The Golden Ratio Encoder

Ingrid Daubechies, C. Sinan Güntürk, Yang Wang, and Özgür Yılmaz

Abstract

This paper proposes a novel Nyquist-rate analog-to-digital (A/D) conversion algorithm which achieves exponential accuracy in the bit-rate despite using imperfect components. The proposed algorithm is based on a robust implementation of a *beta-encoder* with $\beta = \phi = (1+\sqrt{5})/2$, the *golden ratio*. It was previously shown that beta-encoders can be implemented in such a way that their exponential accuracy is robust against threshold offsets in the quantizer element. This paper extends this result by allowing for imperfect analog multipliers with imprecise gain values as well. We also propose a formal computational model for algorithmic encoders and a general test bed for evaluating their robustness.

Index Terms

Analog-to-digital conversion, beta encoders, beta expansions, golden ratio, quantization, robustness

I. Introduction

In A/D conversion, the aim is to quantize analog signals, i.e., to represent analog signals, which take their values in the continuum, by finite bitstreams. Basic examples of analog signals include audio signals and natural images. Frequently, the signal is first sampled on a grid in its domain, which is sufficiently dense so that perfect (or near-perfect) recovery from the acquired sample values is ensured by an appropriate sampling theorem. After this so-called sampling stage, there are two common strategies to quantize the sequence of sample values. *Oversampling* or $\Sigma\Delta$ Analog-to-Digital converters (or ADCs) incorporate memory elements in their structure so that the quantized value of a sample depends on other sample values and their quantization. The accuracy of such converters can be assessed by comparing the continuous input signal with the continuous output signal obtained from the quantized sequence (after a digital-to-analog (D/A) conversion stage). In contrast, *Nyquist-rate* ADCs quantize sample values separately, with the goal of approximating each sample value as closely as possible using a given bit-budget, i.e., the number of bits one is allowed to use to quantize each sample. In the case of Nyquist-rate ADCs, the analog objects of interest reduce to real numbers in some interval, say $[0, 1]$.

In this paper we focus on Nyquist-rate ADCs. Let $x \in [0, 1]$. The goal is to represent x by a finite bitstream, say, of length N. A straightforward approach is to consider the standard binary (base-2) representation of x,

$$x = \sum_{n=1}^{\infty} b_n 2^{-n}, \quad b_n \in \{0, 1\}. \tag{1}$$

Ingrid Daubechies is with the Department of Mathematics and with the Program in Applied and Computational Mathematics, Princeton University, Princeton, NJ 08544 USA (email:ingrid@math.princeton.edu).

C. Sinan Güntürk is with the Courant Institute of Mathematical Sciences, New York, NY 10012 USA (email:gunturk@courant.nyu.edu).

Yang Wang is with the Department of Mathematics, Michigan State University, East Lansing, MI 48824 USA (email:ywang@math.msu.edu).

Özgür Yılmaz is with the Department of Mathematics, The University of British Columbia, Vancouver, BC V6T 1Z2 Canada (email:oyilmaz@math.ubc.ca).

and to let x_N be the N-bit truncation of the infinite series in (1), i.e.,

$$x_N = \sum_{n=1}^{N} b_n 2^{-n}. \tag{2}$$

It is easy to see that $|x - x_N| \leq 2^{-N}$ so that $(b_1, b_2, \ldots, b_N)$ provide an N-bit quantization of x with *distortion* not more than 2^{-N}. This method is known as *pulse code modulation (PCM)* and essentially provides the most efficient encoding in a rate-distortion sense.

As our goal is analog-to-digital conversion, the next natural question is how to compute the bits b_n on an analog circuit. One popular method to obtain b_n, called *successive approximation*, extracts the bits using a recursive operation. Let $x_0 = 0$ and suppose x_n is as in (2) for $n \geq 1$. Define $u_n := 2^n(x - x_n)$ for $n \geq 0$. It is easy to see that the sequence $(u_n)_0^\infty$ satisfies the recurrence relation

$$u_n = 2u_{n-1} - b_n, \quad n = 1, 2, \ldots, \tag{3}$$

and the bits can simply be extracted via the formula

$$b_n = \lfloor 2u_{n-1} \rfloor = \begin{cases} 1, & u_{n-1} \geq 1/2, \\ 0, & u_{n-1} < 1/2. \end{cases} \tag{4}$$

Note that the relation (3) is the *doubling map* in disguise; $u_n = T(u_{n-1})$, where $T : u \mapsto 2u \pmod 1$.

The successive approximation algorithm as presented above provides an algorithmic circuit implementation that computes the bits b_n in the binary expansion of x while keeping all quantities (u_n and b_n) macroscopic and bounded, which means that these quantities can be held as realistic and measurable electric charges. However, despite the fact that base-2 representations, which essentially provide the optimal encoding in the rate-distortion sense, can be computed via a circuit, they are not the most popular choice of A/D conversion method. This is mainly because of robustness concerns: In practice, analog circuits are never precise, suffering from arithmetic errors (e.g., through nonlinearity) as well as from quantizer errors (e.g., threshold offset), simultaneously being subject to thermal noise. All relations hold only approximately, and therefore, all quantities are approximately equal to their theoretical values. In the case of the algorithm described in (3) and (4), this means that the approximation will exceed acceptable error bounds after only a finite (small) number of iterations because the dynamics of an expanding map has "sensitive dependence on initial conditions". This central problem in A/D conversion (as well as in D/A conversion) has led to the development of many alternative bit representations of numbers, as well as of signals, that have been adopted and/or implemented in circuit engineering, such as beta-representations and $\Sigma\Delta$ modulation.

From a theoretical point of view, the imprecision problem associated to the successive approximation algorithm is not a sufficient reason to discard the base-2 representation out of hand. After all, we do not have to use the specific algorithm in (3) and (4) to extract the bits, and conceivably there could be better, i.e., more resilient, algorithms to evaluate $b_n(x)$ for each x. However, the root of the real problem lies deeper: the bits in the base-2 representation are essentially uniquely determined, and are ultimately computed by a greedy method. Since $2^{-n} = 2^{-n-1} + 2^{-n-2} + \ldots$, there exists essentially no choice other than to set b_n according to (4). (A possible choice exists for x of the form $x = k2^{-m}$, with k an odd integer, and even then, only the bit b_m can be chosen freely: for the choice $b_m = 0$, one has $b_n = 1$ for all $n > m$; for the choice $b_m = 1$, one has $b_n = 0$ for all $n > m$.) It is clear that there is no way to recover from an erroneous bit computation: if the value 1 is assigned to b_n even though $x < x_{n-1} + 2^{-n}$, then this causes an "overshoot" from which there is no way to "back up" later. Similarly assigning the value 0 to b_n when $x > x_{n-1} + 2^{-n}$ implies a "fall-behind" from which there is no way to "catch up" later.

Due to this lack of robustness, the base-2 representation is not the preferred quantization method for A/D conversion. For similar reasons, it is also generally not the preferred method for D/A conversion. In practical settings, oversampled coarse quantization ($\Sigma\Delta$ modulation) is more popular [1]–[3], mostly due to its robustness achieved with the help of the redundant set of output codes that can represent each source value [4]. However, standard $\Sigma\Delta$ modulation is suboptimal as a quantization method (even though exponential accuracy in the bit rate can be achieved [5]).

A partial remedy comes with fractional base expansions, called β-representations [6]–[10]. Fix $1 < \beta < 2$. It is well known that every x in $[0, 1]$ (in fact, in $[0, (\beta - 1)^{-1}]$) can be represented by an infinite series

$$x = \sum_{n=1}^{\infty} b_n \beta^{-n}, \tag{5}$$

with an appropriate choice of the bit sequence (b_n). Such a sequence can be obtained via the following modified successive approximation algorithm: Define $u_n = \beta^n(x - x_n)$. Then the bits b_n obtained via the recursion

$$\begin{aligned} u_n &= \beta u_{n-1} - b_n, \\ b_n &= \begin{cases} 0, & u_{n-1} \le a, \\ 0 \text{ or } 1, & u_{n-1} \in (a, b), \\ 1, & u_{n-1} \ge b. \end{cases} \end{aligned} \tag{6}$$

satisfy (5) whenever $1/\beta \le a \le b \le 1/\beta(\beta-1)$. If $a = b = 1/\beta$, the above recursion is called the *greedy selection algorithm*; if $a = b = 1/\beta(\beta-1)$ it is the *lazy selection algorithm*. The intermediate cases correspond to what we call *cautious selection*. An immediate observation is that many distinct β-representations in the form (5) are now available. In fact, it is known that for any $1 < \beta < 2$, almost all numbers (in the Lebesgue measure sense) have uncountably many distinct β-representations [11]. Although N-bit truncated β-representations are only accurate to within $O(\beta^{-N})$, which is inferior to the accuracy of a base-2 representation, the redundancy of β-representation makes it an appealing alternative since it is possible to recover from (certain) incorrect bit computations. In particular, if these mistakes result from an unknown threshold offset in the quantizer, then it turns out that a *cautious* selection algorithm (rather than the greedy or the lazy selection algorithms) is robust provided a bound for the offset is known [7]. In other words, perfect encoding is possible with an imperfect (flaky) quantizer whose threshold value fluctuates in the interval $(1/\beta, 1/\beta(\beta - 1))$.

It is important to note that a circuit that computes truncated β-representations by implementing the recursion in (6) has two critical parameters: the quantizer threshold (which, in the case of the "flaky quantizer", is the pair (a, b)) and the multiplier β. As discussed above, β-encoders are robust with respect to the changes in the quantizer threshold. On the other hand, they are *not robust* with respect to the value of β (see Section II-C for a discussion). A partial remedy for this has been proposed in [8] which enables one to recover the value of β with the required accuracy, provided its value varies smoothly and slowly from one clock cycle to the next.

In this paper, we introduce a novel ADC which we call *the golden ratio encoder (GRE)*. GRE computes β-representations with respect to base $\beta = \phi = (1+\sqrt{5})/2$ via an implementation that is *not* a successive approximation algorithm. We show that GRE is robust with respect to its full parameter set while enjoying exponential accuracy in the bit rate. To our knowledge, GRE is the first example of such a scheme.

The outline of the paper is as follows: In Section II-A, we introduce notation and basic terminology. In Section II-B we review and formalize fundamental properties of *algorithmic converters*. Section II-C introduces a computational model for algorithmic converters, formally defines robustness for such converters, and reviews, within the established framework, the robustness properties of several algorithmic

ADCs in the literature. Section III is devoted to GRE and its detailed study. In particular, Sections III-A and III-B introduce the algorithm underlying GRE and establish basic approximation error estimates. In Section III-C, we give our main result, i.e., we prove that GRE is robust in its full parameter set. Sections III-D, III-E, and III-F discuss several additional properties of GRE. Finally, in Section IV, we comment on how one can construct "higher-order" versions of GRE.

II. Encoding, Algorithms and Robustness

A. Basic Notions for Encoding

We denote the space of analog objects to be quantized by X. More precisely, let X be a compact metric space with metric d_X. Typically d_X will be derived from a norm $\|\cdot\|$ defined in an ambient vector space, via $d_X(x,y) = \|x-y\|$. We say that E_N is an *N-bit encoder* for X if E_N maps X to $\{0,1\}^N$. An infinite family of encoders $\{E_N\}_1^\infty$ is said to be *progressive* if it is generated by a single map $E : X \mapsto \{0,1\}^{\mathbb{N}}$ such that for $x \in X$,

$$E_N(x) = (E(x)_0, E(x)_1, \ldots, E(x)_{N-1}). \tag{7}$$

In this case, we will refer to E as the generator, or sometimes simply as the encoder, a term which we will also use to refer to the family $\{E_N\}_1^\infty$.

We say that a map D_N is a *decoder* for E_N if D_N maps the range of E_N to some subset of X. Once X is an infinite set, E_N can never be one-to-one, hence analog-to-digital conversion is inherently lossy. We define the *distortion* of a given encoder-decoder pair (E_N, D_N) by

$$\delta_X(E_N, D_N) := \sup_{x\in X} \; d_X(x, D_N(E_N(x))), \tag{8}$$

and the *accuracy* of E_N by

$$\mathcal{A}(E_N) := \inf_{D_N:\{0,1\}^N \to X} \; \delta_X(E_N, D_N). \tag{9}$$

The compactness of X ensures that there exists a family of encoders $\{E_N\}_1^\infty$ and a corresponding family of decoders $\{D_N\}_1^\infty$ such that $\delta_X(E_N, D_N) \to 0$ as $N \to \infty$; i.e., all $x \in X$ can be recovered via the limit of $D_N(E_N(x))$. In this case, we say that the family of encoders $\{E_N\}_1^\infty$ is *invertible*. For a progressive family generated by $E : X \to \{0,1\}^{\mathbb{N}}$, this actually implies that E is one-to-one. Note, however, that the supremum over x in (8) imposes uniformity of approximation, which is slightly stronger than mere invertibility of E.

An important quality measure of an encoder is the rate at which $\mathcal{A}(E_N) \to 0$ as $N \to \infty$. There is a limit to this rate which is determined by the space X. (This rate is connected to the Kolmogorov ϵ-entropy of X, $\mathcal{H}_\epsilon(X)$, defined to be the base-2 logarithm of the smallest number k such that there exists an ϵ-net for X of cardinality k [12]. If we denote the map $\epsilon \mapsto \mathcal{H}_\epsilon(X)$ by φ, i.e., $\varphi(\epsilon) = \mathcal{H}_\epsilon(X)$, then the infimum of $\mathcal{A}(E_N)$ over all possible encoders E_N is roughly equal to $\varphi^{-1}(N)$.) Let us denote the number $\inf_{E_N:X\mapsto\{0,1\}^N} \mathcal{A}(E_N)$ by $\mathcal{A}_N(X)$. In general an optimal encoder may be impractical, and a compromise is sought between optimality and practicality. It is, however, desirable when designing an encoder that its performance is close to optimal. We say that a given family of encoders $\{E_N\}_1^\infty$ is *near-optimal for X*, if

$$\mathcal{A}(E_N) \le C \mathcal{A}_N(X), \tag{10}$$

where $C \ge 1$ is a constant independent of N. We will also say that a given family of decoders $\{D_N\}_1^\infty$ is near-optimal for $\{E_N\}_1^\infty$, if

$$\delta_X(E_N, D_N) \le C \mathcal{A}(E_N). \tag{11}$$

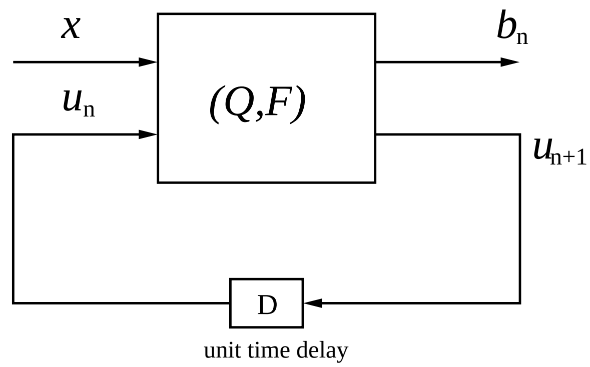

Fig. 1. The block diagram describing an algorithmic encoder.

An additional important performance criterion for an ADC is whether the encoder is robust against perturbations. Roughly speaking, this robustness corresponds to the requirement that for all encoders $\{\widetilde{E}_N\}$ that are small (and mostly unknown) perturbations of the original invertible family of encoders $\{E_N\}$, it is still true that $\mathcal{A}(\widetilde{E}_N) \to 0$, possibly at the same rate as E_N, using the same decoders. The magnitude of the perturbations, however, need not be measured using the Hamming metric on $\{0,1\}^X$ (e.g., in the form $\sup_{x\in X} d_H(E_N(x), \widetilde{E}_N(x))$). It is more realistic to consider perturbations that directly have to do with how these functions are computed in an actual circuit, i.e., using small building blocks (comparators, adders, etc.). It is often possible to associate a set of internal parameters with such building blocks, which could be used to define appropriate metrics for the perturbations affecting the encoder. From a mathematical point of view, all of these notions need to be defined carefully and precisely. For this purpose, we will focus on a special class of encoders, so-called *algorithmic converters*. We will further consider a computational model for algorithmic converters and formally define the notion of robustness for such converters.

B. Algorithmic Converters

By an *algorithmic converter*, we mean an encoder that can be implemented by carrying out an autonomous operation (the algorithm) iteratively to compute the bit representation of any input x. Many ADCs of practical interest, e.g., $\Sigma\Delta$ modulators, PCM, and beta-encoders, are algorithmic encoders. Figure 1 shows the block diagram of a generic algorithmic encoder.

Let $\mathcal{U}$ denote the set of possible "states" of the converter circuit that get updated after each iteration (clock cycle). More precisely, let

$$Q : X \times \mathcal{U} \mapsto \{0,1\}$$

be a "quantizer" and let

$$F : X \times \mathcal{U} \mapsto \mathcal{U}$$

be the map that determines the state of the circuit in the next clock cycle given its present state. After fixing the initial state of the circuit u_0, the circuit employs the pair of functions (Q,F) to carry out the following iteration:

$$\left\{ \begin{array}{rcl} b_n & = & Q(x,u_n) \\ u_{n+1} & = & F(x,u_n) \end{array} \right\}, \quad n = 0,1,\ldots \tag{12}$$

This procedure naturally defines a progressive family of encoders $\{E_N\}_1^\infty$ with the generator map E given by

$$E(x)_n := b_n, \quad n = 0, 1, \ldots$$

We will write $\mathrm{AE}(Q, F)$ to refer to the algorithmic converter defined by the pair (Q, F) and the implicit initial condition u_0. If the generator E is invertible (on $E(X)$), then we say that the converter is invertible as well.

Definition 1 (1-bit quantizer)**.** We define the 1-bit quantizer with threshold value τ to be the function

$$q_\tau(u) := \begin{cases} 0, & u < \tau, \\ 1, & u \geq \tau. \end{cases} \tag{13}$$

Examples of algorithmic encoders:

1. *Successive approximation algorithm for PCM.* The successive approximation algorithm sets $u_0 = x \in [0, 2]$, and computes the bits b_n, $n = 0, 1, \ldots$, in the binary expansion $x = \sum_0^\infty b_n 2^{-n}$ via the iteration
$$\left\{ \begin{array}{rcl} b_n & = & q_1(u_n) \\ u_{n+1} & = & 2(u_n - b_n) \end{array} \right\}, \quad n = 0, 1, \ldots \tag{14}$$
Defining $Q(x, u) = q_1(u)$ and $F(x, u) = 2(u - q_1(u))$, we obtain an invertible algorithmic converter for $X = [0, 2]$. A priori we can set $\mathcal{U} = \mathbb{R}$, though it is easily seen that all the u_n remain in $[0, 2]$.
2. *Beta-encoders with successive approximation implementation* [7]. Let $\beta \in (1, 2)$. A β-representation of x is an expansion of the form $x = \sum_0^\infty b_n \beta^{-n}$, where $b_n \in \{0, 1\}$. Unlike a base-2 representation, almost every x has infinitely many such representations. One class of expansions is found via the iteration
$$\left\{ \begin{array}{rcl} b_n & = & q_\tau(u_n) \\ u_{n+1} & = & \beta(u_n - b_n) \end{array} \right\}, \quad n = 0, 1, \ldots, \tag{15}$$
where $u_0 = x \in [0, \beta/(\beta - 1)]$. The case $\tau = 1$ corresponds to the "greedy" expansion, and the case $\tau = 1/(\beta - 1)$ to the "lazy" expansion. All values of $\tau \in [1, 1/(\beta - 1)]$ are admissible in the sense that the u_n remain bounded, which guarantees the validity of the inversion formula. Therefore the maps $Q(x, u) = q_\tau(u)$, and $F(x, u) = \beta(u - q_\tau(u))$ define an invertible algorithmic encoder for $X = [0, \beta]$. It can be checked that all u_n remain in a bounded interval $\mathcal{U}$ independently of τ.
3. *First-order* $\Sigma\Delta$ *with constant input.* Let $x \in [0, 1]$. The first-order $\Sigma\Delta$ (Sigma-Delta) ADC sets the value of $u_0 \in [0, 1]$ arbitrarily and runs the following iteration:
$$\left\{ \begin{array}{rcl} b_n & = & q_1(u_n + x) \\ u_{n+1} & = & u_n + x - b_n \end{array} \right\}, \quad n = 0, 1, \ldots \tag{16}$$
It is easy to show that $u_n \in [0, 1]$ for all n, and the boundedness of u_n implies
$$x = \lim_{N \to \infty} \frac{1}{N} \sum_{n=1}^{N} b_n, \tag{17}$$
that is, the corresponding generator map E is invertible. For this invertible algorithmic encoder, we have $X = [0, 1]$, $\mathcal{U} = [0, 1]$, $Q(x, u) = q_1(u + x)$, and $F(x, u) = u + x - q_1(u + x)$.
4. *kth-order* $\Sigma\Delta$ *with constant input.* Let Δ be the forward difference operator defined by $(\Delta u)_n = u_{n+1} - u_n$. A kth-order $\Sigma\Delta$ ADC generalizes the scheme in Example 3 by replacing the first-order difference equation in (16) with
$$(\Delta^k u)_n = x - b_n, \quad n = 0, 1, \ldots, \tag{18}$$

which can be rewritten as

$$u_{n+k} = \sum_{j=0}^{k-1} a_j^{(k)} u_{n+j} + x - b_n, \quad n = 0, 1, \ldots, \tag{19}$$

where $a_j^{(k)} = (-1)^{k-1-j}\binom{k}{j}$. Here $b_n \in \{0, 1\}$ is computed by a function Q of x and the previous k state variables $u_n, \ldots, u_{n+k-1}$, which must guarantee that the u_n remain in some bounded interval for all n, provided that the initial conditions $u_0, \ldots, u_{k-1}$ are picked appropriately. If we define the vector state variable $\mathbf{u}_n = \left[u_n, \cdots, u_{n+k-1}\right]^\top$, then it is apparent that we can rewrite the above equations in the form

$$\left\{ \begin{array}{rcl} b_n & = & Q(x, \mathbf{u}_n) \\ \mathbf{u}_{n+1} & = & \mathbf{L}_k \mathbf{u}_n + x\mathbf{e} - b_n \mathbf{e} \end{array} \right\}, \quad n = 0, 1, \ldots, \tag{20}$$

where $\mathbf{L}_k$ is the $k \times k$ companion matrix defined by

$$\mathbf{L}_k := \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & & \ddots & \vdots \\ 0 & 0 & & \cdots & 1 \\ a_0^{(k)} & a_1^{(k)} & & \cdots & a_{k-1}^{(k)} \end{bmatrix}, \tag{21}$$

and $\mathbf{e} = \left[0, \cdots, 0, 1\right]^\top$. If Q is such that there exists a set $\mathcal{U} \subset \mathbb{R}^k$ with the property $\mathbf{u} \in \mathcal{U}$ implies $F(x, \mathbf{u}) := \mathbf{L}_k \mathbf{u} + (x - Q(x, \mathbf{u}))\mathbf{e} \in \mathcal{U}$ for each $x \in X$, then it is guaranteed that the u_n are bounded, i.e., the scheme is *stable*. Note that any stable kth order scheme is also a first order scheme with respect to the state variable $(\Delta^{k-1}u)_n$. This implies that the inversion formula (17) holds, and therefore (20) defines an invertible algorithmic encoder. Stable $\Sigma\Delta$ schemes of arbitrary order k have been devised in [13] and in [5]. For these schemes X is a proper subinterval of $[0, 1]$.

C. A Computational Model for Algorithmic Encoders and Formal Robustness

Next, we focus on a crucial property that is required for any ADC to be implementable in practice. As mentioned before, any ADC must perform certain arithmetic (computational) operations (e.g., addition, multiplication), and Boolean operations (e.g., comparison of some analog quantities with predetermined reference values). In the analog world, these operations cannot be done with infinite precision due to physical limitations. Therefore, the algorithm underlying a practical ADC needs to be robust with respect to implementation imperfections.

In this section, we shall describe a computational model for algorithmic encoders that includes all the examples discussed above and provides us with a formal framework in which to investigate others. This model will also allow us to formally define robustness for this class of encoders, and make comparisons with the state-of-the-art converters.

Directed Acyclic Graph Model: Recall (12) which (along with Figure 1) describes one cycle of an algorithmic encoder. So far, the pair (Q, F) of maps has been defined in a very general context and could have arbitrary complexity. In this section, we would like to propose a more realistic computational model for these maps. Our first assumption will be that $X \subset \mathbb{R}$ and $\mathcal{U} \subset \mathbb{R}^d$.

A *directed acyclic graph* (DAG) is a directed graph with no directed cycles. In a DAG, a *source* is a node (vertex) that has no incoming edges. Similarly, a *sink* is a node with no outgoing edges. Every DAG has a set of sources and a set of sinks, and every directed path starts from a source and ends at a sink.

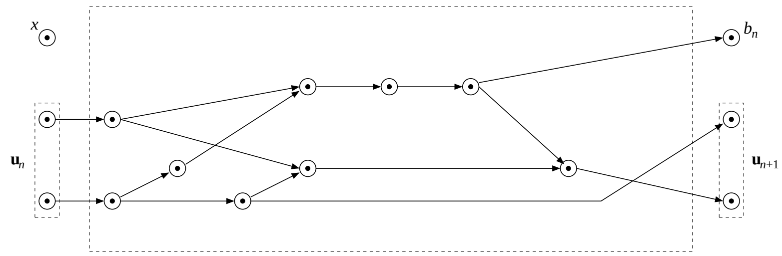

Fig. 2. A sample DAG model for the function pair (Q, F).

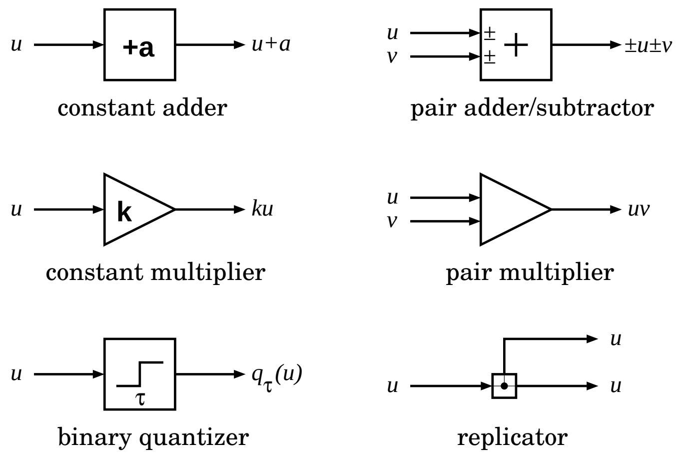

Fig. 3. A representative class of components in an algorithmic encoder.

Our DAG model will always have $d+1$ source nodes that correspond to x and the d-dimensional vector $\mathbf{u}_n$, and $d+1$ sink nodes that correspond to b_n and the d-dimensional vector $\mathbf{u}_{n+1}$. This is illustrated in Figure 2 which depicts the DAG model of an implementation of the Golden Ratio Encoder (see Section III and Figure 4). Note that the feedback loop from Figure 1 is not a part of the DAG model, therefore the nodes associated to x and $\mathbf{u}_n$ have no incoming edges in Figure 2. Similarly the nodes associated to b_n and $\mathbf{u}_{n+1}$ have no outgoing edges. In addition, the node for x, even when x is not actually used as an input to (Q, F), will be considered only as a source (and not as a sink).

In our DAG model, a node which is not a source or a sink will be associated with a *component*, i.e., a computational device with inputs and outputs. Mathematically, a component is simply a function (of smaller "complexity") selected from a given fixed class. In the setting of this paper, we shall be concerned with only a restricted class of basic components: constant adder, pair adder/subtractor, constant multiplier, pair multiplier, binary quantizer, and replicator. These are depicted in Figure 3 along with their defining relations, except for the binary quantizer, which was defined in (13). Note that a component and the node at which it is placed should be consistent, i.e., the number of incoming edges must match the number of inputs of the component, and the number of outgoing edges must match the number of outputs of the component.

When the DAG model for an algorithmic encoder defined by the pair (Q, F) employs the ordered k-tuple of components $\mathcal{C} = (C_1, \ldots, C_k)$ (including repetitions), we will denote this encoder by $\mathrm{AE}(Q, F, \mathcal{C})$.

Robustness: We would like to say that an invertible algorithmic encoder $\mathrm{AE}(Q, F)$ is *robust* if $\mathrm{AE}(\widetilde{Q}, \widetilde{F})$ is also invertible for any $(\widetilde{Q}, \widetilde{F})$ in a given neighborhood of (Q, F) and if the generator

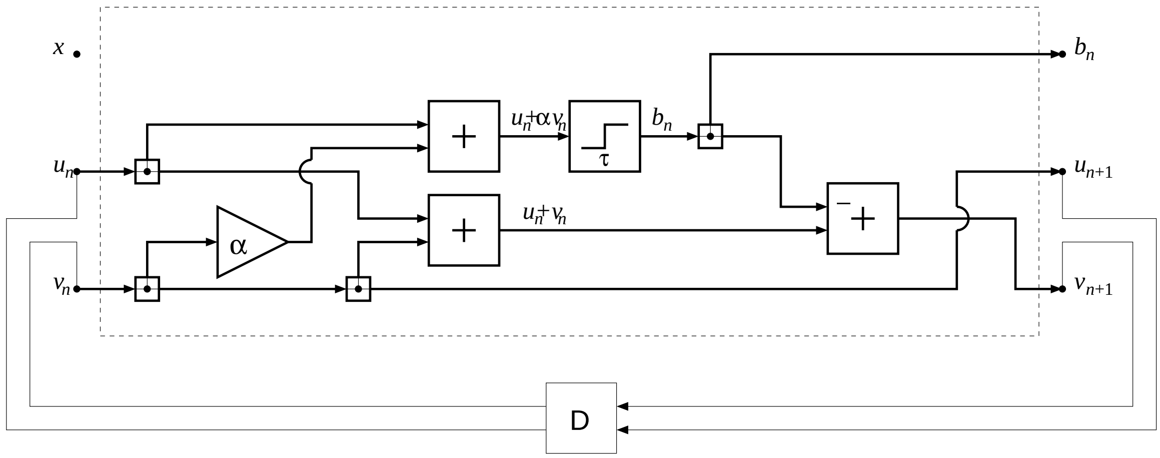

Fig. 4. Block diagram of the Golden Ratio Encoder (Section III) corresponding to the DAG model in Figure 2.

E of $\mathrm{AE}(Q, F)$ has an inverse D defined on $\{0,1\}^{\mathbb{N}}$ that is also an inverse for the generator $\widetilde{E}$ of $\mathrm{AE}(\widetilde{Q}, \widetilde{F})$. To make sense of this definition, we need to define the meaning of "neighborhood" of the pair (Q, F).

Let us consider an algorithmic encoder $\mathrm{AE}(Q, F, \mathcal{C})$ as described above. Each component C_j in $\mathcal{C}$ may incorporate a vector λ_j of parameters (allowing for the possibility of the null vector). Let $\lambda = (\lambda_1, \dots, \lambda_k)$ be the aggregate parameter vector of such an encoder. We will then denote $\mathcal{C}$ by $\mathcal{C}(\lambda)$.

Let Λ be a space of parameters for a given algorithmic encoder and consider a metric ρ_Λ on Λ. We now say that $\mathrm{AE}(Q, F, \mathcal{C}(\lambda_0))$ is *robust* if there exists a $\delta > 0$ such that $\mathrm{AE}(Q, F, \mathcal{C}(\lambda))$ is invertible for all λ with $\rho_\Lambda(\lambda, \lambda_0) \le \delta$ with a common inverse. Similarly, we say that $\mathrm{AE}(Q, F, \mathcal{C}(\lambda_0))$ is *robust in approximation* if there exists a $\delta > 0$ and a family $\{D_N\}_1^\infty$ such that

$$\delta_X(\widetilde{E}_N, D_N) \le C\mathcal{A}(E_N)$$

for all $\widetilde{E}_N = \widetilde{E}_N(Q, F, \lambda)$ with $\rho_\Lambda(\lambda, \lambda_0) \le \delta$, where $E_N = E_N(Q, F, \lambda_0)$.

From a practical point of view, if an algorithmic encoder $\mathrm{AE}(Q, F, \mathcal{C}(\lambda_0))$ is robust, and is implemented with a perturbed parameter λ instead of the intended λ_0, one can still obtain arbitrarily good approximations of the original analog object without knowing the actual value of λ. However, here we still assume that the "perturbed" encoder is algorithmic, i.e., we use the same perturbed parameter value λ at each clock cycle. In practice, however, many circuit components are "flaky", that is, the associated parameters vary at each clock cycle. We can describe such an encoder again by (12), however we need to replace (Q, F) by $(Q_{\lambda^n}, F_{\lambda^n})$, where $\{\lambda^n\}_1^\infty$ is the sequence of the associated parameter vectors. With an abuse of notation, we denote such an encoder by $\mathrm{AE}(Q_{\lambda^n}, F_{\lambda^n})$ (even though this is clearly not an algorithmic encoder in the sense of Section II-B). Note that if $\{\lambda^n\}_1^\infty = \lambda_0^{\mathbb{N}}$, i.e., $\lambda^n = \lambda_0$ for all n, the corresponding encoder is $\mathrm{AE}(Q, F, \mathcal{C}(\lambda_0))$. We now say that $\mathrm{AE}(Q, F, \mathcal{C}(\lambda_0))$ is *strongly robust* if there exists $\delta > 0$ such that $\mathrm{AE}(Q_{\lambda^n}, F_{\lambda^n})$ is invertible for all $\{\lambda^n\}_1^\infty$ with $\rho_{\Lambda^{\mathbb{N}}}(\{\lambda^n\}_1^\infty, \lambda_0^{\mathbb{N}}) \le \delta$ with a common inverse. Here, $\Lambda^{\mathbb{N}}$ is the space of parameter sequences for a given converter, and $\rho_{\Lambda^{\mathbb{N}}}$ is an appropriate metric on $\Lambda^{\mathbb{N}}$. Similarly, we call an algorithmic encoder *strongly robust in approximation* if there exists $\delta > 0$ and a family $\{D_N\}_1^\infty$ such that

$$\delta_X(\widetilde{E}_N^{\mathrm{f}}, D_N) \le C\mathcal{A}(E_N)$$

for all flaky encoders $\widetilde{E}_N^{\mathrm{f}} = \widetilde{E}_N^{\mathrm{f}}(Q_{\lambda_n}, F_{\lambda_n})$ with $\rho_N(\{\lambda_n\}_1^N, \lambda_{0,N}) < \delta$ where ρ_N is the metric on Λ^N obtained by restricting $\rho_{\Lambda^{\mathbb{N}}}$ (assuming such a restriction makes sense), $\lambda_{0,N}$ is the N-tuple whose components are each λ_0, and $E_N = E_N(Q, F, \lambda_0)$.

Examples of Section II-B: Let us consider the examples of algorithmic encoders given in section II-B. The successive approximation algorithm is a special case of the beta encoder for $\beta = 2$ and $\tau = 1$. As we mentioned before, there is no unique way to implement an algorithmic encoder using a given class of components. For example, given the set of rules set forth earlier, multiplication by 2 could conceivably be implemented as a replicator followed by a pair adder (though a circuit engineer would probably not approve of this attempt). It is not our goal here to analyze whether a given DAG model is realizable in analog hardware, but to find out whether it is robust given its set of parameters.

The first order $\Sigma\Delta$ quantizer is perhaps the encoder with the simplest DAG model; its only parametric component is the binary quantizer, characterized by $\tau = 1$. For the beta encoder it is straightforward to write down a DAG model that incorporates only two parametric components: a constant multiplier and a binary quantizer. This model is thus characterized by the vector parameter $\lambda = (\beta, \tau)$. The successive approximation encoder corresponds to the special case $\lambda_0 = (2, 1)$. If the constant multiplier is avoided via the use of a replicator and adder, as described above, then this encoder would be characterized by $\tau = 1$, corresponding to the quantizer threshold.

These three models ($\Sigma\Delta$ quantizer, beta encoder, and successive approximation) have been analyzed in [7] and the following statements hold:

1) The successive approximation encoder is not robust for $\lambda_0 = (2, 1)$ (with respect to the Euclidean metric on $\Lambda = \mathbb{R}^2$). The implementation of successive approximation that avoids the constant multiplier as described above, and thus characterized by $\tau = 1$ is not robust with respect to τ.
2) The first order $\Sigma\Delta$ quantizer is strongly robust in approximation for the parameter value $\tau_0 = 1$ (and in fact for any other value for τ_0). However, $\mathcal{A}(E_N) = \Theta(N^{-1})$ for $X = [0, 1]$.
3) The beta encoder is strongly robust in approximation for a range of values of τ_0, when $\beta = \beta_0$ is fixed. Technically, this can be achieved by a metric which is the sum of the discrete metric on the first coordinate and the Euclidean metric on the second coordinate between any two vectors $\lambda = (\beta, \tau)$, and $\lambda_0 = (\beta_0, \tau_0)$. This way we ensure that the first coordinate remains constant on a small neighborhood of any parameter vector. Here $\mathcal{A}(E_N) = \Theta(\beta^{-N})$ for $X = [0, 1]$. This choice of the metric however is not necessarily realistic.
4) The beta encoder is not robust with respect to the Euclidean metric on the parameter vector space $\Lambda = \mathbb{R}^2$ in which both β and τ vary. In fact, this is the case even if we consider changes in β only: let E and $\widetilde{E}$ be the generators of beta encoders with parameters (β, τ) and $(\beta + \delta, \tau)$, respectively. Then E and $\widetilde{E}$ do not have a common inverse for any $\delta \neq 0$. To see this, let $\{b_n\}_1^\infty = E(x)$. Then a simple calculation shows
$$\Big|\sum_1^\infty b_n((\beta+\delta)^{-n} - \beta^{-n})\Big| = \Omega(\frac{k}{2^{k+1}}\delta)$$
where $k = \min\{n : b_n \neq 0\} < \infty$ for any $x \neq 0$. Consequently, $\widetilde{E}^{-1} \neq E^{-1}$.
This shows that to decode a beta-encoded bit-stream, one needs to know the value of β at least within the desired precision level. This problem was addressed in [8], where a method was proposed for embedding the value of β in the encoded bit-stream in such a way that one can recover an estimate for β (in the digital domain) with exponential precision. Beta encoders with the modifications of [8] are effectively robust in approximation (the inverse of the generator of the perturbed encoder is different from the inverse of the intended encoder, however it can be precisely computed). Still,

even with these modifications, the corresponding encoders are *not strongly robust* with respect to the parameter β.

5) The stable $\Sigma\Delta$ schemes of arbitrary order k that were designed by Daubechies and DeVore, [13], are strongly robust in approximation with respect to their parameter sets. Also, a wide family of second-order $\Sigma\Delta$ schemes, as discussed in [14], are strongly robust in approximation. On the other hand, the family of exponentially accurate one-bit $\Sigma\Delta$ schemes reported in [5] are not robust because each scheme in this family employs a vector of constant multipliers which, when perturbed arbitrarily, result in bit sequences that provide no guarantee of even mere invertibility (using the original decoder). The only reconstruction accuracy guarantee is of Lipshitz type, i.e. the error of reconstruction is controlled by a constant times the parameter distortion.

Neither of these cases result in an algorithmic encoder with a DAG model that is robust in its full set of parameters *and* achieves exponential accuracy. To the best of our knowledge, our discussion in the next section provides the first example of an encoder that is (strongly) robust in approximation while simultaneously achieving exponential accuracy.

III. The Golden Ratio Encoder

In this section, we introduce a Nyquist rate ADC, *the golden ratio encoder* (GRE), that bypasses the robustness concerns mentioned above while still enjoying exponential accuracy in the bit-rate. In particular, GRE is an algorithmic encoder that is strongly robust in approximation with respect to its full set of parameters, and its accuracy is exponential. The DAG model and block diagram of the GRE are given in Figures 2 and 4, respectively.

A. The scheme

We start by describing the recursion underlying the GRE. To quantize a given real number $x \in X = [0,2]$, we set $u_0 = x$ and $u_1 = 0$, and run the iteration process

$$\begin{aligned} u_{n+2} &= u_{n+1} + u_n - b_n, \\ b_n &= Q(u_n, u_{n+1}). \end{aligned} \tag{22}$$

Here, $Q:\ \mathbb{R}^2 \mapsto \{0,1\}$ is a quantizer to be specified later. Note that (22) describes a piecewise affine discrete dynamical system on $\mathbb{R}^2$. More precisely, define

$$T_Q : \begin{bmatrix} u \\ v \end{bmatrix} \mapsto \underbrace{\begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}}_{A} \begin{bmatrix} u \\ v \end{bmatrix} - Q(u,v) \begin{bmatrix} 0 \\ 1 \end{bmatrix}. \tag{23}$$

Then we can rewrite (22) as

$$\begin{bmatrix} u_{n+1} \\ u_{n+2} \end{bmatrix} = T_Q \begin{bmatrix} u_n \\ u_{n+1} \end{bmatrix}. \tag{24}$$

Now, let $X = [0,2)$, $\mathcal{U} \subset \mathbb{R}^2$, and suppose Q is a quantizer on $X \times \mathcal{U}$. The formulation in (24) shows that GRE is an algorithmic encoder, $\mathrm{AE}(Q, F_{\mathrm{GRE}})$, where $F_{\mathrm{GRE}}(x, \mathbf{u}) := A\mathbf{u} - Q(\mathbf{u})$ for $\mathbf{u} \in \mathcal{U}$. Next, we show that GRE is invertible by establishing that, if Q is chosen appropriately, the sequence $b = (b_n)$ obtained via (22) gives a beta representation of x with $\beta = \phi = (1+\sqrt{5})/2$.

B. Approximation Error and Accuracy

Proposition 1. *Let $x \in [0,2)$ and suppose b_n are generated via (22) with $u_0 = x$ and $u_1 = 0$. Then*

$$x = \sum_{n=0}^{\infty} b_n \phi^{-n}$$

if and only if the state sequence $(u_n)_0^\infty$ is bounded. Here ϕ is the golden mean.

Proof: Note that

$$\begin{aligned}
\sum_{n=0}^{N-1} b_n \phi^{-n} &= \sum_{n=0}^{N-1} (u_n + u_{n+1} - u_{n+2})\phi^{-n} \\
&= \sum_{n=2}^{N-1} u_n \phi^{-n}(1+\phi-\phi^2) + u_1 + u_N\phi^{-N+1} + u_0 + u_1\phi^{-1} - u_N\phi^{-N+2} - u_{N+1}\phi^{-N+1} \\
&= x + \phi^{-N+1}((1-\phi)u_N - u_{N+1}), \\
&= x - \phi^{-N}(u_N + \phi\, u_{N+1}). \qquad (25)
\end{aligned}$$

where the third equality follows from $1+\phi-\phi^2 = 0$, and the last equality is obtained by setting $u_0 = x$ and $u_1 = 0$. Defining the N-term approximation error to be

$$e_N(x) := x - \sum_{n=0}^{N-1} b_n \phi^{-n} = \phi^{-N}(u_N + \phi\, u_{N+1})$$

it follows that

$$\lim_{N\to\infty} e_N(x) = 0,$$

provided

$$\lim_{n\to\infty} u_n \phi^{-n} = 0. \qquad (26)$$

Clearly, (26) is satisfied if there is a constant C, independent of n, such that $|u_n| \le C$, . Conversely, suppose (26) holds, and assume that (u_n) is unbounded. Let N be the smallest integer for which $|u_N| > C'\phi^3$ for some $C' > 1$. Without loss of generality, assume $u_N > 0$ (the argument below, with simple modifications, applies if $u_N < 0$). Then, using (22) repeatedly, one can show that

$$u_{N+k} > C'\phi^3 f_{k-1} - f_{k+2} + 1$$

where f_k is the kth Fibonacci number. Finally, using

$$f_k = \frac{\phi^k - (1-\phi)^k}{\sqrt{5}}$$

(which is known as Binet's formula, e.g., see [15]) we conclude that $u_{N+k} \ge C''\phi^k$ for every positive integer k, showing that (26) does not hold if the sequence (u_n) is not bounded.

■

Note that the proof of Proposition 1 also shows that the N-term approximation error $e_N(x)$ decays exponentially in N if and only if the state sequence (u_n), obtained when encoding x, remains bounded by a constant (which may depend on x). We will say that a GRE is *stable* on X if the constant C in Proposition 1 is independent of $x \in X$, i.e., if the state sequences u with $u_0 = x$ and $u_1 = 0$ are bounded by a constant uniformly in x. In this case, the following proposition holds.

Proposition 2. *Let E^{GRE} be the generator of the GRE, described by* (22). *If the GRE is stable on X, it is exponentially accurate on X. In particular, $\mathcal{A}(E_N^{GRE}) = \Theta(\phi^{-N})$.*

Next, we investigate quantizers Q that generate stable encoders.

C. Stability and robustness with respect to imperfect quantizers

To establish stability, we will show that for several choices of the quantizer Q, there exist bounded positively invariant sets R_Q such that $T_Q(R_Q) \subseteq R_Q$. We will frequently use the basic 1-bit quantizer,

$$q_1(u) = \begin{cases} 0, & \text{if } u < 1, \\ 1, & \text{if } u \geq 1. \end{cases}$$

Most practical quantizers are implemented using arithmetic operations and q_1. One class that we will consider is given by

$$Q_\alpha(u, v) := q_1(u + \alpha v) = \begin{cases} 0, & \text{if } u + \alpha v < 1, \\ 1, & \text{if } u + \alpha v \geq 1. \end{cases} \tag{27}$$

Note that in the DAG model of GRE, the circuit components that implement Q_α incorporate a parameter vector $\lambda = (\tau, \alpha) = (1, \alpha)$. Here, τ is the threshold of the 1-bit basic quantizer, and α is the gain factor of the multiplier that maps v to αv. One of our main goals in this paper is to prove that GRE, with the implementation depicted in Figure 4, is strongly robust in approximation with respect to its full set of parameters. That is, we shall allow the parameter values to change at each clock cycle (within some margin). Such changes in parameter τ can be incorporated to the recursion (22) by allowing the quantizer Q_α to be flaky. More precisely, for $\nu_1 \leq \nu_2$, let q^{ν_1,ν_2} be the flaky version of q_τ defined by

$$q^{\nu_1,\nu_2}(u) := \begin{cases} 0, & \text{if } u < \nu_1, \\ 1, & \text{if } u \geq \nu_2, \\ 0 \text{ or } 1, & \text{if } \nu_1 \leq u < \nu_2. \end{cases}$$

We shall denote by $Q_\alpha^{\nu_1,\nu_2}$ the *flaky* version of Q_α, which is now

$$Q_\alpha^{\nu_1,\nu_2}(u, v) := q^{\nu_1,\nu_2}(u + \alpha v).$$

Note that (22), implemented with $Q = Q_\alpha^{\nu_1,\nu_2}$, does not generate an algorithmic encoder. At each clock cycle, the action of $Q_\alpha^{\nu_1,\nu_2}$ is identical to the action of $Q_\alpha^{\tau_n,\tau_n}(u, v) := q_{\tau_n}(u+\alpha v)$ for some $\tau_n \in [\nu_1, \nu_2)$. In this case, using the notation introduced before, (22) generates AE$(F_{\text{GRE}}^{\tau_n}, Q_\alpha^{\tau_n,\tau_n})$. We will refer to this encoder family as *GRE with flaky quantizer*.

1) A stable GRE with no multipliers: the case $\alpha = 1$: We now set $\alpha = 1$ in (27) and show that the GRE implemented with Q_1 is stable, and thus generates an encoder family with exponential accuracy (by Proposition 2). Note that in this case, the recursion relation (22) does not employ any multipliers (with gains different from unity). In other words, the associated DAG model does not contain any "constant multiplier" component.

Proposition 3. *Consider T_{Q_1}, defined as in* (23). *Then $R_{Q_1} := [0, 1]^2$ satisfies*

$$T_{Q_1}(R_{Q_1}) = R_{Q_1}.$$

Proof: By induction. It is easier to see this on the equivalent recursion (22). Suppose $(u_n, u_{n+1}) \in R_{Q_1}$, i.e., u_n and u_{n+1} are in $[0, 1]$. Then $u_{n+2} = u_n + u_{n+1} - q_1(u_n + u_{n+1})$ is in $[0, 1]$, which concludes the proof. ∎

It follows from the above proposition that the GRE implemented with Q_1 is stable whenever the initial state $(x, 0) \in R_{Q_1}$, i.e., $x \in [0, 1]$. In fact, one can make a stronger statement because a longer chunk of the positive real axis is in the basin of attraction of the map T_{Q_1}.

Proposition 4. *The GRE implemented with Q_1 is stable on $[0, 1+\phi)$, where ϕ is the golden mean. More precisely, for any $x \in [0, 1+\phi)$, there exists a positive integer N_x such that $u_n \in [0, 1]$ for all $n \geq N_x$.*

Corollary 5. *Let $x \in [0, 1+\phi)$, set $u_0 = x$, $u_1 = 0$, and generate the bit sequence (b_n) by running the recursion (22) with $Q = Q_1$. Then, for $N \geq N_x$,*

$$|x - \sum_{n=0}^{N-1} b_n \phi^{-n}| \leq \phi^{-N+2}.$$

One can choose N_x uniformly in x in any closed subset of $[0, 1+\phi)$. In particular, $N_x = 0$ for all $x \in [0, 1]$.

Remarks.

1) The proofs of Proposition 4 and Corollary 5 follow trivially from Proposition 3 when $x \in [0, 1]$. It is also easy to see that $N_x = 1$ for $x \in (1, 2]$, i.e., after one iteration the state variables u_1 and u_2 are both in $[0, 1]$. Furthermore, it can be shown that

$$0 < x < 1 + \frac{f_{N+2}}{f_{N+1}} - \frac{1}{f_{N+1}} \quad \Rightarrow \quad N_x \leq N,$$

where f_n denotes the nth Fibonacci number. We do not include the proof of this last statement here, as the argument is somewhat long, and the result is not crucial from the viewpoint of this paper.
2) In this case, i.e., when $\alpha = 1$, the encoder is *not robust* with respect to quantizer imperfections. More precisely, if we replace Q_1 with $Q_1^{\nu_1,\nu_2}$, with $\nu_1, \nu_2 \in (1-\delta, 1+\delta)$, then $|u_n|$ can grow regardless of how small $\delta > 0$ is. This is a result of the mixing properties of the piecewise affine map associated with GRE. In particular, one can show that $[0, 1] \times \{0\} \subset \cup_n S_n$, where $S_n \subset [0, 1]^2$ is the set of points whose nth forward image is outside the unit square. Figure 5 shows the fraction of 10,000 randomly chosen x-values for which $|u_N| > 1$ as a function of N. In fact, suppose $u_0 = x$ with $x \in [0, 1]$ and $u_1 = 0$. Then, the probability that u_N is outside of $[0, 1]$ is $O(N\delta^2 + \phi^{-N})$, which is superior to the case with PCM where the corresponding probability scales like $N\delta$. This observation suggests that "GRE with no multipliers", with its simply-implementable nature, could still be useful in applications where high fidelity is not required. We shall discuss this in more detail elsewhere.

2) Other values of α: stable and robust GRE: In the previous subsection, we saw that GRE, implemented with Q_α, $\alpha = 1$, is stable on $[0, 1+\phi)$, and thus enjoys exponential accuracy; unfortunately, the resulting algorithmic encoder is not robust. In this subsection, we show that there is a wide parameter range for ν_1, ν_2 and α for which the map T_Q with $Q = Q_\alpha^{\nu_1,\nu_2}$ has positively invariant sets R that do not depend on the particular values of ν_i and α. Using such a result, we then conclude that, with the appropriate choice of parameters, the associated GRE is strongly robust in approximation with respect to τ, the quantizer threshold, and α, the multiplier needed to implement Q_α. We also show that the invariant sets R can be constructed to have the additional property that for a small value of $\mu > 0$ one has $T_Q(R) + B_\mu(0) \subset R$, where $B_\mu(0)$ denotes the open ball around 0 with radius μ. In this case, R depends on μ. Consequently, even if the image of any state (u, v) is perturbed within a radius of μ, it still remains within R. Hence we also achieve stability under small additive noise or arithmetic errors.

Lemma 6. *Let $0 \leq \mu \leq (2\phi^2\sqrt{\phi+2})^{-1} \approx 0.1004$. There is a set $R = R(\mu)$, explicitly given in (29), and a wide range for parameters ν_1, ν_2, and α such that $T_{Q_\alpha^{\nu_1,\nu_2}}(R) \subseteq R + B_\mu(0)$.*

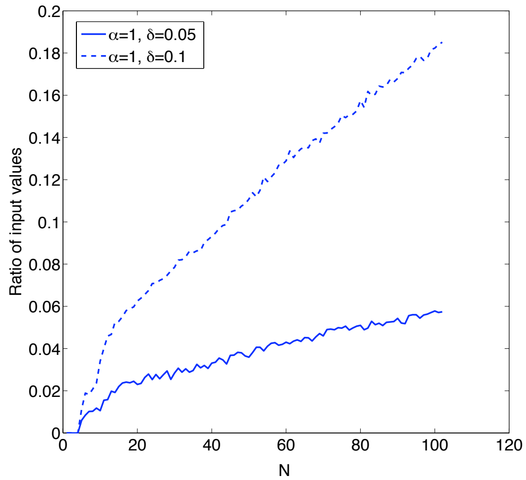

Fig. 5. We choose 10,000 values for x, uniformly distributed in $[0,1]$ and run the recursion (22) with $Q = Q_1^{\nu_1,\nu_2}$ with $\nu_i \in (1-\delta, 1+\delta)$. The graphs show the ratio of the input values for which $|u_N| > 1$ vs. N for $\delta = 0.05$ (solid) and $\delta = 0.1$ (dashed).

Proof: Our proof is constructive. In particular, for a given μ, we obtain a parametrization for R, which turns out to be a rectangular set. The corresponding ranges for ν_1, ν_2, and α are also obtained implicitly below. We will give explicit ranges for these parameters later in the text.

Our construction of the set R, which only depends on μ, is best explained with a figure. Consider the two rectangles $A_1B_1C_1D_1$ and $A_2B_2C_2D_2$ in Figure 6. These rectangles are designed to be such that their respective images under the linear map T_1, and the affine map T_2, defined by

$$T_1 : \begin{bmatrix} u \\ v \end{bmatrix} \mapsto \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}, \quad \text{and} \quad T_2 : \begin{bmatrix} u \\ v \end{bmatrix} \mapsto \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix} \begin{bmatrix} u \\ v \end{bmatrix} - \begin{bmatrix} 0 \\ 1 \end{bmatrix}, \tag{28}$$

are the same, i.e., $T_1(A_1B_1C_1D_1) = ABCD = T_2(A_2B_2C_2D_2)$.

The rectangle $ABCD$ is such that its μ-neighborhood is contained within the union of $A_1B_1C_1D_1$ and $A_2B_2C_2D_2$. This guards against additive noise. The fact that the rectangles $A_1B_1C_1D_1$ and $A_2B_2C_2D_2$ overlap (the shaded region) allows for the use of a flaky quantizer. Call this region F. As long as the region in which the quantizer operates in the flaky mode is a subset of F, and $T_Q = T_1$ on $A_1B_1C_1D_1 \setminus F$ and $T_Q = T_2$ on $A_2B_2C_2D_2 \setminus F$, it follows that $T_Q(A_1B_1C_1D_1 \cup A_2B_2C_2D_2) \subset ABCD$. It is then most convenient to choose $R = R(\mu) = A_1^{\#}B_2C_2^{\#}D_1$ and we clearly have $T_Q(R) + B_\mu(0) \subset R$. Note that if $Q = Q_\alpha^{\nu_1,\nu_2}$, any choice ν_1, ν_2, and α for which the graph of $v = -\frac{1}{\alpha}u + \frac{\nu}{\alpha}$ remains inside the shaded region F for $\nu_1 \le \nu \le \nu_2$ will ensure $T_Q(R) \subseteq R + B_\mu(0)$.

Next, we check the existence of at least one solution to this setup. This can be done easily in terms the parameters defined in the figure. First note that the linear map T_1 has the eigenvalues $-1/\phi$ and ϕ with corresponding (normalized) eigenvectors $\Phi_1 = \frac{1}{\sqrt{\phi+2}}(\phi,\ -1)$ and $\Phi_2 = \frac{1}{\sqrt{\phi+2}}(1,\ \phi)$. Hence T_1 acts as an expansion by a factor of ϕ along Φ_2, and reflection followed by contraction by a factor of ϕ along Φ_1. T_2 is the same as T_1 followed by a vertical translation of -1. It follows after some straightforward

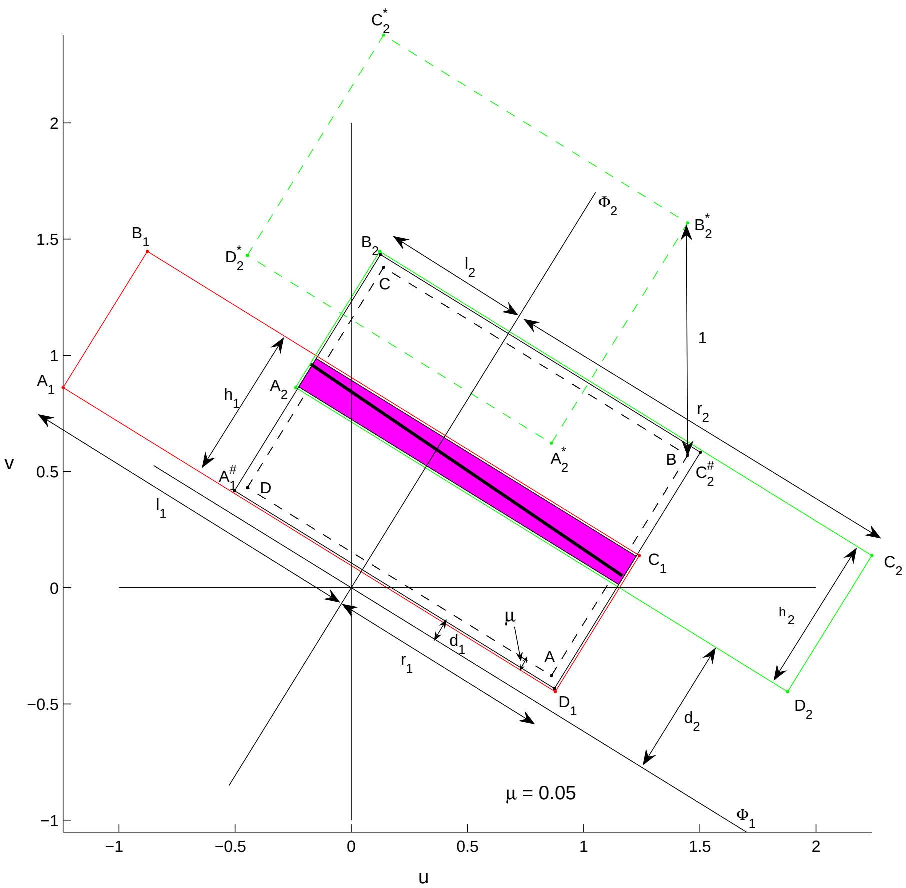

Fig. 6. A positively invariant set and the demonstration of its robustness with respect to additive noise and imperfect quantization.

algebraic calculations that the mapping relations described above imply

$$\begin{aligned}
r_1 &= \frac{\phi}{\sqrt{\phi+2}} + \phi^2\mu \\
l_1 &= \frac{\phi^2}{\sqrt{\phi+2}} + \phi^2\mu \\
r_2 &= \frac{2\phi}{\sqrt{\phi+2}} + \phi^2\mu \\
l_2 &= \frac{1}{\sqrt{\phi+2}} + \phi^2\mu \\
h_1 = h_2 &= \frac{\phi}{\sqrt{\phi+2}} - 2\phi\mu \\
d_1 &= \phi\mu \\
d_2 &= \frac{1}{\sqrt{\phi+2}} + \phi\mu.
\end{aligned}$$

Consequently, the positively invariant set R is the set of all points inside the rectangle $A_1^{\#} B_2 C_2^{\#} D_1$ where

$$\begin{aligned} A_1^{\#} &= -l_2\Phi_1 + d_1\Phi_2 \\ B_2 &= -l_2\Phi_1 + (d_2+h_2)\Phi_2 \\ C_2^{\#} &= r_1\Phi_1 + (d_2+h_2)\Phi_2 \\ D_1 &= r_1\Phi_1 + d_1\Phi_2 \end{aligned} \tag{29}$$

Note that R depends only on μ. Moreover, the existence of the overlapping region F is equivalent to the condition $d_1 + h_1 > d_2$ which turns out to be equivalent to

$$\mu < \frac{1}{2\phi^2\sqrt{\phi+2}} \approx 0.1004.$$

■

Flaky quantizers, linear thresholds: Next we consider the case $Q = Q_\alpha^{\nu_1,\nu_2}$ and specify ranges for ν_1, ν_2, and α such that $T_Q(R(\mu)) \subseteq R(\mu) + B_\mu(0)$.

Proposition 7. *Let* $0 < \mu < \frac{1}{2\phi^2\sqrt{\phi+2}}$ *and* α *such that*

$$\alpha_{min}(\mu) := 1 + 2\mu\phi\sqrt{\phi+2} \le \alpha \le 3 - \frac{10\mu\phi\sqrt{\phi+2}}{1+4\mu\sqrt{\phi+2}} =: \alpha_{max}(\mu) \tag{30}$$

be fixed. Define

$$\nu_{min}(\alpha,\mu) := \begin{cases} 1 + \mu\phi\sqrt{\phi+2}, & \alpha \le \phi, \\ \alpha\left(\frac{\phi+1}{\phi+2} + \frac{2\mu\phi^2}{\sqrt{\phi+2}}\right) + \left(\frac{1-\phi}{\phi+2} - \frac{\mu\phi^2}{\sqrt{\phi+2}}\right), & \alpha > \phi, \end{cases} \tag{31}$$

$$\nu_{max}(\alpha,\mu) := \begin{cases} \alpha - \mu\phi\sqrt{\phi+2}, & \alpha \le \phi, \\ \alpha\left(\frac{1}{\phi+2} - \frac{2\mu\phi^2}{\sqrt{\phi+2}}\right) + \left(\frac{1+2\phi}{\phi+2} + \frac{\mu\phi^2}{\sqrt{\phi+2}}\right), & \alpha > \phi. \end{cases} \tag{32}$$

If $\nu_{min}(\alpha,\mu_0) \le \nu_1 \le \nu_2 \le \nu_{max}(\alpha,\mu_0)$, *then* $T_{Q_\alpha^{\nu_1,\nu_2}}(R(\mu)) \subseteq R(\mu) + B_\mu(0)$ *for every* $\mu \in [0,\mu_0]$.

Remarks.

1) The dark line segment depicted in Figure 6 within the overlapping region F, refers to a hypothetical quantizer threshold that is allowed to vary within F. Proposition 7 essentially determines the vertical axis intercepts of lines with a given slope, $-1/\alpha$, the corresponding segments of which remain in the overlapping region F. The proof is straightforward but tedious, and will be omitted.
2) In Figure 7 we show $\nu_{min}(\alpha,\mu)$ and $\nu_{max}(\alpha,\mu)$ for $\mu = 0, 0.01, 0.03$. Note that ν_{min} and ν_{max} are both increasing in α. Moreover, for α in the range shown in (30), we have $\nu_{min}(\alpha,\mu) \le \nu_{max}(\alpha,\mu)$ with $\nu_{max}(\alpha,\mu) = \nu_{min}(\alpha,\mu)$ at the two endpoints, $\alpha_{min}(\mu)$ and $\alpha_{min}(\mu)$. Hence, ν_{min} and ν_{max} enclose a bounded region, say $G(\mu)$. If $\{\alpha\} \times [\nu_1,\nu_2]$ is in $G(\mu)$, then $T_Q(R(\mu)) \subseteq R(\mu) + B_\mu(0)$ for $Q = Q_\alpha^{\nu_1,\nu_2}$.
3) For any $\alpha_L \in (\alpha_{min}(\mu), \alpha_{max}(\mu))$, note that ν_{min} is invertible at α_L and set

   $$\alpha_U^* := \nu_{min}^{-1}(\nu_{max}(\alpha_L)).$$

   Then, for any $\alpha_U \in (\alpha_L, \alpha_U^*)$, we have

   $$(\alpha_L,\alpha_U) \times (\nu_1,\nu_2) \in G(\mu)$$

   provided

   $$\nu_L := \nu_{min}(\alpha_U) \le \nu_1 < \nu_2 \le \nu_{max}(\alpha_L) =: \nu_U.$$

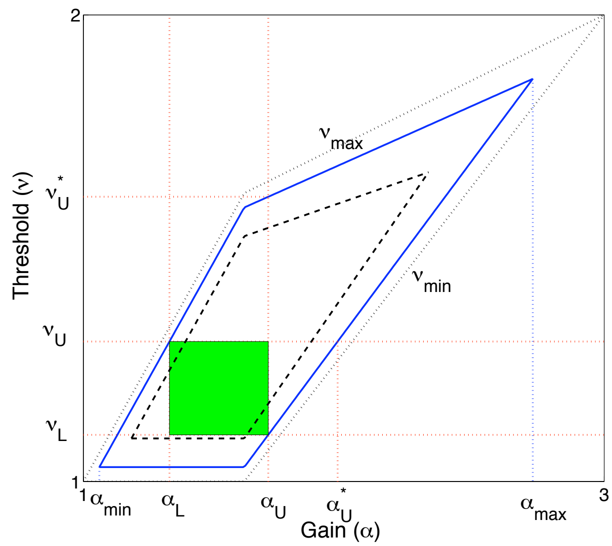

Fig. 7. We plot $\nu_{min}(\alpha, \mu)$ and $\nu_{min}(\alpha, \mu)$ with $\mu = 0.03$ (dashed), $\mu = 0.01$ (solid), and $\mu = 0$ (dotted). If $\{\alpha\} \times [\nu_1, \nu_2]$ remains in the shaded region, then $T_{Q_\alpha^{\nu_1,\nu_2}}(R(\mu)) \subseteq R(\mu) + B_\mu(0)$ for $\mu = 0.01$.

Note also that for any $(\alpha_1, \alpha_2) \subset (\alpha_L, \alpha_U^*)$, we have $\nu_{min}(\alpha_2) < \nu_{max}(\alpha_1)$. Thus,

$$(\alpha_1, \alpha_2) \times (\nu_1, \nu_2) \subset G(\mu)$$

if $\nu_{min}(\alpha_2) \leq \nu_1 < \nu_2 \leq \nu_{max}(\alpha_1)$. Consequently, we observe that

$$T_{Q_\alpha^{\nu_1,\nu_2}}(R(\mu)) \subseteq R(\mu) + B(\mu)$$

for any $\alpha \in (\alpha_1, \alpha_2)$ and $[\nu_1, \nu_2] \subset [\nu_{min}(\alpha_2), \nu_{max}(\alpha_1)]$.

4) We can also determine the allowed range for α, given the range for ν_1, ν_2, by reversing the argument above. The extreme values of ν_{min} and ν_{max} are

$$\nu_{min}(\alpha_{min}) = 1 + \mu\phi\sqrt{\phi + 2}, \tag{33}$$

$$\nu_{min}(\alpha_{max}) = \frac{6\phi + 2 + 3\mu(4\phi + 3)\sqrt{\phi + 2}}{(\phi + 4\mu\phi^2\sqrt{\phi + 2})(\phi + 2)}. \tag{34}$$

For any ν_L in the open interval between these extreme values, set

$$\nu_U^* := \nu_{max}(\nu_{min}^{-1}(\nu_L)).$$

Then, for any $\nu_U \in (\nu_L, \nu_U^*)$, $(\alpha_L, \alpha_U) \times (\nu_L, \nu_U)$ is in the allowed region $G(\mu)$ where

$$\alpha_L := \nu_{max}^{-1}(\nu_U) \tag{35}$$

$$\alpha_U := \nu_{min}^{-1}(\nu_L). \tag{36}$$

This is shown in Figure 7. Consequently, we observe that GRE implemented with $Q = Q_\alpha^{\nu_1,\nu_2}$ remains stable for any ν_1, ν_2 and α provided $[\nu_1, \nu_2] \subset (\nu_L, \nu_U)$ and $\alpha \in (\alpha_L, \alpha_U)$.

5) For the case $\mu = 0$, the expressions above simplify significantly. In (30), $\alpha_{min}(0) = 1$ and $\alpha_{max}(0) = 3$. Consequently, the extreme values ν_{min} and ν_{max} are 1 and 2, respectively. One can repeat the

calculations above to derive the allowed ranges for the parameters to vary. Observe that $\mu = 0$ gives the widest range for the parameters. This can be seen in Figure 7.

We now go back to the GRE and state the implications of the results obtained in this subsection. The recursion relations (22) that define GRE assume perfect arithmetic. We modify (22) to allow arithmetic imperfections, e.g., additive noise, as follows

$$\begin{aligned} u_{n+2} &= u_{n+1} + u_n - b_n + \epsilon_n, \\ b_n &= Q(u_n, u_{n+1}), \end{aligned} \tag{37}$$

and conclude this section with the following stability theorem.

Theorem 8. *Let $\mu \in [0, \frac{1}{2\phi^2\sqrt{\phi+2}})$, $\alpha_{min}(\mu)$ and $\alpha_{max}(\mu)$ as in* (30). *For every $\alpha \in (\alpha_{min}(\mu), \alpha_{max}(\mu))$, there exists ν_1, ν_2, and $\eta > 0$ such that the encoder described by* (37) *with $Q = Q_{\alpha'}^{\nu_1,\nu_2}$ is stable provided $|\alpha' - \alpha| < \eta$ and $|\epsilon_n| < \mu$.*

Proof: This immediately follows from our remarks above. In particular, given α, choose $\alpha_L < \alpha$ such that the corresponding $\alpha_U^* = \nu_{min}^{-1}(\nu_{max}(\alpha_L)) > \alpha$. By monotonicity of both ν_{min} and ν_{max}, any $\alpha_L \in (\nu_{max}^{-1}(\nu_{min}(\alpha)), \alpha)$ will do. Next, choose some $\alpha_U \in (\alpha, \alpha_U^*)$, and set $\eta := \min\{\alpha - \alpha_L, \alpha_U - \alpha\}$. The statement of the theorem now holds with $\nu_1 = \nu_{min}(\alpha_U)$ and $\nu_2 = \nu_{max}(\alpha_L)$. ■

When $\mu = 0$, i.e., when we assume the recursion relations (22) are implemented without an additive error, Theorem 8 implies that GRE is strongly robust in approximation with respect to the parameter α, ν_1, and ν_2. More precisely, the following corollary holds.

Corollary 9. *Let $x \in [0, 1]$, and $\alpha \in (1, 3)$. There exist $\eta > 0$, and ν_1, ν_2 such that b_n, generated via* (22) *with $u_0 = x$, $u_1 = 0$, and $Q = Q_{\alpha'}^{\nu_1,\nu_2}$, approximate x exponentially accurately whenever $|\alpha' - \alpha| < \eta$. In particular, the N-term approximation error $e_N(x)$ satisfies*

$$e_N(x) = x - \sum_{n=0}^{N-1} b_n \phi^{-n} \leq C\phi^{-N}$$

where $C = 1 + \phi$.

Proof: The claim follows from Proposition 1 and Theorem 8: Given α, set $\mu = 0$, and choose ν_1, ν_2, and η as in Theorem 8. As the set $R(0)$ contains $[0, 1] \times \{0\}$, such a choice ensures that the corresponding GRE is stable on $[0, 1]$. Using Proposition 1, we conclude that $e_N(x) \leq \phi^{-N}(u_N + \phi u_{N+1})$. Finally, as $(u_N, u_{N+1}) \in R(0)$,

$$u_N + \phi u_{N+1} \leq \max_{(u,v) \in R(0)} u + \phi v = 1 + \phi.$$

■

D. Effect of additive noise and arithmetic errors on reconstruction error

Corollary 9 shows that the GRE is robust with respect to quantizer imperfections under the assumption that the recursion relations given by (22) are strictly satisfied. That is, we have quite a bit of freedom in choosing the quantizer Q, assuming the arithmetic, i.e., addition, can be done error-free. We now investigate the effect of arithmetic errors on the reconstruction error. To this end, we model such imperfections as additive noise and, as before, replace (22) with (37), where ϵ_n denotes the additive noise. While Theorem 8 shows that the encoder is stable under small additive errors, the reconstruction error is not guaranteed to become arbitrarily small with increasing number of bits. This is observed in

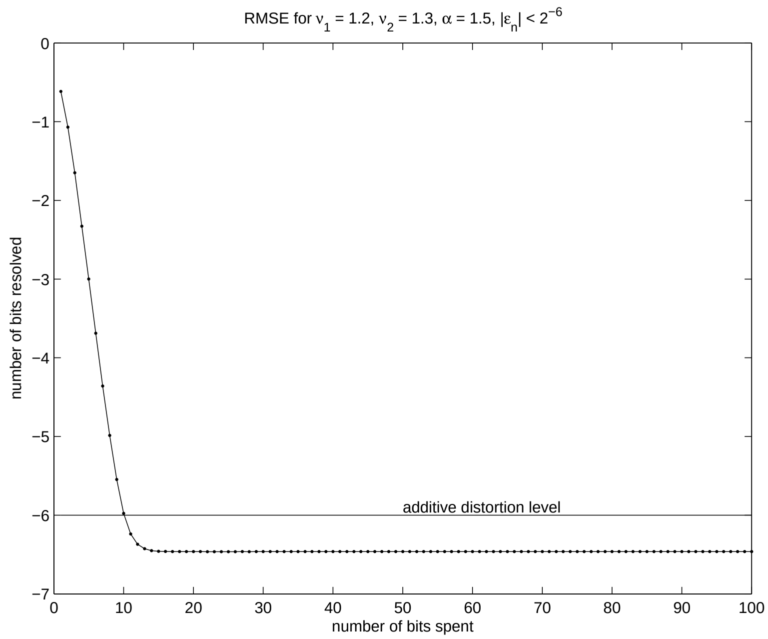

Fig. 8. Demonstration of the fact that reconstruction error saturates at the noise level. Here the parameters are $\nu_1 = 1.2$, $\nu_2 = 1.3$, $\alpha = 1.5$, and $|\epsilon_n| < 2^{-6}$.

Figure 8 where the system is stable for the given imperfection parameters and noise level, however the reconstruction error is never better than the noise level.

Note that for stable systems, we unavoidably have

$$\sum_{n=0}^{N-1} b_n \phi^{-n} = x + \sum_{n=0}^{N-1} \epsilon_n \phi^{-n} + O(\phi^{-N}), \tag{38}$$

where the "noise term" in (38) does not vanish as N tends to infinity. To see this, define $\varepsilon_N := \sum_{n=0}^{N-1} \epsilon_n \phi^{-n}$. If we assume ϵ_n to be i.i.d. with mean 0 and variance σ^2, then we have

$$\mathrm{Var}(\varepsilon_N) = \frac{1-\phi^{-2N}}{1-\phi^{-2}}\sigma^2 \to \phi\,\sigma^2 \quad \text{as} \quad N \to \infty.$$

Hence we can conclude from this the x-independent result

$$\mathbf{E}\left| x - \sum_{n=0}^{\infty} b_n \phi^{-n} \right|^2 = \phi\,\sigma^2.$$

In Figure 8, we incorporated uniform noise in the range $[-2^{-6}, 2^{-6}]$. This would yield $\phi\sigma^2 = 2^{-12}\phi/3 \approx 2^{-13}$, hence the saturation of the root-mean-square-error (RMSE) at $2^{-6.5}$. Note, however, that the figure was created with an average over 10,000 randomly chosen x values. Although independent experiments were not run for the same values of x, the x-independence of the above formula enables us to predict the outcome almost exactly.

In general, if ρ is the probability density function for each ϵ_n, then ε_N will converge to a random variable the probability density function of which has Fourier transform given by the convergent infinite product

$$\prod_{n=0}^{\infty} \widehat{\rho}(\phi^{-n}\xi).$$

E. Bias removal for the decoder

Due to the nature of any 'cautious' beta-encoder, the standard N-bit decoder for the GRE yields approximations that are biased, i.e., the error $e_N(x)$ has a non-zero mean. This is readily seen by the error formula

$$e_N(x) = \phi^{-N}(u_N + \phi u_{N+1})$$

which implies that $e_N(x) > 0$ for all x and N. Note that all points (u, v) in the invariant rectangle R_Q satisfy $u + \phi v > 0$.

This suggests adding a constant (x-independent) term ξ_N to the standard N-bit decoding expression to minimize $\|e_N\|$. Various choices are possible for the norm. For the ∞-norm, ξ_N should be chosen to be average value of the minimum and the maximum values of $e_N(x)$. For the 1-norm, ξ_N should be the median value and for the 2-norm, ξ_N should be the mean value of $e_N(x)$. Since we are interested in the 2-norm, we will choose ξ_N via

$$\xi_N = \phi^{-N} \frac{1}{|I|} \int_I (u_N(x) + \phi\, u_{N+1}(x))\ dx,$$

where I is the range of x values and we have assumed uniform distribution of x values.

This integral is in general difficult to compute explicitly due to the lack of a simple formula for $u_N(x)$. One heuristic that is motivated by the mixing properties of the map T_Q is to replace the average value $|I|^{-1} \int_I u_N(x)\, dx$ by $\int_\Gamma u\, dudv$. If the set of initial conditions $\{(x, 0) : x \in I\}$ did not have zero two-dimensional Lebesgue measure, this heuristic could be turned into a rigorous result as well.

However, there is a special case in which the bias can be computed explicitly. This is the case $\alpha = 1$ and $\nu_1 = \nu_2 = 1$. Then the invariant set is $[0, 1)^2$ and

$$\begin{bmatrix} u_N(x) \\ u_{N+1}(x) \end{bmatrix} = \left\langle \begin{bmatrix} 0 & 1 \\ 1 & 1 \end{bmatrix}^N \begin{bmatrix} x \\ 0 \end{bmatrix} \right\rangle$$

where $\langle\cdot\rangle$ denotes the coordinate-wise fractional part operator on any real vector. Since $\int_0^1 \langle kx \rangle\, dx = 1/2$ for all non-zero integers k, it follows that $\int_0^1 u_N(x)\, dx = 1/2$ for all N. Hence setting $I = [0, 1]$, we find

$$\xi_N = \phi^{-N} \frac{1 + \phi}{2} = \frac{1}{2}\phi^{-N+2}.$$

It is also possible to compute the integral $\int_\Gamma u\, dudv$ explicitly when $\alpha = \phi$ and $\nu_1 = \nu_2 = \nu$ for some $\nu = [1, \phi]$. In this case it can be shown that the invariant set Γ is the union of at most 3 rectangles whose axes are parallel to Φ_1 and Φ_2. We omit the details.

F. Circuit implementation: Requantization

As we mentioned briefly in the introduction, A/D converters (other than PCM) typically incorporate a requantization stage after which the more conventional binary (base-2) representations are generated. This operation is close to a decoding operation, except it can be done entirely in digital logic (i.e., perfect arithmetic) using the bitstreams generated by the specific algorithm of the converter. In principle sophisticated digital circuits could also be employed.

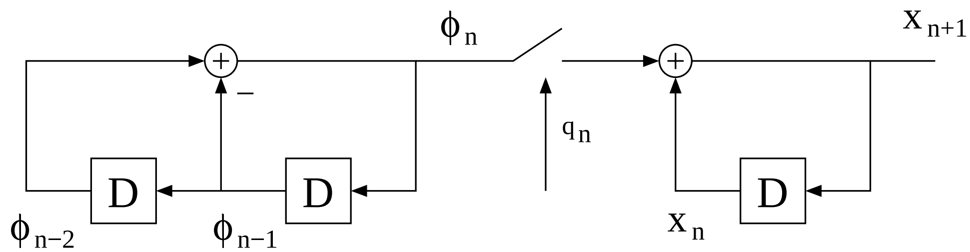

Fig. 9. Efficient digital implementation of the requantization step for the Golden Ratio Encoder.

In the case of the Golden Ratio Encoder, it turns out that a fairly simple requantization algorithm exists that incorporates a digital arithmetic unit and a minimum amount of memory that can be hardwired. The algorithm is based on recursively computing the base-2 representations of powers of the golden ratio. In Figure 9, ϕ_n denotes the B-bit base-2 representation of ϕ^{-n}. Mimicking the relation $\phi^{-n} = \phi^{-n+2} - \phi^{-n+1}$, the digital circuit recursively sets

$$\phi_n = \phi_{n-2} - \phi_{n-1}, \qquad n = 2, 3, ..., N-1$$

which then gets multiplied by q_n and added to x_n, where

$$x_n = \sum_{k=0}^{n-1} q_k \phi^{-k}, \qquad n = 1, 2, ..., N.$$

The circuit needs to be set up so that ϕ_1 is the expansion of ϕ^{-1} in base 2, accurate up to at least B bits, in addition to the initial conditions $\phi_0 = 1$ and $x_0 = 0$. To minimize round-off errors, B could be taken to be a large number (much larger than $\log N / \log \phi$, which determines the output resolution).

IV. Higher order schemes: Tribonacci and Polynacci encoders

What made the Golden Ratio Encoder (or the ‘Fibonacci’ Encoder) interesting was the fact that a beta-expansion for $1 < \beta < 2$ was attained via a difference equation with ± 1 coefficients, thereby removing the necessity to have a perfect constant multiplier. (Recall that multipliers were still employed for the quantization operation, but they no longer needed to be precise.)

This principle can be further exploited by considering more general difference equations of this same type. An immediate class of such equations are suggested by the recursion

$$P_{n+L} = P_{n+L-1} + \cdots + P_n$$

where $L > 1$ is some integer. For $L = 2$, one gets the Fibonacci sequence if the initial condition is given by $P_0 = 0$, $P_1 = 1$. For $L = 3$, one gets the Tribonacci sequence when $P_0 = P_1 = 0$, $P_2 = 1$. The general case yields the Polynacci sequence.

For bit encoding of real numbers, one then sets up the iteration

$$\begin{aligned} u_{n+L} &= u_{n+L-1} + \cdots + u_n - b_n \\ b_n &= Q(u_n, \ldots, u_{n+L-1}) \end{aligned} \tag{39}$$

with the initial conditions $u_0 = x$, $u_1 = \cdots = u_{L-1} = 0$. In L-dimensions, the iteration can be rewritten as

$$
\begin{bmatrix} u_{n+1} \\ u_{n+2} \\ \vdots \\ u_{n+L-1} \\ u_{n+L} \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 & \cdots & 0 \\ 0 & 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & 0 & 1 \\ 1 & 1 & \cdots & 1 & 1 \end{bmatrix} \begin{bmatrix} u_n \\ u_{n+1} \\ \vdots \\ u_{n+L-2} \\ u_{n+L-1} \end{bmatrix} - b_n \begin{bmatrix} 0 \\ 0 \\ \vdots \\ 0 \\ 1 \end{bmatrix} \tag{40}
$$

It can be shown that the characteristic equation

$$
s^L - (s^{L-1} + \cdots + 1) = 0
$$

has its largest root β_L in the interval $(1,2)$ and all remaining roots inside the unit circle (hence β_L is a Pisot number). Moreover as $L \to \infty$, one has $\beta_L \to 2$ monotonically.

One is then left with the construction of quantization rules $Q = Q_L$ that yield bounded sequences (u_n). While this is a slightly more difficult task to achieve, it is nevertheless possible to find such quantization rules. The details will be given in a separate manuscript.

The final outcome of this generalization is the accuracy estimate

$$
\left| x - \sum_{n=0}^{N-1} b_n \beta_L^{-n} \right| = O(\beta_L^{-N})
$$

whose rate becomes asymptotically optimal as $L \to \infty$.

V. Acknowledgments

We would like to thank Felix Krahmer, Rachel Ward, and Matt Yedlin for various conversations and comments that have helped initiate and improve this paper.

Ingrid Daubechies gratefully acknowledges partial support by the NSF grant DMS-0504924. Sinan Güntürk has been supported in part by the NSF Grant CCF-0515187, an Alfred P. Sloan Research Fellowship, and an NYU Goddard Fellowship. Yang Wang has been supported in part by the NSF Grant DMS-0410062. Özgür Yılmaz was partly supported by a Discovery Grant from the Natural Sciences and Engineering Research Council of Canada. This work was initiated during a BIRS Workshop and finalized during an AIM Workshop. The authors greatfully acknowledge the Banff International Research Station and the American Institute of Mathematics.

References

[1] H. Inose and Y. Yasuda, "A unity bit coding method by negative feedback," *Proceedings of the IEEE*, vol. 51, no. 11, pp. 1524–1535, 1963.

[2] J. Candy and G. Temes, "Oversampling Delta-Sigma Data Converters: Theory, Design and Simulation," *IEEE Press, New York*, 1992.

[3] R. Schreier and G. Temes, *Understanding delta-sigma data converters*. John Wiley & Sons, 2004.

[4] C. Güntürk, J. Lagarias, and V. Vaishampayan, "On the robustness of single-loop sigma-delta modulation," *IEEE Transactions on Information Theory*, vol. 47, no. 5, pp. 1735–1744, 2001.

[5] C. Güntürk, "One-bit sigma-delta quantization with exponential accuracy," *Communications on Pure and Applied Mathematics*, vol. 56, no. 11, pp. 1608–1630, 2003.

[6] K. Dajani and C. Kraaikamp, "From greedy to lazy expansions and their driving dynamics," *Expositiones Mathematicae*, vol. 20, no. 4, pp. 315–327, 2002.

[7] I. Daubechies, R. DeVore, C. Güntürk, and V. Vaishampayan, "A/D conversion with imperfect quantizers," *IEEE Transactions on Information Theory*, vol. 52, no. 3, pp. 874–885, March 2006.

[8] I. Daubechies and Ö. Yılmaz, "Robust and practical analog-to-digital conversion with exponential precision," *IEEE Transactions on Information Theory*, vol. 52, no. 8, August 2006.

[9] A. Karanicolas, H. Lee, and K. Barcrania, “A 15-b 1-Msample/s digitally self-calibrated pipeline ADC,” *IEEE Journal of Solid-State Circuits*, vol. 28, no. 12, pp. 1207–1215, 1993.

[10] W. Parry, “On the β-expansions of real numbers,” *Acta Mathematica Hungarica*, vol. 11, no. 3, pp. 401–416, 1960.

[11] N. Sidorov, “Almost Every Number Has a Continuum of β-Expansions,” *American Mathematical Monthly*, vol. 110, no. 9, pp. 838–842, 2003.

[12] A. N. Kolmogorov and V. M. Tihomirov, “ε-entropy and ε-capacity of sets in functional space,” *Amer. Math. Soc. Transl. (2)*, vol. 17, pp. 277–364, 1961.

[13] I. Daubechies and R. DeVore, “Reconstructing a bandlimited function from very coarsely quantized data: A family of stable sigma-delta modulators of arbitrary order,” *Annals of Mathematics*, vol. 158, no. 2, pp. 679–710, 2003.

[14] Ö. Yılmaz, “Stability analysis for several second-order sigma-delta methods of coarse quantization of bandlimited functions,” *Constructive approximation*, vol. 18, no. 4, pp. 599–623, 2002.

[15] N. Vorobiev, *Fibonacci Numbers*. Birkhäuser, 2003.