A Safe Approximation for Kolmogorov Complexity

Peter Bloem[1], Francisco Mota[2], Steven de Rooij[1], Luís Antunes[2], and Pieter Adriaans[1]

[1] System and Network Engineering Group
University of Amsterdam
uva@peterbloem.nl, steven.de.rooij@gmail.com, p.w.adriaans@uva.nl
[2] CRACS & INESC-Porto LA and Institute for Telecommunications
University of Porto
fmota@fmota.eu, lfa@dcc.fc.up.pt

Abstract. Kolmogorov complexity (K) is an incomputable function. It can be approximated from above but not to arbitrary given precision and it cannot be approximated from below. By restricting the source of the data to a specific model class, we can construct a computable function $\overline{\kappa}$ to approximate K in a probabilistic sense: the probability that the error is greater than k decays exponentially with k. We apply the same method to the normalized information distance (NID) and discuss conditions that affect the safety of the approximation.

The Kolmogorov complexity of an object is its shortest description, considering all computable descriptions. It has been described as "the accepted absolute measure of information content of an individual object" [1], and its investigation has spawned a slew of derived functions and analytical tools. Most of these tend to separate neatly into one of two categories: the platonic and the practical.

On the platonic side, we find such tools as the normalized information distance [2], algorithmic statistics [1] and sophistication [3,4]. These subjects all deal with incomputable "ideal" functions: they optimize over all computable functions, but they cannot be computed themselves.

To construct practical applications (ie. runnable computer programs), the most common approach is to take one of these platonic, incomputable functions, derived from Kolmogorov complexity (K), and to approximate it by swapping K out for a computable compressor like GZIP [5]. This approach has proved effective in the case of normalized information distance (NID) [2] and its approximation, the normalized compression distance (NCD) [6]. Unfortunately, the switch to a general-purpose compressor leaves an analytical gap. We know that the compressor serves as an upper bound to K—up to a constant—but we do not know the difference between the two, and how this error affects the error of derived functions like the NCD. This can cause serious contradictions. For instance, the normalized information distance has been shown to be non-approximable [7], yet the NCD has proved its merit empirically [6]. Why this should be the case, and when this approach may fail has, to our knowledge, not yet been investigated.

We aim to provide the first tools to bridge this gap. We will define a computable function which can be said to approximate Kolmogorov complexity, with some practical limit to the error. To this end, we introduce two concepts:

- We generalize resource-bounded Kolmogorov complexity (K^t) to *model-bounded Kolmogorov complexity*, which minimizes an object's description length over any given enumerable subset of Turing machines (a *model class*). We explicitly assume that the source of the data is contained in the model class.
- We introduce a probabilistic notion of approximation. A function approximates another *safely*, under a given distribution, if the probability of them differing by more than k bits, decays at least exponentially in k. [3]

While the resource-bounded Kolmogorov complexity is computable in a technical sense, it is never computed practically. The generalization to model bounded Kolmogorov complexity creates a connection to *minimum description length* (MDL) [8,9,10], which does produce algorithms and methods that are used in a practical manner. Kolmogorov complexity has long been seen as a kind of platonic ideal which MDL approximates. Our results show that MDL is not just an upper bound to K, it also approximates it in a probabilistic sense.

Interestingly, the model-bounded Kolmogorov complexity itself—the smallest description using a single element from the model class—is not a safe approximation. We can, however, construct a computable, safe approximation by taking into account all descriptions the model class provides for the data.

The main result of this paper is a computable function $\overline{\kappa}$ which, under a model assumption, safely approximates K (Th. 3). We also investigate whether an $\overline{\kappa}$-based approximation of NID is safe, in different contexts (Th. 5, Th. 6 and 7).

1 Turing Machines and Probability

Turing Machines Let $\mathbb{B} = \{0,1\}^*$. We assume that our data is encoded as a finite binary string. Specifically, the natural numbers can be associated to binary strings, for instance by the bijection: $(0,\epsilon)$, $(1,0)$, $(2,1)$, $(3,00)$, $(4,01)$, etc, where ϵ is the empty string. To simplify notation, we will sometimes conflate natural numbers and binary strings, implicitly using this ordering.

[3] This consideration is subject to all the normal drawbacks of asymptotic approaches. For this reason, we have foregone the use of big-O notation as much as possible, in order to make the constants and their meaning explicit.

We fix a canonical prefix-free coding, denoted by $\overline{x}$, such that $|\overline{x}| \leq |x|+2\log|x|$. See [11, Example 1.11.13] for an example . Among other things, this gives us a canonical pairing function to encode two strings x and y into one: $\overline{x}y$.

We use the Turing machine model from [11, Example 3.1.1]. The following properties are important: the machine has a read-only, right-moving input tape, an auxiliary tape which is read-only and two-way, two read-write two-way worktapes and a read-write two-way output tape.[4]All tapes are one-way infinite. If a tape head moves off the tape or the machine reads beyond the length of the input, it enters an infinite loop. For the function computed by TM i on input p with auxiliary input y, we write $T_i(p \mid y)$ and $T_i(p) = T_i(p \mid \epsilon)$. The most important consequence of this construction is that the programs for which a machine with a given auxiliary input y halts, form a prefix-free set [11, Example 3.1.1]. This allows us to interpret the machine as a probability distribution (as described in the next subsection).

We fix an effective ordering $\{T_i\}$. We call the set of all Turing machines $\mathscr{C}$. There exists a universal Turing machine, which we will call U, that has the property that $U(\bar{\imath}p \mid y) = T_i(p \mid y)$ [11, Theorem 3.1.1].

Probability We want to formalize the idea of a probability distribution that is *computable*: it can be simulated or computed by a computational process. For this purpose, we will interpret a given Turing machine T_q as a probability distribution p_q: each time the machine reads from the input tape, we provide it with a random bit. The Turing machine will either halt, read a finite number of bits without halting, or read an unbounded number of bits. $p_q(x)$ is the probability that this process halts and produces x: $p_q(x) = \sum_{p:T_q(p)=x} 2^{-|p|}$. We say that T_q *samples* p_q. Note that if p is a semimeasure, $1 - \sum_x p(x)$ corresponds to the probability that this sampling process will not halt.

We model the probability of x conditional on y by a Turing machine with y on its auxiliary tape: $p_q(x \mid y) = \sum_{p:T_q(p|y)=x} 2^{-|p|}$.

The *lower semicomputable semimeasures* [11, Chapter 4] are an alternative formalization. We show that it is equivalent to ours:

Lemma 1 [†] *The set of probability distributions sampled by Turing machines in $\mathscr{C}$ is equivalent to the set of lower semicomputable semimeasures.*

The distribution corresponding to the universal Turing machine U is called m: $m(x) = \sum_{p:U(p)=x} 2^{-|p|}$. This is known as a universal distribution. K and m dominate each other, ie. $\exists c \forall x : |K(x) - \log m(x)| < c$ [11, Theorem 4.3.3].

[4] Multiple worktapes are only required for proofs involving resource bounds.

[†] Proof in the appendix.

2 Model-Bounded Kolmogorov Complexity

In this section we present a generalization of the notion of resource-bounded Kolmogorov complexity. We first review the unbounded version:

Definition 1 *Let* $k(x \mid y) = \arg\min_{p:U(p|y)=x} |p|$*. The prefix-free, conditional* Kolmogorov complexity *is* $K(x \mid y) = |k(x \mid y)|$ *with* $K(x) = K(x \mid \epsilon)$*.*

To find a computable approximation to K, we limit the TMs considered:

Definition 2 *A* model class $C \subseteq \mathscr{C}$ *is a computably enumerable set of Turing machines. Its members are called* models*. A* universal model *for* C *is a Turing machine* U^C *such that* $U^C(\bar{\imath}p \mid y) = T_i(p \mid y)$ *where* i *is an index over the elements of* C*.*

Definition 3 *For a given* C *and* U^C *we have* $K^C(x) = \min\left\{|p| \,:\, U^C(p) = x\right\}$*, called the* model-bounded Kolmogorov complexity*.*

K^C, unlike K, depends heavily on the choice of enumeration of C. A notation like K_{U^C} or $K^{i,C}$ would express this dependence better, but for the sake of clarity we will use K^C.

We can also construct a model-bounded variant of m, $m^C(x) = \sum_{p:U^C(p)=x} 2^{-|p|}$, which dominates all distributions in C:

Lemma 2 *For any* $T_q \in C$*,* $m^C(x) \geq c_q p_q(x)$ *for some* c_q *independent of* x*.*

Proof. $m^C(x) = \sum_{i,p:U^C(\bar{\imath}p)=x} 2^{-|\bar{\imath}p|} \geq \sum_{p:U^C(\bar{q}p)=x} 2^{-|\bar{q}|}2^{-|p|} = 2^{-|\bar{q}|}p_q(x)\,.$ □

Unlike K and $-\log m$, K^C and $-\log m^C$ do not dominate one another. We can only show that $-\log m^C$ bounds K from below ($\sum_{U^C(p)=x} 2^{-|p|} > 2^{-K^C(x)}$). In fact, as shown in Theorem 1, $-\log m^C$ and K can differ by arbitrary amounts.

Example 1 (resource-bounded Kolmogorov complexity [11, Chapter 7]) *Let* $t(n)$ *be some time-constructible function*[5]*. Let* T_i^t *be the modification of* $T_i \in \mathscr{C}$ *such that at any point in the computation, it halts immediately if more than* k *cells have been written to on the output tape and the number of steps that have passed is less than* $t(k)$*. In this case whatever is on the output tape is taken as the output of the computation. If this situation does not occur,* T_i *runs*

[5] Ie. $t : \mathbb{N} \to \mathbb{N}$ and t can be computed in $O(t(n))$ [12].

as normal. Let $U^t(\bar{\imath}p) = T_i^t(p)$. We call this model class C^t. We abbreviate K^{C^t} as K^t.

Since there is no known means of simulating U^t within $t(n)$, we do not know whether $U^t \in C^t$. It can be run in $ct(n) \log t(n)$ [11,13], so we do know that $U^t \in C^{ct \log t}$.

Other model classes include Deterministic Finite Automata, Markov Chains, or the exponential family (suitably discretized). These have all been thoroughly investigated in coding contexts in the field of Minimum Description Length [10].

3 Safe Approximation

When a code-length function like K turns out to be incomputable, we may try to find a lower and upper bound, or to find a function which dominates it. Unfortunately, neither of these will help us. Such functions invariable turn out to be incomputable themselves [11, Section 2.3].

To bridge the gap between incomputable and computable functions, we require a softer notion of approximation; one which states that errors of any size may occur, but that the larger errors are so unlikely, that they can be safely ignored:

Definition 4 *Let f and f_a be two functions. We take f_a to be an approximation of f. We call the approximation b-*safe (from above) *for a distribution (or* adversary*) p if for all k and some $c > 0$:*

$$p(f_a(x) - f(x) \geq k) \leq cb^{-k} \,.$$

Since we focus on code-length functions, usually omit "from above". A safe *function is b-safe for some $b > 1$. An approximation is safe for a model class C if it is safe for all p_q with $T_q \in C$.*

While the definition requires the property to hold for all k, it actually suffices to show that it holds for k above a constant, as we can freely scale c:

Lemma 3 *If $\exists_c \forall_{k:k>k_0} : p(f_a(x) - f(x) \geq k) \leq cb^{-k}$, then f_a is b-safe for f against p.*

Proof. First, we name the k below k_0 for which the ratio between the bound and the probability is the greatest: $k_m = \arg\max_{k \in [0,k_0]} \left[p(f_a(x) - f(x) \geq k)/cb^{-k}\right]$. We also define $b_m = cb^{-k_m}$ and $p_m = p(f_a(x) - f(x) \geq k_m)$. At k_m, we have $p(f_a(x) - f(x) \geq k_m) = p_m = \frac{p_m}{b_m} cb^{-k_m}$. In other words, the bound $c'b^{-k}$ with $c' = \frac{p_m}{b_m} c$ bounds p at k_m, the point where it diverges the most from the old bound. Therefore, it must bound it at all other $k > 0$ as well. □

Safe approximation, domination and lowerbounding form a hierarchy:

Lemma 4 *Let f_a and f be code-length functions. If f_a is a lower bound on f, it also dominates f. If f_a dominates f, it is also a safe approximation.*

Proof. Domination means that for all x: $f_a(x) - f(x) < c$, if f_a is a lower bound, $c = 0$. If f_a dominates f we have $\forall p, k > c : p(f_a(x) - f(x) \geq k) = 0$. □

Finally, we show that safe approximation is transitive, so we can chain together proofs of safe approximation.

Lemma 5 *The property of safety is transitive over the space of functions from $\mathbb{B}$ to $\mathbb{B}$ for a fixed adversary.*

Proof. Let $p(f(x) - g(x) \geq k) \leq c_1 {b_1}^{-k}$ and $p(g(x) - h(x) \geq k) \leq c_2 {b_2}^{-k}$. We need to show that $p(f(x) - h(x) \geq k)$ decays exponentially with k. We start with

$$p\left(f(x) - g(x) \geq k \vee g(x) - h(x) \geq k\right) \;\leq\; c_1 {b_1}^{-k} + c_2 {b_2}^{-k} \,. \tag{1}$$

Since $\{x : f(x) - h(x) \geq 2k\} \subseteq \{x : f(x) - g(x) \geq k \vee g(x) - h(x) \geq k\}$, the probability of the first set is less than that of the second: $p\left(f(x) - h(x) \geq 2k\right) \leq c_1 {b_1}^{-k} + c_2 {b_2}^{-k}$. Which gives us

$$\begin{aligned} &p\left(f(x) - h(x) \geq 2k\right) \leq c b^{-k} && \text{with } b = \min(b_1, b_2) \text{ and } c = \max(c_1, c_2)\,, \\ &p\left(f(x) - h(x) \geq k'\right) \leq c {b'}^{-k'} && \text{with } b' = \sqrt{b}\,. \end{aligned}$$

□

4 A Safe, Computable Approximation of K

Assuming that our data is produced from a model in C, can we construct a computable function which is safe for K? An obvious first choice is K^C. For it to be computable, we would normally ensure that all programs for all models in C halt. Since the halting programs form a prefix-free set, this is impossible. There is however a property for prefix functions that is analogous. We call this *sufficiency*:

Definition 5 *A sufficient model T is a model for which every infinite binary string contains a halting program as a prefix. A* sufficient model class *contains only sufficient models.*

We can therefore enumerate all inputs for U^C from short to long in series to find $k^C(x)$, so long as C is sufficient. For each input U^C either halts or attempts to

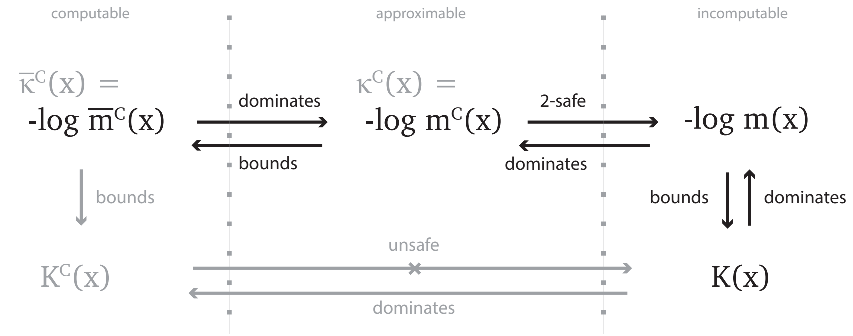

Fig. 1: An overview of how various code-length functions relate to each other in terms of approximation safety. These relations hold under the assumption that the data is generated by a distribution in C and that C is sufficient and complete.

read beyond the length of the input. In certain cases, we also require that C can represent all $x \in \mathbb{B}$ (ie. $m^C(x)$ is never 0). We call this property *completeness.*

We can now say, for instance, that K^C is computable for sufficient C. Unfortunately, K^C turns out to be unsafe:

Theorem 1 *There exist model classes C so that $K^C(x)$ is an unsafe approximation for $K(x)$ against some p_q with $T_q \in C$.*

Proof. We first show that K^C is unsafe for $-\log m^C$. Let C contain a single Turing machine T_q which outputs x for any input of the form $\overline{x}p$ with $|p| = x$ and computes indefinitely for all other inputs. T_q samples from $p_q(x) = 2^{-|\overline{x}|}$, but it distributes each x's probability mass uniformly over many programs much longer than $\overline{x}$.

This gives us $K^C(x) = |\overline{x}| + |p| = |\overline{x}| + x$ and $-\log m^C(x) = |\overline{x}|$, so that $K^C(x) + \log m^C(x) = x$. We get $m^C(K^C(x) + \log m^C(x) \geq k) = m^C(x \geq k) = \sum_{x:x\geq k} 2^{-|\overline{x}|} \geq \sum_{x:x\geq k} 2^{-2\log x} \geq k^{-2}$, so that K^C is unsafe for $-\log m^C$.

It remains to show that this implies that K^C is unsafe for K. In Theorem 2, we prove that $-\log m^C$ is safe for K. Assuming that K^C is safe for K (which dominates $-\log m^C$) implies K^C is safe for $-\log m^C$, which gives us a contradiction. □

Note that the use of a model class with a single model is for convenience only. The main requirement for K^C to be unsafe is that the prefix tree of U^C's programs distributes the probability mass for x over many programs of similar length. The greater the distance between K^C and $-\log m^C$, the greater the likelihood that K^C is unsafe.

Our next candidate for a safe approximation of K is $-\log m^C$. This time, we fare better. We first require the following lemma, called the *no-hypercompression theorem* in [10, p103]:

Lemma 6 *Let p_q be a probability distribution. The corresponding code-length function, $-\log p_q$, is a 2-safe approximation for any other code-length function against p_q. For any p_r and $k > 0$: $p_q(-\log p_q(x) + \log p_r(x) \geq k) \leq 2^{-k}$.*

Theorem 2 *The function $-\log m^C(x)$ is a 2-safe approximation of $K(x)$ against adversaries from C.*

Proof. Let p_q be some adversary in C. We have $p_q(-\log m^C(x) - K(x) \geq k) \leq cm^C(-\log m^C(x) - K(x) \geq k) \leq c2^{-k}$, where the inequalities follow from Lemmas 2 and 6, respectively. □

While we have shown m^C to be safe for K, it may not be computable, even if C is sufficient (since it is an infinite sum). We can, however, define an approximation, which, if C is sufficient and complete[6], is computable and dominates m^C.

Definition 6 *Let $\overline{m}^C_c(x)$ be the function computed by the following algorithm: Dovetail the computation of all programs on $U^C(x)$ in cycles, so that in cycle n, the first n programs are simulated for one further step. After each such step we consider the probability mass s of all programs that have stopped (where each program p contributes $2^{-|p|}$), and the probability mass s_x of all programs that have stopped and produced x. We halt the dovetailing and output s_x if the following stop condition is met:*

$$\frac{1-s}{s_x} \leq 2^c - 1\,.$$

Note that if C is sufficient, s goes to 1 and s_x never decreases. Since all programs halt, the stop condition must be reached.

Lemma 7 *If C is sufficient and complete, $\overline{m}^C_c(x)$ dominates m^C with a constant multiplicative factor 2^{-c} (ie. their code-lengths differ by at most c bits).*

Proof. Note that when the computation of $\overline{m}^C_c$ halts, we have $\overline{m}^C_c(x) = s_x$ and $m^C(x) \leq s_x + (1-s)$. This gives us:

$$\frac{m^C(x)}{\overline{m}^C_c(x)} \leq 1 + \frac{1-s}{s_x} \leq 2^c\,.$$ □

[6] If C is not complete we can prove safety, since x has probability zero, but technically, not dominance as in Lemma 7, since the stop condition may not become defined.

The parameter c in $\overline{m}^C_c$ allows us to tune the algorithm to trade off running time for a smaller constant of domination. We will usually omit it when it is not relevant to the context.

Putting all this together, we have achieved our aim:

Theorem 3 *For a sufficient model class C, $-\log \overline{m}^C$ is a safe, computable approximation of $K(x)$ against any adversary from C*

Proof. We have shown that, under these conditions, $-\log m^C$ safely approximates $-\log m$ which dominates K, and that $-\log \overline{m}^C$ dominates $-\log m^C$. Since domination implies safe approximation (Lemma 4), and safe approximation is transitive (Lemma 5), we have proved the theorem. □

The negative logarithm of m^C will be our go-to approximation of K, so we will abbreviate it with κ:

Definition 7 $\kappa^C(x) = -\log m^C(x)$ *and* $\overline{\kappa}^C(x) = -\log \overline{m}^C(x)$.

For adversaries outside C, we cannot be sure that κ^C is safe:

Theorem 4 *There exist adversaries p_q with $T_q \notin C$ for which neither κ^C nor $\overline{\kappa}^C$ is a safe approximation of K.*

Proof. Consider the following algorithm for sampling from a computable distribution (which we will call p_q): Sample $n \in \mathbb{N}$ from some distribution $s(n)$ which decays polynomially. Loop over all x of length n return the first x such that $\kappa^C(x) \geq n$. At least one such x must exist by a counting argument: if all x of length n have $-\log \overline{m}^C(x) < n$ we have a code that assigns 2^n different strings to $2^n - 1$ different codes.

For each x sampled from q, we know that $\overline{\kappa}(x) \geq |x|$ and $K(x) \leq -\log p_q(x) + c_q$. Thus: $p_q(\overline{\kappa}^C(x) - K(x) \geq k) \;\geq\; p_q(|x| + \log p_q(x) - c_q \geq k)$
$= p_q(|x| + \log s(|x|) - c_q \geq k) \;=\; \sum_{n:n+\log s(n)-c_q \geq k} s(n)$.

Let n_0 be the smallest n for which $2n > n + \log s(n) - c_q$. For all $k > 2n_0$ we have $\sum_{n:n+\log s(n)-c_q \geq k} s(n) \geq \sum_{n:2n \geq k} s(n) \geq s\left(\frac{1}{2}k\right)$ □

For C^t (as in Example 1), we can sample the p_q constructed in the proof in $O(2^n \cdot t(n))$. Thus, we know that $\overline{\kappa}^t$ is safe for K against adversaries from C^t, and we know that it is unsafe against C^{2^t}.

5 Approximating Normalized Information Distance

Definition 8 ([2,6]) *The normalized information distance between two strings x and y is*

$$NID(x, y) = \frac{\max\left[K(x \mid y), K(y \mid x)\right]}{\max\left[K(x), K(y)\right]} .$$

The information distance (ID) is the numerator of this function. The NID is neither lower nor upper semicomputable [7]. Here, we investigate whether we can safely approximate either function using κ. We define ID^C and NID^C as the ID and NID functions with K replaced by $\overline{\kappa}^C$. We first show that, even if the adversary only combines functions and distributions in C, ID^C may be an unsafe approximation.

Definition 9 [7]*A function f is a (b-safe)* model-bounded one-way function *for C if it is injective, and for some $b > 1$, some $c > 0$, all $q \in C$ and all k: $p_q\left(\kappa^C(x) - \kappa^C\left(x \mid f(x)\right) \geq k\right) \leq cb^{-k}$.*

Theorem 5 † *Under the following assumptions:*

- *C contains a model T_0, with $p_0(x) = 2^{-|x|}s(|x|)$, with s a distribution on $\mathbb{N}$ which decays polynomially or slower,*
- *there exists a model-bounded one-way function f for C,*
- *C is* normal, *ie. for some c and all x: $\kappa^C(x) < |\overline{x}| + c$*

ID^C is an unsafe approximation for ID *against an adversary T_q which samples x from p_0 and returns $\overline{x}f(x)$.*

If x and y are sampled from C independently, we can prove safety:

Theorem 6 † *Let T_q be a Turing machine which samples x from p_a, y from p_b and returns $\overline{x}y$. If $T_a, T_b \in C$, $\mathrm{ID}^C(x, y)$ is a safe approximation for $\mathrm{ID}(x, y)$ against any such T_q.*

The proof relies on two facts: (1) $\overline{\kappa}^C(x \mid y)$ is safe for $K(x \mid y)$ if x and y are generated this way, (2) maximization is a *safety preserving operation.*

For *normalized* information distance, which is dimensionless, the error k in bits does not mean much. Instead, we use f/f_a as a measure of approximation error:

[7] This is similar to the Kolmogorov one-way function [14, Definition 11].

Theorem 7 † *We can approximate* NID *with* NID^C *with the following bound:*

$$p_q\left(\frac{\text{NID}(x,y)}{\text{NID}^C(x,y)} \notin \left(1-\frac{k}{c}, 1+\frac{k}{c}\right)\right) \leq c'b^{-k} + 2\epsilon$$

with $p_q(\text{ID}^C(x,y) \geq c) \leq \epsilon$ *and* $p_q\left(\max\left[\kappa^C(x), \kappa^C(y)\right] \geq c\right) \leq \epsilon$*, for some* $b > 1$ *and* $c' > 0$*, assuming that* p_q *samples* x *and* y *independently from models in* C*.*

6 Discussion

We have provided a computable function $\overline{\kappa}^C(x)$ for a given model class C. Under the assumption that x is produced by a model from C, approximates $K(x)$ in a probabilistic sense. We have also shown that $K^C(x)$ is not safe. Finally, we have given some insight into the conditions on C and the adversary, which can affect the safety of NCD as an approximation to NID.

Since, as shown in Example 1, resource-bounded Kolmogorov complexity is a variant of model-bounded Kolmogorov complexity, our results apply to K^t as well: K^t is not necessarily a safe approximation of K, even if the data can be sampled in t and $\overline{\kappa}^t$ *is* safe if the data can be sampled in t. Whether K^t is safe ultimately depends on whether a single shortest program dominates among the sum of all programs, as it does in the unbounded case.

For expensive model classes, we may be able to continue the chain of safe approximation proofs. Ideally, we would show that a model which is only locally optimal, found by an iterative method like gradient descent is still a safe approximation of K. Such proofs would truly close the circuit between the ideal world of Kolmogorov complexity and modern statistical practice.

Acknowledgements We would like to thank the reviewers for their insightful comments. This publication was supported by the Dutch national program COMMIT, the Netherlands eScience center, the ERDF (European Regional Development Fund) through the COMPETE Programme (Operational Programme for Competitiveness) and by National Funds through the FCT (Fundação para a Ciôncia e a Tecnologia, Portuguese Foundation for Science and Technology) within project *FCOMP-01-0124-FEDER-037281*.

References

1. Gács, P., Tromp, J.T., Vitányi, P.M.: Algorithmic statistics. Information Theory, IEEE Transactions on **47**(6) (2001) 2443–2463
2. Li, M., Chen, X., Li, X., Ma, B., Vitányi, P.M.: The similarity metric. Information Theory, IEEE Transactions on **50**(12) (2004) 3250–3264

3. Vitányi, P.M.: Meaningful information. Information Theory, IEEE Transactions on **52**(10) (2006) 4617–4626
4. Adriaans, P.: Facticity as the amount of self-descriptive information in a data set. arXiv preprint arXiv:1203.2245 (2012)
5. Gailly, J., Adler, M.: The gzip compressor (1991)
6. Cilibrasi, R., Vitányi, P.M.: Clustering by compression. Information Theory, IEEE Transactions on **51**(4) (2005) 1523–1545
7. Terwijn, S.A., Torenvliet, L., Vitányi, P.: Nonapproximability of the normalized information distance. Journal of Computer and System Sciences **77**(4) (2011) 738–742
8. Rissanen, J.: Modeling by shortest data description. Automatica **14**(5) (1978) 465–471
9. Rissanen, J.: Universal coding, information, prediction, and estimation. Information Theory, IEEE Transactions on **30**(4) (1984) 629–636
10. Grünwald, P.D.: The minimum description length principle. The MIT Press (2007)
11. Li, M., Vitanyi, P.M.: An introduction to Kolmogorov complexity and its applications. (1993)
12. Antunes, L., Matos, A., Souto, A., Vitányi, P.: Depth as randomness deficiency. Theory of Computing Systems **45**(4) (2009) 724–739
13. Hennie, F.C., Stearns, R.E.: Two-tape simulation of multitape Turing machines. Journal of the ACM (JACM) **13**(4) (1966) 533–546
14. Antunes, L., Matos, A., Pinto, A., Souto, A., Teixeira, A.: One-way functions using algorithmic and classical information theories. Theory of Computing Systems **52**(1) (2013) 162–178

A Appendix

Turing Machines and lsc. Probability Semimeasures (Lemma 1)

Definition 10 *A function $f : \mathbb{B} \to \mathbb{R}$ is* lower semicomputable (lsc.) *iff there exists a total, computable two-argument function $f' : \mathbb{B} \times \mathbb{N} \to \mathbb{Q}$ such that:* $\lim_{i\to\infty} f'(x,i) = f(x)$ *and for all i, $f'(x, i+1) \geq f'(x,i)$.*

Lemma 8 *If f is an lsc. probability semimeasure, then there exists a a function $f^*(x,i)$ with the same properties of the function f' from Definition 10, and the additional property that all values returned by f^* have finite binary expansions.*

Proof. Let x_j represent $x \in \mathbb{D}$ truncated at the first j bits of its binary expansion and x^j the remainder. Let $f^*(x,i) = f'(x,i)_i$. Since $f'(x,i) - f^*(x,i)_i$ is a value with $i+1$ as the highest non-zero bit in its binary expansion, $\lim_{i\to\infty} f^*(x,i) = \lim f'(x,i) = f(x)$.

It remains to show that f^* is nondecreasing in i. Let $x \geq y$, we will show that $x_j \geq y_j$, and thus $x_{j+1} \geq y_j$. If $x = y$ the result follows trivially. Otherwise, we have $x_j = x - x^j > y - x^j = y_j + y^j - x^j \geq y_j - 2^{-j}$. Substituting $x = f'(x, i+1)$ and $y = f'(x,i)$ tells us that $f^*(x, i+1) \geq f^*(x,i)$ □

Theorem 8 *Any TM, T_q, samples from an lsc. probability semimeasure.*

Proof. We will define a program computing a function $p'_q(x, i)$ to approximate $p_q(x)$: Dovetail the computation of T_q on all inputs $x \in \mathbb{B}$ for i cycles.

Clearly this function is nondecreasing. To show that it goes to $p(x)$ with i, we first note that for a given i_0 there is a j such that, $2^{-j-1} < p_q(x) - p_q(x, i_0) \leq 2^{-j}$. Let $\{p_i\}$ be an ordering of the programs producing x, by increasing length, that have not yet stopped at dovetailing cycle i_0. There is an m such that $\sum_{i=1}^{m} 2^{-|p_i|} \geq 2^{-j-1}$, since $\sum_{i=1}^{\infty} 2^{-|p_i|} > 2^{-j-i}$. Let i_1 be the dovetailing cycle for which the last program below p_{m+1} halts. This gives us $p_q(x) - p_q(x, i_1) \leq 2^{-j-1}$. Thus, by induction, we can choose i to make $p(x) - p'(x, i)$ arbitrarily small. □

Theorem 9 *Any lsc. probability semimeasure can be sampled by a TM.*

Proof. Let $p(x)$ be an lsc. probability semimeasure and $p^*(x, i)$ as in Lemma 8. We assume—without loss of generality—that $p^*(x, 0) = 0$. Consider the following algorithm:

```
initialize $s \leftarrow \epsilon$, $r \leftarrow \epsilon$
for $c = 1, 2, \ldots$:
  for $x \in \{b \in \mathbb{B} : |b| \leq c\}$
    $d \leftarrow p^*(x, c - i + 1) - p^*(x, c - i)$
    $s \leftarrow s + d$
    add a random bit to $r$ until it is as long as $s$
    if $r < s$ then return $x$
```

The reader may verify that this program dovetails computation of $p^*(x, i)$ for increasing i for all x; the variable s contains the summed probability mass that has been encountered so far. Whenever s is incremented, mentally associate the interval $(s, s + d]$ with outcome x. Since $p^*(x, i)$ goes to $p(x)$ as i increases, the summed length of the intervals associated with x goes to $p(x)$ and s itself goes to $\overline{s} = \sum_x p(x)$. We can therefore sample from p by picking a number r that is uniformly random on $[0, 1]$ and returning the outcome associated with the interval containing r. Since s must have finite length (due to the construction of p^*), we only need to know r up to finite precision to be able to determine which interval it falls in; this allows r to be generated on the fly. The algorithm halts unless r falls in the interval $[\overline{s}, 1]$, which corresponds exactly to the deficiency of p: if p is a semimeasure, we expect the non-halting probability of a TM sampling it to correspond to $1 - \sum_x p(x)$. □

Theorems 8 and 9 combined prove that the class of distributions sampled by Turing machines equals the lower semicomputable semimeasures (Lemma 1).

Unsafe Approximation of ID (Theorem 5)

Proof.

$$\begin{aligned}
&p_q\left(\mathrm{ID}^C(x,y) - \mathrm{ID}(x,y) \geq k\right) = \\
&\quad p_0\left(\max\left[\overline{\kappa}^C(x \mid f(x)), \overline{\kappa}^C(f(x) \mid x)\right] - \max\left[K(x \mid f(x)), K(f(x) \mid x))\right] \geq k\right)\ . \\
&p_q\left(|x| - \mathrm{ID}^C(x,y) \geq 2k\right) \ \leq\ p_0\left(|x| - \overline{\kappa}^C(x \mid f(x)) \geq 2k\right) \\
&\quad \leq\ p_0\left(|x| - \kappa^C(x) \geq k \ \vee\ \kappa^C(x) - \overline{\kappa}^C(x \mid f(x)) \geq k\right) \\
&\quad \leq\ p_0\left(|x| - \kappa^C(x) \geq k\right) \ +\ p_0\left(\kappa^C(x) - \kappa^C(x \mid f(x)) \geq k\right) \ \leq\ 2^{-k} + cb^{-k}\,.
\end{aligned}$$

K can invert $f(x)$, so $\mathrm{ID}(x,y) = \max\left[K(x \mid f(x)), K(f(x) \mid x)\right] = \max\left[|f^*|, |f^*_{\text{inv}}|\right] < c_f$. Where f^* and f^*_{inv} are the shortest program to compute f on U and the shortest program to compute the inverse of f on U respectively.

$$\begin{aligned}
&p_q\left(\mathrm{ID}^C(x,y) - \mathrm{ID}(x,y) \geq k\right) + p_q\left(|x| - \mathrm{ID}^C(x,y) \geq k\right) \\
&\quad \geq\ p_q\left(\mathrm{ID}^C(x,y) - \mathrm{ID}(x,y) \geq k \vee |x| - \mathrm{ID}^C(x,y) \geq k\right) \\
&\quad \geq\ p_q\left(|x| - \mathrm{ID}(x,y) \geq k\right) \geq\ p_0\left(|x| - c_f \geq k\right) = \sum\nolimits_{i \geq k - c_f} s(i)\,.
\end{aligned}$$

Which gives us:

$$\begin{aligned}
&p_q\left(\mathrm{ID}^C(x,y) - \mathrm{ID}(x,y) \geq k\right) \\
&\quad \geq\ -p_q(|x| - \mathrm{ID}^C \geq k) + \sum\nolimits_{i \geq k-|f|} s(i) \geq -cb^{-k} + \sum\nolimits_{i \geq k-|f|} s(i) \\
&\quad \geq s(k - |f|) - cb^{-k} \geq c' s(k) \qquad \text{for the right } c'.
\end{aligned}$$

□

Corollary 1 *Under the assumptions of Theorem 5, $\overline{\kappa}^C(x \mid y)$ is an unsafe approximation for $K(x \mid y)$ against q.*

Proof. Assuming $\overline{\kappa}^C$ is safe, then since max is safety-preserving (Lemma 10), ID^C should be safe for ID. Since it isn't, $\overline{\kappa}^C$ cannot be safe. □

Safe Approximation of ID (Theorem 6)

Lemma 9 *If q samples x and y independently from models in C, then κ^C is a 2-safe approximation of $\kappa(x \mid y) = -\log m^C(x \mid y)$ against q.*

Proof. Let q sample x from p_r and y from p_s.

$$\begin{aligned}
&p_q(-\log m^C(x \mid y) + \log m(x \mid y) \geq k) && = p_q(m(x \mid y)/m^C(x \mid y) \geq 2^k) \\
&\quad \leq\ 2^{-k} E\left[m(x \mid y)/m^C(x \mid y)\right] && = 2^{-k} \sum_{x,y} p_s(y) m(x \mid y) \frac{p_r(x)}{m^C(x \mid y)} \\
&\quad \leq\ c2^{-k} \sum_{x,y} p_s(y) m(x \mid y) \frac{m^C(x \mid y)}{m^C(x \mid y)} && \leq c2^{-k} \sum_{x,y} p_s(y) m(x \mid y) \leq c2^{-k}\,.
\end{aligned}$$

□

Since m and K mutually dominate, $-\log m^C$ is 2-safe for $K(x \mid y)$, as is $\overline{\kappa}(x \mid y)$.

Lemma 10 *If f_a is safe for f against q, and g_a is safe for g against q, then* $\max[f_a, g_a]$ *is safe for* $\max[f, g]$ *against q* [8].

Proof. We split $\mathbb{B}$ into sets A_k, B_k so that $p(B_k) = 0$ and $p(A_k) \leq cb^{-k}$:

$A_k = \{x : f_a(x) - f(x) \geq k \vee g_a - g(x) \geq k\}$: Since both f_a and g_a are safe, we know that $p_q(A)$ will be bounded above by the sum of two inverse exponentials in k, which from a given k_0 is itself bounded by an exponential in k.

$B_k = \{x : f_a(x) - f(x) < k \wedge g_a - g(x) < k\}$: We want to show that B contains no strings with error over k. If, for a given x the left and right max functions in $\max[f_a, g_a] - \max[f, g]$ select the outcome from matching functions, and the error is below k by definition. Assume then, that a different function is selected on each side. Without loss of generality, we can say that $\max f_a, g_a = f_a$ and $\max f, g = g$. This gives us: $\max f_a, g_a - \max f, g = f_a - g \leq f_a - f \leq k$. □

Corollary 2 ID^C *is a safe approximation of* ID *against sources that sample x and y independently from models in C.*

Safe approximation of NID (Theorem 7)

Lemma 11 *Let f and g be two functions, with f_a and g_a their safe approximations against adversary p_q. Let $h(x) = f(x)/g(x)$ and $h_a(x) = f_a(x)/g_a(x)$. Let $c > 1$ and $0 < \epsilon \ll 1$ be constants such that $p_q(f_a(x) \geq c) \leq \epsilon$ and $p_q(g_a(x) \geq c) \leq \epsilon$. We can show that for some $b > 1$ and $c > 0$*

$$p_q\left(\left|\frac{h(x)}{h_a(x)} - 1\right| \geq \frac{k}{c}\right) \leq cb^{-k} + 2\epsilon\,.$$

Proof. We will first prove the bound from above, using f_a's safety, and then the bound from below using g_a's safety.

$$\begin{aligned}
p_q\left(\frac{h}{h_a} \leq 1 - \frac{k}{c}\right) \;&\leq\; p_q\left(\frac{h}{h_a} \leq 1 - \frac{k}{c} \;\&\; c < f_a\right) + \epsilon \;\leq\; p_q\left(\frac{h}{h_a} \leq 1 - \frac{k}{f_a}\right) + \epsilon \\
&= p_q\left(\frac{f}{f_a}\frac{g_a}{g} \leq 1 - \frac{k}{f_a}\right) + \epsilon \leq p_q\left(\frac{f}{f_a} \leq 1 - \frac{k}{f_a}\right) + \epsilon \\
&= p_q\left(\frac{f + k}{f_a} \leq 1\right) + \epsilon \;=\; p_q\left(f_a - f \geq k\right) + \epsilon \;\leq\; c_f {b_f}^{-k} + \epsilon\,.
\end{aligned}$$

The other bound we prove similarly. Combining the two, we get:
$p_q\left(h/h_a \notin (k/c - 1, k/c + 1)\right) \leq c_f {b_f}^{-k} + c_g {b_g}^{-k} + 2\epsilon \;\leq\; c' {b'}^{-k} + 2\epsilon\,.$ □

Theorem 7 follows as a corollary.

[8] We will call such operations *safety preserving*