IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 1 May 2024, accepted 22 May 2024, date of publication 28 May 2024, date of current version 7 June 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3406912

RESEARCH ARTICLE

On Nonlinear Compression Costs: When Shannon Meets Rényi

ANDREA SOMAZZI[1,2], PAOLO FERRAGINA[3], AND DIEGO GARLASCHELLI[1,4,5]

[1]IMT School for Advanced Studies Lucca, 55100 Lucca, Italy
[2]Scuola Normale Superiore, 56126 Pisa, Italy
[3]Department of Computer Science, University of Pisa, 56127 Pisa, Italy
[4]Lorentz Institute for Theoretical Physics, 2333 CA Leiden, The Netherlands
[5]INdAM-GNAMPA Istituto Nazionale di Alta Matematica, 00185 Rome, Italy

Corresponding author: Andrea Somazzi (andrea.somazzi@imtlucca.it)

The work of Andrea Somazzi was supported by Dutch Econophysics Foundation (Stichting Econophysics, Leiden, The Netherlands). The work of Paolo Ferragina was supported in part by the European Union–Horizon 2020 Program through the scheme "INFRAIA-01-2018–2019–Integrating Activities for Advanced Communities, SoBigData++: European Integrated Infrastructure for Social Mining and Big Data Analytics" http://www.sobigdata.eu under Grant 871042; in part by the NextGenerationEU–National Recovery and Resilience Plan (Piano Nazionale di Ripresa e Resilienza, PNRR) through the Project "SoBigData.it–Strengthening the Italian RI for Social Mining and Big Data Analytics" under Grant Prot. IR0000013–Avviso n. 3264 del 28/12/2021; in part by the spoke "FutureHPC and BigData" of the ICSC–Centro Nazionale di Ricerca in High-Performance Computing, Big Data and Quantum Computing funded by European Union–NextGenerationEU–PNRR. The work of Diego Garlaschelli was supported in part by the European Union–Horizon 2020 Program through the scheme "INFRAIA-01-2018–2019–Integrating Activities for Advanced Communities, SoBigData++: European Integrated Infrastructure for Social Mining and Big Data Analytics" http://www.sobigdata.eu under Grant 871042; in part by the NextGenerationEU–National Recovery and Resilience Plan (Piano Nazionale di Ripresa e Resilienza, PNRR) through the Project "SoBigData.it—Strengthening the Italian RI for Social Mining and Big Data Analytics" under Grant Prot. IR0000013–Avviso n. 3264 del 28/12/2021; and in part by Dutch Econophysics Foundation (Stichting Econophysics, Leiden, The Netherlands).

ABSTRACT In compression problems, the minimum average codeword length is achieved by Shannon entropy, and efficient coding schemes such as Arithmetic Coding (AC) achieve optimal compression. In contrast, when minimizing the exponential average length, Rényi entropy emerges as a compression lower bound. This paper presents a novel approach that extends and applies the AC model to achieve results that are arbitrarily close to Rényi's lower bound. While rooted in the theoretical framework assuming independent and identically distributed symbols, the empirical testing of this generalized AC model on a Wikipedia dataset with correlated symbols reveals significant performance enhancements over its classical counterpart, when considering the exponential average. The paper also demonstrates an intriguing equivalence between minimizing the exponential average and minimizing the likelihood of exceeding a predetermined threshold in codewords' length. An extensive experimental comparison between generalized and classical AC unveils a remarkable reduction, by several orders of magnitude, in the fraction of codewords surpassing the specified threshold in the Wikipedia dataset.

INDEX TERMS Arithmetic coding, Campbell theorem, large deviations, Rényi entropy.

I. INTRODUCTION

In the realm of (lossless) data compression, the main goal is to efficiently represent data in a manner that requires reduced space without compromising its integrity. At the heart of this challenge lies the encoding strategy, which determines how individual symbols or sequences of symbols are transformed into compressed representations. Traditionally, these strategies aim to minimize the average length of the encoded symbols. By achieving a shorter average encoded symbol length, one can ensure a more compact representation of the entire input data and reduce the cost associated to encoding/decoding, if such cost is linearly related to the codewords' length, thereby achieving the central objective of many data compression problems. However, there are scenarios in which it could be desirable to consider a different type of average, namely the *exponential average*. The utility of the exponential average in data compression can be

The associate editor coordinating the review of this manuscript and approving it for publication was Luca Barletta.

understood from two distinct fronts. Firstly, when the costs associated with encoding or decoding amplify, they might grow exponentially with respect to the codewords' length. This leads to a nonlinear relation between compression costs and codewords' lengths. A case potentially falling into such a scenario is DNA coding, where the apparatus involved in encoding and decoding procedures is very costly [1], [2]. Minimizing an exponential cost function could then become essential for effective and efficient data storage. Secondly, at a more theoretical level, the exponential average arises naturally when aiming at curtailing the risk of buffer overflow [3], [4] or bolstering the probability of transmitting a message in a short time frame. This is a typical situation in aerospace communication scenarios, where antennas may be visible for a short fleeting moment, thus necessitating rapid and reliable transmission of information [5]. In such scenarios, estimating the likelihood of large deviations (for these undesired events) involves the cumulant generating function of the probability distribution, which in turn leads to the exponential average. Moreover, it has been shown that minimizing the exponential average, for a certain range of its parameters, is related to maximizing the chance of receiving a message in a single snapshot [6]. Therefore this paper is devoted to studying and designing a coding scheme that is suited to the case in which its costs are non-linear, specifically exponential, in the codewords' length; building on Campbell's classical result [7], we show that the proposed encoding scheme achieves the Rényi entropy, which emerges as the compression lower bound.

Consider a stationary source generating symbols from an alphabet $\Sigma = \{x_1, \dots, x_N\}$ of size $|\Sigma| = N$, with probability $p = \{p_1, \dots, p_N\}$. Then, the "classical" compression problem consists in finding the encoding strategy which maps each symbol $x_i \in \Sigma$ into a D-ary codeword of length $\ell_D(i)$ such that

$$L(0) = \sum_{i=1}^{N} p_i \ell_D(i) \tag{1}$$

is minimized. $L(0)$ is the codewords' average length, and the use of such notation will be clarified later.

In his pioneering work [8], Shannon proved that for a source generating i.i.d. symbols, Eqn. (1) is minimized by all encoding strategies such that $\ell_D(i) = -\log_D p_i$, for all $i = 1, 2, \dots, N$. However, in most cases, strategies that guarantee such equality for each symbol do not exist but only get "close" to it. This leads to the notorious relation

$$L(0) \geq H_1[p], \tag{2}$$

where $H_1[p] = -\sum_{i=1}^{N} p_i \log_D p_i$ is the Shannon entropy of the source, which can be understood as the codewords' minimum average length. The use of the subscript in H_1 will also be clarified later.

Moreover, Eqn. (1) can be seen as a *cost function* C, because minimizing Eqn. (1) is equivalent to minimizing the cost of encoding/decoding $C(0) \propto L(0)$ under the assumption that such cost is linear in the codewords' length.

Beyond the conventional focus on the linear average of codeword lengths, it's essential to acknowledge that this is not the only viable metric to target for minimization, as we briefly mentioned before. Delving deeper into the theoretical underpinnings of averages, we encounter the Kolmogorov-Nagumo (KN) averages [9], [10]: a more general family of averages that offers a richer landscape for exploration. One might be driven to consider minimizing these KN averages, recognizing the possibility of uncovering novel compression strategies and further refining data representation techniques that are suitable in different scenarios. Following the introduced notation, the codewords' KN average length is defined as

$$\langle \ell_D \rangle_\varphi = \varphi^{-1}\left(\sum_{i=1}^{N} p_i \, \varphi(\ell_D(i)) \right), \tag{3}$$

where φ is a continuous injective function. Note that for $\varphi(x) = x$ the usual average length (1) is recovered. While, in general, KN averages depend on φ, there is a natural requirement that an average length measure should satisfy, that restricts the space of admissible functions [11], [12]. Namely, it should be *additive* for independent symbols. In particular, consider two independent sets of symbols $\Sigma^{(1)} = \{x_1, \dots, x_N\}$ and $\Sigma^{(2)} = \{y_1, \dots, y_M\}$, respectively. The associated probabilities are $p = \{p_1, \dots, p_N\}$ and $q = \{q_1, \dots, q_M\}$, and each symbol is encoded in a codeword of length $\{\ell_D^{(1)}(i)\}_{i=1}^{N}$ and $\{\ell_D^{(2)}(j)\}_{j=1}^{M}$. Then, the additivity requirement is formulated as follows:

$$\begin{aligned} &\varphi^{-1}\left(\sum_{i=1}^{N} \sum_{j=1}^{M} p_i q_j \, \varphi\big(\ell_D^{(1)}(i) + \ell_D^{(2)}(j)\big) \right) \\ &= \varphi^{-1}\left(\sum_{i=1}^{N} p_i \varphi(\ell_D^{(1)}(i)) \right) + \varphi^{-1}\left(\sum_{j=1}^{M} q_j \varphi(\ell_D^{(2)}(j)) \right). \end{aligned} \tag{4}$$

It is possible to prove that Eqn. (4) leads to the so-called exponential KN averages corresponding to $\varphi(x) = \varphi_t(x) = \gamma D^{tx} + b$, where γ, t and b are real parameters and $\gamma t > 0$ (see Sec. III of [12] for a detailed derivation, and [13], [14] for further discussion). Substituting φ_t into Eqn. (3), one gets that

$$\langle \ell_D \rangle_{\varphi_t} \equiv L(t) = \frac{1}{t} \log_D \left(\sum_{i=1}^{N} p_i \, D^{t \, \ell_D(i)} \right), \tag{5}$$

where $t > -1$. $L(t)$ is then the *exponential average* of the codewords' length, independent of both γ and b. Notice that for t approaching 0, the exponential average converges to the linear average, i.e. $\lim_{t\to 0} L(t) = L(0)$, which clarifies the notation we have adopted before. In fact, by applying L'Hôpital's rule:

$$\lim_{t\to 0} L(t) = \lim_{t\to 0} \frac{(\ln D) \sum_{i=1}^{N} \ell_D(i) \, p_i \, D^{t \, \ell_D(i)}}{(\ln D) \sum_{i=1}^{N} p_i \, D^{t \, \ell_D(i)}} = L(0). \tag{6}$$

In his valuable paper [7], Campbell proved that the optimal encoding lengths that minimize the exponential cost of Eqn. (5) are

$$\ell_D^{(q)}(i) = -\log_D \frac{p_i^q}{\sum_{j=1}^N p_j^q}, \tag{7}$$

where $q = 1/(1+t)$. Moreover, he proved that the lower bound for the exponential cost is given by the Rényi entropy of order $q = 1/(1+t)$ of the source, defined as

$$H_q[p] = \frac{1}{1-q} \log_D \left(\sum_{i=1}^N p_i^q \right), \tag{8}$$

so that

$$L(t) \geq H_{\frac{1}{1+t}}[p] \tag{9}$$

where the equality holds iff Eqn. (7) is exactly satisfied. Note that $\lim_{q\to 1} H_q[p] = H_1[p]$, i.e. Shannon entropy is a particular case of Rényi entropy. It follows that for $t \to 0$, Eqn. (9) reduces to Eqn. (2).

The probability distribution $p^{(q)} = \left\{ \frac{p_1^q}{\sum_{j=1}^N p_j^q}, \ldots, \frac{p_N^q}{\sum_{j=1}^N p_j^q} \right\}$ which appears in Eqn. (7) is often referred as *escort* or *zooming* probability distribution of p [15], [16], [17]. The reason is that, depending on the value of q, it can amplify/suppress values in the tails of the original distribution p (and, since it is normalized, suppress/amplify the others). Escort distributions have been applied and have emerged in various fields, ranging from non-extensive statistical mechanics [15], chaotic systems [16] and statistical inference [17]. Another notable link among the Rényi entropy, the KN exponential average, and escort distributions comes from an axiomatic point of view. While Shannon entropy can be derived by the four Shannon-Khinchin axioms (SK1-SK4) [18], Rényi entropy is derived by relaxing SK4 (also called *additivity* axiom) to a more general version, which involves both the KN exponential average and the escort distributions [19].

Since Campbell, from the point of view of data compression problems, escort distributions are also the optimal distributions according to which one has to encode symbols in order to minimize the exponential average codeword length $L(t)$. In fact, as we will show in the next sections, depending on the value of the parameter t, the amplification/suppression of the codewords' lengths (with respect to the $q = 1$ classical scenario) driven by Eqn. (7) leads to great advantages in the case of a cost which grows exponentially with such lengths. However, although Campbell provided the existence of an optimal encoding length, he did not suggest any operational strategy to achieve it. Some specific algorithms have been later proposed [3], [4], [20], [21], and [22] noted that, since the optimal lengths defined in Eqn. (7) have the same form of the lengths which minimize the linear average length of Eqn. (1) if p is replaced by its escort $p^{(q)}$, then it is sufficient to feed a standard (i.e. "Shannonian") encoder with $p^{(q)}$ instead of p in order to reach a cost $L(t)$ close to its minimum $H_{\frac{1}{1+t}}[p]$.

In this paper, we provide a series of contributions. i) We lay the mathematical ground to the observations of the previous papers by applying the above conceptual framework to one of the most efficacious algorithms in the realm of data compression: i.e., Arithmetic Coding (AC) (Sec. II). ii) We experimentally analyze the performance of the proposed escort distribution-based compressor in the case of optimizing the exponential average codeword length, over both synthetic and real datasets. We confirm the theoretical results on the former (composed by i.i.d. generated symbols) and achieve surprising results on the latter (composed by correlated symbols). In particular, we show that on a sample of Wikipedia text the application of our compressor with escort probability leads to an improved compression ratio (when the considered metric is the exponential average codeword length) with respect to a standard Shannon compressor, even if the optimal value of q (i.e. the exponent leading to the escort distribution) is unknown to the encoder (Sec. III). iii) Finally, we examine analytically and experimentally the practical case in which its crucial to not exceed a certain threshold in the codewords' lengths (such as in the context of bounded buffers), by showing that the exponential average naturally appears in the probability of large deviations thus further justifying the study performed in the present paper. In particular, we will show that by using our approach it is possible to significantly reduce the probability that the length of the codeword assigned to a given sequence of symbols exceeds a certain threshold with respect to a classic Shannon compressor (Sec. IV).

For the sake of presentation, Tables 1 and 2 report the notation and the main formulas discussed in this paper.

II. METHODS

In the ensuing section, we undertake an examination of the arithmetic coding compression scheme. We commence by providing a theoretical description of AC and delineating its operating principles. Following this, we weigh the pros and cons of AC, offering a balanced viewpoint on its utility and limitations in various application contexts. Finally, we advance the discourse by generalizing AC with an aim to achieve the theoretical limit as predicted by Campbell's theorem.

A. ARITHMETIC CODING

Arithmetic coding is a lossless encoding scheme [23]. Compressor and decompressor both need the alphabet of symbols Σ, the associated probability distribution p and the length of the stream of symbols to encode/decode. Consider a string $\vec{s} = (s_1, \ldots, s_M)$ of length M, where each $s_j = x_{i_j}$ is a symbol randomly generated by a source from alphabet Σ with associated probability p. Then, in order to encode $\vec{s}$ into a D-ary alphabet, the encoder performs the procedure illustrated in Algorithm 1.

Essentially the encoder, starting from the interval $[0, 1)$, iteratively divides it proportionally to the probabilities in p and, at each iteration j, chooses the subinterval corresponding

TABLE 1. Notation table.

Σ	Source alphabet	x_i	Symbol in Σ
D	Size of the encoding alphabet	$\ell_D(i)$	Length of D-ary codeword of symbol x_i
p	Source probability distribution of symbols	$L(t)$	Exponential average of symbols' codewords' length
$H_q[p]$	Rényi entropy of order q of the probabilty distribution p	$p^{(q)}$	Escort distribution of order q of probability p
$\vec{s}$	String, i.e. sequence of symbols	M	Length of a string
$\ell_D(\vec{s})$	Length of D-ary codeword of string $\vec{s}$	$L_M(t)$	Exponential average of string's codewords' length, with original length M
$L_M^{emp}(t,q)$	Empirical exponential average of strings' codewords' length, with original length M and encoded according to their escort distribution of order q	A	String's codewords' length threshold
$\mu(t)$	Symbols' distribution cumulant generating function	$D_{KL}[p\|\|r]$	Kullback-Leibler divergence between distributions p and r
$H_1[p\|\|r]$	Cross entropy between distributions p and r	$\mathrm{ER}_q[p\|\|r]$	Exponential average "waste" of encoding with distribution r instead of the true p

TABLE 2. Summary table.

	Shannon (AC_1)	Campbell (AC_q)
Quantity to minimize	$L(0) = \sum_{i=1}^{N} p_i \ell_D(i)$	$L(t) = \frac{1}{t}\log_D\left(\sum_{i=1}^{N} p_i\, D^{t\,\ell_D(i)}\right)$
Theoretical minimum	$H_1[p] = -\sum_{i=1}^{N} p_i \log_D p_i$	$H_q[p] = \frac{1}{1-q}\log_D\left(\sum_{i=1}^{N} p_i^q\right)$
Optimal strategy	$\ell_D(i) = -\log_D p_i$	$\ell_D^{(q)}(i) = -\log_D \frac{p_i^q}{\sum_{j=1}^{N} p_j^q}$
Arithmetic Coding Performance	$\frac{\ell_2(\vec{s})}{M} < \frac{2}{M} + H_0[p]$	$L(t) = \frac{L_M(t)}{M} < \frac{2}{M} + H_{\frac{1}{1+t}}[p]$
Relation to minimize probability of exceeding threshold a	NA	$H_1[p^{(q^*)}] = a$
Error function (p true, r estimate)	$D_{KL}[p\|\|r] = \sum_{i=1}^{N} p_i \log \frac{p_i}{r_i}$	$\mathrm{ER}_q[p\|\|r] = \frac{q}{1-q}\log_D \sum_{i=1}^{N} p_i r_i^{q-1} + (1-q)H_q[r] - H_q[p]$

Algorithm 1 Arithmetic Coding

Require: The input string $\vec{s} = x_{i_1} x_{i_2} \cdots x_{i_M}$, the probabilities $p = \{p_1, \ldots, p_N\}$ and the cumulative $f = \{f_1, \ldots, f_N\}$ of p.

Ensure: A subinterval $[a, a + \mathcal{S})$ of $[0, 1)$.

1: $\mathcal{S}_0 = 1$
2: $a_0 = 0$
3: $j = 1$
4: **while** $j < M$ **do**
5: $\quad \mathcal{S}_j = \mathcal{S}_{j-1} \cdot p_{i_j}$
6: $\quad a_j = a_{j-1} + \mathcal{S}_{j-1} \cdot f_{i_j}$
7: $\quad j = j + 1$
8: **end while**
9: **return** $\langle k \in [a_M, a_M + \mathcal{S}_M), M \rangle$

to the associated symbol $s_j = x_{i_j}$. After M iterations, the encoder emits a number k, contained in the final subinterval $[a_M, a_M + \mathcal{S}_M)$, with $\mathcal{S}_M = \prod_{j=1}^{M} p_{i_j}$, which is uniquely associated with the original string $\vec{s}$. Such number k is then converted into its D-ary representation and communicated to the decoder (together with the original string length M), which can reverse this procedure to get the original string. It follows that the encoded string's length $\ell_D(\vec{s})$ is equal to the number of symbols (bits if $D = 2$) necessary to encode k in the desired alphabet.

From now on, we will consider for simplicity a binary ($D = 2$) encoding alphabet. Nonetheless, while our focus is on the classic binary AC, the results we present are inherently generalizable to $D > 2$, ensuring that the core features and principles of AC we discuss remain applicable and valid to those other cases too.

It is possible to show that the length of the encoded number k (i.e. encoded string) depends only on the length of the final subinterval $\mathcal{S}_M$. In particular, by choosing $k = a_M + \mathcal{S}_M/2$ and by truncating its binary representation to the first $\lceil \log_2 \frac{2}{\mathcal{S}_M} \rceil$ bits, the approximation error is so small that such truncation is guaranteed to fall into the interval $[a_M, a_M + \mathcal{S}_M)$. Considering that

$$\ell_2(\vec{s}) = \left\lceil \log_2 \frac{2}{\mathcal{S}_M} \right\rceil < 2 - \log_2 \mathcal{S}_M = 2 - \sum_{j=1}^{M} \log_2 p_{i_j}, \tag{10}$$

and that it is possible, for M large enough, to approximate each p_i with the fraction of occurrences of symbol x_i in $\vec{s}$, i.e $p_i \simeq n_i(\vec{s})/M$, one gets that:

$$\ell_2(\vec{s}) < 2 + M \cdot H_0[p]. \tag{11}$$

Eqn. (11) unveils the main strength of the AC scheme: the number of bits that are "wasted" in encoding $\vec{s}$ is 2, thus resulting *intensive* with respect to the string length M (provided that we operate with infinite precision arithmetic [23]). As M increases, the number of wasted bits

per character goes to 0, in fact

$$\frac{\ell_2(\vec{s})}{M} < \frac{2}{M} + H_0[p]. \tag{12}$$

A primary limitation of arithmetic coding (AC) lies in its operational framework. Unlike certain encoding schemes that allocate distinct codewords to individual symbols, AC assigns a codeword to the entire string. This means that the decoding process cannot commence in tandem with encoding so the decoder must wait for the encoder's completion of encoding the entire string (see e.g. the variant Range Coding for relaxing this limitation [24]). As the efficiency of AC generally improves with an increase in M, this waiting period can be time-consuming, rendering AC unsuitable for some applications. Conversely, AC boasts superior performance compared to encoding mechanisms that designate codewords to each symbol particularly when probability distributions are highly skewed. Such encoders mandate a minimum of 1 bit per symbol. However, the optimal length — expressed as $-\log_2 p_i$ — can be significantly less than 1.

B. GENERALIZED AC

We now propose a generalization of AC in order to optimally minimize the exponential cost $L(t)$ defined in Eqn. (5). In analogy with the classical case, we try to execute AC by dividing each segment according to the escort distribution $p^{(q)}$, where $p_i^{(q)} = \frac{p_i^q}{\sum_{j=1}^N p_j^q}$, in order to reach the optimal lengths defined in Eqn. (7). We will call this procedure AC_q. Moreover, we will call $\mathcal{S}_j^{(q)}$ the length of the segment generated by AC_q at iteration j. The logarithm of the length of the final segment $\mathcal{S}_M^{(q)}$ for a string $\vec{s}$ is:

$$\begin{aligned}\log_D \mathcal{S}_M^{(q)}(\vec{s}) &= \log_D \prod_{j=1}^{M} \frac{p_{i_j}^q}{\sum_{i=1}^N p_i^q} \\ &= \sum_{j=1}^{M} \left(\log_D p_{i_j}^q - \log_D \sum_{i=1}^{N} p_i^q \right) \\ &= q \sum_{i=1}^{N} n_i(\vec{s}) \log_D p_i - M \log_D \sum_{j=1}^{N} p_j^q \end{aligned} \tag{13}$$

where $n_i(\vec{s})$ counts how many times the symbol x_i appears in the string $\vec{s}$. From this result, it is possible to evaluate the number of bits emitted to encode a particular string in a binary alphabet ($D = 2$):

$$\ell_2^{(q)}(\vec{s}) = \left\lceil \log_2 \frac{2}{\mathcal{S}_M^{(q)}(\vec{s})} \right\rceil < 2 - \log_2 \mathcal{S}_M^{(q)}(\vec{s}). \tag{14}$$

Let's define now the exponential cost $L_M(t)$ of a string of length M composed by independent symbols:

$$L_M(t) = \frac{1}{t} \log_2 \sum_{\vec{s}} P(\vec{s})\, 2^{t\ell_2(\vec{s})}, \tag{15}$$

where $P(\vec{s}) = \prod_{i=1}^N p_i^{n_i(\vec{s})} = \mathcal{S}_M(\vec{s})$. Given Eqn. 5, it is $L_M(t) = M \cdot L(t)$. Substituting Eqn. (14) in the definition of $L_M(t)$, and considering the optimal parameter value $q = 1/(1+t)$, we get that:

$$\begin{aligned} L_M(t) &= \frac{1}{t} \log_2 \sum_{\vec{s}} P(\vec{s})\, 2^{t\ell_2^{((t+1)^{-1})}(\vec{s})} \\ &< \frac{1}{t} \log_2 \sum_{\vec{s}} P(\vec{s})\, 2^{t(2-\log_2 \mathcal{S}_M^{((t+1)^{-1})}(\vec{s}))} \\ &= \frac{1}{t} \log_2 \left(2^{2t}\, 2^{tM \log_2 \sum_j p_{i_j}^{(t+1)^{-1}}} \right. \\ &\quad \left. \cdot \sum_{\vec{s}} \left(P(\vec{s}) \prod_{i=1}^{N} \left(p_i^{n_i(\vec{s})}\right)^{-\frac{t}{t+1}} \right) \right) \\ &= 2 + \frac{Mt}{t+1} H_{\frac{1}{t+1}}[p] + \frac{1}{t} \log_2 \sum_{\vec{s}} P(\vec{s})^{1-\frac{t}{t+1}} \\ &= 2 + \frac{Mt}{t+1} H_{\frac{1}{t+1}}[p] + \frac{M}{t+1} H_{\frac{1}{t+1}}[p] \\ &= 2 + M H_{\frac{1}{t+1}}[p]. \end{aligned} \tag{16}$$

Which reads:

$$L_M(t) < 2 + M H_{\frac{1}{t+1}}[p], \tag{17}$$

where $H_q[p] = \frac{1}{1-q} \log_2 \sum_{i=1}^N p_i^q$ is the Rényi entropy of the source for a single symbol. Notice that for independent symbols the Rényi entropy is additive, i.e. for i.i.d. symbols $H_q[P] = M \cdot H_q[p]$ holds. So, the compressor AC_q leads to an average cost per symbol which is close to $H_{\frac{1}{1+t}}[p]$ as M increases:

$$L(t) = \frac{L_M(t)}{M} < \frac{2}{M} + H_{\frac{1}{1+t}}[p]. \tag{18}$$

It is possible to visualize this result by considering the cost $L_M(t, q)$, in which the parameters t and q are now decoupled: t is the exponent of the cost function, while q is used in the AC_q procedure. In particular, it reads:

$$L_M(t, q) = \frac{1}{t} \log_2 \sum_{\vec{s}} P(\vec{s})\, 2^{t\,\ell_2^{(q)}(\vec{s})}. \tag{19}$$

Here, $\ell_2^{(q)}$ represents the number of bits emitted by applying the AC_q procedure with the escort distribution of order q (notice that, if $q = 1$, then the $\mathrm{AC}_{q=1}$ reduces to the classic compressor AC).

Figure 1 shows $L_M(t, q)$ for different values of q, and for $t = 0.8$ and $M = 50$. In this simple example, $N = 3$ and the source probability distribution follows a Zipf's law, i.e. $p_i \propto i^{-1}$. The minimum is reached exactly in the value of q predicted by Campbell, that we now call $q_t = \frac{1}{1+t}$. Moreover, the distance between the minimum of $L_M(t, q)$, i.e. $L_M(t, q_t)$, and the orange line (corresponding to $M \cdot H_{\frac{1}{1+t}}[p]$) is very close to 2, confirming the result of Eqn. (17).

C. A NOTE ON THE SEMI-STATIC APPROACH

An encoding scheme can be *static* or *semi-static*, depending on how the probability distribution of the source is computed or updated. In the first case, the probability distribution

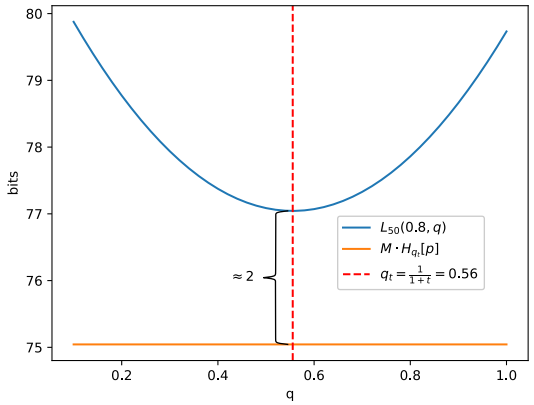

FIGURE 1. Exponential average codeword length of strings of length $M = 50$, composed by i.i.d. symbols sampled according to $p_i \propto i^{-1}$, $i \in [1, 3]$. Here $t = 0.8$. Its minimum is reached in the proximity of the red vertical dashed line, corresponding to the optimal q, i.e. q_t. For that value of q, the distance with respect to the Rényi entropy of the string (flat orange line) is approximately 2.

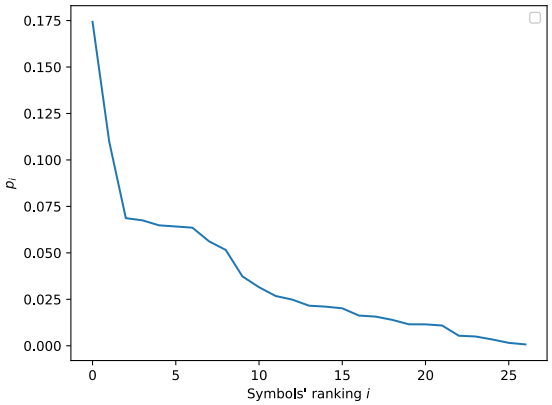

FIGURE 2. Probability distribution of the individual symbols (i.e., characters) in the Wikipedia dataset. Symbols have been ordered by decreasing frequency and assigned a rank. The probability distribution has been estimated in a frequentist approach as $p_i = n_i/W$, with n_i being the number of times that symbol x_i appears in Wikipedia.

approximating the source's one is fixed and never changed while symbols or strings are generated. In the second case, instead, the probability distribution is updated each time a string needs to be encoded, and it is set equal to the frequency of symbols appearing in that string. The main drawback of this scheme is the fact that the encoder must send to the decoder the model approximating the source's distribution each time a sequence of symbols is compressed. For example, in [25], the AMS coding is focused on the second case.

Let's assume that is possible to reach codewords' lengths as expressed in Eqn. (7), for any symbol and any value of q. Then, by calling r the model representing the source's distribution (thus practically used in the algorithm by the encoder and the decoder), it is possible to rewrite Eqn. (7) as:

$$\begin{aligned}\ell_D^{(q)}(\vec{s}) &= \sum_{j=1}^{M}\left(-\log_D \frac{r_{ij}^q}{\sum_{i=1}^{N} r_i^q}\right)\\ &= -\sum_{i=1}^{N} n_i(\vec{s}) \log_D \frac{r_i^q}{\sum_{i=1}^{N} r_i^q} = M \cdot H_1[f(\vec{s})||r^{(q)}],\end{aligned} \tag{20}$$

where $H_1[f\,||\,r] = -\sum_i f_i \log_D r_i$ is the *cross-entropy* between distributions f and r, and $f(\vec{s}) = \left(\frac{n_1(\vec{s})}{M}, \ldots, \frac{n_N(\vec{s})}{M}\right)$ is the empirical frequency of each symbol in the string $\vec{s}$. We also remind that $r^{(q)}$ is the escort distribution of order q of the distribution r. So, Eqn. (19) can be rewritten as:

$$L_M(t, q) = \frac{1}{t}\log_2\left(\sum_{\vec{s}} P(\vec{s})\, 2^{t\,M\,H_1[f(\vec{s})\,||\,r^{(q)}]}\right). \tag{21}$$

In a static scheme, r is kept constant and is ideally very "close" to p. Both the theoretical analysis of Campbell [7] and the results of this paper suppose that $r = p$. In such case, the exponential cost is minimized by setting $q = q_t = 1/(1+t)$. On the other hand, in a semi-static scheme, r is updated each time a string has to be encoded and set equal to the empirical frequency of observed symbols, i.e. $r = r(\vec{s}) = f(\vec{s})$. Then, Eqn. (21) can be rewritten as:

$$L_M(t, q) = \frac{1}{t}\log_2\left(\sum_{\vec{s}} P(\vec{s})\, 2^{t\,M\,H_1[f(\vec{s})\,||\,f^{(q)}(\vec{s})]}\right). \tag{22}$$

The exponential cost is then minimized by taking $q = 1$, because such choice minimizes each term in the sum due to the fact that the cross-entropy is minimal when its two input distributions are equal.

In other words, if r is updated each time a string has to be encoded, $q = 1$ remains the best choice. This scheme, however, has the disadvantage that the encoder must send the updated r to the decoder each time. On the other hand, in a static scheme our approach is optimal and q has to be set according to $q = q_t = 1/(1+t)$.

III. APPLICATION TO WIKIPEDIA

Having delineated the theoretical side of our AC_q in the preceding sections, we now transition to a more empirical scenario. This section is dedicated to the application of our outlined procedure to real-world data.

In particular, we applied AC_q to Wikipedia data.[1] The dataset used for our analysis contains $W \approx 7 \cdot 10^8$ symbols from an alphabet Σ of size $|\Sigma| = N = 27$. In order to perform coding in a static approach as we mentioned earlier, we computed from the whole dataset the empirical frequency of the 27 distinct symbols, shown in Fig.2, and then used it to set the probability distribution $p = \{p_1, \ldots, p_{27}\}$.

Since the theoretical results presented so far are valid for i.i.d. symbols, we first discuss and apply our procedure in the case of i.i.d. symbols. After that, we will move to the real Wikipedia dataset.

To begin with, we generated $\eta = 3.500.000$ strings of length $M = 20$ composed by i.i.d. symbols sampled

[1]The dataset FIL9 can be downloaded from https://fasttext.cc/docs/en/unsupervised-tutorial.html.

Algorithm 2 Wikipedia Data Analysis

Require: data, $\Sigma = (x_1, \ldots, x_N), p = (p_1, \ldots, p_N)$
Ensure: $\text{len}(data) = M \cdot \eta$
Ensure: $q_{end} > 0$
$M \leftarrow 20$
$\eta \leftarrow \frac{\text{len}(data)}{M}$
$q \leftarrow 0$
$q_{idx} \leftarrow 0$
while $q \leq q_{end}$ **do**
  $n \leftarrow 0$
  $p_i^{(q)} \leftarrow \frac{p_i^q}{\sum_j p_j^q} \quad \forall x_i \in \Sigma$
  while $n < \eta$ **do**
    string $\leftarrow$ data$[M \cdot n : M \cdot (n+1) - 1]$
    $\mathcal{S} \leftarrow 1$
    for c in string **do**
      $i^* \leftarrow i | c = x_i$
      $\mathcal{S} \leftarrow \mathcal{S} \cdot p_{i^*}^{(q)}$
    end for
    $\mathcal{L}[q_{idx}, n] \leftarrow \log_2 \lceil \frac{2}{\mathcal{S}} \rceil$
    $n \leftarrow n + 1$
  end while
  $q \leftarrow q + 0.1$
  $q_{idx} \leftarrow q_{idx} + 1$
end while

according to p. We then applied AC_q for different and discretized values of q. For each string we evaluated the length of the corresponding codeword generated by AC_q algorithm, without actually generating it, as $\ell_2^{(q)}(\vec{s}) = \log_2 \lceil \frac{2}{\mathcal{S}_M^{(q)}(\vec{s})} \rceil$. Such lengths have been stored in a matrix $\mathcal{L}$, whose entry $\mathcal{L}_{ij}$ is the length of the codeword of the j-th string, generated with AC_{q_i}, where q_i is the i-th value of q that we encode with. Algorithm 2 summarizes this procedure. By using the matrix $\mathcal{L}$ generated by Algorithm 2, it is possible to evaluate the empirical exponential average length, for different values of t, as:

$$L_M^{emp}(t, q_i) = \frac{1}{t} \log_2 \left(\frac{1}{\eta} \sum_{j=1}^{\eta} 2^{t \mathcal{L}_{ij}} \right). \tag{23}$$

Figure 3 shows the empirical L_M^{emp} as a function of q for three different values of t, with the corresponding Rényi entropy $M \cdot H_{q_t}$ and the optimal $q_t = 1/(1+t)$. It is clear that the minimum of $L_M^{emp}(t, q)$ is reached at $q = q_t$ and that it is very close to the Rényi entropy $M \cdot H_{q_t}[p]$.

Let us now move to analyze the real Wikipedia dataset. We divided it in $\eta = 35.653.488$ strings of length $M = 20$. We applied again Algorithm 2, and proceeded in the same way as we did for the i.i.d. symbols scenario. Figure 5 reports our results. In particular, we can see that $\text{argmin}_q(L_M^{emp}(t, q)) < q_t$ and that, for $t = 0.2$ (top panel), our AC_q can perform better than what Campbell predicted (in fact, $\min_q L_M^{emp}(t, q) < M \cdot H_{q_t}[p]$). The emergence of such discrepancies is not surprising, since real English text is not

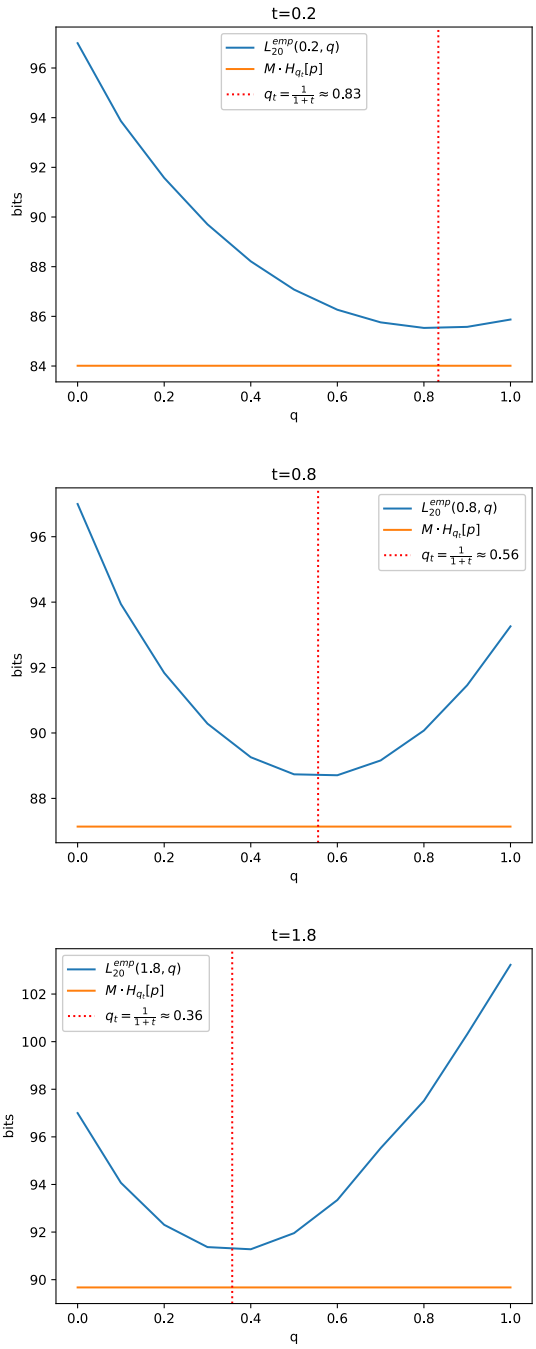

FIGURE 3. This figure shows, for the synthetic i.i.d. generated symbols, the empirical exponential average $L_M^{emp}(t, q)$ for $M = 20$ (blue line), the Rényi entropy $M \cdot H_{q_t}[p]$ (horizontal orange line) with $q_t = 1/(1+t)$ (dotted vertical red line). The three panels show different values of $t \in \{0.2, 0.8, 1.8\}$. We can see that, in each case, the minimum of $L_M^{emp}(t, q)$ is reached at $q = q_t$, and that $L_M^{emp}(t, q_t) - M \cdot H_{q_t}[p] \approx 2$.

composed by i.i.d. symbols, and thus the hypothesis on which our theoretical description lies is not satisfied.

But what does it mean, "physically", the fact that, in this case, the average empirical cost is minimized by considering a q smaller than q_t? Since we are using escort distributions $p^{(q)}$ of order q as encoding strategy in Eqn. (7), decreasing

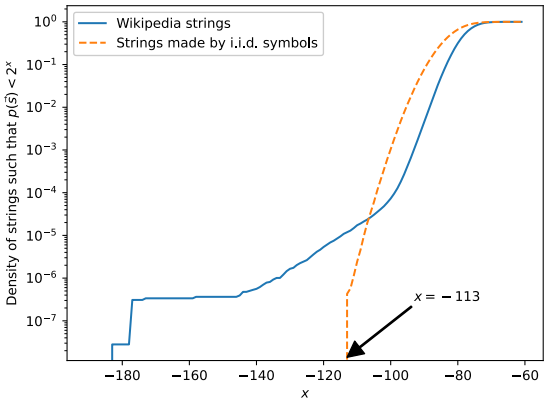

FIGURE 4. Fraction of strings with probability $p(\vec{s}) = \prod_{j=1}^{M} p_{i_j} < 2^x$, plotted against x, for the Wikipedia strings (blue solid line) and the i.i.d. generated strings (orange dashed line). Rare strings, i.e. those with a small $p(\vec{s})$ are more abundant in the Wikipedia dataset, as shown by the fact that for $x < -113$ the number i.i.d. generated strings with such a small probability vanishes, while the same is not true for the real dataset.

the value of q is equivalent to increasing the probability of the rare strings. This translates into assigning them shorter codewords, more than it would be done by using $q = q_t$. In other words, when the real optimal q is smaller than q_t, this means that "rare" strings are actually more abundant in the dataset than they would be if they were generated by a probability distribution calculated as the product of the probability of i.i.d. symbols. In Figure 4 we show that this is indeed true. We have counted, for both Wikipedia and i.i.d. generated strings, the fraction of strings such that $\log_2 p(\vec{s}) < x$. "Rare" strings, which correspond to a very small $p(\vec{s})$ and largely contribute to the exponential cost, are actually more abundant in the real dataset: while, for $x < -113$, there are no synthetic strings such that $\log_2 p(\vec{s}) < x$, there are many in the Wikipedia dataset. Moreover, Figure 6 shows, for different values of t, the real (empirical) optimal q overlapped to the theoretical q_t. For most values of the exponent t, the empirical best q is in fact smaller than q_t.

Finally, we want to stress that even if English text does not satisfy the i.i.d. symbols hypothesis on which Campbell's theoretical description lies, the use of AC_q still outperforms the standard AC if the average length is exponential, although the empirical optimal q is not the one predicted by Campbell. In fact, while the value of q that is actually optimal in the case real English text can not be known a priori, by using the one which is optimal for i.i.d. symbols (i.e. q_t) it is possible to significantly reduce the exponential average length, or the cost, with respect to the standard case $q = 1$. This is shown in Figure 7, where we can see that, even if the true optimal q is different from q_t, by encoding according to $q_t = \frac{1}{1+t}$ there is a notable exponential average length drop with respect to the usual $q = 1$ encoding strategy. Of course, if one would know the true optimal q the advantage would be even greater.

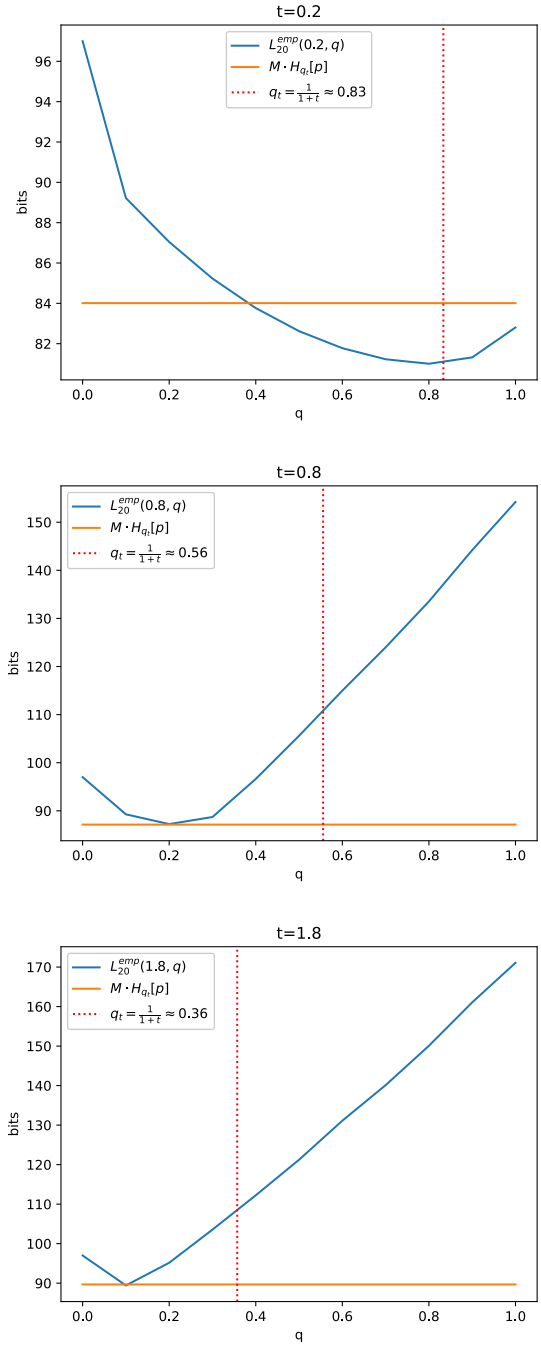

FIGURE 5. This figure shows, for the real Wikipedia dataset,, the empirical exponential average $L_M^{emp}(t, q)$ for $M = 20$ (blue line), the Rényi entropy $M \cdot H_{q_t}[p]$ (horizontal orange line) with $q_t = 1/(1+t)$ (dotted vertical red line). The three panels show different values of $t \in \{0.2, 0.8, 1.8\}$. In all these cases, it is instead evident that the minimum of $L_M^{emp}(t, q)$ is reached for $q < q_t$. Additionally, $\min_q L_M^{emp}(q, t)$ could be lower (first panel) or almost exactly reach (second and third panels) the value $M \cdot H_{q_t}[p]$. We can also see that encoding according to AC_{q_t} (i.e., see the intersection between the blue line and the dotted line) can lead to an exponential average length smaller than the Rényi entropy (first panel), or to an error which greater than 2, i.e. $\min_q L_M^{emp}(t, q) - M \cdot H_{q_t}[p] > 2$ (second and third panels).

IV. DISCUSSION

In this section, we'll take a closer look at the core ideas and findings from our research. We'll first explore one of

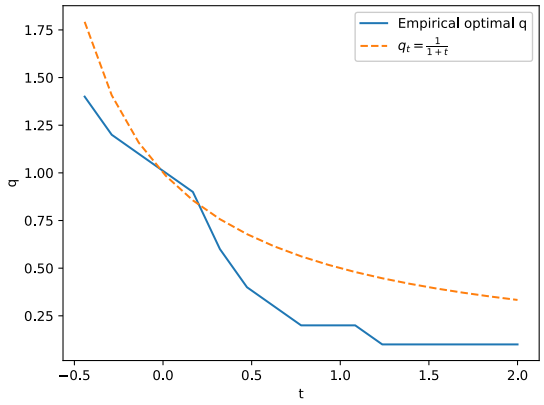

FIGURE 6. Empirical best q for the Wikipedia dataset (blue line solid line) and q_t (orange dashed line) for different values of t. For almost every value of t, the empirical optimal q is smaller than q_t, meaning that in the Wikipedia dataset there is an abundance of "rare" strings.

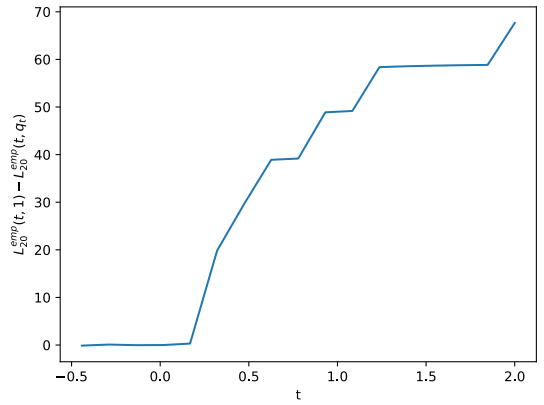

FIGURE 7. Difference $L_M^{emp}(t, q = 1) - L_M^{emp}(t, q = q_t)$ for $M = 20$ as a function of t, for the Wikipedia dataset. Since, for t big enough, this quantity is positive (and increasing), if the average to consider is the exponential one, there is an advantage (which increases with t) in encoding according to q_t instead of $q = 1$.

the reasons behind the use of the exponential cost in our study, thus explaining why it's important and how it fits into the bigger picture of data compression. After that, we will provide further analysis of real data, which confirms our previous findings. Moreover, we will also mention what are the errors that can come up when our guesses about the true probability distribution of the source are not accurate, shedding light on some of the challenges we faced and how they might be addressed in future studies.

A. A JUSTIFICATION TO THE EXPONENTIAL COST WITH CRAMÉR'S THEOREM

In this subsection, we provide a simple yet powerful idea about the usefulness of the exponential average and its minimization. Such idea relies on the linkage between the exponential average and the cumulant generating function of a distribution. As we anticipated in the introduction of this paper, such application could be useful in scenarios in which it is imperative to minimize the probability that codewords' lengths exceed a certain threshold.

Suppose we are interested in encoding strings of fixed length M, and we do not want the lengths of the corresponding codewords to exceed the threshold $A = M \cdot a$. With the usual notation, such problem translates into finding an encoding strategy $x_i \to \ell_D(i)$ that assigns to each symbol x_i of the alphabet Σ a length $\ell_D(i)$ to its codeword, minimizing the probability of exceeding the threshold:

$$\text{Prob}\left[\sum_{j=1}^{M} \ell_D(i_j) \geq A\right] = \text{Prob}\left[\frac{1}{M}\sum_{j=1}^{M} \ell_D(i_j) \geq a\right]. \quad (24)$$

Here, the sum runs over the whole string, and $\ell_D(i_j)$ is the length of the encoded symbol appearing at the j-th position of the string to be encoded. Moreover, since the threshold for the encoded strings is $M \cdot a$, we can see a as the threshold *per symbol*. According to Cramér's theorem, it is possible to write the following Chernoff bound:

$$\text{Prob}\left[\frac{1}{M}\sum_{j=1}^{M} \ell_D(i_j) \geq a\right] \leq e^{-M\left(ta-\mu(t)\right)} \quad \forall t > 0, \quad (25)$$

where $\mu(t) = \ln \mathbb{E}_p[e^{t\ell_D(i)}]$ is the symbols' distribution's cumulant-generating function and ln is the natural logarithm (i.e. with base e). Eqn. (25) gives us an important degree of control on the probability of exceeding the threshold, since, as we will show, it is possible to control its upper bound, i.e. the r.h.s. of the equation, by choosing an appropriate encoding strategy. Of course, we are interested in situations in which the exponent $-M\left(ta-\mu(t)\right)$ is negative, otherwise, we would get an upper bound of a probability distribution greater than 1, thus totally uninformative. It is possible to rewrite the exponent of the upper bound as:

$$\begin{aligned} -M(ta-\mu(t)) &= -M\left(ta - \log\left(\sum_i p_i e^{t\ell_D(i)}\right)\right) \\ &= -M\left(ta - \frac{t\log_D(\sum_i p_i D^{t\ell_D(i)\log_D e})}{t\log_D e}\right) \\ &= -M \cdot t(a - L(t\log_D e)). \end{aligned} \quad (26)$$

Since M, t and a are positive by definition, we are interested in finding the strategy minimizing $L(t\log_D e)$ $\forall t > 0$. We know that, for a given value of $t' = t\log_D e$, the minimum of $L(t')$ is $H_{q_{t'}}[p]$, with $q_{t'} = \frac{1}{1+t'}$. Moreover, we know that such minimum is reached with the strategy pointed out in Eqn. (7). So, substituting (7) in (26) and rewriting Eqn. (26) as a function of $q_{t'}$ (and, for simplicity, by dropping the subscript t'), we get that:

$$-M(ta-\mu(t)) = -M\frac{1-q}{q\log_D e}(a - H_q[p]), \quad (27)$$

for $q = 1/(1 + t\log_D e)$. So, it is possible to write Eqn. (25) as:

$$\text{Prob}\left[\frac{1}{M}\sum_{j=1}^{M} \ell_D(i_j) \geq a\right] \leq e^{-M\frac{1-q}{q\log_D e}(a-H_q[p])} \quad \forall q \in (0, 1]. \quad (28)$$

Having pointed out that the best strategy consists in setting the codewords lengths according to Eqn. (7) with $q = 1/(1 + t \log_D e)$, we have now to determine which is the correct $t > 0$ (and, in turn, q) to consider. We expect that such choice depends on the threshold a. In order to choose the best q, we will minimize the right-hand side of Eqn. (27). Since we are assuming it to be negative, this guarantees that the upper bound in Eqn. (28) is minimized.

Before going into the analytical details of such minimization, we will consider two simple examples which will provide an intuition on how the encoding strategy is related to the threshold a. Recall that $H_q[p]$ is a decreasing function of q, i.e., $H_0[p] \geq \cdots \geq H_1[p]$.

1) CASE $A > H_0[P] = \mathrm{LOG}_D\,|\Sigma|$

In the first case we consider, we assume that the threshold a is bigger than the Rényi entropy of order 0 of the source distribution. Since $H_0[p] = \log_D |\Sigma|$, this is equivalent to assume that the threshold exceeds the Shannon entropy of a distribution that shares the same support as the original p, but with entries replaced by $1/|\Sigma|$, i.e. a uniform distribution. In this scenario, the term $(a - H_q[p])$ in Eqn. (27) is positive and finite $\forall q \in (0, 1]$. The r.h.s. of Eqn. (27) is then maximized by letting $q \to 0$ (i.e. $t \to +\infty$). By writing $a = H_0[p] + \epsilon$, with $\epsilon > 0$, Eqn. (28) reads:

$$P\left[\frac{1}{M}\sum_{j=1}^{M}\ell_D(i_j) \geq H_0[p] + \epsilon\right] \leq \lim_{q\to 0} e^{-M\frac{1-q}{q\log_D e}\epsilon} = 0. \tag{29}$$

So, the probability of emitting a codeword longer than the threshold vanishes. This result is trivial: by setting $q \to 0$, the encoding strategy is equivalent to the Shannon encoding for symbols generated with a uniform probability distribution. In fact, $\ell_D^{(0)}(i) = -\log p_i^{(0)} = \log |\Sigma|$, and this holds for any probability distribution p. In other words, if it is imperative that the average codeword length does not exceed $H_0[p]$, just encode the sequence as if the symbols are uniformly distributed, irrespective of their actual probability distribution.

2) CASE $A < H_1[P]$

In this second case, we are going to consider a threshold smaller than the Shannon entropy of the underlying probability distribution p. So, it follows that $(a - H_q[p])$ is negative $\forall q$ because $H_0[p] \geq \cdots \geq H_1[p] > a$. Then, by setting $a = H_1[p] - \epsilon$, with $\epsilon > 0$, Eqn. (28) reads:

$$P\left[\frac{1}{M}\sum_{j=1}^{M}\ell_D(i_j) \geq H_1[p] - \epsilon\right] \leq e^{-M\frac{1-q}{q\log_d e}\left(H_1[p]-\epsilon-H_q[p]\right)}. \tag{30}$$

The exponent $-M\frac{1-q}{q\log_d e}\left(H_1[p] - \epsilon - H_q[p]\right) > 0$ is positive $\forall M \in \mathbb{N}$, and so it does not satisfy our hypothesis of a negative exponent. As previously mentioned, this means that the above right-hand side term is greater than 1. For this reason, it gives no information on the probability of exceeding the threshold. We can however see that since $H_1[p]$ is the shortest achievable codewords' (linear) average length, the latter can be smaller than $H_1[p]$ only due to fluctuations in the observed symbols frequency, which are suppressed in the large M limit. The best strategy is then letting $q = 1$, but still, the threshold will be exceeded almost always if M is not unrealistically small.

3) CASE $H_1[P] \leq A \leq H_0[P]$

Now that we have shown the two extreme cases $a > H_0[p]$ and $a < H_1[p]$, let's focus our attention on the most interesting case: i.e. $H_0[p] \leq a \leq H_1[p]$. As previously mentioned, we are interested in finding the value $q = q^*$ for which $-M\frac{1-q}{q\log_D e}(a - H_q[p])$ is minimized. Taking the derivative, one gets:

$$\begin{aligned}\frac{d}{dq}\left(-M\frac{1-q}{q\log_D e}(a - H_q[p])\right) \\ = -\frac{M}{\log_D e}\left(\frac{1}{q(1-q)}D_{KL}[p^{(q)}||p] - \frac{1}{q^2}(a - H_q[p])\right),\end{aligned} \tag{31}$$

where $D_{KL}[p^{(q)}||p] = \sum_i p_i^{(q)} \log_D \frac{p_i^{(q)}}{p_i}$ is the Kullback-Leibler divergence between the escort of p and p itself. The minimum is then found by setting the derivative to zero, leading to the condition:

$$\begin{aligned} a - H_{q^*}[p] &= \frac{q^*}{1-q^*}D_{KL}[p^{(q^*)}||p] \\ &= \frac{q^*}{1-q^*}\sum_{i=1}^{|\Sigma|} p_i^{(q^*)} \log_D \frac{p_i^{(q^*)}}{p_i} \\ &= \frac{q^*}{1-q^*}\left[\sum_{i=1}^{|\Sigma|}\left(\frac{p_i^{q^*}}{\sum_{j=1}^{|\Sigma|}p_j^{q^*}}\log_D p_i^{q^*-1}\right)\right. \\ &\quad \left. - \sum_{i=1}^{|\Sigma|}\left(\frac{p_i^{q^*}}{\sum_{j=1}^{|\Sigma|}p_j^{q^*}}\log_D \sum_{j=1}^{|\Sigma|}p_j^{q^*}\right)\right] \\ &= q^*(H_1[p^{(q^*)}||p] - H_{q^*}[p]). \end{aligned} \tag{32}$$

Moreover, it is useful to write the Shannon entropy of the escort $p^{(q)}$:

$$\begin{aligned} H_1[p^{(q)}] &= -\sum_{i=1}^{|\Sigma|}\frac{p_i^q}{\sum_{j=1}^{|\Sigma|}p_j^q}\log_D\frac{p_i^q}{\sum_{j=1}^{|\Sigma|}p_j^q} \\ &= qH_1[p^{(q)}||p] + (1-q)H_q[p] \end{aligned} \tag{33}$$

By plugging Eqn. (33) into Eqn. (32), one gets that the value q^* which sets the derivative to 0 (i.e. minimizes the upper bound in the r.h.s. of Eqn. (28)) satisfies:

$$H_1[p^{(q^*)}] = a. \tag{34}$$

This equation relates the threshold a to the encoding strategy driven by p^{q^*}. In particular, such relation unveils that, if $a \in [H_1[p], H_0[p]]$, the optimal encoding strategy is the one that takes $q = q^*$ by solving Eqn. (34), and then uses that q^* into the escort distribution of Eqn. (7). This means that even if setting $q = 1$ guarantees the smallest linear average, such a choice would lead to larger fluctuations in the codewords' length, and thus to a larger probability of exceeding the threshold a. Instead, by setting $q = q^*$, these fluctuations are reduced and, in turn, the probability of exceeding the threshold a in the codewords' length is significantly decreased.

Summarizing our contributions in this section, we note that we have justified the use of the exponential average by the necessity of not exceeding a certain threshold in the length of the encoded string. In particular, given the value of the threshold a as an input, the procedure has three steps:

1) Estimate the probability distribution p of the input symbols.
2) Find q^* by solving Eq. (34).
3) Encode the input data with AC_{q^*}.

Such procedure guarantees that, if $a > H_1[p]$, it is possible to reduce the number of codewords exceeding the threshold with the use of the described AC_q model, which reaches the Rényi entropy bound with an error of at most 2 bits.

In the following paragraph, we will show a couple of examples over real and simulated data on how to infer the proper q^*, and how much this choice impacts the fraction of strings exceeding the threshold.

B. EXAMPLE

Throughout this section, we will apply our procedure to both the usual Wikipedia dataset and simulated strings composed by i.i.d. symbols. We will generate the latter according to the probability $p = (p_1, \ldots, p_{27})$ extracted from the Wikipedia dataset (see Figure 2 for a visual reference). In order to understand which is the range of interest for the threshold a, we have evaluated that $H_0[p] \approx 4.75$ and $H_1[p] \approx 4.12$. For this reason, we will consider a threshold $a \in [4.12, 4.75]$. Figure 8 shows both the value q^* for different values of a, evaluated as the solution of Eqn. (34), and the corresponding upper bound (UB) of the probability of exceeding the threshold, evaluated as:

$$\mathrm{UB} = e^{\left(-M \frac{1-q^*}{q^* \log_D e}(a - H_{q^*}[p])\right)}, \tag{35}$$

where we set $M = 20$. For $a \leq H_1[p]$, the upper bound UB is equal to 1, thus it gives no information on the probability of exceeding the threshold. Instead, when a increases, UB gets smaller until, for $a \geq H_0[p]$, it reaches 0 (and so does q^*), meaning that if the threshold is bigger than $H_0[p]$, by encoding with escort distribution of order 0 it becomes impossible to exceed the threshold. This agrees with our previous analysis.

So, we expect that, by applying AC_{q^*} to both Wikipedia and simulated data, the fraction of strings that exceed the threshold $M \cdot a$ is smaller than the one obtained by using the classic arithmetic coder, i.e. AC_1. Figure 9 shows, as a function of a, the fraction of strings of length $M = 20$ exceeding the threshold $M \cdot a$ when AC_{q^*} and AC_1 are applied, over Wikipedia and simulated data (i.i.d. symbols). It can be noted that, by generalizing the encoding procedure, the number of codewords exceeding the threshold can be decreased significantly, especially for "large" a. Such a drop is more pronounced in the case of the Wikipedia data. The reason is that, since there is an abundance of "rare" strings in the real data (as we already discussed), the encoding strategy with escort distribution, which penalizes frequent symbols in favor of rare ones, is more efficient than it is for truly i.i.d. symbols.

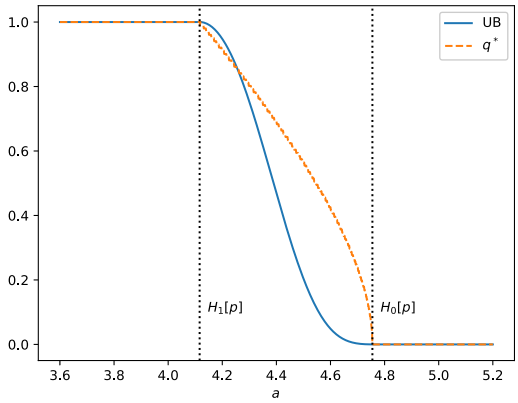

FIGURE 8. Dashed orange line: q^* as solution of Eqn. (34). Solid blue line: upper bound of the probability of exceeding the threshold. Black dotted vertical lines: $H_1[p]$ and $H_0[p]$. As the threshold a increases, the probability of exceeding it decreases until it reaches 0 for $a = H_0[p]$.

Moreover, we also want to explain the presence of the spike, followed by a short plateau, in the fraction of strings exceeding the threshold shown in the bottom panel of Figure 9, occurring for $a \gtrsim H_0[p]$. It is caused by the intrinsic 2-bits error of AC_{q^*} procedure (see Eqn. (17)). In fact, if $a = H_0[p] \approx 4.75$, then $M \cdot a \approx 95$. If we could exactly reach the desired symbols' length of Eqn. (7) with $q^* = 0$, we would never exceed the threshold. But AC_{q^*} carries an intrinsic error: the encoded strings' lengths are all 97 bits, in accordance with the predicted AC_{q^*} error. The fraction of strings exceeding the threshold is then 1 until a becomes such that $M \cdot a = 97$, i.e. $a = 4.85$. After such value, the exceeding fraction drops to 0. In other words, when the threshold is close to $H_0[p]$, even the very small error of the Arithmetic Coding procedure can lead to exceed it. Despite that, the ensemble of such cases is very small with respect to all the possibilities: for every $a \in (4.12, 4.75)$ the AC_{q^*} procedure performs better than the usual AC_1, both for real (correlated) symbols and simulated (independent) ones.

C. A NOTE ON THE ESTIMATION OF THE SOURCE PROBABILITY DISTRIBUTION

So far, we have considered that the probability p of the source generating i.i.d. symbols is known to the encoder. In reality,

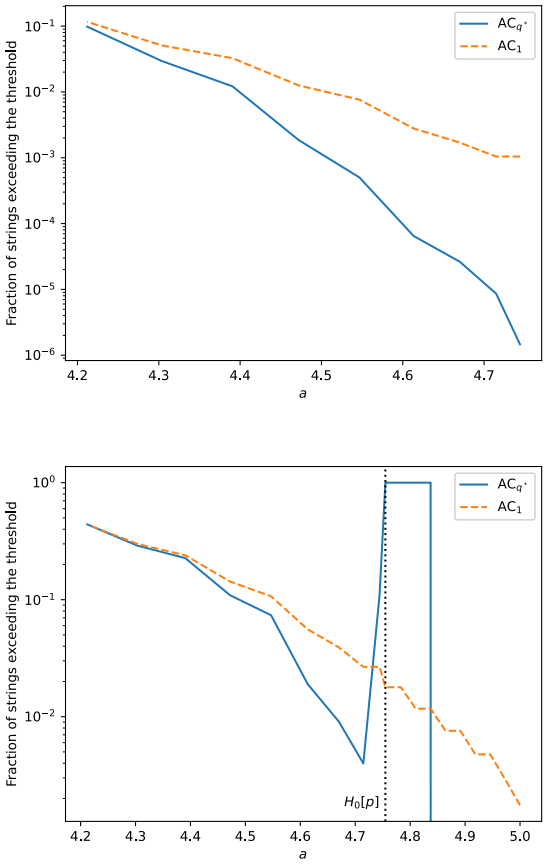

FIGURE 9. Fraction of strings exceeding the threshold for Wikipedia data (top panel) and for the simulated i.i.d. symbols (bottom panel). The orange dashed line is obtained with the classic arithmetic coder $\mathbf{AC_1}$, while the solid blue line with $\mathbf{AC_{q^*}}$. In the second panel, the spike before the plateau comes from the fact that an error $2/M = 2/20 = 0.1$ is intrinsically made by the $\mathbf{AC_q}$ procedure. In particular, recalling that for $a \geq H_0[p]$ we have that $q^* = 0$ (i.e. all the strings are encoded as if the source distribution is uniform), the plateau occurs when $a \geq H_0[p]$ and persists as long as $a < H_0[p] + 2/M$. In such range of values, where ideally we would like to encode as if the source distribution is uniform, the error carried by $\mathbf{AC_q}$ is enough to make all the encoded strings exceed the threshold, which is very close to the Shannon entropy $H_0[p]$. Instead, when $a > H_0[p] + 2/M$, the threshold is less stringent and the $\mathbf{AC_q}$ error does not make the codewords exceed a.

this could not be the case and a measure of error is needed if the probability $r = \{r_1, \ldots, r_N\}$ is used to encode symbols generated by the probability p. In the classical case, this is a well known problem. Assuming that it is possible to achieve the best encoding length which minimize the average length $L(0)$, i.e. $\ell_D(i) = -\log p_i$, then if the probability r is practically used to encode symbols generated according to p, the average codewords length is simply given by

$$H_1[p||r] = -\sum_{i=1}^{N} p_i \log_D r_i. \tag{36}$$

$H_1[p||r]$ is called cross-entropy. From this, it is possible to define the number of bits that are wasted by encoding according to r as the difference between the cross-entropy (i.e. the actual average length) and the Shannon entropy (i.e. the lowest possible average length), thus getting the

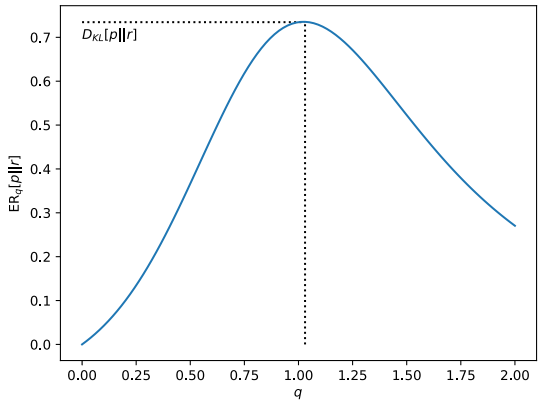

FIGURE 10. Error $\mathbf{ER_q[p||r]}$ for two instances of p and r: $p_i \propto i^{-1}$ and $r_i \propto i^{-2}$. The horizontal dashed black line corresponds to $D_{KL}[p||r]$, which corresponds to $\mathbf{ER_1[p||r]}$.

Kullback-Leibler divergence:

$$D_{KL}[p||r] = H_1[p||r] - H_1[p] = \sum_{i=1}^{N} p_i \log \frac{p_i}{r_i}. \tag{37}$$

Following the same path, we would like to provide a measure in the case of an exponential average. While Rényi himself defined a generalized D_{KL} [26], further analyzed in [27] and [28], and different definitions of a generalized cross-entropy exist [29], we would like to define such quantities in the framework of data compression. In particular, the exponential average codeword length when r is used to perform the compression is given by:

$$\begin{aligned} H_q[p||r] &= \frac{1}{t} \log_D \sum_{i=1}^{N} p_i D^{-t \log_D(r_i^{(q)})} \\ &= \frac{q}{1-q} \log_D \sum_{i=1}^{N} p_i r_i^{q-1} + (1-q) H_q[r], \end{aligned} \tag{38}$$

where $r^{(q)}$ is the escort distribution of r, $q = 1/(1+t)$ and $H_q[r]$ is the Rényi entropy of the distribution r. From this definition, it is possible to write a function for the error of encoding with distribution r instead of the true p, as the difference between the actual exponential average length $H_q[p||r]$, and the lowest possible exponential average length $H_q[p]$, that would be obtained by the exact guessing of p, i.e. with $r = p$:

$$\begin{aligned} \mathrm{ER}_q[p||r] &= H_q[p||r] - H_q[p] \\ &= \frac{q}{1-q} \log_D \sum_{i=1}^{N} p_i r_i^{q-1} \\ &\quad + (1-q) H_q[r] - H_q[p]. \end{aligned} \tag{39}$$

It is easy to see that $\mathrm{ER}_q[p||p] = 0 \ \forall q > 0$ and that $\lim_{q \to 1} \mathrm{ER}_q[p||r] = D_{KL}[p||r]$. Figure 10 shows the error

function for varying q, with given p and r such that $p_i \propto i^{-1}$ and $r_i \propto i^{-2}$.

To our knowledge, despite the different definitions of generalized divergences and cross-entropies in the literature, the quantity ER_q has not been defined. Yet, it has a direct interpretation and provides a measure of how a wrong estimate of the probability p propagates on the exponential average codeword length $L(t)$.

V. CONCLUSION

In this article, we have provided an operational scheme to encode sequences of symbols in order to minimize the exponential average codeword length. Our algorithm leads to an exponential average length per symbol that is arbitrarily close to the Rényi entropy of the source distribution. Moreover, we have detailed a possible application of the exponential average, based on its connection with the cumulant generating function of the source's probability distribution. Namely, if the encoder's priority is to minimize the risk of exceeding a certain codewords' threshold length, minimizing the exponential average is a better solution than minimizing the linear average. Even if all our theoretical considerations are based on the hypothesis that the symbols are i.i.d. distributed and that the encoder knows the true source distribution p, we have both shown empirically that AC_q is advantageous also in the presence of correlations and provided a measure of the error when the encoder guesses the incorrect source distribution.

Our study offers promising insights into nonlinear data compression that may inspire some future research activities. First of all, we notice that all our theoretical results and experimental achievements could benefit from the use of more recent statistical compressors (i.e., ANS [25]) in place of AC, whose simplicity has been exploited in this paper just for clarity of explanation. Moreover, it would be interesting to generalize our approach to lossy compression techniques, extending it to multimedia files such as images and videos. We also foresee further applications of our study on nonlinear compression techniques to meet the evolving demands of modern data processing and communication systems. Finally, on the technical side, it would be interesting to explain quantitatively the phenomenon, highlighted in Section III, about how correlations among input symbols lead to an optimal q different from q_t; and evaluate the expected error of guessing the source distribution given a certain (''small'') training dataset.

REFERENCES

[1] L. C. Meiser, B. H. Nguyen, Y.-J. Chen, J. Nivala, K. Strauss, L. Ceze, and R. N. Grass, ''Synthetic DNA applications in information technology,'' *Nature Commun.*, vol. 13, no. 1, p. 352, 2022.

[2] P. Mishra, C. Bhaya, A. K. Pal, and A. K. Singh, ''Compressed DNA coding using minimum variance Huffman tree,'' *IEEE Commun. Lett.*, vol. 24, no. 8, pp. 1602–1606, Aug. 2020.

[3] F. Jelinek, ''Buffer overflow in variable length coding of fixed rate sources,'' *IEEE Trans. Inf. Theory*, vol. IT-14, no. 3, pp. 490–501, May 1968.

[4] P. Humblet, ''Generalization of Huffman coding to minimize the probability of buffer overflow (corresp.),'' *IEEE Trans. Inf. Theory*, vol. IT-27, no. 2, pp. 230–232, Mar. 1981.

[5] L. David and A. Zaman, ''Simulating iridium satellite coverage for CubeSats in low earth orbit,'' in *Proc. AIAA/USU Conf. Small Satell., Mission Lessons*, 2018, Paper SSC18-PI-06. [Online]. Available: https://digitalcommons.usu.edu/smallsat/2018/all2018/336/

[6] M. B. Baer, ''Optimal prefix codes for infinite alphabets with non-linear costs,'' *IEEE Trans. Inf. Theory*, vol. 54, no. 3, pp. 1273–1286, Mar. 2008.

[7] L. L. Campbell, ''A coding theorem and Rényi's entropy,'' *Inf. Control*, vol. 8, no. 4, pp. 423–429, 1965.

[8] C. E. Shannon, ''A mathematical theory of communication,'' *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, Jul. 1948.

[9] A. N. Kolmogorov and G. Castelnuovo, *Sur La Notion De La Moyenne*. G. Bardi, Ed. Rome: Accademia dei Lincei, 1930.

[10] M. Nagumo, ''Über eine klasse der mittelwerte,'' *Jpn. J. Math., Trans. Abstr.*, vol. 7, pp. 71–79, Jan. 1930.

[11] J. Aczél and Z. Daróczy, *On Measures of Information and Their Characterizations*. New York, NY, USA: Academic, 1975.

[12] L. Campbell, ''Definition of entropy by means of a coding problem,'' *Zeitschrift Wahrscheinlichkeitstheorie Verwandte Gebiete*, vol. 6, no. 2, pp. 113–118, 1966.

[13] G. H. Hardy, J. E. Littlewood, and G. Pólya, *Inequalities*. Cambridge, U.K.: Cambridge Univ. Press, 1952.

[14] P. A. Morales, J. Korbel, and F. E. Rosas, ''Thermodynamics of exponential Kolmogorov–Nagumo averages,'' *New J. Phys.*, vol. 25, Jul. 2023, Art. no. 073011.

[15] C. Tsallis, *Introduction to Nonextensive Statistical Mechanics: Approaching a Complex World*. Springer, 2009.

[16] C. Beck and F. Schögl, *Thermodynamics of Chaotic Systems: An Introduction* (Cambridge Nonlinear Science Series). Cambridge, U.K.: Cambridge Univ. Press, 1993.

[17] A. Somazzi and D. Garlaschelli, ''Learn your entropy from informative data: An axiom ensuring the consistent identification of generalized entropies,'' 2023, *arXiv:2301.05660*.

[18] A. Khinchin, *Mathematical Foundation of Information Theory*. New York, NY, USA: Dover, 1957.

[19] P. Jizba and J. Korbel, ''When Shannon and Khinchin meet shore and Johnson: Equivalence of information theory and statistical inference axiomatics,'' *Phys. Rev. E, Stat. Phys. Plasmas Fluids Relat. Interdiscip. Top.*, vol. 101, no. 4, 2020, Art. no. 042126.

[20] N. Merhav, ''Universal coding with minimum probability of codeword length overflow,'' *IEEE Trans. Inf. Theory*, vol. 37, no. 3, pp. 556–563, Jun. 1991.

[21] A. C. Blumer and R. J. McEliece, ''The Rényi redundancy of generalized Huffman codes,'' *IEEE Trans. Inf. Theory*, vol. 34, no. 5, pp. 1242–1249, Sep. 1988.

[22] J.-F. Bercher, ''Source coding with escort distributions and Rényi entropy bounds,'' *Phys. Lett. A*, vol. 373, no. 36, pp. 3235–3238, 2009.

[23] I. H. Witten, R. M. Neal, and J. G. Cleary, ''Arithmetic coding for data compression,'' *Commun. ACM*, vol. 30, no. 6, pp. 520–540, 1987.

[24] P. Ferragina, *Algorithm Engineering*. Cambridge, U.K.: Cambridge Univ. Press, 2023.

[25] A. Moffat and M. Petri, ''Large-alphabet semi-static entropy coding via asymmetric numeral systems,'' *ACM Trans. Inf. Syst.*, vol. 38, no. 4, pp. 1–33, 2020.

[26] A. Rényi, ''On measures of entropy and information,'' in *Proc. 4th Berkeley Symp. Math. Statist. Probab., Contrib. Theory Statist.*, vol. 1, 1961, pp. 547–562.

[27] T. Van Erven and P. Harremos, ''Rényi divergence and Kullback–Leibler divergence,'' *IEEE Trans. Inf. Theory*, vol. 60, no. 7, pp. 3797–3820, Jul. 2014.

[28] Z. Rached, F. Alajaji, and L. L. Campbell, ''Rényi's divergence and entropy rates for finite alphabet Markov sources,'' *IEEE Trans. Inf. Theory*, vol. 47, no. 4, pp. 1553–1561, May 2001.

[29] F. C. Thierrin, F. Alajaji, and T. Linder, ''Rényi cross-entropy measures for common distributions and processes with memory,'' *Entropy*, vol. 24, no. 10, p. 1417, 2022.

ANDREA SOMAZZI was born in Wichita Falls, TX, USA, in 1994. He received the B.Sc. degree in physical engineering and the M.Sc. degree in physics of complex systems from Politecnico di Torino, the M1 degree in physics of complex systems from Université Paris Diderot, and the Ph.D. degree in data science from Scuola Normale Superiore di Pisa with thesis "Challenges in Data Science for Complex Systems." He is currently a Postdoctoral Research Fellow with the IMT School for Advanced Studies Lucca. His research interests include complex systems, statistical inference, maximum entropy models, and information theory.

PAOLO FERRAGINA received the Ph.D. degree in computer science from the University of Pisa, in 1996. From 1997 to 1998, he was a Postdoctoral Researcher with Max-Planck Institut für Informatik, Saabrücken, Germany. He is currently a Professor of algorithms with the University of Pisa, where he founded and leads the Acube Laboratory, whose research activities regard the design of algorithms for big data, mainly in the form of texts and graphs, in collaboration with companies worldwide, such as Bloomberg, European Broadcasting Union (EBU), Google, Tiscali, and Yahoo!. He coauthored more than 180 (refereed) publications, some books, and chapters, achieving an H-index of 34 on Scopus and more than 10 000 citations on Google Scholar. His research results got five U.S./ITA Patents and some international awards, such as the 1995 Best Land Transportation Paper Award from the IEEE Vehicular Technology Society, the 1997 Best Ph.D. Thesis in Theoretical Computer Science from the Italian Chapter of the EATCS, the 1997 Philip Morris Award on Science and Technology, one Yahoo! Research Faculty Award, three Google Research Awards, and the 2022 ACM Paris Kanellakis Award. He is serving on the editorial board for the *Journal of Graph Algorithms and Applications* (JGAA). He is an Area Editor of *Encyclopedias of Algorithms* (Springer) and *Big Data Technologies* (Springer).

DIEGO GARLASCHELLI received the master's degree in theoretical physics from the University of Rome III, in 2001, and the Ph.D. degree in physics from the University of Siena, in 2005. He was a Postdoctoral Researcher with The Australian National University, Canberra, Australia; the University of Siena, Italy; the University of Oxford, U.K., and the Sant'Anna School for Advanced Studies, Pisa, Italy. He is currently a Professor of theoretical physics with the IMT School of Advanced Studies Lucca, Italy, where he directs the Networks Research Unit and the Lorentz Institute for Theoretical Physics, Leiden University, The Netherlands, and leads the Econophysics and Network Theory Group. He teaches courses in complex networks, econophysics, and complex systems. He has given more than 70 invited talks at international conferences, workshops, and scientific schools. He is the author of more than 100 publications in peer-reviewed international journals and peer-reviewed book chapters, and one coauthored monograph. His research interests are strongly interdisciplinary and include network theory, statistical physics, random graphs, information theory, financial complexity, social dynamics, ecological networks, and biological systems.

Open Access funding provided by 'Scuola IMT Alti Studi Lucca' within the CRUI CARE Agreement