Kolmogorov complexity

Last modified: 2026-10-30 13:54:17 (PDT)

Shannon entropy measures the information in a random source: how many binary digits, on average, its outcomes need. Kolmogorov complexity measures the information in one string: the length of the shortest program that prints it. This page defines Kolmogorov complexity, shows that most strings cannot be compressed and that no program computes it, and shows that its expected value per symbol approaches the entropy. It builds on the entropy and source coding pages in *Information Theory for Data Science*.

Shortest descriptions

Definition 0.1 (Kolmogorov complexity)**.** Fix a universal computer U: a Turing machine, or an interpreter for a programming language, that reads a program p, a finite string of binary digits, and either runs forever or halts and outputs a binary string $U(p)$. Assume no program on which U halts is a proper prefix of another, as happens when U reads its program one digit at a time and must itself decide where the program ends.
The **Kolmogorov complexity** of a binary string x is the length of a shortest program that outputs x:

$$K(x) \stackrel{\text{def}}{=} \min\{|p| : U(p) = x\},$$

where $|p|$ is the number of digits in p. This is the definition of Cover and Thomas (2006, sec. 14.2).

Definition 0.2 (Conditional Kolmogorov complexity)**.** Let the universal computer U of Definition 0.1 also receive an integer n as an input, so that it outputs $U(p, n)$ if it halts on program p. Assume that, for each fixed n, no program on which U halts is a proper prefix of another. The **conditional Kolmogorov complexity** of a binary string x given n is

$$K(x \mid n) \stackrel{\text{def}}{=} \min\{|p| : U(p, n) = x\}.$$

Cover and Thomas (2006, sec. 14.2) define it the same way.

Theorem 0.1 (The choice of computer changes K by at most a constant)**.** *Let K_U and K_A be the Kolmogorov complexities and conditional Kolmogorov complexities defined with universal computers U and A. There is a constant c_A, which depends on U and A but not on x or n, such that*

$$K_U(x) \leq K_A(x) + c_A \quad \text{and} \quad K_U(x \mid n) \leq K_A(x \mid n) + c_A$$

for every binary string x and every integer n.

Proof

Proof. Because U is universal, some program s_A makes U simulate A: given s_A followed by a program p for A, U outputs whatever A outputs on p. So if p is a shortest program for x on A, then s_A followed by p is a program for x on U, and $K_U(x) \leq |s_A|+|p| = K_A(x)+|s_A|$. The same program s_A also passes n on to A, which gives the conditional bound in the same way. Take $c_A = |s_A|$. Cover and Thomas (2006, sec. 14.2) give this argument in full. □

Remark 0.1 (Why the choice of computer hardly matters). Applying Theorem 0.1 with the roles of U and A swapped gives a constant c'_A with $K_A(x) \leq K_U(x) + c'_A$. Together, $|K_U(x) - K_A(x)| \leq \max(c_A, c'_A)$ for every x, and in the same way $|K_U(x \mid n) - K_A(x \mid n)| \leq \max(c_A, c'_A)$ for every x and n. The constant can be large, so the complexity of one short string means little, but statements about how $K(x)$ grows with the length of x do not depend on the computer. From here on, U is fixed and dropped from the notation.

Theorem 0.2 (A string is never much more complex than its length)**.** *There is a constant c such that every binary string x of length n has*

$$K(x \mid n) \leq n + c$$

and, when $n \geq 1$,

$$K(x) \leq n + 2\log_2 n + c.$$

Proof

Proof. For the first bound, the program "print the next n digits" followed by the n digits of x outputs x. Because U is given n, the program needs no marker to show where it ends, and its fixed part has some length c_1. So $K(x \mid n) \leq n + c_1$.
For the second bound, the program must also describe n in a way that shows where the description ends. Write n in binary, which takes $\lfloor \log_2 n \rfloor + 1$ digits, repeat each digit twice, and end with the pair 01. This takes $2\lfloor \log_2 n \rfloor + 4 \leq 2 \log_2 n + 4$ digits. A program that reads this description of n and then prints the next n digits has length at most $n + 2 \log_2 n + 4 + c_2$ for a constant c_2. Take $c = \max(c_1, c_2 + 4)$. Cover and Thomas (2006, sec. 14.2) prove both bounds. □

Example 0.1 (Simple strings)**.**

- For $n \geq 1$, the string x of n zeros has $K(x \mid n) \leq c$ for a constant c, since the program "print n zeros" does not grow with n. Without n given, the program must also describe n, and the proof of Theorem 0.2 gives $K(x) \leq 2 \log_2 n + c'$.
- The string $x = 0101 \ldots 01$ of length n, for even $n \geq 2$, also has $K(x \mid n) \leq c$, since the program "print 01 until n digits are printed" does not grow with n. Half its digits are ones, as expected of n fair coin flips.
- The string x of the first n binary digits of π also has $K(x \mid n) \leq c$: a fixed program computes as many digits of π as asked. The digits show no obvious pattern, yet they are as compressible as a string of zeros.

Incompressible strings

Theorem 0.3 (Most strings are incompressible)**.** *Let $n \geq 1$ and $0 \leq k \leq n$ be integers. Fewer than 2^{n-k} of the 2^n binary strings of length n satisfy $K(x \mid n) < n-k$. In particular, at least one string of length n has $K(x \mid n) \geq n$.*

Proof

Proof. There are 2^j binary strings of length j, so the number of programs of length less than $n - k$ is

$$\sum_{j=0}^{n-k-1} 2^j = 2^{n-k} - 1,$$

where the sum is empty, and equal to 0, when $k = n$.
Given n, each program outputs at most one string, so at most $2^{n-k} - 1$ strings x have a program shorter than $n - k$. With $k = 0$, at most $2^n - 1$ of the 2^n strings of length n have $K(x \mid n) < n$, which leaves at least one with $K(x \mid n) \geq n$. □

Remark 0.2 (Incompressible strings look random). By Theorem 0.3, if x is drawn uniformly from the strings of length n, for example by n fair coin flips, then $\Pr(K(x \mid n) < n - k) < 2^{n-k}/2^n = 2^{-k}$. So a string of n fair coin flips can be compressed by more than k digits with probability less than 2^{-k}. A string with $K(x \mid n) \geq n$ has no description shorter than itself. Li and Vitányi (2019, chap. 2) call a string random when it is incompressible in this sense, for a variant of K whose programs need not be prefix-free. A proof can pick an incompressible string and show that some property must hold for it, or else the string would have a short description. Li and Vitányi (2019, chap. 6) call this the *incompressibility method*.

Kolmogorov complexity is uncomputable

Theorem 0.4 (Kolmogorov complexity cannot be computed)**.** *No program takes a binary string x as input and outputs $K(x)$.*

Proof

Proof. Suppose a program Q halts on every binary string x and outputs $K(x)$. For each integer $m \geq 1$, consider the program P_m that does the following: list binary strings in order of length, and in lexicographic order within each length, use Q to compute $K(x)$ for each one, and output the first x with $K(x) \geq m$. Such an x exists, because fewer than 2^m programs are shorter than m, so P_m halts, and its output x has $K(x) \geq m$.
The program P_m consists of a fixed part, of some length c, and a description of m that shows where it ends, which the proof of Theorem 0.2 writes in at most $2 \log_2 m + 4$ digits. So the output x of P_m also has $K(x) \leq 2 \log_2 m + 4 + c$, and together

$$m \leq K(x) \leq 2 \log_2 m + 4 + c.$$

The left side grows faster than the right, so this fails once m is large enough, and no such Q exists. The argument formalizes the Berry paradox, "the smallest positive integer not definable in under sixty letters", and Cover and Thomas (2006, chap. 14) discuss why K cannot be computed. □

Kolmogorov complexity and entropy

Theorem 0.5 (Expected Kolmogorov complexity per symbol approaches the entropy)**.** *Let $X_1, X_2, \ldots$ be independent copies of a random variable X that takes values in a finite set $\mathcal{X}$, each value written as a different binary string of one fixed length, and let $H(X)$ be the entropy of X with base-2 logarithms. Write $X^n = (X_1, \ldots, X_n)$. There is a constant c, which depends on $\mathcal{X}$ and U but not on n, such that, for every $n \geq 1$,*

$$H(X) \leq \frac{1}{n}\mathbb{E}[K(X^n \mid n)] \leq H(X) + \frac{(|\mathcal{X}| - 1)\log_2 n}{n} + \frac{c}{n}.$$

So $\frac{1}{n}\mathbb{E}[K(X^n \mid n)] \to H(X)$ as $n \to \infty$. This is Theorem 14.3.1 of Cover and Thomas (2006).

Proof

Proof. For the lower bound, send each outcome x^n of X^n to a shortest program that outputs it given n. Different outcomes get different programs, because a program determines its output, and no halting program is a prefix of another, so this is a prefix code for X^n, which makes it uniquely decodable. Its expected length is $\mathbb{E}[K(X^n \mid n)]$. If X takes only one value with positive probability, then $H(X) = 0$ and the lower bound holds trivially. Otherwise, X^n also takes at least two values with positive probability, and part 1 of the source coding theorem gives $\mathbb{E}[K(X^n \mid n)] \geq H(X^n)$. Applying the chain rule for entropy $n - 1$ times, and using that conditioning on independent variables does not change entropy, gives $H(X^n) = nH(X)$. Dividing by n gives the lower bound.
For the upper bound, Cover and Thomas (2006, sec. 14.3) describe x^n in two parts: first how many times each value in $\mathcal{X}$ occurs, which takes about $(|\mathcal{X}| - 1)\log_2 n$ digits, then which of the strings with those counts x^n is, which takes about $nH(X)$ digits on average. □

Remark 0.3 (Entropy describes a distribution; Kolmogorov complexity describes a string). Entropy is the expected surprisal of a random variable, so it is a property of a distribution and says nothing about any one outcome. Kolmogorov complexity is a property of one string and needs no distribution. Theorem 0.5 says that the two agree on average, per symbol, for long independent sequences. For a single string they can disagree: the string 0101 ... 01 in Example 0.1 has as many ones as zeros, like a typical outcome of fair coin flips, whose entropy is 1 bit per flip, but its Kolmogorov complexity per digit, $K(x \mid n)/n \leq c/n$, tends to 0.

References

Cover, Thomas M., and Joy A. Thomas. 2006. *Elements of Information Theory*. 2nd ed. Wiley-Interscience. https://doi.org/10.1002/047174882X.

Li, Ming, and Paul Vitányi. 2019. *An Introduction to Kolmogorov Complexity and Its Applications*. 4th ed. Springer. https://doi.org/10.1007/978-3-030-11298-1.