


Approximate Calculation of Information

Angel Kuri-Morales
Instituto Tecnológico Autónomo
de México

akuri@itam.mx

Agenda

- ◆ Coding
 - ◆ Statistical Theory of Information
 - ◆ Algorithmic Theory of Information
 - ◆ Contrasting SI and AI
 - ◆ Randomness
 - ◆ Metasymbols
 - ◆ Applying Evolutionary Computation
 - ◆ Lossless and Lossy Data Compression
 - ◆ Conclusions
- 
- A stylized, dark teal silhouette of a mountain range is positioned in the bottom right corner of the slide, partially overlapping the background.

Digital Encoding

- ◆ Data is expressed in a modern computer by encoding it digitally. There are many ways to encode such data and one of the main issues in present day telematics (in view of the need for efficient transmission in computer networks) is how to best achieve it.
- ◆ To determine what "best" encoding means it is necessary to measure the information contents in the data.

Binary Encoding

- ◆ In the binary system we may represent any number whatsoever

Example: 1101

$$1101 = 1 \times 2^3 + 1 \times 2^2 + 0 \times 2^1 + 1 \times 2^0$$

Weights:	8	4	2	1	
Symbols:	<div>1</div>	<div>1</div>	<div>0</div>	<div>1</div>	Sum
Value:	8	4	0	1	<div>13</div>

Positional binary representation

Encoding

- ◆ The same concept may be applied to a general base “b”
 - x : size, b : base, $\{x\}$: digits
 - n : number of integer digits
 - m : number of fractional digits

$$x = x_{n-1} \times b^{n-1} + \dots + x_1 \times b^1 + x_0 \times b^0 + x_{-1} \times b^{-1} + x_{-2} \times b^{-2} + \dots + x_{-m} \times b^{-m}$$

Natural Binary Encoding

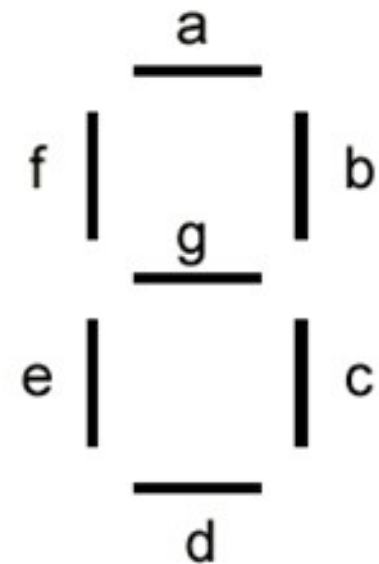
- ◆ The “natural” binary encoding assigns binary codes to positive integers
- ◆ The assigned word corresponds to the representation of the number in base 2

Symbol	Code
0	000
1	001
2	010
3	011
4	100
5	101
6	110
7	111

Non-positional Binary Encoding

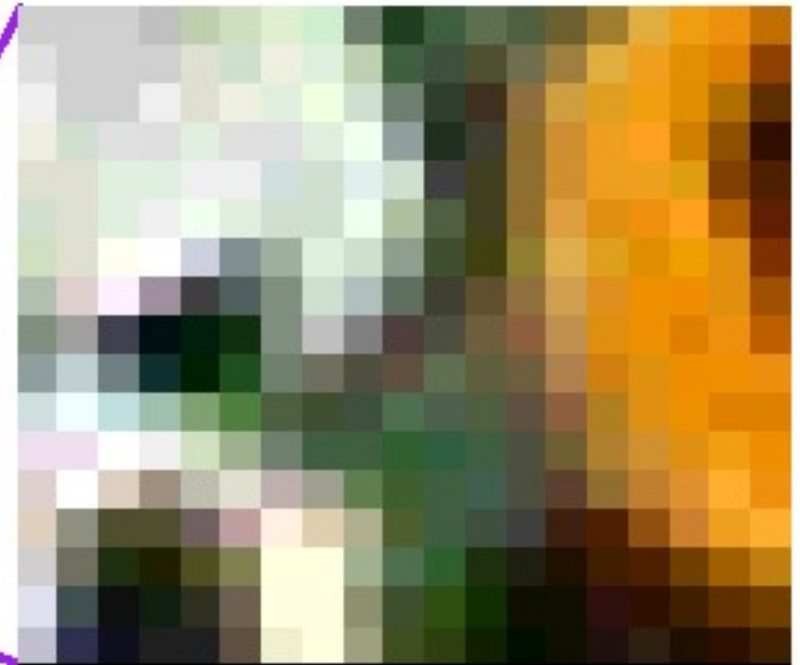
- ◆ Not all binary codes are positional

number	natural	Excess 3	2-of-5	7 segments abcdefg
0	0000	0011	00011	1111110
1	0001	0100	00101	0110000
2	0010	0101	00110	1101101
3	0011	0110	01001	1111001
4	0100	0111	01010	0110011
5	0101	1000	01100	1011011
6	0110	1001	10001	1011111
7	0111	1010	10010	1110000
8	1000	1011	10100	1111111
9	1001	1100	11000	1110011



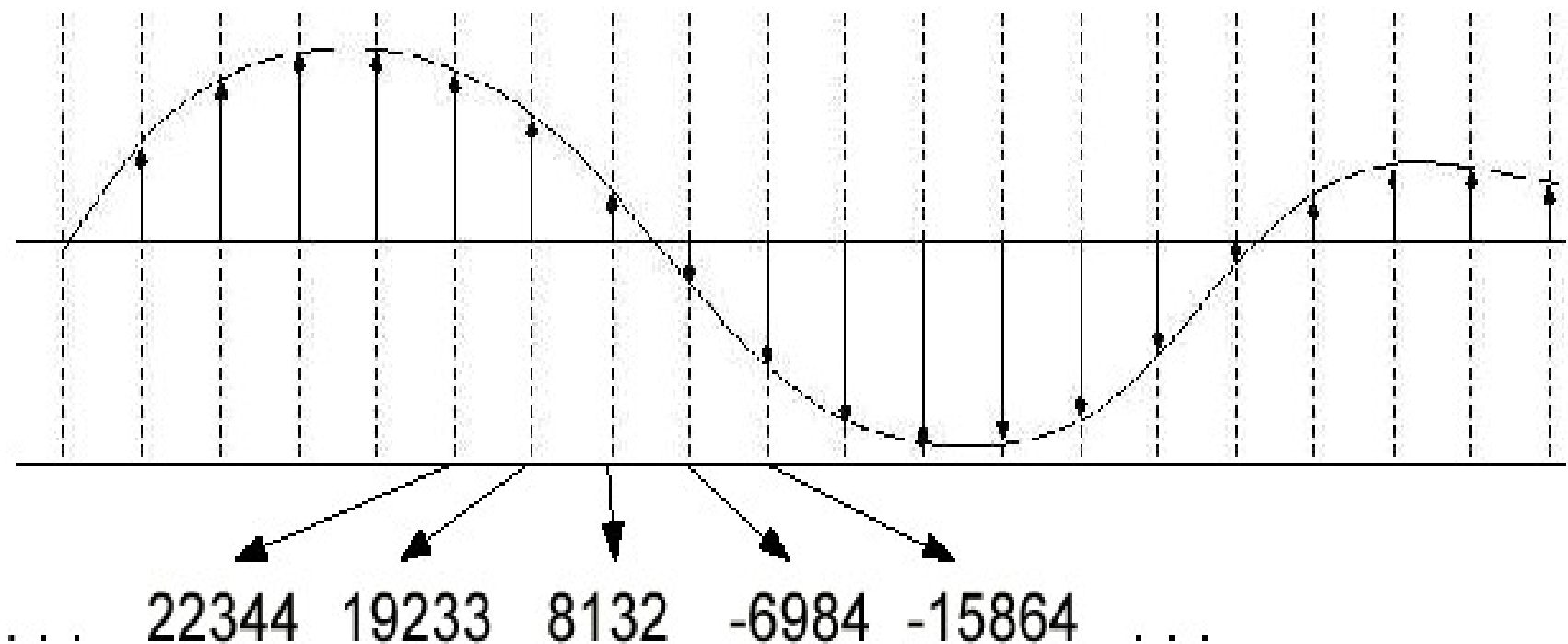
Encoding Images

- ◆ To represent images we must encode colors. A color is composed from the three primary colors in its different intensities. If we use 24 bits, 8 of these are assigned to each primary color.



Encoding Sounds

- ◆ A sound must be sampled to digitally represent it. Typical sampling rate is 44,100 hz. with 16 bits per sample. 2:30 minutes of sound map into 13,230,000 bytes!



Some Simple Conclusions

- ◆ If something may be quantified, then it may be encoded.
- ◆ If something may be encoded, then we may do it in some binary code.
- ◆ Therefore:
Any kind of data may be expressed with some binary code.

Transmitting Data

- ◆ One of the consequences of the above is that data may be shared by many users if an adequate transmission media is found.
- ◆ In the world of today, of course, Internet has become the preferred media to share data.
- ◆ But, of course, *sharing* means *transmitting* and that means *time* and *storage* resources must be made available.

Optimizing Representation

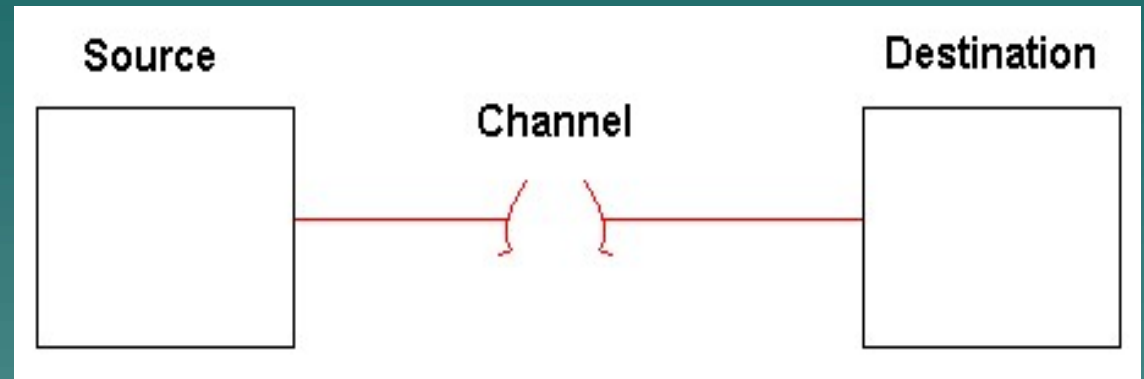
- ◆ This brings us to the crux of our talk:
 - How to best represent arbitrary data in order to minimize the needed resources?
- ◆ To answer this question we must, first, determine the amount of information present in the data.

Statistical Information

- ◆ Shannon took a statistical approach to answer this question.
- ◆ He assumed that a message is to be transmitted between a source and a destination.
- ◆ He also assumed that the source is a black box but that we know the probability distribution of the symbols coming out of it

The Source – Destination Model

- ◆ In Shannon's Proposal, symbols "flow"



from the source to the destination with some known probability.

- ◆ There is, hence, a probability p_i associated to each symbol s_i .

Information in the Statistical Theory

- ◆ The information conveyed by a symbol is defined as follows:

$$I(s_i) = \log_2 \frac{1}{p_i}$$

- ◆ The information in a message is, thus:

$$I(s_1 \dots s_n) = - \sum_{i=1}^n \log_2 p_i$$

Entropy

- ◆ We may, then, define the *average* information for a source with known characteristics.
- ◆ We call this quantity the *entropy* of the source and denote it by $H(S)$.

$$H(S) = \sum_{i=1}^n p_i I_i$$

A Simple Example

◆ Assume that,

$$\begin{aligned} - p(s_1) &= 0.500 & p(s_2) &= 0.250 \\ - p(s_3) &= 0.125 & p(s_4) &= 0.125 \end{aligned}$$

◆ Then,

$$- I(s_1) = 1; I(s_2) = 2; I(s_3) = I(s_4) = 3$$

◆ And,

$$\begin{aligned} H(S) &= .5 \times 1 + .25 \times 2 + .125 \times 3 \times 2 \\ &= .5 + .5 + .75 = 1.75 \end{aligned}$$

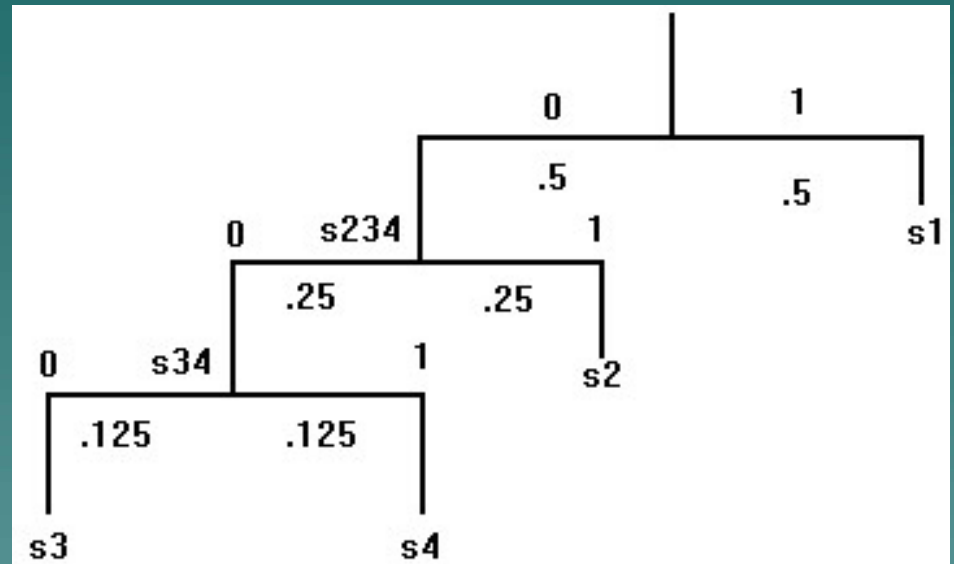
Variable Length Codes

- ◆ We may take advantage of the known probabilities of the symbols of the source to assign shorter codes to those symbols with higher probabilities; longer codes to those with higher probabilities.
- ◆ Codes which take optimum advantage of this fact are called Huffman codes

The Huffman Tree for Variable Length Codes

Huffman's idea was to construct a tree such as the one shown to assign the codes to the different symbols. The assignment of the "labels" of the branches is arbitrary.

The choice shown in the tree determines the code on the right.



```
S1 <- 1
S2 <- 01
S3 <- 000
S4 <- 001
```

One Simple Application

- ◆ One immediate conclusion derived from Shannon's theory is that no code may yield shorter average length (\bar{L}) than the entropy.
- ◆ In the code illustrated

$$\begin{aligned}\bar{L} &= 1 \times 0.5 + 2 \times 0.25 + (3 \times 0.125) \times 2 \\ &= 1.75\end{aligned}$$

Shortcomings

- ◆ Even though Shannon's theory provides us with a simple, yet powerful, tool to assess the information in a message it has two main disadvantages:
 - a) It assumes that we know the probability distribution of the symbols in the source
 - b) It assumes that the source is ergodic

Ergodicity

- ◆ When a source is stationary, and when every possible ensemble average (of letters, digrams, trigrams, etc.) is equal to the corresponding time average, the source is said to be ergodic.
- ◆ The practical limitation when considering a source to be ergodic is that ergodicity somewhat depends on how we define a symbol.

Shortcomings

- ◆ A more subtle disadvantage lies in the fact that we must agree on the meaning of the codes for the symbols a priori.
- ◆ For instance, in ASCII, the binary string "1000001" is ASSUMED to be the code for the letter "A".
- ◆ Once we have agreed on a certain convention, it is certainly possible to combine the symbols of our language to form complex messages. But the fact remains that the sender and the receiver MUST agree on such conventions.

An Example

- ◆ Assume that we work with only two symbols. Further, we shall agree on the fact that the first “symbol” is the text for the Bible and that the second symbol is the phrase “Let’s eat”.
- ◆ Of course we need only 1 bit to encode this two symbols. In which case there is exactly the same amount of information in the Bible as in the phrase “Let’s eat”.
- ◆ And that information is exactly 1 bit!

Algorithmic Information

- ◆ Given the mentioned limitations, Andrei Kolmogorov took a different path to define information.
- ◆ *The information in a message is the length (in bits) of the shortest program which is able to reproduce the message.*
- ◆ This is called the *algorithmic information*.

Algorithmic Information

- ◆ For example, the message comprised of 100,000,000 consecutive 1's may be easily reproduced by the following program:

```
for i=1 to 1000000000  
    print "1"  
endfor
```

Kolmogorov's Complexity

- ◆ If we use one ASCII character for each letter, space and control signs (such as CR) in our program we see that we need 37×7 or 259 bits.
- ◆ That, according to Kolmogorov is the algorithmic information ("AI"; also known as *Kolmogorov's complexity*) contained in a sequence of *one hundred million* consecutive 1's.

Kolmogorov's Complexity

- ◆ For this definition to make sense in general, we must specify the programming language.
- ◆ Kolmogorov took advantage of concepts of computer theory by establishing that the machine to be programmed would be the so-called "Turing Machine".
- ◆ But it may be proven that the algorithmic information is independent of the language except for an additive constant.

Algorithmic Information

- ◆ If we find the shortest program for M in language $C1$ and also in language $C2$, the information as per $C1$ will be $I + K1$ and as per $C2$ will be $I + K2$.
- ◆ Therefore, it is really not important which language we choose, as long as we stick to it during the course of all our calculations.

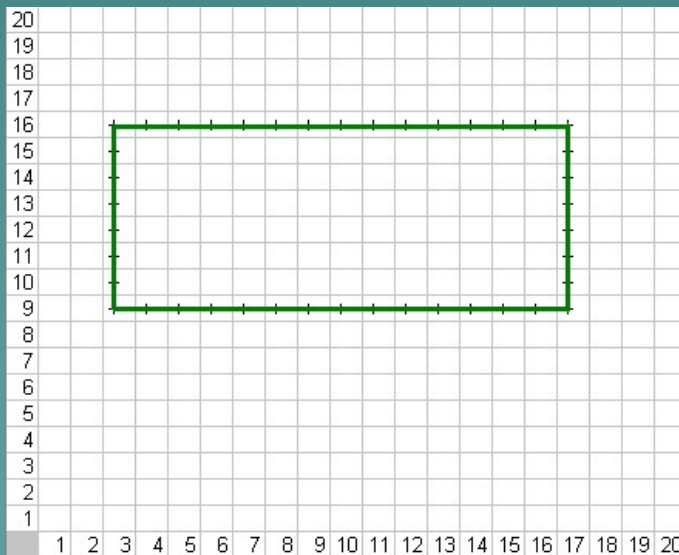
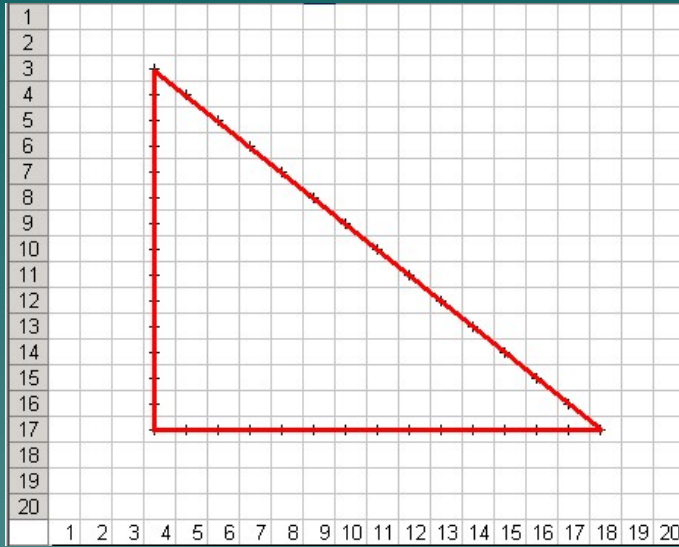
Computability of Algorithmic Information

- ◆ Unfortunately, the algorithmic theory of information yields a fundamental negative result:
 - *It is, in general, impossible to calculate the AI for an arbitrary message in a bounded amount of time.*

A Contrasting Case

- ◆ To illustrate the difference between both approaches we analyze a simple example.
- ◆ We shall extract the information in two figures.
- ◆ The first one corresponds to a triangle and the second to a square.

Triangle and Square



- ◆ Clearly, the figures seem to convey different information.
- ◆ We have enclosed both figures in a 20x20 grid.

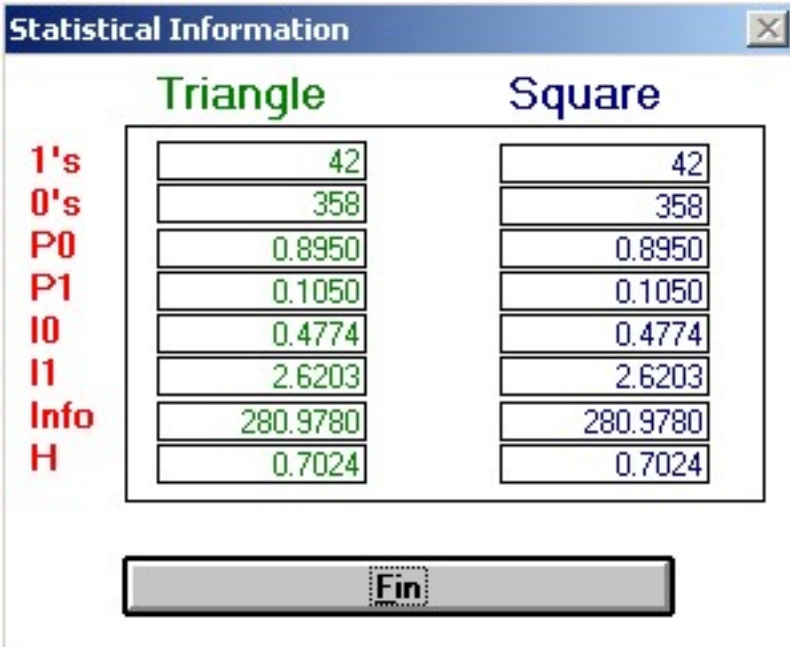
The Statistical Information

- ◆ We wrote a program which calculates the basic characteristic of both figures as per Shannon's theory.

Statistical Information				
			Triangle	Square
1's		42		42
0's		358		358
P0		0.8950		0.8950
P1		0.1050		0.1050
I0		0.4774		0.4774
I1		2.6203		2.6203
Info		280.9780		280.9780
H		0.7024		0.7024
			Fin	

The Statistical Information

- ◆ We obtain identical measures for both figures in spite of their differences because the number of non-zero elements is the same.



A screenshot of a software window titled "Statistical Information". It displays two columns of data, "Triangle" and "Square", with identical values for each row. The rows are labeled on the left: "1's", "0's", "P0", "P1", "I0", "I1", "Info", and "H". The values are displayed in green text within white boxes. At the bottom of the window is a button labeled "Fin".

	Triangle	Square
1's	42	42
0's	358	358
P0	0.8950	0.8950
P1	0.1050	0.1050
I0	0.4774	0.4774
I1	2.6203	2.6203
Info	280.9780	280.9780
H	0.7024	0.7024

The Algorithmic Information

- ◆ We express the elements of the figures in a table.
- ◆ This is a partial view of the table for the triangle.
- ◆ "x" and "y" denote the coordinates of the grid.
- ◆ "z" is "1" if there is a non-zero element.
- ◆ "z" is "0" otherwise

x	y	z
14	6	0
15	6	1
16	6	0
17	6	0
18	6	0
19	6	0
20	6	0
1	5	0
2	5	0
3	5	0
4	5	0
5	5	1
6	5	1
7	5	1

The Algorithmic Information

- ◆ We find the coefficients of the following polynomial:

$$z(x, y) = \sum_{i=0}^{g_1} \sum_{j=0}^{g_2} c_{ij} x^i y^j$$

where g_1, g_2 denote the highest degrees of the corresponding monomials

The Algorithmic Information

- ◆ We may say that the representative coefficients provide a measure of the information contained in both figures algorithmically.
- ◆ We (arbitrarily) select $g_1 = g_2 = 4$ to approximate the figures.
- ◆ The calculated coefficients are shown in the next slide.

Coefficients for the Triangle

C00	-1.2461	C20	-0.0239	C40	0.0000
C01	0.6215	C21	0.0099	C41	0.0000
C02	-0.0928	C22	-0.0018	C42	0.0000
C03	0.0063	C23	0.0001	C43	0.0000
C04	-0.0002	C24	0.0000	C44	0.0000
C10	0.3326	C30	0.0008		
C11	-0.1244	C31	-0.0003		
C12	0.0206	C32	0.0001		
C13	-0.0015	C33	0.0000		
C14	0.0000	C34	0.0000		

Notice that some values are equal to zero, yielding only 16 effective coefficients.

Algorithmic Information for the Triangle

- ◆ Assuming that every coefficient is stored in a 32 bit floating point variable, therefore, we need $16 \times 32 = 512$ bits to capture the essence of the figure.
- ◆ Therefore, the algorithmic information is, in this case 512 bits.

Algorithmic Information for the Square

- ◆ A similar procedure for the square yields 22 effective coefficients.
- ◆ In that case the algorithmic information is $22 \times 32 = 704$ bits
- ◆ The AI of a triangle is different from the one of a square.
- ◆ Furthermore, there is more information in a square than in a triangle.
- ◆ Therefore, the algorithmic approach seems to be intuitively better.

Approximately computing AI

- ◆ We note that all compression methods rely on the fact that we may find regularities in the data we wish to compress.
- ◆ If there is some regularity on the data Kolmogorov's complexity is smaller than the data itself.

Approximately computing AI

- ◆ An immediate consequence of the above is that random sequences are not compressible.
- ◆ Hence, any data set compressed to its utmost will look as a random sequence.
- ◆ To calculate the algorithmic information, therefore, we must seek for algorithms with the ability to detect arbitrary patterns.

Approximately computing AI

- ◆ Assume the following data (M1):

MSMQETQVQHFRKDRQARRAERAQAKDNRFQAHQQTI
QMRDMEKQVKKQYKDHHDQMKESETHVDNIKKDDKH
AIEEDQMDKDNRTSYKSHIIRHLQMHTQQHQQYMQHQ
DQAKKEKTHGAIKGALKA

Can you find regularities in it?

Metasymbols

- ◆ Shannon's theory assumes that symbols are defined a priori.
- ◆ We may define "meta-symbols" as arbitrary sets of symbols which exhibit regularities.

Metasymbols

We have rearranged M1 in a 16 x 8 matrix:

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
M	S	M	Q	E	T	Q	V	Q	H	F	R	K	D	R	Q	1
A	R	R	A	E	R	A	Q	A	K	D	N	R	F	Q	A	2
H	Q	Q	T	I	Q	M	R	D	M	E	K	Q	V	K	K	3
Q	Y	K	D	H	H	D	Q	M	K	E	S	E	T	H	V	4
D	N	I	K	K	D	D	K	H	A	I	E	E	D	Q	M	5
D	K	D	N	R	R	S	Y	K	S	H	I	I	R	H	L	6
Q	M	H	T	Q	Q	H	Q	Q	Y	M	Q	H	Q	D	Q	7
A	K	K	E	K	T	H	G	A	I	K	G	A	L	K	A	8

Metasymbol 1

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
M	S	M	Q	E	T	Q	V	Q	H	F	R	K	D	R	Q	1
A	R	R	A	E	R	A	Q	A	K	D	N	R	F	Q	A	2
H	Q	Q	T	I	Q	M	R	D	M	E	K	Q	V	K	K	3
Q	Y	K	D	H	H	D	Q	M	K	E	S	E	T	H	V	4
D	N	I	K	K	D	D	K	H	A	I	E	E	D	Q	M	5
D	K	D	N	R	R	S	Y	K	S	H	I	I	R	H	L	6
Q	M	H	T	Q	Q	H	Q	Q	Y	M	Q	H	Q	D	Q	7
A	K	K	E	K	T	H	G	A	I	K	G	A	L	K	A	8

All Instances of ms_1

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
M	S	M	Q	E	T	Q	V	Q	H	F	R	K	D	R	Q	1
A	R	R	A	E	R	A	Q	A	K	D	N	R	F	Q	A	2
H	Q	Q	T	I	Q	M	R	D	M	E	K	Q	V	K	K	3
Q	Y	K	D	H	H	D	Q	M	K	E	S	E	T	H	V	4
D	N	I	K	K	D	D	K	H	A	I	E	E	D	Q	M	5
D	K	D	N	R	R	S	Y	K	S	H	I	I	R	H	L	6
Q	M	H	T	Q	Q	H	Q	Q	Y	M	Q	H	Q	D	Q	7
A	K	K	E	K	T	H	G	A	I	K	G	A	L	K	A	8

Metasymbol 2

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	
M	S	M	Q	E	T	Q	V	Q	H	F	R	K	D	R	Q	1
A	R	R	A	E	R	A	Q	A	K	D	N	R	F	Q	A	2
H	Q	Q	T	I	Q	M	R	D	M	E	K	Q	V	K	K	3
Q	Y	K	D	H	H	D	Q	M	K	E	S	E	T	H	V	4
D	N	I	K	K	D	D	K	H	A	I	E	E	D	Q	M	5
D	K	D	N	R	R	S	Y	K	S	H	I	I	R	H	L	6
Q	M	H	T	Q	Q	H	Q	Q	Y	M	Q	H	Q	D	Q	7
A	K	K	E	K	T	H	G	A	I	K	G	A	L	K	A	8

Metasymbol 2 appears, for the second time, in position (15,4)

M1 as a Collection of Metasymbols

[illegible]

M1 Revisited

◆ Original:

MSMQETQVQHFRKDRQARRAERAQAKDNRFQAHQQTI
QMRDMEKQVKKQYKDHHQMKESETHVDNIKKDDKH
AIEEDQMDKDNRTSYKSHIIRHLQMHTQQHQQYMQHQ
DQAKKEKTHGAIKGALKA

◆ As Metasymbols:

αδδδγδγγγβαγαβγγγεδεδδ

Relative Performance

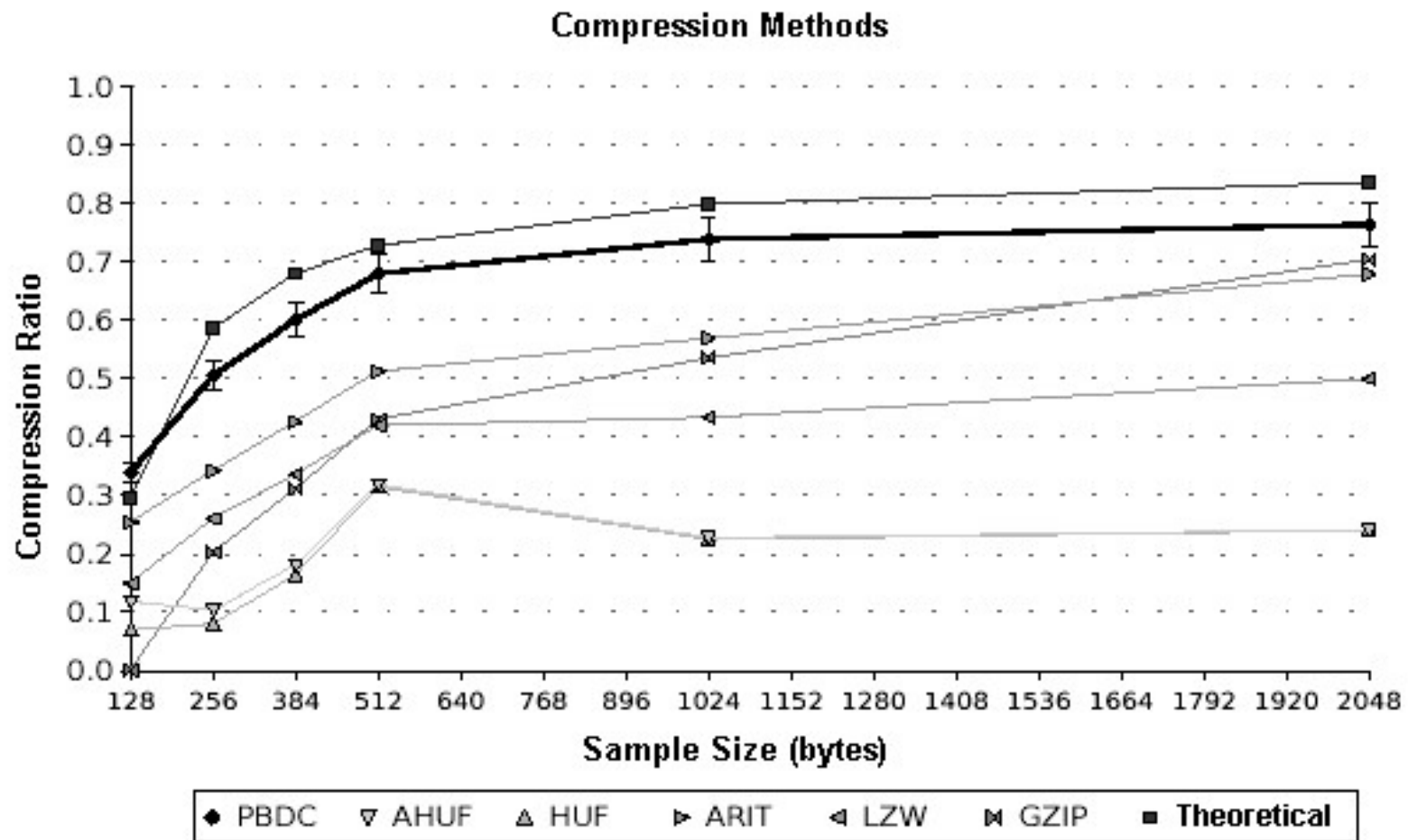
- ◆ **Compare traditional compression methods:**

- Huffman
- Adaptive Huffman
- Arithmetic
- Lempel-Ziv-Welch
- GZIP (LZ77)

and

- **Pattern (metasymbolic) Based Data Compression**

Relative Performance



Approximate Algorithmic Information

- ◆ The compressed “messages” with pattern based data compression are not amenable to be improved.
- ◆ For all practical purposes finding metasymbols is the utmost in compression
- ◆ Therefore, the packed messages are the closest we can hope to get to evaluating Kolmogorov’s complexity

Lossless and Lossy Compression

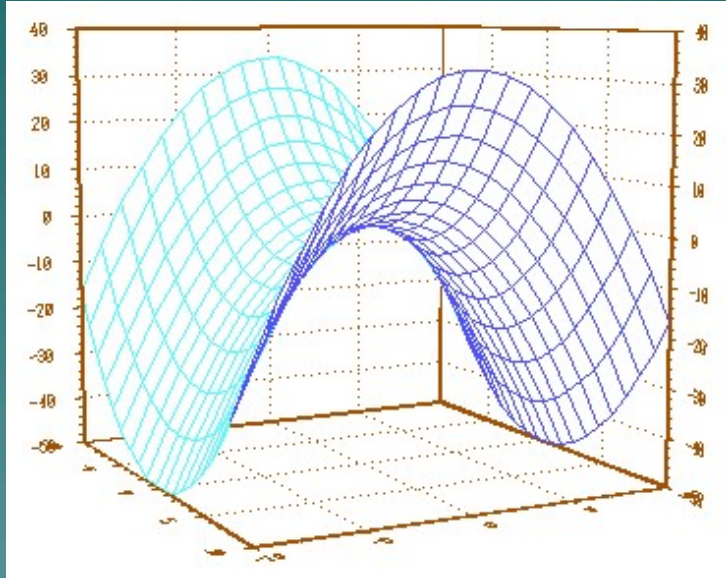
- ◆ The examples of metasymbolic application are all *lossless*, i.e. the original message may be fully reproduced
- ◆ In *lossy* compression schemes, however, we agree to certain losses in retrieving the original message
- ◆ When approximating the data we may decide not to fully cover all symbols

Lossy Compression

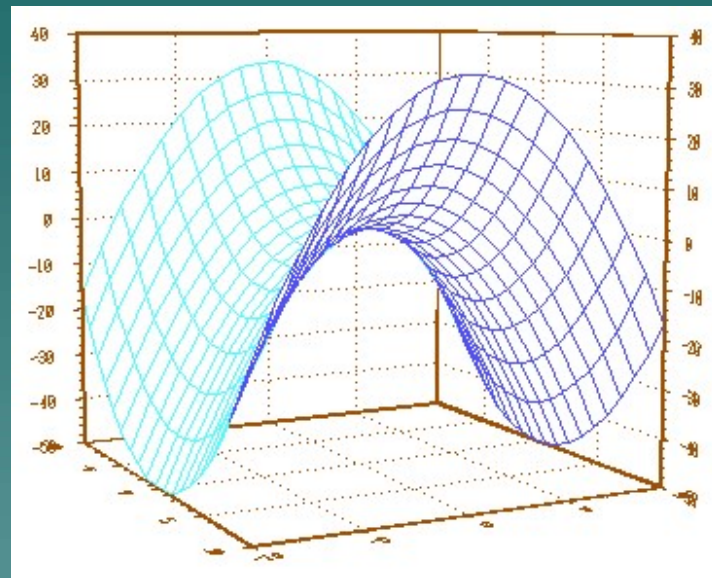
- ◆ The original data was expressed with 14,080 bits
- ◆ Partial coverage with metasymbols was calculated
- ◆ The metasymbolic representation required of only 288 bits
- ◆ Compression ratio (CR):

$$CR = 14,080/288 = 48.8889$$

A Simple Example of Lossy Compression

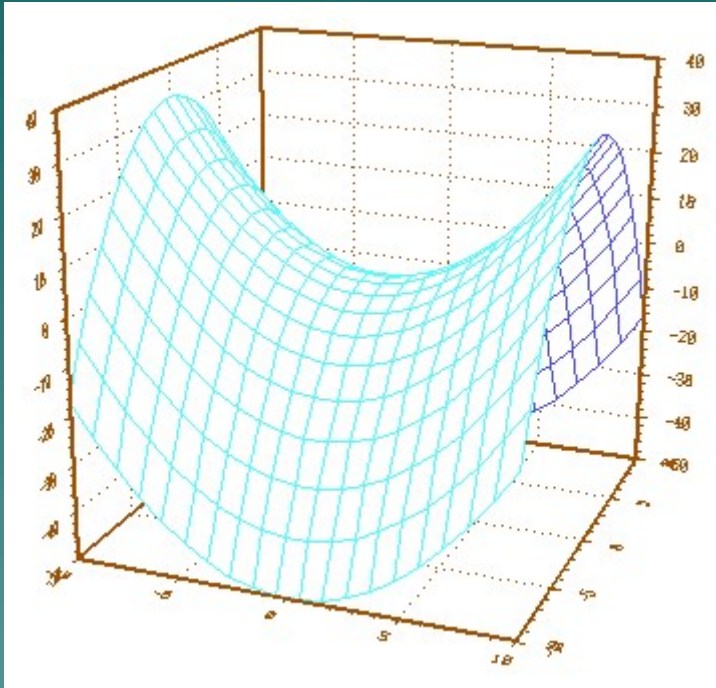


Original

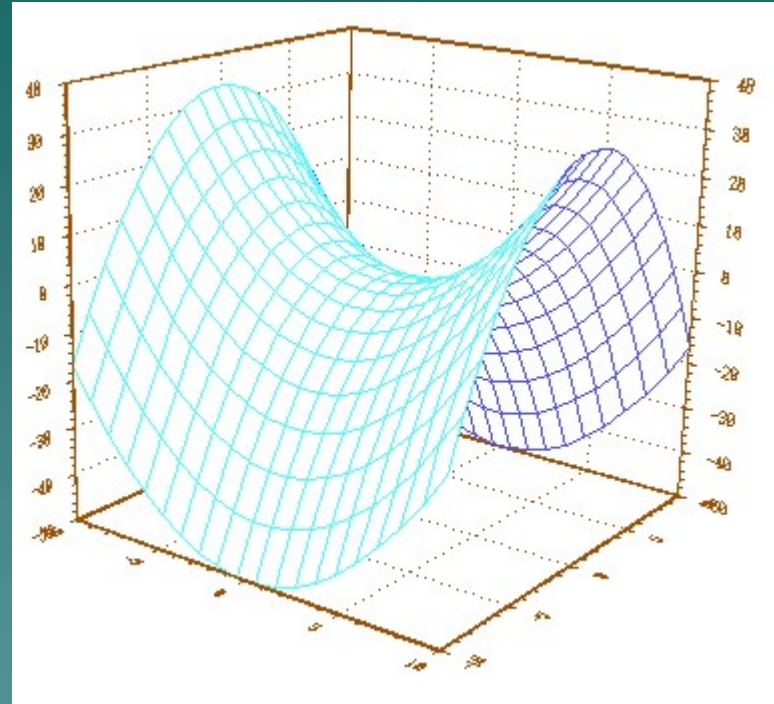


Lossy

A Simple Example of Lossy Compression

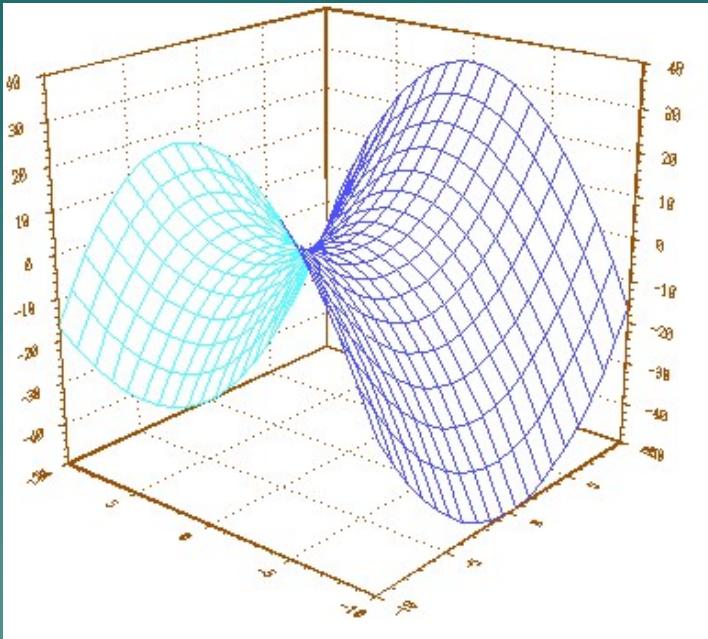


Original

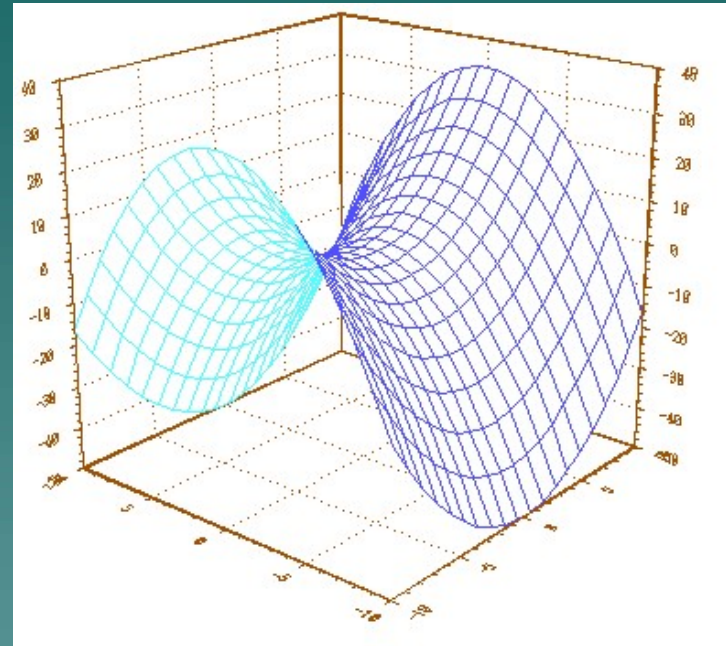


Lossy

A Simple Example of Lossy Compression



Original



Lossy

Conclusions

- ◆ Metasymbols allow us to approximately calculate the most compact way of representing a “message”
- ◆ This representation is, very nearly, Kolmogorov’s complexity: the algorithmic information of the message

Conclusions

- ◆ When absolutely covering the message, the compression competes with older (lossless) schemes
- ◆ Lossless data compression is achieved

Conclusions

- ◆ When partially covering the message, the metasymbolic representation allows for lossy compression
- ◆ The level of quality competes with other (lossy) compression schemes
- ◆ Lossless compression is achieved

Applying the Method to Bioinformatics

- ◆ The metasymbolic approach has been used to represent large sets of proteins
- ◆ At present, we are looking into sets of proteins with similar biological characteristics to determine whether they share the same (or similar) metasymbols