KARYNA TRUBCHANINOVA ● OLEKSANDR SERKOV ● OLEG KASILOV

THEORY OF INFORMATION AND ENCODING

TUTORIAL FOR TRAINING STUDENTS IN FOREIGN LANGUAGES MAJORINGIN IN ELECTRONIC COMMUNICATIONS AND RADIO ENGINEERING

Karyna TRUBCHANINOVA
Oleksandr SERKOV
Oleg KASILOV

THEORY OF INFORMATION AND ENCODING

Tutorial for training students in foreign languages majoring in Electronic Communications and Radio Engineering

Kharkiv 2023

UDC 621.391
T77

Recommended for publication of Academic Council by the Institute of Computer Modeling, Applied Physics and Mathematics of National Technical University "Kharkiv Polytechnic Institute" (protocol № 2 from 26.09.2023)

Reviewers:
Ivan OBID, Doctor of Technical Science, Professor, Professor of the of Microprocessor Technologies and Systems Department, Kharkiv National University of Radioelectronics;

Nina KUCHUK, Doctor of Technical Science, Professor, Professor of the of Computer Science and Programming Department, National Technical University "Kharkiv Polytechnic Institute"

Trubchaninova K, Serkov O, Kasilov O.
T77 Theory of information and encoding. Tutorial for training students in foreign languages majoring in Electronic Communications and Radio Engineering / K. Trubchaninova, O. Serkov, O. Kasilov. Kharkiv: Tochka Publishing House, 2023. 120 p.

ISBN 978-617-7856-56-5

This textbook has been given material about the basic principles of information theory and coding, to ensure the security of data transfer, protection from the transfer process of the exchange of knowledge from the theory of coding, protection and formats, architecture of information and computer systems and data transfer protocols. A review of the main theorems of information theory for discrete communication channels in telecommunications and information-computer systems is presented. Information about the principles of optimal and efficient coding is displayed. The material of the textbook is recommended for students majoring in "Electronic Communications and Radio Engineering".

У підручнику надано матеріал про основні положення теорії інформації та кодування, здатних забезпечувати безпеку передачі даних, захист від втручання у процес їх передачі за рахунок застосування знань з теорії кодування, захисту інформації, архітектури інформаційно-комп’ютерних систем та протоколів передачі даних. Викладено огляд основних теорем теорії інформації для дискретних каналів зв’язку у телекомунікаційних та інформаційно-комп’ютерних системах. Висвітлюються відомості про принципи оптимального і завадостійкого кодування. Матеріал підручника рекомендовано для студентів спеціальності «Електронні комунікації та радіотехніка».

UDC 621.391

ISBN 978-617-7856-56-5

CONTENT

INTRODUCTION TO THE SUBJECT

"The fundamental task of communication is the exact or approximate reproduction at some point of a message selected at another point."

Claude Shannon, 1948.

Information theory is a branch of applied mathematics, radio engineering (signal processing theory) and computer science that relates to the measurement of the quantity of information, its properties and establishes limit relations for data transmission systems.

Consider the general scheme of data transmission (*Figure 1*), and briefly describe each block of the presented scheme.

Source coding is used to minimise the number of bits per unit time required to represent the source output data. This process is known as source coding or data compression. Examples are Huffman coding, Lempel-Ziv algorithm.

Encryption is used to secure the transmission of the source bits. The process of converting the source bits (message) into a stream of meaningless-looking bits (ciphertext) is called encryption. Examples: Data Encryption Standard (DES), RSA.

Channel coding is used to correct errors introduced by the transmission medium. The process consists of introducing a number of redundant bits into the sequence according to a given rule to correct errors that occur. Example: repetition codes, Hamming codes, Reed-Muller codes, cyclic codes, CRC codes.

Modulation - the process of converting a digital signal into an analogue signal for transmission over a physical channel. Examples: PSK, QAM.

Channel - the physical medium in which data is transmitted. During transmission, data may be distorted due to various effects: noise, interference, signal attenuation. Examples: binary channel (with erasure), channel with white noise added.

Demodulation, channel decoding, decryption and source decoding are the reverse procedures for modulation, channel coding, encryption and source coding, respectively.

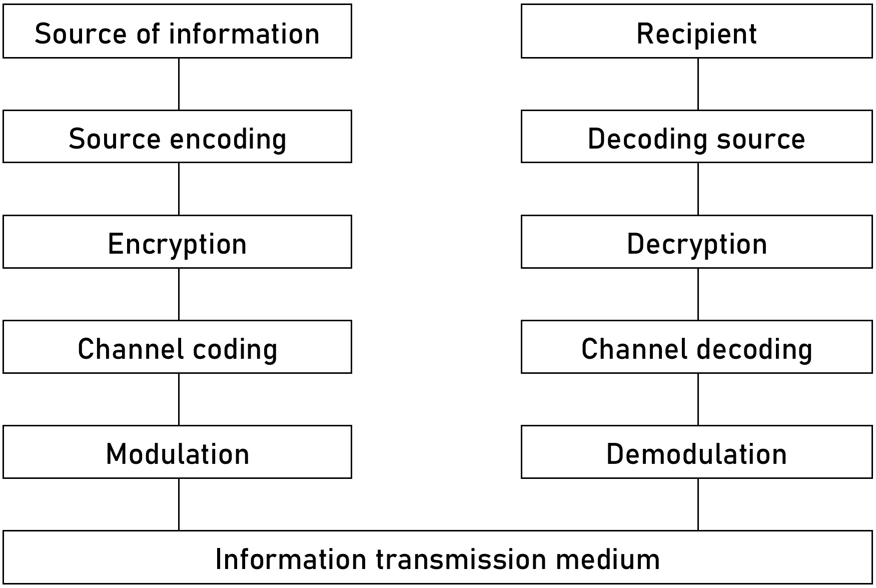

Figure 1. Data transmission scheme

Actual problems of information theory are:

- Estimation of source and channel characteristics (entropy and quantity of information).
- Data interpretation (Bayesian inference).
- Data compression (efficient coding).
- Error correction (noise tolerant coding).

TOPIC 1
ENSEMBLES AND PROBABILITIES. BAYESIAN INFERENCE

An ensemble *X* is a triple (x, A_x, P_x), where an outcome *x* is the value of some random variable taking one of a set of possible values $A_X = \{a_1, a_2, ..., a_i, ..., a_I\}$ with probabilities $P_x = \{p_1, p_2, ..., p_i, ..., p_I\}$.

$$P(x = a_i) = p_i\,,\ \ p_i \geq 0\,,\ \sum_{a_i \in A_x} P(x = a_i) = 1.$$

Probability of a subset.

If *T* is a subset of A_X, then

$$P(T) = P(x \in T) = \sum_{a_i \in T} P(x = a_i).$$

A joint ensemble *XY* is an ensemble, each outcome of which is an ordered pair *x,y* each outcome is an ordered pair *x,y*, in which

$$x \in A_x = \{a_1, a_2, ..., a_I\}\,,\ \text{a}\ \ y \in A_y = \{b_1, b_2, ..., b_J\}\,.$$

The probability *P(x, y)* is called the joint probability of *x* and *y*. In such a notation the comma is optional, so *P(x, y)* and *P(xy)* are the same thing. Note that the random variables quantities *x* and *y* included in the ensemble *XY* may not be independent.

The probabilities of the individual quantities *P(x)* and *P(y)* included in the ensemble are defined through the joint probabilities of *P(x)* and *P(y)* of the ensemble are defined through joint probabilities as

$$P(x = a_i) = \sum_{b_j \in A_Y} P(x = a_i, y = b_j),$$

$$P(y) = \sum_{y \in A_Y} P(x, y)\,.$$

The probability that *x* equals a_i given that $y = b_j$ is called the conditional probability and is denoted and defined as follows:

$$P(x = a_i \mid y = b_j) = \frac{P(x = a_i, y = b_j)}{P(y = b_j)}$$

$$P(y = b_j) \neq 0$$

Multiplication rule:

$$P(x,y\mid H)=P(x\mid y,H)P(y\mid H)=P(y\mid x,H)P(x\mid H).$$

Summation rule:

$$P(x\mid H)=\sum_y P(x,y\mid H)=\sum_y P(x\mid y,H)P(y\mid H).$$

Bayes' theorem:

$$P(y\mid x,H)=\frac{P(x\mid y,H)P(y\mid H)}{P(x\mid H)}=\frac{P(x\mid y,H)P(y\mid H)}{\sum_{y'} P(x\mid y',H)P(y'\mid H)}.$$

Independence.

Two random variables *X* and *Y* are independent if and only if

$$P(x,y)=P(x)P(y).$$

Example:

A spam filter works with 95% reliability, that is, spam messages are filtered with 95% probability and non-spam messages with 5% probability.

On average, 25% of incoming messages are spam.

What is the is the probability that the filtered message does not contain spam?

Solution:

Let $a = 1$ – the message contains spam, $a = 0$ – it does not.

The result of filtering $b = 1$ – the message is filtered, $b = 0$ – it is not.

Then

$$P(b=1\mid a=1)=0{,}95, \qquad P(b=1\mid a=0)=0{,}05,$$
$$P(b=0\mid a=1)=0{,}05, \qquad P(b=0\mid a=0)=0{,}95.$$

A priori probability of having no spam

$$P(a=1)=0{,}25, \qquad P(a=0)=0{,}75$$

What is the overall probability of the message being filtered?

$$P(b=1)=P(b=1\,|\,a=1)P(a=1)+P(b=1\,|\,a=0).$$

By Bayes formula, the probability that the filtered message did not contain spam:

$$P(a=0\,|\,b=1)=\frac{P(b=1\,|\,a=0)P(a=0)}{P(b=1\,|\,a=1)P(a=1)+P(b=1\,|\,a=0)}=$$

$$=\frac{0,05\cdot 0,75}{0,95\cdot 0,25+0,05\cdot 0,75}=0,1(36).$$

The mathematical expectation of a random variable.

Discrete case:

$$MX=\sum_i x_i p_i\ .$$

Continuous case:

$$MX=\int_{x=-\infty}^{+\infty} xf(x)\,dx\ .$$

The variance of a random variable.

Discrete case:

$$DX=\sum_i (x_i-MX)^2\,p_i\ .$$

Continuous case:

$$DX=\int_{x=-\infty}^{+\infty}(x-MX)^2\,f(x)\,dx\ ,$$

$$DX=MX^2-(MX)^2.$$

Discrete distribution laws

Bernoulli (parameter *p*) – describes success (or failure) in a single trial:

$$p(x)=\begin{cases} p, & k=1, \\ 1-p, & k=0, \end{cases}$$

$$MX = p,$$

$$DX = p(1-p).$$

Binomial law of distribution (parameters *p* and *n*) – describes the number of successes in *n* independent Bernoulli trials.

$$p(k)=C_n^k p^k (1-p)^{n-k},\ k=0,\dots,n,$$

$$MX = np,$$

$$DX = np(1-p).$$

The geometric distribution (parameter *p*) is the number of attempts before the first success

$$p(k)=(1-p)^{k-1} p, \qquad k=1,\dots,n,\dots$$

$$MX = \frac{1}{p},$$

$$DX = \frac{1-p}{p^2}.$$

Continuous distribution laws

Uniform distribution law on the interval [a,b]

$$f(x)=\begin{cases} 0, & x<a \\ \dfrac{1}{b-a}, & a \le x \le b, \\ 0, & x>b, \end{cases} \qquad \begin{aligned} MX &= \frac{a+b}{2}, \\ DX &= \frac{(b-2)^2}{12}. \end{aligned}$$

Exponential:

$$f(x)=\begin{cases}\lambda e^{-\lambda x}, & x\ge 0,\\ 0, & x<0,\end{cases}\qquad \begin{aligned}MX&=\frac{1}{\lambda},\\ DX&=\frac{1}{\lambda^2}.\end{aligned}$$

Normal:

$$f(x)=\frac{1}{\sqrt{2\pi\sigma}}e^{-\frac{(x-\mu)^2}{2\sigma^2}},\qquad \begin{aligned}MX&=\mu,\\ DX&=\sigma^2.\end{aligned}$$

Bayesian inference

Bayesian formula:

$$P(\lambda\mid\{x_1,\dots,x_N\})=\frac{P(\{x_1,\dots,x_N\}\mid\lambda)P(\lambda)}{P(\{x_1,\dots,x_N\})}.$$

Three problems can be solved based on the observation $\{x_1,\dots,x_N\}$:

- Estimation of the source distribution;

- prediction of the outcome;

- comparison of hypotheses.

Task 1.

Two people left traces of their blood at the scene of the crime. According to the results of the blood test of the suspect Oliver, the blood type 1 was determined. In the traces at the crime scene, the blood found was of two groups: the first (the most common, observed in 60% of the population) and the fourth (rare, 1% of the population).

Do the results obtained (blood types one and four) favour Oliver's presence at the crime scene or vice versa?

Solution.

Denote D by the data, S by the assumption of "the suspect and one other unknown person were present at the crime scene" and $\overline{S}$ is "two unknown people were present at the crime scene."

$$P(D\,|\,S,H) = p_4,$$

$$P(D\,|\,\bar{S},H) = 2p_1p_4,$$

$$\frac{P(D\,|\,S,H)}{P(D\,|\,\bar{S},H)} = \frac{1}{2p_1} = \frac{1}{1{,}2} \approx 0{,}83\ .$$

It is possible to solve the previous problem in the general case for *n₁* blood samples of blood group 1 and *n₄* of blood group 4. Distribution of blood groups *p₁* and *p₄*.

$$P(n_1,n_4\,|\,S) = \frac{(N-1)!}{(n_1-1)!n_4!}p_1^{n_1-1}p_4^{n_4}\ ,$$

$$P(n_1,n_4\,|\,\bar{S}) = \frac{(N)!}{n_1!n_4!}p_1^{n_1}p_4^{n_4}\ ,$$

$$\frac{P(n_1,n_4\,|\,S)}{P(n_1,n_4\,|\,\bar{S})} = \frac{\dfrac{(N-1)!}{(n_1-1)!n_4!}p_1^{n_1-1}p_4^{n_4}}{\dfrac{(N)!}{n_1!n_4!}p_1^{n_1}p_4^{n_4}} = \frac{n_1\,/\,N}{p_1}\ .$$

Task 2.

A curved coin is tossed F times. Observed a sequence of s eagles and tails, denoted by a and b, respectively. It is required to find out how unequal the results of the throws are and to predict the probability that the next throw will be an eagle.

Assume that the a priori distribution is uniform, and obtain the posterior distribution by multiplying it by the likelihood ratio.

Solution.

Let H_1 be our assumption. Given p_a, the probability that F throws will form a sequence **s** containing $\{F_a\ , F_b\}$ eagles and tails is equal to

$$P(\mathbf{s}\,|\,p_a,F,H_1) = p_a^{F_a}(1-p_a)^{F_b}\ .$$

For example,

$$P(\mathbf{s}=\{a,a,b,a\}\mid p_a,F=4,H_1)=p_a p_a(1-p_a)p_a\,.$$

In the first model, a uniform distribution is assumed for p_a

$$P(p_a\mid H_1)=1,\quad p_a\in[0,1],$$

and

$$p_b\equiv 1-p_a\,.$$

Estimation of unknown parameters

Given a string of length F in which F_a is the number of eagles and F_b is the number of tails, it is estimated (a) what p_a might be, (b) predict whether the next symbol will be a or b. Predictions are usually expressed as probabilities, so "predicting whether the next symbol will be a "is the same as computing the probability that the next symbol will be a).

Suppose that H_1 is true. The posterior probability p_a, given a string of length F that contains $\{F_a, F_b\}$ eagles and tails, is by Bayes' theorem equal to

$$P(p_a\mid \mathbf{s},F,H_1)=\frac{P(\mathbf{s}\mid p_a,F,H_1)P(p_a\mid H_1)}{P(\mathbf{s}\mid F,H_1)}\,.$$

The multiplier $P(\mathbf{s}\mid p_a, F, H_1)$, which is a function of p_a and is known as the likelihood function, and the a priori probability $P(p_a\mid H_1)$ are defined by us earlier. Thus, our estimate of p_a is

$$P(p_a\mid \mathbf{s},F,H_1)=\frac{p_a^{F_a}(1-p_a)^{F_b}}{P(\mathbf{s}\mid F,H_1)}\,.$$

The normalizing constant is given by the beta integral

$$P(\mathbf{s}\mid F,H_1)=\int_0^1 \mathrm{d}\,p_a^{F_a}(1-p_a)^{F_b}=\frac{\Gamma(F_a+1)\Gamma(F_b+1)}{\Gamma(F_a+F_b+2)}=\frac{F_a!F_b!}{(F_a+F_b+1)!}\,.$$

From estimation to prediction.

Our assumption about the next throw - the probability that its outcome is a - is obtained by integrating over p_a.

This is the result of the uncertainty of p_a under our assumptions. By the rule of summation

$$P(a|\mathbf{s},F)=\int \mathrm{d}p_a P(a|p_a)P(p_a|\mathbf{s},F).$$

The probability of occurrence of a at p_a is p_a, therefore

$$P(a|\mathbf{s},F)=\int \mathrm{d}p_a p_a \frac{p_a^{F_a}(1-p_a)^{F_b}}{P(\mathbf{s}|F)}=$$

$$=\int \mathrm{d}p_a \frac{p_a^{F_a+1}(1-p_a)^{F_b}}{P(\mathbf{s}|F)}=$$

$$=\left[\frac{(F_a+1)!F_b!}{(F_a+F_b+2)!}\right]/\left[\frac{F_a!F_b!}{(F_a+F_b+1)!}\right]=$$

$$=\frac{F_a+1}{F_a+F_b+2}$$

– an expression known as **Laplace's law.**

Model comparison

Suppose that one scientist has suggested a different theory for our data. He insists that the source is not a bent coin, but a perfectly formed cube with six faces, one of which has an eagle and the other five have tails. Thus, the parameter p_a, which can take any value from 0 to 1 in the original model, is strictly equal to 1/ 6 in the new H_0 hypothesis.

How can the two models be compared using data? It is necessary to assess how likely the model H_1 compared to H_0.

To compare the models, write down Bayes' theorem, but this time with a different variable in the left-hand side. It is necessary to find out how likely hypothesis H_1 is given the available data.

According to Bayes' theorem

$$P(H_1|\mathbf{s},F)=\frac{P(\mathbf{s}|F,H_1)P(H_1)}{P(\mathbf{s}|F)}.$$

Similarly, the posterior probability H_0 is equal to

$$P(H_0 \mid \mathbf{s}, F) = \frac{P(\mathbf{s} \mid F, H_0) P(H_0)}{P(\mathbf{s} \mid F)}.$$

The normalizing constant in both cases $P(\mathbf{s} \mid F)$ represents the overall probability of obtaining the observed data. If H_1 and H_0 are the only models under consideration, this probability is determined by the sum rule

$$P(\mathbf{s} \mid F) = P(\mathbf{s} \mid F, H_1) P(H_1) + P(\mathbf{s} \mid F, H_0) P(H_0).$$

To calculate the posterior probabilities of hypotheses it is necessary to set their a priori probabilities $P(H_1)$ and $P(H_0)$. In our case, set them equal to ½. It is necessary to compute the data-dependent values $P(\mathbf{s}|F,H_1)$ and $P(\mathbf{s}|F,H_0)$. Could be called these values.

The value $P(\mathbf{s}|F,H_1)$ is a measure of how well the data fit H_1, and call this value the evidence of the H_1 model. This value has been calculated previously.

The proof of model H_0 is much simpler to compute, since the model has no estimated parameters. Setting p_0 equal to 1/6, obtain

$$P(\mathbf{s} \mid F, H_0) = p_0^{F_a} (1 - p_0)^{F_b}.$$

Thus, the ratio of the posterior probabilities of model H_1 to model H_0 is equal to

$$\frac{P(H_1 \mid \mathbf{s}, F)}{P(H_0 \mid \mathbf{s}, F)} = \frac{P(\mathbf{s} \mid F, H_1) P(H_1)}{P(\mathbf{s} \mid F, H_0) P(H_0)} =$$

$$= \left[\frac{F_a! F_b!}{(F_a + F_b + 1)!} \right] / \, p_0^{F_a} (1 - p_0)^{F_b}.$$

Some values of this ratio are given in *Table 1*.

Generally, with small sample sizes, the probabilities of the models are not too different from each other, but the larger the data, the larger this ratio can be.

Table 1. Ratio of posterior probabilities of models

F_a	F_b	$P(a \mid \mathbf{s}, F)$	$\frac{P(H_1 \mid \mathbf{s}, F)}{P(H_0 \mid \mathbf{s}, F)}$	$\frac{P(H_0 \mid \mathbf{s}, F)}{P(H_1 \mid \mathbf{s}, F)}$
1	1	0,5	1,200	
2	1	0,6	3,600	
5	1	0,75	222,171	
10	1	0,846	549692,509	
1	2	0,4		1,389
1	5	0,25		2,813
1	20	0,087		2,008
4	4	0,5	4,266	
2	6	0,25		3,198
6	2	0,75	4886,156	

Test questions on the topic

1. Definition of Ensemble of Signals.
2. Definition of Joint Ensemble of Signals.
3. Analytical expression of Joint probability of signals.
4. Meaning of numerical characteristics of ensembles of signals under discrete distribution laws.
5. The value of numerical characteristics of ensembles of signals at continuous laws of distribution.
6. Bayes' theorem for comparison of two alternative hypotheses.

TOPIC 2
ENTROPY

The concept of entropy

It is necessary to define quantify the information that is obtained by observing an of an event occurring with a given probability.

Let a discrete probabilistic ensemble $\{Z, p(z)\}$ with N possible states and a set of probability distributions on it $p(z_i)$such that for all

$$i=\overline{1,N} \;\; p(z_i)\geq 0\,, \sum p(z_i)=1:$$

$$Z=\begin{bmatrix} z_1, z_2, \ldots, z_i, \ldots, z_N \\ p_1, p_2, \ldots, p_i, \ldots, p_N \end{bmatrix}, \qquad (1)$$

The first measure introduced to determine the amount of information obtained by observing some discrete set was a measure proposed by ***Hartley***:

$$I(Z)=\log_a N\,, \qquad (2)$$

where N is the number of possible outcomes. The base a determines the unit of information, for example, if a is 2, the unit of information will be *bit*, at *3 – trit*, at *e – nat*, at *10 – dit.*

Note that an important condition for the occurrence of information on a Hartley is the existence of several possible outcomes. Obviously, if the outcome of an event is guaranteed, don't get information from observation.

What is the drawback of Hartley's measure? Hartley's measure does not take into account the fact that the probabilities p_i in (1) may be different. Therefore, it is used only in the case of equal probability events of the set. For unequal probability events, the uncertainty is smaller. For example, the uncertainty of choice in the case of two elements with a priori probabilities of 0.9 and 0.1 is smaller than in the case of equally probable elements (0.5; 0.5). Therefore, it is natural to require that the uncertainty measure be a continuous function of the probabilities p_i of N elements.

A measure of information satisfying this requirement is proposed by ***K. Shannon*** and is called entropy:

$$H(Z) = -\sum_{i=1}^{N} p(z_i)\log_a p(z_i). \tag{3}$$

The most widely used binary unit of information is the ***bit***, which will be used below.

For independently realizable elements of a set, a priori private uncertainty can be used as a priori private uncertainty can be used as a measure:

$$H(z_i) = -\log_2 p(z_i). \tag{4}$$

It is easy to see that the K. Shannon measure (3), which characterizes the uncertainty of the source as a whole, is obtained by averaging the of partial uncertainties (4) over all elements of the set.

Show the connection between the ***K. Shannon*** measure and the ***R. Hartley*** measure. If all elements of a set are equally probable, p_i=1/N for all $i = \overline{1, N}$, then

$$H(Z) = -\sum_{i=1}^{N} \frac{1}{N}\log_2 \frac{1}{N} = \log_2 N. \tag{5}$$

Thus, R. Hartley's measure is a special case of the measure of K. Shannon's measure for equal probability elements. It can also be shown that C. Shannon's measure is a generalization of Hartley's measure to the case of unequal probability elements.

Properties of entropy

1. *Entropy is real and non-negative*. The property is easily verified by formula (3) taking into account that $\mathbf{0 \le p(z_i) \le 1}$ for all $i = \overline{1, N}$.

2. *Entropy is a finite quantity*. At $\mathbf{0 \le p_i \le 1}$ this property follows directly from formula (5). At **p=0** have:

$$\lim_{p\to 0}(-p\log_2 p) = \lim_{p\to 0}\frac{\log_2 \frac{1}{p}}{\frac{1}{p}} = \lim_{\alpha\to\infty}\frac{\log_2 \alpha}{\alpha} = \lim_{\alpha\to\infty}\frac{\log_2 e}{\alpha\cdot 1} = 0$$

(here substitute *1/p* = *α* and further disclose the uncertainty by Lopital's rule). Thus, for any values

$$0 \le p_i \le 1,\ i = \overline{1,N} \quad H(Z) < \infty .$$

3. In the course of the proof of Property 2, it is easy to see that *H(Z)=0*, if the probability of one of the elements of the set is 1.

4. Entropy is *maximal* when all elements of the set are equal probable and

$$H_{\max}(Z) = \max_{\forall p_i} H(Z) = \log_2 N . \qquad (6)$$

It will be searched for the maximum in (3) under the condition $\sum p_i = 1$. Lagrangian function for the corresponding unconditional extremum problem

$$F(p,\lambda) = -\sum_{p=1}^{N} p_i \log_2 p_i + \lambda\left(\sum_{i=1}^{N} p_i - 1\right) \rightarrow extr .$$

Necessary conditions of extremum:

$$\frac{\partial F(p,\lambda)}{\partial p_i} = -\log_2 p_i - \log_2 e + \lambda = 0 ,$$

$$\frac{\partial F(p,\lambda)}{\partial \lambda} = \sum_{i=1}^{N} p_i - 1 = 0 ,$$

it follows that

$$p_i = 2^{\lambda - \log_2 e} = Const = 1/N .$$

It is easy to verify that a given value provides the maximum.

5. In the particular case of a set with two elements, the dependence of entropy on the probability of one of the elements has the form shown in *Figure 2*. This can be verified by applying the relations and conclusions obtained when considering properties 2 and 3 to relation (3), which in this case has the following form

$$H(Z) = -p\log_2 p - (1-p)\log_2(1-p) . \qquad (7)$$

In conclusion, emphasized that entropy characterizes only the average uncertainty of choosing one element from a set, completely ignoring their content.

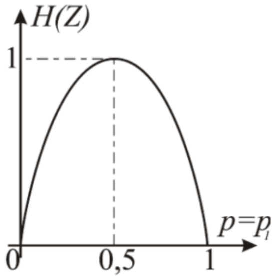

Figure 2. Entropy change in the case of two elements

The concept of conditional entropy

Now consider the case when two sets are given: $Z=\{z_1, z_2, ..., z_N\}$ and $V=\{v_1, v_2, ..., v_K\}$, between whose elements there are relations. The product of sets $\{ZV\}$ is called is the set whose elements represent all possible ordered pairs of products $z_i v_j$, $i = \overline{1,N}, j = \overline{1,K}$. If each pair z_i, v_j is assigned a probability $p(z_i, v_j)$, then have the product of ensembles $\{ZV, p(zv)\}$. The elements of the combined ensemble have the usual properties of probabilities:

$$\sum_{j=1}^{K} p(z_i, v_j) = p(z_i), \quad \sum_{i=1}^{N} p(z_i, v_j) = p(v_j). \tag{8}$$

In particular, it follows from these properties that if a product of ensembles is given, then the original ensembles are $\{Z, p(z)\}$ and $\{V, p(v)\}$. The converse is possible only in the case, when the elements of the initial ensembles are independent, in which case

$$p(z_i, v_j) = p(z_i)\, p(v_j).$$

Note that since in this case

$$\log_2 p(z_i, v_j) = \log_2 p(z_i) + \log_2 p(v_j)$$

have

$$H(ZV) = -\sum_{i=1}^{N}\sum_{j=1}^{K} p(z_i, v_j)\log_2 p(z_i, v_j) =$$

$$= -\sum_{i=1}^{N}\sum_{j=1}^{K} p(z_i)p(v_j)\log_2\left[p(z_i)p(v_j)\right] =$$

$$= -\sum_{i=1}^{N} p(z_i)\log_2 p(z_i)\underbrace{\sum_{j=1}^{K} p(v_j)}_{1} - \sum_{j=1}^{K} p(v_j)\log_2 p(v_j)\underbrace{\sum_{i=1}^{N} p(z_i)}_{1} =$$

$$= H(Z) + H(V).$$

Similarly, formulas for combining any number of independent sources can be obtained.
In the general case for dependent ensembles

$$p(z_i, v_j) = p(z_i)p(v_j / z_i) = p(v_j)p(z_i / v_j),$$

that is, to determine the probability of occurrence of an element of the combined ensemble, it is necessary to specify the conditional probability of occurrence of an element of one of the ensembles under the condition that the realizable element of the other ensemble:

$$p(z_i / v_j) = \frac{p(z_i, v_j)}{p(v_j)}, \qquad p(v_j / z_i) = \frac{p(z_i, v_j)}{p(z_i)}.$$

Let the joint ensemble $\{ZV\}$ be given by the probability matrix of all its possible elements z_iv_j, $i = \overline{1,N}$, $j = \overline{1,K}$:

$$\begin{bmatrix} p(z_1,v_1) & p(z_2,v_1) & \dots & p(z_N,v_1) \\ p(z_1,v_2) & p(z_2,v_2) & \dots & p(z_N,v_2) \\ \dots & \dots & \dots & \dots \\ p(z_1,v_K) & p(z_2,v_K) & \dots & p(z_N,v_K) \end{bmatrix}. \qquad (9)$$

By summing up the row and column probabilities of (9) according to according to (8) can also define the ensembles $\{Z, p(z)\}$ and $\{V, p(v)\}$:

$$\{Z, p(z)\} = \begin{bmatrix} z_1 & z_2 & \dots & z_N \\ p(z_1) & p(z_2) & \dots & p(z_N) \end{bmatrix},$$

$$\{V, p(v)\} = \begin{bmatrix} v_1 & v_2 & \dots & v_K \\ p(v_1) & p(v_2) & \dots & p(v_K) \end{bmatrix}.$$

Since in the case of dependent elements

$$p(z_i, v_j) = p(z_i)\,p(v_j/z_i) = p(v_i)\,p(z_i/v_j), \tag{10}$$

using the first equality mentioned in (10) can write

$$\begin{aligned} H(ZV) &= -\sum_{ij} p(z_i, v_j)\log_2 p(z_i, v_j) = \\ &= -\sum_i p(z_i)\log_2 p(z_i)\sum_j p(v_j/z_i) - \\ &\quad -\sum_i p(z_i)\sum_j p(v_j/z_i)\log_2 p(v_j/z_i). \end{aligned} \tag{11}$$

By the normalization condition

$$\sum_j p(v_j/z_i) = 1$$

for any $i = \overline{1, N}$, so the first summand in the right-hand side is the entropy $H(Z)$ of the ensemble $\{Z, p(z)\}$. The second summand (by j) in the second summand characterizes the partial uncertainty attributable to one state of the ensemble *V*, provided that the state z_i of the ensemble *Z*. It is called the private conditional entropy and is denoted by:

$$H_{z_i}(V) = -\sum_{j=1}^{K} p(v_j/z_i)\log_2 p(v_j/z_i). \tag{12}$$

The value $H_Z(V)$, obtained by averaging the partial conditional entropy over all elements z_i

$$H_Z(V) = \sum_{i=1}^{N} p(z_i)\,H_{z_i}(V), \tag{13}$$

is called total conditional entropy or simply conditional entropy. Thus, (11), taking into account (12), (13), can be written as

$$H(ZV)=H(Z)+H_Z(V). \tag{14}$$

Using the second equality in (12), can write by analogy:

$$H(ZV)=H(V)+H_V(Z). \tag{15}$$

It can also be shown that in the case of union of any number of sets {*ZVW*...} with dependent elements, the equality is satisfied

$$H(ZVW...)=H(Z)+H_Z(V)+H_{ZV}(W)+... \quad .$$

Eemphasized that conditional entropy is always less than or equal to unconditional entropy:

$$H_V(Z)\le H(Z),\;\; H_Z(V)\le H(V). \tag{16}$$

The fairness of inequality (16) is intuitive: the uncertainty of choosing an element from some set can decrease only if an element of another set is known, with elements of which there is a connection. elements of which there is a connection. elements of which there is a connection. From (14)-(16), in particular, it follows that

$$H(ZV)\le H(Z)+H(V). \tag{17}$$

There is often another type of relationship, namely statistical dependence between elements of a sequence. If a relationship exists only between two neighboring elements of a sequence, it is characterized by a conditional probability $p(z_i / z_j)$. A sequence of elements possessing the above property is called a single-link Markov chain. The connection of each element with two previous elements is characterized by the conditional probability $p(z_i / z_j z_k)$, and the corresponding sequence is called a two-link Markov chain.

For a single-connected Markov chain under the assumption that a known (accepted) element z_j from the alphabet of volume N, partial conditional entropy

$$H(Z/z_j)=-\sum_{i=1}^{N}p(z_i/z_j)\log_2 p(z_i/z_j).$$

At that, the total (average) conditional entropy is defined as

$$H(Z)=-\sum_{j=1}^{N}p(z_j)\sum_{i=1}^{N}p(z_i/z_j)\log_2 p(z_i/z_j). \tag{18}$$

Similarly, for a two-connected Markov chain

$$H\left(Z / z_j z_k\right) = -\sum_{i=1}^{N} p(z_i / z_j z_k)\log_2 p(z_i / z_j z_k)\,,$$

$$H(Z) = -\sum_{j,k} p(z_j, z_k)\sum_{i} p(z_i / z_j z_k)\log_2 p(z_i / z_j z_k)\,. \qquad (19)$$

It is possible to construct expressions for entropy also at a more connection between the elements of the sequence.

The concept of differential entropy

Proceed to the consideration of information sources whose output signals are continuous random variables.
The set of possible states of such a source is a continuum, and the probability of any particular value is 0, which makes it impossible to apply, for example, measure (3). Construct uncertainty measures for such sources on the basis of the previously introduced measures for discrete ensembles.
The uncertainty of choosing any value of a continuous random variable according to formula (3) can be approximated by limiting the range of its permissible values and dividing this range, for example, into equal intervals, the probability of falling into each of which is different from zero and is determined as follows

$$\mathrm{P}\left\{z_i \le Z < z_i + \Delta z\right\} \cong w\left(z_i^*\right)\Delta z\,.$$

Here $w\left(z_i^*\right)$is the ordinate of the distribution density $w\left(z\right)$ of a continuous random variable at the value z_i^* belonging to the interval $\left[z_i, z_i + \Delta z\right]$.

Replacing in (3) $w\left(z_i\right)$ by its approximate value $w\left(z_i^*\right)$ have

$$H\left(Z\right) = -\sum_{i=1}^{N} w\left(z_i^*\right)\Delta z\log_2\left(w\left(z_i^*\right)\Delta z\right) =$$
$$= -\sum_{i=1}^{N} w\left(z_i^*\right)\log_2 w\left(z_i^*\right)\Delta z - \log_2 \Delta z\sum_{i=1}^{N} w\left(z_i^*\right)\Delta z. \qquad (20)$$

Then carry out the limit transition at $\Delta z \to 0$. In this case, the sum becomes an integral, $\Delta z \to dz$, and

$$\sum_{i=1}^{N} w(z_i^*)\Delta z \to 1\,.$$

Considering that, in general, the range of variation of the continuous

$(-\infty;+\infty)$, obtain

$$H(Z) = -\int_{-\infty}^{+\infty} w(z)\log_2 w(z)dz - \lim_{\Delta z\to 0}\log_2 \Delta z\ . \tag{21}$$

It follows from formula (21) that the entropy of a continuous random variable is equal to infinity irrespective of the type of density The entropy of a continuous random variable is equal to infinity irrespective of the type of probability density. This fact, generally speaking, is not surprising, since the probability of a particular value of a continuous signal is 0, and the set of states is infinite. Obviously, it is not possible to use such a measure in practice.
To obtain a finite characterization of the information property, it is possible to use only the first term, called differential entropy:

$$h(Z) = -\int_{-\infty}^{+\infty} w(z)\log_2 w(z)dz\ . \tag{22}$$

The term differential entropy is related to the fact that for its definition in formula (22) the differential law of distribution is used $p(z)$. A natural question arises: is not this convention artificial and meaningless.
It turns out that differential entropy has the meaning of the average uncertainty of the choice of a random variable with an arbitrary law of distribution minus the uncertainty of a random variable uniformly distributed in a unit interval of the interval.
Indeed, the entropy (2.2) of the random variable Z_r uniformly distributed on the interval δ is defined as

$$H(Z_r) = -\int_{-\infty}^{\infty} \frac{1}{\delta}\log_2 \frac{1}{\delta} dz - \lim_{\Delta z\to 0}\log_2 \Delta z_r\ .$$

If δ=1

$$H(Z_r) = -\lim_{\Delta z\to 0}\log_2 \Delta z_r\ . \tag{23}$$

Comparing (22) and (23) it is easy to see that at $\Delta z = \Delta z_r$

$$H(Z) - H(Z_r) = h(z)\ . \tag{24}$$

The concept of differential conditional entropy

Now consider the situation when two (hereafter two) continuous random variables are statistically related. As before, divide ranges of admissible values of random variables into equal intervals so that

$$P\{z_i \le Z < z_i + \Delta z, \quad v_j \le V < v_j + \Delta v\} \cong w(z_i^*, v_j^*) \cdot \Delta z \Delta v\,, \qquad (25)$$

where $w\left(z_{i,}^{*} v_j^{*} \right)$ is the ordinate of the two-dimensional density distribution at the point $\left(z_{i,}^{*} v_j^{*} \right)$belonging to the rectangle with sides Δz, Δv:

$$(z_i \le z_i^* < z_i + \Delta z, \qquad v_j \le v_j^* < v_j + \Delta v)\ .$$

Substituting approximate values of probabilities into the formula for entropy (3) obtain

$$H(Z,V) = -\sum_i \sum_j w(z_i^*, v_j^*) \log_2 w(z_i^*, v_j^*) \Delta z \Delta v -$$
$$-\log_2 \Delta z \sum_i \sum_j w(z_i^*, v_j^*) \Delta z \Delta v - \log_2 \Delta v \sum_i \sum_j w(z_i^*, v_j^*) \Delta z \Delta v.$$

Due to the fact that

$$w(z_i^*, v_j^*) = w(z_i^*) w(v_j^* / z_i^*)$$

first summand in of the right-hand side of the last equality can be represented as the sum of

$$-\sum_i w(z_i^*) \log_2 w(z_i^*) \Delta z \sum_j w(v_j^* / z_i^*) \Delta v - \sum_i \sum_j w(z_i^*, v_j^*) \log_2 w(v_j^* / z_i^*) \Delta v \Delta z\ .$$

Further carrying out of the limiting transition at $\Delta z \to 0$, $\Delta v \to 0$ taking into account that according to the normalization condition

$$\lim_{\substack{\Delta z \to 0 \\ \Delta v \to 0}} \sum_i \sum_j w(z_i^*, v_j^*) \Delta z \Delta v = 1\,,$$

$$\lim_{\Delta v \to 0} \sum_i \sum_j w(v_j^* / z_i^*) \Delta v = 1\,,$$

$$\lim_{\Delta z \to 0} \sum_i \sum_j w(z_i^*) \Delta z = 1\,,$$

get

$$H(Z,V) = -\int_{-\infty}^{\infty} w(z) \log_2 w(z) dz - \int_{-\infty}^{\infty} \int_{-\infty}^{\infty} w(z,v) \log_2 w(v/z) dz dv - \qquad (26)$$
$$- \lim_{\Delta z \to 0} \log_2 \Delta z \quad - \lim_{\Delta v \to 0} \log_2 \Delta v.$$

The first and third summands are the entropy *H(Z)* of a continuous source (22), whose output signal is a random quantity *Z*, and the value

$$H_Z(V) = -\int_{-\infty}^{\infty}\int_{-\infty}^{\infty} w(z,v)\log_2 w(v/z)dzdv - \lim_{\Delta v \to 0}\log_2 \Delta v$$

is the conditional entropy of a continuous random variable. As one would expect, it is equal to infinity because of the second summand in the right-hand side is equal to infinity. Therefore, as in the case of a single independent source, only the first summand:

$$h_Z(V) = -\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} w(z,v)\log_2 \frac{w(z,v)}{w(z)} dzdv\,. \tag{27}$$

The value (27) is called conditional differential entropy.

Conditional differential entropy characterizes the average uncertainty of the choice of a continuous random variable with an arbitrary distribution law. At the same time, the results of realization of another, statistically related to it, continuous random variable are known, minus the average uncertainty of the choice of the random variable, having uniform distribution on the unit interval.

The differential entropy of two continuous statistically related sources can be represented in the form of related sources can be represented as

$$h(ZV) = h(Z) + h_Z(V) = h(V) + h_V(Z)\,. \tag{28}$$

The second equality is obtained by the same scheme as in first one, at

$$w(z_i^*, v_j^*) = w(v_j^*)w(z_i^* / v_j^*)\,.$$

Note also that for continuous sources one can write equations for discrete messages $H(ZV) = H(Z) + H_Z(V) = H(V) + H_V(Z)$, but they have only theoretical value, since in practice it is impossible to work with infinite uncertainties.

Properties of differential entropy

Differential entropy, unlike the entropy of a discrete source, is a relative measure of uncertainty because its values depend on the scale of the continuous quantity.

Suppose that the continuous random variable *Z* has changed by a factor of *k*. Subject to the normalization condition:

$$\int_{-\infty}^{+\infty} w(kz)d(kz) = k\int_{-\infty}^{+\infty} w(kz)dz = 1,$$

the following relation for the densities of the initial and scaled quantities takes place

$$w(kz) = \frac{w(z)}{k}. \tag{29}$$

Taking into account (29) in accordance with (22) have

$$\begin{aligned} h(kZ) &= -\int_{-\infty}^{+\infty} w(kz)\cdot\log_2 w(kz)\cdot d(kz) = \\ &= -\int_{-\infty}^{+\infty} w(z)\left[\log_2 w(z) - \log_2 k\right]dz = \\ &= -\int_{-\infty}^{+\infty} w(z)\log_2 w(z)dz + \log_2 k\int_{-\infty}^{+\infty} w(z)dz = h(Z) + \log_2 k. \end{aligned} \tag{30}$$

It follows from (30) that due to the choice of different *k* differential entropy can take positive, negative and zero values.

The differential entropy does not depend on the shift parameter $\Theta = Const$, $h(Z+\Theta) = h(Z)$. Indeed, using the substitution $V = Z + \Theta$, at which the limits of integration do not change, and $dz = dv$ have:

$$\begin{aligned} h(Z+\Theta) &= -\int_{-\infty}^{+\infty} w(z+\Theta)\log_2 w(z+\Theta)dz = \\ &= -\int_{-\infty}^{+\infty} w(v)\log_2 w(v)dv = h(V). \end{aligned}$$

Distributions with maximum differential entropy

Formulate the following task.
Determine the density *p(z)*, providing the maximum value of the functional

$$h(Z) = -\int_{\alpha}^{\beta} w(z)\log_2 w(z)dz\ ,$$

in limiting

$$\int_{\alpha}^{\beta} w(z)dz = 1 .$$

The Lagrangian function in the above (isoperimetric) task has the form

$$F(w,\mu) = w(z)\log_2 w(z) + \mu \cdot w(z) , \tag{31}$$

where μ, in this case a constant, indefinite multiplier Lagrangian. The necessary conditions of extremum (31) are given by the relation

$$\frac{\partial F(w,\mu)}{\partial w} = \log_2 w(z) + \log_2 e + \mu = 0 . \tag{32}$$

The desired density

$$w(z) = 1/(\beta - \alpha), \quad \alpha \le z \le \beta$$

is obtained as a result of the joint solution of (31), (32). This means that if the only constraint for a random variable is region of possible values: $Z \in [\alpha, \beta]$, then the maximal differential entropy possesses a uniform distribution of probabilities in this region.

Now remove the restriction on the region of possible values, but add a restriction on the variance value:

$$h(Z) = -\int_{-\infty}^{\infty} w(z)\log_2 w(z)dz \to мax , \tag{33}$$

When

$$\int_{-\infty}^{\infty} w(z)dz = 1 , \tag{34}$$

$$\int_{-\infty}^{\infty} z^2 w(z)dz = \sigma^2 . \tag{35}$$

The Lagrangian function in this case takes the form

$$F(w,\mu_1,\mu_2) = w(z)\log_2 w(z) + \mu_1 \cdot w(z) + \mu_2 z^2 w(z) ,$$

and the corresponding Euler equation

$$\frac{\partial F(w,\mu)}{\partial p} = \log_2 w(z) + \log_2 e + \mu_1 + \mu_2 z^2 = 0 . \tag{36}$$

By direct substitution can verify that the Gaussian density

$$w(z) = \frac{1}{\sqrt{2\pi}\sigma}\exp\left\{-\frac{z^2}{2\sigma^2}\right\}$$

satisfies the necessary condition (36) of extremum (in this case maximum) of the functional (33) and the given isoperimetric constraints (34), (35). Note that in derivation, for simplicity, have taken the mathematical expectation equal to zero, since the differential entropy does not depend on the shift parameter anyway.

Test questions on the topic

1. Entropy and its properties.
2. Conditional entropy and its properties.
3. For which ensembles does the concept of conditional entropy make sense?
4. What is the conditional entropy of an association of statistically independent sources?
5. Conditional differential entropy and its properties.
6. Quantity of information as a measure of removed uncertainty.

TOPIC 3
QUANTITY OF INFORMATION

The quantity of information in the transmission of one element of a discrete message

Suppose that there is a discrete source of information characterized by a discrete probabilistic ensemble:

$$Z = \begin{bmatrix} z_1 & z_2 & \cdots & z_N \\ p(z_1) & p(z_2) & \cdots & p(z_N) \end{bmatrix},$$

where z_i, $i = \overline{1, N}$, – its possible states. Each state of the source can be matched with a separate primary signal. Some given set of primary signals, coming from the output of the information source to the input of the communication channel is called a message, and z_i is called a message element.

If the source states are realized independently of each other, then the partial a priori uncertainty of occurrence at the input of the channel message element z_i is defined as

$$H(z_i) = -\log_2 p(z_i).$$

Suppose that there is no statistical relationship between the interference and the message elements and the conditional probability is known that v_j is received instead of z_i:

$$p(z_i / v_j), \quad i = \overline{1, N}, \quad j = \overline{1, K}.$$

Thus, if an element v_j is received at the output of the channel, the posterior probability $p(z_i / v_j)$ becomes known.
Consequently, can define the posterior partial uncertainty:

$$H_{v_j}(z_i) = -\log_2 p(z_i / v_j).$$

The partial amount of information obtained as a result of the element v_j becoming known is defined as the difference of a priori and posterior uncertainties:

$$I\left(z_i,v_j\right)=H\left(z_i\right)-H_{v_j}\left(z_i\right)=$$
$$=-\log_2 p\left(z_i\right)+\log_2 p\left(z_i/v_j\right)=\log_2\frac{p\left(z_i/v_j\right)}{p\left(z_i\right)}. \quad (37)$$

Thus, the private quantity of information is equal to the amount of uncertainty that is removed as a result of receiving the message element v_j.

Properties of the private quantity of information

1. The private quantity of information decreases with increasing a priori probability $p\left(z_i\right)$, increases with increasing posterior probability $p\left(z_i/v_j\right)$. Depending on the ratio between them can be positive, negative and zero.

2. If $p\left(z_i/v_j\right)=p\left(z_i\right)$, then according to (37) $I\left(z_i,v_j\right)=0$.

3. In the absence of interference, the private quantity of information is equal to the private a priori uncertainty of the element z_i:

$$I(z_i,v_j)=H(z_i)=-\log_2 p(z_i)$$

since at this

$$H_{v_j}\left(z_i\right)=0\,.$$

4. The partial amount of information about z_i, contained in v_j, is equal to the partial amount of information about v_j contained in z_i. Indeed:

$$I(z_i,v_j)=\log_2\frac{p\left(z_i/v_j\right)}{p(z_i)}=\log_2\frac{p(v_j)p\left(z_i/v_j\right)}{p(v_j)p(z_i)}=$$
$$\log_2\frac{p(z_i)p\left(v_j/z_i\right)}{p(z_i)p(v_j)}=\log_2\frac{p\left(v_j/z_i\right)}{p(v_j)}=I(v_j,z_i).$$

The average quantity of information in any element discrete message

The a priori uncertainty in the average per element of the of the message is characterized by entropy:

$$H(Z)=-\sum_{i=1}^{N}p(z_i)\cdot\log_2 p(z_i)\,, \quad (38)$$

and the posterior uncertainty is the conditional entropy:

$$H_V(Z) = -\sum_{j=1}^{K} p(v_j) \sum_{i=1}^{N} p\left(z_i / v_j\right) \log_2 p\left(z_i / v_j\right). \qquad (39)$$

According to (38), (39) by analogy with the private quantity of information quantity on average per one element message is defined as

$$I(Z,V) = H(Z) - H_V(Z) =$$
$$= -\sum_i p(z_i) \log_2 p(z_i) + \sum_j p(v_j) \sum_i p\left(z_i / v_j\right) \log_2 p\left(z_i / v_j\right).$$

In the last equality nothing will change if the first summand in the right-hand side is multiplied by

$$\sum_{j=1}^{K} p\left(v_j / z_i\right) = 1.$$

Then, with given that

$$\sum_i p(z_i) \sum_j p\left(v_j / z_i\right) = \sum_j p(v_j) \sum_i p\left(z_i / v_j\right) = \sum_{ij} p\left(z_i, v_j\right)$$

and using the properties of the logarithm, the formula for the amount of information on average for one element of the message can be written as in the form of

$$I(Z,V) = \sum_{ij} p(z_i, v_j) \log_2 \frac{p\left(z_i / v_j\right)}{p(z_i)} = \sum_{ij} p(z_i, v_j) \log_2 \frac{p\left(z_i, v_j\right)}{p(z_i) p(v_j)} \qquad (40)$$

Further, unless the private nature of the quantity of information is specifically stated, the quantity of information on mean of one element of the message will always be implied (40).

Properties of the average quantity of information in an element messages

1. Non-negativity. $I(Z,V) \geq 0$, so always $H(Z) \geq H_V(Z)$.

2. $I(Z,V) = 0$ in the absence of a statistical relationship between *Z* and *V*, since in this case $H(Z) = H_V(Z)$.

3. $I(Z,V) = I(V,Z)$, that is, the quantity of information in *V* with respect to *Z* is equal to the quantity of information in *Z* with respect to *V*. Indeed

$$I(Z,V)-I(V,Z)=H(Z)-H_V(Z)-(H(V)-H_Z(V))=$$
$$=H(Z)+H_Z(V)-(H(V)+H_V(Z))=H(Z,V)-H(V,Z)=0.$$

4. If there is no interference $I(Z,V)=H(Z)$, since in this case $H_V(Z)$=0. It's the maximum amount of information can be received from a source.

Quantity of information when transmitting messages from a continuous source

The ratio for the quantity of information from a continuous source is obtained from formula (40) for the discrete case.
Denoting the transmitted and received continuous signals *Z* and *V*, respectively. Divide the region of admissible values of these signals into equal intervals and write down the approximate probabilities (*Figure 3*):

$$\mathrm{P}\left\{z_i \le Z < z_i+\Delta z, v_j \le V < v_j+\Delta v\right\} \cong w\left(z_i^*, v_j^*\right)\Delta z\Delta v\ ,$$

where $w\left(z^*,v^*\right)$ -ordinate of the two-dimensional distribution density $w(z,v)$ at some point belonging to the rectangle with number *i,j*.

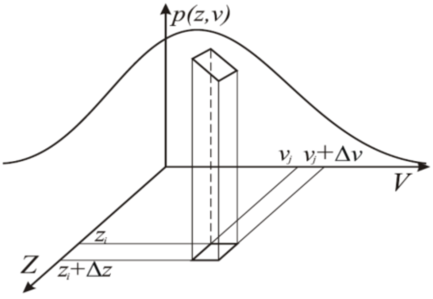

Figure 3. Discretization of the area *Z,V*

For corresponding to a given two-dimensional density $w(z,v)$ one-dimensional densities $w(z_i)$, $w(v_j)$, by analogy with the way it was done at obtaining the relation for the differential entropy, can write

$$\mathrm{P}\left\{z_i \le Z < z_i + \Delta z\right\} \cong w\left(z_i^*\right)\Delta z\,,$$
$$\mathrm{P}\left\{v_j \le V < v_j + \Delta v\right\} \cong w\left(v_j^*\right)\Delta v\,,$$

where $w\left(z_i^*\right)$, $w\left(v_j^*\right)$ – ordinates of one-dimensional densities for values z_i^* and v_j^*, taken in the intervals $[z_i, z_i+\Delta z]$ and $[v_j, v_j+\Delta v]$ respectively.

Replacing $w\left(z_i, v_j\right)$, $w\left(z_i\right)$, $w\left(v_j\right)$ by their approximate values $w\left(z_i^*, v_j^*\right)\Delta z\Delta v$, $w\left(z_i^*\right)\Delta z$, $w\left(v_j^*\right)\Delta v$ respectively, can write down

$$I(Z,V) = \sum_i \sum_j w(z_i^*, v_j^*)\Delta z\Delta v \cdot \log_2 \frac{w(z_i^*, v_j^*)}{w(z_i^*)p(v_j^*)}\,. \qquad (41)$$

Performing the limit transition in (41) at $\Delta z \to 0$, $\Delta v \to 0$ obtain:

$$\begin{aligned} I(Z,V) &= \lim_{\substack{\Delta z \to 0 \\ \Delta v \to 0}} \sum_i \sum_j w^*(z_i, v_j)\log_2 \frac{w^*(z_i, v_j)}{w^*(z_i)p^*(v_j)}\Delta z\Delta v = \\ &= \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} w(z,v)\log_2 \frac{w(z,v)}{w(z)w(v)}dzdv. \end{aligned} \qquad (42)$$

The formula (42) can also be obtained by using the notion of differential entropy. Indeed, by analogy with discrete case, define the quantity of information as the difference between a priori and a posteriori (in this case differential) entropy:

$$\begin{aligned} I(Z,V) &= h(Z) - h_v(Z) = \\ &= -\int_{-\infty}^{\infty} w(z)\log_2 w(z)dz + \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} w(z,v)\log_2 w\left(z/v\right)dzdv. \end{aligned} \qquad (43)$$

In (43) nothing changes if the first summand in the right-hand side multiplied by

$$\int_{-\infty}^{\infty} w(v/z)dv = 1\,.$$

Then, given that

$$w(z,v) = w(v)w(z/v) = \quad = w(z)w(v/z)\,,$$

relation (43) can be rewritten in the following form:

$$I(Z,V) = h(Z) - h_v(Z) = \int_{-\infty}^{\infty}\int_{-\infty}^{\infty} w(z,v)\log_2 \frac{w(z,v)}{w(z)w(v)} dzdv \,. \qquad (44)$$

Since *I(Z,V)* in (44) is defined as the difference $h(Z) - h_v(Z)$, the quantity of information at transmission from a continuous source, unlike differential entropy, does not depend on the scale of the random variable. Note that the relationship between the concepts of entropy and quantity of information for a continuous source of information is similar to the relationship between potential, defined as the work of transferring a charge from infinity to a given point of a field, and voltage, defined as the potential difference, which is considered in physics.

Test questions on the topic

1. Describe the quantitative assessment of information.
2. Explain the relationship between the Shannon and Hartley measures.
3. Partial quantity of information and its properties.
4. The quantity of information when transmitting messages from a continuous and discrete source.
5. Determine the entropy of a message with the following probabilities of occurrence of symbols: p(z1) = p(z2) = p(z3) = p(z4) = 0.01, p(z5) = 0.96.

1. 6.Why does the quantity of information transmitted from a continuous source, unlike differential entropy, no longer depend on the scale of the random variable?

TOPIC 4
DATA CHANNELS

Discrete channel without memory.

A discrete channel without memory is characterized by the following values:

- input alphabet A,

- output alphabet B,

- conditional probabilities $P_{Y|X}\left(\cdot \mid x\right)$ for all $x \in A$

$$P\left(y_n \mid x_1,\ldots,x_{n-1},y_1,\ldots,y_{n-1}\right) = P_{Y|X}\left(y_n \mid x_n\right).$$

From the presented expression can see that the probability of occurrence of the value y_n on the output for a given input x_n and the previous values of the output, $y_n, \ldots, y_{n-1}$ depends only on the current value of the input x_n. Such a channel is called a memoryless channel because the output values do not depend on the previous values of the output. And depends only on the current input value. A discrete channel without memory and without feedback is described by the following probability

$$P\left(x_n \mid x_1,\ldots,x_{n-1},y_1,\ldots,y_{n-1}\right) = P\left(x_n \mid x_1,\ldots,x_{n-1}\right).$$

Since there is no feedback, the input values x_n are independent of the previous input values , $x_n, \ldots, x_{n-1}$.

Theorem.

For a discrete channel without memory without feedback

$$P\left(y_1,\ldots,y_n \mid x_1,\ldots,x_n\right) = \prod_{i=1}^{n} P_{Y|X}\left(y_i \mid x_i\right), \quad n = 1,2,\ldots.$$

Proof:

Set the joint probability $P\left(x_1,\ldots,x_n,y_1,\ldots,y_n\right)$. This probability can be written as

$$P(x_1,...,x_n,y_1,...,y_n)=P(x_1,...,x_n,y_1,...,y_{n-1})\cdot P(y_n\,|\,x_1,...,x_n,y_1,...,y_{n-1})=$$
$$=P(x_1,...,x_{n-1},y_1,...,y_{n-1})\cdot P(x_n\,|\,x_1,...,x_{n-1},y_1,...,y_{n-1})\cdot$$
$$\cdot P(y_n\,|\,x_1,...,x_n,y_1,...,y_{n-1})=$$
$$=\prod_{i=1}^{n}P(x_i\,|\,x_1,...,x_{i-1},y_1,...,y_{n-1})\cdot P(y_i\,|\,x_1,...,x_i,y_1,...,y_{i-1}).$$

From the definition of a discrete channel with no memory and no feedback, the multipliers under the product sign can be written as

$$P(x_1,...,x_n,y_1,...,y_n)=\prod_{i=1}^{n}P(x_i\,|\,x_1,...,x_{i-1})\cdot P_{Y|X}(y_i\,|\,x_1).$$

This expression can be further simplified by splitting the product into two

$$P(x_1,...,x_n,y_1,...,y_n)=\left[\prod_{i=1}^{n}P(x_i\,|\,x_1,...,x_{i-1})\right]\cdot\left[\prod_{i=1}^{n}P_{Y|X}(y_i\,|\,x_1)\right].$$

The first part of this expression is equal to $P(x_1,...,x_n)$.

$$P(x_1,...,x_n,y_1,...,y_n)=P(x_1,...,x_n)\cdot\left[\prod_{i=1}^{n}P_{Y|X}(y_i\,|\,x_1)\right].$$

Dividing both parts of the expression by $P(x_1,...,x_n)$ and using the definition of conditional probability, obtain:

$$\frac{P(x_1,...,x_n,y_1,...,y_n)}{P(x_1,...,x_n)}=\prod_{i=1}^{n}P_{Y|X}(y_i\,|\,x_1),$$

$$P(y_1,...,y_n\,|\,x_1,...,x_n)=\prod_{i=1}^{n}P_{Y|X}(y_i\,|\,x_1).$$

The ***memoryless channel capacity*** is defined as the maximum of the average mutual information *I (X,Y)*, which can be obtained by selecting *P(x):*

$$C=\max_{P_x}I(X,Y).$$

This is the maximum of the mutual information between the input and output of the channel, given that the maximization is done over all possible distributions of the input P_x. Fully equivalently, from the definition of the quantity of information, the capacity can be written as

$$C = \max_{P_x}\left[H(X) - H(Y \mid X)\right].$$

Uniformly dispersing channel

Discrete channel without memory have K possible values at the input and J at the output. Call the channel uniformly dispersive (*Figure 4*) if the transition probabilities, ordered in in decreasing order are the same for each of K inputs.

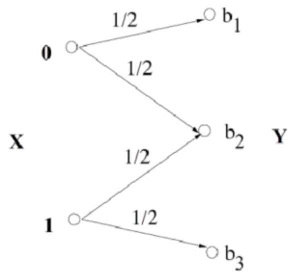

Figure 4. Uniformly dispersing channel

Theorem.

Regardless of the choice of the distribution P_x for UDC

$$H(Y \mid X) = -\sum_{j=1}^{J} p_j \log p_j \ .$$

where $p_1, p_2, \ldots, p_j$ are the transition probabilities.

Proof:

From the definition of UDC, know that the conditional entropy of an output symbol Y for a given input $X=a_k$ is equal to

$$H(Y \mid X = a_k) = -\sum_{j=1}^{J} p_j \log p_j, \quad k = 1, \ldots, K$$

From the definition of $H\ (Y|X)$ it follows that

$$H(Y\mid X)=\sum_{k=1}^{K}P(a_k)H(Y\mid X=a_k).$$

Since *H (Y|X=a$_k$)* is constant regardless of *a$_k$* due to the UDC, the expression takes the form

$$H(Y\mid X)=H(Y\mid X=a_k)\sum_{k=1}^{K}P(a_k)=H(Y\mid X=a_k).$$

Therefore

$$H(Y\mid X)=-\sum_{j=1}^{J}p_j\log p_j\ .$$

Thus

$$C=\max_{P_x}H(Y)+\sum_{j=1}^{J}p_j\log p_j\ .$$

Uniformly focussing channel

Consider a discrete channel without memory with *K* input values and *J* output values. Call this channel uniformly focussing (*Figure 5*) if the transition probabilities ordered in descending order are the same for each of the *J* outputs.

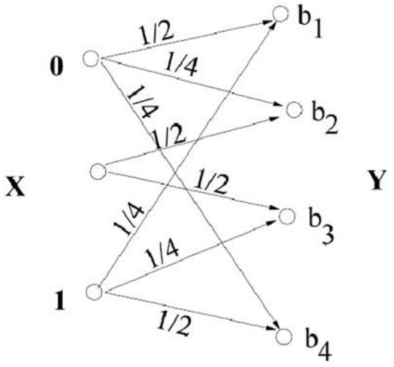

Figure 5. Uniformly focussing channel

Theorem.

In a uniformly focussing channel, uniform probabilities of input values lead to uniform distributions of probabilities of outputs. In this case

$$\max_{P_x} H(Y) = \log J .$$

Proof:

Since the input distribution *P(x)* is uniformly distributed with *K* possible values, can write down following equality

$$P(y) = \sum_x P(y \mid x) P(x) = \frac{1}{K} \sum_x P(y \mid x) .$$

The summands *P(y|x)* correspond to the transition probabilities to each value of *y* from all possible inputs. From the definition of a uniformly focussed channel, *P(y|x)* are constant for all values of *y*. Hence, the sum in the right-hand side will be the same for all *J* values of *y*, and hence *Y* has a uniform distribution. Since its distribution is uniform, based on the entropy property, obtain:

$$\max_{P_x} H(Y) = \log J .$$

Strongly symmetrical channel

A discrete memoryless channel that is both uniformly dispersing and uniformly focusing is called a strongly symmetric channel (*Figure 6*). A binary symmetric channel is a symmetric channel in in which the number of inputs and outputs is equal to two.

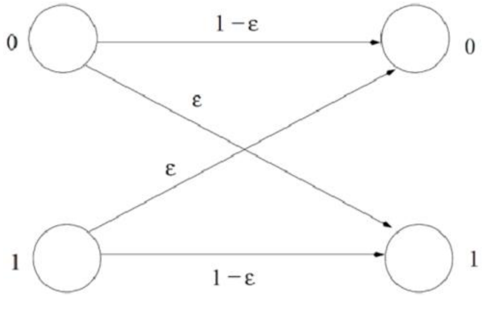

Figure 6. Binary strongly symmetric channel

The capacity of a strongly symmetric channel is equal to

$$C = \max_{P_x} H(Y) + \sum_{j=1}^{J} p_j \log p_j \ .$$

From the definition of a uniformly focusing channel

$$\max_{P_x} H(Y) = \log J \ .$$

Combining these expressions, obtain the capacity of the strongly symmetric channel

$$C = \log J + \sum_{j=1}^{J} p_j \log p_j \ .$$

For a binary symmetric channel J=2 the capacity is equal to

$$C = 1 - h(\varepsilon) \ .$$

Symmetrical channel

Features of the symmetrical channel (*Figure 7*):

- All composite channels have the same input alphabet, the input values of each channel are selected from the same symbols;
- All composite channels have different non-overlapping output alphabets;
- *X* and *Z* are statistically independent, that is, the input symbol value does not affect the composite channel selection;
- The probabilities of channel selection are $q_1, q_2, ..., q_L$.

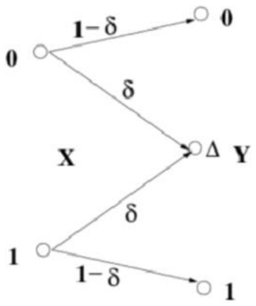

Figure 7. Symmetrical channel

Theorem:

For a symmetric channel without memory satisfying the following requirements

$$I(X,Y)=\sum_{i=1}^{L} I\left(X,Y^{(i)}\right)q_i \ .$$

Proof:

The joint entropy of *Y* and *Z* can be written as

$$H(Y,Z)=H(Y)+H(Z\,|\,Y)=H(Z)+H(Y\,|\,Z).$$

It is clear from the diagram that for a given *Y*, when the output value is selected, the value of *Z* is also known, therefore

$$H(Z\,|\,Y)=0$$

$$H(Y,Z)=H(Y)=H(Z)+H(Y\,|\,Z).$$

Then the uncertainty *Y* can be written as

$$H(Y)=H(Y,Z)=H(Z)+H(Y\,|\,Z).$$

From the definition of conditional entropy, can write its as follows

$$H(Y)=H(Z)+H(Y\,|\,Z)=H(Z)+\sum_{i=1}^{L} H(Y\,|\,Z=i)P_Z(i)=$$

$$=H(Z)+\sum_{i=1}^{L} H\left(Y^{(i)}\right)q_i.$$

The latter expression follows from the fact that, given a chosen *Z,* the uncertainty of *Y* is related to the uncertainty of its choice in the corresponding channel. Similarly, can write the conditional entropy *H(YZ\X)* and similarly *H(Z\XY)* is zero, since for a known *Y* is exactly known *Z*.

$$H(Y,Z\,|\,X)=H(Y\,|\,X)=H(Z\,|\,X)+H(Y\,|\,XZ).$$

In addition, *Z* and *X* are statistically independent, so

$$H(Y\,|\,X)=H(Z)+H(Y\,|\,XZ).$$

Expanding the notation *H(Y\XZ)*, obtain

$$H(Y|X)=H(Z)+\sum_{i=1}^{L}H(Y|X,Z=i)P_Z(i)=H(Z)+\sum_{i=1}^{L}H\left(Y^{(i)}|X\right)q_i.$$

Now can define the quantity of information as

$$I(X,Y)=H(Y)-H(Y|X).$$

Substituting the corresponding expressions for *H(Y)* and *H(Y\X),* obtain

$$I(X,Y)=\sum_{i=1}^{L}\left[H\left(Y^{(i)}\right)-H\left(Y^{(i)}|X\right)\right]q_i\ .$$

And, by the definition of the amount of information, the expression in brackets becomes

$$I(X,Y)=\sum_{i=1}^{L}I\left(X,Y^{(i)}\right)q_i\ .$$

Continuous Gaussian channel

One of the most important continuous channels is the Gaussian channel. Consider the capacity of the Gaussian channel and prove the achievability of this capacity.

The Gaussian channel is usually represented as a sum of a useful signal *X* and noise *Z*. It is assumed that the distributions of the signal and noise are independent. The capacity of a Gaussian channel with constraints on the input signal power *P* and noise variance *N* is given as

$$C=\max_{MX^2\le P}I(X,Y)=\frac{1}{2}\log\left(1+\frac{P}{N}\right),$$

where the maximum is reached when *X ~ N(0,K)*.

Proof:

The capacity is given by the expression

$$C=\max_{p(x):MX^2\le P}I(X,Y),$$

where maximization is performed over all possible distributions of the input signal *p(x)*.

From the definition of the quantity of information can write

$$\begin{aligned} I(X,Y) &= h(Y) - h(Y \mid X) = \\ &= h(Y) - h(X+Z \mid X) \\ &= h(Y) - h(Z \mid X) \qquad (*) \\ &= h(Y) - h(Z), \qquad (**) \end{aligned}$$

* – for a given *X*, the uncertainty of *X*+*Z* is equal to uncertainty of *Z*,

** – since *X* and *Z* are independent.

As it is known, *Z* is normally distributed, and since the differential entropy *Z ~ N(0,K)* is calculated as

$$h(Z) = \frac{1}{2}\log_2(2\pi e N).$$

And it's also known that

$$MY^2 = M(X+Z)^2 = MX^2 + 2 \cdot MX \cdot MZ + MZ^2 = P + N.$$

The differential entropy *Y* with variance *P*+*N* is bounded from above by the differential entropy of a Gaussian random variable

$$\begin{aligned} I(X,Y) &= h(Y) - h(Z) \\ &\le \frac{1}{2}\log_2(2\pi e(P+N)) - \frac{1}{2}\log_2(2\pi e N) \\ &= \frac{1}{2}\log_2\left(1 + \frac{P}{N}\right). \end{aligned}$$

The maximum is reached at *X ~ N(0,P)* when *Y* is a Gaussian random variable and the equality is satisfied.

The capacity of a Gaussian channel with constraint on the signal power and noise variance equal to

$$C = \frac{1}{2}\log_2\left(1 + \frac{P}{N}\right)$$

bits for a single transmission is achievable.

Test questions on the topic

1. What are the characteristics of data transmission channels?
2. What channel is called continuous, discrete and discrete-continuous?
3. What does the term " without memory" mean?
4. Capacity for discrete data transmission channels.
5. Capacity for continuous data channels.
6. Gaussian channel capacity.

TOPIC 5
SYMBOLIC CODES. PREFIX CODES

Variable length coding. Prefix codes

In this lecture will start a new topic on efficient source coding, also known as compression.

In general, all of the codes discussed can be used in any number system with an integer greater than one. Since the most common number system is the number system with base two, binary codes will be considered.

First discuss variable length codes that encode one source symbol at a time, instead of encoding strings of N source symbols. These codes are lossless. They guarantee error-free compression and recovery, but there is a chance that the encoded string is longer than the source string. The idea that enables compression, in general, is to assign shorter sequences of symbols to more likely outcomes and longer sequences to less likely outcomes.

Consider three basic requirements for a useful code.

- Firstly, any encoded string must be unambiguously decodable.
- Secondly, the symbol code must be simple to decoding.
- Thirdly, the code must provide the maximum possible as much compression as possible.

Examples of symbol codes are shown in Table 2.

Table 2: Examples of symbol codes

Symbol	Code 1	Code 2	Code 3	Code 4	Code 5
a	00	00	0	1	0
b	00	01	1	10	10
c	01	10	11	100	110
d	11	11	100	1000	111

Any encoded string must be unambiguously decodable.

A code is called unambiguously decodable if any finite sequence of the code corresponds to no more than one message. Codes 2, 4, 5 in the table are examples of unambiguously decodable code.

A symbolic code must be simple to decode.

The most simple for decoding the code in which the end of the codeword can be detected at the same time with reception of the corresponding symbol. This can occur when no codeword is a prefix of another codeword.

A binary sequence z is a prefix of another binary sequence z' if z is of length n and the first n symbols of z' are exactly constitute the sequence z.

A symbolic code in which no codeword is a prefix for another codeword is called a prefix code. A prefix code is also known as "instantly decodable" or "self-separable" because the encoded string can be decoded from left to right without receiving subsequent codewords. The end of the codeword is detected immediately.

Prefix codes are unambiguously decodable.

Codes 2 and 5 given in the table are prefix codes.

To decode a prefix code a binary tree is constructed (*Figure 9*). Below is an example of such tree for binary prefix code {011, 10, 11, 11} code {011, 10, 11, 00}

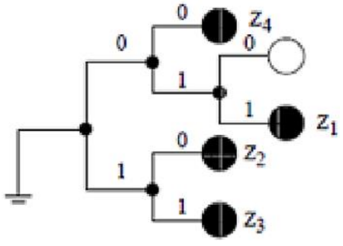

Figure 9. Binary tree for prefix code

The code should provide as much compression as possible

Medium length *L(X)* symbolic for ensemble *X*

$$L(X)=\sum_{x}(P(x)l(x)).$$

Example.

Let the set of symbols of the alphabet correspond to the probabilities and symbolic codes given in Table 4.

Table 4. Symbolic code with probabilities

Symbol	Probabilities	Code
a	1/2	0
b	1/4	10
c	1/8	110
d	1/8	111

The entropy *H (X) =1,75,* and the expected length *L(X)* is also 1.75.

The sequence ***acdbac*** is encoded as ***0110111100100110***.

Find the encoding limit for uniquely decodable codes.

First, consider the code {00, 01, 10, 11}.

- Reduce the length of one codeword 00 $\rightarrow$ 0. Uniquely decodability can be restored only by increasing the lengths of other codewords.
- Add a new codeword: 110. Uniquely decodability can be restored only by increasing the length of one of the codewords

Kraft's inequality

A binary prefix code which contains codewords with lengths equal to $l_1, l_2, \dots, l_K$ (positive integers) exists if and only if

$$\sum_{i=1}^{K}(2^{-l_i}) \le 1 .$$

Proof: There is

$$S = \sum_i D^{-l_i} .$$

Look at the number

$$S^N = \left[\sum_i 2^{-l_i}\right]^N = \sum_{i_i=1}^{I}\sum_{i_2=1}^{I}\dots\sum_{i_N=1}^{I} 2^{-\left(l_{i_1}+l_{i_1}+\dots+l_{i_N}\right)} .$$

The value of the sum in the degree exponent $\left(l_{i1} + l_{i2} + \ldots + l_{iN}\right)$ is equal to the length of the encoded string **x**= $a_{i1}a_{i2}\ldots a_{iN}$. For each string **x** of length *N* it contains one summand. Introduce an array A_l , which counts how many strings **x** are encoded with length *l*. Then

$$l_{\min} = \min_i l_i \qquad l_{\max} = \max_i l_i$$

$$S^N = \sum_{l=Nl_{\min}}^{Nl_{\max}} 2^{-l} A_l \; .$$

Now assume that the code is uniquely decodable. Consider **x** with code length *l*. There are only 2^l distinct bit strings of length *l*, so the inequality $A \leq 2^l$ must be satisfied. Thus

$$S^N = \sum_{l=Nl_{\min}}^{Nl_{\max}} 2^{-l} A_l \leq \sum_{l=Nl_{\min}}^{Nl_{\max}} 1 \leq Nl_{\max} \; .$$

Hence, $S^N \leq Nl_{max}$ for all *N*. Now, if *S* is greater than 1, then as *N* increases, S^N grows exponentially and for large enough *N* the exponent always exceeds a polynomial such that Nl_{max}. But our result ($S^N \leq Nl_{max}$) must be true for any *N*. Hence, $S \leq 1$.

For binary code, Kraft's inequality is of the form

$$\sum_{i=1}^{K} (2^{-l_i}) \leq 1 \, .$$

Best achievable compression

The expected length of a uniquely decodable code is limited from below by entropy *H(X).*

Proof: Define the probabilities

$$q_i \equiv 2^{-l_i} / z \, ,$$

where

$$z = \sum_{i'} 2^{-l_{i'}} \; ,$$

thus

$$l_i = \log \frac{1}{q_i} - \log z \ .$$

Using Gibbs inequality

$$\sum_i p_i \log \frac{1}{q_i} \geq \sum_i p_i \log \frac{1}{p_i}$$

with equality only at $q_i = p_i$ and Kraft's inequality:

$$L(X) = \sum_i p_i l_i = \sum_i p_i \log \frac{1}{q_i} - \log z \geq$$
$$\geq \sum_i p_i \log \frac{1}{p_i} - \log z \geq H(X).$$

The inequality *L(X) = H (X)* holds if and only if Kraft's inequality becomes the equality *z =1* and the lengths of codewords words satisfy

$$l_i = \log \frac{1}{p_i}$$

Thus, it is impossible to compress information below entropy. How close can one get to entropy?

Source coding theorem

For an ensemble *X*, there exists a prefix code *C* with average length satisfying the inequality:

$$L(C,X) \in [H(X), H(X)+1]$$

Proof:

Set the codeword lengths equal to integers slightly larger than the optimal lengths

$$l_i = \left\lceil \log_2 \frac{1}{p_i} \right\rceil,$$

where $[l^*]$ denotes the smallest integer greater than or equal to l^*.

Check that there exists a prefix code with such lengths, by checking Kraft's inequality.

$$\sum_i 2^{-l_i} = \sum_i 2^{-\lceil \log_2(1/p_i) \rceil} \le \sum_i 2^{-\log_2(1/p_i)} = \sum_i p_i = 1 .$$

Then be confirmed

$$L(X) = \sum_i 2^{-p_i \lceil \log_2(1/p_i) \rceil} < \sum_i 2^{-p_i(\log_2(1/p_i)+1)} = H(X) + 1 .$$

Test questions on the topic

1. Shannon's basic theorem for interference-free channel coding
2. Typical classification of optimal coding methods
3. Potential capabilities of continuous communication channels in transmission
4. Theorem on coding in a continuous channel with interference.
5. Fixed and variable length codes.
6. The concept of optimal (effective) coding.

TOPIC 6
SHANNON-FANO CODING. HUFFMAN CODING. ARITHMETIC CODING

Optimal Shannon coding

In Shannon coding, symbols are arranged in order from most probable to least probable.

They are assigned codes by taking the first $l_i = \left[log_2\, p_i \right]$ digits from the binary decomposition of the cumulative probability. of the cumulative probability

$$\sum_{k=1}^{i-1} p_i \ .$$

An example of coding is given in Table 5. Shannon coding provides *H(X)+2*.

Table 5. Shannon coding

a_i	$p(a_i)$	l_i	Sum of p_i to $i-1$	Sum by $p(a_i)$	Final code
a_1	0,37	2	0,0	0,0000	00
a_2	0,18	3	0,37	0,0101	010
a_3	0,18	3	0,55	0,1000	100
a_4	0,12	4	0,73	0,1011	1011
a_5	0,09	4	0,85	0,1101	1101
a_6	0,06	4	0,94	0,1111	1111

Shannon-Fano optimal coding

The code is constructed as follows.

1. The coded signs are written out in a table in descending order of their probabilities in messages.
2. Then they are divided into two groups so that the values of probability sums in each group are close.
3. All signs of one of the groups in the corresponding digit are coded, for example, by one, while the signs of the second group are coded by zero.
4. Each group obtained in the process of division is subjected to the above described operation until, as a result of the next division, one sign remains in each group.

An example of coding is given in Table 6.

Table 6: Shannon-Fano coding

a_i	$p(a_i)$	Coding process				Final code
a_1	0,37	0	0			00
a_2	0,18		1			01
a_3	0,18	1	0			10
a_4	0,12		1	0		110
a_5	0,09			1	0	1110
a_6	0,06				1	1111

Huffman optimal coding

1. Take the two least probable symbols in the alphabet. These two symbols will get maximum length codewords differing by the last symbol.

2. Combine the two symbols into one, repeat 1.

3. The coded signs, as well as when using the Shannon-Fano method, are arranged in descending order of their probabilities (Table 7).

4. Further, at each stage, the last two positions of the list are replaced by one and it is assigned a probability equal to the sum of probabilities of the replaced positions.

5. The list is then re-sorted in descending order of probability, retaining information about which signs were combined at each step.

6. The process continues until there remains a single position with probability equal to 1.

7. A code tree is then constructed. A node with probability equal to 1 is assigned to the root of the tree.

Table 7. Huffman coding

a_i	$p(a_i)$	Building a tree probabilities					Final code
a_1	0,37	0,37	0,37	0,37	0,63	1	0
a_2	0,18	0,18	0,27	0,36	0,37		111
a_3	0,18	0,18	0,18	0,27			110
a_4	0,12	0,15	0,18				100
a_5	0,09	0,12					1011
a_6	0,06						1010

8. Then to each node are assigned two descendants with probabilities that participated in the formation of the probability value of the processed node.

9. This continues until the nodes corresponding to the probabilities of the original signs are reached.

The coding process on the code tree is carried out as follows.

1. one of the branches coming out of each node, for example, with a higher probability, is put in correspondence with the symbol 1, and with a lower - 0.

2. Descending from the root to the desired sign gives the code for that sign. The coding rule in the case of equal probabilities

is specified.

Table 7 and Figure 10 illustrate the application of the Huffman methodology.

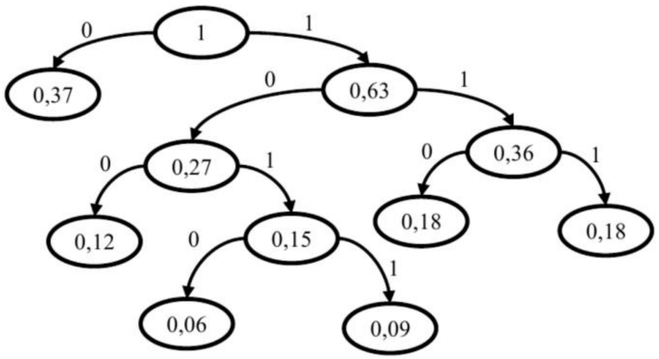

Figure 10. Binary tree of the Huffman code

Block coding

If the value of the average number of characters per sign is much higher than the entropy, it indicates redundancy of the code. This redundancy can be eliminated by switching to block coding. Consider a simple example of encoding two characters z_1, z_2 with probabilities of their occurrence in messages 0.1 and 0.9 respectively.

If one of these characters is encoded, for example, by zero and the other by one (one character per character), have, respectively

$$l_{cp} = 0{,}1 \cdot 1 + 0{,}9 \cdot 1 = 1{,}0, \qquad H(z) = -0{,}1 \cdot \log_2 0{,}1 - 0{,}9 \cdot \log_2 0{,}9 = 0{,}47.$$

When switching to coding in blocks of two signs (Table 8)

$$l_{cp} = \frac{l_{cp,бл}}{2} = \frac{1}{2}\left(0{,}81 \cdot 1 + 0{,}09 \cdot 2 + 0{,}09 \cdot 3 + 0{,}01 \cdot 3\right) = 0{,}645.$$

It can be verified that when encoding in blocks of three symbols, the average number of symbols per sign decreases and turns out to be about 0.53. The effect is achieved due to the fact that when blocks are enlarged, groups can be divided into subgroups that are closer in terms of total probabilities. In this case

$$\lim_{n\to\infty} l_{cp} = H(z),$$

where *n* is the number of symbols in the block.

Table 8: Block coding

Blocks	Probabilities	Codes
$z_1 z_1$	0,81	1
$z_1 z_2$	0,09	01
$z_2 z_1$	0,09	001
$z_2 z_2$	0,01	000

Arithmetic coding

Arithmetic coding is a lossless information compression algorithm that matches a real number from the interval [0; 1] to the text during encoding.

This method, like the Huffman algorithm, is entropic – the length of the code of a particular symbol depends on the frequency of occurrence of this symbol in the text.

Arithmetic coding shows better compression results than the Huffman algorithm for data with non-uniform probability distributions of encoded symbols. In addition, with arithmetic coding each character is encoded with a non-integer number of bits, which is more efficient than the Huffman code (theoretically, the character a with probability of occurrence p(a) is allowed to correspond to a code of length $-log_2\, p(a)$, therefore, when encoding with the Huffman algorithm this is achieved only with probabilities equal to inverse powers of two).

Encoding

The input to the algorithm is the text to be encoded and a list of character occurrence frequencies (Table 9).

1. Consider the segment [0; 1] on a coordinate line.
2. Match each character of the text with a segment whose length is equal to the frequency of its occurrence.
3. Read a symbol from the input stream and consider the segment corresponding to this symbol. Divide this segment into parts proportional to the frequencies of occurrence of symbols.
4. Repeat (3) until the end of the input stream.
5. Choose any number from the resulting segment, which will be the result of arithmetic coding.

Note: to optimize the code size, you can select the number from the range [left; right] obtained at the last step that contains the smallest number of characters in the binary record.

Consider the string ***abacaba*** as an example (Table 10):

Table 9.

Symbol	Frequency of occurrence
a	0,571429
b	0,285714
c	0,142857

Table 10.

Counted symbol	Left border segment	Right border segment
	0	1
a	0	0,571429
b	0,326531	0,489796
a	0,326531	0,419825
c	0,406497	0,419825
a	0,406497	0,414113
b	0,410849	0,413025
a	0,410849	0,412093

Decoding

An algorithm on a real number reconstructs the original text.

1. On the segment [0; 1], divided into parts whose lengths are equal to the probabilities of occurrence of symbols in the text, choose a sub segment containing the input real number. The symbol corresponding to this sub segment is added to the answer.
2. Normalize the subtract and the real number.
3. Repeat steps 1-2 until get the answer.

Test questions on the topic

1. Necessary and sufficient conditions for matching the signal with the information transmission channel.
2. What is the reason for the need to match the signal with the information transmission channel?
3. What factors determine the information transmission rate and channel capacity?
4. What is the essence of Shannon's theorem for a discrete channel without interference?
5. Explain the method of constructing the Shannon-Fano code.
6. Explain the method of constructing the Huffman code.

TOPIC 7
OTHER EFFECTIVE CODES

Elias codes

Gamma code

The Elias Gamma Code is a universal code for encoding for positive integers, developed by Peter Elias. It is usually used to encode integers whose maximum value whose maximum value cannot be determined in advance.

The coding algorithm of Elias's gamma code is (Table 11):

1. Write the number in binary representation.

2. Add zeros before the binary representation, the number of zeros is one less than the number of bits of the binary representation.

Algorithm for decoding Elias's gamma code

1. Count all zeros occurring before the first unit. Let N be the number of these zeros.

2. Taking into account the 1, which will be the first bit of the full whole number, with a value of 2^N to count the remaining N digits of the whole number.

Table 11. The coding algorithm of Elias's gamma code

Number	Binary view	Number of bits	Elias's gamma code
1	1	1	1
2	10	2	0 10
3	11	2	0 11
4	100	3	00 100
5	101	3	00 101
6	110	3	00 110
7	111	3	00 111
8	1000	4	000 1000
9	1001	4	000 1001
10	1010	4	000 1010
11	1011	4	000 1011
12	1100	4	000 1100

Delta code

Elias's delta code is a modification of Elias's gamma code, in which the number of digits of the binary representation of a number is in turn also encoded by Elias's delta code.

The coding algorithm of Elias's delta code is (Table 12):

1. Count L the number of significant bits in the binary representation of the number N.

2. Encode L using Elias's gamma code.

3. Add to L on the right side the binary representation of the number N without a high unit.

Algorithm for decoding Elias's delta code

1. Count M - the number of zeros in the input stream to the first 1.

2. Without including 1 count M bits. The counted number summed with 2^M gives L.

3. Next come L - 1 low bits of the number N. Count them and add 2^{L-1} to the counted number.

Table 12. The coding algorithm of Elias's delta code

Number	Binary view	Number of bits	Elias's gamma code number of bits	Elias's delta code
1	1	1	1	1
2	1 0	2	0 10	010 0
3	1 1	2	0 10	010 1
4	1 00	3	0 11	011 00
5	1 01	3	0 11	011 01
6	1 10	3	0 11	011 10
7	1 11	3	0 11	011 11
8	1 000	4	00 100	00100 001
9	1 001	4	00 100	00100 010
10	1 010	4	00 100	00100 011
11	1 011	4	00 100	00100 100
12	1 100	4	00 100	00100 101

Elias's omega code (recursive code)

Like Elias's gamma and delta code, it assigns to the beginning of an integer the order of its value in the universal code. However, unlike the other two codes mentioned above, the omega code recursively encodes the prefix, which is why it is also known as the recursive Elias code.

The coding algorithm of Elias' omega code is as follows (Table 13):

1. Rewrite the group of zeros to the end of the representation.

2. If the number to be encoded is a one, stop; if not, add the binary representation of the number as a group to the beginning of the representation.

3. Repeat the previous step, with the number of digits(bits) just written, minus one, as the new number to be encoded.

Table 13. The coding algorithm of Elias' omega code

Number	Elias's omega code	Step 1	Step 2	Step 3
1	0	0		
2	10 0	0	10 0	
3	11 0	0	11 0	
4	10 100 0	0	100 0	10 100 0
5	10 101 0	0	101 0	10 101 0
6	10 110 0	0	110 0	10 110 0
7	10 111 0	0	111 0	10 111 0
8	11 1000 0	0	1000 0	11 1000 0
9	11 1001 0	0	1001 0	11 1001 0
10	11 1010 0	0	1010 0	11 1010 0
11	11 1011 0	0	1011 0	11 1011 0
12	11 1100 0	0	1100 0	11 1100 0

Algorithm for decoding Elias's omega code

1. Start with the variable N set to the value 1.

2. Read the first "group" following the remaining N digits, which will consist of either "0" or "1". If it consists of "0", it means that the value of the integer is 1; if it starts with "1", then N gets the value of the group, which is interpreted as a binary number.

3. Read each successive group; it will consist of either "0" or "1" following the remaining N digits. If the group is "0", it means that the value of the

integer is N; if it starts with a "1", then N gets the value of the group, interpreted as a binary number.

Dictionary codes

LZW algorithm

The Lempel-Ziv-Welch (LZW) algorithm is a general-purpose lossless data compression algorithm.

The immediate predecessor of LZW is the LZ78 algorithm published by Abraham Lempel and Jacob Ziv in 1978. This algorithm was perceived as a mathematical abstraction until 1984 when Terry A. Welch published his work with a modified algorithm, later called LZW (Lempel-Ziv-Welch).

Description

The compression process is as follows.

1. The characters of the input stream are read sequentially and a check is made whether such a string exists in the created string table.

2. If such a string exists, the next character is read, and if the string does not exist, the code for the previous found string is put into the stream, the string is put into the table, and the search starts again.

For example, if byte data (text) is compressed, there will be 256 rows in the table (from "0" to "255"). If a 10-bit code is used, then values between 256 and 1023 remain as codes for the rows. New rows form the table sequentially - you can consider the index of the row as its code.

The decoding algorithm needs only the coded text as input, since it can reconstruct the corresponding transformation table directly from the coded text. The algorithm generates unambiguously decodable code by adding a new string to the string table each time a new code is generated. LZW constantly checks if the string is already a known string and, if so, outputs the existing code without generating new code. Thus, each string will be stored in a single instance and will have its own unique number. Hence, during decryption, when a new code is received, a new string is generated and when an already known code is received, the string is retrieved from the dictionary.

Pseudocode of the algorithm

1. Initialisation of the dictionary with all possible one-character phrases. Initialisation of the input phrase *ω* with the first character of the message.

2. Read the next character *K* from the encoded message.

3. If *END_MESSAGE*, output the code for *ω*, otherwise:

4. If the phrase *ω(K)* is already in the dictionary, assign the value of *ω(K)* to the input phrase and proceed to Step 2, otherwise output the code for ω, add *ω(K)* to the dictionary, assign the value of *K* to the input phrase and proceed to Step 2.

5. End.

Encoding Example (Table 14).

Suppose compress the sequence ***"abacabadabacababae"***.

Step 1: Add "a" to the initially empty string and check if the string "a" is in the table. Since all single-character strings were entered into the table during initialisation, there is a string "a" in the table.

Step 2: Next, read the next symbol "b" from the input stream and check if there is a row "ab" in the table. There is no such row in the table yet.

Add "ab" to table <5>. To the stream: <0>;

Step 3: "ba" is not present. In the table: <6> "ba". To stream: <1>;

Step 4: "ac" - no. To table: <7> "ac". To stream: <0>;

Step 5: "ca" - no. To table: <8> "ca". To stream: <2>;

Step 6: "ab" is in the table; "aba" is not. To table: <9> "aba". To stream: <5>;

Step 7: "ad" is not in the table. To table: <10> "ad". To stream: <0>;

Step 8: "da" - no. To table: <11> "da". To stream: <3>;

Step 9: "aba" is in the table; "abac" is not. To table: <12> "abac". To stream: <9>;

Step 10: "ca" is in the table; "cab" is not. To table: <13> "cab". To stream: <8>;

Step 11: "ba" is in the table; "bae" is not. To table: <14> "bae". To stream: <6>;

Step 12: And finally, the last line "e", followed by the end of the message, so just output <4> to the stream.

Table 14. Example of sequence coding "abacabadabacababae"

Current line	Current symbol	Next symbol	Output		Dictionary
			Code	Bits	
ab	a	b	0	000	5: ab
ba	b	a	1	001	6: ba
ac	a	c	0	000	7: ac
ca	c	a	2	010	8: ca
ab	a	b	-	-	-
aba	b	a	5	101	9: aba
ad	a	d	0	000	10: ad
da	d	a	3	011	11: da
ab	a	b	-	-	-
aba	b	a	-	-	-
abac	a	c	9	1001	12: abac
ca	c	a	-	-	-
cab	a	b	8	1000	13: cab
ba	b	a	-	-	-
bae	a	e	6	0110	14: bae
e	e	-	4	0100	-

So get the coded message "0 1 0 2 2 5 0 3 9 8 6 6 4", which is 11 bits shorter.

Decoding

The specialty of LZW is that for decompression, don't need to save the row table to a file for decompression. The algorithm is constructed in such a way that it is possible to reconstruct the table of strings using only the code stream.

Now imagine that having received the encoded message described above, it is necessary to decode it (Table 15.). First of all, it is necessary to know the initial vocabulary, and subsequent vocabularies can be reconstructed as needed, since they are simply concatenations of the previous ones.

+ No need to calculate probabilities of occurrence of symbols or codes.

+ Decompression does not require saving the string table to a file for decompression. The algorithm is constructed in such a way that it is possible to reconstruct the row table using only the code stream.

+ This type of compression does not distort the original graphic file, and is suitable for compressing raster data of any type.

- The algorithm does not analyse the input data, so it is not optimal.

Table 15. Example of sequence decoding "abacabadabacababae"

Data		Output	New entry	
Bits	Code		Full	Partial
000	0	a	-	5: a?
001	1	b	5: ab	6: b?
000	0	a	6: ba	7: a?
010	2	c	7: ac	8: c?
101	5	ab	8: ca	9: ab?
000	0	a	9: aba	10: a?
011	3	d	10: ad	11: d?
1001	9	aba	11: da	12: aba?
1000	8	ca	12: abac	13: ca?
0110	6	ba	13: cab	14: ba?
0100	4	e	14: bae	-

Application

The publication of the LZW algorithm made a great impression on all information compression specialists. This was followed by a large number of programs and applications with different variants of this method.

This method allows to achieve one of the best compression ratios among other existing methods of compression of graphic data, with no loss or distortion in the original files. It is currently used in TIFF, PDF, GIF, PostScript and other files, as well as partially in many popular data compression programs (ZIP, ARJ, LHA).

Test questions on the topic

1. Dictionary methods of data compression.
2. Statistical methods of data compression.
3. Arithmetic methods of data compression.
4. Elias's algorithm for encoding and decoding.
5. Algorithm for encoding and decoding by the LZW method.
6. Modern trends in the development of data compression.

TOPIC 8
NOISE -RESISTANT CODING. HAMMING CODE

Basic characteristics of noise-resistant coding

Redundant encoding (redundant encoding) is a type of encoding that uses an excessive amount of information for the purpose of subsequent control of data integrity when recording/reproducing information or when transmitting it over communication lines.

In the simplest case (constant length encoding) the encoding procedure consists in matching k information symbols corresponding to the symbol to be encoded, a block of n symbols.

k – the number of information bits,

n – the total number of digits in the noise-resistant code,

(n – k) – the number of check digits.

In this case the code is denoted as ***(n, k)***

The corrective ability of the code is characterized by two values.

r – multiplicity of detectable errors. This is the number of digits, in case of simultaneous occurrence of errors in which only detection of the fact of an error is guaranteed, while detection of the exact digits in which these errors occurred is not guaranteed.

s – multiplicity of corrected errors. This is the number of digits, in case of simultaneous occurrence of errors in which not only detection of the fact of an error is guaranteed, but also determination of the exact digits in which these errors occurred.

Codes in which automatic error correction is possible are called self-correcting. At present the most interesting are binary block correcting codes. When using such codes, information is transmitted in the form of blocks of equal length and each block is encoded and decoded independently of each other. In almost all block codes symbols can be divided into information and verification. Thus, all combinations of codes are divided into allowed (for which the ratio of information and check characters is possible) and forbidden.

Repetition codes

In this code, each transmitted character is repeated exactly *n* times. Accordingly, denotes this code *(n,1)*.

The correction abilities of the repetition code for different *n* are summarized in Table 16.

Table 16. The correction abilities of the repetition code for different n

n	2	3	4
Source message 1	11	111	1111
Message with **one error**	01	101	1101
Output on message with **one error**	Error is, digit unknown	Error is, digit 2	Error is, digit 2
Message with **two errors**	00	100	1100
Output on message with **two errors**	No error	Error is, digit 3	Double error, digit unknown
Message with **three errors**	-	000	1000
Output on a message with **three errors**	-	No error	Error is, digit 4

Consider a number of requirements to the length of noise-resistant codes.

Allowed combinations are code sequences that do not contain errors. In other words, these are sequences that can appear as a result of coding.

An error vector is a binary sequence containing ones in error-prone bits and zeros in other bits. Accordingly, any distorted combination can be considered as the result of modulo 2 addition of the original allowed combination and the error vector.

Obviously, the number of allowed code sequences *(n,k)* is equal to 2^k.

Consider the number of possible multiple errors.

Number of multiplicity errors:

$$1 - C_n^1, 2 - C_n^2, \mathrm{i} - C_n^i.$$

For any selected error vector a subgroup of *2^k* code sequences can be formed from those solved by adding to this vector (introducing this error). Suppose that the code *(n,k)* must correct errors of multiplicity up to s inclusive. Then the above-mentioned subgroups for all error vectors of multiplicity at most *s* must be non-intersecting for there to be a possibility of correcting these errors.

The total number of subgroups is:

1(allowed) + C_n^1 (multiplicity 1) +

+C_n^2 (multiplicity 2) + ... + C_n^s (multiplicity s)

Since the subgroups do not overlap and each has *2^k* combinations, their total number is:

$$2^k\left(1+C_n^1+C_n^2+\ldots+C_n^s\right).$$

The maximum number of combinations that can be formed by the code *(n,k)* is *2^n*.

$$2^k\left(1+C_n^1+C_n^2+\ldots+C_n^s\right)\le 2^n.$$

After the transformations, obtain the lower Hamming bound of the noise-resistant code length:

$$C_n^1+C_n^2+\ldots+C_n^s\le 2^{n-k}-1$$

$$n\ge k+\log_2\left(1+C_n^1+C_n^2+\ldots+C_n^s\right).$$

Correlation of corrective power with coding distance

The (Hamming) weight of a coding sequence is defined as the number of non-zero components of this sequence. It is clear that the coding distance between two sequences is equal to the weight of some third sequence, which is their sum, which (due to the property of the modulo-two addition operation) must also be a sequence of this code. Hence, the minimum code distance for a linear code is equal to the minimum weight of its non-zero vectors.

The code distance d is expressed as the number of symbols in which the sequences differ from each other. To determine the code distance between two combinations of a binary code, it is sufficient to add them mod 2, and count the number of units in the result.

The minimum distance counted over all pairs of allowed code combinations is called the ***minimum code distance of the given code***.

Usually decoding is carried out in such a way that any accepted disallowed code combination is identified with the allowed combination located from it on the minimum code distance. If the minimum code distance of a given code ***d =1***, (all code combinations are allowed), then ***the error cannot be detected***. If ***d = 2***, then a ***single error will be detected***, etc.

In general, if it is necessary to detect an error of multiplicity up to and including ***r***, the minimum code distance must satisfy the following condition

$$d_{\min} \geq r+1\,.$$

To correct errors of multiplicity ***s***, each allowed code combination must be matched with a subset of disallowed combinations so that these subsets do not overlap. For this purpose, the inequality must be fulfilled

$$d_{\min} \geq 2s+1\,.$$

To correct errors of multiplicity s and at the same time detect all errors of multiplicity r ($r \geq s$), the minimum coding (Hamming) distance must satisfy the inequality

$$d_{\min} \geq r+s+1\,.$$

Give a *geometrical interpretation* of the above relations.

Any n-bit binary code combination can be interpreted as a vertex of an *n*-dimensional hypercube with edge length equal to 1. For example, at n = 2 it is a square, at n = 3 it is a unit cube. In general case *n* - dimensional hypercube contains 2^n vertices that coincides with possible number of *n* -bit binary code combinations.

The code distance can be interpreted as the smallest number of edges that must be traversed to get from one allowed combination to another. The subset of each allowed combination includes all vertices that are in the sphere of radius

$$s \le (d-1)/2\,.$$

If, as a result of noise action, the resolved combination moves to a point belonging to the sphere, it can be corrected.

Hamming coding

Hamming codes are self-checking codes – codes that allow automatic detection of errors during data transmission. For their construction it is enough to assign to each word one additional (control) binary digit and choose the digit of this digit so that the total number of units in the image of any number was, for example, odd. A single error in any bit of the transmitted word (including, perhaps, the control bit) will change the parity of the total number of units. Modulo 2 counters, counting the number of units, which are contained among the binary digits of the number, can signal the presence of errors.

There is no way to know in which digit the error occurred and, therefore, there is no possibility to correct it. Errors occurring simultaneously in two, four, and so on. - in an even number of digits. However, double, and even more so fourfold errors are considered unlikely.

The construction of Hamming codes is based on the principle of checking the parity of the number of single symbols: an element is added to the sequence so that the number of single symbols in the resulting sequence is even.

The following algorithm generates a code that corrects single errors for any number of bits.

1. Number the digits starting from 1: 1, 2, 3, 4, 5, ...
2. Write the digit numbers in binary representation: 1, 10, 11, 100, 101, ...
3. All digits that are powers of two are bits of parity: 1, 2, 4, 8, ... (1, 10, 100, 1000)
4. All other bits are data bits.

Each data bit is included in a unique set of two or more parity bits according to the binary representation of its sequence number.

Parity bit 1 covers all bit positions whose number contains a one in the low bit: 1, 3, 5, 7, 9, ...

The parity 2 bit covers all bit positions whose number contains one in the second bit: 2, 3, 6, 7, 10, ...

The parity 4 bit covers all bit positions whose number contains one in the third bit: 4-7, 12-15, ...

Parity bit 8 covers all bit positions whose number contains a one in the fourth bit: 8-15, 24-31, ...

In general, each parity bit P covers all bit positions for whose number the bitwise I with the number of that bit is not zero.

This rule can be represented visually (Table 17):

Table 17. Algorithm of Hamming code construction

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16		17	18	19	20	
	P1	P2	I1	P4	I2	I3	I4	P8	I5	I6	I7	I8	I9	I10	I11	P16		I12	I13	I14	I15	
P1	1		1		1		1		1		1		1		1			1		1		…
P2		1	1			1	1			1	1			1	1				1	1		
P4				1	1	1	1					1	1	1	1						1	
P8								1	1	1	1	1	1	1	1							
P16																1		1	1	1	1	

Only 20 encoded bits are shown (5 for parity P, 15 for information I), but the pattern can be continued indefinitely. The key feature of the Hamming code, which can be seen from the example, is that each selected bit is part of a unique set of parity bits. All parity bits are checked for errors. An error pattern called ***syndrome*** shows the error bits. If all parity bits are correct, there is no error; otherwise, the sum of the positions of the erroneous parity bits will show the error bit. If exactly one parity bit is wrong – the error is in it.

As you can see, if there are m parity bits, they can cover from 1 to 2^m-1 bits. If to subtract parity bits, $2^m - m - 1$ information bits remain. By changing m, get all possible Hamming codes. Hamming code fragment given in the Table 18 below.

In the left part of the table one column is left empty, which will contain the results of calculations of control bits. Calculation of control bits is performed as follows. Take one of the rows of the transformation matrix and find its scalar product with the codeword. In other words, the corresponding bits of both rows are multiplied and the sum of products is found. If the product is greater than one, find the remainder of its division by 2. In other words, calculate how many times in the codeword and the corresponding row of the matrix in the same positions there are units, and take this number modulo 2.

Table 18. Hamming code fragment

Discharge		1	2	3	4	5	6	7	
Title bit		P1	P2	I1	P4	I 2	I 3	I4	
Code		0	1	1	0	0	1	1	…
P1	0	1	0	1	0	1	0	1	
P2	1	0	1	1	0	0	1	1	
P4	0	0	0	0	1	1	1	1	

Hamming code with additional parity check (Single error correction, double error detection)

Hamming codes have a minimum distance equal to three. This means that these codes can detect and correct a single error. Thus, such a code can detect double errors if it is not used for error correction.

To correct this drawback, Hamming codes can be extended with an additional parity bit. In this way, the minimum coding distance can be increased to four, which can distinguish between single and double errors. In this way the decoder can detect and correct single errors and detect (but not correct) double errors. If such a code is not used for error correction, it can detect triple errors.

Such extended Hamming codes are used in computer memory systems, where they are known as SECDED (an abbreviation of Single Error Correction, Double Error Detection). In particular, codes (72,64), (127,120) are used.

Decoding algorithm

The Hamming decoding algorithm is absolutely identical to the coding algorithm. The transformation matrix of the corresponding dimension is multiplied by the codeword column matrix and each element of the obtained column matrix is taken mod 2. The resulting matrix-column has been called "syndrome matrix". It is easy to check that the codeword generated according to the algorithm described in the previous section always gives a zero syndrome matrix.

The syndrome matrix becomes non-zero if, as a result of an error (for example, when transmitting the word over a communication line with noise), one of the bits of the source word has changed its value. Assume for example that in the codeword obtained in the previous section, the sixth bit has changed its value from zero to one (in the figure it is marked in red). Then obtain the following matrix of syndromes (Table 19).

Table 19. Hamming code syndrome matrix

Discharge		1	2	3	4	5	6	7	
Title bit		P1	P2	I1	P4	I2	I3	I4	
Code		0	1	0	0	0	1	1	…
S1	1	1	0	1	0	1	0	1	
S2	1	0	1	1	0	0	1	1	
S4	0	0	0	0	1	1	1	1	

Test questions on the topic

1. What is the essence of Shannon's theorem for a discrete channel with interference?
2. Explain the nature of the dependence of the throughput of a continuous channel with interference on the channel capacity.
3. What codes are called corrective?
4. How is the distance between code combinations determined?
5. What is meant by code distance?
6. Highlight the methodology for constructing the Heming code?

TOPIC 9 CYCLIC CODES

Operations on polynomials

It is convenient to describe cyclic codes using polynomials. For this purpose introduce a dummy variable *x*, the degrees of which correspond to the numbers of digits starting from 0. The digits 0 and 1 are taken as coefficients of polynomials – introduce polynomials over the field GF (2).

For example, the first line 1001011 is described by the polynomial

$$1\cdot x^6+0\cdot x^5+0\cdot x^4+1\cdot x^3+0\cdot x^2+1\cdot x^1+1\cdot x^0=x^3+x+1\ .$$

The polynomial for each next line is formed by multiplying by *x*. In this case, if the leftmost symbol differs from zero, to realize the operation of moving the unit to the end of the combination, it is necessary to subtract (add mod 2) from the result polynomial $x^n + 1$.

All combinations of the cyclic code can be constructed on the ring of polynomials by setting on the set of *n* -digit code combinations of two operations – addition and multiplication.

The polynomial *addition* operation in this case is realized as an addition of the corresponding coefficients mod 2.

The *multiplication* operation is realized in the following sequence. The polynomials are multiplied as usual with the subsequent addition of the coefficients mod 2. If multiplication results in a polynomial of degree *n* or higher, then it is divided by the given polynomial of degree *n*, and the result of multiplication is the remainder of the division. It is clear that the higher degree of this remainder will not exceed the value *n -1*, and the obtained remainder will correspond to some *n* -digit code combination - closedness is provided.

The *division* operation is the usual division of polynomials, only instead of subtraction the addition mod 2 is used.

$$\begin{array}{l|l}
x^6+x^4+x^2+1 & \underline{x^3+x+1} \\
\underline{x^6+x^4+x^3} & x^3+1 \\
\quad x^3+x^2+1 & \\
\quad \underline{x^3+x+1} & \\
\qquad x^2+x &
\end{array}$$

To realize a cyclic shift using the multiplication operation described above, after multiplying by *x*, it is necessary to divide by the dipartite $x^n + 1$. This operation is called taking the remainder or modulo conversion $x^n + 1$, and the remainder itself is called the subtraction:

$$\begin{array}{lcl|l}
(x^{n-1}+x^{n-2}+\ldots+x+1)\cdot x & = & x^n+x^{n-1}+\ldots+x^2+x & \underline{x^n+1} \\
 & \oplus & \underline{x^n+1\qquad\qquad\qquad\qquad} & 1 \\
 & & \quad 0+x^{n-1}+\ldots+x^2+x+1 &
\end{array} .$$

It is easy to notice that in this case the remainder (deduction) is formed by adding modulo 2 the dipartite $x^n + 1$ with the result of multiplication by *x*.

The concept and general scheme of constructing a cyclic code

A cyclic code is a code, each combination of which can be constructed as a linear combination of codes, each of which is obtained by cyclic shift of some base combination belonging to the same code. If the shift is performed from right to left, the leftmost character is moved to the end of the code combination.

It is convenient to describe cyclic codes using polynomials. For this purpose, a dummy variable *x*, whose degrees correspond to the numbers of digits, starting from 0, is introduced. The digits 0 and 1 are taken as coefficients of polynomials – polynomials over the field GF(2) are introduced into consideration.

For example, the first line **1001011** is described by the polynomial

$$1\cdot x^6+0\cdot x^5+0\cdot x^4+1\cdot x^3+0\cdot x^2+1\cdot x^1+1\cdot x^0=x^3+x+1\,.$$

Select a subset of all polynomials in the ring that are divisible by some polynomial *g(x)*. Clearly, this subset will be an ideal, and the polynomial *g(x)* will be the generating or forming polynomial of the ideal. If *g(x)=0*, then the whole ideal consists of one of these polynomials. If $g(x)=1$, then the ideal consists of all polynomials of the ring.

In the ring 2^n of all possible polynomials of degree n-1 over the field GF(2), the irreducible polynomial $g(x)$ of degree $m=n - k$ generates 2^k elements of the ideal. Hence, a cyclic binary code can be defined as an ideal, each polynomial of which corresponds to an n -bit allowed code combination. Determine what requirements the generating polynomial of the ideal - $g(x)$ - must satisfy.

The generating polynomial must satisfy the following requirements:

- $p(x)$ must be non-zero;

- the weight of $p(x)$ must not be less than the minimum coding distance: $v(p(x)) \geq d_{min}$;

- $p(x)$ must have maximum degree k (k is the number of redundant elements in the code);

- $p(x)$ must be a divisor of the polynomial (x^n - 1).

If $g(x)$ satisfies this requirement, then the ring of polynomials can be decomposed into classes of ideal deductions.

For clarity, the decomposition scheme is shown in Table 20. The first row in this table is the ideal itself together with the zero polynomial. As forming elements of the classes take (corresponding to the error vectors) polynomials $r(x)$ that do not belonging to the ideal, and the classes of deductions on the ideal are formed by adding the elements of the ideal with the forming polynomials.

If the above scheme of formation of deduction classes is implemented, and the polynomial $g(x)$ of degree $m = n - k$ is a divisor of the bipartite $x^n + 1$, then each element of the ring is either divisible by $g(x)$ without remainder (then it is an ideal element), or the remainder of the division $r(x)$ appears - it is a polynomial of degree not higher than m -1. The elements of the ring giving the same remainder $r(x)$ belong to the same class of deductions.

Table 20. The scheme of decomposition of polynomials into classes of deductions by the ideal

0	$r_1(x)$	$\cdots$	$r_z(x)$
$g(x)$	$g(x)+r_1(x)$	$\cdots$	$g(x)+r_z(x)$
$xg(x)$	$xg(x)+r_1(x)$	$\cdots$	$xg(x)+r_z(x)$
$(x+1)g(x)$	$(x+1)g(x)+r_1(x)$	$\cdots$	$(x+1)g(x)+r_z(x)$
$\cdots$	$\cdots$	$\cdots$	$\cdots$
$f(x)\cdot g(x)$	$f(x)\cdot g(x)+r_1(x)$	$\cdots$	$f(x)\cdot g(x)+r_z(x)$

The corrective power of the code is higher the more classes of deductions, residues *r(x)*. The largest number of residues $2^m -1$ is given by an irreducible polynomial. As an example, Table 21 shows the irreducible polynomials up to the third degree inclusive. Tables including a large number of irreducible polynomials can be found, for example, in [1], [2].

Table 21. Table fragment of irreducible polynomials up to degree three

M	Code	$g(x)$
1	11	$x+1$
2	111	x^2+x+1
3	1011	x^3+x+1
3	1101	x^3+x^2+1

Selection of forming polynomials for detection and correction of single errors

Correction of single errors. Each single error in one of the n digits must correspond to its own class of deductions and its identifier – the residue from division by the forming polynomial *g(x)*. As mentioned above, the largest number of residues is given by an irreducible polynomial. If $m = n - k$ degree of this polynomial, the number of non-zero residues will be $2^{n-k} -1$. Thus, to correct all n single errors it is necessary that the following inequality is satisfied

$$2^{n-k}-1 \geq C_n^1 = n .$$

Hence the degree of the forming polynomial

$$m = n-k \geq \log_2(n+1) .$$

It has been shown above that the forming polynomial must be a divisor of the form $x^n + 1$. On the other hand, it is known that any binomial of the form

$$x^{2^m-1}+1=x^n+1$$

can always be represented as the product of all irreducible polynomials whose degrees are divisors of *m* from 1 to *m* inclusive. Consequently, for any *n* there exists at least one irreducible polynomial of degree *m*, which is a minor in the expansion of the bipartite $x^n + 1$. This polynomial can be taken as a forming polynomial.

Methods of formation of combinations and decoding of the cyclic code

Construction of a non-systematic code.

For the construction of *n* -bit allowed combination polynomial *a(x),* corresponding to the encoded sequence of information symbols, is multiplied by the forming polynomial:

$$q(x)=a(x)g(x).$$

At decoding (possibly different from *q(x)* polynomial $\tilde{q}(x)$ corresponding to the accepted combination is divided by *g(x).* It is clear that in case of absence of errors the initial polynomial *a(x)* will be obtained at once. If the accepted combination contains an error, the division produces the remainder *r(x)*

$$\tilde{q}(x)/g(x)=f(x)+r(x)/g(x).$$

The remainder is used to determine the class of deductions and correct the error.

The disadvantage of this coding method is that after error detection and correction it is necessary to divide by *g (x)* again in order to select information symbols.

Constructing a systematic code.

The polynomial corresponding to the original information parcel *a(x),* is multiplied by x^m. The free lower digits are filled with the remainder from dividing the given expression by the forming polynomial:

$$q(x) = a(x) \cdot x^m + r(x).$$

The polynomial *q(x)* must be divisible by *g(x)* without remainder. Show that.

When dividing $a(x)x^m$ by g(x) generally have

$$a(x) \cdot x^m / g(x) = c(x) + r(x)/g(x),$$

where c(x) is an integer polynomial. This equality (taking into account that the operations of subtraction and addition modulo two coincide) can be rewrite as

$$a(x) \cdot x^m / g(x) + r(x)/g(x) = c(x),$$

or

$$q(x) = a(x) \cdot x^m + r(x) = c(x) g(x).$$

In this case, the information symbols always remain in the first k positions. Such a code is called a systematic code. In this coding method, the original code sequence occupying the first k positions immediately becomes known after error correction.

Test questions on the topic

1. What are the main properties of cyclic codes?
2. What are the ways to construct cyclic codes?
3. What is the method of decoding cyclic codes?
4. What are the principles of choosing a generating polynomial?
5. What condition is necessary to correct any single error?
6. What operations are performed to detect and correct an erroneous bit?

TOPIC 10
ERROR PACKET CORRECTION. CYCLIC REDUNDANT CODE

Error packet correction

The codes previously considered were designed to correct random errors. In general, a code that corrects *t* errors can correct any error patterns of weight *t* or less in a code word-block of length *n*. However, there are channels in which errors occur in small intervals rather than completely randomly. For example, in storage media, errors arise due to physical changes, so are concentrated rather than randomly dispersed. Similarly, interference at short intervals causes packets of errors. There is a family of codes used to correct such multiple errors. Consider them in this lecture.

Call an ***error packet*** of length *t* an error vector whose non-zero components are within t neighbouring digits.

Call a ***cyclic error packet*** of length *t* an error vector whose non-zero components are within t neighbouring digits given at least one cyclic permutation of this vector.

Examples:

(0**101011**0000) is a cyclic error packet of length 6,

(000000**10001**) is a cyclic error packet of length 5,

(**01**000000100**101**) is a cyclic error packet of length 5.

An error packet of length *t* can be described in terms of a polynomial as follows

$$e(x) = x^i b(x) \quad \left(\operatorname{mod} x^n - 1\right)$$

where *b(x)* is a polynomial of degree *t-1* describing the error pattern, and *i* indicates where the error starts. For the examples proposed above:

(0**101011**0000) $\qquad e(x) = \left(x^5 + x^3 + x + 1\right)x^4$

(0**101011**0000) $\qquad e(x) = \left(x^4 + 1\right)x^0$

(0**101011**0000) $\qquad e(x) = \left(x^4 + x^2 + 1\right)x^9$

Consider a linear code *C*. If all error packets of length *t* or less occur in different subsets, then each can be identified by its syndrome and all such errors are correctable. Moreover, if *C* is a linear code capable of correcting all error packets of length t or less, then all such errors must occur in distinct subsets.

Suppose that *C* can correct two such distinct errors e_1 and e_2 that lie in different subsets of c_i. Then $e_1 – e_2 = c –$ is a non-zero codeword. Suppose that $e_1 –$ is the received vector. How can it be decoded? The codeword 0 can be converted to e_1 by introducing error e_1 or codeword *c* can be converted to e_1 by error injection e_2. Here a contradiction is reached, as this code is not capable of correcting error packets of length *t* or less.

Error catching

A cyclic code can correct all error packets of length *t* or less if and only if the syndromes of these errors are different. It is possible to decode packets with cyclic errors by error trapping.

It is possible to prove that a *(n, k)* code correcting error packets of length *t* satisfies the constraint $n-k \geq 2t$. Hence, $n-k \geq t$ and $n-t \geq k$. Now an error packet of length *t* in a codeword of length *n* has a cyclic sequence of $n-t$ zeros, which is a requirement for the error-catching algorithm to work. Here is a modification of the error capture algorithm that can be used for all error packets of length t or less in a cyclic code to correct error packets of length *t.*

(1) Compute the syndrome.

(2) Set $i = 0$

(3) If $s_i(x)$ is a non-cyclic error packet of length $\leq t$, then

$$e(x) = x^{n-i}\left[s_i(x), 0\right].$$

(4) Let $I = I + 1$.

(5) If $I = n$, stop, the error pattern is undefined.

(6) Compute $s_i(x) = xs_{i-1}(x)$. If $s_i(x) > n-k$, $s_i(x) = s_i(x) - g(x)$

(7) Return to step (3)

Example

$$g(x) = 1 + x + x^2 + x^3 + x^6 \quad (1001111)$$

generates cyclic code to correct error packets of length 3 or less. Vector is received.

$$(000.0011.0111.0111)$$

$$r(x)=\left(x^2+x^3\right)g(x)+\left(1+x+x^4+x^5\right).$$

Calculating syndromes (Table 22)

Table 22. Syndrome table

i	Syndrome
0	110011
1	101001
2	011101
3	111010
4	111011
5	111001
6	111101
7	110101
8	100101
9	000101

$e(x)=10100000$

Some suitable polynomials

Constitutive polynomials satisfying the conditions of the theorem are hard to find, so give some examples of such codes for small *t* (Table 23).

Table 23. Fragment of the table of forming polynomials

g(x)	(n, k)	t
$1+x^2+x^3+x^4$	(7, 3)	2
$1+x^2+x^4+x^5$	(15, 10)	2
$1+x^4+x^5+x^6$	(31, 25)	2
$1+x^3+x^4+x^5+x^6$	(15, 9)	3
$1+x+x^2+x^3+x^6$	(15, 9)	3

Cyclic redundancy code

Cyclic redundancy check (CRC) is a checksum algorithm for checking data integrity. CRC is a practical application of noise-resistant code based on certain mathematical properties of cyclic code.

The CRC algorithm is based on the properties of division with remainder of binary polynomials. The CRC value is essentially the remainder from dividing the polynomial corresponding to the input data by some fixed generating polynomial.

Algorithm parameters

One of the main parameters of the CRC is the generating polynomial.

Another parameter associated with the generating polynomial is its degree, which determines the number of bits used to compute the CRC value. In practice, the most common are 8, 16- and 32-bit words, which is a consequence of the peculiarities of the architecture of modern computing technology.

Another parameter is the initial (start) value of the word. These parameters completely define "traditional" algorithm of CRC calculation (Table 24). There are also modifications of the algorithm, for example, using reverse order of bit processing.

Table 24. Algorithm parameters of the CRC

CRC-1		0x1
CRC-4-ITU		0x3
CRC-5-EPC		0x09
CRC-5-ITU		0x15
CRC-5-USB		0x05
CRC-6-CDMA2000-A		0x27
CRC-6-CDMA2000-B		0x07
CRC-6-DARC		0x19
CRC-6-ITU		0x03

Procedure description

Realization of CRC on logic elements

A string of n zeros is added to the original string. If the high bit in the string is "1", the word is shifted to the left by one digit, followed by the XOR operation with the generating polynomial. Correspondingly, if the high bit in the word is "0", the XOR operation is not executed after the shift. The residue obtained after passing through the whole string is the checksum.

A checksum of n bits is added to the original string. If the above procedure shows 0 in the remainder, the string corresponds to the specified checksum.

Test questions on the topic

1. Stages of building a group code.
2. What are the advantages of cyclic codes?
3. Properties of cyclic codes.
4. What cyclic codes are widely used?
5. Stages of building a cyclic code.
6. Characteristics of methods of decoding cyclic codes.

TOPIC 11
MATRIX CODES. ADAMAR CODES

Matrix codes

In coding theory, a matrix whose rows form the basis of a linear code is called a forming (generating) matrix. Code words in this case are all linear combinations of the rows of this matrix.

If **G** is a generating matrix, the codewords of a noise-resistant code are formed by the following multiplication.

$$w = sG,$$

where **s** is any vector.

The forming matrix for a *(n, k)* code contains *k* rows and *n* columns.

The standard form of the **forming matrix** is:

$$\mathbf{G} = \left[\mathbf{I}_k \vdots \mathbf{P}_{k,n-k}\right].$$

The forming matrix can be used to form a **verification matrix** (and vice versa):

$$\mathbf{H} = \left[\mathbf{P}^T_{n-k,k} \vdots \mathbf{E}_{n-k}\right]$$

Binary codes are equivalent if the matrix of one code can be obtained from the other by the following transformations:

- column permutation
- row permutation
- addition of one row with another

Constructing a complement matrix

The complement matrix contains all the information about the code construction scheme.

There is a formal way of constructing the complement matrix based on the following requirement. Vector-string, resulting from summation of any *l*, $(1 \leq l \leq k)$ rows of the augmentation matrix must contain not less than $d_{min} = l$ non-zero symbols where d_{min} – minimum code distance.

In accordance with the above requirement, the complement matrix can be constructed by following the following rules:

- the number of units in a row must be at least d_{min}-*1*;

- the sum modulo two of any two rows must contain not less than d_{min}-*2* units.

If these requirements are met, the combination obtained by summation of any 2 rows of the forming matrix will contain not less than d_{min} non-zero symbols.

The cyclic code is a group code, so it can be constructed using matrix representations as described above. However, in this case some additional possibilities related to the cyclic property also appear. Consider the ways of constructing the forming matrix of a cyclic code.

Method 1: Forming polynomial be given in the form

$$g(x) = g_m x^m + \ldots + g_1 x + g_0 .$$

Then the forming matrix can be constructed by multiplying *g(x)* by the polynomial x^{k-1}, $k = n - m$ followed by a cyclic shift so that each *i*-th row of the forming matrix is composed of the coefficients of the polynomial

$$g(x) \cdot x^{k-i} \quad (i = \overline{1,k}):$$

$$\mathbf{M}_{n,k} = \begin{bmatrix} g_m & g_{m-1} & \ldots & g_0 & 0 & \ldots & 0 \\ 0 & g_m & g_{m-1} & \cdots & g_0 & \ldots & 0 \\ \ldots & \ldots & \ldots & \ldots & \ldots & \ldots & \ldots \\ 0 & \ldots & 0 & g_m & g_{m-1} & \ldots & g_0 \end{bmatrix}.$$

Method 2. Consider polynomials $Q_i(x)$ corresponding to the code containing only one non-zero digit:

$$Q_i(x) = x^{n-i}, i = \overline{1,k} .$$

The residuals are calculated for them

$$r_i(x) = Q_i(x) / g(x) .$$

Each *i*-th row of the forming matrix is formed by adding modulo two of these polynomials and their corresponding residues. In this case, the forming matrix (in this case of the systematic code) is represented by two submatrices:

$$\mathbf{M}_{n,k} = \left[\mathbf{E}_k \vdots \mathbf{P}_{k,n-k}\right],$$

where $\mathbf{E}_k$ –unit $k \times k$ - matrix, and rows of the matrix augmentation matrix $\mathbf{P}_{k,n-k}$, are residues $r_i(x)$, $i = 1, k$.

Adamar codes

Consider the rule of formation of the Adamar matrix.

$$H_1 = [1], \quad H_2 = \begin{bmatrix} 1 & 1 \\ 1 & - \end{bmatrix}, \quad H_8 = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & - & 1 & - & 1 & - & 1 & - \\ 1 & 1 & - & - & 1 & 1 & - & - \\ 1 & - & - & 1 & 1 & - & - & 1 \\ 1 & 1 & 1 & 1 & - & - & - & - \\ 1 & - & 1 & - & - & 1 & - & 1 \\ 1 & 1 & - & - & - & - & 1 & 1 \\ 1 & - & - & 1 & - & 1 & 1 & - \end{bmatrix}, \quad H_{2i} = \begin{bmatrix} H_i & H_i \\ H_i & -H_i \end{bmatrix},$$

$$H_4 = \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & - & 1 & - \\ 1 & 1 & - & - \\ 1 & - & - & 1 \end{bmatrix},$$

To encode and decode the value of –1 of the Adamar matrix (denoted as " – ") corresponds to the value of the code bit equal to 0, and the value of +1 of the Adamar matrix (denoted as "1") – code bit value equal to 1.

Coding

The Adamar code is designed to encode *n* symbols of the input sequence into 2^n output sequence. For this purpose, the corresponding row of the Adamar matrix is taken as the output sequence.

$$c_j(x_i) = \begin{cases} \dfrac{H_{ij} + 1}{2}, & 1 \le i \le n, \\ \dfrac{-H_{ij} + 1}{2}, & n + 1 \le i \le 2n, \end{cases}$$

$$\begin{bmatrix} 0000 \\ 0001 \\ 0010 \\ 0011 \\ 0100 \\ 0101 \\ 0110 \\ 0111 \end{bmatrix} \rightarrow \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & - & 1 & - & 1 & - & 1 & - \\ 1 & 1 & - & - & 1 & 1 & - & - \\ 1 & - & - & 1 & 1 & - & - & 1 \\ 1 & 1 & 1 & 1 & - & - & - & - \\ 1 & - & 1 & - & - & 1 & - & 1 \\ 1 & 1 & - & - & - & - & 1 & 1 \\ 1 & - & - & 1 & - & 1 & 1 & - \end{bmatrix} \begin{bmatrix} 1000 \\ 1001 \\ 1010 \\ 1011 \\ 1100 \\ 1101 \\ 1110 \\ 1111 \end{bmatrix} \rightarrow \begin{bmatrix} - & - & - & - & - & - & - & - \\ - & 1 & - & 1 & - & 1 & - & 1 \\ - & - & 1 & 1 & - & - & 1 & 1 \\ - & 1 & 1 & - & - & 1 & 1 & - \\ - & - & - & - & 1 & 1 & 1 & 1 \\ - & 1 & - & 1 & 1 & - & 1 & - \\ - & - & 1 & 1 & 1 & 1 & - & - \\ - & 1 & 1 & - & 1 & - & - & 1 \end{bmatrix},$$

Decoding

Multiply the Adamar matrix by the resulting sequence, obtaining the vector **F**.

Define the coordinate *a* (the numbering starts from zero), which corresponds to the maximum modulo value of **F**.

If F_a is negative, then the first coordinate of the initial message is equal to 1, if F_a is positive – 0. The other coordinates are equal to the binary representation of *a* .

Example 1

Let the row be encoded

$$\mathbf{x} = \begin{bmatrix} 0 & 1 & 1 & 0 \end{bmatrix}.$$

After encoding, this row corresponds to the message

$$\mathbf{c} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

Suppose that an error occurred during transmission, resulting in the reception of message

$$\mathbf{r}_1 = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 1 \end{bmatrix}.$$

Write the vector corresponding to this message in the notation of the of the Adamar matrix:

$$\mathbf{v} = \begin{bmatrix} 1 & - & - & - & - & - & 1 & 1 \end{bmatrix}.$$

After multiplying the Adamar matrix by the resulting vector, obtain the vector

$$\mathbf{F} = \begin{bmatrix} -2 & 2 & -2 & 2 & -2 & 2 & 6 & 2 \end{bmatrix}.$$

The smallest value of the vector **F** corresponds to the coordinate *a=6*, at that F_a is positive, hence the initial message is formed as

$$\left[0 \vdots 110\right]$$

Example 2

Let a row be encoded

$$\mathbf{x} = \begin{bmatrix} 1 & 0 & 1 & 1 \end{bmatrix}.$$

After coding get

$$\mathbf{c} = \begin{bmatrix} 0 & 1 & 1 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}.$$

As a result of an error, a message is received

$$\mathbf{r}_1 = \begin{bmatrix} 0 & 1 & 1 & 1 & 0 & 1 & 1 & 0 \end{bmatrix}.$$

Vector corresponding to the code

$$\mathbf{v} = \begin{bmatrix} - & 1 & 1 & 1 & - & 1 & 1 & - \end{bmatrix}.$$

The result of multiplication

$$\mathbf{F} = \begin{bmatrix} 2 & -2 & -2 & -6 & 2 & 2 & -2 & 2 \end{bmatrix}.$$

The smallest value of the vector **F** corresponds to the coordinate *a=3*, at that F_a is negative, hence the initial message is formed as

$$\left[1 \vdots 011\right].$$

Test questions on the topic

1. Types of matrix codes.
2. Description of the algorithm of coding by the Adamar code.
3. Description of the algorithm of decoding by the Adamar code.
4. Advantages of the Adamar code.
5. Disadvantages of Adamar code.

TOPIC 12
REED-MALLER CODES

Reed-maller codes

Reed-Maller codes are a family of linear noise-resistant codes (Table 25).

Further it will be consider binary codes of length *N* as Boolean (binary) functions from *N* variables.

Table 25. Reed-maller codes

v1v2v3	000	001	010	011	100	101	110	111
f_1	0	1	1	0	1	1	0	0
f_2	1	0	1	0	1	0	0	1
f_1f_2	0	0	1	0	1	0	0	0

Define a set *M* of all possible uninomials:

$$M = \{1, v_1, v_2, \ldots, v_m, v_1v_2, \ldots, v_{m-1}v_m, \ldots, v_1v_2\ldots v_m\} .$$

These functions are linearly independent, as is their vector representation.

A binary Reed-Maller code R(m.r) of degree *r* and length 2^m contains all linear combinations of vectors that are representations of uninomials of degree at most r from m variables.

Aalgorithms for first-order codes

There are also a number of efficient algorithms for first-order codes. Consider the code (1,3) (Table 26).

Its matrix is simply all quadruples from 1000 to 1111 in their binary representation.

Table 26. Reed-Muller R(1,3)

v1v2v3	000	001	010	011	100	101	110	111
1	1	1	1	1	1	1	1	1
v1	0	0	0	0	1	1	1	1
v2	0	0	1	1	0	0	1	1
v3	0	1	0	1	0	1	0	1

These vectors can be taken as rows of the forming matrix. This is the (2,4) d = 4 code, which is also an extended Hamming code (with parity check) and is intended for single error correction and double error detection (Table 27).

Table 27. Reed-Muller R(2,4)

v_1v_2 v_3v_4	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1
v_1	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	1
v_2	0	0	0	0	1	1	1	1	0	0	0	0	1	1	1	1
v_3	0	0	1	1	0	0	1	1	0	0	1	1	0	0	1	1
v_4	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1
v_1v_2	0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1
v_1v_3	0	0	0	0	0	0	0	0	0	0	1	1	0	0	1	1
v_1v_4	0	0	0	0	0	0	0	0	0	1	0	1	0	1	0	1
v_2v_3	0	0	0	0	0	0	1	1	0	0	0	0	0	0	1	1
v_2v_4	0	0	0	0	0	1	0	1	0	0	0	0	0	1	0	1
v_3v_4	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1

In the general case, the Reed-Muller code matrix R(m.r) contains *k* rows and *2^m* columns

$$k = 1 + C_m^1 + C_m^2 + \ldots + C_m^r$$

The minimum code distance R(r,m) is 2^{m-r}.

Four large groups (partially overlapping) of Reed-Maller codes can be distinguished:

R(0,m) – repetition codes of length 2^m, minimum coding distance *N.*

R(1,m) – noise-resistant codes of length 2^m, minimum code distance *d=N/2.*

R(m-1,m) – single parity check codes of length 2^m, minimum code distance *d*=2.

R(0,m) – extended Hamming codes of length 2^m, minimum code distance *d*=4.

The most interesting part of Reed-Maller codes is that for them there is an effective algorithm of decoding for any multiplicity of errors.

$$G = \begin{bmatrix} 1 & v_1 & v_2 & v_3 & v_1 v_2 & v_1 v_3 & v_2 v_3 \end{bmatrix}^T .$$

Write a vector of 7 input bits in the form

$$\mathbf{m} = (m_0, m_1, m_2, m_3, m_{12}, m_{13}, m_{23})$$

and the vector of output bits

$$\mathbf{c} = (c_0, c_1, c_2, c_3, c_4, c_5, c_6, c_7).$$

Coding and decoding

The coding operation in this case is multiplication

$$\mathbf{c} = \mathbf{mG}.$$

The decoding operation is performed as follows. For the input vector *r* the input bits are estimated first of all of "higher" order. In this case it is

$$m_{12}, m_{13}, m_{23} .$$

Then bits are evaluated to an order of magnitude lower, and so on. The following algorithm is used for evaluation.

$$c_0 = m_0, \quad c_1 = m_0 + m_1, \quad c_2 = m_0 + m_2, \quad c_3 = m_0 + m_1 + m_2 + m_{12}.$$

Adding these bits together, get

$$c_0 + c_1 + c_2 + c_3 = m_{12} .$$

Similarly, by adding up the following bits, get:

$$c_4 + c_5 + c_6 + c_7 = m_{12} .$$

Thus, using the obtained sequence

$$\hat{m}_{12} = r_0 + r_1 + r_2 + r_3 ,$$

$$\hat{m}_{12} = r_4 + r_5 + r_6 + r_7 .$$

After all the "higher" order bits have been evaluated, the input vector is recalculated so as to ignore these bits:

$$\mathbf{r'} = \mathbf{r} - \mathbf{m}_2 \mathbf{G}_2 .$$

Then the same procedure is carried out for all bits up to the most least

$$m_1 = c_0 + c_1,$$

$$m_1 = c_2 + c_3,$$

$$m_1 = c_4 + c_5,$$

$$m_1 = c_6 + c_7.$$

Test questions on the topic

1. Give the definition of a code.
2. Description of the algorithm of encoding by the Reed-Maller code.
3. Description of the algorithm of decoding by the Reed-Maller code.
4. Advantages of the Reed-Muller code.
5. Disadvantages of Reed-Maller code.

TOPIC 13
CONVOLUTIONAL CODES. TRELLIS DIAGRAMS

The concept of convolutional codes

This lecture discusses an important widely used class of codes called convolutional codes. In particular, these codes are used by the 802.11 standard and in satellite communications.

Convolutional codes work with bits like all previously discussed codes, but, unlike block codes in systematic form, the sender does not send the message as a sequence of information bits interspersed with check bits. In convolutional codes, only the check bits are sent.

The encoder uses a sliding window to compute *r>1* check bits by combining different sets of bits within this window. The combination is a simple summation in F2 (sum mod two, exclusive OR), as in previous lectures. Unlike block codes, the window overlaps its previous position by shifting 1 bit each time, as shown in the figure. The size of the window in bits is called the length of the coding constraint of a convolutional code. The larger this value, the greater the number of check bits that will be affected by each input symbol. Since only the check bits are transmitted over the channel, a larger code limit length corresponds to a greater corrective ability of the code. In return, the decoding process of codes with a large code-limit length slows down, which does not allow an unlimited increase of this length.

If a convolutional produces *r* check bits per window and moves the window forward one bit per so, its ratio is *1/r*. The larger the value of *r*, the higher the noise immunity of the code, but more bits are transmitted per clock and a larger channel width is used for transmission. In practice, r and the length of the code constraint restriction length is tried to be chosen as small as possible, providing sufficient noise immunity. Will denote the length of the coding constraint *K*.

An example of a convolution code with two parity bits per message bit (r = 2) and constraint length (shown in a rectangular window) K = 3 is shown in Figure 11.

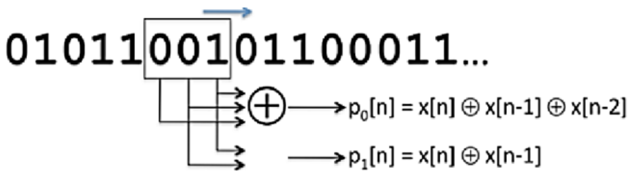

Figure 11. An example of a convolutional code with two parity bits per message bit (*r* = 2) and constraint length (shown in the rectangular window) *K* = 3

Encoding process

The encoder processes *K* bits per clock cycle and produces *r* check bits according to a chosen function that processes different sets among these *K* bits. (Assume that each message contains *K-1* zero bits at the beginning in order for the algorithm to function correctly.) One example is shown in the figure and corresponds to a scheme with *K=3* and *r=2.* The encoder produces r bits, which are then sent, moves the window 1 to the right, and repeats the process.

Encoding equations

The example in the figure demonstrates one set of coding equations that defines how the check bits are generated based on the *X* information bits. In our case, the equations are as follows:

$p_0[n]=x[n]+x[n-1]+x[n-2]$,

$p_1[n]=x[n]+x[n-1]$.

Example of coding equations for a code with *r=3*

$p_0[n]=x[n]+x[n-1]+x[n-2]$,

$p_1[n]=x[n]+x[n-1]$,

$p_2[n]=x[n]+ x[n-2]$.

In general, it can be seen that each coding equation is a combination of information bits and a forming polynomial *g*. In the first example, the coefficients of the forming polynomial are (1,1,1) and (1,1,0), and in the second (1,1,1), (1,1,0) and (1,0,1).

Denote g_i K-bit forming polynomial for the check bit p_i. Write p_i as

$$p_i[n] = \sum_{j=0}^{k-1} g_i[j]x[n-j] \bmod 2$$

The form of the above equation is a convolution of g and x – hence the term "convolution codes". The number of forming polynomials is equal to the number of generated check bits r for each sliding window.

Example.

Consider two forming polynomials

$g_0 = (1,1,1), g_1 = (1,1,0).$

In the transmitted message, X = [1,0,1,1,1,...] (assume $x[n] = 0\ V_n<0$). Then the check bits will be as follows:

$p_0[0]=(1+0+0)=1, p_1[0]=(1+0)=1,$

$p_0[1]=(0+1+0)=1, p_1[1]=(0+1)=1,$

$p_0[2]=(1+0+1)=0, p_1[2]=(1+0)=1,$

$p_0[3]=(1+1+0)=0, p_1[3]=(1+1)=0.$

And the final message transmitted through the channel looks like [1,1,1,1,1,1,0,1,0,0,0,...].

There are several forming polynomials, but the way they are formed is beyond the scope of this lecture.

A few examples are given below.

Table 28. Example of forming polynomials for r = 2

Restriction length	G1	G2
3	110	1110
4	1101	1110
5	11010	11101
6	110101	111011
7	110101	110101
8	110111	1110011
9	110111	111001101
10	110111001	1110011001

Two ways of representing a convolutional coder

Block diagram

The Figure 12 shows the same encoder as shown in the block diagram view.

The input bits of the message x[n] come from the left, the "black box" calculates the values of the check bits based on the input bits and the "encoder state" (previous input bits). After all r output bits are generated, the encoder state is shifted by 1, x[n] takes the place of x[n-1], x[n-1] takes the place of x[n-2], and so on, and the last bit x[n-K+1] is reset.

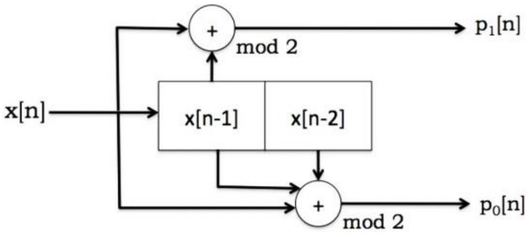

Figure 12. Block diagram of convolutional code

Finite state machine

The finite state machine (Figure 13) is identical for all codes of a given length K and the number of states is always 2^K. Only the values of the output bits p_i depend on the specific number and coefficients of the selected forming polynomials.

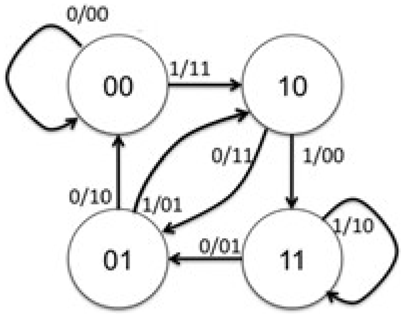

Figure 13. Finite state machine of convolutional code

A finite state machine is a way of specifying the encoding process. The encoder starts in the initial state and processes one bit per clock cycle. For each bit of the message, the encoder state changes (or stays the same) according to the value of the input bit. At the same time, a certain set of bits is fed to the output.

Decoding problem.

The decoding problem is to find a sequence of input bits that best matches the received (possibly distorted) state.

For definition of such sequence the decoder based on search of maximum likelihood is used. It is possible to show, that the best will be such sequence of input information bits to which corresponds the closest to the received on Hamming distance sequence of output bits.

However, determination of such an input sequence is in general a non-trivial task.

For example, the Table 30 shows the values of check bits for a convolutional code with k = 3 and r = 2. If the receiver has received 111011000110, it is obvious that an error has occurred during transmission, because this message does not correspond to any possible sequence. The last column shows the Hamming distance values for each of the possible transmitted sequences.

Table 30. Example of forming polynomials for r = 2

Input message	Output message	Received message	Hamming distance
0000	000000000000	111011000110	7
0001	000000111110		8
0010	000011111000		8
0011	000011010110		4
0100	001111100000		6
0101	001111011110		5
0110	001101001000		7
0111	001100100110		6
1000	111110000000		4
1001	111110111110		5
1010	111101111000		7
1011	111101000110		2
1100	110001100000		5
1101	110001011110		4
1110	110010011000		6
1111	110010100110		3

It is obvious, that such way of decoding is inapplicable, as for a message of length N the number of possible variants becomes equal to 2^N. For decoding of such code trellis-diagrams are used.

Trellis-diagrams

A trellis-diagram is a graph whose nodes are divided into groups representing time slices, and each node is connected with at least one node preceding it in time and one node following it in time (Figure 14).

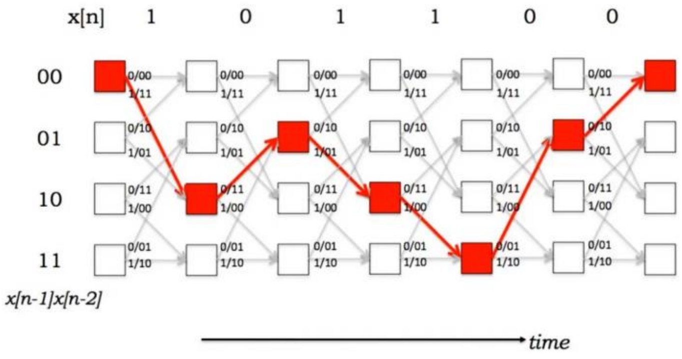

Figure 14. Trellis diagram of the convolutional code

Viterbi's algorithm for convolutional codes is to minimize the path metric over all possible routes in a Trellis diagram.

Initially, the starting state corresponds to the metric value equal to zero, the other states - to the infinitely large value (Figure 15).

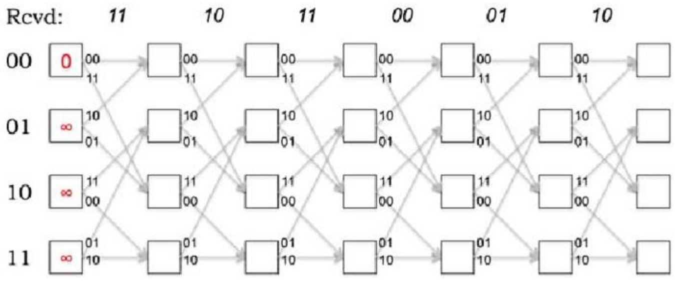

Figure 15. Initial state of the Trellis diagram

At each step the weight of an edge is calculated as the Hamming distance between the fragment of the decoded sequence and the output fragment of the corresponding transition in the finite state machine (Figure 16.).

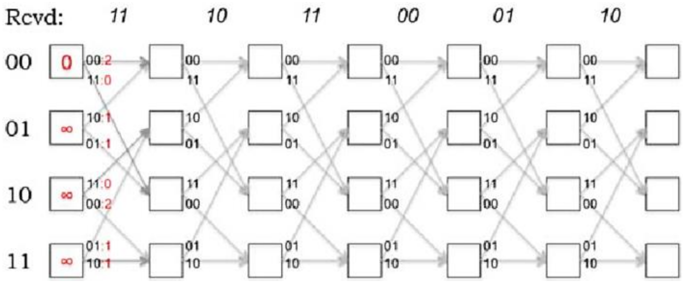

Figure 16. Calculation of the weight of edges of the Trellis diagram

The values of the metrics for each vertex (state) in the next step are determined as a minimum of the sum of the weight of the edge entering that vertex and the value at the vertex from which that edge originates (Figure 17).

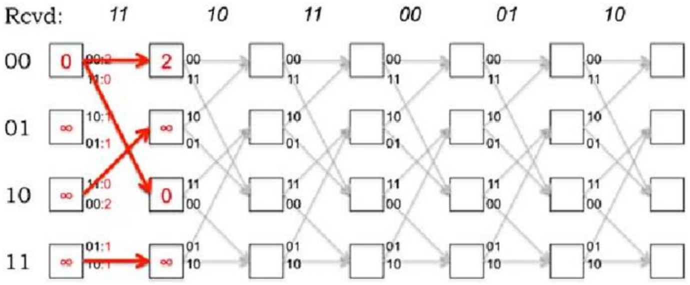

Figure 17. Calculation of vertex values

In this way the whole Trellis diagram is filled in. For the last column of states, a path is defined that provides the corresponding minimum path metric to each state (Figure 18).

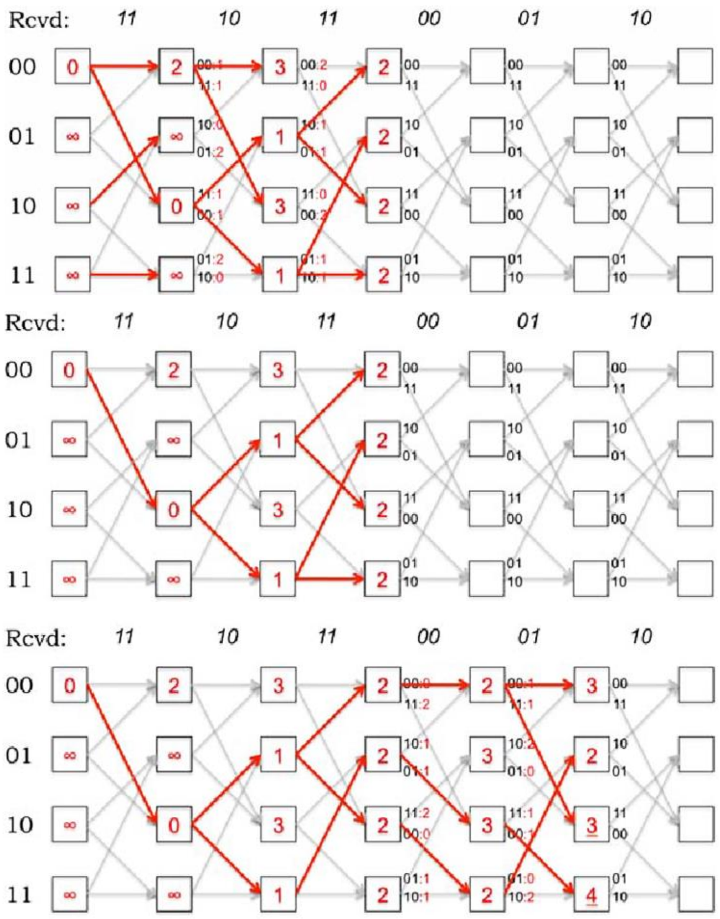

Figure 18. Filling in the Trellis diagram

The values of the input bits are determined for the smallest value of the path metric. If several states contain the minimum value of the metric, reliable decoding of the received message is not possible (Figure 19).

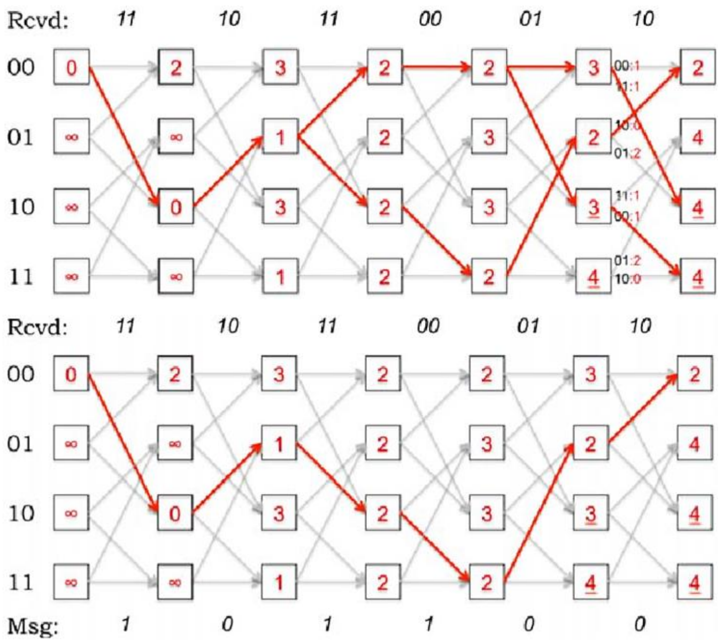

Figure 19: Determining the path for the minimum value of the metric

Test questions on the topic

1. Characteristics of tree codes.
2. How to encode a sequence using the impulse response?
3. Characteristics of coding with convolutional codes using a lattice diagram.
4. What are the known algorithms for decoding convolutional codes?
5. Characteristics of the coding algorithm using a trellis diagram.
6. Characteristics of the Viterbi decoding algorithm.

TOPIC 14
MODELS OF DETERMINISTIC SIGNALS

Frequency representation of periodic signals

Consider the representation of deterministic signals with using as basis functions

$$\varphi\,(t) = e^{pt},$$

$$p = \pm j\omega.$$

Such a representation is called a ***Fourier transform***. By virtue of Euler's formula

$$\cos\omega t = \left(e^{j\omega t} + e^{-j\omega t}\right)/2$$

Fourier transform makes it possible to represent a complex signal as a sum of harmonics.

Suppose that the function *u(t)*, describing the deterministic realisation of the signal on the interval $[t_1, t_2]$, satisfies the Dirichlet conditions (is continuous or has a finite number of breakpoints of the first kind, and also a finite number of a finite number of breakpoints of the first kind, as well as a finite number of extrema) and repeats with period $T = t_2 - t_1, t \in (-\infty, +\infty)$.

Using the above basis function

$$\varphi\,(t) = e^{\pm j\omega t},$$

the function *u(t)*can be represented as

$$u(t) = \frac{1}{2}\sum_{k=-\infty}^{\infty} A(jk\omega_1)\cdot e^{jk\omega_1 t}, \tag{45}$$

$$A(jk\omega_1) = \frac{2}{T}\int_{t_1}^{t_2} u(t)\cdot e^{-jk\omega_1 t}\,dt, \tag{46}$$

and the period

$$T = t_2 - t_1 = 2\pi/\omega_1.$$

The coefficients $A(jk\omega_1)$ in this spectral representation are called the complex spectrum of the periodic signal *u(t)*, and the value $A(jk\omega_1)$ for a particular *k* is called the complex amplitude.

The complex spectrum is discrete, but by replacing $k\omega_1 = \omega$ for it it is possible to construct an envelope

$$A(j\omega) = \frac{2}{T}\int_{t_1}^{t_2} u(t)\cdot e^{-j\omega t}\,dt\,. \tag{47}$$

Like any complex number, the complex spectrum can be represented:

- *in exponential form*:

$$A(jk\omega_1) = A(k\omega_1)\cdot e^{-j\varphi(k\omega_1)}\,, \tag{48}$$

where $A(k\omega_1)$ is the amplitude spectrum and $\varphi(k\omega_1)$is the phase spectrum (also discrete);

- in algebraic form:

$$A(jk\omega_1) = A_k - jB_k\,, \tag{49}$$

where

$$A_k = \frac{2}{T}\int_{t_1}^{t_2} u(t)\cdot\cos(k\omega_1 t)\,dt\,,\ B_k = \frac{2}{T}\int_{t_1}^{t_2} u(t)\cdot\sin(k\omega_1 t)\,dt\,.$$

The (49) is obtained from (46) by substituting by the formula Euler's formula:

$$e^{-jk\omega_1 t} = \cos(k\omega_1 t) - j\sin(k\omega_1 t)\,.$$

It is clear that

$$A(k\omega_1) = \sqrt{A_k^2 + B_k^2} \qquad \varphi(k\omega_1) = arctg(B_k / A_k)\,.$$

At *k = 0*, obtain the equality for the constant component of the signal:

$$\frac{A_0}{2} = \frac{1}{T}\int_{t_1}^{t_2} u(t)\,dt\,. \tag{50}$$

By combining the complex-conjugate components in (45) one can obtain the Fourier series in trigonometric form:

$$u(t)=\frac{A_0}{2}+\frac{1}{2}\sum_{k=1}^{\infty}\left[A(jk\omega_1)\cdot e^{jk\omega_1 t}+A(-jk\omega_1)\cdot e^{-jk\omega_1 t}\right]=$$

$$=\frac{A_0}{2}+\frac{1}{2}\sum_{k=1}^{\infty}\left[A(k\omega_1)\cdot e^{j[k\omega_1 t-\varphi(k\omega_1)]}+A(k\omega_1)\cdot e^{-j[k\omega_1 t-\varphi(k\omega_1)]}\right]=$$

$$=\frac{A_0}{2}+\sum_{k=1}^{\infty}A(k\omega_1)\cos(k\omega_1 t-\varphi(k\omega_1)).$$

(51)

The spectra of amplitudes – $A(k\omega_1)$ and phases – $\varphi(k\omega_1)$ can be represented by spectral diagrams as a set of lines, each of which corresponds to a certain frequency (one of the summands). Therefore, these spectra are called linear spectra. Signals, whose line spectra include harmonics of multiple frequencies, are called almost periodic.

Frequency representation of non-periodic signals

Suppose that the function corresponding to the real non-periodic signal function *u(t)* satisfies the conditions Dirichlet conditions and is absolutely integrable:

$$\int_{-\infty}^{\infty}|u(t)|\cdot dt<\infty .$$

Then the spectral representation of the non-periodic signal *u(t)* can be can be constructed by increasing the period of the periodic signal to infinity. For this purpose, proceed as follows.

Substitute the expression (46) for the complex amplitude $A(jk\omega_1)$ of the periodic signal into (45). Taking into account that $T=2\pi/\omega_1$ have

$$u(t)=\frac{1}{2}\sum_{k=-\infty}^{\infty}\left[\frac{\omega_1}{\pi}\int_{t_1}^{t_2}u(t)\cdot e^{-jk\omega_1 t}dt\right]\cdot e^{jk\omega_1 t} . \qquad (52)$$

Next, carry out the limit transition at $T\to\infty$. In this case the sum goes to the integral, $\omega_1=\Delta\omega\to d\omega$, $k\omega_1\to\omega$. As a result obtain:

$$u(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty}\left[\int_{-\infty}^{\infty}u(t)\cdot e^{-j\omega t}dt\right]\cdot e^{j\omega t}d\omega .$$

Introducing in the last equality for the integral in square brackets notation $S(j\omega)$, write down a pair of Fourier transforms:

$$u(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty}S(j\omega)\cdot e^{j\omega t}d\omega , \qquad (53)$$

$$S(j\omega)=\int_{-\infty}^{\infty}u(t)\cdot e^{-j\omega t}dt . \qquad (54)$$

The complex function $S(j\omega)$ is called the complex spectral density or spectral response. Also as in the case of a periodic signal, for a non-periodic signal the following representations of the spectral characteristic:

a) *The exponential form*:

$$S(j\omega)=S(\omega)\cdot e^{-j\varphi(\omega)} , \qquad (55)$$

where $S(\omega)= S(j\omega)$ is the spectral density of amplitudes, and $\varphi(\omega)$ is the phase spectrum;

b) algebraic form obtained from (54) by substituting

$$e^{-j\omega t}=\cos(\omega t)-j\sin(\omega t)):$$

$$S(j\omega)=A(\omega)-jB(\omega), \qquad (56)$$

where

$$A(\omega)=\int_{-\infty}^{+\infty}u(t)\cdot\cos(\omega t)dt ,\ B(\omega)=\int_{-\infty}^{+\infty}u(t)\cdot\sin(\omega t)dt . \qquad (57)$$

In this case

$$S(\omega)=|S(j\omega)|=\sqrt{|A(\omega)|^2+|B(\omega)|^2} ,\quad \varphi(\omega)=arctg\left[B(\omega)/A(\omega)\right].$$
(58)

Getting

$$u(t)=\frac{1}{2\pi}\int_{-\infty}^{\infty}S(\omega)\cdot e^{j[\omega t-\varphi(\omega)]}d\omega=$$

$$\frac{1}{2\pi}\left[\int_{-\infty}^{\infty}S(\omega)\cdot\cos[\omega t-\varphi(\omega)]d\omega+j\int_{-\infty}^{\infty}S(\omega)\cdot\sin[\omega t-\varphi(\omega)]d\omega\right].$$

The second integral of the odd function is equal to zero, and the first integral (due to the parity of the integrand) can be written only in due to the parity of the integrand) can be written only for positive frequencies. Thus, obtain the trigonometric form of the Fourier series:

$$u(t)=\frac{1}{\pi}\int_{0}^{\infty}S(\omega)\cdot\cos[\omega t-\varphi(\omega)]d\omega, \tag{59}$$

that allows for a clear physical interpretation.

Finally, consider another interesting property. For a function *u(t)* defined on the interval $[t_1,t_2]$ can write

$$S(j\omega)=\int_{t_1}^{t_2}u(t)\cdot e^{-j\omega t}dt. \tag{60}$$

Comparing the right parts of (47) and (60), it is easy to see that there is equality

$$A(j\omega)=\frac{2}{T}\cdot S(j\omega),$$

in other words, from the $S(j\omega)$ of a single pulse it is possible to construct a linear spectrum of their periodic sequence.

Relationship between the duration of signals and the width of their spectra

Suppose that a signal *u(t)* of a certain duration has a spectral characteristic $S(j\omega)$. Find the corresponding characteristic $S_\lambda(j\omega)$ for the signal $u(\lambda t)$, the duration of which has been changed λ times

$$S_{\lambda}(j\omega)=\int_{-\infty}^{\infty}u(\lambda t)\cdot e^{-j\omega t}dt=\frac{1}{\lambda}\int_{-\infty}^{\infty}u(\tau)\cdot e^{-j\frac{\omega\tau}{\lambda}}d\tau=\frac{1}{\lambda}S\left(j\frac{\omega}{\lambda}\right), \qquad (61)$$

where $\tau=\lambda t$

From (61) it can be seen that the spectrum of the signal shortened (lengthened) in λ times is λ times wider (narrower). At the same time, the coefficient $1/\lambda$ changes only the amplitudes of harmonics and does not affect the width of the spectrum.

The specified property is connected with the fact that the variables t and ω are included in the exponent of the exponential function of the forward and inverse Fourier transform in the form of a product. It follows that the duration of the signal and the width of its spectrum cannot be simultaneously limited to finite intervals. In particular, the relation:

$$\Delta t\cdot\Delta f=Const,$$

where Δt is the pulse duration, Δf is the spectrum width.

Test questions on the topic

1. What are the characteristics of the spectrum of a periodic signal?
2. How can the spectrum of a periodic signal be interpreted energetically?
3. What is the Fourier transform?
4. How can the spectrum of a non-periodic signal be obtained directly from the spectrum of the corresponding periodic signal?
5. How can the spectrum of a non-periodic signal be determined energetically?
6. How does the spectrum of a non-periodic signal differ from the spectrum of a periodic signal?

TOPIC 15 RECONSTRUCTING A SIGNAL FROM ITS DISCRETE VALUES

Formulation of discretization and reconstruction problems

Signal discretization is the transformation of a function of a continuous argument into a function of discrete time. It consists in replacing the continuous signal *u(t)* by a set of coordinates:

$$\left[c_1, c_2, \ldots, c_N\right] = A\left[u(t)\right], \tag{62}$$

where $A[\cdot]$ is some operator.

From the point of view of simplicity of implementation it is advisable to use linear operators. In particular, to determine the coordinates of the signal it is convenient to use the ratio

$$c_i = Au(t) = \int_T \varphi_i(t) \cdot u(t) \cdot dt, \quad i = \overline{1, N}, \tag{63}$$

where

$$\varphi_i(t), \quad i = \overline{1, N}$$

– given basis (in particular, orthogonal) functions can be used.

Discretization according to the ratio (63), due to the use of the integration operation, has high noise immunity.

However, there is a delay of the signal for the integration time *T*. Therefore, more often the discretization is reduced to replacement the signal by a set of its instantaneous values (samples):

$$c_i = u(t_i). \tag{64}$$

This is achieved by using the delta function in (63):

$$\varphi_i(t) = \delta(t - t_i).$$

Representation of a continuous signal by a set of equidistant samples in the form of a lattice function:

$$u_g(t) = \sum_{k=-\infty}^{\infty} u(t) \cdot \delta(t - k\Delta t) \tag{65}$$

– is the most common type of discretization. The function $u_g(t)$ is equal to $u(k\Delta t)$ at points $t = k\Delta t$ and zero at other points. If the step discretization step

$$\Delta t_i = t_i - t_{i-1} = Const$$

– the discretization is called uniform. The summation limits in (65) can be set finite, based on the conditions of physical realisability.

Usually discretisation is carried out for the purpose of further conversion of the signal into digital form. As a result of digital coding of a discrete signal, its quantisation takes place – replacement of instantaneous values of the signal with the nearest allowed ones at the appropriate moments of time. In this case the signal turns out to be discrete both in time and in a set of values.

An important advantage of digital form of signal representation is that many quantisation levels can be represented by a small number of digits. Besides, at representation in digital form complex algorithms of processing on computer can be realised, including construction of codes detecting and correcting errors.

When the discrete signal is subsequently used for control purposes, it is usually restored using some specified operator:

$$u^*(t) = B[c_1, c_2, \ldots, c_N], \tag{66}$$

If the discretisation was performed by an operator of the form (63) with using orthogonal functions

$$\varphi_i(t), \quad i = \overline{1, N},$$

for the reconstruction of the continuous signal can be used operator

$$u^*(t) = \sum_{i=1}^{N} c_i \varphi_i(t). \tag{67}$$

The following criteria are used to assess the quality of signal recovery The following criteria are used.

Uniform approximation (criterion of the largest deviation):

$$\max_{t \in T} \left| u(t) - u^*(t) \right| \le \varepsilon_{\partial on}.$$

A uniform approximation for an ensemble of realisations:

$$\sup_{u_i(t)\in U} \left| u_i(t) - u_i^*(t) \right| \le \varepsilon_{\partial on} .$$

The criterion of standard deviation (SD):

$$\sigma = \sqrt{\frac{1}{T}\int_T \left| u(t) - u^*(t) \right|^2 dt} \le \sigma_{\partial on} .$$

SD for the ensemble of N realisations – σ_Σ is calculated by averaging over the ensemble taking into account probabilities realisations

$$p_i\,,\ i = \overline{1, N}:$$

$$\sigma_\Sigma = \sum_{i=1}^{N} p_i \sigma_i \le \sigma_{\Sigma,\partial on} .$$

Integral criterion:

$$\varepsilon = \frac{1}{T}\int_T \left| u(t) - u^*(t) \right| dt \le \varepsilon_{\partial on} .$$

The value of the integral criterion $\varepsilon_\Sigma$for N realisations is calculated by averaging over the ensemble:

$$\varepsilon_\Sigma = \sum_{i=1}^{N} p_i \varepsilon_i .$$

A probabilistic criterion is also used, defined as the following an acceptable level of probability $P_{\partial on}$ that the error will not exceed an acceptable value. permissible value $\varepsilon_{\partial on}$:

$$\mathrm{P}\left\{ \left| u(t) - u^*(t) \right| \le \varepsilon_{\partial on} \right\} \le \mathrm{P}_{\partial on} .$$

The use of one of these criteria in each case depends on the requirements of the system and the available resources.

Kotelnikov's theorem

As noted above, uniform discretization is the most widely used. In this case, a model of the signal in the form of an ergodic random process, each realisation of which is a function with a limited spectrum, is used to select the value of the sampling step. The theoretical basis of this approach is the following Kotelnikov theorem.

Any function *u(t)*, admitting a Fourier transform and having a continuous spectrum bounded by the frequency band from 0 to

$$f_c = \omega_c / 2\pi ,$$

is completely determined by a discrete series of its instantaneous values counted at time intervals

$$\Delta t = 1/(2 \cdot f_c) = \pi / \omega_c .$$

Proof. Since by assumption the function *u(t)* has a bounded spectrum – $S(j\omega) = 0$ at $\omega > \omega_c$, equality can be written

$$u(t) = \frac{1}{2\pi} \int_{-\omega_c}^{+\omega_c} S(j\omega) \cdot e^{j\omega t} d\omega . \quad (68)$$

The function $S(j\omega)$ on a finite interval $[-\omega_c, \omega_c]$ can be decomposed into a Fourier series.

The *pair of Fourier transforms* is written by assuming $S(j\omega)$ to be conditionally continuous with period $2\omega_c$ and formally replacing in (45), (46) t by ω, and ω_1 by $\Delta t = \pi/\omega_c$:

$$S(j\omega) = \frac{1}{2} \sum_{-\infty}^{+\infty} A_k \cdot e^{jk\Delta t\omega} , \quad (69)$$

$$A_k = \frac{1}{\omega_c} \int_{-\omega_c}^{\omega_c} S(j\omega) \cdot e^{-jk\Delta t\omega} d\omega . \quad (70)$$

Compare relations (70) and (68), having previously rewritten equality (68) for discrete moments of time $t_k = k\Delta t$:

$$u(k\Delta t) = \frac{1}{2\pi} \int_{-\omega_c}^{\omega_c} S(j\omega) \cdot e^{j\omega k\Delta t} d\omega . \quad (71)$$

$$A_k = \frac{2\pi}{\omega_c} \cdot u(-k\Delta t) . \quad (72)$$

Substituting the value of A_k from (72) into (69) can be written:

$$S(j\omega)=\frac{\pi}{\omega_c}\sum_{-\infty}^{+\infty}u(-k\Delta t)\cdot e^{jk\Delta t\omega}.$$

In the last equality the minus sign in front of *k* can be changed to the minus sign in front of *k* can be reversed, because the summation is carried out on both positive and negative numbers:

$$S(j\omega)=\frac{\pi}{\omega_c}\sum_{-\infty}^{+\infty}u(k\Delta t)\cdot e^{-jk\Delta t\omega}. \tag{73}$$

Now substitute $S(j\omega)$ from (73) into (68):

$$u(t)=\frac{1}{2\omega_c}\int_{-\omega_c}^{+\omega_c}\left(\sum_{-\infty}^{+\infty}u(k\Delta t)\cdot e^{-jk\Delta t\omega}\right)\cdot e^{j\omega t}d\omega=\frac{1}{2\omega_c}\sum_{-\infty}^{+\infty}u(k\Delta t)\int_{-\omega_c}^{+\omega_c}e^{j\omega(t-k\Delta t)}d\omega.$$

After integration in the right part of the last equality obtain

$$u(t)=\sum_{-\infty}^{+\infty}u(k\Delta t)\frac{\sin\omega_c(t-k\Delta t)}{\omega_c(t-k\Delta t)}=\sum_{-\infty}^{+\infty}u(k\Delta t)\operatorname{sinc}\omega_c(t-k\Delta t). \tag{74}$$

So, have expressed the function *u(t)* through its discrete values taken at times $t_k = k\Delta t$. Suppose $t = n\Delta t$, where *n* is some integer. Since $\Delta t=\pi/\omega_c$, for any integers *k* and *n*

$$\omega_c(n\Delta t-k\Delta t)=(n-k)\omega_c\Delta t=(n-k)\pi.$$

Hence

$$\frac{\sin\omega_c(t-k\Delta t)}{\omega_c(t-k\Delta t)}=\begin{cases}1, & t=k\Delta t,\\ 0, & t=n\Delta t,\quad n\neq k.\end{cases}$$

This means that the values of the function *u(t)* at the moments of time *k* $t_k=k\Delta t$ are nothing but its samples. Thus, a function with a limited spectrum can be represented by the series (74), the coefficients of which are samples of the function values taken at time intervals

$$\Delta t=\frac{\pi}{\omega_c}=\frac{1}{2\cdot f_c}. \tag{75}$$

Based on this, the following scheme can be presented transmit-receive. On the transmitting side, the instantaneous values of the signal *u(t)* are transmitted in intervals Δt, determined by the ratio (75). On the receiving side the sequence of pulses are passed through an ideal low-pass filter with a cut-off frequency f_c. Then at long transmission theoretically the signal at the output of the filter will accurately reproduce the transmitted continuous signal *u(t)*.

In reality, the real signal always has a finite duration, hence its spectrum is unlimited. The error arises not only due to the forced limitation of the spectrum, but also due to the finite number of samples in the time interval *T*, which according to the theorem will be $N = 2\, f_c\, T$.

The model of a signal with a limited spectrum has also a fundamental theoretical inconvenience. It cannot reflect the main property of a signal - the ability to carry information. The fact is that the behaviour of a function with a limited spectrum can be accurately predicted on the entire time axis if it is accurately known over any small time interval.

Nevertheless, Kotelnikov's theorem has an important applied significance. In practice, the spectrum width fc is defined as the frequency interval outside of which the spectral density is less than some given value. Under this assumption, the function on the interval *T* is determined with some degree of accuracy (depending on the accuracy of the spectral density representation) by means of $N= 2\, f_c\, T$ samples, the general meaning of Kotelnikov's theorem is preserved.

Quantisation of signals

A physically realisable continuous signal *u(t)* is always limited to some range $[u_{min}\,,\ u_{max}]$. In addition, a device can often reproduce only a finite set of fixed signal values from this range. In particular, a continuous scale of instantaneous values $u_n = u_{max} - u_{min}$ may be divided into n identical intervals, and the allowed values of the signal are equidistant from each other, then it speaks about uniform quantisation. If the constancy of the interval (quantisation step) is not observed, then quantisation is non-uniform.

From the set of instantaneous values belonging to the *i*-th interval (quantisation step), only one value u_i' is allowed (*i*-th quantisation level), and any other value is rounded to u_i'. Suppose uniform quantisation with step

$$\Delta = (u_{\max} - u_{\min}) / n$$

is performed such that the quantisation levels u_i' are placed in the middle of each step. It is clear that in this case the error quantisation error is minimal and does not exceed $0{,}5\Delta$. Define for this standard deviation (SD) of the quantisation error of quantisation error.

In general case standard deviation of quantisation error σ_i for i - step is determined by the ratio

$$\sigma_i = \sqrt{\int\limits_{u_{i-1}}^{u_i} \left(u(t) - u_i'\right)^2 w(u)\,du}\ , \tag{76}$$

where $w(u)$ is the probability density function of instantaneous values of the signal U. If quantisation steps are small in comparison with the range of signal variation range, the density of $w(u)$ within each step can be be considered constant and equal, for example, $w(\ u_i'\ )$. Then, introducing a new variable $y=u(t)–u_i'$, for the specified method of quantisation in accordance with (76) it has

$$\sigma_i = \sqrt{w(u_i')\int\limits_{-\frac{\Delta_i}{2}}^{\frac{\Delta_i}{2}} y_i^2 dy_i} = \sqrt{w(u_i')\frac{\Delta_i^3}{12}}\ . \tag{77}$$

Given that $p\ (u_i')\ > 0$ and $\Delta_i > 0$ for all $i = 1, n$, according to (77) can be written the variance of quantisation error at the i-th step

$$\sigma_i^2 = \left[w(u_i')\Delta_i\right]\frac{\Delta_i^2}{12}\ . \tag{78}$$

It turns out that it is equal to the value $\Delta_i^2/12$, multiplied by the probability $w(u_i')\Delta_i$ of the instantaneous signal value falling within the given interval. The variance of the total error is defined as the mathematical expectation of the variances $\Delta_i^2/12$ at individual steps:

$$\sigma^2 = \sum_{i=1}^{n}\left[w(u_i')\Delta_i\right]\frac{\Delta_i^2}{12}\ .$$

If the intervals are the same – $\Delta_i = \Delta$ for all $i = 1, n$, taking into account the normalisation condition

$$\sum_{i=1}^{n}\left[w\left(u_i^{'}\right)\Delta\right]=1$$

get

$$\sigma^2=\frac{\Delta^2}{12}\sum_{i=1}^{n}\left[w\left(u_i^{'}\right)\Delta\right]=\frac{\Delta^2}{12}\,.$$

If the quantised signal is influenced by an interference, it can fall into the interval corresponding to another quantisation level. It is intuitively clear (and it can be strictly shown) that in the case when the interference ξ has a uniform distribution $w(\xi) = 1/a$, where $a/2$ is the amplitude of the interference symmetric with respect to the instantaneous value of the signal, the probability of incorrect quantisation of the signal sharply increases at $a > \Delta$. The impact of normally distributed interference with parameters $(0, \sigma^2)$ is equivalent to the impact of uniformly distributed interference at $a = 3\sigma$.

Test questions on the topic

1. Explain the processes of converting a continuous signal to digital?
2. The essence of sampling a continuous signal.
3. The essence of quantisation of a sampled signal.
4. What is called the quantisation error and how is it determined for a uniform quantiser?
5. What is called quantisation noise and how is it determined for a uniform quantiser?
6. What is called signal compounding and what transformations does it consist of?

REFERENCE

1. *Arndt, C.* Information Measures, Information and its Description in Science and Engineering. Springer Series: Signals and Communication Technology, 2004. 603 p.

2. *Cover, T.,* Thomas J.A. Elements of information theory (2-nd ed.) New York: Wiley-Interscience, 2006. 776 p.

3. *MacKay, D.J.C.* Information Theory, Inference, and Learning Algorithms. Cambridge: Cambridge University Press, 2003. 640 p.

4. *McEliece, R.* The Theory of Information and Coding. Cambridge, 2002. 410 p.

5. *Yeung, R.W.* A First Course in Information. Theory Kluwer Academic/Plenum Publishers, 2002. 431 p.

6. *Shannon C.E.* Mathematical Theory of Communication. BSTJ, Vol. 27. 1948. PP. 379–423, 623–656.

7. *Chitode, Dr. J. S.* Information Theory & Coding (Information, Source Coding & Channel Coding). Technical Publications, 2021. 95 p.

8. *Borda, M.* Fundamentals in Information Theory and Coding. 1 Springer, 2011. 504 p.

9. *Hamming, R.* Coding and information theory. Prentice-Hall, Englewood Cliffs, 1980. 255 p.

10. *Sklar, B.* Digital communications. Prentice-Hall, Englewood Cliffs, 1988. 1011 p.

11. *Van Trees, H.L.* Detection, Estimation and Modulation Theory, Part.I. John Wiley & Sons, Chichester, 1968. 716 p.

12. *Kotelnikov V.A.* Theory of potential noise immunity. Gosenergoizdat, 1956. 151 p.

Наукове видання

Карина ТРУБЧАНІНОВА
Олександр СЕРКОВ
Олег КАСІЛОВ

ТЕОРІЯ ІНФОРМАЦІЇ ТА КОДУВАННЯ

ПІДРУЧНИК ДЛЯ ПІДГОТОВКИ ІНОЗЕМНИХ СТУДЕНТІВ СПЕЦІАЛЬНІСТЬ: ЕЛЕКТРОННІ КОМУНІКАЦІЇ ТА РАДІОТЕХНІКА

Тех. редактор – Гобельовська Л. П.

Підписано до друку 20.10.2023 р.

Формат 60х84/16. Папір офсетний. Друк цифровий.
Ум. друк. арк. 6,74. Тираж 100 прим. Зам № 290.

Видавець: ТОВ «Видавництво Точка»
61024, м. Харків, вул. Максиміліанівська, 11.
Тел.: 0800-33-67-62
Свідоцтво суб'єкта видавничої справи:
Серія ДК № 6863 від 02.08.19 р.

Віддруковано: ТОВ «Друкарня Мадрид»
61024, м. Харків, вул. Максиміліанівська, 11.
Свідоцтво суб'єкта видавничої справи:
Серія ДК № 4399 від 27.08.12 р.
www.madrid.in.ua || info@madrid.in.ua

9 786177 856565

9 786177 856565