Noname manuscript No.
(will be inserted by the editor)

Generalized Elias Schemes for Efficient Harvesting of Truly Random Bits

Riccardo Bernardini · **Roberto Rinaldo**

the date of receipt and acceptance should be inserted later

Abstract The problem of generating a sequence of true random bits (suitable for cryptographic applications) from random discrete or analog sources is considered. A generalized version, including Vector Quantization, of the classical approach by Elias for the generation of truly random bits is introduced, and its performance is analyzed, both in the finite case and asymptotically. The theory allows us to provide an alternative proof of the optimality of the original Elias' scheme. We also consider the problem of deriving random bits from measurements of a Poisson process and from vectors of iid Gaussian variables. The comparison with the scheme of Elias, applied to geometric-like non binary vectors, originally based on the iso-probability property of permutations of iid variables, confirms the potential of the generalized scheme proposed in our work.

Keywords Random number generation · Random Number Conditioning · Vector Quantization · Geometric random variables · Gaussian random variables.

1 Introduction

The problem of true random number generation dates back to von Neumann [1] who considered the problem of simulating an unbiased coin by using a biased coin with unknown probability. Denoting with T and H the tail and head outcomes, respectively, he observed that considering two consecutive independent coin tosses, the events TH and HT are exactly equiprobable. Thus, mapping $TH \to 0$, $HT \to 1$, while discarding the events TT, HH, generates a sequence of truly random bits even if the original coin is biased.

More efficient algorithms for generating random bits from a biased coin were proposed by various authors [2–5]. See [6] for a more comprehensive bibliography, where the problem to generate random bits from a correlated source is considered. Elias [3] was the first to devise an optimal procedure in terms of information efficiency, namely, such that the expected number of unbiased random bits generated per coin toss is asymptotically equal to the entropy of the biased coin. Starting from a source that produces bit vectors $x = [X_0, \ldots, X_{L-1}]$ of binary independent random variables $X_i \in \{0,1\}$, $\mathrm{P}[X_i = 0] = q$, the procedure partitions the range of x into classes of equiprobable outcomes consisting of all the permutations (with repetition) of the bit string with a given Hamming weight. Due to independence, the elements of each class are therefore equiprobable. The elements of a class are then mapped into binary strings, by means of a procedure called a *conditioner*, in such a way that, due to the iso-probability property, the corresponding bits will be independent and equiprobable. This approach is easily extended to non-binary alphabets.

In this paper, we extend the original approach of Elias by considering a Generalized Elias Scheme (GES) that partitions the range of x, in the non-binary or even continuous case, into generic classes of equiprobable vectors, not just permutations as in the original procedure. After formalizing the definition of a block-based admissible conditioner in Section 4, we introduce the GES in Section 5.1. We derive lower and upper bounds for the efficiency of this generalized approach, and consider its asymptotic behavior for increasing block-length. One of the results that we derive from the generalized approach is an alternative proof of the original Elias' scheme optimality (Section 6.1), and a detailed analysis of what happens when processing a finite number of blocks. Section 6.2 analyzes the case of non-binary iid

A reduced 4-page version of this manuscript, with only the claims of the major results, has been submitted for presentation at ICASSP 2014.

Riccardo Bernardini, Roberto Rinaldo
Università di Udine – DPIA
Via delle Scienze 208, 33100, Udine – Italy
E-mail: riccardo.bernardini@uniud.it
E-mail: rinaldo@uniud.it

random variables with a geometric-like distribution, as they may arise from measurements of physical processes modeled as Poisson processes. These vectors can be used as the input to a specialized conditioner, whose performance is analyzed in Section 7 and compared to the original Elias' scheme, based on permutations of non-binary vectors. Section 6.3 considers the case of quantized vectors of iid Gaussian random vectors. The more technical proofs are reported in Appendix A.

2 Informal Overview

In order to give an intuitive idea of the results described in this paper, in this section we give an informal overview of the considered scenario and of the proposed solution.

2.1 Application scenario

Suppose one has a random process X_n made of iid random variables assuming values in an alphabet $\mathscr{A}$. Differently from most other works on conditioning, we do not require $\mathscr{A}$ discrete, allowing also for continuous alphabets like $\mathbb{R}$ or $\mathbb{R}_{>0}$. We want to process X_n in order to produce a sequence of bits b_k iid and unbiased, that is, an Unbiased Bernoulli Process (UBP). Actually, we will require something a bit stronger than just b_k being an UBP (see Section 4.1), but for the moment the UBP constraint will suffice.

A natural performance metric for a conditioner is its *rate* defined as the average number of output bits produced per input value (see Definition 2). It is intuitive that if $\mathscr{A}$ is discrete and the output process is an UBP, the entropy $H(X_n)$ of a symbol is an upper bound to the conditioner rate. It is not clear, however, what is the upper bound when $\mathscr{A}$ is not discrete. We give an answer to this in Theorem 1.

A key requirement for practical applicability is the *parametric robustness* of any conditioning scheme. It is reasonable to suppose that in most application we know the "type" of the random process, but not its exact "intensity." For example, if $\mathscr{A} = \{0,1\}$ (binary source), the probability $p_1 := P[X_n = 1]$ of getting "1" could not be exactly known; if X_n is the inter-arrival time of a Poisson process, we know that X_n is exponential, but its mean (related to the intensity of the Poisson process) could be not exactly known; finally, if X_n are obtained by sampling some Gaussian random noise, we can assume that X_n is zero-mean Gaussian, but with the variance not exactly known. It is clear that the practical usability of any conditioner that requires a very precise intensity of to input process to guarantee an UBP output is somehow doubtful. It is instead acceptable that the output rate changes with the intensity of the input process. For example, the well known von Neumann [1] and Elias' schemes [3] for binary sources always produce an UBP output, whose rate depends on p_1.

2.2 The proposed solution

Fig. 1a shows the general structure of the conditioner proposed in this paper. The conditioner in Fig. 1a works like a Vector Quantizer (VQ): first the input samples are collected in blocks of size L, then the current block is mapped to a discrete set of symbols $\mathscr{Q}_L$, finally, each symbol is mapped to an output bitstring. As well known, the map, from blocks to symbols, partitions the set of blocks $\mathscr{A}^L$ into a finite number of *quantization regions*. Every symbol in $\mathscr{Q}_L$ can be considered as a *label* for the corresponding quantization region.

Despite the similarity of Fig. 1a with a lossy coding scheme based on a VQ, there are some very important differences. For example, in this case we have no interest in distortion, but only on the statistical properties of the output process. Actually, it will be seen that the VQ used for conditioning is usually a bad VQ from a lossy coding viewpoint. Moreover, the indexing block in a coding context assigns to every symbol a unique bitstring, since we want to be able to recover a good approximation of the quantized values; instead, it will be seen that the VQ used as a conditioner in this paper often maps different symbols into the same bitstring. Moreover, some symbol can be mapped to an empty bitstring (as it happens in the von Neumann [1] and Elias' schemes [3]), while no VQ for lossy coding would do that.

Note that the scheme Fig. 1a is very general since it can emulate even Elias-like schemes. Indeed, if $\mathscr{A}$ is finite, one can choose $\mathscr{Q}_L = \mathscr{A}^L$ and use the "trivial quantizer" represented by the identity map. By using as indexer the map used in Elias' scheme, one can see that Fig. 1a becomes equivalent to Elias' solution.

It is worth to anticipate that the indexer in Fig. 1a is the key part of the scheme and it is worth to describe briefly how it is designed. By reading the original Elias' work [3] one can see that a key step is the partition of the set of all blocks $\mathscr{A}^L$ into subsets that enjoy a *iso-probability* condition, that is, two symbol strings belonging to the same subset must have the same probability. By exploiting this iso-probability condition, it is easy to assign bitstrings to blocks in a way that grants the UBP property of the output process. Elias' solution puts in the same subset blocks which are permutations of each other. This, together with the iid property of the input process, grants for the iso-probability condition.

The approach followed in this work is similar, but using quantizer symbols in place of input blocks. More precisely, the quantizer symbols are collected in sets that enjoy the same iso-probability condition, that is, two symbols belonging to the same set must have the same probability. By exploiting the iso-probability condition, the assignment

of output bit-strings to quantizer symbols can be done as it is done in the Elias' scheme [3]. The non-trivial part is to choose the quantizer regions so that the iso-probability condition holds independently on the input process intensity (in order to have parametric robustness).

3 Notation

Sets If $\mathscr{A}$ is any set, its cardinality will be denoted as $|\mathscr{A}|$. In this paper we take the convention that the set of natural numbers $\mathbb{N}$ includes zero. We will use J_N to denote the subset of the first N natural numbers, that is, $J_N := \{0,\dots,N-1\}$. The conventions $J_\infty = \mathbb{N}$ and $J_0 = \emptyset$ will be useful for notation uniformity.

Sequences If $\mathscr{A}$ is any set, a *sequence X of length N with elements in $\mathscr{A}$* is defined as a function $X : J_N \to \mathscr{A}$, i.e., for $n \in J_N$, $X(n) \in \mathscr{A}$. In this context N can be finite or not. We will use the simpler notation $X_n \equiv X(n)$ to denote the n-th element of sequence X. The length of X will be denoted as $|X|$. We will denote with $\mathscr{A}^*$ the set of all finite sequences (even the empty one) with elements in $\mathscr{A}$. If $a,b \in \mathscr{A}^*$, we will denote with $a \cdot b$ their concatenation. As usual, vectors will be denoted with bold letters (e.g., $\mathbf{k}$), while the n-th entry of a vector will be denoted as non-bold with n as subscript (e.g., k_n).

We will use the notation $\mathbf{J_N}^L$ to identify the set of all the possible $L-$length vectors with entries in J_N in increasing order, that is,

$$\mathbf{J_N}^L := \{\mathbf{k} = [k_1,k_2,\dots,k_L] \in J_N^L : k_1 < k_2 < \cdots < k_L\} \quad (1)$$

For example, if $N=3$, $L=2$, then $\mathbf{J_N}^L = \{[0,1],[0,2],[1,2]\}$.

We will sometimes need to consider a subsequence of the original sequence. Let X be a sequence of length N and let $\mathbf{k} \in \mathbf{J_N}^L$, be a vector of increasing entries. We will denote with $X_\mathbf{k}$ the sequence obtained by taking the elements of X corresponding to the indexes in $\mathbf{k}$. More formally, $X_\mathbf{k} : J_L \to \mathscr{A}$ is defined as $[X_\mathbf{k}]_n = X_{k_n}$ for all $n \in J_L$ This operation will be called *subsampling* and $X_\mathbf{k}$ a *subsampled version* of X. We will use the colon notation $X_{n:m}$ as a shorthand for $X_{[n,n+1,\dots,m]}$.

Signatures Let $\mathscr{A}$ be any set. If $a \in \mathscr{A}$ and $x \in \mathscr{A}^L$, we will denote with $\#_a(x)$ the number of times that a is present in x. If $x,y \in \mathscr{A}^L$ are such that $\#_a(x) = \#_a(y)$ for every $a \in \mathscr{A}$, we will say that x and y *have the same signature*. It is clear that x and y have the same signature if and only one is a permutation of the other.

4 Blockwise Conditioners

In this section we introduce some notions and properties related with the problem of *conditioning* a sequence of random variables in order to obtain a sequence of truly random bits. More precisely, the problem of conditioning can be formulated as follows.

Problem 1 Let a sequence of iid random variables $\{X_n\}_{n\in\mathbb{N}}$, assuming values in the alphabet (finite or not) $\mathscr{A}$, be given. Process $\{X_n\}_{n\in\mathbb{N}}$ with a deterministic function (a *conditioner*) in order to obtain an UBP $\{b_n\}_{n\in\mathbb{N}}$, that is, a process where b_n are iid and $P[b_n=0] = P[b_n=1] = 1/2$.

Remark 1 In Section 4.1 we are going to tighten Problem 1 by requiring a condition stronger than being only UBP.

In this paper we consider only *blockwise* conditioners, that is, conditioners that partition the input sequences into non-overlapping blocks of equal size L and process each block separately. More precisely, a blockwise conditioner with block size L is represented by a function $\mathscr{C} : \mathscr{A}^L \to \{0,1\}^*$ that maps blocks of L input values into sequences of bits of finite length (even zero). The n-th block will be denoted as

$$\mathscr{X}_n := X_{nL:(n+1)L-1} = [X_{nL}, X_{nL+1}, \dots, X_{(n+1)L-1}] \quad (2)$$

The output process b can be defined as the concatenation of the strings obtained by applying the conditioner to every block, that is

$$b = \mathscr{C}(\mathscr{X}_0)\cdot\mathscr{C}(\mathscr{X}_1)\cdots\mathscr{C}(\mathscr{X}_n)\cdots \quad (3)$$

It will be convenient to have symbols to denote the bit-string obtained after processing a finite number of blocks. We will denote with S_N, $N \in \mathbb{N}$ the bit-string obtained after processing blocks $\mathscr{X}_0$, …, $\mathscr{X}_N$, that is, $S_N = \mathscr{C}(\mathscr{X}_0)\cdot\mathscr{C}(\mathscr{X}_1)\cdots\mathscr{C}(\mathscr{X}_N)$. More precisely, S_N is recursively defined as

$$S_N = \begin{cases} \mathscr{C}(\mathscr{X}_0) & \text{if } N = 0 \\ S_{N-1}\cdot\mathscr{C}(\mathscr{X}_N) & \text{if } N > 0 \end{cases} \quad (4)$$

We will denote with $\mu_n := |\mathscr{C}(\mathscr{X}_n)|$ the number of bits obtained by processing $\mathscr{X}_n$ and with $\ell_N := |S_N| = \sum_{n=0}^N \mu_n$ the length of bit-string S_N (that is, the number of bits generated after processing blocks $\mathscr{X}_0$, …, $\mathscr{X}_N$). Note that both μ_n and ℓ_n are random variables. If $\mathbf{k} \in \mathbb{N}^L$ and $\ell_N > k_L$, we will denote $[S_N]_\mathbf{k}$ (the subsampled version of S_N, i.e., the $L-$length bit substring corresponding to indexes in $\mathbf{k}$, see Section 3), with the more compact notation $S_{N,\mathbf{k}}$. If $\ell_N \le k_L$, S_N is not long enough and $S_{N,\mathbf{k}}$ is not defined.

Remark 2 Note that $\mathrm{Im}(\mu_0)$ is the set of the lengths of the bit-strings that can be generated by processing an input block; a similar interpretation holds for $\mathrm{Im}(\ell_N)$.

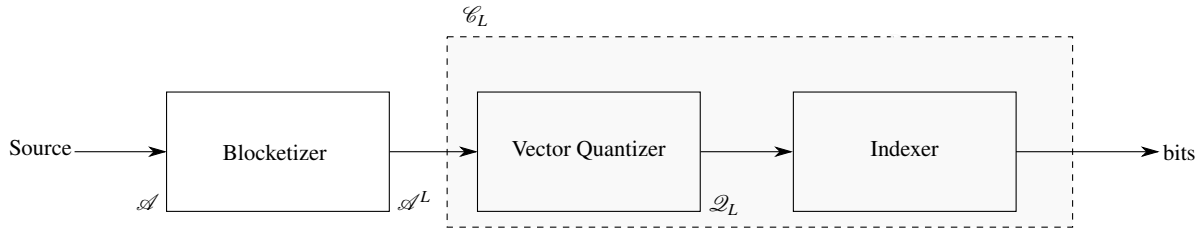

Fig. 1 The proposed scheme for random bit generation

4.1 Admissible Conditioners

So far we did not put any constraint on the statistical properties of the output process b. If the bit sequence is used in a cryptographic application, the requirement is that an opponent cannot get any information about bits b_n. Usually this is expressed by requiring that the output process b is a UBP. However, this could not be enough for cryptographic applications.

An example will clarify this point. Suppose that the input process X has alphabet $\{\alpha,\beta,\gamma\}$ with $P[X_n=\alpha]=P[X_n=\beta]=1/4$ and $P[X_n=\gamma]=1/2$ and consider the conditioner with $L=1$ specified by the Huffman code mapping

$$\alpha\mapsto 00;\quad \beta\mapsto 01;\quad \gamma\mapsto 1. \tag{5}$$

It is easy to check that this scheme is UBP (see Proof A.1 in Appendix A.2). However, if the opponent knows (maybe via some side-channel) that the output string after processing the first symbol has two bits, it can deduce that the input symbol was α or β and this implies $S_{0,0}=0$. That is, from the knowledge that the conditioner processed a single input symbol and obtained two output bits, an opponent can deduce the value of the first bit, despite the fact that the "final" process b is a UBP. Many other examples of this type can be constructed as Huffmann codes of random variable with probabilities of the form $2^{-\ell}$.

In order to avoid that an opponent could gain any information about the generated bit-string even if he/she were able to measure the length of the generated bit-string, we are going to require a slightly stronger condition than the UBP property for an *admissible* conditioner.

Definition 1 Conditioner $\mathscr{C}:\mathscr{A}^L\to\{0,1\}^*$ will be said to be *admissible* if the following condition holds

$$\begin{gathered}\forall G\geq 1,\mathbf{k}\in\mathbb{N}^G,m\in\mathrm{Im}(\mu_0),m>k_G,\mathbf{b}\in\{0,1\}^G\\ P[S_{0,\mathbf{k}}=\mathbf{b}|\mu_0=m]=2^{-G}\end{gathered} \tag{6}$$

Informally, condition (6) requires that any G-length substring of the output *of the first block* $S_0=\mathscr{C}(\mathscr{X}_0)$ must be a random variable uniformly distributed over $\{0,1\}^G$ (constraint $m\in\mathrm{Im}(\mu_0)$ grants that the conditioning event $\mu_0=m$ has not zero probability; constraint $m>k_G$ is necessary in order to have $S_{0,\mathbf{k}}$ defined). The reason for asking for (6) is that in a real case an opponent could get an estimate of the length of the bit-string obtained after processing the first block μ_0 (e.g., by some side-channel attack that measures the time required to store the produced bits), even without knowing the actually generated bits. If condition (6) is not satisfied, the opponent could derive some information about the generated bits, reducing the security of the whole system.

Fact 1 *Condition (6) is* stronger *than requiring that the overall process (3) is a UBP. That is, if (6) is true, then b is a UBP; but the reverse implication does not hold: one can have a non-admissible conditioner that gives rise to a UBP.*

Proof The proof that (6) implies that b is a UBP is given in Appendix A.2, where it is also shown that knowledge of the length of the generated bit-string after processing N blocks does not give any information about the generated bits. Scheme (5) is an example of a conditioner that is UBP, but not admissible.

Condition (6) seems complex to verify. However, it is possible to give a simpler equivalent condition that uses only $\mathrm{Im}(\mathscr{C})$.

Property 1 A conditioner $\mathscr{C}:\mathscr{A}^L\to\{0,1\}^*$ is admissible *if and only if*

$$\forall\mathbf{b}\in\mathrm{Im}(\mathscr{C})\qquad P[\mathscr{C}(\mathscr{X}_0)=\mathbf{b}|\mu_0=|\mathbf{b}|]=2^{-|\mathbf{b}|} \tag{7}$$

In words, Property 1 says that, if the conditioner generates a bit-string $\mathbf{b}$ of length $|\mathbf{b}|$, i.e., $\mathbf{b}\in\mathrm{Im}(\mathscr{C})$, then all the $2^{|\mathbf{b}|}$ bit-strings with the same length are in $\mathrm{Im}(\mathscr{C})$ and equiprobable. One direction of the proof (i.e., that (6) implies (7)) is easy, since (7) is just a specialized version of (6). The other direction is more complex and it is given in Appendix A.3.

4.2 Conditioner rate

A parameter of interest is the *rate* of a conditioner, that is, the amount of random bits per input symbol that a conditioner can produce.

Definition 2 If $\mathscr{C}:\mathscr{A}^L\to\{0,1\}^*$ is a conditioner, its *rate* R (in bit per input symbol) is defined as

$$R=\frac{\mathbb{E}[\mu_0]}{L} \tag{8}$$

In the case of $\mathscr{A}$ finite, we expect that the rate of an admissible conditioner will not be larger than the entropy of the input symbol. Indeed, from Lemma 1 proved in Section 5 one can easily show that if $\mathscr{A}$ is discrete and X is iid, then

$$R \le H(X) - \frac{H(\mu_0)}{L} \tag{9}$$

5 VQ-Based Generalized Elias Schemes

So far we did not impose any special structure on the conditioner $\mathscr{C}$. Now we are going to suppose that it has the special structure shown in Fig. 1: every block is processed by a VQ $Q_L : \mathscr{A}^L \to \mathscr{Q}_L$, where $\mathscr{Q}_L$, the quantizer alphabet, is a finite set. We will denote with $V_n = Q_L(\mathscr{X}_n)$ the r.v. associated with the n-th VQ output. Note that if the alphabet $\mathscr{A}$ is finite, then one can choose $\mathscr{Q}_L = \mathscr{A}^L$ and use for Q_L the identity map. In this case we will say that the *trivial quantizer* is used.

Successively, the output of the Q_L is mapped to an output bit-string by an *indexer* $I_L : \mathscr{Q}_L \to \{0,1\}^*$. Note that one could merge the VQ with the indexer block, getting rid of the VQ output. The introduction of the VQ, however, makes the analysis easier.

The following lemma gives a very general bound about the rate of a VQ-based Generalized Elias Scheme (GES).

Theorem 1 *If X is a stationary process (not necessarily iid) and $\mathscr{C} : \mathscr{A}^L \to \{0,1\}^*$ is an* admissible *conditioner for X, the following inequality holds*

$$R = \frac{\mathbb{E}[\mu_0]}{L} \le \frac{H(V_0)}{L} - \frac{H(\mu_0)}{L} \tag{10}$$

Proof Since the conditioner is a deterministic function, we have $H(S_0) = H(\mathscr{C}(\mathscr{X}_0)) \le H(V_0)$. In order to obtain (10), observe that

$$\begin{aligned} H(V_0) &\ge H(S_0) \\ &= H(S_0\mu_0) \\ &= H(S_0|\mu_0) + H(\mu_0) \\ &= \sum_{k \in \mathrm{Im}(\ell_0)} H(S_0|\mu_0 = k)P[\mu_0 = k] + H(\mu_0) \\ &= \sum_{k \in \mathrm{Im}(\ell_0)} kP[\mu_0 = k] + H(\mu_0) \\ &= \mathbb{E}[\mu_0] + H(\mu_0) \end{aligned} \tag{11}$$

Equation (10) follows by dividing (11) by L.

Corollary 1 *If X is an iid process, $\mathscr{A}$ is discrete, the VQ is trivial and $\mathscr{C} : \mathscr{A}^L \to \{0,1\}^*$ is an admissible conditioner for X, then*

$$R \le H(X) - \frac{H(\mu_0)}{L} \tag{12}$$

5.1 Indexer Design

The indexer used in this paper is based on the same principle (*two-level partition*) used by Elias in his work [7]. In this section, we generalize the original technique by Elias in a way that can be reused for the scheme proposed in this paper. This will allow us to derive lower and upper bounds to the efficiency of a generic GES, making it possible to derive quantitative results for the proposed scheme, as well as derive alternative proofs of the performance of the original Elias' scheme. It is easier to describe the construction as a sequence of steps

1. *Partitioning* The first (and key) step in the construction of the indexer is an *iso-probability partitioning* of $\mathscr{Q}_L$. More precisely, $\mathscr{Q}_L$ is partitioned as
$$\mathscr{Q}_L = \bigcup_{k=1}^{\mathscr{P}_L} P_k, \qquad i \ne j \Rightarrow P_i \cap P_j = \emptyset \tag{13}$$
where $\mathscr{P}_L$ is the cardinality of the partition, and every set P_k satisfies the following *iso-probability condition*
$$x, y \in P_k \Rightarrow P[V_0 = x] = P[V_0 = y] \tag{14}$$
that is, all the symbols belonging to P_k must have the same probability. Let $\phi_k(x)$ a one-to-one enumeration map for the elements of P_k.
2. *Splitting P_k* The second step is also done off-line and it requires that every set P_k in (13) is partitioned as
$$P_k = \bigcup_j V_{k,j}, \qquad i \ne j \Rightarrow V_{k,i} \cap V_{k,j} = \emptyset, \tag{15}$$
where the cardinality of every $V_{k,j}$ is a power of two. The cardinalities of sets $V_{k,j}$ can be easily found by expressing $|P_k|$ in binary [7]
$$|P_k| = \sum_{j=1}^{C} 2^{b_{k,j}} \tag{16}$$
3. *Indexing* Let $\nu_{k,j} = \log_2 |V_{k,i}|$. To every element of $V_{k,i}$ one assigns a unique $\nu_{k,j}$-bit binary string corresponding to the binary expansion of a unique index $\phi_{k,i}(x)$.
4. *On-line procedure* The last step is the only one done on-line. In the on-line step the conditioner just receives a symbol $x \in \mathscr{Q}_L$, it finds the unique $V_{k,i}$ that includes x and produces the $\nu_{k,i}$-bit binary expansion of $\phi_{k,i}(x)$. If $|V_{k,i}| = 1$, then $\nu_{k,i} = 0$ and the empty bit-string is produced.

See Fig. 2 for an example of this two-level partition.

Example 1 For example, suppose $P_k = \{0, 1, \ldots, 25\}$. Since $|P_k| = 26 = 2^1 + 2^3 + 2^4$, P_k is partitioned in three sets: set $V_{k,1}$ contains the 2^4 elements $\{0, 1, \ldots, 15\}$, $V_{k,2}$ contains the 2^3 elements $\{16, 17, \ldots, 23\}$ and $V_{k,3}$ contains the 2^1

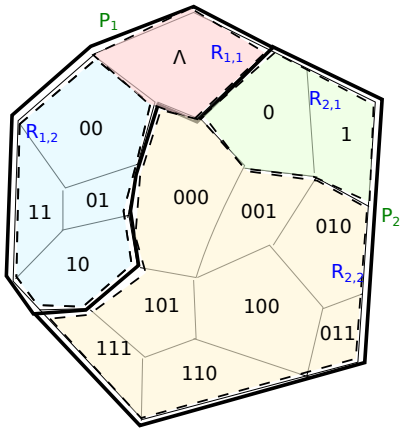

Fig. 2 Partitioning of a fictional VQ. The set of VQ regions has been partitioned into two subset P_1 and P_2 and each subset has been partitioned into subsets $R_{i,j}$ with $2^{n_{i,j}}$ elements. Every set $R_{i,j}$ is labelled with the corresponding bit-string. Note that $R_{1,1}$ has only one element and the corresponding bit-string is the empty string (shown as Λ).

elements $\{24,25\}$. In this example $\nu_{k,1}=4$, $\nu_{k,2}=3$ and $\nu_{k,3}=1$. If x belongs to P_k with index, say, $\phi_k(x)=18$, we have $x\in V_{k,2}$, $|V_{k,2}|=8$ and $\phi_{k,2}(x)=2$, so that the conditioner will output the bit-string 010.

Observe that in the construction above the only non-trivial step is the first one, namely, the partitioning of $\mathscr{Q}_L$ into sets that satisfy the iso-probability condition (14); the other two steps are very easy. The major difficulty in choosing an iso-probability partition is that the iso-probability condition must continue to hold even when there is some uncertainty about the statistical description of the input symbols.

Note that Elias' original scheme satisfies the iso-probability condition by putting in the same P_k blocks with the same signature (see Section 3). These blocks have the same probability as soon as values X_n are iid. Elias' construction can clearly be applied to every source with finite alphabet. As we will show later, it can happen that in some cases (e.g., the Poisson case, see Section 6.2) a smaller partition (i.e., with $\mathscr{P}$ smaller) is possible and this can improve the efficiency of the conditioner.

A key property of the just described indexer is that it always produces an admissible conditioner.

Property 2 Every GES is an admissible conditioner (in the sense of Definition 1).

Proof By exploiting the iso-probability condition (14) and the fact that every $V_{k,j}\subset P_k$, it is very easy to prove that all the binary strings of a given output length are in $\mathrm{Im}(\mathscr{C})$ and have the same probability, that is, (7) holds. The thesis follows by applying Property 1.

5.2 Performance of Generalized Elias Schemes

The main result of this section is the following theorem that gives conditions that imply that a GES is *asymptotically efficient*. In the most commonly considered case of finite alphabet $\mathscr{A}$, a conditioner is considered asymptotically efficient if one can get a rate as close as desired to $H(X_0)$ – the maximum achievable rate – by choosing L large enough. However, if the alphabet $\mathscr{A}$ is continuous (e.g., $\mathbb{R}$), it does not make sense to talk about $H(X_0)$. Instead, the bound against which we need to measure efficiency is the entropy $H(V_0)$ of the output of the VQ. Note that one can make $H(V_0)$ as large as desired by choosing a high-resolution (and expensive) VQ.

Definition 3 Let $I\subset\mathbb{N}$ with $\sup I=\infty$ and suppose that for every $L\in I$ one has a GES conditioner $\mathscr{C}_L:\mathscr{A}^L\to\{0,1\}^*$ for blocks of size L. The family of conditioners $\{\mathscr{C}_L\}_{L\in I}$ is said to be *asymptotically efficient* if

$$\lim_{L\in I,L\to\infty}\frac{\mathbb{E}\left[\mu_{0,L}\right]}{L}-\frac{H(V_0^{(L)})}{L}=0 \tag{17}$$

where $\mu_{0,L}$ is the variable μ_0 associated with $\mathscr{C}_L$.

Theorem 2 *Let the notation be as in Definition 3. Let $\mathscr{Q}_L$ and $\mathscr{P}_L$ be, respectively, the quantizer alphabet and cardinality of the partition associated with $\mathscr{C}_L$. If $|\mathscr{Q}_L|$ and $\mathscr{P}_L$ grow slowly enough in the sense that*

$$\lim_{L\in I,L\to\infty}\frac{\log\mathscr{P}_L}{L}=0 \tag{18a}$$

$$\lim_{L\in I,L\to\infty}\frac{\log\log|\mathscr{Q}_L|}{L}=0 \tag{18b}$$

then the family of conditioners $\{\mathscr{C}_L\}_{L\in I}$ is asymptotically efficient.

Note that in the case of the trivial VQ and of an iid process X, $H(V_0^{(L)})/L=H(X)$ for every L.

Remark 3 Hypothesis (18b) can be expected to be verified in every case of interest. Indeed, it is reasonable to assume that $|\mathscr{Q}_L|\le\rho^L$ for some ρ. Note that in the discrete case we directly have $|\mathscr{Q}_L|\le|\mathscr{A}|^L$. In the continuous case, observe first that every practical implementation will have to put a bound over the maximum range of X_n. Therefore, let $U\subset\mathbb{R}$ be an interval of length $|U|$, chosen so that the probability that $X_n\notin U$ is as small as desired (e.g., if $X_n\sim\mathscr{N}(0,\sigma)$, one can choose $|U|=[-k\sigma,k\sigma]$ with k large enough). Suppose that in the process of block construction we discard X_n if $X_n\notin U$. It follows that every block belongs to a region with volume $|U|^L$. Moreover, it is reasonable to assume that the volume of the smallest region of the VQ is not smaller than D^L for some D independent on L (D can thought as the "resolution" of the VQ). It follows that the number of the VQ quantization regions (i.e., the cardinality of $\mathscr{Q}_L$) is not larger than $(|U|/D)^L$.

A key step in proving Theorem 2 is the following lemma.

Lemma 1 *For every admissible GES with block size L the following inequalities hold*

$$\frac{H(V_0^{(L)})}{L} - \frac{H(\mu_0)}{L} - \frac{\log_2 \mathscr{P}_L}{L} \leq \frac{\mathbb{E}[\mu_0]}{L} \leq \frac{H(V_0^{(L)})}{L} - \frac{H(\mu_0)}{L} \tag{19a}$$

$$0 \leq \frac{H(\mu_0)}{L} \leq \frac{\log_2 \log_2 |\mathscr{Q}_L|}{L} \tag{19b}$$

The proof is given in Appendix A.4. By using Lemma 1 it is easy to prove Theorem 2.

Proof (Proof of Theorem 2) Rewrite (19a) as

$$-\frac{H(\mu_0)}{L} - \frac{\log_2 \mathscr{P}_L}{L} \leq \frac{\mathbb{E}[\mu_0]}{L} - \frac{H(V_0^{(L)})}{L} \leq -\frac{H(\mu_0)}{L} \tag{20}$$

Observe that if (18b) is satisfied, inequality (19b) shows that $\lim_{L\to\infty} H(\mu_0)/L = 0$. Remembering hypothesis (18a), one deduces that when $L \to \infty$ both the left hand side and the right hand side of (20) go to zero. Therefore, the central quantity must go to zero, too.

6 Application of the theory

In this section we are going to show three examples of application of the theory above. First we will re-analyze the original Elias' scheme and provide a different proof of its optimality, successively we will develop two non-trivial GES: one for Poisson sources and one for Gaussian sources.

6.1 The original Elias' scheme

It is interesting to analyze the original Elias' scheme using the results and the framework outlined above. Elias considers the case of blocks of iid variables, taking values in an alphabet of size $|\mathscr{A}| = M$, which are directly partitioned into iso-probable classes P_k by considering all the permutations (with repetition) of the symbols appearing in a given input block. We are interested in giving an estimate of $\log_2 \mathscr{P}_L/L$, where $\mathscr{P}_L$ is the size of the resulting partition, which we can use in (19a) to evaluate the distance between the upper and the lower bound of the scheme rate.

Property 3 In the case of an Elias' scheme for an alphabet $\mathscr{A}$ of size $|\mathscr{A}| = M$, the following inequalities hold

$$\frac{\log_2 \log_2 |\mathscr{Q}_L|}{L} = \frac{\log_2 L}{L} + \frac{\log_2 \log_2 M}{L} \tag{21a}$$

$$\log_2\left(\frac{M-1}{L}+1\right) \leq \frac{\log_2 \mathscr{P}_L}{L} \leq \frac{1}{L}\log_2\left(\frac{1}{\sqrt{2\pi}}\frac{(M+L-1)^{M+L-0.5}}{(M-1)^{M-0.5}L^{L+0.5}}\right) \tag{21b}$$

Note that since both bounds in (21) go to zero as L increases, Property 3 together with Theorem 2 implies that the family of Elias' conditioners is asymptotically efficient, finding in a different way the original result of Elias [7].

Proof Equation (21a) follows at once from $\mathscr{Q}_L = \mathscr{A}^L$. In order to prove (21b), remember that if $x \in \mathscr{A}^L$ and $b \in \mathscr{A}$, $\#_b(x)$ is the number of times that $b \in \mathscr{A}$ appears in x. In the Elias' scheme, every set P_k is characterized by the fact that $x, y \in P_k$ if x and y have the same *signature*, i.e., $\#_b(x) = \#_b(y)$ for every $b \in \mathscr{A}$ (see Section 3). Therefore, the set P_k containing x is characterized by the $|\mathscr{A}| = M$ values $\#_b(x)$, $b \in \mathscr{A}$. The size $\mathscr{P}_L$ of partition (13) is therefore equal to the number of ways of writing L as the sum of $|\mathscr{A}|$ non-negative numbers, that is the number of *weak compositions* of L in $|\mathscr{A}|$ parts. As well known, the number of such weak compositions is

$$\mathscr{P}_L = \binom{M+L-1}{L} \tag{22}$$

By using the known bounds (see [8] for the upper bound)

$$\left(\frac{n}{k}\right)^k \leq \binom{n}{k} \leq \frac{1}{\sqrt{2\pi}}\frac{n^{n+0.5}}{(n-k)^{n-k+0.5}k^{k+0.5}} \tag{23}$$

one deduces

$$L\log_2\left(\frac{M-1}{L}+1\right) \leq \log_2 \mathscr{P}_L \leq \log_2\left(\frac{1}{\sqrt{2\pi}}\frac{(M+L-1)^{M+L-0.5}}{(M-1)^{M-0.5}L^{L+0.5}}\right) \tag{24}$$

from which (21) follows.

6.2 Finite Geometric variables

In this section we apply the above results to the conditioning of a sequence of random variables distributed according to a generalization of geometric random variables. Since the variables have finite alphabet, we use the trivial VQ, and input blocks of such variables are directly partitioned into equiprobable classes, using a scheme which is different from the Elias' one based on permutations.

Definition 4 A random variable X will be said to be an *M-finite geometric* (or simply *finite geometric*) random variable *with parameter* $p \in (0,1)$ if it assumes values in $J_M = \{0,1,\dots,M-1\}$ and its probability mass function is

$$P[X=\ell] = \frac{p}{1-(1-p)^M}(1-p)^\ell \tag{25}$$

Note that for $M=\infty$, Definition 4 reduces itself to the definition of the usual geometric random variable. The entropy of an M-finite geometric has the expression

$$H(X) = -\log_2 \frac{p}{1-q^M} - \frac{q(Mq^M - Mq^{M-1} + 1 - q^M)}{(1-q^M)p}\log_2 q. \tag{26}$$

where $q = 1-p$.

Remark 4 The reason for considering M-finite geometric random variables is that they are the natural outcome of the sampling of a Poisson process, for instance originating from natural sources. More precisely, suppose a Poisson process with intensity λ is given. Partition the positive time axis into intervals $I_n = [nD,(n+1)D)$, $n \in \mathbb{N}$, of size D and "mark" the intervals where *at least* one event of the Poisson process happens (e.g., at least one atom decays or at least one photon arrives). Let $t_k \in \mathbb{N}$ be the index of the k-th marked interval and define inter-arrival time $T_n := t_{n+1} - t_n - 1$. if the inter-arrival time T_n is measured by using a ν-bit counter fed with a clock of frequency $1/D$, so that if $T_n > 2^\nu - 1$ the counter wraps around, it follow that the value read from the counter will be $X_n := T_n \bmod 2^\nu \in \{0,\dots,2^\nu-1\}$ It is easy to prove that random variables T_n are iid and geometric with parameter $p = 1-\exp(-\lambda D)$, while random variables X_n are iid, 2^ν-finite with parameter p.

While original Elias' method to construct partition (13) can be used with any process with iid variables, not necessarily binary, we can exploit the fact that $X_n \sim \mathscr{G}(p;M)$ in order to develop a more efficient conditioner.

More precisely, we partition $\mathscr{A}^L$ according to the sum of the entries of the block, that is, for every $k \in J_{L(M-1)+1}$ and $x \in \mathscr{A}^L$, $x = [x_0,\dots,x_{L-1}]$, we define set P_k as

$$P_k = \{x \in \mathscr{A}^L : \sum_{i=0}^{L-1} x_i = k\} \qquad k \in J_{L(M-1)+1} \tag{27}$$

Note that if $x \in \mathscr{A}^L$, $0 \le \sum_{i=0}^{L-1} x_i \le L(M-1)$, so that sets (27) are a partition of $\mathscr{A}^L$.

Property 4 Partition (27) satisfies the iso-probability constraints, for any value of parameter p.

Proof It suffices to compute the probability of the generic $x = [x_0,\dots,x_{L-1}] \in \mathscr{A}^L$. More precisely, let $C = p/(1-(1-p)^M)$ in the M-finite case and $C = p$ in the infinite case and observe that

$$\begin{aligned} P[X_0 = x_0,\dots,X_{L-1} = x_{L-1}] &= \prod_{k=0}^{L-1} P[X_k = x_k] \\ &= \prod_{k=0}^{L-1} C(1-p)^{x_k} \\ &= C^L (1-p)^{\sum_k x_k} \end{aligned} \tag{28}$$

Therefore, $P[X_0 = x_0,\dots,X_{L-1} = x_{L-1}]$ depends only on the sum $\sum_{k=1}^{L} x_k$ and the iso-probability of partition (27) follows.

The proposed scheme is also asymptotically efficient, as claimed by the following property.

Property 5 The proposed conditioner for M-finite geometric variables, is efficient, that is

$$\lim_{L\to\infty} \frac{\mathbb{E}[\mu_0]}{L} = H(X) \tag{29}$$

Proof Note that $\mathscr{P}_L = L(M-1)+1 < LM$ and $|\mathscr{Q}_L| = M^L$, so that both

$$\frac{\log_2 \mathscr{P}_L}{L} < \frac{\log_2(LM)}{L} = \frac{\log_2 L + \log_2 M}{L} \tag{30}$$

and (21a) hold. Therefore, hypotheses (18) are satisfied and the conditioner is efficient.

6.3 Gaussian Source

Gaussian random variables are very common in practice. In order to construct a suitable VQ, we are going to exploit the fact that the the joint probability density function (pdf) of L iid Gaussian variables depends only on the length of the argument. More in detail, we are going to partition the space into spherical shells and successively partition every spherical shell into portions of equal volume. Every portion will be a region of the VQ. Therefore, the output of the VQ can be represented by a pair of indexes: one index denotes the spherical shell, the other the portion inside the spherical shells. The first index can be interpreted as a quantized version of the length of the vector, while the second index as a quantized version of the "angle." We will denote the r.v. associated to the two indexes respectively as $\mathfrak{R}$ and Θ. Therefore, the r.v. associated with the output of the VQ is the pair $(\mathfrak{R},\Theta)$.

More into details, the VQ construction is done as follows

1. Choose a maximum radius $R_{\max}$.
2. Choose K radii $r_1 < r_2 < \cdots < r_K = R_{\max}$. Set $r_0 = 0$ for notation convenience. Define the k-th *spherical shell* S_k as

$$S_k := \{\mathbf{x} \in \mathbb{R}^L : r_{k-1} \le \|\mathbf{x}\| < r_k\} \tag{31}$$

Moreover, define, for notation convenience, $S_{K+1} = \{\mathbf{x} \in \mathbb{R}^L : \|\mathbf{x}\| > R_{\max}\}$.

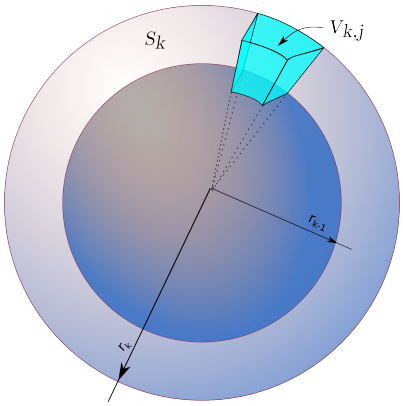

Fig. 3 An example of quantizer region $V_{k,j}$ obtained by intersecting the spherical shell S_k with a cone.

3. For every $k = 1,\dots,K+1$, choose the number of sets N_k used to partition shell S_k. Choose a radius r and partition the sphere $\|\mathbf{x}\| = r$ into N_k subsets with the same area (using, for example, the algorithm in [9, 10]). Let $U_{k,j}$ be the j-th element of such a partition. Of course, one can always reduce itself to the case $r = 1$, but allowing for a general r will simplify the study later.
4. For every $k = 1,\dots,K+1$ and every $j = 1,\dots,N_k$, define set

$$V_{k,j} := \{\mathbf{x} \in S_k : r\mathbf{x}/\|\mathbf{x}\| \in U_{k,j}\} \tag{32}$$

   In other words, $V_{k,j}$ is the set of vectors belonging to the k-th shell S_k that "fall" inside $U_{k,j}$ when projected on the sphere $\|\mathbf{x}\| = r$ (see Fig. 3). Note that $S_k = \cup_j V_{k,j}$. Every $V_{k,j}$ is a region of the VQ. Therefore, the set of pairs (k,j) can be considered as the quantizer alphabet $\mathscr{Q}_L$.
5. It is easy to see that as soon as r.v. X_n are Gaussian, zero mean and iid, for every $k \in \{1,\dots,K+1\}$ and $j_1, j_2 \in \{1,\dots,N_k\}$, the following holds

$$P[\mathscr{X}_0 \in V_{k,j_1}] = P[\mathscr{X}_0 \in V_{k,j_2}] \tag{33}$$

   Therefore, the spherical shell S_k is partitioned into regions with the same probability, and the corresponding VQ outputs $(\mathfrak{R},\Theta) = (k,j)$ satisfy the iso-probability condition.

In order to verify if the proposed VQ is asymptotically efficient and compute its rate, it is necessary to get an estimate of both $\mathscr{P}_L = K+1$ and $|\mathscr{Q}_L| = \sum_k N_k$, in order to check hypotheses (18). To such an end, it is necessary to describe explicitly how $R_{\max}$, K and values N_k are chosen for every block-size L. The following is a possible, reasonable choice, although not the only possible one.

1. Choose $\varepsilon < 1$ and choose the maximum radius $R_{\max}$ in order to have $P[\|\mathscr{X}_0\| > R_{\max}] \le \varepsilon$
2. Choose a constant D (i.e., independent on L) that will represent the resolution of the VQ. Note that D can be considered as a measure of the "cost" of the VQ. We will partition the space into spherical shells of thickness at least D, that is, we will have $r_k - r_{k-1} \ge D$.
3. One possible choice for $U_{k,j}$ is as follows. The same angular partition is used for all k. More precisely, the sphere with radius $r_L = \sigma\sqrt{L}$ is quantized, with the constraint that the measure of each element is not smaller than βD^{L-1} where β is a constant that does not depend on L. The choice for r_L is due to the fact that $\|\mathscr{X}_0\|$ is approximately distributed as $\mathscr{N}(\sigma\sqrt{L-1},\sigma)$ and we can expect $\|\mathscr{X}_0\|$ be close to r_L.
   This solution has the advantage of making the VQ "separable:" first the norm $\|\mathscr{X}_0\|$ is quantized with step D, successively $\mathscr{X}_0/\|\mathscr{X}_0\|$ is quantized using the chosen angular partition.

We will denote as $\mathscr{C}_L^{\mathscr{N}}$ the conditioner resulting from the above choices.

Theorem 3 *The family of conditioners $\{\mathscr{C}_L^{\mathscr{N}}\}_{L\in\mathbb{N}}$ is asymptotically efficient and, moreover, its rate for large L satisfies*

$$\lim_{L\to\infty} \frac{\mathbb{E}[\mu_0]}{L} = \log_2\sqrt{2\pi e} + \log_2\frac{\sigma}{D} \tag{34}$$

Note that (34) is the entropy of a r.v. $\mathscr{N}(0,\sigma^2)$ quantized with step D. In order to prove Theorem 3 we need few lemmas whose proofs are in Appendix A.5.

Lemma 2 *Suppose the L dimensional sphere surface of radius r is partitioned into pieces of area βD^{L-1}. Let $W_{L,r}$ the number of pieces. The following equality holds*

$$\frac{\log_2 W_{L,r}}{L} = \log_2\sqrt{2\pi e} - \log_2 D + \log_2\frac{r}{\sqrt{L}} + o(1) \tag{35}$$

where $o(1)$ denotes a term that goes to zero when $L \to \infty$.

Lemma 3 *There exist constants C_1, C_2, C_3, C_4 and L_0 such that for all $L > L_0$*

$$R_{max} \le C_1\sqrt{L} \tag{36a}$$
$$\mathscr{P}_L \le C_2\sqrt{L} \tag{36b}$$
$$|\mathscr{Q}_L| \le \left(C_3\sqrt{L}\right)^L \tag{36c}$$

Proof (Proof of Theorem 3) It is immediate to show that (36b) and (36c) imply (18), therefore the family of conditioners it is asymptotically efficient. Since the family is asymptotically efficient, the rate converges (in the sense of (17)) to $H(V_0)/L$. In order to compute an estimate of $H(V_0)/L$,

remember that V_0 is the pair $(\mathfrak{R},\Theta)$ and observe that since $H(V_0)=H(\mathfrak{R})+H(\Theta)$ and $H(\mathfrak{R})\le \log_2 \mathscr{P}_L$

$$\begin{aligned}\frac{H(\Theta|\mathfrak{R})}{L} &\le \frac{H(V_0)}{L}\\ &\le \frac{\log_2 \mathscr{P}_L}{L}+\frac{H(\Theta|\mathfrak{R})}{L}\\ &\le \frac{\log_2 L+2\log_2 C_2}{2L}+\frac{H(\Theta|\mathfrak{R})}{L}\end{aligned} \tag{37}$$

where we exploited bound (36b). It turns out that the rate for large L is dominated by the entropy $H(\Theta|\mathfrak{R})$ and that $\mathfrak{R}$ gives a negligible contribution. Observe that $H(\Theta|\mathfrak{R})=H(\Theta)=\log_2 W_{L,r_L}$ (with the notation of Lemma 2). The thesis follows by using Lemma 2 with $r=r_L=\sigma\sqrt{L}$.

7 Experiments

In this section, we focus on the case of finite geometric variables and compare the performance of the proposed scheme with the one obtained using the original Elias' method to create the iso-probable classes. We assume that $x=[X_0,....,X_{L-1}]$ is a vector of independent $M-$finite geometric random variables, each with probability mass function given by (25) and entropy (19a). Let $[x_0,...,x_{L-1}]$ denote an instance of x.

As explained in Section 5.1, the conditioner generates, for an input block x, a bit string $\mathbf{b}_x=\mathscr{C}(x)$ of length $|\mathbf{b}_x|$. The actual string $\mathbf{b}_x$ will depend on the method used to form classes (e.g., the original Elias' procedure or the proposed one).

It is therefore easy to compute the efficiency of the two schemes on the basis of the average output bit-string length

$$\mathbb{E}[\mu_0]=\sum_x |\mathbf{b}_x| p(x),$$

where $p(x)$ is the product of L probability mass functions of type (25). Table 1 shows $\mathbb{E}[\mu_0]/L$ for the two methods when the variables are represented modulo $M=16$. We set $p=0.1$. Note that the entropy of a geometric random variable with parameter $p=0.1$ is $H(T)=4.6900$, while the entropy of the finite geometric $X=T \mod M$, $M=16$, is $H(X)=3.8411$. As an example, for $L=2$, the Elias' partition method will output 1 bit for all realizations where $x_0\neq x_1$. Using (25), one finds, for Elias' method,

$$\frac{\mathbb{E}[\mu_0]}{2}=\frac{1}{2}-\frac{1}{2}\frac{(1-(1-p)^{2M})p^2}{(1-(1-p)^2)(1-(1-p)^M)^2},$$

which evaluates to 0.4617 when $p=0.1$, and should be compared with the value 1.1272 attained by the proposed approach.

For larger values of L and M we simulated $N=15000$ realizations of x, concatenate the output binary strings into one string S_N and plot in Fig. 4 the values $|S_N|/(LN)$ for $M=16$ and $M=64$, $L=2,...,10$. Note that for $M=64$, we have $H(X)=4.6768$, due to the fact that the modulo operation has less influence for larger M. Simulations were performed in ®Matlab using the default uniform random number generator *Mersenne Twister* and code

```
mod(floor(log(rand)/log(1-p)),M)
```

to generate M-finite geometric random variables.

Although both methods approach the entropy of the source as L increases, the table and the figures clearly show the possible advantage of the proposed method[1].

Table 1 Average length, in bit/symbol for the Elias' partition and the proposed one.

	$L=2$	$L=3$	$L=4$	$L=5$
Elias part.	0.4617	0.4820	0.8104	0.9164
Proposed	1.1272	1.8488	2.2985	2.5738

8 Conclusions

In this paper, we introduced the concept of a block-based admissible conditioner and of a Generalized Elias Scheme which considers generic classes of equi-probable vectors, derived from both discrete or continuous variables. We analyzed the scheme in detail an derived efficiency bounds that can be used to quantify its performance with finite length blocks, as well as the asymptotic behavior for increasing block length. We applied the theory to provide an alternative proof of the optimality of the original Elias' scheme based on permutations. We also considered the problem of measuring randomness from a Poisson process and proposed a conditioner for the resulting geometric-like random vectors. Finally, we analyzed the case of quantized random vectors of iid Gaussian variables. A comparison with the original Elias' scheme for non-binary vectors, in a special case which exploits the particular structure of the geometric distribution, confirms the potential advantage of the proposed generalized approach.

A Proofs and Technical Details

A.1 Formal definition of output process b_k

In order to make the exposition simpler, we will extend the set of bit values with a symbol Λ to be used as an "undefined bit." As a first step, we define $b_k^{(N)}$ as the k-th output bit after processing block number N as $S_{N,k}$ if $\ell_N>k$ or Λ if if $\ell_N\le k$. In other words, if after processing the N-th block, the k-th bit has been generated, $b_k^{(N)}$ is equal to the generated bit; otherwise its value is the undefined bit value Λ. Note that $b_k^{(N)}$ is "stable" in the following sense: if $b_k^{(N)}\neq\Lambda$ and $M>N$ then $b_k^{(N)}=b_k^{(M)}$.

[1] We consider here non binary vectors. One could exploit the fact that geometric variables can be equivalently represented by sparse binary strings and use the original Elias' scheme based on permutations.

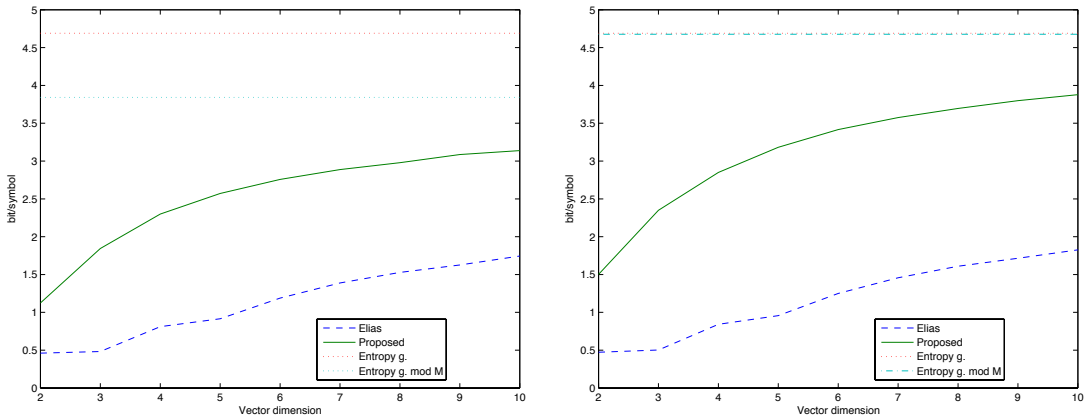

Fig. 4 Comparison between the Elias' method and the proposed one. (a) $M = 16$, (b) $M = 64$.

We will define $\mathscr{N}(k) \in \mathbb{N} \cup \{\infty\}$ as the smallest N such that $\ell_N > k$; if no such N exists (that is, it is always $\ell_N \leq k$), we define $\mathscr{N}(k) = \infty$.

Now we can define b_k, the k-th bit of the output process as $S_{\mathscr{N}(k),k}$ if $\mathscr{N}(k) < \infty$ or Λ if $\mathscr{N}(k) = \infty$ In words, if at some time the k-th bit is generated, b_k is the generated value, while if the number of generated bits remains not larger then k, then b_k is undefined.

Remark 5 It is intuitive that the case $\mathscr{N}(k) = \infty$ is very unlikely (actually, we will show that it is a zero probability event); nevertheless, we must take it into account since one can find realizations of the input process X that correspond to a finite string of output bits; for example, if the conditioner uses the von Neumann algorithm, any realization ending with an infinite sequence of zeros produces a finite length output.

Property 6 If $\mathbb{E}[\mu_0] > 0$, then for every $k \in \mathbb{N}$, the event $b_k \neq \Lambda$ (or equivalently, $\mathscr{N}(k) < \infty$) happens almost surely.

Remark 6 Note that $\mathbb{E}[\mu_0] = 0$ only if the conditioner produces the empty string for every possible input block.

Proof It suffices to show that for every $k \in \mathbb{N}$ the probability of having $\ell_N \leq k$ for all $N \in \mathbb{N}$ is zero. Let $E_{N,k}$ denote the event $\ell_N \leq k$ and observe that the $E_{1,k} \subseteq E_{2,k} \subseteq \cdots$. Therefore, it suffices to show that $\lim_{N\to\infty} P[E_{N,k}] = 0$.

We will use the Chebyshev inequality. Observe that ℓ_N is a random variable with mean $N\mathbb{E}[\mu_0]$ and variance $N\sigma^2$, where σ^2 is the variance of μ_0. If $\ell_N \leq k$, then

$$\ell_N - \mathbb{E}[\ell_N] = \ell_N - N\mathbb{E}[\mu_0] \leq k - N\mathbb{E}[\mu_0] \tag{38}$$

When $N > k/\mathbb{E}[\mu_0]$, we have $k - N\mathbb{E}[\mu_0] < 0$, and event (38) implies (that is, is contained in) $|\ell_N - \mathbb{E}[\ell_N]| \geq N\mathbb{E}[\mu_0] - k$, so that

$$\begin{aligned} P[E_{N,k}] &\leq P\left[|\ell_N - \mathbb{E}[\ell_N]| \geq N\mathbb{E}[\mu_0] - k\right] \\ &\leq \frac{N\sigma^2}{(N\mathbb{E}[\mu_0] - k)^2} = \left(\frac{\sigma}{\sqrt{N}\mathbb{E}[\mu_0] - k/\sqrt{N}}\right)^2 \end{aligned} \tag{39}$$

and the last term in (39) goes to zero when $N \to \infty$.

A.2 Admissibility is stronger than UBP

In this appendix we are going to show that if a conditioner is admissible according to Definition 1, then the output process is an UBP. Although this result is quite intuitive, its formal proof has few technicalities. First, however, we prove the fact that scheme (5) is UBP, providing a counterexample that proves that UBP does not imply admissibility.

Proof A.1. It is possible to see that scheme (5) is UBP in several ways. An easy, albeit informal, approach is to observe that mapping (5) is, actually, the Huffmann code for X_n, which in this case is optimal (and *not only asymptotically* optimal) since the symbol probabilities are of the form $2^{-\ell}$. This implies that the average number of bits per input symbol generated by the conditioner is equal to the source entropy, so the resulting bit string b is not compressible and iid (i.e., b is a UBP).

A more precise proof is the following.

Consider bit b_k, let n_k be the index of the input symbol that generates b_k. Note that it must be $j_k := \ell_{n_k-1} \in \{k-1, k-2\}$. Compute $P[b_k = 0]$ by conditioning with respect to j_k

$$P[b_k = 0] = P[b_k = 0, j_k = k-1] + P[b_k = 0, j_k = k-2] \tag{40}$$

If $j_k = k-1$ and $b_k = 0$ it must be $X_{n_k} \in \{a, b\}$. Since $P[X_{n_k} \in \{a,b\}] = 1/2$, the first conditional probability is $1/2$. If $j_k = k-2$ and $b_k = 0 \ldots$ □

Our first step will be to prove that (6), used in the definition of an admissible conditioner, implies a more general relation.

Lemma 4 *If $\mathscr{C}$ is an admissible conditioner, then the following holds*

$$\begin{aligned} &\forall N \in \mathbb{N}, G \geq 1, \mathbf{k} \in \mathbb{N}^G, m > k_G, m \in \mathrm{Im}(\ell_N), \mathbf{b} \in \{0,1\}^G \\ &\qquad P[S_{n,\mathbf{k}} = \mathbf{b} | \ell_N = m] = 2^{-G} \end{aligned} \tag{41}$$

Remark 7 The difference between (6) and (41) is that (6) takes into account only a single block, while (41) is relative to the output after processing the N-th block.

Proof The proof is complicated by the fact that two different bits of S_N, say $S_{N,k}$ and $S_{N,j}$, can derive from the same block or from different blocks, depending on the specific realization of X. In order to take into account this problem, we are going to condition on the specific sequence of lengths $\mu_0, \mu_1, \ldots, \mu_N$ that gave rise to $\ell_N = m$.

In order to simplify the manipulations, it is worth to introduce some notation. With $\mu_{0:N}$ we will denote the vector $[\mu_0, \mu_1, \ldots, \mu_N]$; with $\mathfrak{M}$ we denote the set of vectors $\mu_{0:N}$ such that $\ell_N = m$, formally,

$$\mathfrak{M} := \{\mathbf{u} \in \mathrm{Im}(\mu_0)^{N+1} : \sum_{k=0}^{N} \mathbf{u}_k = m\} \tag{42}$$

Note that the condition $\mathbf{u} \in \mathrm{Im}(\mu_0)^{N+1}$ in (42) forces every component of $\mathbf{u}$ to be a possible length for an output block.

It is clear that the event $\ell_N = m$ is equal to the event $\mu_{0:N} \in \mathfrak{M}$, so that the following holds

$$P[S_{N,\mathbf{k}} = \mathbf{b} | \ell_N = m] = \sum_{\mathbf{u} \in \mathfrak{M}} P[S_{N,\mathbf{k}} = \mathbf{b} | \mu_{0:N} = \mathbf{u}] \, P[\mu_{0:N} = \mathbf{u} | \ell_N = m] \tag{43}$$

Therefore, if we prove that $P[S_{N,\mathbf{k}} = \mathbf{b} | \mu_{0:N} = \mathbf{u}] = 2^{-G}$ for every $\mathbf{u} \in \mathfrak{M}$, the thesis will follow.

Given $\mathbf{u} \in \mathfrak{M}$, split $\mathbf{k}$ into $N+1$ pieces ξ_j, $j = 0, \dots, N$, as follows

$$\xi_j = \{ o \in \mathbf{k} : \ell_{j-1} \le o < \ell_j \} \tag{44}$$

that is, ξ_j contains the indexes in $\mathbf{k}$ that correspond to bits output after processing block number j.

Define $\mathbf{b}_j$ as the subword of $\mathbf{b}$ corresponding to the indices in ξ_j and observe that

$$\begin{aligned} P[S_{N,\mathbf{k}} = \mathbf{b} | \mu_{0:N} = \mathbf{u}] &= P\left[\bigwedge_{j=0}^{N} S_{N,k_j} = \mathbf{b}_j | \mu_{0:N} = \mathbf{u} \right] \\ &= \prod_{j=0}^{N} P\left[S_{N,k_j} = \mathbf{b}_j | \mu_{0:N} = \mathbf{u} \right] \\ &= \prod_{j=0}^{N} P\left[S_{N,k_j} = \mathbf{b}_j | \mu_j = \mathbf{u}_j \right] = \prod_{j=0}^{N} 2^{-|k_j|} \\ &= 2^{-\sum_{j=0}^{N} |k_j|} = 2^{-G} \end{aligned} \tag{45}$$

where we exploited the fact that different blocks are independent at second step and the admissibility hypothesis at the last one.

Theorem 4 *If $\mathscr{C}$ is admissible, then process b is an UBP*

Proof We need to prove that for every $G \ge 1$, every $\mathbf{k} \in \mathbb{N}^G$ and every $\mathbf{b} \in \{0,1\}^G$, the following equality holds

$$P[b_{\mathbf{k}} = \mathbf{b}] = 2^{-G} \tag{46}$$

We are going to compute probability (46) by conditioning over the values assumed by $\mathscr{N}(k_L)$. It is

$$P[b_{\mathbf{k}} = \mathbf{b}] = \sum_{N \in \mathbb{N}} P[b_{\mathbf{k}} = \mathbf{b} | \mathscr{N}(k_G) = N] P[\mathscr{N}(k_G) = N] \tag{47}$$

where we take the convention that in sum (47) the terms with $P[\mathscr{N}(k_G) = N] = 0$ are removed (otherwise conditional probability $P[b_{\mathbf{k}} = \mathbf{b} | \mathscr{N}(k_G) = N]$ would not make sense). Since by Lemma 4 every conditional probability in (47) is equal to 2^{-G}, the thesis follows.

A.3 Proof of Property 1: (7) implies (6)

Proof Let G, $\mathbf{k}$, m and $\mathbf{b}$ be chosen according to (6). Let $\overline{\mathbf{k}} \in \mathbb{N}^{m-G}$ be the "complement" of $\mathbf{k}$ in the sense that the union of $\mathbf{k}$ and $\overline{\mathbf{k}}$ is $J_m = \{0, 1, \dots, m-1\}$ and $\mathbf{k}$ and $\overline{\mathbf{k}}$ are disjoint. Moreover, if $\mathbf{b} \in \{0,1\}^G$ and $\mathbf{a} \in \{0,1\}^{m-G}$, we will denote with $[\mathbf{b}_{\mathbf{k}}, \mathbf{a}_{\overline{\mathbf{k}}}]$ the m-bit bitstring equal to $\mathbf{b}$ in the positions specified by $\mathbf{k}$ and equal to $\mathbf{a}$ in the positions specified by $\overline{\mathbf{k}}$.

Note that the event $S_{0,\mathbf{k}} = \mathbf{b} \wedge \mu_0 = m$ can be written as

$$\bigvee_{\mathbf{a} \in \{0,1\}^{m-G}} (S_{0,\mathbf{k}} = \mathbf{b}) \wedge (S_{0,\overline{\mathbf{k}}} = \mathbf{a}) \wedge (\mu_0 = m) \tag{48}$$

and that all the events in (48) are disjoint. By using (48) one can write

$$\begin{aligned} P[S_{0,\mathbf{k}} = \mathbf{b} | \mu_0 = m] &= \frac{P[S_{0,\mathbf{k}} = \mathbf{b}, \mu_0 = m]}{P[\mu_0 = m]} \\ &= \frac{\sum_{\mathbf{a} \in \{0,1\}^{m-G}} P[S_{0,\mathbf{k}} = \mathbf{b}, S_{0,\overline{\mathbf{k}}} = \mathbf{a}, \mu_0 = m]}{P[\mu_0 = m]} \\ &= \sum_{\mathbf{a} \in \{0,1\}^{m-G}} P[S_{0,\mathbf{k}} = \mathbf{b}, S_{0,\overline{\mathbf{k}}} = \mathbf{a} | \mu_0 = m] \\ &= \sum_{\mathbf{a} \in \{0,1\}^{m-G}} P[S_0 = [\mathbf{b}_{\mathbf{k}}, \mathbf{a}_{\overline{\mathbf{k}}}] | \mu_0 = m] \\ &= \sum_{\mathbf{a} \in \{0,1\}^{m-G}} 2^{-m} \\ &= 2^{m-G} 2^{-m} = 2^{-G} \end{aligned} \tag{49}$$

where we exploited decomposition (48) at second step and Hypothesis (7) at the last one.

A.4 Proof of bounds (19)

The key observation used to derive bounds (19) is given by the following general lemma.

Lemma 5 *Let A and B be two finite sets. Let $\mathscr{P} := \{Q_1, Q_2, \dots, Q_M\}$ be a partition of A and let $\pi : A \to \mathscr{P}$ be the corresponding* projection, *that is, the function mapping every $x \in A$ to the set of $\mathscr{P}$ that includes x.*
Let $f : A \to B$ be a map such that for every $k = 1, \dots, M$, its restriction $f|_{Q_k}$ to Q_k is injective. Finally, let X be a random variable assuming values in A and let $Y = f(X)$ and $U = \pi(X)$.
The following relations hold

$$\begin{aligned} H(Y) &= H(X) - H(U|Y) \\ &\ge H(X) - H(U) \\ &\ge H(X) - \log_2 |\mathscr{P}| \end{aligned} \tag{50}$$

Proof The key observation is that, with the theorem hypothesis, map $x \mapsto (\pi(x), f(x))$ is injective. Indeed, if $x, y \in A$ and $(\pi(x), f(x)) = (\pi(y), f(y))$, then x and y must belong to the same set $Q = \pi(x)$ of $\mathscr{P}$ (because $\pi(x) = \pi(y)$). Then $f(x) = f(y)$ implies $f|_Q(x) = f|_Q(y)$ which in turn implies $x = y$ since f restricted to Q is injective.

By exploiting the injectivity of $x \mapsto (\pi(x), f(x))$ one deduces $H(X) = H(YU)$. By observing that $H(YU) = H(Y) + H(U|Y)$, (50) follows.

The following lemma allows us to apply Lemma 5 to the specific case of GES.

Lemma 6 *If $\mathscr{C} : \mathscr{Q}_L \to \{0,1\}^*$ is an indexing map, then its restriction to any P_k is injective, that is, for every $k \in \{1, \dots, \mathscr{P}\}$, if $x, y \in P_k$ and $\mathscr{C}(x) = \mathscr{C}(y)$, then $x = y$.*

Proof Suppose $x, y \in P_k$. Since $\mathscr{C}(x)$ and $\mathscr{C}(y)$ have the same length, both x and y must belong to the same $V_{k,i}$, so that

$$\phi_{k,i}(x) = \mathscr{C}(x) = \mathscr{C}(y) = \phi_{k,i}(y) \tag{51}$$

Since $\phi_{k,i}$ is bijective, $x = y$ follows.

By using Lemma 5 and Lemma 6 it is easy to prove the following corollary

Corollary 2 *In a GES the following inequality holds.*

$$H(V_0) \;\ge\; H(S_0) \;\ge\; H(V_0) - \log_2 \mathscr{P} \tag{52}$$

Corollary 2 is interesting because it shows that a GES with a small number of sets in partition (13) has the potential of being more efficient than a scheme with a larger number of sets.

Corollary 3 *In a GES inequality (19a) holds.*

Proof Use (11) in (52).

Finally, we are going to prove bound (19b).

Lemma 7 *For every GES (19b) holds. In particular, for every GES* $\lim_{L\to\infty} H(\mu_0)/L = 0$.

Proof Observe that $\mathrm{Im}(\mu_0)$ is the set of the lengths of the bit-strings in the image of $\mathscr{C}$ and that

$$H(\mu_0) \le \log_2 |\mathrm{Im}(\mu_0)| \tag{53}$$

The maximum value in $\mathrm{Im}(\mu_0)$ is $\max_{k,i} v_{k,i}$, so that

$$|\mathrm{Im}(\mu_0)| \le \max_{k,i} v_{k,i} = \max_{k,i} \log_2 |V_{k,i}| \le \log_2 |\mathscr{Q}_L| \tag{54}$$

where the second inequality follows from $V_{k,i} \subseteq \mathscr{Q}_L$. Using (54) in (53) we obtain

$$\frac{H(\mu_0)}{L} \le \frac{\log_2 |\mathrm{Im}(\mu_0)|}{L} \le \frac{\log_2 \log_2 \mathscr{Q}_L}{L} \tag{55}$$

A.5 Bounds on VQ size for the Gaussian case

Proof A.2. *Proof of Lemma 2* Define, for notation convenience, $\theta = \sqrt{\pi}$. As well known, the surface area of an L-dimensional sphere of radius r is

$$\frac{2\pi^{L/2} r^{L-1}}{\Gamma\left(\frac{L}{2}\right)} = \frac{2}{r} \frac{\theta^L r^L}{\Gamma\left(\frac{L}{2}\right)} \tag{56}$$

Therefore, the number of pieces is

$$W_{L,r} = \frac{2D}{\beta r} \frac{\theta^L r^L}{D^L \Gamma\left(\frac{L}{2}\right)} \tag{57}$$

Let, for notation convenience, $C = \ln(2D/\beta r)$. By remembering the Stirling's approximation

$$\ln \Gamma(x) = x \ln x - x + O(\ln x) \tag{58}$$

it follows that

$$\begin{aligned} \frac{\ln W_{L,r}}{L} &= \underbrace{\frac{C}{L}}_{o(1)} + \ln \frac{\theta r}{D} - \frac{1}{2} \ln \frac{L}{2} + \frac{1}{2} + \underbrace{\frac{O(\ln L)}{L}}_{o(1)} \\ &= \ln \sqrt{2\pi e} - \ln D + \ln \frac{r}{\sqrt{L}} + o(1) = \ln \sqrt{2\pi e} + \ln \frac{\sigma}{D} + o(1) \end{aligned} \tag{59}$$

The thesis follows by multiplying (59) by $\log_2 e$. □

Proof A.3. *Proof of (36a)* It is well known that r.v. $\|\mathscr{X}_0\|^2/\sigma^2$ is a chi-squared r.v. with L degrees of freedom. The requirement that the overflow probability is smaller than ε can be written as

$$\begin{aligned} \varepsilon &= P[\|\mathscr{X}_0\| > R_{\max}] \\ &= P[\|\mathscr{X}_0\|^2/\sigma^2 > R_{\max}^2/\sigma^2] \\ &= P[\chi_L^2 > R_{\max}^2/\sigma^2] \end{aligned} \tag{60}$$

With the notation of [11], from (60) one deduces $u(\varepsilon, L) = R_{\max}^2/\sigma^2$. By using bounds [11]

$$u(\varepsilon, L) \le L - 2\ln\varepsilon + 2\sqrt{-L\ln\varepsilon} \tag{61}$$

one deduces

$$R_{\max} \le \sigma \sqrt{L - 2\ln\varepsilon + 2\sqrt{-L\ln\varepsilon}} \tag{62}$$

The final step is to upper bound the argument of the first square root in (62) with $2L$. This happens if

$$-2\ln\varepsilon + 2\sqrt{-L\ln\varepsilon} \le L \tag{63}$$

which is equivalent to

$$\frac{-\ln\varepsilon}{L} + 2\sqrt{\frac{-\ln\varepsilon}{L}} \le 1 \tag{64}$$

It is immediate to verify that the left hand side of (64) is monotone decreasing in L and it goes to zero when $L \to \infty$. Therefore, there exists L_0 such that (64) is true for every $L > L_0$. Therefore, (36a) is true for $L > L_0$ and $C_1 = \sigma\sqrt{2}$. □

Proof A.4. *Proof of (36b)* For the sake of simplicity, we will suppose that D was chosen in order to make (65) integer. Since every shell has thickness D and every shell is an element of the partition, the partition size is the total number of shells, that is,

$$\mathscr{P}_L = K + 1 = \frac{R_{\max}}{D} + 1 \le \left(\frac{\sqrt{2}\sigma}{D} + 1\right)\sqrt{L} \tag{65}$$

which is (36b) with $C_2 = \sqrt{2}\sigma/D + 1$. □

Proof A.5. *Proof of (36c)* According to (57), with $W_{L,1} = N_1$, the number of regions in the VQ can be upper-bounded as

$$\begin{aligned} |\mathscr{Q}_L| = (K+1)N_1 &\le (C_2\sqrt{L}) \frac{2D}{\sigma\sqrt{L}\beta} \frac{\theta^L \sigma^L L^{L/2}}{D^L \Gamma\left(\frac{L}{2}\right)} \\ &\le A \left(\frac{\theta\sigma\sqrt{L}}{De^{\gamma/2}}\right)^L = A\left(B\sqrt{L}\right)^L \le \left(AB\sqrt{L}\right)^L \end{aligned} \tag{66}$$

with obvious meaning of A and B. In (66) we used bound $\Gamma(x+1) > \exp(\gamma x)$, $x > 0$, [12, 13], where $\gamma \approx 0.577$ is the Euler-Mascheroni constant. Equation (66) is (36c) with $C_3 = AB$. □

References

1. J. von Neumann, "Various techniques used in connection with random digits," *Appl. Math. Ser., Notes by G. E. Forstyle, Nat. Bur. Stand.*, vol. 12, pp. 36–38, 1951.
2. W. Hoeffding and G. Simon, "Unbiased coin tossing with a biased coin," *Ann. Math. Statist.*, vol. 41, pp. 341–352, 1970.
3. P. Elias, "The efficient construction of an unbiased random sequence," *Ann. Math. Statist.*, vol. 43, pp. 865–870, 1972.
4. Q. Stout and B. Warren, "Unbiased coin tossing with a biased coin," *Ann. Probab.*, vol. 12, pp. 212–222, 1984.
5. Y. Peres, "Unbiased coin tossing with a biased coin," *Ann. Statist.*, vol. 20, pp. 590–597, 1992.
6. H. Zhou and J. Bruck, "Efficient generation of random bits from finite state markov chains," *Information Theory, IEEE Transactions on*, vol. 58, no. 4, pp. 2490–2506, 2012.
7. P. Elias, "The efficient construction of an unbiased random sequence," *The Annals of Mathematical Statistics*, vol. 43, no. 3, pp. 865–870, 1972.

8. P. Stănică, "Good lower and upper bounds on binomial coefficients." *JIPAM. Journal of Inequalities in Pure & Applied Mathematics [electronic only]*, vol. 2, no. 3, pp. Paper No. 30, 5 p., electronic only–Paper No. 30, 5 p., electronic only, 2001. [Online]. Available: http://eudml.org/doc/121842
9. P. Leopardi, "Diameter bounds for equal area partitions of the unit sphere," *Electronic Transactions on Numerical Analysis*, pp. 1–16, 2009.
10. ——, "A partition of the unit sphere into regions of equal area and small diameter," *Electronic Transactions on Numerical Analysis*, vol. 25, 2006.
11. T. Inglot, "Inequalities for quantiles of the chi-square distribution," *PROBABILITY AND MATHEMATICAL STATISTICS*, vol. 30, no. 2, pp. 339–351, 2010.
12. F. Qi, "Bounds for the ratio of two gamma functions," *Journal of Inequalities and Applications*, 2010.
13. A. Laforgia and P. Natalini, "On some inequalities for the gamma function," *Advances in Dynamical Systems and Applications*, vol. 8, no. 2, pp. 261–267, 2013.