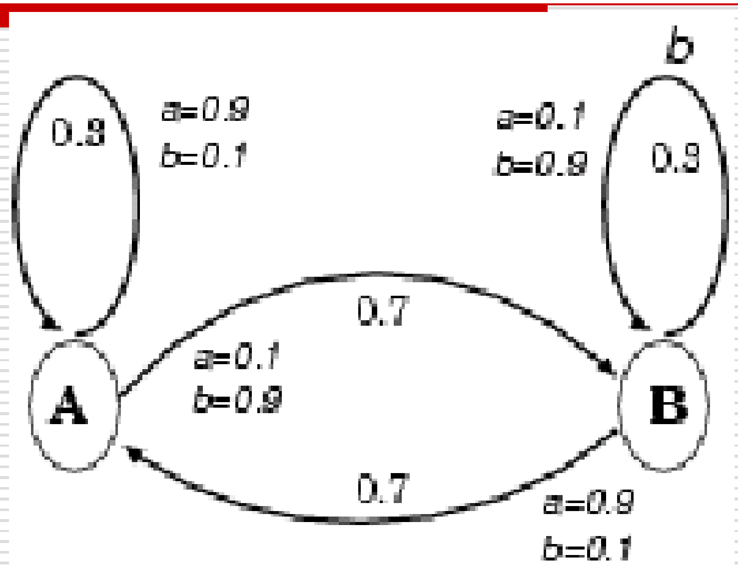
$$p(x \mid q_j) = \sum_k a_{jk} b_{jk}(x)$$

- We obtain Entropy for the HMM taking expectation over all states.

$$H(X) = \sum_j \pi(q_j) H(X \mid q_j)$$

,where $\pi(q_j)$ is steady state probabilities for the HMM states.

An example



- Estimate HMM parameters based on observed symbol sequence *"ba"*.
- Step 1:

State seq.	b		a	$P(a_i, X)$
AAA	0.3×0.1	$\times$	0.3×0.9	$=0.0081$
AAB	0.3×0.1	$\times$	0.7×0.1	$=0.0021$
ABA	0.7×0.9	$\times$	0.7×0.9	$=0.3969$
ABB	0.7×0.9	$\times$	0.3×0.1	$=0.0189$
				$sum\,0.426$

An example

- Step 2:

$$P(a|A) \quad 0.3\times0.9 + 0.7\times0.1 = 0.34$$
$$P(b|A) \quad 0.3\times0.1 + 0.7\times0.9 = 0.66$$
$$P(a|B) \quad 0.7\times0.9 + 0.3\times0.1 = 0.66$$
$$P(b|B) \quad 0.7\times0.1 + 0.3\times0.9 = 0.34$$

$$H(X\mid A) = -0.34\log 0.34 - 0.66\log 0.66 = 0.9264$$
$$H(X\mid B) = -0.66\log 0.66 - 0.34\log 0.34 = 0.9264$$

$$(\log 0.34 = -1.56, \log 0.66 = -0.60)$$

$$P(A) = P(B) = \frac{1}{2}$$

Now we have Entropy for the HMM,

$$H(X) = H(X\mid A)P(A) + H(X\mid B)P(B) = 0.9264$$

An example

- Step 3: Parameter estimation of the HMM.

$$\hat{a}_{AA} = \frac{\sum prob.\, of\;\; state\ sequences\ with\ transition\ A \rightarrow A}{\sum prob.\, of\;\; state\ sequences\ with\ transition\;\; from\ A}$$

$$= \frac{\dfrac{0.0081}{0.426} \times 2 + \dfrac{0.0021}{0.426}}{\dfrac{0.0081}{0.426} \times 2 + \dfrac{0.0021}{0.426} \times 2 + \dfrac{0.3969}{0.426} + \dfrac{0.0189}{0.426}}$$

$$\hat{b}_{AA} = \frac{\sum prob.\, state\ sequences\ with\ transition\ A \rightarrow A\ producing\ symbol\ "\mathrm{a}"}{\sum prob.\, state\ sequences\ with\ transition\ A \rightarrow A}$$

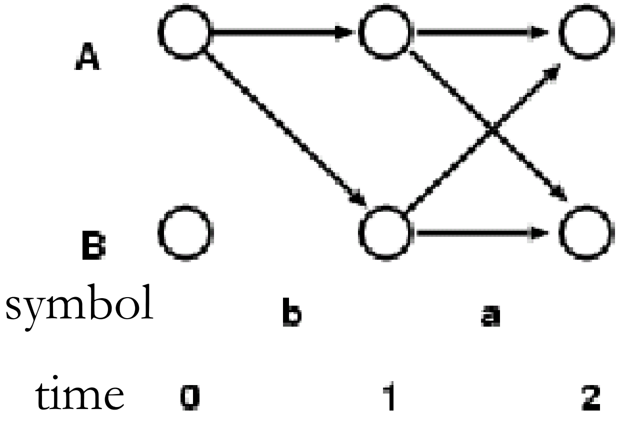
$$= \frac{0.0081}{0.0081 \times 2 + 0.0021} = 0.4426$$

An example

- There is another way of estimation using α, β.

Time			Symbol
0	$\alpha_0(A)=1.0$	$\alpha_0(B)=0.0$	
1	$\alpha_1(A)=\alpha_0(A)\times 0.3\times 0.1 = 0.03$	$\alpha_1(B)=\alpha_0(A)\times 0.7\times 0.9 = 0.63$	b
2	$\alpha_2(A)=\alpha_1(A)\times 0.3\times 0.9 + \alpha_1(B)\times 0.7\times 0.9 = 0.405$	$\alpha_2(B)=\alpha_1(A)\times 0.7\times 0.1 + \alpha_1(B)\times 0.3\times 0.1 = 0.021$	a

2	$\beta_2(A)=1.0$	$\beta_2(B)=1.0$	
1	$\beta_1(A)=\beta_2(A)\times 0.3\times 0.9 + \beta_2(B)\times 0.7\times 0.1 = 0.34$	$\beta_1(B)=\beta_2(A)\times 0.7\times 0.9 + \beta_2(B)\times 0.3\times 0.1 = 0.66$	a
0	$\beta_0(A)=\beta_1(A)\times 0.3\times 0.1 + \beta_1(B)\times 0.7\times 0.9 = 0.426$		b



$$\sum \alpha_2(S_i)=0.426$$

An example

b   a

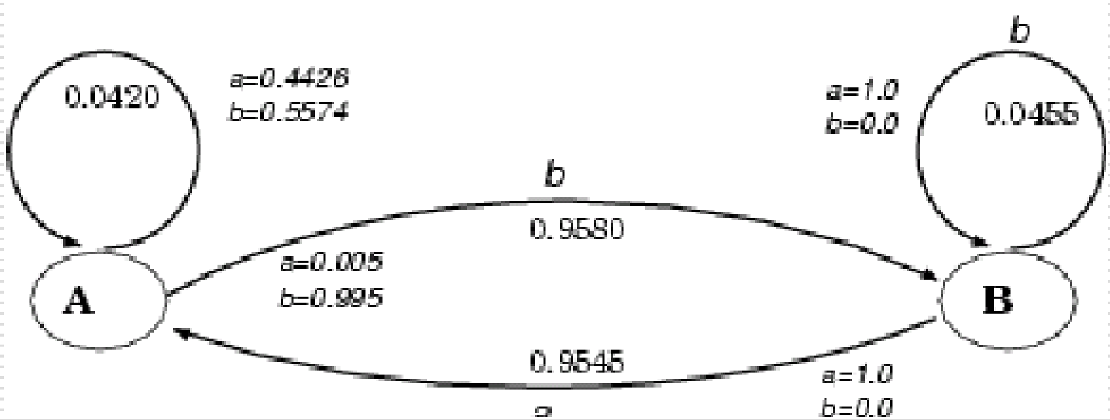
$$\hat{a}_{AA} = \frac{\alpha_0(A)a_{AA}b_{AA}(x_t)\beta_1(A) + \alpha_1(A)a_{AA}b_{AA}(x_t)\beta_2(A)}{\alpha_0(A)\beta_0(A) + \alpha_1(A)\beta_1(A)} = 0.0420$$

a

$$\hat{b}_{AA}(a) = \frac{\alpha_1(A)a_{AA}b_{AA}(x_t)\beta_2(A)}{\alpha_0(A)a_{AA}b_{AA}(x_t)\beta_1(A) + \alpha_1(A)a_{AA}b_{AA}(x_t)\beta_2(A)} = 0.4426$$

b   a

An example



$$P(a\,|\,A) = 0.0420\times0.4426+0.9580\times0.0050 = 0.0234 \quad (18)$$

$$P(b\,|\,A) = 0.0420\times0.5574+0.9580\times0.9950 = 0.9766 \quad (19)$$

$$P(a\,|\,B) = 0.0455\times1.0+0.9545\times1.0 = 1.0 \quad (20)$$

$$P(b\,|\,B) = 0.0 \quad (21)$$

An example

$$H(X\mid A) = -0.0234\log 0.0234 - 0.9766\log 0.9766 \qquad (22)$$

$$= 0.1601 \qquad (23)$$

$$H(X\mid B) = 0 \qquad (24)$$

$$ZP = Z, \sum Z_i = 1$$

$$(Z_{A,} Z_B)\begin{pmatrix} 0.0420 & 0.9580 \\ 0.9545 & 0.0455 \end{pmatrix} = (Z_{A,} Z_B)$$

$$Z_A = 0.499, Z_B = 0.501$$

$$H(X) = H(X\mid A)P(A) + H(X\mid B)P(B) \qquad (25)$$

$$= 0.1601 \times 0.499 = 0.0799 \qquad (26)$$

First goal of 3rd day

Some properties of codes

Information source

$$\left.\begin{array}{c} S_1 \\ S_2 \\ S_3 \\ \cdot \\ \cdot \\ \cdot \\ S_q \end{array}\right\} S \qquad \left.\begin{array}{c} x_1 \\ x_2 \\ x_3 \\ \cdot \\ \cdot \\ \cdot \\ x_r \end{array}\right\} X$$

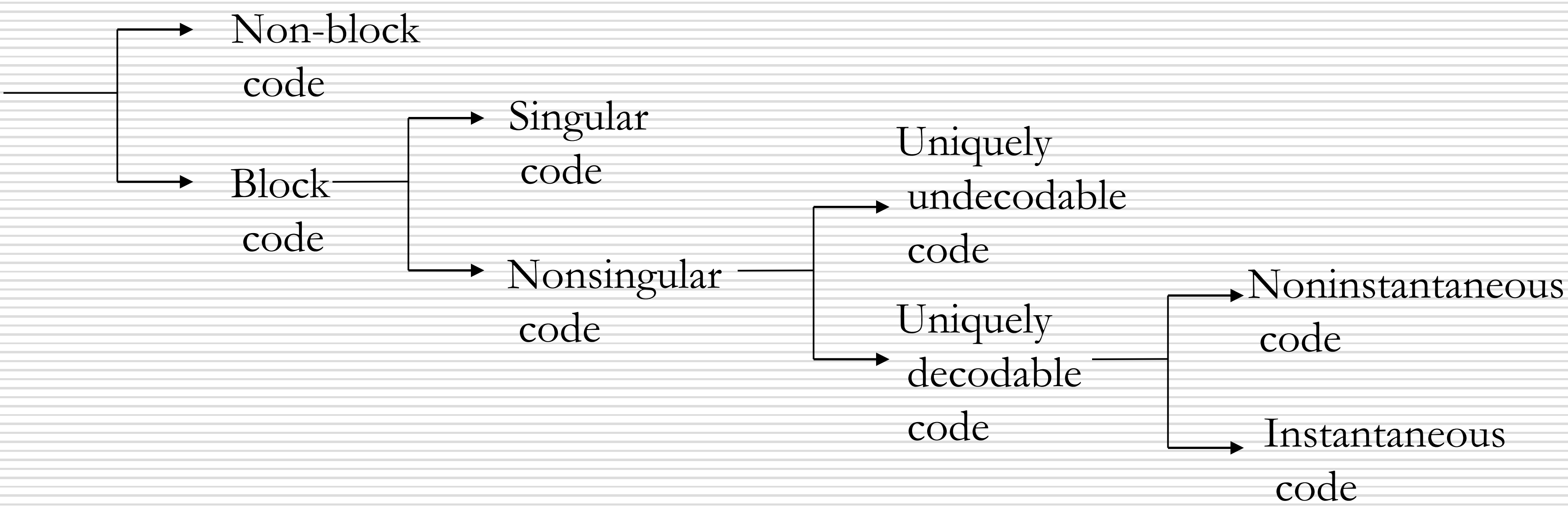
Source alphabet Code alphabet

$$S_i \rightarrow X_i = (xi_1, xi_2, \ldots\ldots, xi_j)$$

Code word

- Definition: Let the set of symbols comprising a given alphabet be called $S=\{s_1,s_2,\ldots,s_q\}$. Then we define a code as a mapping of all possible sequences of symbols of S into sequences of symbols of some other alphabet $X=\{x_1,x_2,\ldots,x_r\}$. We call S the source alphabet and X the code alphabet.

Classification of coding



Block code

- Definition: A block code is a code which maps each of the symbols of the source alphabet S into a fixed sequence of symbols of the code alphabet X. These fixed sequences of the code alphabet (sequences of x_j) are called code words. We denote the code word corresponding to the source symbol s_i by X_i. Note that X_i denotes a sequence of x_j's.

Source symbols	code
S_1	0
S_2	11
S_3	00
S_4	01

Nonsingular block code

- Definition: A block code is said to be *nonsingular* if all the words of the code are distinct.

Source symbols	code
S_1	0
S_2	11
S_3	00
S_4	01

It is still possible for a given sequence of code symbols to have an ambiguous origin. For example, the sequence 0011 might represent either s_3s_2 or $s_1s_1s_2$.

Extension of block code

- Definition: The *n*th extension of a block code which maps the symbols s_i into the code words X_i is the block code which maps the sequences of source symbols $(s_{i1}, s_{i2}, \ldots, s_{in})$ into the sequences of code words $(X_{i1}, X_{i2}, \ldots, X_{in})$.

Source symbols	code	Source symbols	code
S_1S_1	00	S_3S_1	000
S_1S_2	011	S_3S_2	0011
S_1S_3	000	S_3S_3	0000
S_1S_4	001	S_3S_4	0001
S_2S_1	110	S_4S_1	010
S_2S_2	1111	S_4S_2	0111
S_2S_3	1100	S_4S_3	0100
S_2S_4	1101	S_4S_4	0101

Uniquely decodable code

- Definition: A block code is said to be *uniquely decodable* if, and only if, the nth extension of the code is nonsingular for every finite n.

- Any two sequences of source symbols of *the same length* are distinct sequences of code symbols, if the code is uniquely decodable.

- Two sequences of the different length should also be distinct, if the code is uniquely decodable.

Suppose we have source symbol sequences S_1 and S_2 which lead to the same sequence of code symbols, X_0, and S_1 and S_2 may be sequences of source symbols of different lengths.
Now let us form two new sequence source symbols, S_1' and S_2', where $S_1'=S_2S_1$, $S_2'=S_1S_2$. Both of S_1' and S_2' are sequence X_0 followed by X_0 with the same length. Thus, the code doesn't satisfy the condition of unique decodability.

Instantaneous code

Source symbol	Code A	Code B	Code C
S_1	00	0	0
S_2	01	10	01
S_3	10	110	011
S_4	11	1110	0111

- Code A : This code is uniquely decodable, since all codes have the same length and distinct.
- Code B : This code is also uniquely decodable, since it is non-singular. It is called "*Comma code*", which separates code by comma, *0* in this example.
- Code C : This code is also uniquely decodable. However, we are not able to decode the sequence, word by word, as it is received. We can decode only after receiving 0 of the next code word.

Instantaneous code

- Definition: A uniquely decodable code is said to be *instantaneous* if it is possible to decode each word in a sequence without reference to succeeding code symbols.

- Code A and code B are instantaneous. However, code C is not instantaneous. A more general method to know whether instantaneous or not would be helpful.

- Definition: Let $X_i = x_{i1}x_{i2}\ldots x_{im}$ be a word of some code. The sequence of code symbols $(x_{i1}x_{i2}\ldots x_{ij})$, where $j \le m$, is called a *prefix* of the code word X_i.

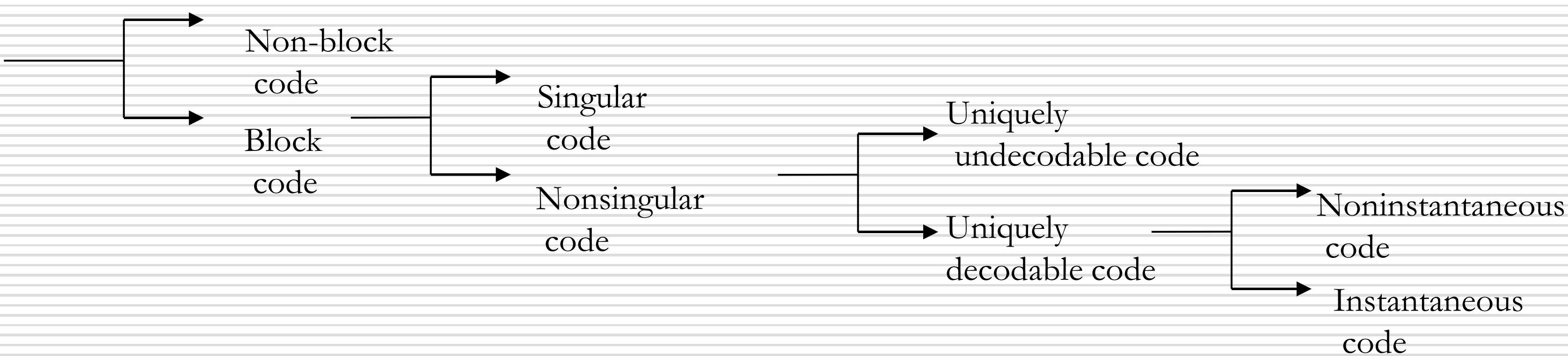
- Ex. *0,01,011,0111* are prefixes of *0111*.

Instantaneous code

- A necessary and sufficient condition for a code to be instantaneous is that no complete word of the code be a prefix of some other code word,.
- Sufficient part:
  - If no word is the prefix of some other word, we may decode any received sequence of code symbols comprised of code words in a direct manner.
  - We scan the received sequence of code symbols until we come to a subsequence which comprises a complete code word.
  - The subsequence must be this code word since by assumption it is not the prefix of any other code word.

Instantaneous code

- Necessary part:
  - We assume that there exists some word of our code, say Xi, which is also a prefix of some other word X_j.
  - Now, if we scan a received sequence of code symbols and come upon the subsequence X_i, this subsequence may be a complete word, or it may be just the first part of word X_j.
  - We cannot possibly tell which of these alternatives is true, however, until we examine more code symbols of the main sequence-thus the code is not instantaneous.



Construction of an Instantaneous code

- Example code synthesis:
  - Assign 0 to symbol s_1: $s_1 \to 0$
  - If we assign 1 to symbols s2, this would leave us with no symbols. we might have, $s_2 \to 10$
  - This, in turn, would require us to start remaining code words with 11. If , $s_3 \to 110$ then the only three-binit prefix still unused is 111.
  - And we might set, $s_4 \to 1110$ and $s_4 \to 1111$
- Other alternatives:
  - If we synthesize another binary instantaneous code.
  - Then we may set.
  - We still have two prefixes of length 2 unused.

$$s_1 \to 00$$
$$s_2 \to 01$$
$$s_3 \to 10$$
$$s_4 \to 110$$
$$s_5 \to 111$$

Kraft inequality

- Constraints on the size of words of an instantaneous code. Consider an instantaneous code with source alphabet,
$$S = \{s_1, \cdots, s_q\}$$
and code alphabet $X=\{x_1,x_2,\ldots,x_r\}$. Let the code words be $X_1,X_2,\ldots,X_q$ and define the length (number of code symbols) of word X_i as l_i. It is often desirable that the lengths of the code words of our code be as small as possible. Necessary and sufficient conditions for the existence of an instantaneous code with word lengths $l_1,l_2,\ldots,l_q$ are provided by the *Kraft inequality*.
- Kraft inequality: A necessary and sufficient condition for the existence of an instantaneous code with word lenghts $l_1,l_2,\ldots,l_q$ is that
$$\sum_{i=1}^{q} r^{-l_i} \le 1$$
where r is the number of different symbols in the code alphabet.

Kraft inequality

- For the binary case, the Kraft inequality tell us that the li must satisfy the equation.

$$\sum_{i=1}^{q} 2^{-l_i} \leq 1$$

Source symbols	Code *A*	Code *B*	Code *C*	Code *D*	Code *E*
S_1	00	0	0	0	0
S_2	01	100	10	100	10
S_3	10	110	110	110	110
S_4	11	111	111	11	11

Kraft inequality

- Code A: $\sum_{i=1}^{4} 2^{-l_i} = 2^{-2} + 2^{-2} + 2^{-2} + 2^{-2} = 1$
- Kraft inequality does not tell that code A is an instantaneous code. The inequality is merely a condition on the word lengths of the code and not on the words themselves.
- Code B: $\sum_{i=1}^{4} 2^{-l_i} = 2^{-1} + 2^{-3} + 2^{-3} + 2^{-3} = \frac{7}{8} \le 1$
- Code C: $\sum_{i=1}^{4} 2^{-l_i} = 2^{-1} + 2^{-2} + 2^{-3} + 2^{-3} = 1$
- Code D: $\sum_{i=1}^{4} 2^{-l_i} = 2^{-1} + 2^{-3} + 2^{-3} + 2^{-2} = 1$

  Code D is not an instantaneous code.
- Code E: $\sum_{i=1}^{4} 2^{-l_i} = 2^{-1} + 2^{-2} + 2^{-3} + 2^{-2} = 1\frac{1}{8}$

  Code E is not an instantaneous code.

One more example

- Suppose we wish to encode the outputs of a decimal source, S={0,1,2,…,9}, into a binary instantaneous code.
  Suppose there is some reason for encoding the 0 and 1 symbols of the decimal source into relatively short binary code words.
  If we were to encode 0s and 1s from the source as,

$$0 \to 0$$

$$1 \to 10$$

  If we require all these eight code words to be of the same length, say l, the Kraft inequality will provide us with a direct answer to the equation.

$$\sum_{i=0}^{9} 2^{-l_i} \leq 1$$

  By assumption we have $l_0=1, l_1=2,$ *and* $l_2=l_3=\ldots=l_9=l$. Then,

  or
$$\frac{1}{2}+\frac{1}{4}+8(2^{-l}) \leq 1$$

$$l \geq 5$$

The Kraft inequality - Proof

- First we prove that the inequality is sufficient for the existence of an instantaneous code by actually constructing an instantaneous code, satisfying

$$\sum_{i=1}^{q} r^{-l_i} \le 1 \quad (1) \qquad \sum_{i}^{L} n_i = q \qquad L \text{ is largest of } l_i$$

(1) can be written as, $\sum_{i=1}^{L} n_i r^{-i} \le 1$

on multiplying by r^L, $\sum_{i=1}^{L} n_i r^{-i+L} \le r^L$

$$n_1 r^{-1+L} + n_2 r^{-2+L} + \ldots + n_L \le r^L$$

rearranging terms,

$$n_L \le r^L - n_1 r^{L-1} - n_2 r^{L-2} - \ldots - n_{L-1} r$$

$$n_L \ge 0$$

$$n_{L-1} r \le n_L + n_{L-1} r \le r^L - n_1 r^{L-1} - \ldots - n_{L-2} r^2$$

dividing by r,

$$n_{L-1} \le r^{L-1} - n_1 r^{L-2} - \ldots - n_{L-2} r$$

iterate the operation,

$$n_3 \le r^3 - n_1 r^2 - n_2 r = ((r - n_1) r - n_2) r$$

$$n_2 \le r^2 - n_1 r = r(r - n_1)$$

$$n_1 \le r$$

The Kraft inequality - Proof

- Steps:
  - We assign n_1 word of length 1.
  - There are r possible such words that we may form, using an r-symbol code alphabet.
  - We can select these n_1 code symbols arbitrarily, $n_1 \leq r$
  - We are then left with r-n_1 permissible prefixes of length 1.
  - By adding one symbol to the end of each of these permissible prefixes, we may form as many as, $(r-n_1)r = r^2 - n_1 r$ words of length 2.
  - As before, we choose our n_2 words arbitrarily among our r_2-$n_1 r$ choices, we are left with, $(r-n_1)r - n_2$ unused prefixes of length 2, from which we may form permissible prefixes of length 3. $(r^2 - n_1 r - n_2)r = r^3 - n_1 r^2 - n_2 r$

McMillan's inequality

- Proof for the necessity conditions for uniquely decodable codes ?
  - Consider the quantity,
    we have q_n terms, each terms of

    $$(\sum_{i=1}^{q} r^{-l_i})^n = (r^{-l_1} + r^{-l_2} + \ldots + r^{-l_q})^n$$

    $$r^{-l_{i_1}-l_{i_2}-l_{i_3}\cdots-l_{i_n}} = r^{-k}, \quad k = l_{i_1} + l_{i_2} + \ldots l_{i_n}.$$

    If we let L be the maximum of the word length l_i.

    $$n \le k \le nL$$
  - We define N_k as the number of terms of the form r-k, then,

    $$(\sum_{i=1}^{q} r^{-l_i})^n = (\sum_{k=n}^{nL} N_k r^{-k})$$
  - N_k is also the number of strings of n code words that can be formed so that each string has a length of exactly k code symbols.
  - If the code is uniquely decodable, N_k must be no greater than r^k, the number of distinct *r-ary* sequences of length k. Thus, we have

    $$(\sum_{i=1}^{q} r^{-l_i})^n \le \sum_{k=n}^{nL} r^k r^{-k}$$

    $$\le nL - n + 1 \le nL \quad (*).$$
  - *Bernulli*'s inequality:
    For $x>1$, n is arbitrarily large, $x^n > nl$ holds.
    Considering this inequality and equation (*), we can prove,

    $$\sum_{i=1}^{q} r^{-l_i} \le 1$$

Example

- Assume we wish to encode a source with 10 source symbols into a trinary instantaneous code with word length 1,2,2,2,2,2,3,3,3,3.
  Applying the test of the Kraft inequality, we have,

$$\sum_{i=1}^{10} 3^{-l_i} = \frac{1}{3} + 5\left(\frac{1}{9}\right) + 4\left(\frac{1}{27}\right)$$

$$= \frac{28}{27} > 1$$

  This doesn't satisfy the inequality.
- Assume we with to encode symbols from a source with nine symbols into a trinary instantaneous code with lengths 1,2,2,2,2,2,3,3,3. Applyint the test of the Kraft inequality, we have,
  We show the example.

$$\sum_{i=1}^{9} 3^{-l_i} = \frac{1}{3} + 5\left(\frac{1}{9}\right) + 3\left(\frac{1}{27}\right)$$

$$= 1$$

$$s_1 \to 0, s_2 \to 10, s_3 \to 11,$$

$$s_4 \to 12, s_5 \to 20, s_6 \to 21,$$

$$s_7 \to 220, s_8 \to 221, s_9 \to 222$$

Coding information sources

- For a given source alphabet and a given code alphabet, however, we can construct many instantaneous codes forces us to find a criterion by which we may choose among the codes.
  Perhaps the natural criterion for this selection, although by no means the only possibility, is length.

- Definition: Let a block code transform the source symbols $s_1, s_2, \ldots, s_q$ into the code words $X_1, X_2, .., X_q$. Let the probabilities of the source symbols be $P_1, P_2, \ldots, P_q$, and let the lengths of the code words be $l_1, l_2, \ldots, l_q$. Then we define L, the average length of the code, by the equation

$$L = \sum_{i=1}^{q} P_i l_i$$

Coding information source

- Average length and Entropy:
  Definition: Consider an instantaneously decodable code which maps the symbols from a source S, $s_1, s_2, \ldots, s_q$ with probabilities $P_1, P_2, \ldots, P_q$ into code word composed of symbols from an *r-ary* code alphabet. We have the following relationships.

$$H(S) \le L \log r$$

$$H_r(S) \le L$$

- Compact code:
  Definition: Consider a uniquely decodable code which maps the symbols from a source S into code word composed of symbols from an *r-ary* code alphabet. This code will be called *compact* (for the source S) if its average length is less than or equal to the average length of all other uniquely decodable codes for the same source and the same code alphabet.

Compact code

- Proof of the relationship:
  - Consider a zero-memory source S, with symbols $s_1,s_2,\ldots,s_q$ and symbol probabilities $P_1,P_2,\ldots,P_q$, respectively. Let a block code encode these symbols into a code alphabet of r symbols, and let the length of the word corresponding to s_i be l_i. Then the entropy of this zero-memory source is,
$$H(S) = -\sum_{i=1}^{q} P_i \log P_i$$
  - Let $Q_1,Q_2,\ldots,Q_q$ be any q numbers such that $Q_i \geq 0$ for all if and $\sum_{i=1}^{Q} Q_i = 1.$
  - By the Jensen's inequality, we know that
$$\sum_{i=1}^{q} P_i \log \frac{1}{P_i} \leq \sum_{i=1}^{q} P_i \log \frac{1}{Q_i}$$
with equality if and only if $P_i=Q_i$ for all i. Hence,
$$H(S) \leq -\sum_{i=1}^{q} P_i \log Q_i \cdots (1)$$

Compact code

- Equation is valid for any set of nonnegative numbers Qi which sum to 1. We may choose,

$$Q_i = \frac{r^{-li}}{\sum_{i=1}^{q} r^{-lj}}$$

- We obtain,

$$H(S) \le -\sum_{i=1}^{q} P_i(\log r^{-li}) + \underbrace{\underbrace{\sum_{i=1}^{q} P_i}_{=1}(\log \underbrace{\sum_{j=1}^{q} r^{-lj}}_{\le 1})}_{\le 0} \cdots (2)$$

$$\le \log r \sum_{i=1}^{q} P_i l_i = \log rL$$

$$\frac{H(S)}{\log r} \le L, or\ H_r(S) \le L$$

Compact code

- A method of encoding for special source.
  Considering eqns. (1)(2), a condition for equality in the last inequality is,

$$\sum_{j=1}^{q} r^{-l_j} = 1$$

Then we see that a necessary and sufficient condition for equlality is,

$$\begin{aligned} P_i &= Q_i \\ &= \frac{r^{-l_i}}{\sum_{j=1}^{q} r^{-l_j}} \\ &= r^{-l_i} \quad \textit{for all } i. \end{aligned}$$

or

$$\log_r \frac{1}{P_i} = l_i \quad \textit{for all } i \cdots (4.9b)$$

Compact code

- We may say that, for an instantaneous code and a zero-memory source, L must be greater than or equal to $H_r(S)$. Furthermore, L can achieve this lower bound if and only if we can choose the word lengths l_i equal to $\log_r (1/P_i)$ *for all* i. For the equality, therefore, $\log_r (1/P_i)$ must be an integer for each i.

- In other words, for the equality the symbol probabilities P_i must all be of the form $(1/r)^{ai}$, where a_i is an integer.
  Note that if these conditions are met, we have derived the word lengths of a compact code. We simply choose l_i equal to a_i.

Compact code

Source symbol	Symbol prob.	code
S_1	1/2	0
S_2	1/4	10
S_3	1/8	110
S_4	1/8	111

$$P_i = \left(\frac{1}{2}\right)^{l_i}$$

$$L = \sum_{i=1}^{4} P_i l_i = 1\frac{3}{4}$$

$$H = \sum_{i=1}^{4} P_i \log \frac{1}{P_i} = 1\frac{3}{4}$$

Example: Compact code

Source symbol	Symbol prob.	code
S_1	1/4	00
S_2	1/4	01
S_3	1/4	10
S_4	1/4	11

Source symbol	Symbol prob.	code
S_1	1/2	0
S_2	1/4	10
S_3	1/8	110
S_4	1/8	111

Example: Compact code

Source symbol	Symbol prob.	code
S_1	1/3	0
S_2	1/3	1
S_3	1/9	20
S_4	1/9	21
S_5	1/27	220
S_6	1/27	221
S_7	1/27	222

Shannon's first theorem

- We now turn to zero-memory source with arbitrary symbol probabilities.
- Equation (4-9b) tells us that if $\log_r (1/P_i)$ is an integer, we should choose the word length l_i equal to this integer. If $\log_r (1/P_i)$ is not an integer, it might seem reasonable that a compact code could be found by selecting l_i as the first integer greater than this value. This tempting conjecture is, in fact , not valid, but we shall find that selecting l_i in this manner can lead to some important results.

$$\log_r \frac{1}{P_i} \le l_i \le \log_r \frac{1}{P_i} + 1 \cdots (4-10)$$

- First, we check to see that the word lengths satisfy the Kraft inequality.

$$\frac{1}{P_i} \le r^{l_i} \ or\ P_i \le r^{l_i} \cdots (4-11)$$

Summing (4-11) over all i, we obtain,

$$1 \ge \sum_{i=1}^{q} r^{-l_i}$$

Shannon's first theorem

- If we multiply (4-10) by *Pi* and sum over all *i*,

$$H_r(S) \le L < H_r(S) + 1. \cdots (4-12)$$

- In this way, if we construct the code in the way of (4-10), we can have the lower and upper bounds of *L*. This is valid for any zero-memory source, we may apply it to the nth extension of our original source *S*.

$$H_r(S^n) \le L_n < H_r(S^n) + 1. \cdots (4-13)$$

L_n represents the average length of the code words corresponding to symbols from the nth extension of the source S. If λ_i is the length of the code word corresponding to symbol σ_i and, $P(\sigma_i)$ is the probability of σ_i, then

$$L_n = \sum_{i=1}^{q^n} P(\sigma_i)\lambda_i \cdots (4-14)$$

L_n/n is the average number of code symbols used per single symbol from S.

$$\boxed{H_r(S) \le \frac{L}{n} < H_r(S\ ) + \frac{1}{n} \cdots (4-15a)}$$

Shannon's first theorem

- It is possible to make L_n/n as close to $Hr(S)$ as we wish by coding the nth extension of S rather than S:

$$\boxed{\lim_{n\to\infty}\frac{L_n}{n} = H_r(S)\cdots(4-15b)}$$

Equation (4-15a) is known as Shannon's first theorem or the noiseless coding theorem. The price we pay for decreasing L_n/n is the increased coding complexity caused by the large number (q^n) of source symbols.

Shannon's first theorem for Markov source

- We define the first-order Markov source S, with source symbols $s_1, s_2, \ldots, s_q$ and conditional symbols probabilities $P(s_i / s_j)$. We also define S_n, the nth extension of S, with symbols $\sigma_1, \sigma_2, \ldots, \sigma_{q^n}$, and conditional symbols probabilities $P(\sigma_i / \sigma_j)$. We refer to the first-order (unconditional) symbol probabilities of S and S_n as P_i and $P(\sigma_i / \sigma_j)$, respectively.

- The process of encoding the symbols $s_1, s_2, \ldots, s_q$ into an instantaneous block code is identical for he source S and its adjoint source $\bar{S}$. If the length of the code word corresponding to s_i is l_i, the average length of the code is,

$$L = \sum_{i=1}^{q} P_i l_i$$

Shannon's first theorem for Markov source

- The average length is identical for S and $\bar{S}$ since P_i, the first-order symbol probability of s_i, is the same for both these sources.
  $\bar{S}$ is a zero-memory source, and we have,
  $$H_r(\bar{S}) \le L$$
  This inequality may be augmented to read,
  $$H_r(S) \le H_r(\bar{S}) \le L$$
  and,
  $$H_r(S^n) \le H_r(\overline{S^n}) \le L_n$$
  If we now select the l_i according to (4-10), we may bound L above and below (4-12),
  $$H_r(\bar{S}) \le L < H_r(\bar{S}) + 1$$
  for the extended source,
  $$H_r(\overline{S^n}) \le L_n < H_r(\overline{S^n}) + 1$$
  using (2-41) and dividing by n,
  $$H_r(S) + \frac{H_r(\bar{S}) - H_r(S)}{n} \le \frac{L_n}{n} < H_r(S) + \frac{\left[H_r(\bar{S}) - H_r(S)\right] + 1}{n}$$

Coding without extensions

- Shannon's theorem shows the bound above and below considering its extension. The theorem doesn't tell us what value of L (or L_n/n) we shall obtain. It doesn't guarantee that choosing the word lengths according to (4-10) will give us the smallest possible value of L (or L_n/n) it is possible to obtain for that fixed n.

Source symbol	P_i	Log $1/P_i$	l_i	Code A	Code B
S_1	2/3	0.58	1	0	0
S_2	2/9	2.17	3	100	10
S_3	1/9	3.17	4	1010	11

$$L_A = \frac{2}{3}\times 1 + \frac{2}{9}\times 3 + \frac{1}{9}\times 4 = 1.78 \mathit{binits} / \mathit{source\,symbol} \qquad H_A(S) = \sum_{i=1}^{3} P_i \log\frac{1}{P_i} = 1.22 \mathit{bits} / \mathit{source\,symbol}$$

This satisfies, $$H_A(S) \le L_A \le H_A(S) + 1$$

However, code B gives shorter L. $$L_B = \frac{2}{3}\times 1 + \frac{2}{9}\times 2 + \frac{1}{9}\times 2 = 1.33 \mathit{binits} / \mathit{source\,symbol}$$

Binary Compact Codes – Huffman Codes

- A compact code for a source S is a code which has the smallest average length possible if we encode the symbols from S one at a time. We develop a method of constructing compact codes for the case of a binary code alphabet.

- Consider the source S with symbols $s_1, s_2, \ldots, s_q$ and symbol probabilities $P_1, P_2, \ldots, P_q$. Let the symbols be ordered so that $P_1 \geq P_2 \cdots \geq P_q$. By regarding the last two symbols of S as combined into one symbol, we obtain a new source from S containing only $q\text{-}1$ symbols.
We refer to this new source as a *reduction* of S.

- The symbols of this reduction of S may be reordered, and again we may combine the two last least probable symbols to form a reduction of this reduction of S. By proceeding in this manner, we construct a sequence of sources, each containing one fewer symbol than the previous one, until we arrive at a source with only two symbols.

Huffman codes

Original Source					Reduced Source
Symbols	Prob.	$S1$	$S2$	$S3$	$S4$
s_1	0.4	0.4	0.4	0.4	0.6
s_2	0.3	0.3	0.3	0.3	0.4
s_3	0.1	0.1	0.2	0.3	
s_4	0.1	0.1	0.1		
s_5	0.06	0.1			
s_6	0.04				

- Construction of a sequence of reduced sources is the first step in the construction of a compact instantaneous code for the original source S.
- The second step is merely the recognition that a binary compact instantaneous code for the last reduced source (a source with only two symbols) is the trivial code with the two words 0 and 1.
- The final step is to construct a compact instantaneous code for the source immediately preceding the reduced source in the sequence of reduced sources.

Huffman codes

Huffman codes for two symbols		
Symbols	Prob.	Code
S_1	$3/4$	0
S_2	$1/4$	1

$$L = \sum_{i=1}^{5} P_i l_i = 1$$

$$H = \sum_{i=1}^{5} P_i \log \frac{1}{P_i} = 0.8113$$

Huffman codes

Synthesis of a compact code

symbols	Prob.	Code	S1	Code	S2	Code	S3	Code	4	Code
s_1	0.4	1	0.4	1	0.4	1	0.4	1	0.6	0
s_2	0.3	00	0.3	00	0.3	00	0.3	00	0.4	1
s_3	0.1	011	0.1	011	0.2	010	0.3	01		
s_4	0.1	0100	0.1	0100	0.1	011				
s_5	0.06	01010	0.1	0101						
s_6	0.04	01011								

- We assign to each symbol of S_{j-1} (s_{a0} and s_{a1}) the code word used by the corresponding symbol of S_j. The code words used by s_{a0} and s_{a1} are formed by adding a 0 and 1, respectively, to the code word used for s_a.

- There are another possibilities to decompose a reduced source in code S_3 and S_1.

Huffman codes

Synthesis of compact codes

Symbols	Prob.	Codes	$S1$	Codes	$S2$	Codes	$S3$	Codes	S4	Codes
s_1	0.4	1	0.4	1	0.4	1	0.4	1	0.6	0
s_2	0.3	00	0.3	00	0.3	00	0.3	00	0.4	1
s_3	0.1	0100	0.1	011	0.2	010	0.3	01		
s_4	0.1	0101	0.1	0100	0.1	011				
s_5	0.06	0110	0.1	0101						
s_6	0.04	0111								

- There are three choices in S_1. If we choose the fist one, we obtain a code with word lengths ,

  1, 2, 4, 4, 4, 4.

  If we choose the second or third, we obtain,

  1, 2, 3, 4, 5, 5.

$$L=\sum_{i=1}^{5} P_i l_i = 1.875 \qquad H=\sum_{i=1}^{5} P_i \log\frac{1}{P_i} = 1.8402$$

Huffman codes

$$L = 1(0.4) + 2(0.3) + 4(0.1) + 4(0.1) + 4(0.06) + (0.04) = 2.2 binits / symbol$$

$$L = 1(0.4) + 2(0.3) + 3(0.1) + 4(0.1) + 5(0.06) + 5(0.04) = 2.2 binits / symbol$$

$$H = \sum_{i=1}^{6} P_i \log \frac{1}{P_i} = 2.1435$$

- Two codes have the same average code lengths. These are shortest average length codes that can construct.

Synthesis of compact code

Symbols	Prob.	code	$S1$	code
s_1	0.5	0	0.5	0
s_2	0.25	10	0.25	10
s_3	0.125	110	0.125	110
s_4	0.100	1110	0.125	111
s_5	0.025	1111		

Compact code

$$L = \sum_{i=1}^{5} P_i l_i = 1 \qquad H = \sum_{i=1}^{5} P_i \log \frac{1}{P_i} = 0.8113$$

Proof of Huffman codes

- Assume that we have found a compact code C_j for some reduction, say S_j, of an original source S. Let the average length of this code be L_j.
- One of the symbols of S_j, say s_a, is formed from the two least probable symbols of the preceding reduction S_{j-1}. Let these two symbols be s_{a0} and s_{a1}, and let their probabilities be P_{a0} and P_{a1}, respectively.
- The probability of s_a is then $P_a = P_{a0} + P_{a1}$. Let the code for S_{j-1} formed according to rule (4-24) be called C_{j-1}, and let its average length be L_{j-1}.
- L_{j-1} is easily related to L_j since the words of C_j and C_{j-1} are identical except that the (two) words for s_{a0} and s_{a1} are one binit longer than the (one) word for s_a. Thus we know that

$$L_{j-1} = L_j + P_{\alpha 0} + P_{\alpha 1} \cdots (4.25)$$

- What we want to show is if C_j is compact, then C_{j-1} must also be compact. In other words, if L_j is the smallest possible average length of an instantaneous code for S_j, then L_{j-1} is the smallest possible average length for S_{j-1}.

Proof of Huffman codes

$$L_{j-1} = \sum_{i=1}^{k-1} P_i l_i + P_{\alpha 0} l_{\alpha 0} + P_{\alpha 1} l_{\alpha 1}$$

$$= \sum_{i=1}^{k-1} P_i l_i + P_{\alpha 0}(l_\alpha + 1) + P_{\alpha 1}(l_\alpha + 1)$$

$$= \sum_{i=1}^{k} P_i l_i + P_{\alpha 0} + P_{\alpha 1}$$

$$= L_j + P_{\alpha 0} + P_{\alpha 1}$$

where,

$$L_j = \sum_{i=1}^{k} P_i l_i, \quad P_\alpha = P_{\alpha 0} + P_{\alpha 1}$$

Proof of Huffman codes

- A proof by demonstrating that assuming the contrary leads to a contradiction.
- Assume that we have found a compact code for S_{j-1} with average length $\tilde{L}_{j-1} < L_{j-1}$. Let the words of the code be $\tilde{X}_1, \tilde{X}_2, \ldots, \tilde{X}_{a1}$, with lengths $\tilde{l}_1, \tilde{l}_2, \ldots, \tilde{l}_{a1}$, respectively. We assume that the subscripts are ordered in order of decreasing symbol probabilities so that,

$$\tilde{l}_1 \le \tilde{l}_2 \le \cdots \le \tilde{l}_{a1}$$

- One of the words of this code (call it $\tilde{X}_{a0}$) must be identical with $\tilde{X}_{a1}$ except in its last digit. If this were not true, we could drop the last digit from $\tilde{X}_{a1}$ and decrease the average length of the code without destroying its instantaneous property.
- Finally, we form $\tilde{C}_j$, a code for S_j, by combining $\tilde{X}_{a0}$ and $\tilde{X}_{a1}$ and dropping their last binit while leaving all other words unchanged. This gives us an instantaneous code for S_j with average length , related by

$$\tilde{L}_{j-1} = \tilde{L}_j + P_{\alpha 0} + P_{\alpha 1}$$

Proof of Huffman codes

- If we compare the last equation to (4-25), we see that our assumption

$$\tilde{L}_{j-1} < L_{j-1}$$

implies that we may construct a code with average length

$$\tilde{L}_j < L_j$$

This is the contradiction we seek since the code with average length L_j is compact.

- Two properties of Huffman codes.
  - If the probabilities of the symbols of a source are ordered so that

$$P_1 \geq P_2 \geq \cdots \geq P_q$$

, the lengths of the words assigned to these symbols will be ordered so that,

$$l_1 \leq l_2 \leq \cdots \leq l_q$$

  - The lengths of the last two words (in order of decreasing probability) of a compact code are identical:

$$l_q = l_{q-1}$$

If there are several symbols with probability P_q, we may assign their subscripts so that the words assigned to the last two symbols differ only in their last digit.

r-ary compact codes

- We would like the last source in the sequence to have exactly *r* symbols. The last source will have r symbols if and only if the original source has *r+a(r-1)* symbols, where *a* is an integer. Therefore, if the original source doesn't have *r+a(r-1)* symbols, we add "dummy symbols" with probability 0 to the source until this number is reached.

Synthesis of compact codes

Symbols	Prob.	Codes	$S1$	Codes	$S2$	Codes	$S3$	Codes
s_1	0.22	2	0.22	2	0.23	1	0.40	0
s_2	0.15	3	0.15	3	0.22	2	0.23	1
s_3	0.12	00	0.12	00	0.15	3	0.22	2
s_4	0.10	01	0.10	01	0.12	00	0.15	3
s_5	0.10	02	0.10	02	0.10	01		
s_6	0.08	03	0.08	03	0.10	02		
s_7	0.06	11	0.07	10	0.08	03		
s_8	0.05	12	0.06	11				
s_9	0.05	13	0.05	12				
s_{10}	0.04	100	0.05	13				
s_{11}	0.03	101						
(s_{12})	0.00	102						
(s_{13})	0.00	103						

Dummy symbols

Code efficiency and redundancy

- Shannon's first theorem shows that there exists a common measure for any information source. The value of a symbol from an information source S may be measured in terms of an equivalent number of binary digits needed to represent one symbol from that source.
  Let the average length of a uniquely decodable r-ary code for the source S be L. L cannot be less than Hr(s). Accordingly, we define the efficiency of the code, by

$$\eta = \frac{H_r(S)}{L}.$$

  It is also possible to define the redundancy of a code.

$$\begin{aligned} \text{Redundancy} &= 1 - \eta \\ &= \frac{L - H_r(S)}{L}. \end{aligned}$$

Example – *n*th extension

Huffman codes for two symbols		
Symbols	Prob.	Code
S_1	3/4	0
S_2	1/4	1

$$H(S) = \frac{1}{4}\log 4 + \frac{3}{4}\log\frac{4}{3} = 0.811 bit$$

The average length of this code is 1 binit, so the efficiency is,

$$\eta = 0.811.$$

To improve the efficiency, we might code S^2, the second extension of S:

Huffman codes for two symbols		
Symbols	Prob.	Code
S_1	9/16	0
S_2	3/16	10
S_3	3/16	110
S_4	1/16	111

$$\eta_2 = 0.985$$

Extending to higher order, $\eta_3 = 0.985, \quad \eta_4 = 0.991$

Example – *n*th extension

TABLE 4-5. COMPACT CODES FOR DIFFERENT CODE ALPHABETS

		Compact codes for r =											
$P(s_i)$	s_i	13	12	11	10	9	8	7	6	5	4	3	2
$\frac{1}{4}$	s_1	0	0	0	0	0	0	0	0	0	0	0	00
$\frac{1}{4}$	s_2	1	1	1	1	1	1	1	1	1	1	1	01
$\frac{1}{16}$	s_3	2	2	2	2	2	2	2	2	2	20	200	1000
$\frac{1}{16}$	s_4	3	3	3	3	3	3	3	3	30	21	201	1001
$\frac{1}{16}$	s_5	4	4	4	4	4	4	4	4	31	22	202	1010
$\frac{1}{16}$	s_6	5	5	5	5	5	5	5	50	32	23	210	1011
$\frac{1}{16}$	s_7	6	6	6	6	6	6	60	51	33	30	211	1100
$\frac{1}{16}$	s_8	7	7	7	7	7	70	61	52	34	31	212	1101
$\frac{1}{16}$	s_9	8	8	8	8	80	71	62	53	40	32	220	1110
$\frac{1}{64}$	s_{10}	9	9	9	90	81	72	63	54	41	330	221	111100
$\frac{1}{64}$	s_{11}	A	A	A0	91	82	73	64	550	42	331	2220	111101
$\frac{1}{64}$	s_{12}	B	B0	A1	92	83	74	65	551	43	332	2221	111110
$\frac{1}{64}$	s_{13}	C	B1	A2	93	84	75	66	552	44	333	2222	111111
Average length L . . .		1	$\frac{33}{32}$	$\frac{67}{64}$	$\frac{17}{16}$	$\frac{9}{8}$	$\frac{19}{16}$	$\frac{5}{4}$	$\frac{87}{64}$	$\frac{23}{16}$	$\frac{25}{16}$	$\frac{131}{64}$	$\frac{25}{8}$

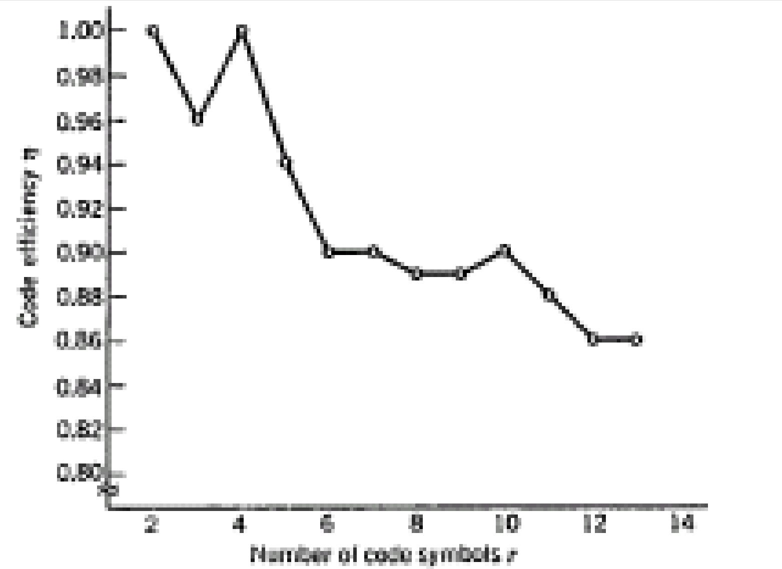
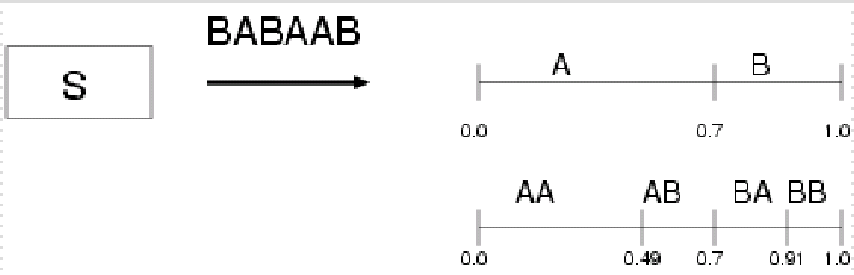
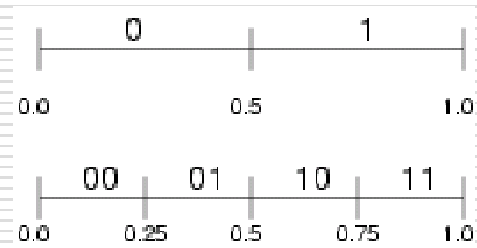


FIGURE 4-7. Code efficiency versus number of code symbols.

Compact codes: Elias codes





2-ary coding

011 is a point of region [0.375,0.50] . An initial symbol is A.

0110 is a point of region [0.375,0.4375]. The source symbols are AAB.

- Elias code:
  Elias codes is non-block compact codes in contrast to the Huffman codes, which are the block codes. This is also called arithmetic codes.

- Elias code assign a sequence of source symbols to a fractional number, which is obtained by dividing a number line according to the symbol probabilities.

Elias code

- In Huffman codes it is necessary to consider extension of codes in order to improve code efficiency. If the block size is large, it becomes difficult.
  Also in Huffman codes code length should be an integer number.

- Elias code assigns a sequence of source symbols to one code. It is not necessary to calculate all of probabilities of nth extension of symbols and we can decode the codes iteratively.

Elias code

- Procedure:
  - Suppose we have binary codes s_0 and s_1 with probabilities P_0 and P_1.
  - Divide a region of number line [0,1) according to P_0:P_1 and make a region A_0 and A_1. A_0 corresponds $[0,P_0)$, A_1 corresponds $[P_0,1)$.
  - If a first source symbol, S_0 is s_0 then choose a region A_0, else choose a region A_1.
  - If $S_0=s_0$ and a region A_0 is selected, divide a region A_0 according to P_0:P_1 and obtain a region A_{00} and A_{01}.
    Then a next code $S_1=s_0$, then choose a region A_{00}, else A_{01}.
  - Iterate this procedure until the end of the source symbol sequence and represent a chosen region with a fractional number, which is lower value of the region.

Elias code

- Average code length: The size of the region for symbol sequence SN becomes,

$$P_0^{N_0} \times P_1^{N_1}$$

  where letting number of s_0, s_1 be N_0, N_1, respectively.
- The necessary resolution to represent a point in this region with binary fraction number is,

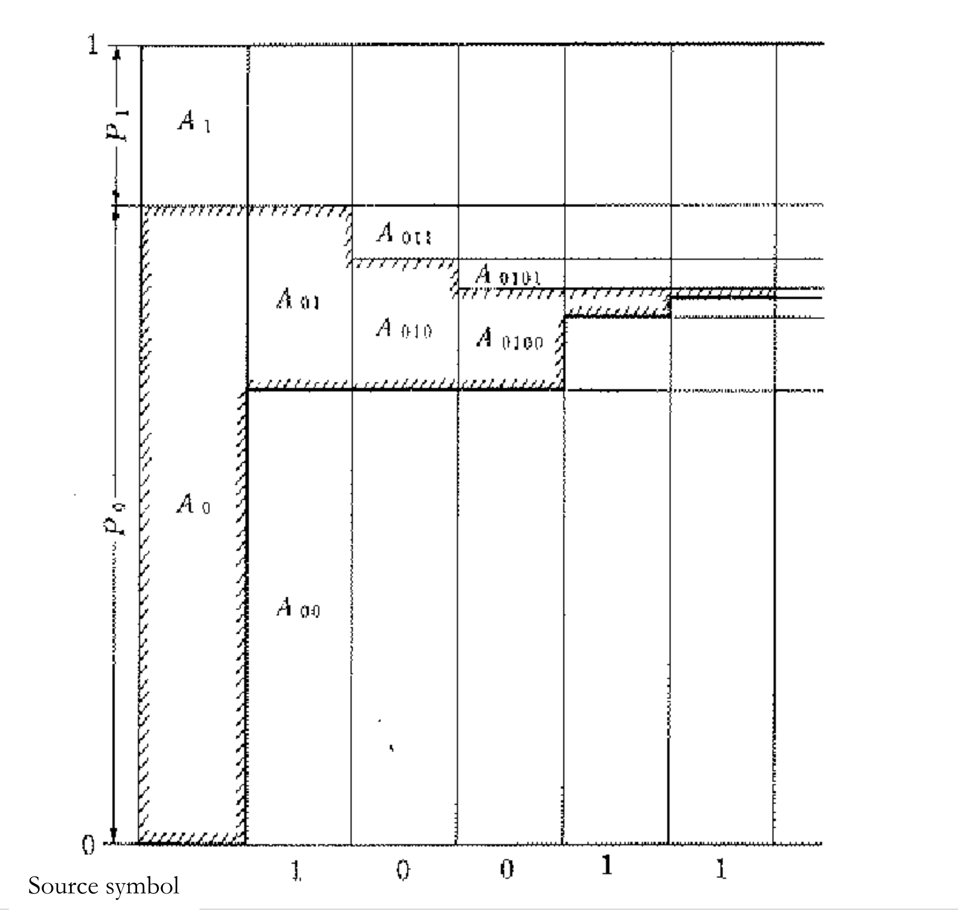
$$-N_0 \log_2 P_0 - N_1 \log_2 P_1$$

- If we take longer source symbol length N,

$$P_0^{N_0} = \frac{N_0}{N}, \quad P_1^{N_1} = \frac{N_1}{N}$$

  in this way the average code length approaches to the Entropy according to the length N.

Elias code



This figure depicts the process where the source symbol sequence 010011.. is encoded by the Elias code.
First a region [0,1) is divided into A0 A1 according to P0:P1.
A0 is chosen since a first symbol is 0.
In this way the subregion is divided and chosen.

L-R arithmetic codes

- Problems of Elias code:
  - Multiplication by a probability per coding one source symbol is necessary. Required precision for calculation increases according to N.
  - Coding cannot be started until receiving the last symbol.

- L-R arithmetic code: One approach to solve the problem for binary code.
  - Approximate an inferior symbol probability by 2^{-Q}.
  - Assign a value U of the region [U,V) to the symbol sequence. Prevent bit-reverse propagation by carry introducing bit-stuffing.

- Average code length:
  An average code length of L-R code is given by,

$$L = P_1 \times \log_2 2^{-Q} - P_0 \times \log_2 (1 - 2^{-Q})$$

  Coding efficiency becomes 1 if an inferior symbol probability is 2^{-Q}.

L-R arithmetic code

- Coding algoritm:
  - Initialization:
    Prepare a register C and a register A with V bits.

    $$C \leftarrow 000...0$$

    $$A \leftarrow 111...1$$

    C is an initial code and A is an initial value of the region.

  - Coding of source symbol Xi.
    - Divide the register A into A0 and A1 according P0:P1 of the superior symbol "0" and the inferior symbol "1".

      $$A_1 \leftarrow A \times P_1 \quad (1)$$

      $$A_0 \leftarrow A \times P_0 \quad (2)$$

    $P_1 \approx 2^{-Q}$ (Q: integer, called SKEW) (1) is calculated by right shift and (2) can be calculated by A-A1=A0

L-R arithmetic code

- Code is,

  If $X_i = 0$ $\quad C$ is same as it was.

  If $X_i = 1$ $\quad C = C + A_0$

  update the region,

  If $X_i = 0$ $\quad A \leftarrow A_0$

  If $X_i = 1$ $\quad A \leftarrow A_1$

  ,where C represents the lower bound of the chosen region.

L-R arithmetic code

- Decoding algorithm:
  - Initialization:

    C $\leftarrow$ copied from the received codes.

    A $\leftarrow$ the initial value set by the coding algorithm

  - Decoding:
    - Every time we receive a code, divide the region A.

$$A_0 \leftarrow A \times P_0$$
$$A_1 \leftarrow A \times P_1$$

For registers,

If C-A0 is negative, keep C as it was and choose source symbol 0.

If C-A0 is non-negative, set C $\leftarrow$ C-A0, and choose source symbol 1.

Next update A,

$$X_i = 0 \qquad A \leftarrow A_0$$
$$X_i = 1 \qquad A \leftarrow A_1$$

Another advantage of L-R arithmetic code

- We can change a inferior probability, SKEW, according to change of a symbol probability. If we use the same SKEW in decoding, we can decode in the same way.

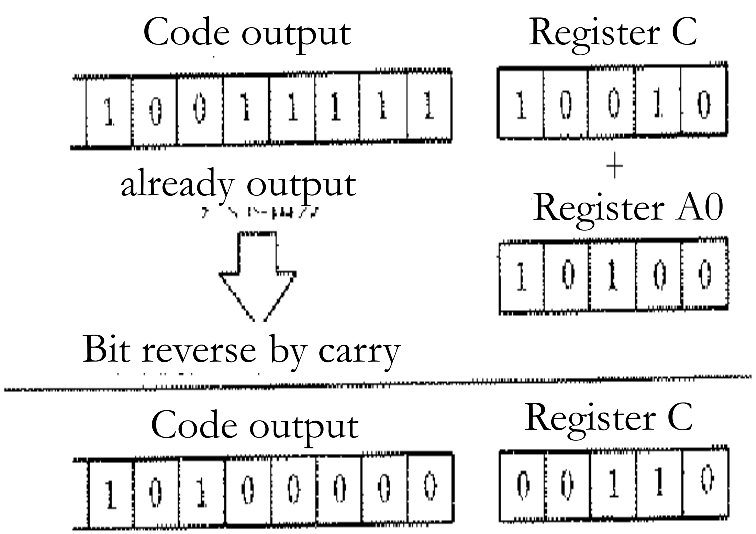
Coding example by L-R code

Sym.	A	A_0	A_1	Code Output	C
0	1111	1100	0011		0000
1	1100	1001	0011		1001
ren.	0011	Shift 2 bit		10	01
0	1100	1001	0011	10	0100
0	1001	0111	0010	10	0100
ren.	0111	Shift 1 bit		100	100
1	1110	1011	0011	101	0011
ren.	0011	Shift 2 bit		10100	11
1	1100	1001	0011	10101	0101
ren.	0011	Shift 2 bit		1010101	01
0	1100	1001	0011	1010101	0100
0	1001	0111	0010	1010101	0100
ren.	0111	Shift 1 bit		10101010	100
0	1110	1011	0011	10101010	1000
1	1011	1001	0010	10101011	0001
		Code string = 101010110001			

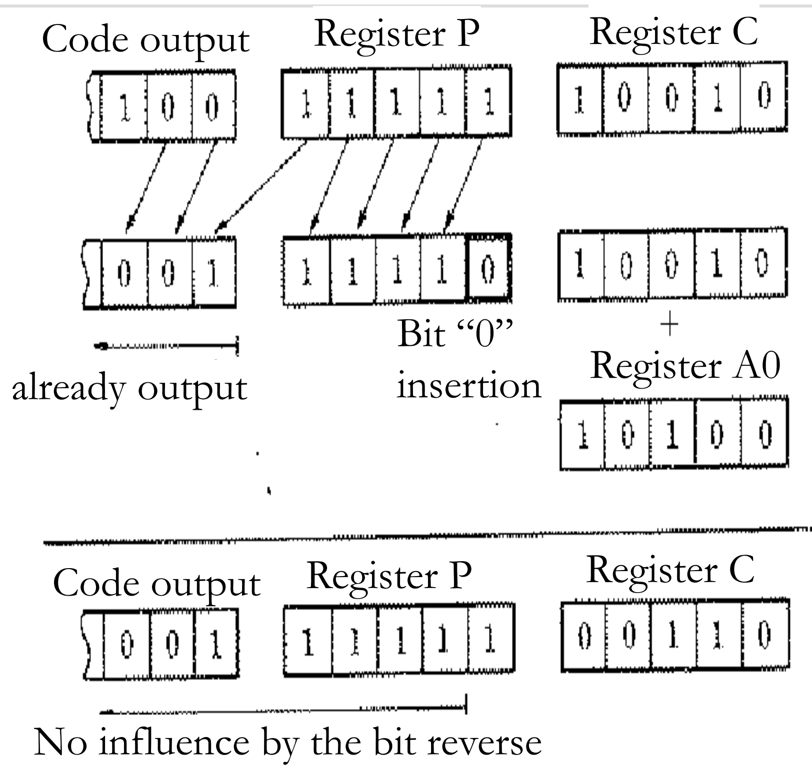
Decoding example of L-R code

A	A_0	A_1	C	Code String	Sym.
1111	1100	0011	1010	10110001	0
1100	1001	0011	0001	10110001	1
0011	Shift 2 bit		0110	110001	ren.
1100	1001	0011	0110	110001	0
1001	0111	0010	0110	110001	0
0111	Shift 1 bit		1101	10001	ren.
1110	1011	0011	0010	10001	1
0011	Shift 2 bit		1010	001	ren.
1100	1001	0011	0001	001	1
0011	Shift 2 bit		0100	1	ren.
1100	1001	0011	0100	1	0
1001	0111	0010	0100	1	0
0111	Shift 1 bit		1001		ren.
1110	1011	0011	1001		0
1011	1001	0010	0000		1
	Decoded symbol string=0100110001				

Bit-stuffing- L-R code

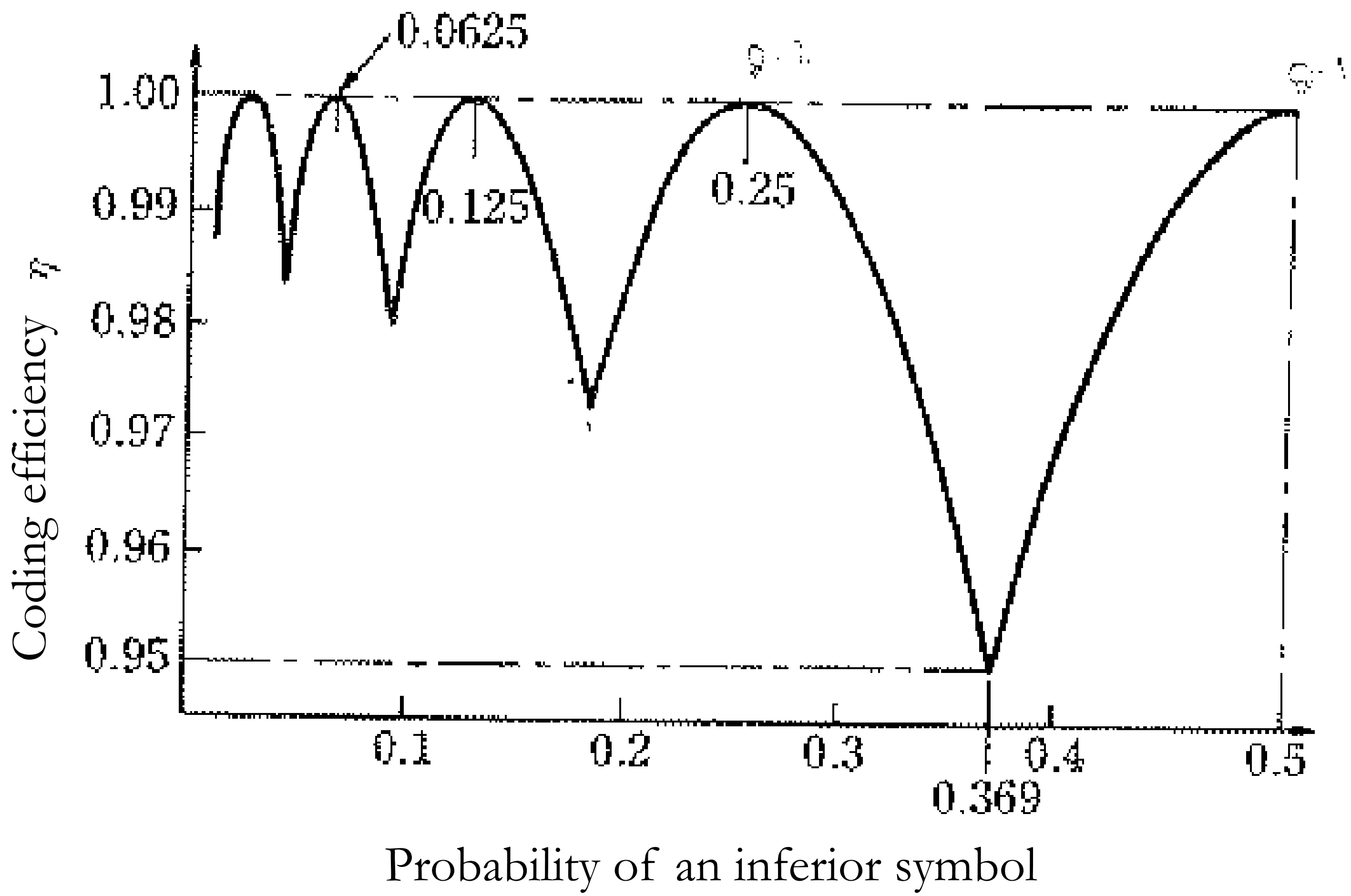


(a) Without bit-stuffing



(a) With bit-stuffing

Coding efficiency of L-R code



Universal code

- What is universal code?
  - Coding which can compress source symbols belong to a fixed class, optimally or very efficiently.
  - Coding algorithm independent of a prior probabilities of source symbols. Or coding algorithm for source symbols which have varying probabilities.
- Three coding algorithms:
  - Adaptive Huffman code
  - Context Modeling
  - Dictionary code

Adaptive Huffman code

- Adaptive Huffman code (1)
  - Algorithm:
    Every time when we receive N source symbols (one block), update a probability table of source symbols and re-synthesis Huffman codes. Then send them to the decoder.

  - Problems:
    According to the size of a block the size of the probability table seems relatively small, however, we cannot send a code until N source symbols. It is very inefficient to re-synthesis Huffman code every N symbols.

Adaptive Huffman code

- Adaptive Huffman code (2)
  - Algorithm:
    - Code a source symbols and send the code based on the Huffman codes designed by a prior symbol probabilities.
    - Let probabilities of source symbols $a_0, a_1, \ldots, a_{M-1}$ at time N-1 be,

$$P_{N-1}(a_i) = \frac{n_{N-1}(a_i)}{N-1}$$

      If we let a source symbol at time N be a, the code for the symbol is synthesized by Huffman codes based on the symbol probability,

$$P_N(a) = \frac{n_N(a)}{N} = \frac{(N-1)p_{N-1}(a)+1}{N}$$

$$P_N(a_i) = \frac{n_N(a_i)}{N} = \frac{(N-1)p_{n-1}(a_i)}{N} \; if \; a_i \neq a$$

    - This algorithm doesn't need to send a probability table of source symbols since a decoder can update the probability table simultaneously.
  - Problems:
    In worst case an update of Huffman codes will be necessary for each symbol.
    Higher resolution is necessary according to the size N.

Adaptive Huffman code

- Adaptive Huffman code (3)
  - Algorithm:
    - Update the probability table only when the tree of Huffman codes changed. The timing of the table update is calculated based not on the true symbol probabilities but on the following approximated probabilities.

$$P(a_i) = \frac{w_i}{\sum_{k=0}^{M-1} w_k}$$

      Normalization in this equation will not be applied in reality.
    - Initialization:
      Synthesize Huffman codes and their tree according to the a prior source symbol probabilities.
    - Assign w_i to the symbol according to the a prior probability.
    - When we increment one count to w_i when receiving the source symbol.
    - If there is a change in the Huffman code tree, re-synthesize the Huffman code tree until it satisfies a Huffman code property.
  - Huffman code property:
    This means that the structure of Huffman codes takes a form of ordered list by probabilities. This property is also called "*Sibling Property*".

Adaptive Huffman code

P_i						
s_1	0.5		0.5		0.5	0
s_2	0.3		0.3	10	0.5	1
s_3	0.1	110	0.2	11		
s_4	0.1	111				

W_i						
s_1	50		50		50	0
s_2	30		30	10	50	1
s_3	10	110	20	11		
s_4	10	111				

- Increment w_i when we receive $S_i=\{s_1,s_2,s_3,s_4\}$. If the sibling property doesn't hold, re-synthesize a partial Huffman code tree.

Adaptive Huffman codes

Symbol	S1	S2	S3	S4
Frequency	10	7	5	3

If we receive 10 times S4, how does Huffman tree change?

#S4 +1

Symbol	Freq.	Code	Freq.	Code	Freq.	Code
S1	10	1	10	1	16	0
S2	7	01	9	00	10	1
S3	5	000	7	01		
S4	4	001				

#S4 +2

Symbol	Freq.	Code	Freq.	Code	Freq.	Code
S1	10	1	10	1	17	0
S2	7	01	10	00	10	1
S3	5	000	7	01		
S4	5	001				

Adaptive Huffman codes

#S4 +3

Symbol	Freq.	Code	Freq.	Code	Freq.	Code
S1	10	00	11	1	17	0
S2	7	01	10	00	11	1
S4	6	10	7	01		
S3	5	11				

#S4 +4

Symbol	Freq.	Code	Freq.	Code	Freq.	Code
S1	10	00	12	1	17	0
S2	7	01	10	00	10	1
S4	7	10	7	01		
S3	5	11				

Adaptive Huffman codes

#S4 +5

Symbol	Freq.	Code	Freq.	Code	Freq.	Code
S1	10	00	12	1	18	0
S4	8	01	10	00	12	1
S2	7	10	8	01		
S3	5	11				

#S4 +6

Symbol	Freq.	Code	Freq.	Code	Freq.	Code
S1	10	00	12	1	19	0
S4	9	01	10	00	12	1
S2	7	10	9	01		
S3	5	11				

Adaptive Huffman codes

#S4 +7

Symbol	Freq.	Code	Freq.	Code	Freq.	Code
S1	10	00	12	1	20	0
S4	10	01	10	00	12	1
S2	7	10	10	01		
S3	5	11				

#S4 +8

Symbol	Freq.	Code	Freq.	Code	Freq.	Code
S4	11	00	12	1	21	0
S1	10	01	11	00	12	1
S2	7	10	10	01		
S3	5	11				

Adaptive Huffman codes

#S4 +9

Symbol	Freq.	Code	Freq.	Code	Freq.	Code
S4	12	1	12	1	22	0
S1	10	01	12	00	12	1
S2	7	000	10	01		
S3	5	001				

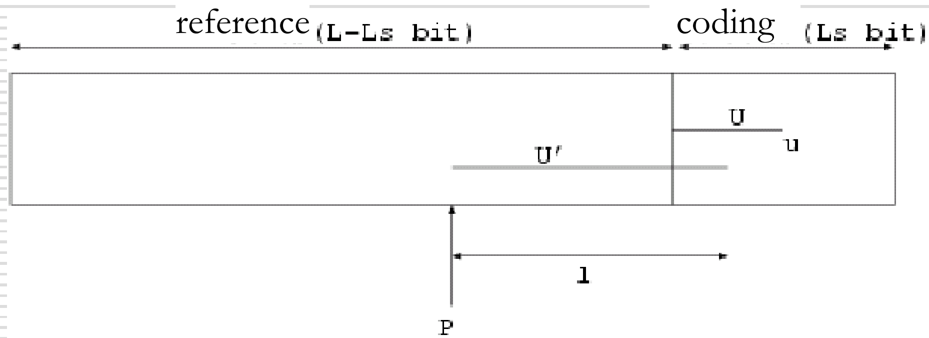
symbol	S4	S4	S4	S4	S4	S4	S4	S4	S4	S4
Code	001	001	001	001	10	10	01	01	00	1

Dictionary code

- Lempel-Ziv coding:
  Coding algorithm using a dictionary (code table) including source symbol sequences had been appeared.
  - Do not require a prior probability distributions of source symbols.
  - Non-block codes as well as the arithmetic code.
  - Compact codes as well as the arithmetic code.

- In this method, coding from a source symbol sequence to a code sequence is obtained in the following procedure.
  - 1. Retrieval: Look for a source symbol sequence in the dictionary.
  - 2. Coding: Code a source symbol sequence into a code sequence considering an order in the dictionary.
  - 3. Update: Update the dictionary in the decoding side.

LZ77 algorithm



- Set empty sequence (Λ) into the reference buffer.
- Set a source symbol sequence into coding buffer.
- Find a max symbol sub-sequence of the source symbol sequence in the reference buffer . Here let sub-sequence starting from left most side in the coding buffer be *U* and let sub-sequence with the same symbol sequence in the reference buffer be *U'*. Let *u* be a next symbol of *U*, and let *P* be a starting address pointer of *U'*. Let *l* be a length of *U*.
  Now we encode a source symbol sequence into *(P,l,u)*.
- Shift left by l+1 bit until there will be no source symbol.

LZ77 algorithm

source symbol sequence “abcabcdef”

source symbol	code
a	a
b	b
c	c
a	(-3,3,d)
b	
c	
d	
e	e
f	f

LZ77 algorithm

- Properties of LZ77 algorithm:
  - LZ77 approaches to the compact code if the buffer length L and Ls become large.
  - Sending u as a first mismatched symbol is inefficient.
    If l is very short, the code length is longer than the original source symbol length. In this case we just send the original source symbol sequence.
  - Use a fixed length of U.
    Also use the relative address from the left most side of coding buffer or use "Recency-Rank" meaning a number of different types of source symbols instead of the relative address.

LZ78 algorithm

- LZ78 algorithm:
  Universal coding based on "Incremental parsing".
  Let the source symbol sequence be,

  $$u = u_1, u_2, \ldots, u_T$$

  Incremental parsing

  $$u = U_0, U_1, \ldots, U_{t+1}$$

  is decomposition into a partial code sequence $U_m (0 \le m \le t+1)$

- The partial code sequence satisfies,
  - $U_0 = \Lambda$
  - $U_0, U_1 \ldots U_t$ are different each other except U_{t+1} .
  - If we take a last symbol u_m , $U_m (1 \le m \le t)$ equals to $U_s (0 \le s \le m-1)$.

Example

$$[0110011001011000001001110]$$

$$\begin{array}{ccccccccc} \Lambda & 0 & 1 & 10 & 01 & 100 & 101 & \\ U_0 & U_1 & U_2 & U_3 & U_4 & U_5 & U_6 & \cdots \end{array}$$

$$U_m = U_s u_m$$

- Each Um satisfies three properties and $U_m = U_s u_m$. We can code the source symbol sequence into (s, u_m) using $s(0 \le s \le m-1)$ and u_m.

Example

[01100110010110000100110]

Encoder

Time	In	Out(s, u_m)	Add to Table	Index
0	0	(-, 0)	0	0
1	1	(-, 1)	1	1
2	10	(1, 0)	10	2
3	01	(0, 1)	01	3
4	100	(2, 0)	100	4
5	101	(2, 1)	101	5
6	1000	(4, 0)	1000	6
7	010	(3, 0)	010	7
8	011	(3, 1)	011	8

Decoder

Time	In	Out	Add to Table	Index
0	0	0	0	0
1	1	1	1	1
2	(1, 0)	10	10	2
3	(0, 1)	01	01	3
4	(2, 0)	100	100	4
5	(2, 1)	101	101	5
6	(4, 0)	1000	1000	6
7	(3, 0)	010	010	7
8	(3, 1)	011	011	8

Example

Initial code table

Input String	Index
0	0
1	1

Encoder

Time	In	Out(*s*)	Add to Table	Index
0	01	0	01	2
1	11	1	11	3
2	10	1	10	4
3	00	0	00	5
4	011	2	011	6
5	100	4	100	7
6	010	2	010	8
7	011	2	011	9

Decoder

Time	In	Out	Add to Table	Index
0	0	0(? → 1)	? → 01	2
1	1	1 (? → 1)	? → 11	3
2	1	1 (? → 0)	? → 10	4
3	0	0(? → 0)	? → 00	5
4	2	01(? → 1)	? → 011	6
5	4	10(? → 0)	? → 100	7
6	2	01(? → 0)	? → 010	8
7	2	01(? →)	?	9

Move pointer to the position of next decomposed code -1.

Example

Initial code table

Input String	Index
0	0
a	1
b	2

Ternary Encoder

Time	Send	New Entry	Index
0	1 (for 0)	(a, 0)	3
1	0 (for 0)	(0, 0)	4
2	4 (for 00)	(0,0,b)	5
3	2 (for b)	(b,0)	6
4	3 (for (a,0))	(a,0,a)	7

Ternary Decoder

In	Out	Reconstructed Sequence	Add to Table
1	a	$(a)_{a,?}$	
0	0	$(a)_{a,0}\,(0)_{0,?}$	(a, 0) (as 3)
4	?	$(a)_{a,0}\,(0)_{0,?}$	?

LZ78

- Problems of LZ78
  - Coding by (s, u_m) is inefficient since we have to send u_m as it is. The solution is to send only (s). This method is used in "compress command" of Unix.
  - Incremental parsing stores all symbol sub-sequence in the dictionary and assign addresses.
  - This algorithm may cause memory overflow of the dictionary. In such a case we delete LRU (Least Recent Used) entry from the dictionary by "Self-organizing list".

Other code

- Run-length code:
  - abbbbbbbab: a(b,7)ab

Rate Distortion

- Coding with distortion:
  An average code length per one source symbol can be reduced if we allow coding distortion. Here, the distortion includes redundancy and errors which prevent uniquely decodability.

- Distortion measure:
  Let x be an source information symbol of L.
  Let y be a decoded output of the code.
  The distance between x and y is d(x,y), and called distortion measure. We evaluate the source coding efficiency by average distortion measure.

$$\bar{d} = \sum_{x}\sum_{y} d(x,y)\,p(x,y)$$

  where, p(x,y) are a joint probability distribution of a source symbol variable X and a coded symbol variable Y.

Rate distortion

- Mutual information:
  For a channel without any distortion we can easily know the source symbol x by knowing the decoded output y. The average amount of information is $H(X)$. If there is distortion, the average amount of information is,

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

  Therefore the lower bound of the average code length is the mutual information $I(X;Y)$.
  Distortion will be different while the mutual information is the same. For this case we try to find codes whose distortion $\bar{d}$ satisfies

$$\bar{d} \le D.$$

  Under this condition we try to find codes which minimizes the $I(X;Y)$

$$R(D) = \min_{\bar{d} \le D} I(X;Y)$$

  This $R(D)$ is called "*Rate-Distortion Function*" of the information source..

Rate distortion

- Definition:
  Under the condition that the average distortion is less than D, there exist codes whose average code length per one source symbol satisfy,

$$R(D) < L \le R(D) + \varepsilon$$

  for an integer ε.
  But there is no codes that has smaller average code length than R(D).

Rate distortion

- Derivation of RD function:
  The mutual information *I(X;Y)* is written in the following given $P_x(x)$ and conditional probabilities $P(y|x)$,

$$I(X;Y) = \sum_x P(x) \sum_y P(y|x) \frac{P(y|x)}{P(y)}$$

  We also know *P(y)* and the conditional probabilities $P(y|x)$,

$$P(y) = \sum_x P(x) P(y|x)$$

  Next, $\bar{d} \le D$ is written by,

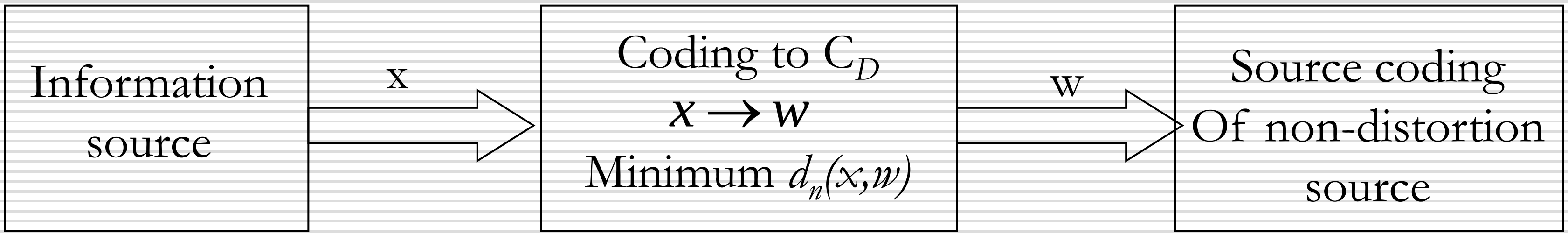
$$\bar{d} = \sum_x P(x) \sum_y P(y|x) d(x,y) \le D$$

  And probability constraints requests,

$$P(y|x) \ge 0 \qquad \sum_y P(y|x) = 1$$

  What we need is to minimize *I(X;Y)* under the above three constraints by the Lagrangean method.

Rate distortion



- Source coding with distortion:

  Suppose we choose a symbol sequence $\vec{x}_i = \vec{x}_{i1}, \vec{x}_{i2}, \ldots \vec{x}_{in} (i = 1, 2, \ldots k^n)$ of length n from an information source S with k symbols. Now we choose m codes that gives minimum average distortion.

$$C_D = \{\vec{W}_j = w_{j1}, w_{j2}, \ldots, w_{jn} (j = 1, 2, \ldots, m)\}$$

,here the average distortion is given by,

$$\overline{d_n} = \sum_{i=1}^{k^n} d_n(\vec{x}_i, \vec{w}_j(i)) \cdot p(\vec{x}_i, \vec{w}_j(i))$$

is $\vec{w}_{j(i)}$ minimizing $\{d(\vec{x}_i, \vec{w}_j); j = 1, 2, \ldots, m\}$, that is,

$$j(i) = \arg\min_j \ d_n(\vec{x}_i, \vec{w}_j) = \arg\min_j \sum_{k=1}^{n} d_n(x_i, w_j)$$

Rate distortion

- Then apply distortion-less source coding. This method provides average distortion $\bar{d}/n$ per each source symbol.

- Decoding:
  Decoding can be obtained by finding $\vec{x}_i$ that minimizes the distortion to code word $\vec{w}_j$.

  - Maximum likelihood decoding:
    Find $\vec{x}_m$ that satisfies,
    $$p(\vec{W}_j, \vec{x}_m) > p(\vec{W}_j, \vec{x}_{m'})$$
    for all m' except m.

Rate distortion

- Maximum a posteriori probability decoding:
  Find $\vec{x}_m$, which maximizes,

$$p(\vec{x}_m|\vec{w}_j) = \frac{p(\vec{x}_m)\cdot p(\vec{w}_j|\vec{x}_m)}{p(\vec{w}_j)}$$

  However, a prior probability $P(\vec{x}_m)$ needs to be given. This method is equivalent to a method maximizes the mutual information.

$$I(\vec{w}_j;\vec{x}_m) = E\left[\log \frac{p(\vec{w}_j|\vec{x}_m)}{p(\vec{w}_j)}\right]$$

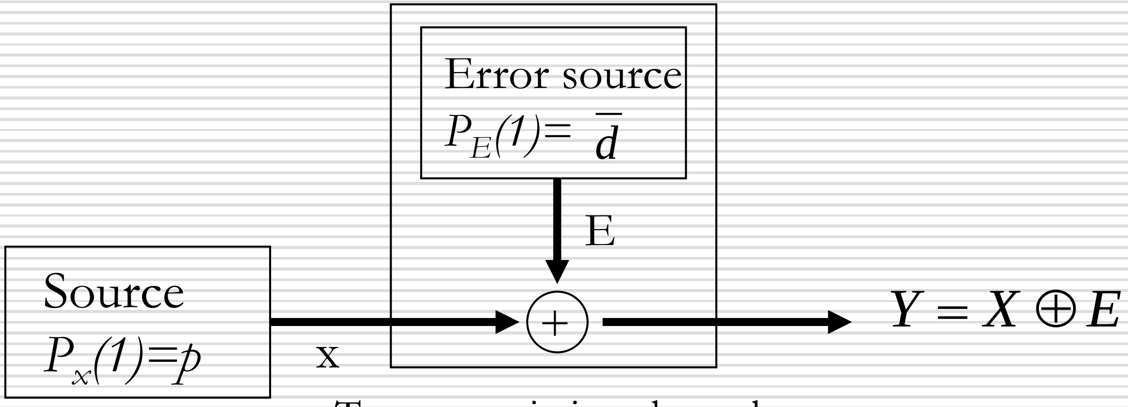
Binary source

- Suppose we have a binary information source of {0,1} with probabilities of p, 1-p and let a bit error rate be distortion measure.

$$d(x,y) = \begin{cases} 0 & x = y \\ 1 & x \neq y \end{cases}$$

- This source coding can be thought as a test transmission channel problem where the following mutual information is minimized under distortion $\bar{d}$,

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



Test transmission channel

Rate distortion

- Y can be thought as a symbol of which an error symbol added to a source symbol x is with probability $\bar{d}$. Here, since the addition is "XOR", $Y = X \oplus E$ is equivalent to $X = Y \oplus E$, then,

$$H(X|Y) = H(Y \oplus E|Y) = H(E|Y)$$

Furthermore, let $\tilde{H}(p)$ be a zero memory binary source,

$$\tilde{H}(p) = -p \log p - (1-p) \log(1-p).$$

If the error source is zero-memory source, $H(E) = \tilde{H}(\bar{d})$ holds, and even if the error source has a memory, $H(E) \le \tilde{H}(\bar{d})$ holds.

$$H(E|Y) \le H(E) \le \tilde{H}(\bar{d})$$

Therefore,

$$H(X|Y) \le \tilde{H}(\bar{d})$$

Rate distortion

If $0 \le D \le 0.5$, the Entropy function says,

$$\bar{d} \le D \Rightarrow \tilde{H}(\bar{d}) \le \tilde{H}(D)$$

then,

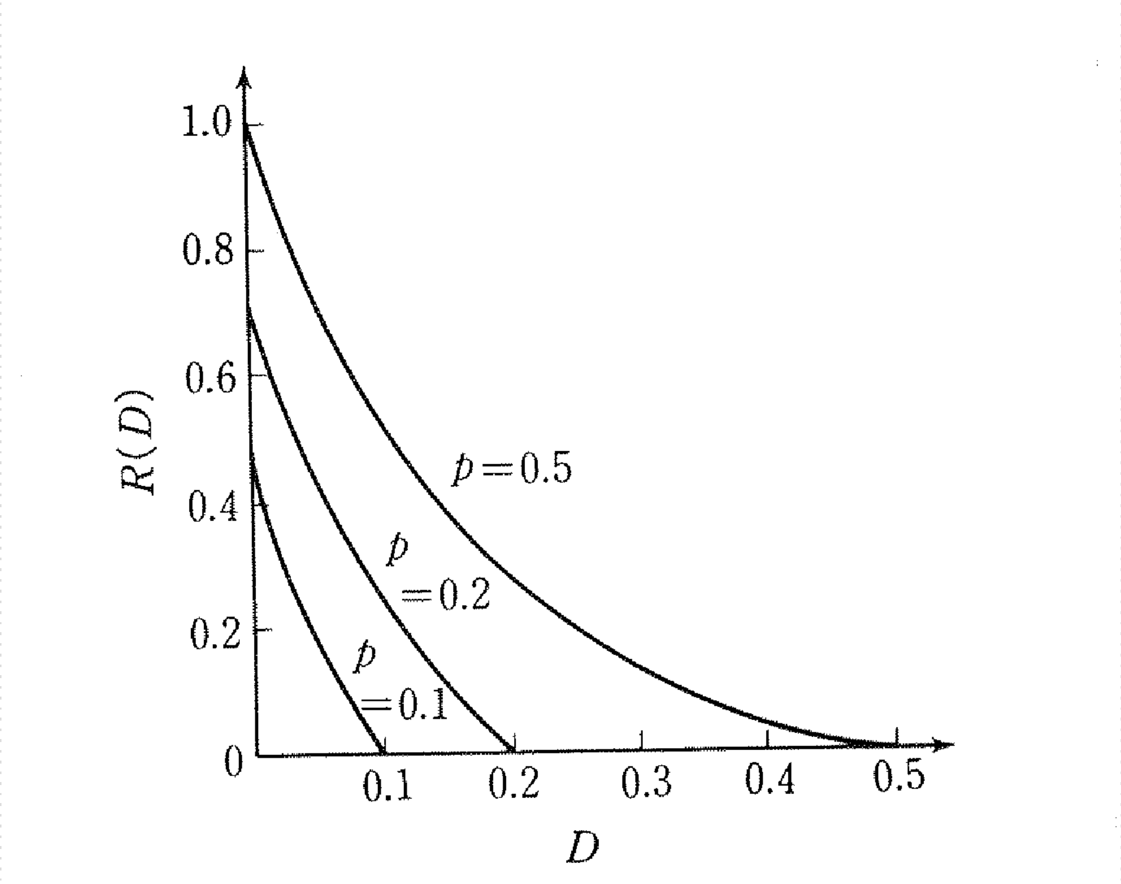
$$I(X;Y) \ge \tilde{H}(p) - \tilde{H}(\bar{d}) \ge \tilde{H}(p) - \tilde{H}(D)$$

Finally, we have a RD function in the following.

$$R(D) = \tilde{H}(p) - \tilde{H}(D)$$

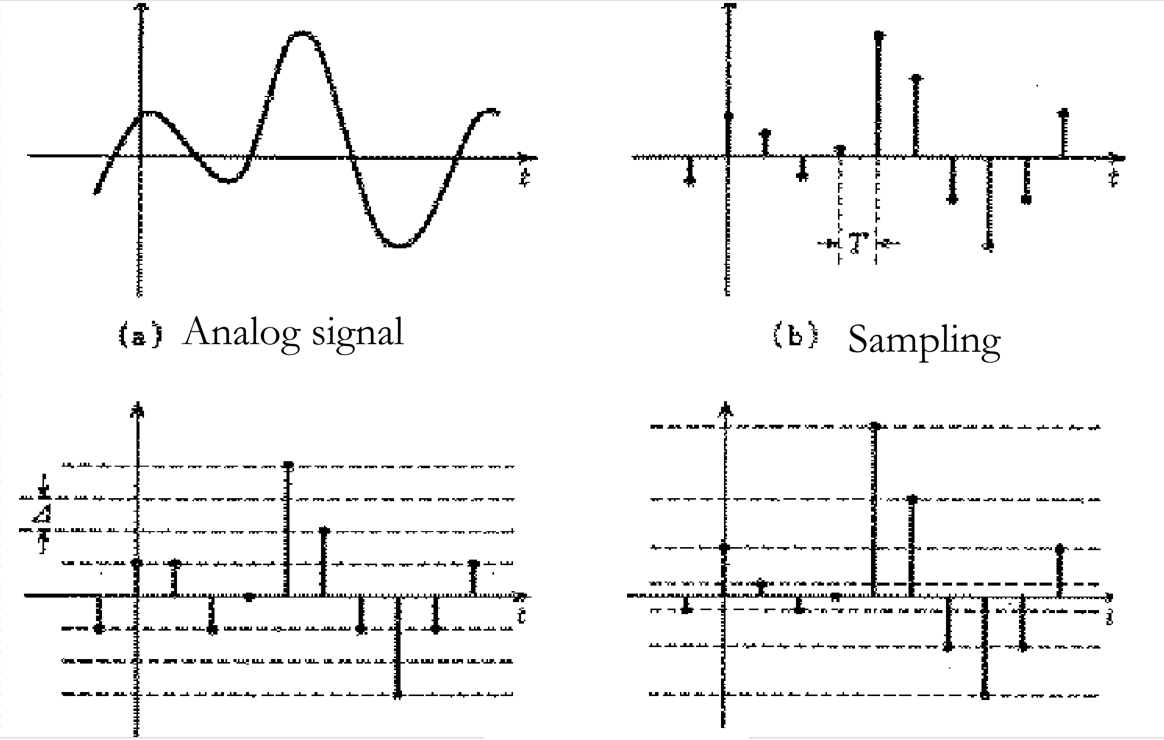
Rate distortion

- RD function for a binary information source



Source coding of analog information

- Analog source coding:
  Here we treat analog source information that can take continuous value not a symbol. (ex. Speech, Image, Sensory input)



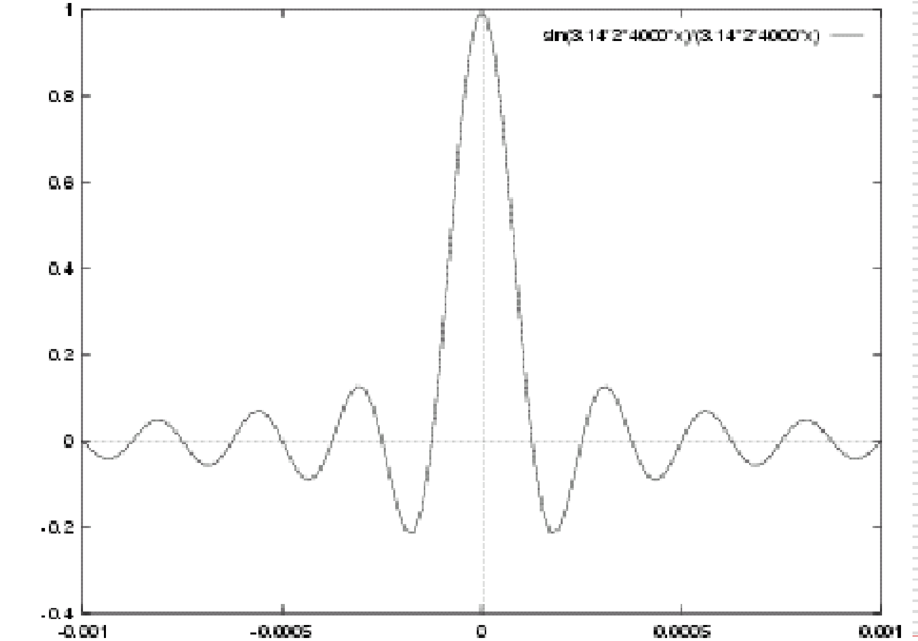
Quantization

Sampling

- If the frequency band is limited to 0-W[Hz], the function f(t) can be written by,

$$f(t) = \sum_{k=-\infty}^{\infty} X_k \frac{\sin \pi(2W_t - k)}{\pi(2W_t - k)}$$

$$X_k = f(k/2W)k = \ldots -1, 0, 1, 2, \ldots$$



Sampling

- Let spectrum of f(t) be a F(w), it can be written,

$$F(w)=\int_{-\infty}^{\infty} f(t)e^{-iwt}dt \qquad (1)$$

If F(w) is band limited in $-2\pi W \le w \le 2\pi W$, it can be transformed by Fourier expansion.

$$F(w)=\sum_{k=-\infty}^{\infty} a_k e^{-i\frac{2\pi k w}{4\pi W}} = \sum_{k=-\infty}^{\infty} a_k e^{-i\frac{kw}{2W}} \qquad (2)$$

, here

$$a_k = \frac{1}{4\pi W}\int_{-2\pi W}^{2\pi W} F(w)e^{iw\frac{k}{2W}}dw \qquad (3)$$

From (1) ,

$$f(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty} F(w)e^{iwt}dw$$

$$=\frac{1}{2\pi}\int_{-2\pi W}^{2\pi W} F(w)e^{iwt}dw \qquad (4)$$

Sampling

Now we set $t = \dfrac{k}{2W}$,

$$f\left(\frac{k}{2W}\right) = \frac{1}{2\pi}\int_{-2\pi W}^{2\pi W} F(w) e^{iw\frac{k}{2W}} dw \qquad (5)$$

comparing (3) and (5),

$$a_k = \frac{1}{2W} f\left(\frac{k}{2W}\right) \qquad (6)$$

Therefore, we get

$$F(w) = \sum_{k=-\infty}^{\infty} \frac{1}{2W} f\left(\frac{k}{2W}\right) e^{-iw\frac{k}{2W}} \qquad (7)$$

$$\begin{aligned} f(t) &= \frac{1}{4\pi W}\sum_{k=-\infty}^{\infty} f\left(\frac{k}{2W}\right)\int_{-2\pi W}^{2\pi W} e^{iw\left(t-\frac{k}{2W}\right)} dw \\ &= \sum_{k=-\infty}^{\infty} f\left(\frac{k}{2W}\right)\frac{\sin \pi(2Wt-k)}{\pi(2Wt-k)} \qquad (8) \end{aligned}$$

Entropy of analog source

- The Entropy for digital source is defined,

$$H = -\sum_i p_i \log p_i$$

How can we define Entropy for stochastic variable x that takes continuous value?
Now we divide a region into small region Δx of x. The probability of which x takes a value between xi and $x_i + \Delta x$ can be approximated by,

$$p(x_i)\Delta x$$

The smaller the Δx is, the better approximation we have. Then,

$$H' = \lim_{\Delta x \to 0}(-\sum_i p(x_i)\Delta x \log\{p(x_i)\Delta x\})$$

$$= \lim_{\Delta x \to 0}(-\sum_i p(x_i)\{\log p(x_i)\}\Delta x + \lim_{\Delta x \to 0}(-\sum_i p(x_i)\{\log \Delta x\}\Delta$$

$$= -\int p(x)\log p(x)dx - \lim_{\Delta x \to 0} \log \Delta x$$

Entropy of analog source

- The second term goes to infinity. We only use this Entropy to compare various analog sources. We define Entropy of analog source only by the first term.

$$H = -\int p(x)\log p(x)dx$$

- Unit Entropy:
  The analog source has n stochastic variables $x_1, x_2, \ldots, x_n$, we define Entropy by,

$$H = -\int \ldots \int p(x_1, x_2, \ldots x_n)\log p(x_1, x_2, \ldots x_n)dx_1 dx_2 \ldots dx_n$$

  The Entropy per one variable is,

$$H = -\lim_{n\to\infty}\frac{1}{n}\int \ldots \int p(x_1, x_2, \ldots x_n)\log p(x_1, x_2, \ldots x_n)dx_1 dx_2 \ldots dx_n$$

  This is called an unit Entropy. And, Entropy normalized by T is called an Entropy per second, H'. By $n = 2TW$, the following relationship holds.

$$H' = 2WH$$

Conditional Entropy

- Definition:

$$H(Y|X) = \int p(x)H(Y|x)dx = -\int\int p(x,y)\log\frac{p(x,y)}{p(x)}dxdy$$

$$H(X|Y) = \int p(y)H(X|y)dy = -\int\int p(x,y)\log\frac{p(x,y)}{p(y)}dxdy$$

,here P(x),P(y) are marginal probability distributions.

$$p(x) = \int p(x,y)dy$$

$$p(y) = \int p(x,y)dx$$

The following relationship holds as well as in the digital information source.

$$H(X,Y) \le H(X) + H(Y)$$

With equality if and only if,

$$p(x,y) = p(x)\cdot p(y)$$

Entropy of Gaussian distribution

- Probability distribution of Gaussian (Normal) distribution is,

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp - \frac{x^2}{2\sigma^2}$$

The Entropy is given,

$$H(X) = -\int_{-\infty}^{\infty} p(x) \log p(x) dx$$

$$= -\int_{-\infty}^{\infty} \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{x^2}{2\sigma^2})\{\log \frac{1}{\sqrt{2\pi\sigma^2}} \exp(-\frac{x^2}{2\sigma^2})\} dx$$

$$= \log \sqrt{2\pi\sigma^2} + \frac{1}{2}$$

$$= \log \sqrt{2\pi e \sigma^2}$$

Entropy of analog source

- Gaussian process:
  [Definition] Let probability distribution of variables $X_{t1}, X_{t2}, \ldots, X_{tn}$ at time $t_1, t_2, \ldots, t_n$ be $P(X_{t1}, X_{t2}, \ldots, X_{tn})$. If P is subject to multi-dimensional Gaussian distribution, we call this process as a Gaussian process. If this process is subject to stationary Markov process, we call it a stationary Markov process. If a power spectrum density $n(w)$ of Gaussian process has a constant value regardless to frequencies, we call it a white Gaussian noise or process.
  If a white Gaussian noise is band limited in frequency range W,

$$n(w) = \begin{cases} \frac{N_0}{2} & |f| \le W \quad (|w| \le 2\pi W) \\ 0 & |f| > W \quad (|w| > 2\pi W) \end{cases}$$

  Furthermore if a time period this white Gaussian noise is limited in T, this process is determined by a sample by $1/2W$, $x_1, x_2, \ldots, x_{2TW}$. Let a power at each sample be , the Entropy at each point is given by,

$$H = \log \sqrt{2\pi e \sigma^2}$$

Entropy of analog source

- Therefore a Entropy for all 2TW samples is,

$$H_{total} = 2TW \log \sqrt{2\pi e \sigma^2}$$

Maximum Entropy

- Distribution function with maximum Entropy:
  Find a probability distribution function with a maximum Entropy under specific conditions. Now we have following relationships,

$$\int_{a_1}^{b_1} \varphi_1(x, p(x))dx = k_1$$

$$\int_{a_2}^{b_2} \varphi_2(x, p(x))dx = k_2$$

$$\ldots\int_{a_n}^{b_n} \varphi_n(x, p(x))dx = k_n$$

  We find p(x) that maximizes an objective function I by the Lagrangean method.

$$I = \int_a^b F(x, p(x))dx$$

Maximum Entropy

- [Case an average power of x given]
  Let an average power to be σ^2,

$$H(X) = -\int_{-\infty}^{\infty} p(x)\log p(x)dx$$

$$\int_{-\infty}^{\infty} x^2 p(x)dx = \sigma^2$$

$$\int_{-\infty}^{\infty} p(x)dx = 1$$

  We have p(x) maximizes H(X) by,

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp - \frac{x^2}{2\sigma^2}$$

  The Entropy with the p(x) is,

$$H(X) = \int_{-\infty}^{\infty} p(x)\log p(x)dx = \log\sqrt{2\pi e\sigma^2}$$

Maximum Entropy

- Maximum Entropy Theorem:
  A probability distribution function of an average power σ^2 that has a maximum Entropy is Gaussian distribution.

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp - \frac{x^2}{2\sigma^2}$$

The Entropy of Gaussian distribution is given by,

$$H(X) = \int_{-\infty}^{\infty} p(x) \log p(x) dx = \log \sqrt{2\pi e \sigma^2}$$

Mutual information

- Let joint probability distribution be p(x,y), If we divide a region of x into Δx and a region of y into Δy. Here $p(x_i)\Delta x, p(y_i)\Delta y, p(x_i, y_i)\Delta x\Delta y$ are probabilities for x takes a value between x and $x+\Delta x$, y takes a value between y and $y+\Delta y$, x and y jointly take values in the region, respectively. The mutual information is given by,

$$
\begin{aligned}
I(x_i; y_i) &= \log \frac{p(x_i, y_i)\Delta x\Delta y}{p(x_i)\Delta x p(y_i)\Delta y} \\
&= \log \frac{p(x_i, y_i)}{p(x_i)p(y_i)}
\end{aligned}
$$

Mutual information

- An average mutual information is,

$$I(X;Y) = \lim_{\Delta x \to 0 \, \Delta y \to 0} \sum_i \sum_j [p(x_i, y_i)\Delta x \Delta y] \log \frac{p(x_i, y_i)}{p(x_i)p(y_i)}$$

$$= \int_{-\infty}^{\infty} p(x,y) \log \frac{p(x,y)}{p(x)p(y)} dxdy$$

$$= H(Y) - H(Y|X)$$

$$= H(X) - H(X|Y)$$

I(X;Y) is non negative value,

$$I(X;Y) \geq 0$$

with equality if and only if,

$$p(x,y) = p(x)(y)$$

Rate distortion for analog source

- Rate distortion function:
  Let an average distortion rate be d(x,y), the average distortion is given by,

$$\bar{d} = \int_{-\infty}^{\infty} d(x,y)\,p(x,y)\,dxdy$$

here, p(x,y) is a joint probability distribution function of a source sample value x and its decoded result y. We have a Rate-distortion function in the similar manner as the discrete symbol case.

$$R(D) = \min_{\bar{d} \le D} I(X;Y)$$

Let R(D) bit/sample be the minimum mutual information I(X;Y) of X and Y under condition that the average distortion $\bar{d}$ is smaller than the threshold D. R(D) provides the lower bound of the average code length per source symbol when we code it by the binary codes under the condition that $\bar{d}$ is smaller than D.

Rate distortion of Gaussian source

- We use the mean squire error,

$$d(x,y) = (x-y)^2$$

The average distortion is given an average squire error.

$$\bar{d} = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} p(x)p(y|x)(x-y)^2 dxdy \le D$$

Under the above condition, we minimize I(X;Y) with P(y|x),

$$I(X;Y) = \int_{-\infty}^{\infty} p(x)[\int_{-\infty}^{\infty} p(y|x)\log\frac{p(y|x)}{p(y)}dy]dx$$

, here

$$p(y) = \int_{-\infty}^{\infty} p(x)p(y|x)dx$$

and

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

If the information is Gaussian source, we can use,

$$H(X) = \log\sqrt{2\pi e\sigma^2}$$

We maximize H(X|Y) instead of minimizing I(X;Y).

Rate distortion of Gaussian source

- Let Z be a probabilistic variable Z=X-Y,

$$H(X|Y) = H(X-Y|Y) = H(Z|Y) \le H(Z)$$

with equality if and only if Z and Y are independent. $\bar{d}$ is smaller than D.

$$\bar{d} = \bar{Z}^2 < D$$

H(Z) will be maximized when p(y|x) follows a Gaussian distribution of mean 0 and variation D according to the maximum Entropy theorem. Then we have,

$$H(Z) = \log\sqrt{2\pi eD}$$

Therefore,

$$I(X;Y) \ge \log\sqrt{2\pi e\sigma^2} - \log\sqrt{2\pi eD}$$

$$= \frac{1}{2}\log\frac{\sigma^2}{D}$$

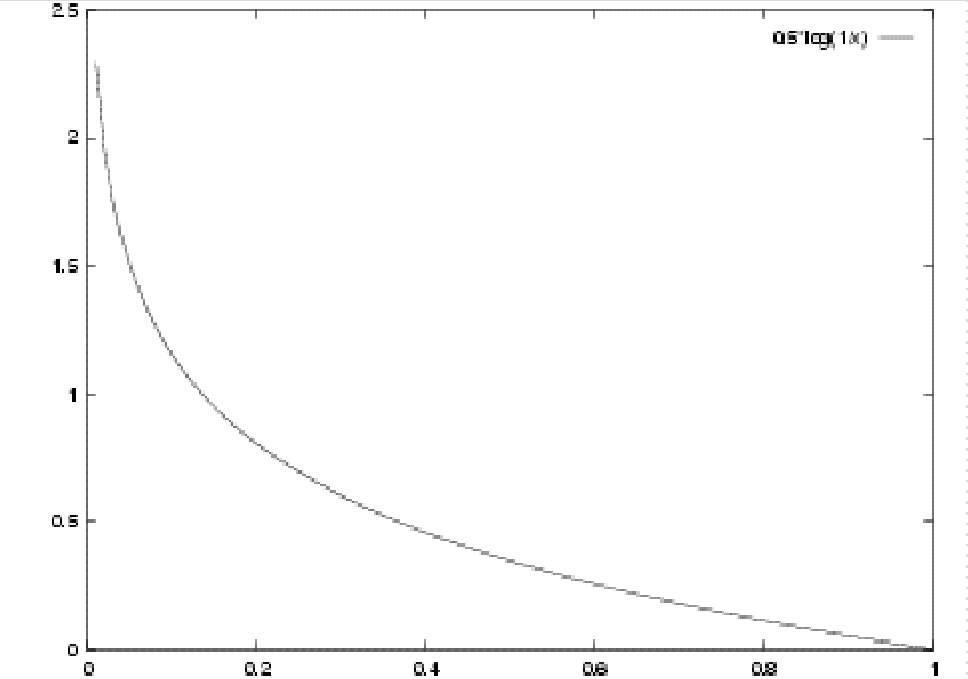
Finally, R(D) is given by,

$$R(D) = \frac{1}{2}\log\frac{\sigma^2}{D} \qquad \text{Bit/sample}$$

Rate distortion of Gaussian source

- When the source signal is band-limited to 0-W, we can have 2W samples per second, the rate-distortion function per second is given by,

$$R(D) = W \log \frac{\sigma^2}{D}$$



Coding of analog signal

- Scalar quantization:
  Scalar quantization is a discretization of value of a source sample. We call this sample as a quantized sample. If we use B bit binary representation, a quantized sample is represented by 2B bits. Therefore, the necessary information for transmission or storage is,

  $$I = B \cdot F_s \quad \text{Bit/second}$$

  This coding is called PCM (Pulse Code mudulation).
  Important thing is to reduce necessary bit rates. Therefore, we utilize a probability distribution of the amplitude distribution. The quantization that minimize mean square errors with fixed quantization level N is called a optimal quantization property.

Coding of analog source

- Signal to Noise ratio:

$$SNR = \frac{E[x^2(n)]}{E[e^2(n)]} = \frac{\sum_n x^2(n)}{\sum_n e^2(n)} = \frac{\sigma_x^2}{\sigma_e^2}$$

Let peak-to-peak ratio of the target signal be 2Xmax, the quantization level of B bit quantization is,

$$\Delta = \frac{2X_{\max}}{2^B}$$

If we assume that the noise amplitude distribution is uniform, we get,

$$E[e^2(n)] = \frac{1}{\Delta}\int_{-\frac{\Delta}{2}}^{\frac{\Delta}{2}} x^2 dx = \frac{\Delta^2}{12} = \frac{x_{\max}^2}{3 \cdot 2^{2B}}$$

SNR will be,

$$SNR = 3 \cdot 2^{2B} \left(\frac{\sigma_x^2}{X_{\max}} \right)^2$$

Representation in dB will be,

$$SNR[dB] = 6B + 4.77 - 20\log_{10}\left(\frac{X_{\max}}{\sigma_x}\right)$$

Coding of analog source

- Transform coding:
  If we have consecutive two sample x1,x2 that have a uniform probability distribution depicted in figure, where p(x1,x2) is,

$$p(x_1,x_2)=p(x)=\begin{cases}\frac{1}{ab} & x\in C\\ 0 & \in C\end{cases}$$

range of x1, and x2 values is, $\dfrac{a+b}{-2\sqrt{2}}<x_1,x_2<\dfrac{a+b}{2\sqrt{2}}$

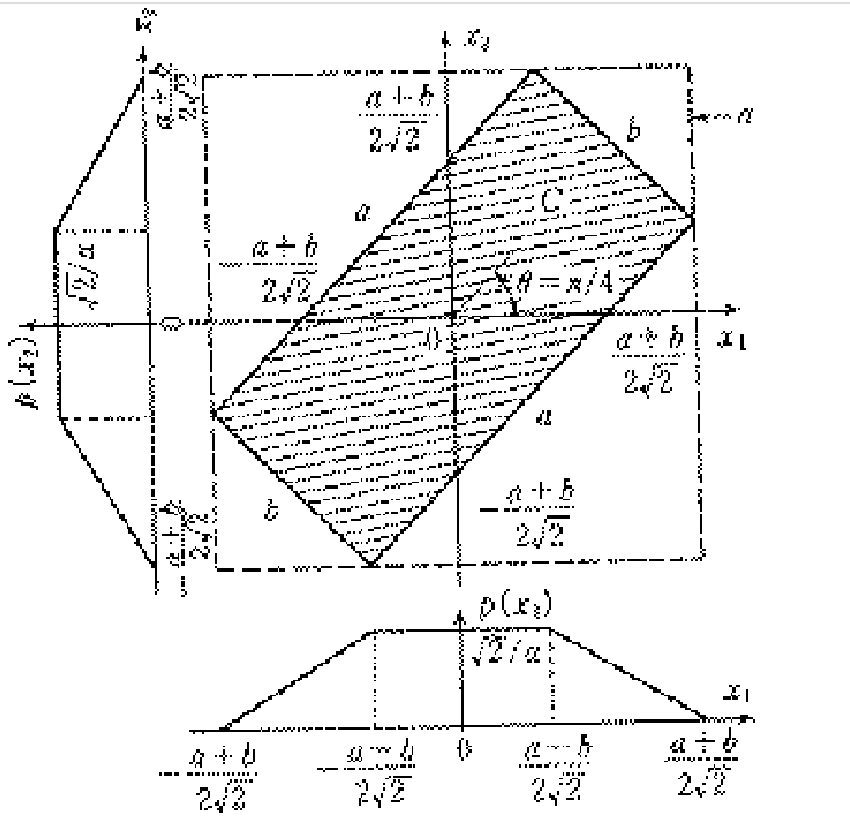
Quantization level will be, $L_1=L_2=\dfrac{a+b}{\sqrt{2}\Delta}$

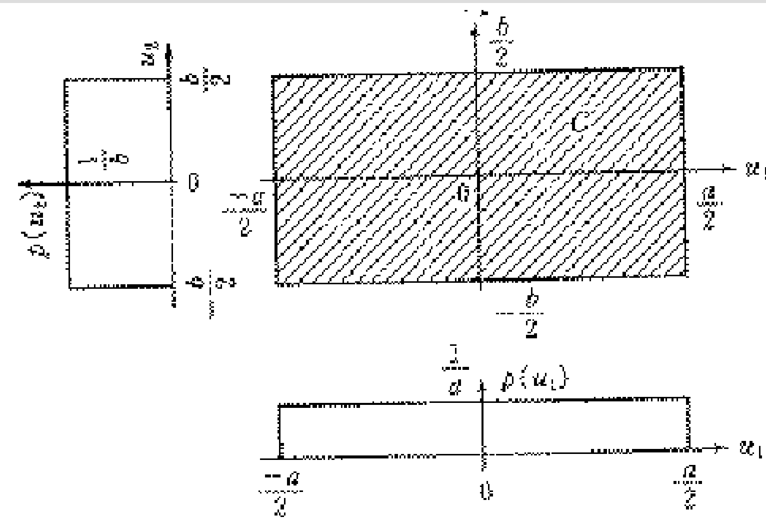
We need $B_x=\log L_1 \log L_2=\log\dfrac{(a+b)^2}{2\Delta^2}$ bits to quantize x=(x1,x2) bits.
If we rotate 45 degree to have new basis (u1,u2). U1 and u2 are independent, necessary quantization levels are L1 for u1, and L2 for u2.

$$L_1=\frac{a}{\Delta},\ L_2=\frac{b}{\Delta}$$

Coding of analog source

- Namely we need $B_u = \log L_1 \log L_2 = \log \frac{ab}{\Delta^2}$ bits to quantize u=(u1,u2).
  For example if a=2b, $B_x - B_u = 1.17$

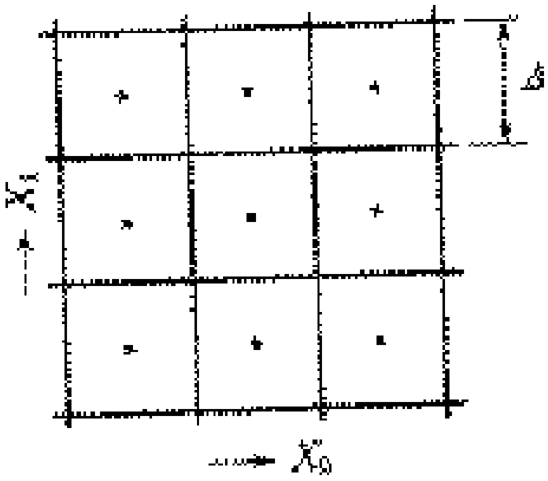




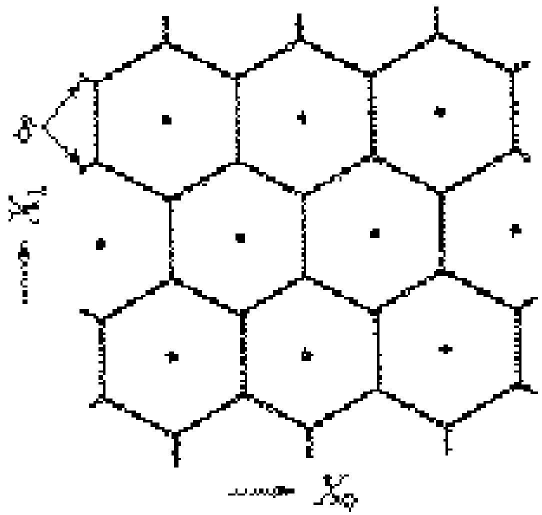
Vector quantization

- A method quantizes not a single sample but a set of n samples.
- Suppose we have source samples that are independent each other and have a uniform distribution. This quantization is equivalent to assigning this sample to a center point of he square area that is made by splitting x0,x1 2dimensional area by squares. The size of the area is Δ^2 , and quantization error is $\Delta^2/6$, average mean square error per one sample is $\Delta^2/12$, this is a same as scalar quantization.
- If we change the shape of the region to a hexagon, the size of the area is $3\sqrt{3}\delta^2/2$ and average quantization error is $5\sqrt{3}\delta^2/8$ with the same number of the representative points.
- If we set the area size to be the same of the square and the hexagon, the average power of the hexagon becomes $5\sqrt{3}/9 = 0.962$

Vector quantization



(a) Representative points of a square



(b) Representative points of a hexagon

Vector quantization

Vector quantization

- Vector quantization is a quantization method that codes a source sample $(x_0,x_1,\ldots,x_{n-1})$ composed of n consecutive samples to a closest representative code chosen from representative codes in n dimensional sample space $(X_0,X_1,\ldots,X_{n-1})$.
- If we apply vector quantization to a source sample so as to minimize an average distortion and apply distortion-less source coding, we can have a code, of which average length per sample approaches to the lower bound R(D) according to the size of n.

Vector quantization

- Representative points are called code words or code vectors. A set of code words is called a codebook.

- Codebook design algorithm:
  There is no optimal algorithm for the codebook design. Here we introduce a semi-optimal iterative codebook design algorithm.
  Now we have k training samples $x_1, x_2, \ldots, x_k$ and centroids defined in the following.

$$\hat{x} = C(x_1, x_2, ..., x_k) = \arg\min_{x} \sum_{i=1}^{k} d(x, x_i)$$

, here $\arg\min_x f(n)$ means an operation to find n minimizes *f(n)*.

Vector quantization

- LBG(Linde,Buzo,Gray) Algorithm
  - Initialization(Step1)
    Let training sample set be xj, j=0,…,n-1,
    N: Codebook size, m=0, ε : distortion, and $D_{-1} = \infty$
    Set an initial codebook $A_N^{(0)} = y_0^{(0)}, \ldots, y_{N-1}^{(0)}$ randomly.
  - Partitioning(Step2)
    Cluster xj into N partial sets Si: i=0,…,N by $A_N^{(m)}$.

$$\hat{i} = \arg\min_i d(x_j, y_i^{(m)}), \quad x_j \in S_{\hat{i}}$$

, here the average distortion is given by,

$$D_m = \frac{1}{n} \sum_{i=1} N \sum_{j=0}^{n-1} \{ d(x_i, y_i^{(m)}) \mid \ x_j \in S_i \}$$

  - If $(D_{m-1} - D_m)/D_m < \varepsilon$, then stop, else set $A_N^{(m)}$ be a codebook .
  - Calculate $A_N^{(m+1)} = y_0^{(m+1)}, \ldots, y_{N-1}^{(m+1)}$, $y_i^{(m+1)} = C(\{S_i\})$ $m \leftarrow m+1$ go to step 2.

Vector quantization

- Splitting algorithm:
  - (Step1)Initialization:
    Δ : Arbitrary vector with small norm.
    $M=1$, $A_{0,1} = C(x_1, x_2, \ldots, x_{n-1})$
  - (Step2)Split $A_{0,M} = (y_0, y_1, \ldots, y_{M-1})$ into neighboring two vectors, $y_i + \Delta, y_i - \Delta$
    Let $\{y_0 - \Delta, y_0 + \Delta, y_i - \Delta, y_i + \Delta, \ldots, y_{M-1} - \Delta, y_{M-1} + \Delta\}$ be,
    $A_{0,2M} = \{y_0, y_1, \ldots, y_{2M-1}\}$
  - (Step3)Letting $A_{0,2M}$ be initial values, find sub-optimal codebook $A_{0,2M} = \{y_i; i = 0,1,\ldots, y_{2M-1}\}$ by a LBG algorithm. If $M=N$ then stop, else set $M=2N$ and go to step 2.
  - Splitting and LBG algorithm generate a codebook of size 2^N.

D(R) function

- Distortion rate function:
  Let x be N consecutive samples of x(n), vector quantization that codes x into y with a codebook size of L is given by,

$$D_N(R) = \min_y E[d(x, y)]$$

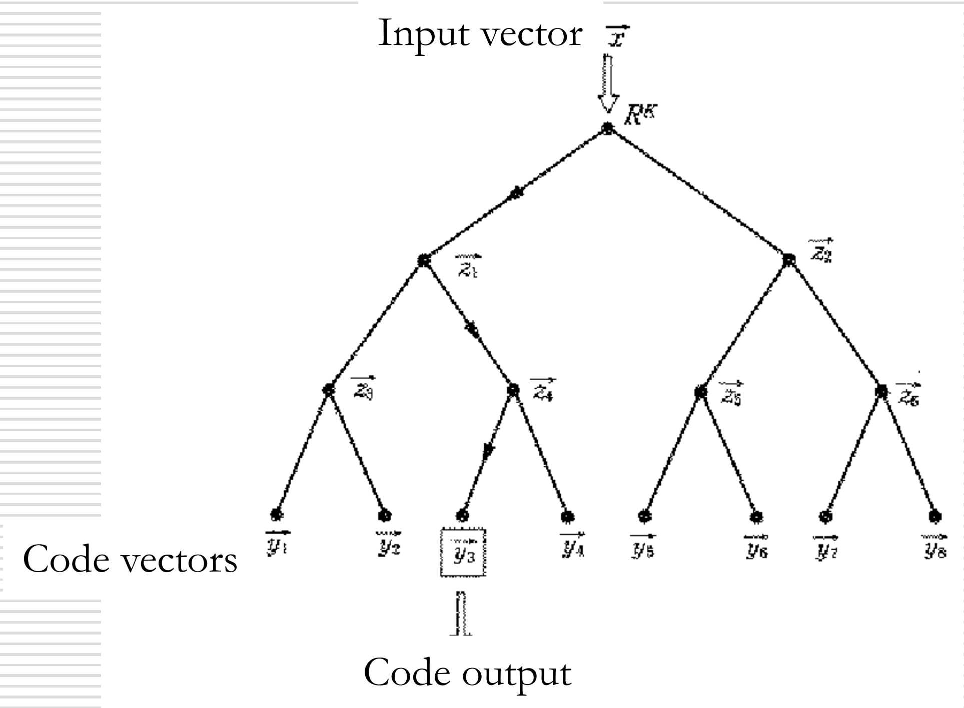
, where
$$\frac{1}{N} H(y) \le R$$

$$D(R) = \lim_{N \to \infty} D_N(R)$$

D(R) represents a minimul average distortion with given range of the rate R. On the other hand, R(D) represents a maximum rate or minimum average code length with given range of the distortion D.

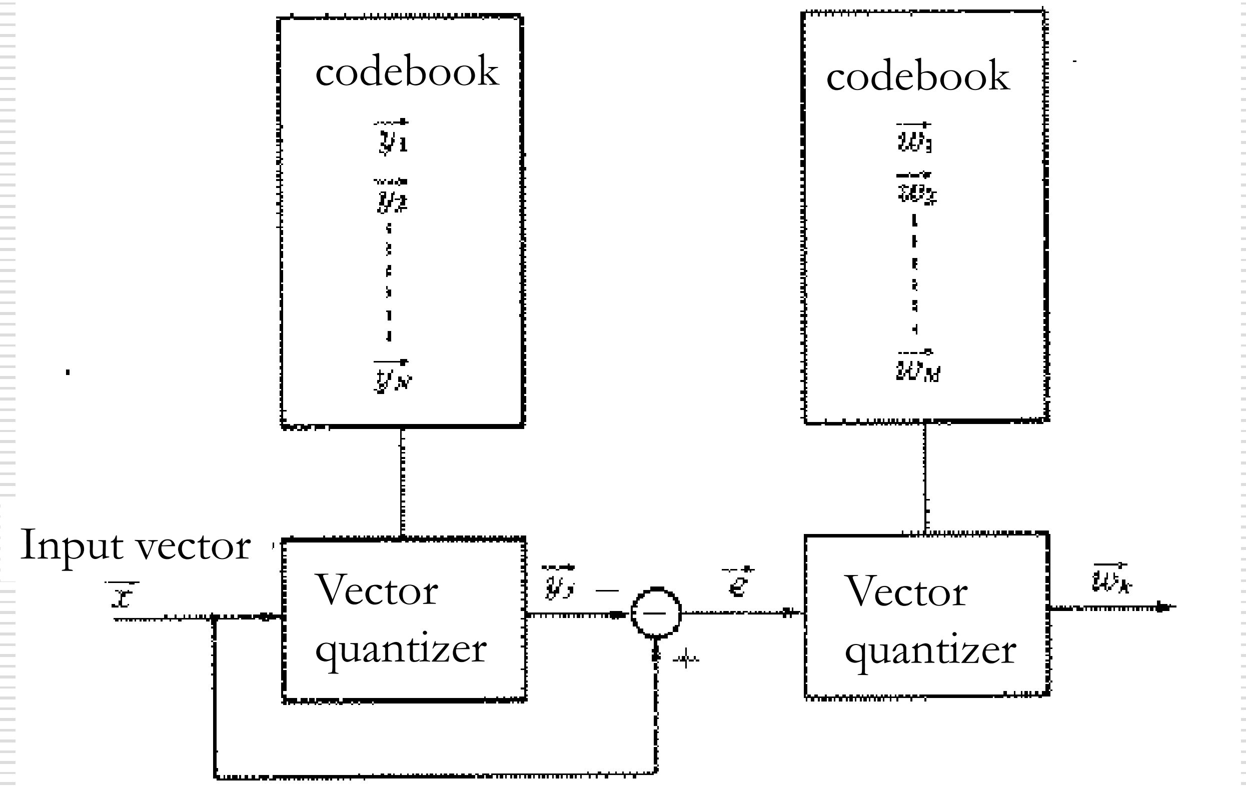
Vector quantization

- Tree search VQ:
  Make tree structure codebook. Each node in the tree represents a code obtained in the splitting algorithm. The computation of the tree search VQ is $K\log_2 N$ to compared to K*N with a parameter dimension of K. The memory size increases about to twice.



Multi-step VQ

- Combine multiple vector quantizers to reduce calculation. Codes of each quantizers are sent to the channel. Number of multiplication can be reduced from K*N*M to K*(N+M).



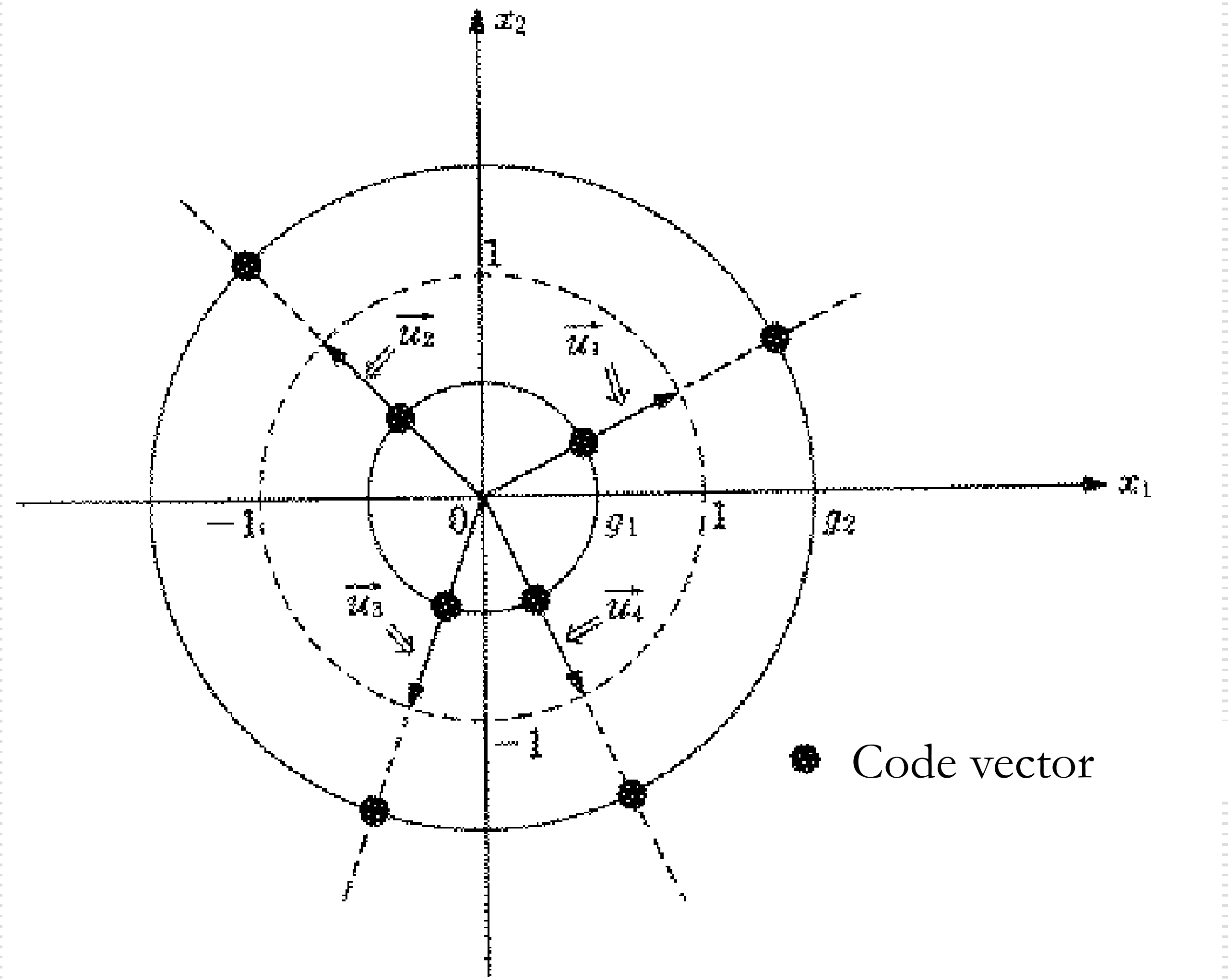
Gain/Shape vector quantization

- Gain/Shape vector quantization:
  Codebook is composed of multiplication of N_g scalar values, $g_1, g_2,..,g_{Ng}$ and N_g unit vectors, $u_1, u_2, \ldots, u_{Ng}$.

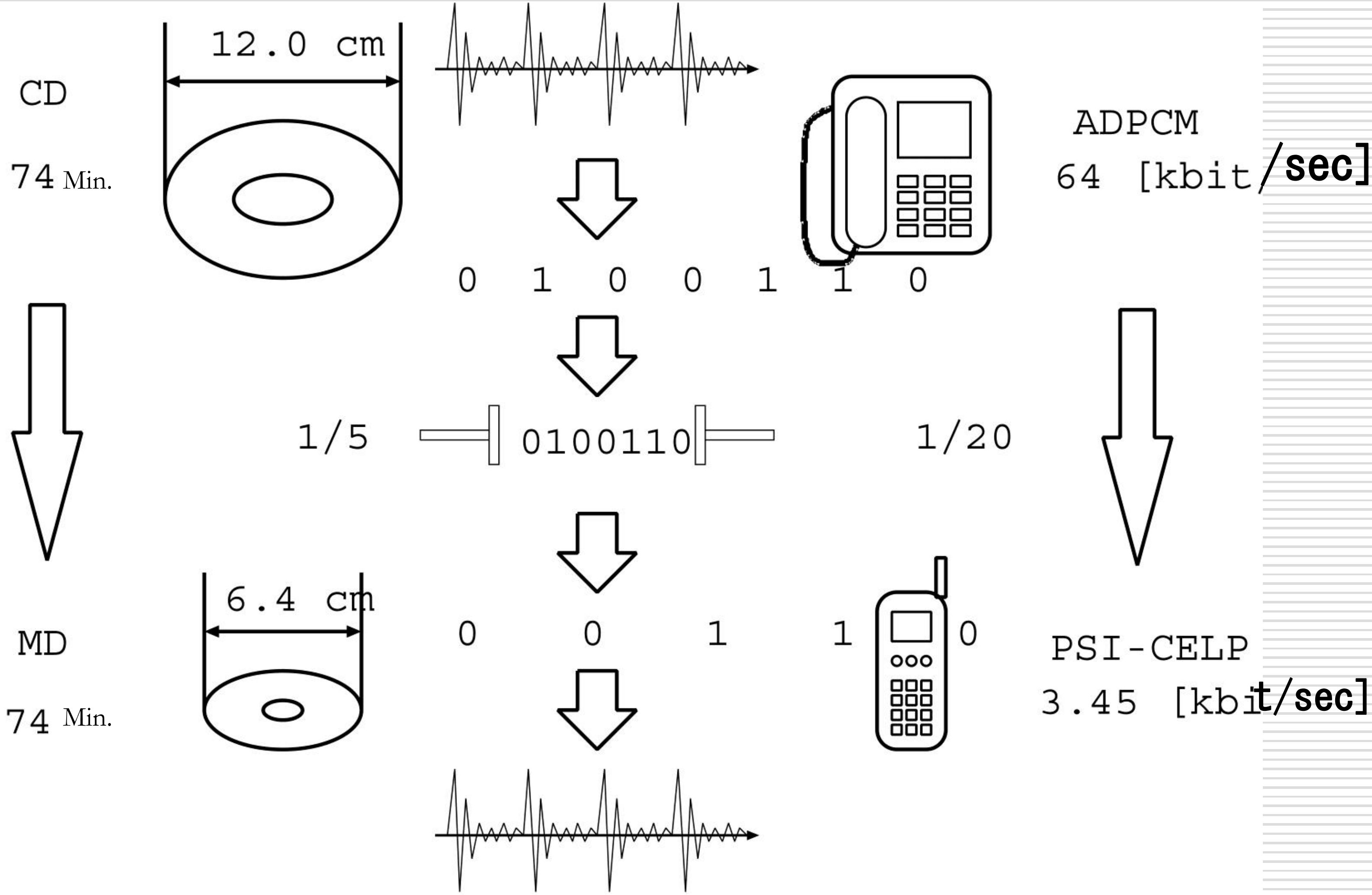
$$g_i \times u_j,\ i = 1, 2, \ldots, N_g,\ j = 1, 2, \ldots, N_s$$

  , here we call $g_1, g_2, \ldots g_{Ng}$ a gain codebook, and $u_1, u_2, \ldots u_{Ns}$ a shape codebook. Coding algorithm is shown in the following.
  - Shape quantization:
    Make inner product between an input vector x and u in the shape codebook $u_1, u_2, \ldots, u_{Ns}$ and find a unit vector u_l gives maximum inner product.
  - Gain quantization:
    Find a closest scalar value from a gain codebook $g_1, g_2, \ldots, g_{Ng}$ to the maximum inner product of (x, u_l). Here $g_k * u_l$ is a quantization vector out of $N_g * N_s$ quantization samples. Therefore number of calculation is reduced from $K*N_g*N_s$ to $K*N_s$ and memory size from N_g*N_s to $K*(N_s+N_g)$.

Gain/shape vector quantization

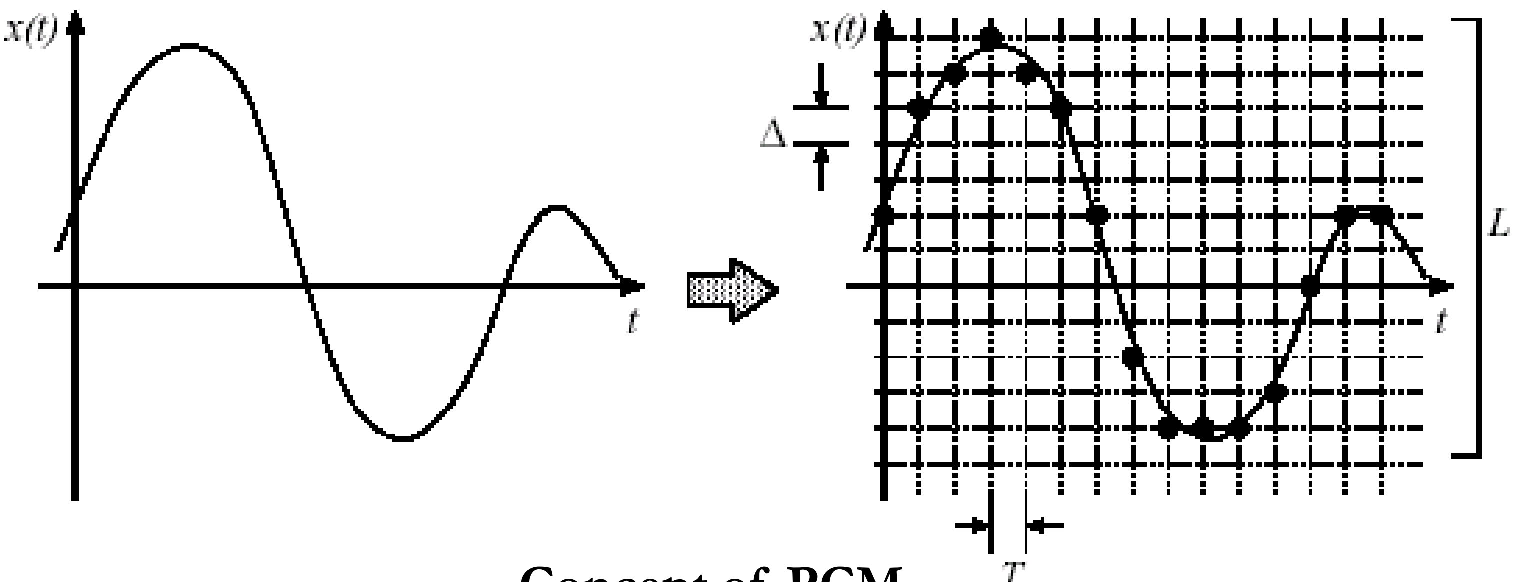


Speech Coding



Waveform Coding

- PCM (Pulse Code Modulation) used in CD, DAT



Concept of PCM -

If signal is band-limited to 0-W[Hz]

$$T \leq \frac{1}{2W}$$

T: Sampling Interval [s]

$$x(t) = \sum_{i=-\infty}^{\infty} x(iT) \frac{\sin\left\{\frac{\pi}{T}(t-iT)\right\}}{\frac{\pi}{T}(t-iT)}$$

Waveform coding (PCM)

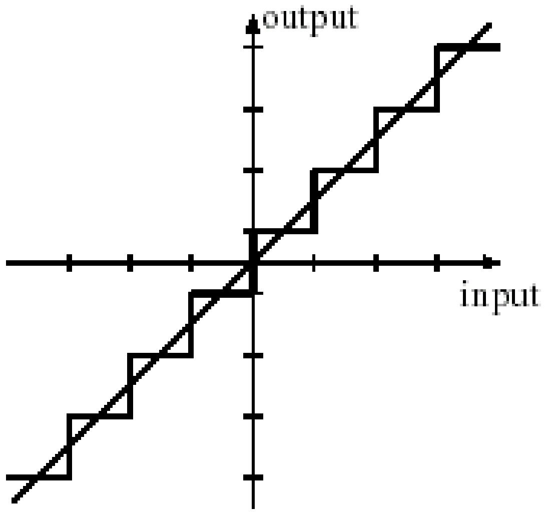
- Quantization

  Let quantization step to be Δ, quantization bit to be *B*, range of signal amplitude to be L.

$$
\begin{aligned}
\Delta 2^{B} &\geq L \\
B &\geq \log_2\left(\frac{L}{\Delta}\right)
\end{aligned}
$$

Waveform coding

- Speech waveform

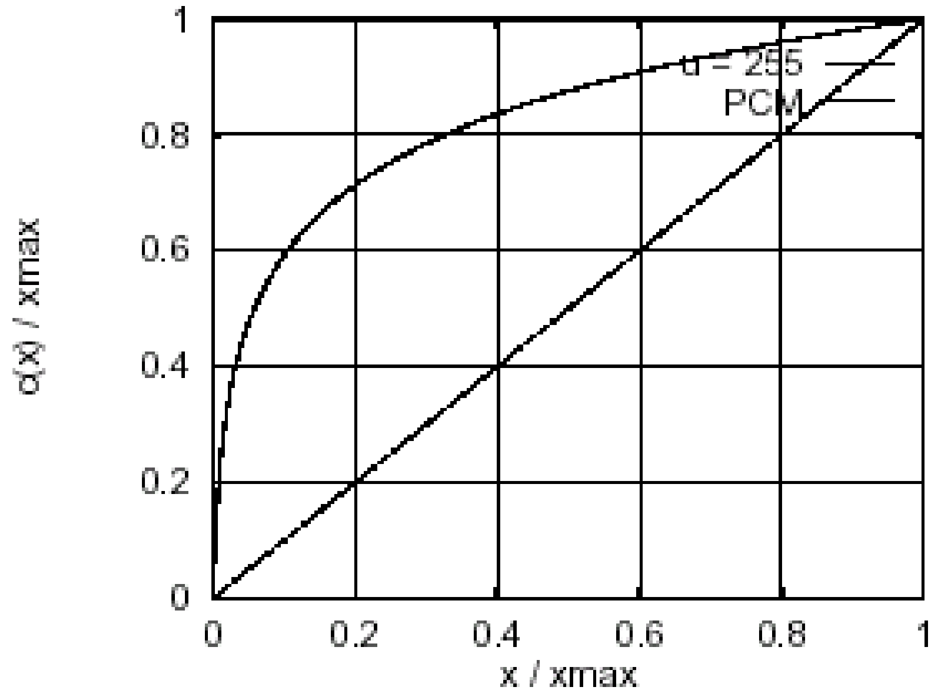


Non-uniform

$$\mu - law$$

Waveform coding (u-law)

- $\mu - law$ is used for ISDN.

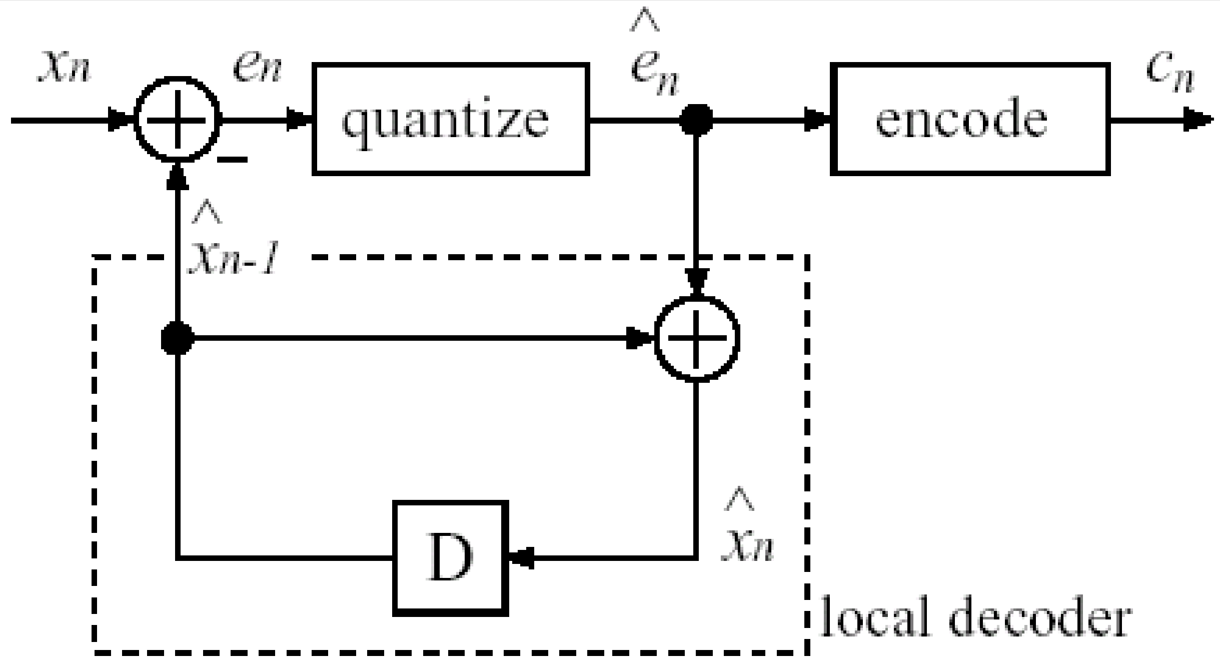


$\mu - law$ **(u=255)**

$$c(x) = x_{max} \frac{\log(1 + \mu \frac{|x|}{x_{max}})}{\log(1+\mu)} \mathrm{sgn}(x)$$

Waveform coding (DPCM)

- DPCM (Differential PCM)



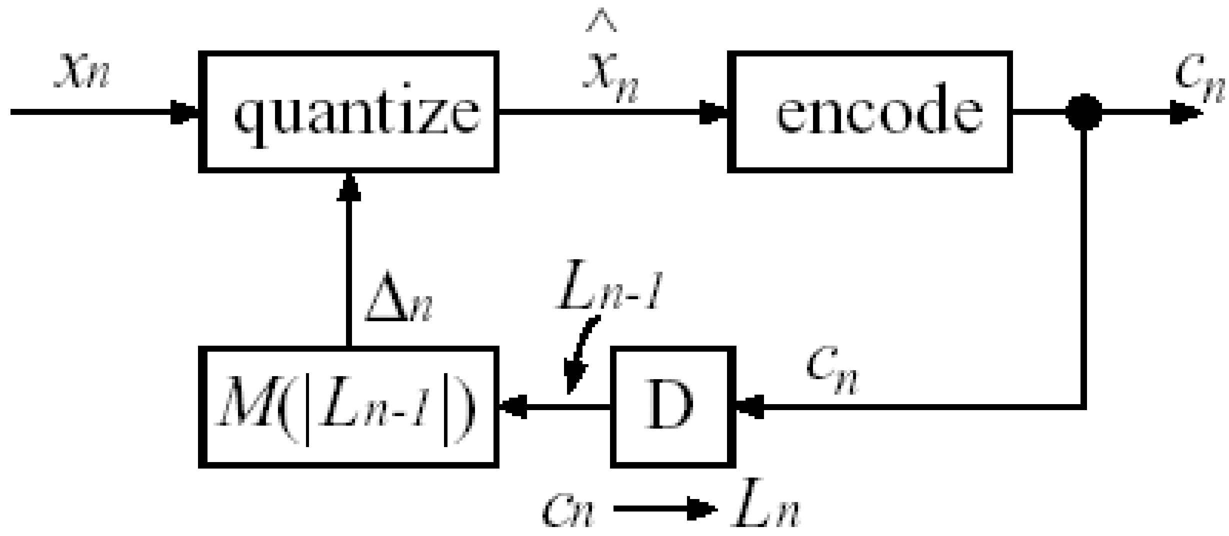
Waveform coding (DPCM)

- If quantization step Δ is 1, quantization bit B is 5.

x_n	3.0	4.0	6.5	8.0	5.3
$\hat{x}_{n-1}$	0	3	4	7	8
e_n	3.0	1.0	2.5	1.0	-2.7
$\hat{e}_n$	3	1	3	1	-3
$\hat{x}_n$	3	4	7	8	5
c_n	10011	10001	10011	10001	00011

Waveform coding (APCM)

- APCM (Adaptive PCM)



$$\Delta_n = \Delta_{n-1} \bullet M(|L_{n-1}|)$$

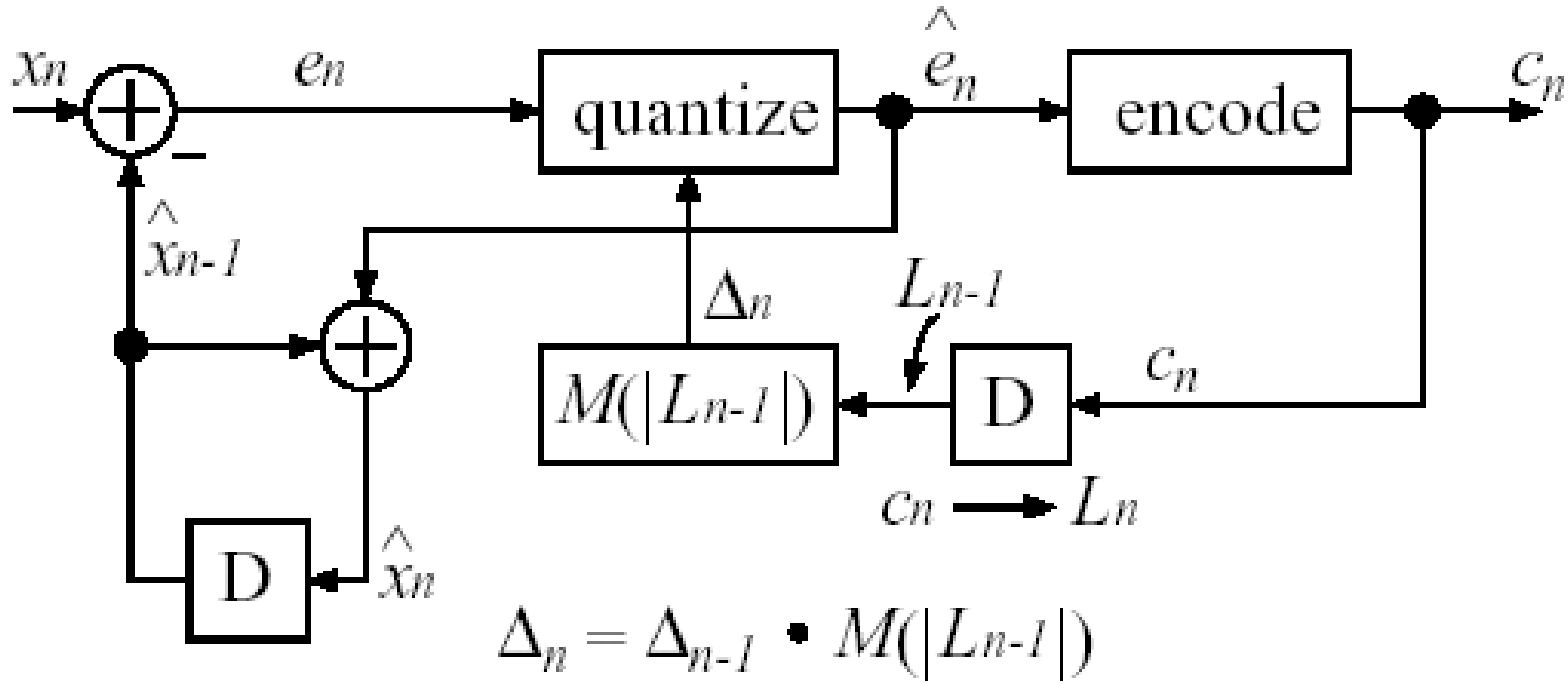
Waveform coding (APCM)

- APCM (Adaptive PCM)

Quant. Bits	$M(\|L_{n-1}\|)$
2	0.6, 2.2
3	0.85, 1, 1, 1.5
4	0.8, 0.8, 0.8, 0.8 1.2, 1.6, 2.0, 2.4
5	0.85, 0.85, 0.85, 0.85 0.85, 0.85, 0.85, 0.85 1.2, 1.4, 1.6, 1.8 2.0, 2.2, 2.4, 2.6

Waveform coding (ADPCM)

- ADPCM (Adaptive Differential PCM)

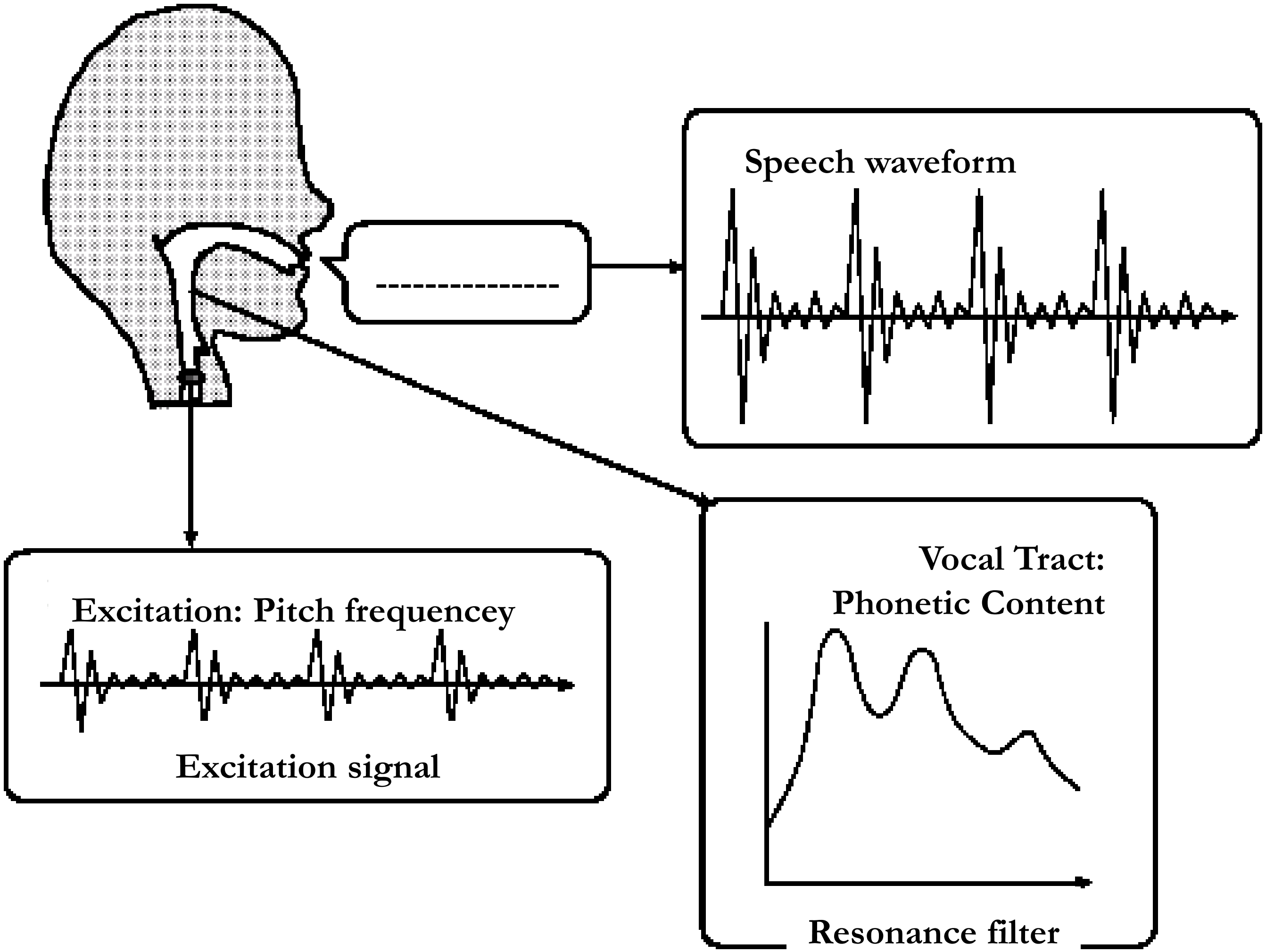


Waveform coding (ADPCM)

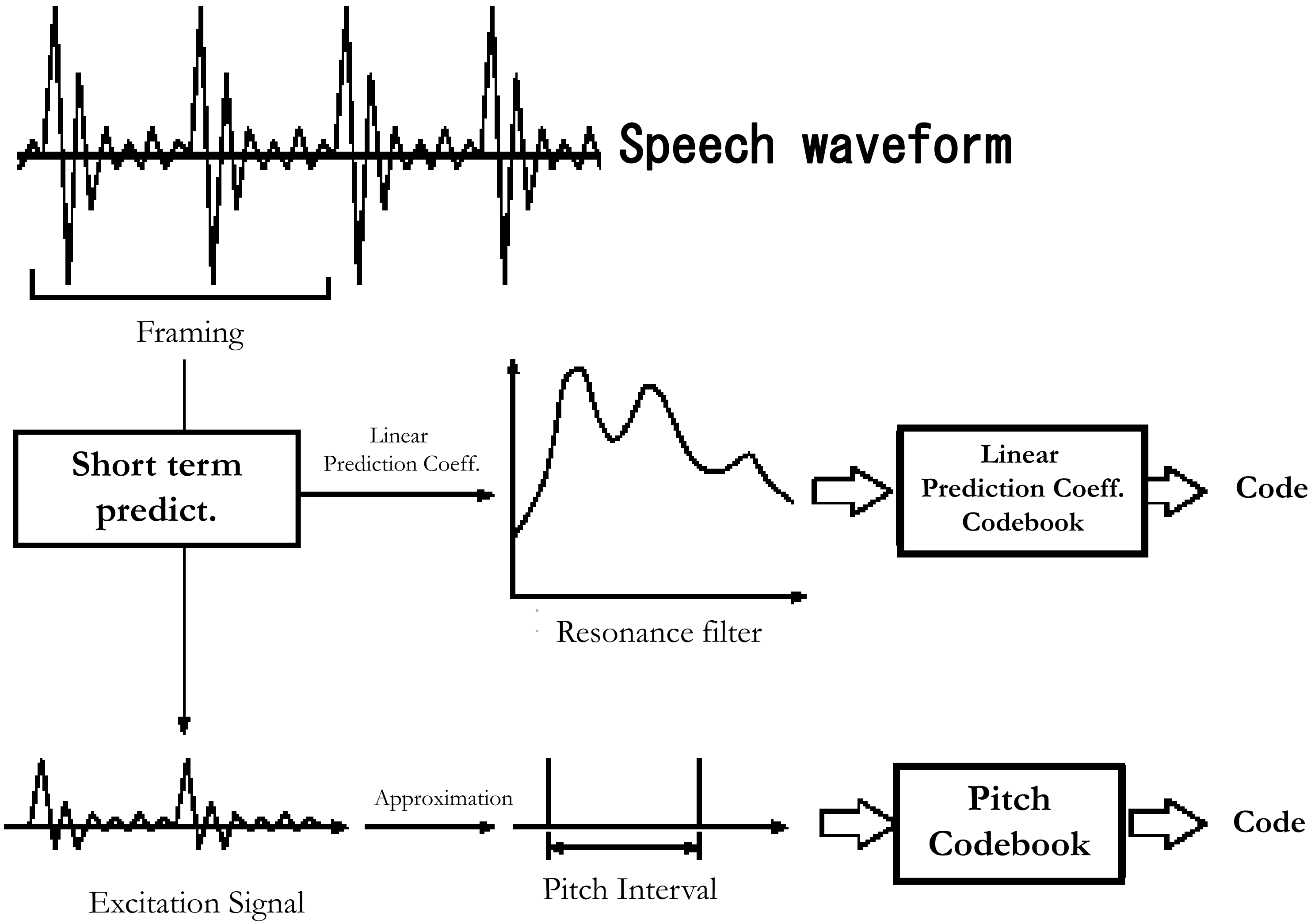
- ADPCM (Adaptive Differential PCM)

Quant. Bits	$M(\|L_{n-1}\|)$
2	0.8, 1.6
3	0.9, 0.9, 1.25, 1.75
4	0.9, 0.9, 0.9, 0.9 1.2, 1.6, 2.0, 2.4
5	0.9, 0.9, 0.9, 0.9 0.95, 0.95, 0.95, 0.95 1.2, 1.5, 1.8, 2.1 2.4, 2.7, 3.0, 3.3

Parametric speech coding



Parametric speech coding

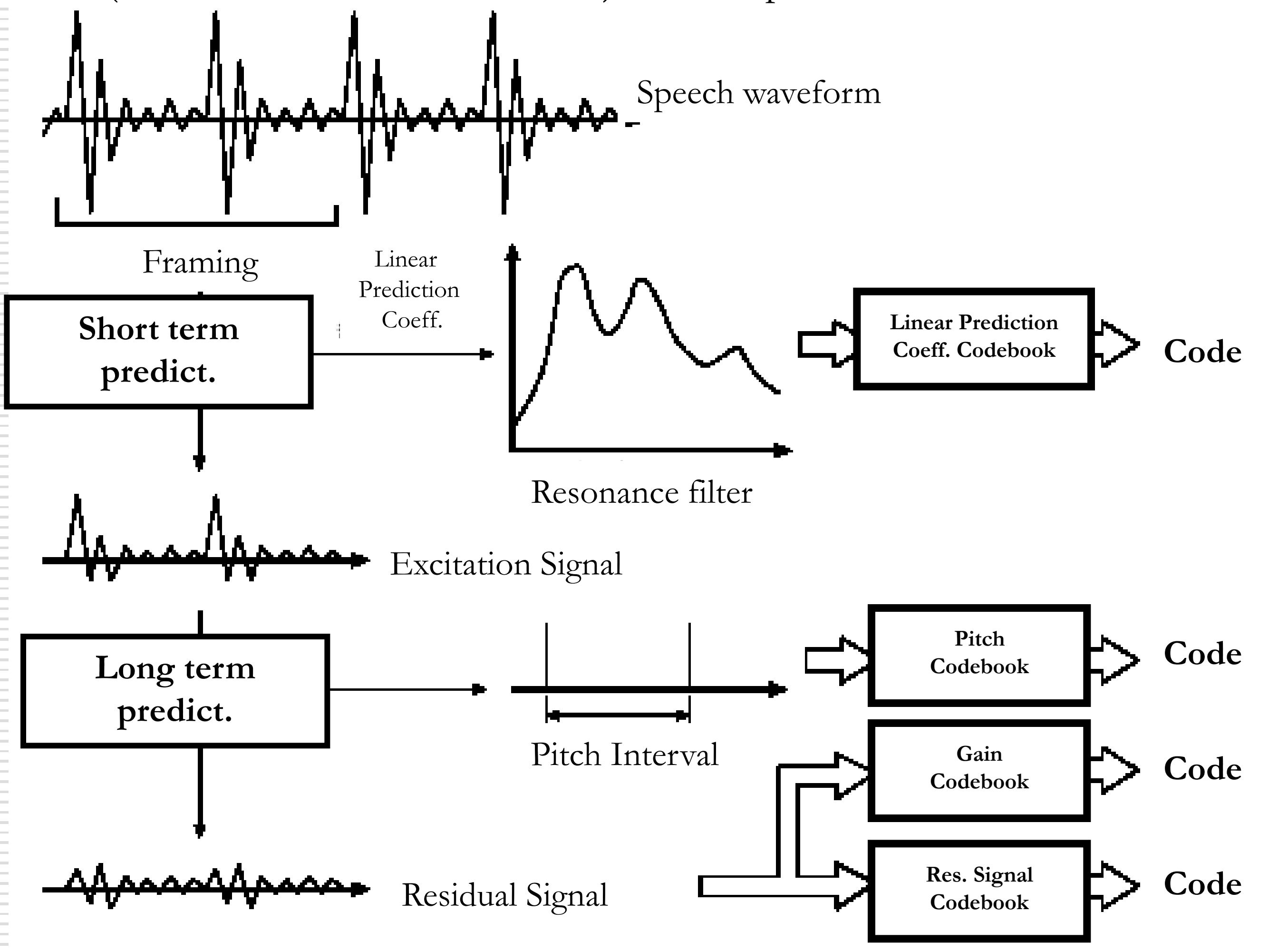


Parametric speech coding

- Points of the parametric speech model
  - Approximation of excitation signal by the Impulse sequence.
  - Bit rates can decrease.
  - However, speech quality degrades seriously.

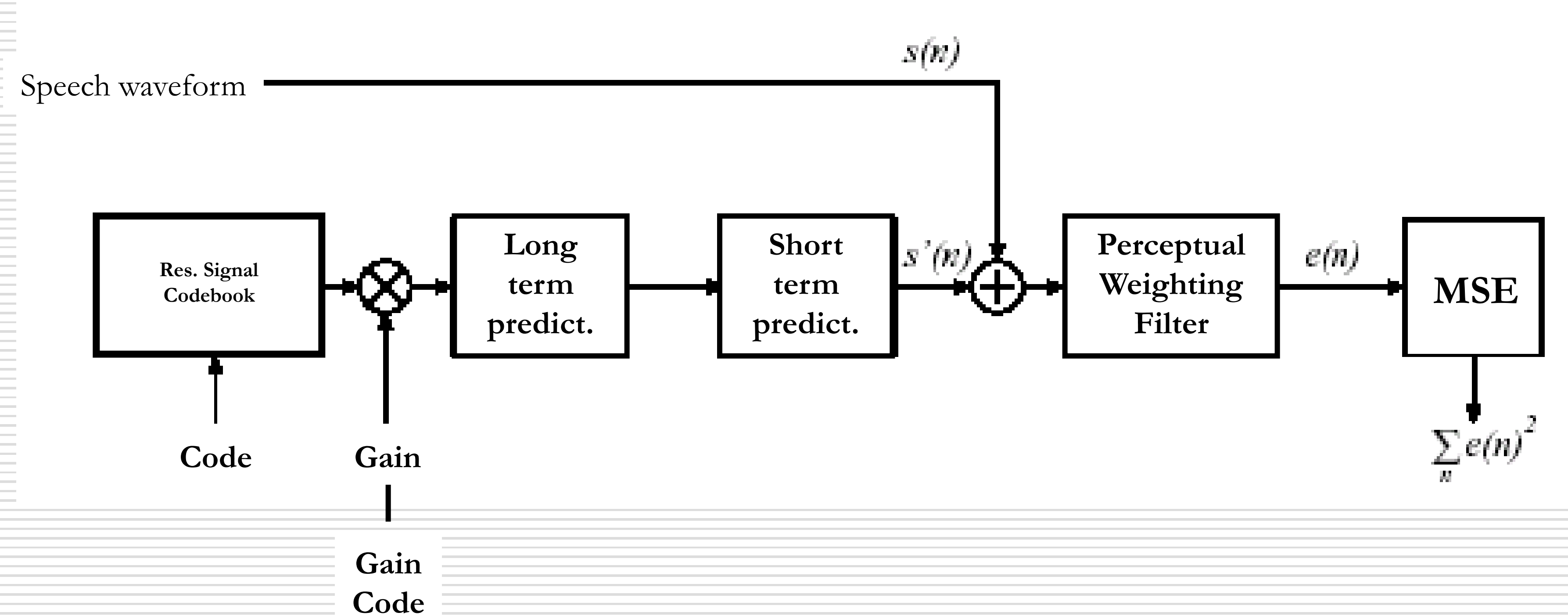
Parametric speech coding (CELP)

- CELP (Code-excited Linear Prediction): Cellular phones

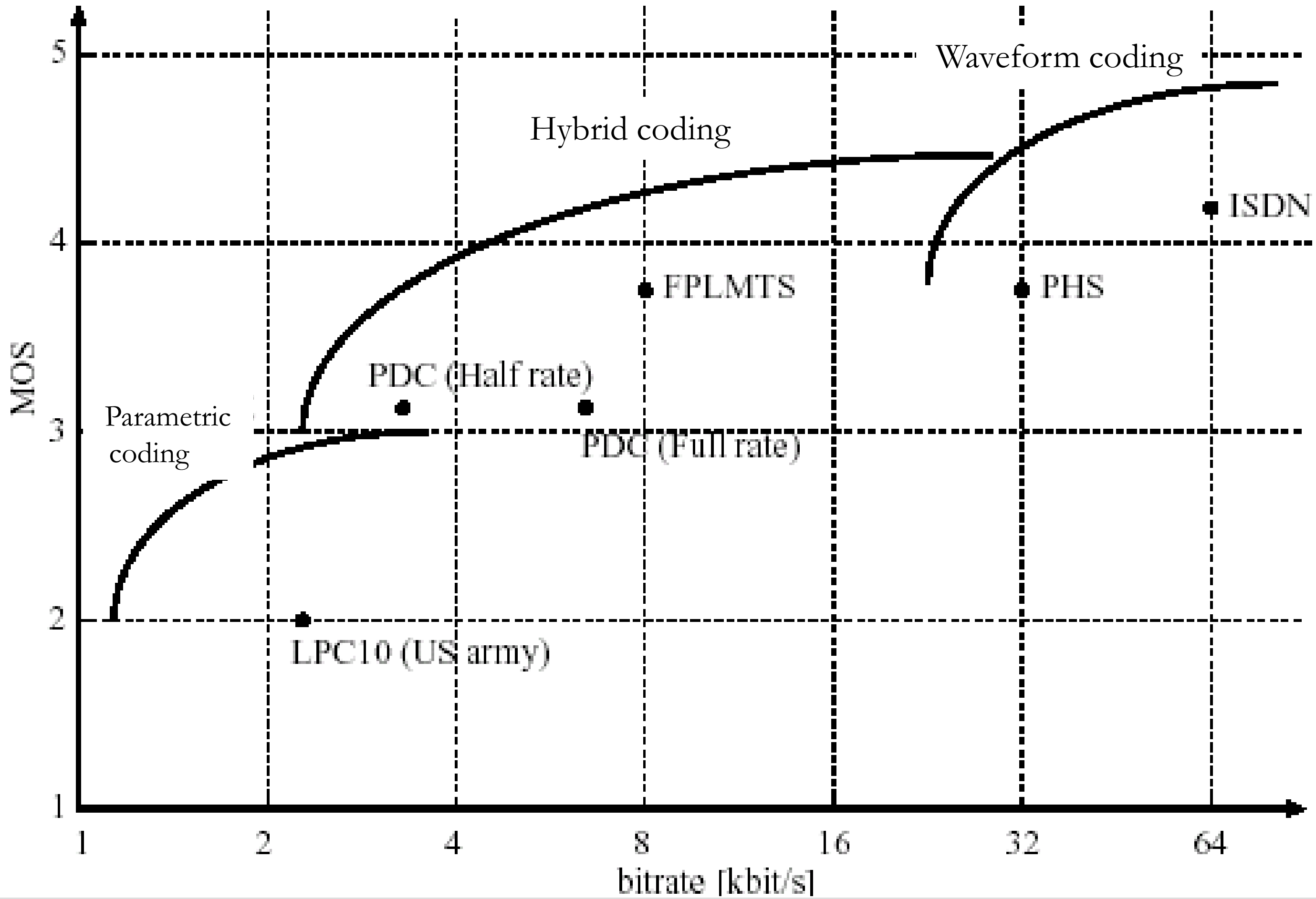


Parametric speech coding (CELP)

- CELP (Code-excited Linear Prediction): Cellular phones

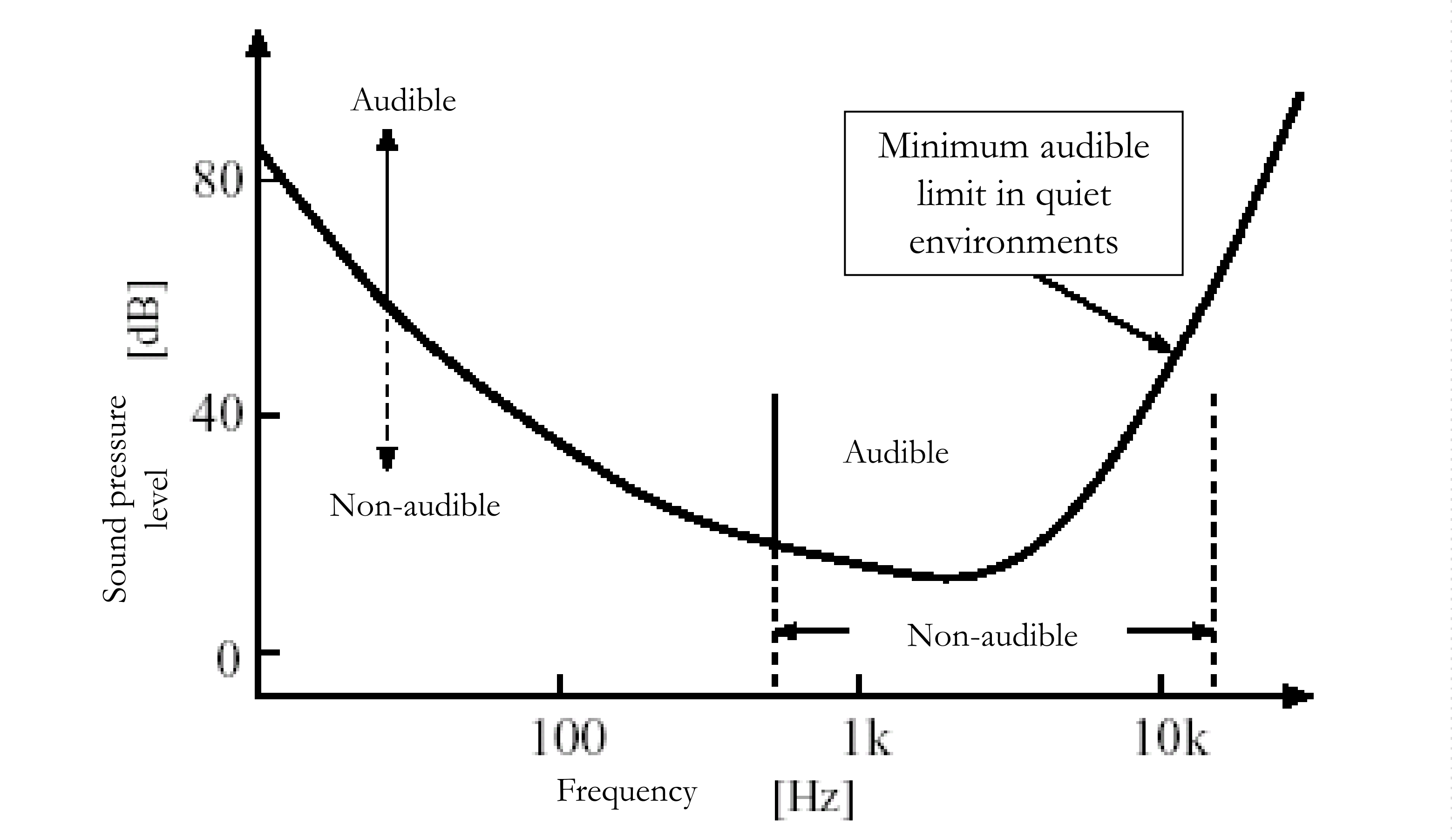


Speech coding



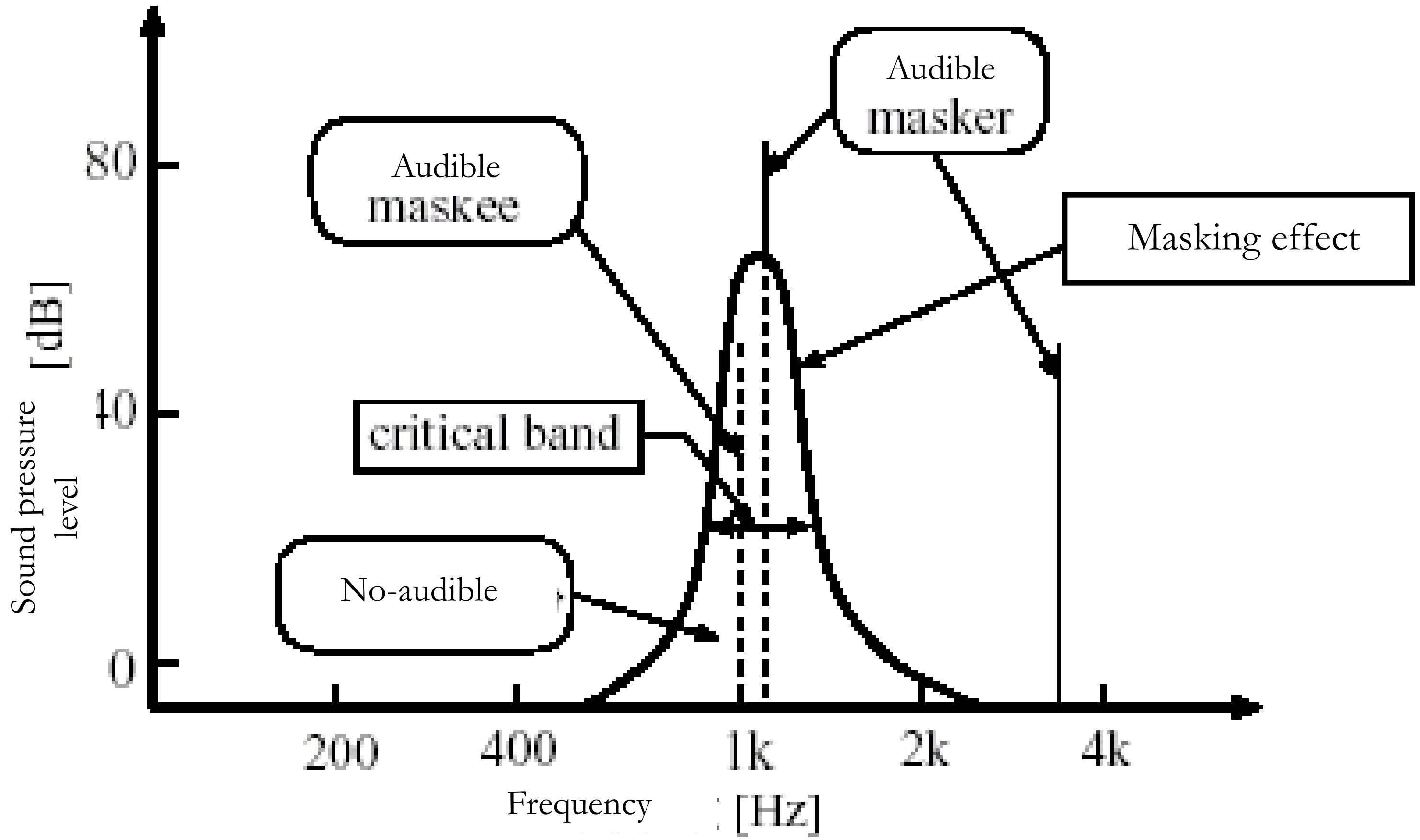
Music coding

- Usage of auditory characteristics for coding not of source model.



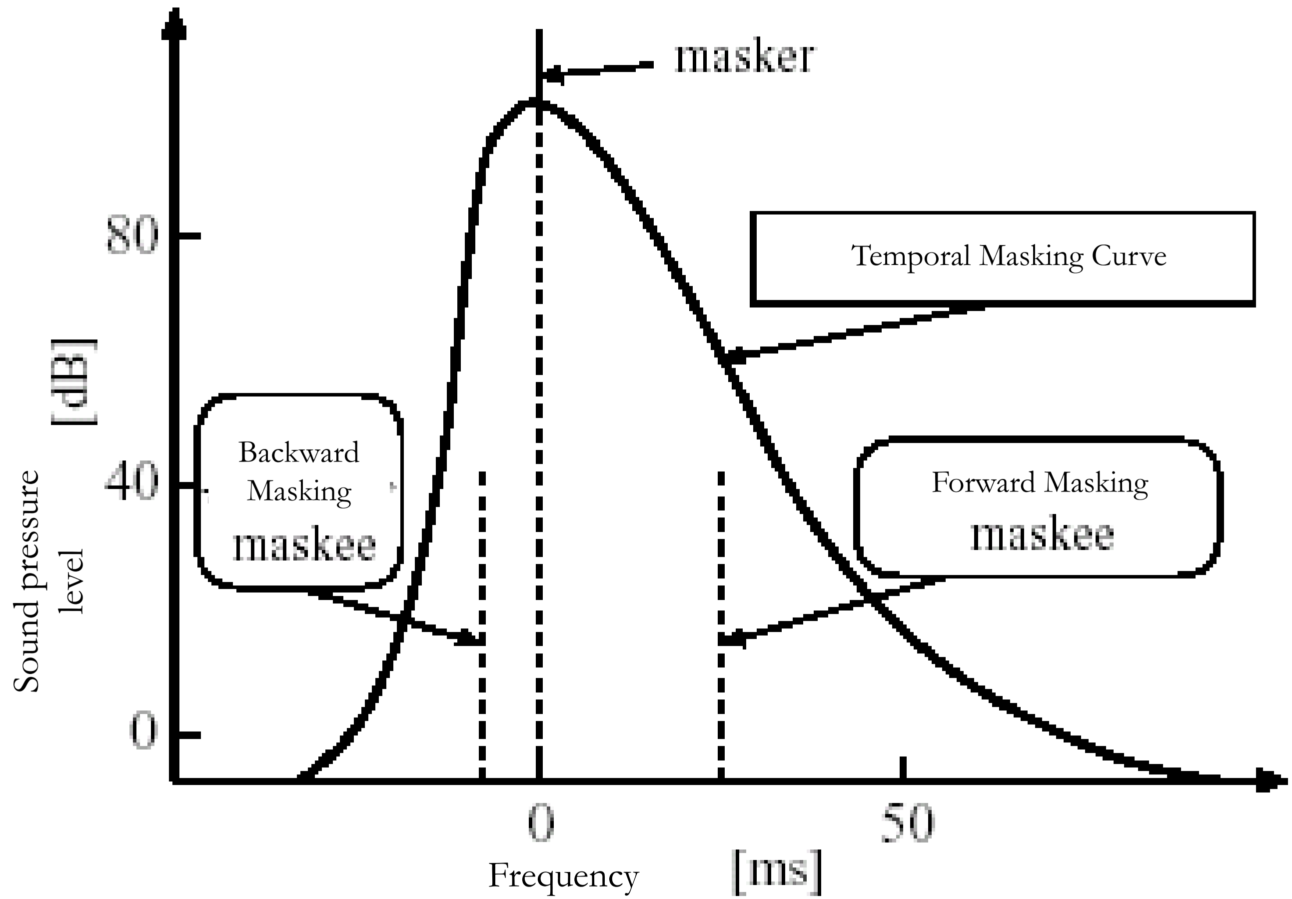
Music coding

- Frequency masking



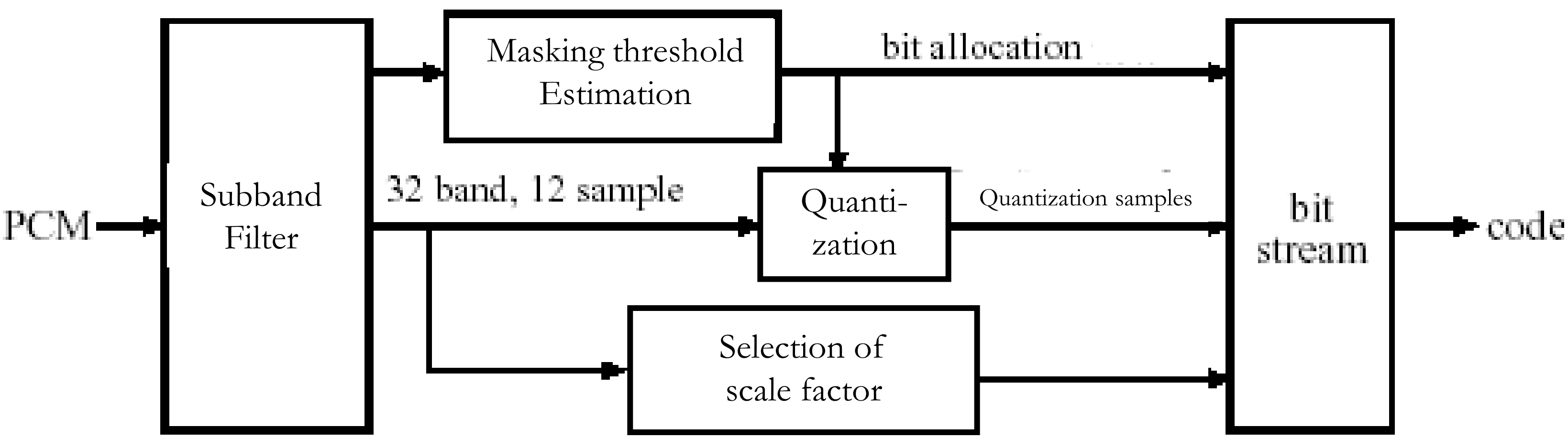
Music coding

- Temporal Masking



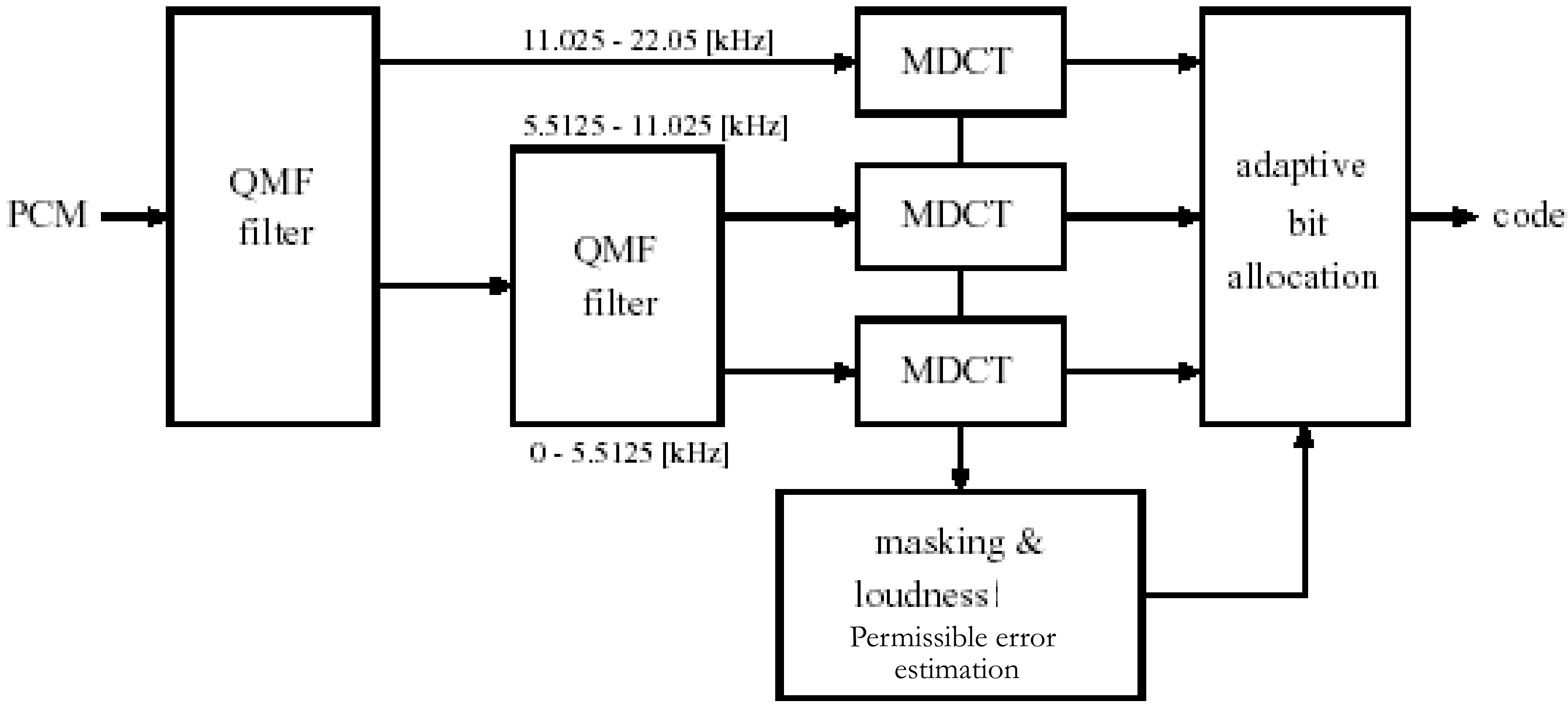
Music coding

【MPEG1 Audio (Moving Picture Experts Group)】



Music coding

【ATRAC (Adaptive TRansform Acoustic Coding)】



MP3: MPEG-1/L3, MPEG-2

		MPEG1/Audio			MPEG2/Audio					
No.		11172.3			13818.3					
IS year		1992			1994					
					Low Sampling Fq.			Multi-lingual, Multi-channnel		
Sampling FQ.		32,44.148			16,22.05,24			32,44.1,48		
Layer		I	II	III	I	II	III	I	II	III
Bit rate	min	32	32	32	32	8	8	32	32	32
	max	448	384	320	256	160	160			
channel		1/0, 2/0			1/0, 2/0			1/0,2/0,3/0,2/1,2/2,3/1,3/2		