A/D Conversion with Imperfect Quantizers

I. Daubechies[*], R. DeVore[†], S. Güntürk[‡], and V. Vaishampayan[§]

April 9, 2005

Abstract

We analyze mathematically the effect of quantization error in the circuit implementation of Analog to Digital (A/D) converters such as Pulse Code Modulation (PCM) and Sigma Delta Modulation ($\Sigma\Delta$). $\Sigma\Delta$ modulation, which is based on oversampling the signal, has a self correction for quantization error that PCM does not have, and that we believe to be a major reason why $\Sigma\Delta$ modulation is preferred over PCM in A/D converters with imperfect quantizers. Motivated by this, we construct other encoders that use redundancy to obtain a similar self correction property, but that achieve higher order accuracy relative to bit rate than "classical" $\Sigma\Delta$. More precisely, we introduce two different types of encoders that exhibit exponential bit rate accuracy (in contrast to the polynomial rate of classical $\Sigma\Delta$) and still retain the self correction feature.

[*]Department of Mathematics and Program in Applied and Computational Mathematics, Princeton University, Fine Hall, Washington Road, Princeton, NJ 08544. ingrid@math.princeton.edu

[†]Department of Mathematics, University of South Carolina, Columbia, SC 29208. devore@math.sc.edu

[‡]Department of Mathematics, Courant Institute of Mathematical Sciences, 251 Mercer Street, New York, NY 10012. gunturk@cims.nyu.edu

[§]Information Sciences Research, AT&T Labs, 180 Park Avenue, Florham Park, NJ 07932. vinay@research.att.com

1 Introduction

Analog-to-digital conversion is a dynamic area of research that is driven by the need for circuits that are faster, and have higher resolution, lower power and smaller area. Major applications are speech, high quality audio, images, video and software radio.

This paper concerns the mathematical virtues and shortcomings of various encoding strategies in analog-to-digital (A/D) conversion. While the information theory community has focused on algorithm design and analysis within a rate-distortion framework, circuit designers have also had to wrestle with problems related to robustness of A/D circuits to component variations, since the manufacturing cost increases dramatically as component tolerances are reduced. In this paper, we discuss a generalized rate-distortion framework that incorporates robustness related to component variability. Within this framework, certain classes of A/D converters are studied, allowing us to establish upper bounds on what is ultimately achievable.

A/D converters can be divided into two broad classes: (i) Nyquist-rate converters, (ii) oversampled converters. Nyquist-rate converters use an analog front-end filter with suitably sharp cutoff characteristics to limit the bandwidth of the signal to less than W Hz, after which the signal is sampled at a rate of $2W$ Hz. These real-valued samples are then converted into binary words using an A/D converter, the most popular algorithm being the Successive Approximation algorithm (see e.g., [**?**]). On the other hand, sigma-delta A/D converters overcome some of the problems associated with building sharp analog bandlimiting filters through the use of oversampling. Perhaps more importantly, sigma-delta conversion has the benefit of being robust to circuit imperfections [**?**, **?**, **?**], a quality that is not shared by Nyquist-rate converters based on the successive approximation method. In some sense, sigma-delta converters are able to effectively use redundancy in order to gain robustness to component variations. We will see that a similar principle also applies to a certain class of Nyquist-rate converters that is not so well known within the information theoretic community.

We begin with an analysis of Nyquist-rate converters based on the successive approximation algorithm. As is customary in audio applications, we model these signals by bandlimited functions, i.e., functions with compactly supported Fourier transforms. Any bandlimited function can be recovered perfectly from its samples on a sufficiently close-spaced grid; this is known as the "sampling theorem". Let $\mathcal{B}(\Omega)$ denote the class of real-valued functions $x \in \mathcal{L}^2(\mathbb{R})$ whose Fourier transforms are supported on $[-\Omega, \Omega]$. The Shannon-Whittaker formula gives a way to reconstruct a function $x \in \mathcal{B}(\pi)$ from its samples $(x(n))_{n\in\mathbb{Z}}$ taken on the integer grid:

$$x(t) = \sum_{n\in\mathbb{Z}} x(n)S(t-n), \tag{1.1}$$

where S is the *sinc*-kernel

$$S(t) := \frac{\sin \pi t}{\pi t}. \tag{1.2}$$

The functions $S(\cdot - n)$, $n \in \mathbb{Z}$, form a complete orthonormal system for $\mathcal{B}(\pi)$. Clearly, the formula above can be extended through dilation to functions in $\mathcal{B}(\Omega)$ for arbitrary Ω:

$$\text{for } y \in \mathcal{B}(\Omega): \; y(t) = \sum_{n\in\mathbb{Z}} y\left(\frac{n\pi}{\Omega}\right) S\left(\frac{\Omega t}{\pi} - n\right). \tag{1.3}$$

Let us note that the formulas (**??**) and (**??**) should be treated with care if the samples $x(n)$ or $y\left(\frac{n\pi}{\Omega}\right)$ have to be replaced by approximate values. There is no guarantee of convergence of the series under arbitrary bounded perturbations of the samples, regardless of how small these may be, because

$$\sum_{n\in\mathbb{Z}} |S(t-n)| = \infty \; .$$

This instability problem is caused by the slow decay of $|S(t)|$ as $|t| \to \infty$; it can easily be fixed by oversampling, since one can then employ reconstruction kernels that have faster decay than the *sinc*-kernel (see Section 2). For simplicity of the discussion in this introduction, we shall not worry about this problem here.

In practice one observes signals that are bounded in amplitude and takes samples only on a finite portion of the real line. For a positive number A and an interval $I \subset \mathbb{R}$, we denote by $\mathcal{B}(\Omega, A, I)$ the class of functions that are restrictions to I of functions $y \in \mathcal{B}(\Omega)$ that satisfy $|y(t)| \leq A$ for all $t \in \mathbb{R}$. It will be sufficient in all that follows to consider the case where $\Omega = \pi$ and $A = 1$; the more general case can easily be derived from this particular case. We shall use the shorthand $\mathcal{B}_I$ for $\mathcal{B}(\pi, 1, I)$, and $\mathcal{B}$ for $\mathcal{B}(\pi, 1, \mathbb{R})$.

Modulo the stability issue mentioned above, the sampling theorem (in the form (**??**)) can be employed to reduce the problem of A/D conversion of a signal $x \in \mathcal{B}_I$ to the quantization of a finite number of $x(n)$, $n \in I$, all of which are moreover bounded by 1. (In fact, one really needs to quantize the $x(n)$ with $n \in \bar{I}$, where $\bar{I}$ is an "enlarged" version of I, starting earlier and ending later; more details will be given in Section **??**. In practice $|I|$ is very large compared to the duration of one Nyquist interval, and these extra margins do not contribute to the average bit rate as $|I| \to \infty$.) This method is known as Pulse Code Modulation (PCM). Here quantization means replacing each $x(n)$ by a good (digital) approximation $\tilde{x}(n)$, such as the truncated version of its binary expansion.

Let us consider the successive approximation algorithm. The binary expansion of a real number $r \in [-1, 1]$ has the form

$$r = b_0 \sum_{i=1}^{\infty} b_i 2^{-i},$$

where $b_0 = b_0(r) \in \{-1, 1\}$ is the sign bit, and $b_i = b_i(r) \in \{0, 1\}$, $i \geq 1$, are the binary digits of $|r|$. The sign bit is given by $b_0 = Q_0(r)$, where the quantizer Q_0 is simply the *sign*-function,

$$Q_0(z) := \begin{cases} -1, & z < 0 \\ 1, & z \geq 0. \end{cases} \tag{1.4}$$

For the simplicity of notation, we define a second quantizer function Q_1 by

$$Q_1(z) := \begin{cases} 0, & z < 1 \\ 1, & z \geq 1; \end{cases} \tag{1.5}$$

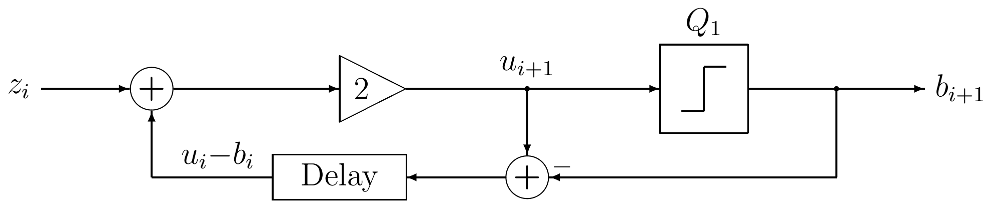

Figure 1: *With input* $z_0 = |r|$, $z_i = 0$, $i \geq 1$, *and "initial conditions"* $u_0 = 0$, $b_0 = 0$, *the output* $(b_i)_{i \geq 1}$ *gives the binary representation for* $|r| \in [0, 1]$.

Q_0 and Q_1 can be simply derived from each other via the relation $Q_1(z) = \frac{1}{2}\big(Q_0(z-1)+1\big)$. The b_i can be computed in the following algorithm. Let $u_1 := 2|r|$; the first bit b_1 is then given by $b_1 := Q_1(u_1)$. The remaining bits are computed recursively; if u_i and b_i have been defined, we let

$$u_{i+1} := 2(u_i - b_i) \tag{1.6}$$

and

$$b_{i+1} := Q_1(u_{i+1}). \tag{1.7}$$

It is easy to see that $u_i \in [0, 2]$ for all $i \geq 1$, and

$$\varepsilon_L := |r| - \sum_{i=1}^{L} b_i 2^{-i} = \frac{u_{L+1}}{2^{L+1}}$$

satisfies $0 \leq \varepsilon_L \leq 2^{-L}$.

The resulting PCM encoding $E_{\text{PCM},L}(x)$ of the signal $x \in \mathcal{B}_I$ lists, for each $x(n)$ in I, its bits $b_0, b_1, \ldots, b_L$, thus yielding an approxiation of x with precision $O(2^{-L})$. (Here we have again ignored the instability of the expansion (**??**).) As explained in Section 2.2, this representation is optimal in a certain deterministic rate-distortion sense based on Kolmogorov entropy. Yet in the practice of A/D conversion this representation is not as popular as its optimality might suggest. This is mainly because the PCM algorithm given above is not *robust* in an important and very practical sense. Up to the point where the signal is truly digitized, it "lives" in the analog world, and all operations in the circuit that implements the A/D conversion have to be made with analog components. Figure **??** shows a block diagram for the determination of the bits $b_1, b_2, \ldots$, given $|r| \in [0, 1]$, corresponding to (**??**), (**??**). The output of Q_1 in this figure is "digital" in the sense that it can take only two values, 0 and 1. However, the input to Q_1 is analog, so Q_1 must be, at least in part, an analog component. This means that there is no guarantee that $Q_1(z)$ "toggles" exactly at $z = 1$. Of course, the same can be said for Q_0 as well. For instance, Q_0 could deviate from the *sign*-function by a shift, yielding an imprecise quantizer

$$Q_{0,\rho}(z) := \begin{cases} -1, & z < \rho \\ 1, & z \geq \rho, \end{cases} \tag{1.8}$$

where the shift ρ is not known precisely, except that it lies within a certain known tolerance,

$$|\rho| < \delta \ , \tag{1.9}$$

where $\delta > 0$ is fixed; moreover, its value may differ from one application to the next. More generally, Q_0 could be replaced by a "flaky" version $Q^{\mathrm{f}}_{0,\delta}$ for which we know only that

$$Q^{\mathrm{f}}_{0,\delta}(z) := \begin{cases} -1, & z \le -\delta \\ 1, & z \ge \delta \\ -1 \text{ or } 1, & z \in (-\delta, \delta). \end{cases} \tag{1.10}$$

This notation means that for each z, the flaky quantizer $Q^{\mathrm{f}}_{0,\delta}$ will assign a value in $\{-1, 1\}$, but that for $z \in (-\delta, \delta)$ we just do not know what that value will be; moreover, if $Q^{\mathrm{f}}_{0,\delta}$ acts on the same z in $(-\delta, \delta)$ in two different instances, it need not assign the same value both times. We could model this by a (possibly input-dependent) probability distribution on $\{-1, 1\}$, but since our results will hold for any such distribution, however ill-behaved, it is easier to stick to (**??**). The imprecise versions of Q_1 are, in the first case,

$$Q_{1,\rho}(z) := \begin{cases} 0, & z < 1 + \rho \\ 1, & z \ge 1 + \rho, \end{cases} \tag{1.11}$$

where ρ is constrained only by (**??**), and in the "flaky" case,

$$Q^{\mathrm{f}}_{1,\delta}(z) := \begin{cases} 0, & z \le 1 - \delta \\ 1, & z \ge 1 + \delta \\ 0 \text{ or } 1, & z \in (1 - \delta, 1 + \delta). \end{cases} \tag{1.12}$$

We define $Q^{\mathrm{f}}_{0,0} = Q_0$ and $Q^{\mathrm{f}}_{1,0} = Q_1$.

The debilitating effect of the quantization error can make itself felt in every bit, even as early as b_0 or b_1. Assume, for example, that we are working with $Q_{1,\rho}$, with $\rho > 0$. If $|r| \in \left(\frac{1}{2}, \frac{1+\rho}{2}\right)$, then $b_1(r)$ will be computed to be 0, which is incorrect since $|r| > \frac{1}{2}$. No matter how the remaining bits are assigned, the difference between r and the number represented by the computed bits will be at least $|r| - \frac{1}{2}$, which can be as large as $\frac{\rho}{2}$, and effectively as large as $\frac{\delta}{2}$ by (**??**). This could easily translate to a uniform error of magnitude $\frac{\delta}{2}$, as it would happen for the constant function $x(t) \equiv r$ and for the scaled *sinc*-function $x(t) = rS(t)$. A similar anomaly can of course occur with the sign bit b_0, too, if it is determined via $Q_{0,\rho}$ rather than Q_0: for $r \in [0, \rho)$, the sign bit $b_0(r)$ will be incorrectly assigned, causing an error that may be as large as δ.

In this paper, we present a study of different encoding schemes, and see how they trade off accuracy for robustness. In Section 2, we define a framework for the discussion of our different coders, and we carry out a more systematic analysis of PCM, its optimality and its lack of robustness. Next we discuss two other schemes: the first one, in Section 3, is the Sigma-Delta modulation; the second, in Section 4, replaces the binary expansion in PCM by an expansion in a fractional base $\beta \in (1, 2)$. We will show that both schemes are robust, in the sense that even if the quantizer used to derive the digital representations is imperfect, it is possible to reconstruct the signals with arbitrarily small error by taking more and more bits from their digital representations; neither of them is optimal in the bit-rate, however. Sigma-Delta modulation is well-known and popular in A/D conversion [**?**, **?**, **?**]; fractional base expansions are less common, but also have been considered before, e.g., in [?]. In both cases, robustness is linked to the fact that there is a certain amount of redundancy in the codewords produced by these algorithms. It is this interplay between redundancy and robustness that constitutes the major theme of this paper.

2 Encoding-Decoding, Kolmogorov ϵ-Entropy and Robustness

2.1 The average Kolmogorov ϵ-entropy of $\mathcal{B}$

Let $\mathcal{B}$ be the space of our signals, as defined in Section **??**. By an encoder E for $\mathcal{B}_I$ we mean a mapping

$$E : \mathcal{B}_I \to \mathscr{B}$$

where the elements in $\mathscr{B}$ are finite bitstreams. The result of the encoding is to take the analog signal x to the digital domain. We also have a decoder D that maps bitstreams to signals

$$D : \mathscr{B} \to \mathcal{C}_I$$

where $\mathcal{C}_I$ denotes the space of continuous functions on I. Note that we may consider many decoders D associated to a given E, and vice versa.

In general $DEx \neq x$. We can measure the distortion in the encoding-decoding by

$$\|x - DEx\|$$

where $\|\cdot\|$ is a norm defined on the signals of interest. We shall restrict our attention to the norm

$$\|x\|_{\mathcal{C}_I} := \|x\|_{L^\infty(I)} := \sup_{t\in I} |x(t)|.$$

One way of assessing the performance of an encoder-decoder pair (E, D) is to measure the worst-case distortion for the class $\mathcal{B}_I$, defined by

$$d(\mathcal{B}_I; E, D) := \sup_{x\in\mathcal{B}_I} \|x - DEx\|_{L^\infty(I)}.$$

In order to have a fair competition between various encoding-decoding schemes, we can look at the performance as a function of the bit-cost. We define the bit-cost of an encoder E on the space $\mathcal{B}_I$ by

$$n(E, \mathcal{B}_I) := \sup_{x\in\mathcal{B}_I} \#(Ex),$$

where $\#(Ex)$ is the number of bits in the bitstream Ex. Given a distortion tolerance $\epsilon > 0$, we let $\mathscr{C}_\epsilon(\mathcal{B}_I)$ denote the class of encoders $E : \mathcal{B}_I \to \mathscr{B}$ for which there exists a decoder $D : \mathscr{B} \to \mathcal{C}_I$ such that $d(\mathcal{B}_I; E, D) \leq \epsilon$. The smallest number of bits that can realize the distortion ϵ for $\mathcal{B}_I$ is then given by

$$n_\epsilon(\mathcal{B}_I) := \inf_{E\in\mathscr{C}_\epsilon(\mathcal{B}_I)} n(E, \mathcal{B}_I). \tag{2.1}$$

It is well known that $n_\epsilon(\mathcal{B}_I)$ is determined by the Kolmogorov ϵ-entropy $H_\epsilon(\mathcal{B}_I, \mathcal{C}_I)$ of the class $\mathcal{B}_I$ as a subset of $\mathcal{C}_I$. Here $H_\epsilon(\mathcal{B}_I, \mathcal{C}_I) := \log_2 N_\epsilon(\mathcal{B}_I)$ where $N := N_\epsilon(\mathcal{B}_I)$ is the smallest number of functions $f_1, \dots, f_N$ in $\mathcal{C}_I$ forming an ϵ-net for $\mathcal{B}_I$, i.e., for each $x \in \mathcal{B}_I$ there exists an element f_i in this collection such that $\|x - f_i\|_\infty \le \epsilon$. (In other words, N is the smallest number of balls in $\mathcal{C}_I$, with radius ϵ in the $\|\cdot\|_\infty$-norm, that can cover $\mathcal{B}_I$.) The number $N_\epsilon(\mathcal{B}_I)$ is finite because the space $\mathcal{B}_I$ is a compact subset of $(\mathcal{C}_I, \|\cdot\|_\infty)$. It is then easy to see that

$$n_\epsilon(\mathcal{B}_I) = \lceil \log_2 N_\epsilon(\mathcal{B}_I) \rceil = \lceil H_\epsilon(\mathcal{B}_I, \mathcal{C}_I) \rceil \tag{2.2}$$

with $\lceil y \rceil$ denoting the smallest integer $\ge y$. Indeed, an encoder-decoder pair with bit-cost $\lceil \log_2 N_\epsilon(\mathcal{B}_I) \rceil$ can easily be given: use the $\lceil \log_2 N_\epsilon(\mathcal{B}_I) \rceil$ bits to label the functions $\{f_i,\ 1 \le i \le N_\epsilon(\mathcal{B}_I)\}$ and assign each function $x \in \mathcal{B}_I$ to the label of the nearest f_i. On the other hand, $n_\epsilon(\mathcal{B}_I)$ cannot get any smaller because otherwise one could construct an ϵ-net via the same method with fewer elements than $N_\epsilon(\mathcal{B}_I)$.

In many applications in signal processing, the interval I on which one wishes to recover the signals $x \in \mathcal{B}_I$ is varying, and $|I|$ is large relative to the Nyquist sample spacing. In the case of audio signals, sampling at $44,000$ times per second corresponds to high quality audio, and sampling at $8,000$ times per second corresponds to wireless voice applications. Recovering $\mathfrak{m}$ minutes of such a signal means that the interval I corresponds to a number of $2,640,000\mathfrak{m}$ or $480,000\mathfrak{m}$ consecutive samples, respectively. So, the interval I, while finite, is typically very large. Given a family of encoders $\mathbf{E} = \{E_I\}_I$, defined for all intervals I, we define the average bit-cost (i.e., *bit rate*) of this family to be

$$\bar{n}(\mathbf{E}) := \limsup_{|I|\to\infty} \frac{n(E_I, \mathcal{B}_I)}{|I|}. \tag{2.3}$$

The optimal encoding efficiency for the space $\mathcal{B}$ can be measured by the optimal average bit rate that can be achieved within distortion ϵ. More formally, we define the quantity

$$\bar{n}_\epsilon(\mathcal{B}) := \lim_{|I|\to\infty} \frac{n_\epsilon(\mathcal{B}_I)}{|I|}. \tag{2.4}$$

There is a corresponding concept of average ϵ-entropy (per unit interval) of $\mathcal{B}$, given by

$$\bar{H}_\epsilon(\mathcal{B}) := \lim_{|I|\to\infty} \frac{H_\epsilon(\mathcal{B}_I, \mathcal{C}_I)}{|I|}. \tag{2.5}$$

It easily follows from (**??**) that

$$\bar{n}_\epsilon(\mathcal{B}) = \bar{H}_\epsilon(\mathcal{B}). \tag{2.6}$$

The average ϵ-entropy of the class $\mathcal{B}$ can be derived from results of Kolmogorov and Tikhomirov [**?**] on the average ϵ-entropy of certain classes of analytic functions. These results yield

$$\bar{H}_\epsilon(\mathcal{B}) = (1 + o(1)) \log_2 \frac{1}{\epsilon}, \quad 0 < \epsilon < 1. \tag{2.7}$$

2.2 PCM encoders achieve the Kolmogorov ϵ-entropy

The Kolmogorov ϵ-entropy can be defined for any pair $(\mathcal{F}, \mathcal{V})$, where $\mathcal{F}$ is a compact subset of the metric space $\mathcal{V}$. Even when $H_\epsilon(\mathcal{F}, \mathcal{V})$ can be computed explicitly, we may not know of any constructive example of an encoder-decoder pair that achieves (or near-achieves) the Kolmogorov entropy. However, in the present case, where $\mathcal{F} = \mathcal{B}_I$, $\mathcal{V} = \mathcal{C}_I$, and we are interested in $H_\epsilon(\mathcal{B}_I, \mathcal{C}_I)$ when $|I|$ is large (or equivalently, $\bar{H}_\epsilon(\mathcal{B})$ defined by the limit (**??**)), the PCM encoding scheme does achieve (arbitrarily nearly) the optimal bit rate $\bar{n}_\epsilon(\mathcal{B})$. This is shown by the following argument, which uses methods that are similar to those used in [**?**], with more generality.

The first issue to be taken care of is the instability of the expansion (**??**), as stated in Section **??**. We can circumvent this problem easily in the following way. We choose a real number $\lambda > 1$ and sample $x(t)$ at the points $t_n = n/\lambda$, $n \in \mathbb{Z}$, thereby obtaining the sequence $x_n := x(\frac{n}{\lambda})$, $n \in \mathbb{Z}$. When we reconstruct x from these sample values, we have more flexibility since the sample values are redundant. If φ is any function such that its Fourier transform $\hat{\varphi}$ satisfies

$$\hat{\varphi}(\xi) = \begin{cases} 1, & |\xi| \le \pi \\ 0, & |\xi| \ge \lambda\pi, \end{cases} \tag{2.8}$$

and is sufficiently smooth for $|\xi| \in [\pi, \lambda\pi]$, then we have the uniformly convergent reconstruction formula (see, e.g., [**?**, p. 10] when φ is a Schwartz function, or [**?**])

$$x(t) = \frac{1}{\lambda} \sum_{n\in\mathbb{Z}} x_n \varphi\Big(t - \frac{n}{\lambda}\Big). \tag{2.9}$$

Let the sequence of numbers $C_{n,\lambda}(\varphi)$, $n \in \mathbb{Z}$, be defined by

$$C_{n,\lambda}(\varphi) := \sup_{0\le t<\frac{1}{\lambda}} \Big|\varphi\Big(t - \frac{n}{\lambda}\Big)\Big|. \tag{2.10}$$

Note that since $\hat{\varphi}$ is compactly supported, φ is infinitely differentiable. If, moreover, $\hat{\varphi}$ is chosen to be twice differentiable, then $|\varphi(t)| = O(|t|^{-2})$; therefore we have

$$C_\lambda(\varphi) := \sum_{n\in\mathbb{Z}} C_{n,\lambda}(\varphi) < \infty. \tag{2.11}$$

For each $\lambda > 1$, let $\mathcal{F}_\lambda$ be the class of reconstruction kernels φ such that (**??**) and (**??**) hold. We restrict ourselves to the class

$$\mathcal{F} := \bigcup_{\lambda>1} \mathcal{F}_\lambda.$$

Next, we define a family of PCM encoding-decoding pairs. For each real number $\lambda > 1$, we select an arbitrary reconstruction filter $\varphi_\lambda \in \mathcal{F}_\lambda$. We set $C_{n,\lambda} := C_{n,\lambda}(\varphi_\lambda)$ and $C_\lambda := C_\lambda(\varphi_\lambda)$. For any interval $I = [a, b]$, let $\bar{I}_{\lambda,m} := [a - M, b + M]$ where $M := M_{\lambda,m}$ is chosen so that

$$\sum_{|n|\ge\lambda M} C_{n,\lambda} \le 2^{-m} C_\lambda. \tag{2.12}$$

We define the encoder $E_{\mathrm{PCM}} := E_{\mathrm{PCM}}(I,\lambda,m)$ by its output

$$E_{\mathrm{PCM}}\, x := \big\{ b_i(x_n)\ ,\ i = 0,\dots,m,\ n/\lambda \in \bar{I}_{\lambda,m},\ n\in\mathbb{Z}\big\}$$

for $x \in \mathcal{B}_I$. On the other hand, we define the output of the decoder D_{PCM} to be

$$\tilde{x}(t) := \frac{1}{\lambda}\sum_{n\in\lambda\bar{I}_{\lambda,m}} [x_n]_m \varphi_\lambda\Big(t-\frac{n}{\lambda}\Big)\,, \tag{2.13}$$

where we use the shorthand

$$[r]_m := b_0(r)\sum_{i=1}^{m} b_i(r)2^{-i}.$$

We can easily bound the distortion for this encoding-decoding. Decompose the error signal $x-\tilde{x}$ into two components as

$$x - \tilde{x} = e_1 + e_2,$$

where

$$e_1(t) := \frac{1}{\lambda}\sum_{n\in\lambda\bar{I}_{\lambda,m}} \left(x_n - [x_n]_m\right)\varphi_\lambda\Big(t-\frac{n}{\lambda}\Big)$$

and

$$e_2(t) := \frac{1}{\lambda}\sum_{n\notin\lambda\bar{I}_{\lambda,m}} x_n \varphi_\lambda\Big(t-\frac{n}{\lambda}\Big)\,.$$

For the first error component, we use the sample error bound $|x_n - [x_n]_m| \le 2^{-m}$ together with the bound $\lambda > 1$ to obtain

$$|e_1(t)| \le 2^{-m}\sum_{n\in\lambda\bar{I}_{\lambda,m}} \Big|\varphi_\lambda\Big(t-\frac{n}{\lambda}\Big)\Big| \le 2^{-m}\sum_{n\in\mathbb{Z}} C_{n,\lambda} \le 2^{-m}C_\lambda.$$

Similarly for the second component, we note that $t\in I$ and $n\notin\lambda\bar{I}_{\lambda,m}$ implies $|t-\frac{n}{\lambda}|\ge M$ so that

$$|e_2(t)| \le \sum_{n\notin\lambda\bar{I}_{\lambda,m}} \Big|\varphi_\lambda\Big(t-\frac{n}{\lambda}\Big)\Big| \le \sum_{|k|\ge\lambda M} C_{k,\lambda} \le 2^{-m}C_\lambda.$$

Putting these two estimates together we obtain

$$\|x-\tilde{x}\|_{L^\infty(I)} \le C_\lambda 2^{-m+1} \tag{2.14}$$

uniformly for all $x\in\mathcal{B}$. The number of bits $n(E_{\mathrm{PCM}}(I,\lambda,m),\mathcal{B}_I)$ used in this encoding satisfies the bound

$$n(E_{\mathrm{PCM}}(I,\lambda,m),\mathcal{B}_I) \le \big\lceil \lambda|\bar{I}_{\lambda,m}|(m+1)\big\rceil = \big\lceil \lambda(|I|+2M)(m+1)\big\rceil.$$

Apart from I, note that these encoders have two parameters, λ and m. Given any $\epsilon>0$, we now choose $\lambda=\lambda_\epsilon$ and $m=m_\epsilon$ as follows: We first define the set

$$\Lambda_\epsilon := \left\{\lambda\ :\ \log_2(8C_\lambda) \le \left(\log_2\frac{1}{\epsilon}\right)^{1/2}\right\};$$

here the function $\left(\log_2 \frac{1}{\epsilon}\right)^{1/2}$ can be replaced by any other function that is $o\left(\log_2 \frac{1}{\epsilon}\right)$ but still diverges to ∞ as $\epsilon \longrightarrow 0$. Note that $\Lambda_\epsilon \supset \Lambda_{\epsilon'}$ if $\epsilon < \epsilon'$ and

$$\bigcup_{\epsilon>0} \Lambda_\epsilon = \lim_{\epsilon \longrightarrow 0} \Lambda_\epsilon = (1, \infty).$$

In particular, Λ_ϵ is nonempty for all sufficiently small ϵ. We can therefore select $\lambda_\epsilon \in \Lambda_\epsilon$ such that $\lambda_\epsilon \longrightarrow 1$ as $\epsilon \longrightarrow 0$; in other words, we have

$$\lambda_\epsilon = 1 + o(1) \tag{2.15}$$

and

$$\log_2(8C_{\lambda_\epsilon}) \leq \left(\log_2 \frac{1}{\epsilon}\right)^{1/2}. \tag{2.16}$$

We next define m_ϵ to be

$$m_\epsilon := \min\left\{m \in \mathbb{N} \;:\; C_{\lambda_\epsilon} 2^{-m+1} \leq \epsilon\right\}. \tag{2.17}$$

The definition of m_ϵ and the bound (**??**) imply that $E_{\text{PCM}}(I, \lambda_\epsilon, m_\epsilon)$ belongs to the class $\mathcal{C}_\epsilon(\mathcal{B}_I)$ of encoders that achieve maximum distortion ϵ for all I. Denoting the family $\{E_{\text{PCM}}(I, \lambda_\epsilon, m_\epsilon)\}_I$ by $\mathbf{E}_{\text{PCM}}^{[\epsilon]}$, we have

$$\bar{n}(\mathbf{E}_{\text{PCM}}^{[\epsilon]}) := \limsup_{|I| \to \infty} \frac{n(E_{\text{PCM}}(\lambda_\epsilon, m_\epsilon), \mathcal{B}_I)}{|I|} \leq \lambda_\epsilon(m_\epsilon + 1). \tag{2.18}$$

By definition of m_ϵ, we know that

$$C_{\lambda_\epsilon} 2^{-m_\epsilon+1} \leq \epsilon < C_{\lambda_\epsilon} 2^{-m_\epsilon+2};$$

hence

$$m_\epsilon + 1 \leq \log_2\left(\frac{8C_{\lambda_\epsilon}}{\epsilon}\right) \leq \log_2 \frac{1}{\epsilon} + \left(\log_2 \frac{1}{\epsilon}\right)^{1/2} = (1 + o(1)) \log_2 \frac{1}{\epsilon}.$$

This implies, together with (**??**) and (**??**), that

$$\bar{n}(\mathbf{E}_{\text{PCM}}^{[\epsilon]}) \leq (1 + o(1)) \log_2 \frac{1}{\epsilon};$$

thus PCM encoders achieve the Kolmogorov entropy given by (**??**).

2.3 Robustness

As explained in Section **??**, circuit implementation of A/D conversion, regardless of what encoding scheme is used, necessarily incorporates at least one encoding step that makes a transition from analog values to a discrete range, and that is accompanied by some "uncertainty". In other words, the encoder used by the A/D convertor is not known exactly; rather, it can be any element of a restricted class $\mathcal{E}$ of encoders (corresponding to the different values of the parameters within the uncertainty interval). We therefore have an additional distortion in the encoding+decoding procedure that is due to our inability to know precisely which encoder E in $\mathcal{E}$ was used for the encoding. For the decoding, we use the same decoder for the whole class $\mathcal{E}$. In analogy with the analysis in section 2.1 we introduce then

$$d(\mathcal{B}_I; \mathcal{E}, D) := \sup_{E\in\mathcal{E}} d(\mathcal{B}_I, E, D) = \sup_{E\in\mathcal{E}} \sup_{x\in\mathcal{B}_I} \|x - DEx\|_\infty \ .$$

The bit cost for $\mathcal{E}$ is now defined by

$$\mathbf{n}(\mathcal{E}, \mathcal{B}_I) := \sup_{E\in\mathcal{E}} n(E, \mathcal{B}_I) = \sup_{E\in\mathcal{E}} \sup_{x\in\mathcal{B}_I} \#(Ex) \ .$$

To generalize (**??**), we must minimize the bit cost for $\mathcal{E}$ over the choices $(\mathcal{E}, D)$. This requires that we have several possible $\mathcal{E}$ at our disposal. Suppose that for each encoding algorithm for $\mathcal{B}_I$ we can identify the components that have to be analog, and the realistic tolerances on their parameters, within e.g. a fixed \$-budget for the construction of the corresponding circuit. Then each combination (algorithm and parameter-tolerances) defines a family $\mathcal{E}$. Define $\boldsymbol{\mathcal{E}}_\tau$ to be the set of all these families, where the shorthand τ stands for the tolerance limits imposed by the analog components. Then the minimum bit-cost that can achieve distortion ϵ for a given parameter tolerance level τ is

$$n_{\epsilon,\tau}(\mathcal{B}_I) := \inf\{\mathbf{n}(\mathcal{E}, I); \exists D : \mathcal{B} \to \mathcal{B}_I \text{ and } \exists \mathcal{E} \in \boldsymbol{\mathcal{E}}_\tau \text{ so that } d(\mathcal{B}_I; \mathcal{E}, D) \le \epsilon\} \ .$$

By definition, we have $n_{\epsilon,\tau}(\mathcal{B}_I) \ge n_\epsilon(\mathcal{B}_I)$.

There is a fundamental difference between $n_\epsilon(\mathcal{B}_I)$ and $n_{\epsilon,\tau}(\mathcal{B}_I)$ in the following sense. The encoders E over which the infimum is taken to obtain $n_\epsilon(\mathcal{B}_I)$ can be viewed as just abstract maps from $\mathcal{B}_I$ to $\mathcal{B}$. On the other hand, in order to define the classes $\mathcal{E}$ over which we have to optimize to define $n_{\epsilon,\tau}(\mathcal{B}_I)$, we have to consider a concrete implementation for each encoder E and consider the class of its variations within realistic constraints on the analog components. For this reason, unlike $n_\epsilon(\mathcal{B}_I)$ which was closely related to Kolmogorov entropy, it is not straightforward how to relate $n_{\epsilon,\tau}(\mathcal{B}_I)$ to a more geometric quantity. Even so, it continues to make sense to define an average bit-cost,

$$\bar{n}_{\epsilon,\tau}(\mathcal{B}) = \lim_{|I|\to\infty} \frac{n_{\epsilon,\tau}(\mathcal{B}_I)}{|I|} \ ,$$

and to consider its behavior as a function of ϵ.

In what follows, we shall consider the special class $\boldsymbol{\mathcal{E}}_\tau$ in which every $\mathcal{E}$ is constructed as follows. We start from an encoder E along with an algorithmic implementation of it. We denote this pair together by $(E, \mathtt{b})$, where $\mathtt{b}$ stands for the "block diagram", i.e., the algorithm, that is used to implement E. We assume that $\mathtt{b}$ contains one or more quantizers of type (**??**) or (**??**) but no other types of quantizers (all other operations are continuous). We then consider all the encoders that have the same block diagram $\mathtt{b}$, but with Q_0 or Q_1 replaced by a $Q_{0,\rho}$ or $Q_{1,\rho}$ (see (**??**, **??**)) with $|\rho| < \delta$, or by $Q^{\mathrm{f}}_{0,\delta}$ or $Q^{\mathrm{f}}_{1,\delta}$ (see (**??**, **??**)). We call this class $\mathcal{E}_{[E,\mathtt{b},\delta]}$, and we define $\boldsymbol{\mathcal{E}}_{[\delta]}$ to be the set of all $\mathcal{E}_{[E,\mathtt{b},\delta]}$ for which the encoder E maps $\mathcal{B}_I$ to $\mathcal{B}$, and has an implementation with a block diagram $\mathtt{b}$ that satisfies the criteria above. We can then define $n_{\epsilon,[\delta]}(\mathcal{B}_I)$ and $\bar{n}_{\epsilon,[\delta]}(\mathcal{B})$ to be $n_{\epsilon,\tau}(\mathcal{B}_I)$ and $\bar{n}_{\epsilon,\tau}(\mathcal{B})$, respectively, for the choice $\boldsymbol{\mathcal{E}}_\tau = \boldsymbol{\mathcal{E}}_{[\delta]}$.

For the PCM encoder E_{PCM} defined above in (**??**, **??**), the substitution of $Q_{0,\rho}$ for Q_0, with $|\rho| < \delta$ and $\delta > 0$, leads to (see the examples given at the end of the Introduction)

$$d(\mathcal{B}_I; \mathcal{E}_{[\text{PCM},\delta]}, D) \geq \delta \ ,$$

for every decoder D, regardless of how many bits we allow in the encoding. Consequently, for $\epsilon < \delta$, it is impossible to achieve $d(\mathcal{B}_I; \mathcal{E}_{[\text{PCM},\delta]}, D) \leq \epsilon$. It follows that considering $\mathcal{E}_{[\text{PCM},\delta]}$ does not lead to a finite upper bound for $n_{\epsilon,[\delta]}(\mathcal{B}_I)$ or the average $\bar{n}_{\epsilon,[\delta]}(\mathcal{B})$ when $\epsilon < \delta$; hence the PCM-encoder is not "robust".

In the next two sections we discuss two pairs $(E, \mathtt{b})$ that are robust, in the sense that they can achieve $d(\mathcal{B}_I; \mathcal{E}_{[E,\mathtt{b},\delta]}, D) \leq \epsilon$ for arbitrarily small $\epsilon > 0$, regardless of the value of δ, provided an extra number of bits is spent, compared to what is dictated by the Kolmogorov entropy. Both constructions will provide us with finite upper bounds for $n_{\epsilon,[\delta]}(\mathcal{B}_I)$ and $\bar{n}_{\epsilon,[\delta]}(\mathcal{B})$. Our purpose next is to study how small these quantities can be made.

3 The Error Correction of Sigma-Delta Modulation

In this section, we shall consider encoders given by Sigma-Delta ($\Sigma\Delta$) Modulation; these behave differently from PCM when imprecise quantization is used. We start by describing the simplest case of a first order $\Sigma\Delta$ encoder E, i.e. the case where the quantizer is given by the unperturbed (**??**). Then we show that this encoder is robust, in contrast to the PCM encoder: if we replace the precise quantizer of (**??**) by a quantizer of type (**??**) , then the encoding can nevertheless be made arbitrarily accurate. This leads to a discussion of what we can conclude from this example, and from others in [**?**][**?**], about $n_{\epsilon,[\delta]}(\mathcal{B}_I)$ and $\bar{n}_{\epsilon,[\delta]}(\mathcal{B})$.

3.1 First order $\Sigma\Delta$ modulation with a precise quantizer

We again choose $\lambda > 1$ (although now we are interested in the limit $\lambda \to \infty$ rather than $\lambda \to 1$). We continue with the notation $x_n := x(\frac{n}{\lambda})$ as above. Given $(x_n)_{n\in\mathbb{Z}}$, the algorithm creates a bitstream $(b_n)_{n\in\mathbb{Z}}$, where $b_n \in \{-1, 1\}$, whose running sums track those of x_n. There is an auxiliary sequence $(u_n)_{n\in\mathbb{Z}}$ which evaluates the error between these two running sums. Given the interval of interest $I = [a, b]$, we take $u_n = 0$ for $\frac{n}{\lambda} < (a - M)$, where $M := M_\lambda$ is to be specified later. For $\frac{n}{\lambda} \in \bar{I}_\lambda = [a - M, b + M]$, we define

$$u_{n+1} := u_n + x_n - b_n \quad , \tag{3.1}$$

where

$$b_n := Q_0(u_n) \ , \tag{3.2}$$

and Q_0 is the quantizer defined by (**??**). The $\Sigma\Delta$ encoder $E_{\Sigma\Delta} := E_{\Sigma\Delta}(I, \lambda)$ maps x to the bitstream $(b_n)_{n\in\mathbb{Z}}$. We see that $E_{\Sigma\Delta}$ appropriates one bit for each sample, which corresponds

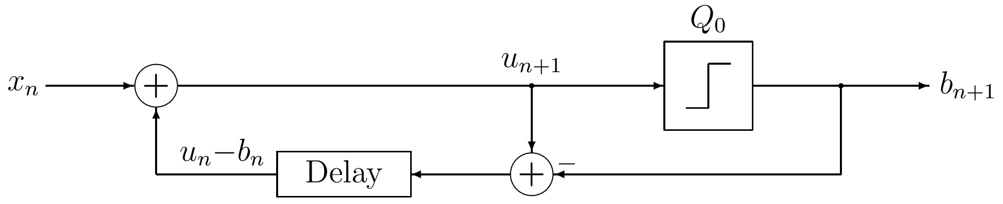

Figure 2: *Block diagram for the simplest, first order $\Sigma\Delta$ convertor, implementing the algorithm $u_{n+1} = u_n + x_n - \text{sign}(u_n)$. (Note that this is nearly the same diagram as in Figure* **??***; the only difference is that the multiplier by 2 has been removed.)*

to λ bits per Nyquist interval, therefore resulting in $|\bar{I}|\lambda$ bits for the whole signal x. Figure 2 shows a block diagram for $E_{\Sigma\Delta}$.

To decode the $\Sigma\Delta$ bitstream, we use the recovery formula (**??**) applied to the bit sequence (b_n), but now with a fixed kernel $\varphi \in \mathcal{F}$. (As in the PCM case, we shall make λ depend on ϵ, but as $\epsilon \to 0$, we shall have $\lambda_\epsilon \to \infty$, so that a fixed φ can satisfy constraint (**??**) for all λ_ϵ; see below.) This gives

$$\tilde{x}(t) := D_{\Sigma\Delta}\left((b_n)_{n\in\mathbb{Z}}\right) := \frac{1}{\lambda} \sum_{n\in\lambda\bar{I}_\lambda} b_n \, \varphi\Big(t - \frac{n}{\lambda}\Big) . \tag{3.3}$$

An easy derivation allows us to bound the approximation error. We first note that $|u_n| \le 2$ for all n. (This is a special case of Theorem 3.1 below; see also [**?**].) Next, we decompose the error signal $x - \tilde{x}$ into two components e_1 and e_2 exactly as it was done in Section 2.2. Given this decomposition, we have

$$\begin{aligned}
|e_1(t)| &= \frac{1}{\lambda}\left| \sum_{n\in\lambda\bar{I}_\lambda} (x_n - b_n)\, \varphi\Big(t - \frac{n}{\lambda}\Big)\right| \\
&= \frac{1}{\lambda}\left| \sum_{n\in\lambda\bar{I}_\lambda} (u_{n+1} - u_n)\, \varphi\Big(t - \frac{n}{\lambda}\Big)\right| \\
&= \frac{1}{\lambda}\left| \sum_{n\in\lambda\bar{I}_\lambda} u_n \left[\varphi\Big(t - \frac{n}{\lambda}\Big) - \varphi\Big(t - \frac{n-1}{\lambda}\Big)\right] + O(1)\right| \\
&\le \frac{2}{\lambda}\sum_n \left|\varphi\Big(t - \frac{n}{\lambda}\Big) - \varphi\Big(t - \frac{n-1}{\lambda}\Big)\right| + O\left(\frac{1}{\lambda}\right) \\
&\le \frac{2}{\lambda}\Big(\int |\varphi'(t)|dt + O(1)\Big) . \qquad (3.4)
\end{aligned}$$

Note that in the above calculation, the $O(1)$ term is a boundary term that appears after the summation by parts, and depends only on the maximum value of φ. Based on this estimate, we choose φ to have a derivative with finite integral.

We have not yet specified how we choose M_λ. It suffices to choose M_λ so that

$$\sum_{|n|\geq \lambda M_\lambda} C_{n,\lambda}(\varphi) \leq 1$$

(see Section 2.2 for the definition of $C_{n,\lambda}(\varphi)$); then we guarantee that

$$\|e_2\|_{L^\infty(I)} \leq \frac{1}{\lambda}$$

and thus

$$\|x - \tilde{x}\|_{L^\infty(I)} \leq \frac{C_\varphi}{\lambda} \tag{3.5}$$

with the constant C_φ depending only on the choice of φ. By choosing $\lambda_\epsilon = \lceil C_\varphi/\epsilon \rceil$ it follows that

$$\begin{aligned} \bar{n}(\mathbf{E}^{[\epsilon]}_{\Sigma\Delta}) &:= \limsup_{|I|\to\infty} \frac{n(E_{\Sigma\Delta}(I,\lambda_\epsilon))}{|I|} \\ &= \lim_{|I|\to\infty} \frac{\lambda_\epsilon(|I| + 2M_{\lambda_\epsilon})}{|I|} \\ &= \lambda_\epsilon \\ &= (1 + o(1))\frac{C_\varphi}{\epsilon}, \end{aligned} \tag{3.6}$$

where $\mathbf{E}^{[\epsilon]}_{\Sigma\Delta}$ denotes the family of encoders $\{E_{\Sigma\Delta}(I,\lambda_\epsilon))\}_I$.

One may wonder if a better bit-rate can be given by improving the error estimate (**??**). It is possible to improve the exponent of λ when the signal x is fixed, but not uniformly in x. In fact, it follows from the analysis in [**?**] that there exists an absolute constant $c > 0$ such that

$$d(\mathcal{B}_I; E_{\Sigma\Delta}(I,\lambda), D) \geq \frac{c}{\lambda} \tag{3.7}$$

for *every* decoder D, linear or nonlinear.

We see in (**??**) that for an average bit rate λ, first order $\Sigma\Delta$ modulation results in an accuracy that is much worse (inversely proportional to a power of λ) than what would be achieved if PCM was used (exponential decay in λ). In other words, the bit rate changes from logarithmic to linear in the reciprocal of the maximum allowable distortion. However, $\Sigma\Delta$ encoders have the remarkable property to be impervious to imprecision in the quantizer used in the circuit implementation; we shall see this next.

3.2 First order $\Sigma\Delta$ modulation with an imprecise quantizer

Suppose that in place of the quantizer Q_0 of (**??**), we use the imprecise quantizer $Q^{\mathrm{f}}_{0,\delta}$ of (**??**) in the first order Sigma-Delta algorithm; i.e., we have the same block diagram as in Figure 2, but Q_0 is replaced by $Q^{\mathrm{f}}_{0,\delta}$. (Note that this includes the case where Q_0 is replaced by a $Q_{0,\rho}$ where ρ can conceivably vary with n within the range $[-\delta,\delta]$.) Using these quantizers will result in a different bitstream than would be produced by using Q_0. In place of the auxiliary sequence u_n of (**??**), which would be the result of exact quantization, we obtain the sequence u_n^{f}, which satisfies $u_n^{\mathrm{f}} = 0$ for $\frac{n}{\lambda} < a - M$ and

$$u_{n+1}^{\mathrm{f}} = u_n^{\mathrm{f}} + x_n - b_n^{\mathrm{f}}, \quad n \in \lambda \bar{I} \tag{3.8}$$

where

$$b_n^{\mathrm{f}} = Q_{0,\delta}^{\mathrm{f}}(u_n^{\mathrm{f}}).$$

Theorem **??**, below, shows that if we are given any $\delta > 0$, then the error of at most δ, in each implementation of flaky quantization in the $\Sigma\Delta$ encoder, will not affect the distortion bound (**??**) save for the constant C_0.

Theorem 3.1 *Suppose that $\Sigma\Delta$ modulation is implemented by using, at each occurence, a quantizer $Q_{0,\delta}^{\mathrm{f}}$, with $\delta > 0$, in place of Q_0. If the sequence $(b_n^{\mathrm{f}})_{n \in \lambda \bar{I}}$ is used in place of $(b_n)_{n \in \lambda \bar{I}}$ in the decoder* (**??**)*, the result is a function $\tilde{x}^{\mathrm{f}}$ which satisfies*

$$\|x - \tilde{x}^{\mathrm{f}}\|_{L^\infty(I)} \leq C_{\delta,\varphi} \lambda^{-1} \tag{3.9}$$

with $C_{\delta,\varphi} = C_\varphi(1 + \delta/2)$ and C_φ the constant in (**??**) .

Proof: This theorem was proved in [**?**]. We do not repeat its simple proof in detail, but give a sketch here. The main idea is to establish the following bound:

$$|u_n^{\mathrm{f}}| \leq 2 + \delta; \tag{3.10}$$

this can be proved by induction on n by the following argument. It is clear for $n < \lambda(a - M)$. Assume that (**??**) has been shown for $n \leq N$. If $u_N^{\mathrm{f}} \leq \delta$, then $b_N^{\mathrm{f}} := -1$. Consequently, $x_N - b_N^{\mathrm{f}} \in [0, 2]$ and hence $u_{N+1}^{\mathrm{f}} = u_N^{\mathrm{f}} + x_N - b_N^{\mathrm{f}} \in [-2 - \delta, 2 - \delta] \subset [-(2 + \delta), 2 + \delta]$. A similar argument applies if $u_N^{\mathrm{f}} > \delta$. Finally, if $|u_n^{\mathrm{f}}| < \delta$, then $|x_N - b_N^{\mathrm{f}}| \leq 2$ implies $|u_N^{\mathrm{f}} + x_N - b_N^{\mathrm{f}}| \leq 2 + \delta$. Therefore we have advanced the induction hypothesis. This proves (**??**) for all n. The remainder of the proof uses summation by parts again to obtain (**??**) (see [**?**]). ∎

Hence, despite the use of a quantizer that is inherently inaccurate, we nevertheless obtain an arbitrarily precise approximation to the original signal. The average bitrate of this algorithm can be computed exactly as in subsection **??**, leading to the estimate

$$\bar{n}_{\epsilon,[\delta]}(\mathcal{B}) \leq \frac{C_\delta}{\epsilon},$$

where C_δ is the infimum of $C_{\delta,\varphi}$ over all admissable reconstruction filters φ.

We shall see next that this estimate can be improved significantly.

3.3 Higher order $\Sigma\Delta$ modulators

Higher order $\Sigma\Delta$ modulators provide faster decay of the reconstruction error as $\lambda \to \infty$. In [**?**], an explicit family of arbitrary order single-bit $\Sigma\Delta$ modulators is constructed with the property that the reconstruction error is bounded above by $O(\lambda^{-k})$ for a k-th order modulator. (The k-th order modulator in this context corresponds to replacing the first order recursion (**??**) by a k-th order recursion $\Delta^k u_n = x_n - b_n$, where the estimates follow from a choice of b_n that ensures that the u_n are uniformly bounded.)

Note that because of the single-bit quantization, the average bit-rate is still equal to the sampling rate λ. The error bound $O(\lambda^{-k})$ involves a hidden constant C'_k (for a fixed, admissable filter φ). Following the same steps as in the first order case we can bound the average bit-rate in terms of the error tolerance ϵ as

$$\bar{n}_{\epsilon,[\delta]}(\mathcal{B}) \le {C'_k}^{1/k}\epsilon^{-1/k} .$$

One can still optimize this over k by adapting to ϵ. For this, one uses the quantitative bound $C'_k \le C''\gamma^{k^2}$ for some $\gamma > 1$ (see [**?**]), to conclude that

$$\bar{n}_{\epsilon,[\delta]}(\mathcal{B}) \le \inf_k \; {C'_k}^{1/k}\epsilon^{-1/k} \le C'''2^{\Gamma\sqrt{\log_2(1/\epsilon)}} , \tag{3.11}$$

where $\Gamma = 2\sqrt{\log_2(\gamma)} > 0$. Although this is an improvement over the bound we obtained from the first order $\Sigma\Delta$ scheme, it is still a long way from the optimal average bitrate, which, as we saw in section 2, is proportional to $\log_2(1/\epsilon)$. In the next subsection we shall examine a different type of $\Sigma\Delta$ modulators constructed in [**?**] which employ filters that are more general than pure integration. These modulators achieve exponential accuracy in the bit-rate while being robust with respect to small quantization errors, hence yielding much better bounds on the average bit-rate $\bar{n}_{\epsilon,[\delta]}(\mathcal{B})$.

3.4 Exponentially accurate $\Sigma\Delta$ modulator with an imprecise quantizer

The following result can be thought of as a generalization of the robustness result for the first order $\Sigma\Delta$ algorithm.

Theorem 3.2 *Let $\delta \ge 0$ and consider the nonlinear difference equation*

$$u_n = \sum_{j\ge1} h_j u_{n-j} + x_n - b_n, \qquad n \ge n_0, \tag{3.12}$$

where

$$b_n = Q^{\mathrm{f}}_{0,\delta}\left(\sum_{j\ge1} h_j u_{n-j} + x_n\right), \tag{3.13}$$

and $|u_n| \le 1+\delta$ for $n < n_0$. If $(1+\delta)\|h\|_1 + \|x\|_\infty + \delta \le 2$, then $\|u\|_\infty \le 1+\delta$.

Proof: The proof is by induction. Let the coefficients $(h_j)_{j\ge1}$ and the input (x_n) satisfy $(1+\delta)\|h\|_1 + \|x\|_\infty + \delta \le 2$ and assume that $|u_l| \le 1+\delta$ for all $l < n$. Let $w_n = \sum_{j\ge1} h_j u_{n-j} + x_n$ so that $u_n = w_n - b_n$. First, it is clear that

$$|w_n| \le (1+\delta)\|h\|_1 + \|x\|_\infty \le 2-\delta.$$

Without loss of generality, let $\delta > 0$. Three cases appear:

• If $w_n \in [\delta, 2-\delta]$, then $b_n = 1$, so that $u_n = w_n - 1 \in [-1+\delta, 1-\delta]$.

• If $w_n \in [-2+\delta, -\delta]$, then $b_n = -1$, so that $u_n = w_n + 1 \in [-1+\delta, 1-\delta]$.

• If $w_n \in (-\delta, \delta)$, then $|u_n| \le |w_n| + |b_n| < \delta + 1$, regardless of the value $b_n \in \{-1, 1\}$ is assigned by the flaky quantizer.

The case $\delta = 0$ is similar except the last case does not exist. ∎

In [?], a two-parameter family of $\Sigma\Delta$-filters $\{h^{(k,\sigma)} : k \in \mathbb{Z}_+, \sigma \in \mathbb{Z}_+\}$ was designed that achieves exponential accuracy in the bit-rate. More specifically, the following results hold:

(i) $\|h^{(k,\sigma)}\|_1 \le \cosh(\pi\sigma^{-1/2})$ uniformly in k,

(ii) Let $x \in \mathcal{B}$. Given any interval I and for each λ, if we set $x_n := x(n/\lambda)$, $n \in \lambda\bar{I}_\lambda$ as before, and if b_n satisfies (**??**) with $h = h^{(k,\sigma)}$, then

$$\sup_{t\in I}\left| x(t) - \frac{1}{\lambda}\sum_{n\in\lambda\bar{I}_\lambda} b_n \varphi\Big(t - \frac{n}{\lambda}\Big)\right| \le \|u\|_\infty (C_\varphi \sigma k)^k \lambda^{-k}.$$

The exponential accuracy in λ is achieved by choosing, for each λ, the optimal order k. This optimization results in the error bound

$$\|x - \tilde{x}\|_{L^\infty(I)} \le C_0 2^{-r\lambda}$$

where $r = r(\sigma) = 1/(\sigma\pi\log 2) > 0$.

Theorem **??** allows us now to achieve the same result with flaky quantizers. Let us say that a $\Sigma\Delta$-filter $h := (h_j)$ is δ-tolerant for $\mathcal{B}(\pi, \mu, \mathbb{R})$ if the stability condition of Theorem **??** is satisfied, i.e., if $(1+\delta)\|h\|_1 + \mu + \delta \le 2$. Given $\mu < 1$ and $\delta > 0$, let $\sigma_0 = \sigma_0(\mu, \delta)$ to be the smallest positive integer σ such that

$$(1+\delta)\cosh(\pi\sigma^{-1/2}) + \mu + \delta \le 2.$$

It is easy to check that σ_0 exists if and only if $\mu + 2\delta < 1$ and

$$\sigma_0(\mu, \delta) = \left\lceil \left[\frac{1}{\pi}\cosh^{-1}\left(\frac{2-\mu-\delta}{1+\delta}\right)\right]^{-2}\right\rceil.$$

Hence we deduce that the $\Sigma\Delta$-filters $h^{(k,\sigma_0)}$, $k \in \mathbb{Z}_+$, are all δ-tolerant for $\mathcal{B}(\pi, \mu, \mathbb{R})$. If for each reconstruction error tolerance ϵ we choose $\lambda = \lambda_\epsilon$ such that $C_0 2^{-r(\sigma_0)\lambda} = \epsilon$, then we find that

$$\begin{aligned} \bar{n}_{\epsilon,[\delta]}(\mathcal{B}(\pi, \mu, \mathbb{R})) &\le \lambda_\epsilon \\ &= \sigma_0(\mu, \delta)\pi \ln\frac{C_0}{\epsilon} \\ &= [\sigma_0(\mu, \delta)\pi\log 2 + o(1)]\,\bar{n}_\epsilon(\mathcal{B}). \end{aligned} \tag{3.14}$$

In this estimate, the smallest value of σ_0 is found in the limit $\mu \to 0$ and $\delta \to 0$. A simple calculation reveals this value to be 6; the corresponding minimum increase factor in the average bit-rate (when this algorithm is used) is $6\pi \log 2 \approx 13.07$.

The question arises whether there exist other redundant encoders which can achieve self correction for quantization error *and* exponential accuracy in terms of the bit rate (with possibly a better constant) by means of other approaches than heavy oversampling. In the next section, we show that this is indeed the case. Not only will we obtain such encoders; we will also show that they are asymptotically optimal in the limit as $\delta \to 0$.

4 Beta-Encoders with Error Correction

We shall now show that it is possible to obtain exponential bit rate performance while retaining quantization error correction by using what we shall call *beta-encoders*[1]. These encoders will introduce redundancy in the representation of the signals, but will, contrary to $\Sigma\Delta$ modulation, start from samples at the Nyquist rate (or rather slightly over the Nyquist rate to ensure stability). The essential idea is to replace the binary representation $\sum_{i=1}^{\infty} b_i 2^{-i}$ of a real number $y \in [0,1)$ by a representation of type $\sum_{i=1}^{\infty} b_i \beta^{-i}$, where $1 < \beta < 2$, and where each of the b_i can still take only the values 0 or 1. Introducing $\gamma := 1/\beta$, it is more convenient to write

$$y = \sum_{i=1}^{\infty} b_i \gamma^i \ . \tag{4.1}$$

In fact there are many such representations; this redundancy will be essential to us.

4.1 Redundancy in beta-encoding

We start by showing one strategy that gives, for each $y \in [0,1]$, an assignment of bits, i.e. values $b_i := b_i(y) \in \{0,1\}$, such that (**??**) is satisfied. Given y, we define $u_1 = \beta y$, and we set $b_1 = 1$ if $u_1 \geq 1$, $b_1 = 0$ if $u_1 < 1$, i.e. $b_1 = Q_1(u_1)$, with Q_1 as defined in (**??**). We then proceed recursively for $i \geq 1$, defining $u_{i+1} = \beta(u_i - b_i)$ and $b_{i+1} = Q_1(u_{i+1})$. It is easy to see that u_i always lies in $[0,\beta]$. By the definition of the u_i,

$$\sum_{i=1}^{L} u_i \gamma^i - \sum_{i=1}^{L} b_i \gamma^i = \sum_{i=1}^{L} (u_i - b_i)\beta\gamma^{i+1} = \sum_{i=1}^{L} u_{i+1}\gamma^{i+1} \ ,$$

so that from above,

$$\sum_{i=1}^{L} b_i \gamma^i = \gamma u_1 - u_{L+1}\gamma^{L+1} = y - u_{L+1}\gamma^{L+1} \ ,$$

which implies, since $u_{L+1} \in [0,\beta]$,

$$0 \leq \varepsilon_L(y) := y - \sum_{i=1}^{L} b_i \gamma^i \leq \gamma^L.$$

[1]This name is motivated by the term β-expansion in [**?**].

The successive approximations $\sum_{i=1}^{L} b_i \gamma^i$ to y thus achieve an accuracy that decays exponentially in L, albeit slower than the usual binary successive approximations. In fact, the algorithm is very similar to the binary successive approximation algorithm in subsection 2.2; the only difference is the substitution of multiplication by β for the doubling operation we had there.

Although we will be ultimately interested in applying this algorithm to numbers in $[0,1]$ only, it is useful to note that it works just as well for numbers y in the larger interval $[0,(\beta-1)^{-1})$. In this case, one finds that the corresponding u_i lie in $[0,\beta(\beta-1)^{-1})$, resulting in the bound

$$0 \le \varepsilon_L(y) < (\beta-1)^{-1}\gamma^L.$$

We can exploit this observation to define a different beta-expansion of $y \in [0,1]$, by the following argument: If $y < \gamma(\beta-1)^{-1}$, then we know, by the observation just made, that there exists a bit sequence $(d_j)_{j\in\mathbb{N}}$ so that $y = \gamma \sum_{j=1}^{\infty} d_j \gamma^j$, yielding a beta-expansion of y whose first bit is equal to zero. Motivated by this, let us set $\mathring{b}_1 = 0$ if $u_1 := \beta y < (\beta-1)^{-1}$, and $\mathring{b}_1 = 1$ if $\beta y \ge (\beta-1)^{-1}$. In either case, one checks easily that $y - \mathring{b}_1\gamma < \gamma(\beta-1)^{-1}$. (In the first case, there is nothing to prove; in the second case, we have $y-1 \le 1-\gamma = \gamma(\beta-1) < \gamma(\beta-1)^{-1}$.) Since $\beta(y-\mathring{b}_1\gamma) < (\beta-1)^{-1}$, we can repeat the argument with y replaced by $\beta(y-\mathring{b}_1\gamma) =: \gamma u_2$ to define $\mathring{b}_2$ and so on. If we extend the definition of Q_1 by setting $Q_\nu := \frac{1}{2}[1+Q_0(\cdot-\nu)]$ for $\nu \ge 1$, then we can write the above iteration explicitly as

$$\begin{array}{rcl} u_1 &=& \beta y \\ \mathring{b}_1 &=& Q_{(\beta-1)^{-1}}(u_1) \\ \text{for } i \ge 1 \ : \ u_{i+1} &=& \beta(u_i - \mathring{b}_i) \\ \mathring{b}_{i+1} &=& Q_{(\beta-1)^{-1}}(u_{i+1}). \end{array} \tag{4.2}$$

Because the resulting u_i are all in the range $[0,\beta(\beta-1)^{-1})$, we still have the error bound

$$0 \le y - \sum_{i=1}^{L} \mathring{b}_i \gamma^i < (\beta-1)^{-1}\gamma^L.$$

We can understand the difference between the two algorithms given so far by visualizing the definitions of $b_1(y)$ and $\mathring{b}_1(y)$ as y increases from 0 to 1, as shown in Figure 3. As long as y is smaller than γ, both algorithms keep the first bit equal to zero. However, if y is greater than or equal to γ, then the first algorithm seizes its chance and allocates $b_1 = 1$, and starts over again with $\beta y - 1$. We call this algorithm “greedy” because the bits are assigned greedily; in every iteration, given the first j bits $b_1, \ldots, b_j$, the algorithm will set b_{j+1} equal to 1 as long as $\gamma^{j+1} + \sum_{i=1}^{j} b_i \gamma^i$ is less than y. The second algorithm does the opposite: if it can get away with keeping $\mathring{b}_1 = 0$, i.e., if $y \le \sum_{i=2}^{\infty} \gamma^i = \gamma(\beta-1)^{-1}$, it is content with $\mathring{b}_1 = 0$; only when y exceeds this value, and hence the $\mathring{b}_i$ for $i \ge 2$ would not be able to catch up on their own, does this algorithm set $\mathring{b}_1 = 1$. This approach is then continued in the assignment of all the remaining bits. Because it only sets a bit equal to 1 when forced to do so, we call this algorithm the “lazy” algorithm.[2] Laziness has its price:

[2]For some of the recent mathematical accounts of greedy and lazy expansions, see, e.g., [?], [?].

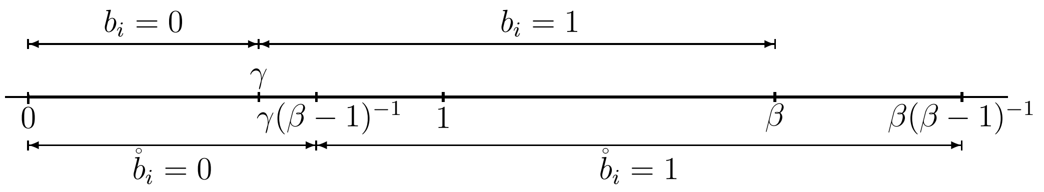

Figure 3: *For the choice $\beta = 1.8$, this picture shows the ranges for y (when $i = 1$) or γu_{i-1} (when $i \geq 2$) that lead to the different values of b_i (top) and $\mathring{b}_i$ (bottom).*

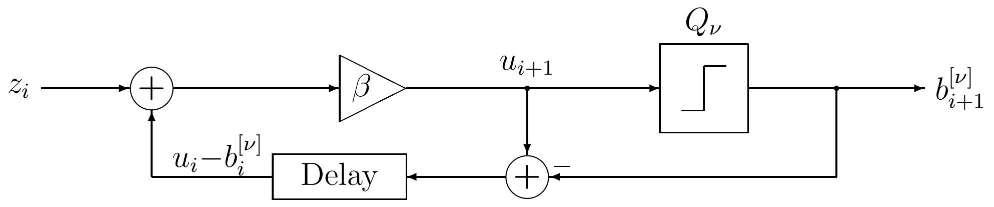

Figure 4: *With input $z_0 = y \in [0,1), z_i = 0$ for $i > 0$ and "initial conditions" $u_0 = b_0 = 0$ the output $(b_i^{[\nu]})_i$ of this block diagram gives the beta-representation for y defined by the quantizer $Q_\nu := [1 + Q_0(\cdot - \nu)]/2$, with $\nu \in [1, (\beta-1)^{-1}]$. For $\nu = 1$, this is the "greedy" scheme, for $\nu = (\beta-1)^{-1}$, it is the "lazy" scheme.*

in the lazy algorithm the upper bound on the u_i is larger than in the greedy algorithm. The difference between the two algorithms becomes more marked as β decreases and gets close to 1; on the other hand, their difference becomes less marked when β increases and gets close to 2; for $\beta = 2$ they coincide with each other and the base-2 expansion.

For $1 < \beta < 2$, there is a range for y or γu_i where we can choose whether to set the next bit equal to 0 or 1: both options are viable. The greedy and lazy algorithms gave us the two extreme possibilities, but we could also decide to take a "cautious" algorithm, using the quantizer Q_ν with ν between the two extremes used in the greedy and lazy algorithms, i.e., $\nu \in [1, (\beta-1)^{-1}]$. Let us denote the resulting bit sequence by $(b_i^{[\nu]})_{i\in\mathbb{N}}$, where we have added a superscript to indicate the dependence of the bits on ν. Figure 4 shows a block diagram for this algorithm, which is again very similar to the one in Figure 1: the two differences are the substitution of β for 2 in the multiplier, and the freedom in the choice of the quantizer. An easy inductive argument shows that the u_i in this algorithm are uniformly bounded by $\nu\beta$, so that the approximation error satisfies

$$0 \leq y - \sum_{i=1}^{L} b_i^{[\nu]} \gamma^i \leq \nu\gamma^L$$

for all $y \in [0,1]$. In the next subsection we shall see how this type of redundancy benefits us for the construction of robust encoders.

4.2 Beta-encoding with an imprecise quantizer

The approaches of the last subsection can be reduced to the observation that, given $y \in [0,1]$, no matter what bits b_i, $i = 1, \ldots, m$, have been assigned, then, as long as

$$y - \frac{\gamma^{m+1}}{1-\gamma} \leq \sum_{i=1}^{m} b_i \gamma^i \leq y, \tag{4.3}$$

there is a bit assignment b_k, $k > m$, which, when used with the previously assigned bits, will exactly recover y.

We used this observation to show that greedy, lazy and cautious algorithms, with quantizers that "toggle" at different points, all give bit sequences that can be used in a beta-expansion for y. We shall use it here even more generally, to prove the following theorem.

Theorem 4.1 *Let $\beta \in (1,2)$, $y \in [0,1)$, and $1 \leq \nu_0 < \nu_1 \leq (\beta-1)^{-1}$. Define $Q^{\mathrm{f}}_{[\nu_0,\nu_1]} := [Q^{\mathrm{f}}_{0,\frac{\nu_1-\nu_0}{2}}\left(\cdot - \frac{\nu_0+\nu_1}{2}\right) + 1]/2$, i.e. $Q^{\mathrm{f}}_{[\nu_0,\nu_1]}$ is a flaky quantizer such that*

$$Q^{\mathrm{f}}_{[\nu_0,\nu_1]}(z) = \begin{cases} 0 & \text{if } z \leq \nu_0 \\ 1 & \text{if } z \geq \nu_1 \\ 0 \text{ or } 1 & \text{if } z \in (\nu_0, \nu_1). \end{cases}$$

Define u^{f}_i, b^{f}_i by the following algorithm:

$$\begin{aligned} u^{\mathrm{f}}_1 &= \beta y \\ b^{\mathrm{f}}_1 &= Q^{\mathrm{f}}_{[\nu_0,\nu_1]}(u^{\mathrm{f}}_1) \\ \text{for } i \geq 1 \ :\ u^{\mathrm{f}}_{i+1} &= \beta(u^{\mathrm{f}}_i - b^{\mathrm{f}}_i) \\ b^{\mathrm{f}}_{i+1} &= Q^{\mathrm{f}}_{[\nu_0,\nu_1]}(u^{\mathrm{f}}_{i+1}). \end{aligned} \tag{4.4}$$

Then, for all $L \in \mathbb{N}$,

$$0 \leq y - \sum_{i=1}^{L} b^{\mathrm{f}}_i \gamma^i \leq \nu_1 \gamma^L.$$

Proof: The definition of the u^{f}_i implies that

$$y - \sum_{i=1}^{L} b^{\mathrm{f}}_i \gamma^i = y - \sum_{i=1}^{L} (u^{\mathrm{f}}_i - \gamma u^{\mathrm{f}}_{i+1})\gamma^i = y - \gamma u^{\mathrm{f}}_1 + \gamma^{L+1} u^{\mathrm{f}}_{L+1} = \gamma^{L+1} u^{\mathrm{f}}_{L+1}. \tag{4.5}$$

Next, we use an induction argument to show that $0 \leq u^{\mathrm{f}}_i \leq \nu_1 \beta$ for all i. Clearly, $u^{\mathrm{f}}_1 = \beta y \leq \beta \leq \beta \nu_1$, since $\nu_1 \geq 1$. Suppose now that $0 \leq u^{\mathrm{f}}_i \leq \nu_1 \beta$. If $u^{\mathrm{f}}_i \leq \nu_0$, then $b^{\mathrm{f}}_i = 0$, and $u^{\mathrm{f}}_{i+1} = \beta u^{\mathrm{f}}_i \leq \beta \nu_0 \leq \beta \nu_1$. If $u^{\mathrm{f}}_i \geq \nu_1$, then $b^{\mathrm{f}}_i = 1$, and $0 \leq u^{\mathrm{f}}_{i+1} = \beta(u^{\mathrm{f}}_i - 1) \leq \beta[\beta\nu_1 - 1] \leq \beta\nu_1$, where we have used $\nu_1 \geq 1$ and $\nu_1 \leq (\beta-1)^{-1}$ in the first and last inequality, respectively. If $u^{\mathrm{f}}_i \in (\nu_0, \nu_1)$, then b^{f}_i can be either 0 or 1; in either case we have $0 \leq \beta(\nu_0 - 1) \leq \beta(u^{\mathrm{f}}_i - b^{\mathrm{f}}_i) = u^{\mathrm{f}}_{i+1} \leq \beta u^{\mathrm{f}}_i \leq \beta\nu_1$, implying $u^{\mathrm{f}}_{i+1} \in [0, \beta\nu_1]$. This concludes the proof. ∎

In practice, we are given the tolerance δ on the quantizer, and we then have to determine β from δ, rather than the reverse. The precise statement is given in the following easy corollary to Theorem 4.1:

Corollary 4.2 *Given $\delta > 0$, define $\beta = 1 + (2\delta + 1)^{-1}$. Let $y \in [0, 1]$ be arbitrary. Let $(b_i^{\mathrm{f}})_{i\in\mathbb{N}}$ be the bit sequence defined by the iteration (**??**), with $\nu_0 = 1$, $\nu_1 = (\beta - 1)^{-1}$, i.e., $Q^{\mathrm{f}}_{[\nu_0,\nu_1]} = [Q^{\mathrm{f}}_{0,\delta}(\cdot - (1 + \delta)) + 1]/2 = Q^{\mathrm{f}}_{1,\delta}(\cdot - \delta)$. Then*

$$0 \le y - \sum_{i=1}^{m} b_i^{\mathrm{f}} \beta^{-i} \le (2\delta + 1)\beta^{-m}.$$

We have thus achieved the goal we set ourselves at the start of this section: we have an algorithm that starts from samples at (slightly above) the Nyquist rate, and that has sufficient redundancy to allow the use of an imprecise quantizer, without significantly affecting the exponential accuracy in the number of bits that is achieved by using a precise quantizer in the same algorithm. In the next subsection we look at the corresponding average bitrate.

4.3 Bitrate for beta-encoding with an imprecise quantizer

As in the PCM case, we start from (**??**). We want to have an accurate representation of a signal $x \in \mathcal{B}_I$, with $I = [a, b]$, starting from an imprecise beta-quantization of its samples. We assume that $\delta > 0$ is given, and we define $\beta = 1 + (2\delta + 1)^{-1}$. For given sampling rate λ and number of bits per sample m, the encoder $E_{\beta-\mathrm{PCM}}(\lambda, m)$ takes samples $x_n = x\left(\frac{n}{\lambda}\right)$, from the enlarged interval $\bar{I}_{\lambda,m} = [a - M, b + M]$, where $M = M_{\lambda,m}$ is chosen such that (**??**) holds (with 2^{-m} replaced by β^{-m}). For each $n \in \lambda\bar{I}_{\lambda,m}$, we compute the first m bits, b_i^{f}, $i = 1, \ldots, m$ of the flaky beta-quantization, described in Corollary **??**, of the numbers $y_n = (1 + x_n)/2 \in [0, 1]$, and we define

$$\begin{aligned} [x_n]_m &:= 2[y_n]_m - 1 := 2\left(\sum_{i=1}^{m} b_i^{\mathrm{f}} \beta^{-i}\right) - 1, \\ \tilde{x}(t) &= \frac{1}{\lambda} \sum_{n\in\lambda\bar{I}} [x_n]_m \, \varphi_\lambda\Big(t - \frac{n}{\lambda}\Big). \end{aligned}$$

Then we obtain, analogously to the analysis in subsection **??**, and using Corollary **??**, the error estimate

$$\|x - \tilde{x}\|_{L^\infty(I)} \le C_\lambda (2\delta + 2)\beta^{-m}. \tag{4.6}$$

As before, the total number of bits used to obtain this accuracy is bounded by $\lceil m(2M + |I|)\lambda\rceil$. For each $\epsilon > 0$, we choose $\lambda = \lambda_\epsilon$ and $m = m_\epsilon$ exactly in the same way as it was done in subsection **??** (again replacing 2 by β wherever appropriate). This selection results in the average bit-rate bound

$$\bar{n}(\mathbf{E}^{[\epsilon]}_{\beta-\mathrm{PCM}}) \le (1 + o(1)) \log_\beta \frac{1}{\epsilon},$$

or in other words, noting that $\beta = 1 + (2\delta + 1)^{-1}$,

$$\bar{n}_{\epsilon,[\delta]}(\mathcal{B}) \le \left(\log_2 \left[1 + \frac{1}{2\delta + 1}\right]\right)^{-1} \bar{n}_\epsilon(\mathcal{B}) \ . \tag{4.7}$$

This estimate is a significant improvement over the result of subsection **??** in that first, it is valid for all $\delta > 0$, and second, it recovers the optimal bit-rate $\bar{n}_\epsilon(\mathcal{B})$ in the limit as $\delta \to 0$.

5 Discussion

The two examples of robust encoders presented here exhibit a large amount of redundancy in the representation of the signal $x(t)$: many different bit sequences can in fact correspond to the same signal. It is this feature that makes it possible to still obtain good approximations even though the imprecise quantizers produce other bit sequences than the precise ones. This suggests that in order to develop other schemes that may improve on the average bitrate under imprecise analog-to-digital conversion, one has to build in redundancy from the start.

Although (**??**) is a significant improvement over (**??**), it would be suprising if it were already optimal (for each δ). One could argue that, in some sense, "imprecise" quantizers are similar to a noisy channel; one would then expect, by analogy with Shannon's result, that $\bar{n}_{\epsilon,[\delta]}(\mathcal{B}) = \bar{n}_{\epsilon}(\mathcal{B})$, even for arbitrary $\delta > 0$ (as opposed to only in the limit for $\delta \to 0$, as in (**??**)). We do not know whether this analogy is valid, and/or whether this equality holds.

It should be emphasized that all our results are asymptotic. The limit for $|I| \to \infty$ is fairly realistic, given the enormous difference in scale between the sampling period and the duration in time of the signals of interest. The limits for very large λ (for $\Sigma\Delta$ modulation), or λ very close to 1 (for β-encoders) are less realistic.

Finally, although the β-encoders give such a startling improvement in the average bit-rate, it is clear that many other aspects have to be taken into account as well before this promise can be realized in practice. In particular, analog implementations of the coder described in subsection 4.3 are bound to have imprecise values of β as well as imprecise quantizers. (In [**?**] it is shown how imprecise values of β can be taken into account as well.) Moreover, any realistic approach should also take into account upper bounds, that are imposed a priori, on all internal variables, such as the u_i; this is in contrast with the upper bounds on the u_i that follow from the theoretical analysis, which may be too large to be possible in reality. The analysis in this paper has not addressed any of these questions.

6 Summary

Nyquist rate A/D converters based on the successive approximation method are known to be susceptible to component mismatch error, whereas $\Sigma\Delta$ modulators are not. An analysis of the robustness of the $\Sigma\Delta$ modulator to component mismatch leads to the construction of two alternate algorithms that achieve exponential bitrate (as opposed to a much slower rate for simple $\Sigma\Delta$) yet are still robust under component mismatch. One of these algorithms is a "filtered $\Sigma\Delta$ modulator", with a appropriate filter; the other is a successive approximation algorithm, based on the β-expansion of a number. Both are robust to circuit mismatch error. The Kolmogorov rate-distortion framework is extended to incorporate component mismatch error and the algorithms described provide upper bounds on what is ultimately achievable within this framework.

References

[1] J. C. Candy, "A Use of Limit Cycle Oscillations to Obtain Robust Analog-to-Digital Converters," *IEEE Trans. Commun.* , vol. COM-22, No. 3, pp. 298-305, March 1974.

[2] C. C. Cutler, "Transmission System Employing Quantization," U.S. Patent 2927962, Mar. 8, 1960.

[3] I. Daubechies, R. DeVore, "Reconstructing a Bandlimited Function From Very Coarsely Quantized Data: A Family of Stable Sigma-Delta Modulators of Arbitrary Order", *Ann. of Math.*, vol. 158, no. 2, pp. 679–710, Sept. 2003.

[4] I. Daubechies, Ö. Yılmaz, preprint.

[5] C. S. Güntürk, "One-Bit Sigma-Delta Quantization with Exponential Accuracy," *Comm. Pure Appl. Math.*, vol. 56, pp. 1608–1630, no. 11, 2003.

[6] H. Inose and Y. Yasuda, "A Unity Bit Coding Method by Negative Feedback," *Proc. IEEE,*, vol. 51, pp. 1524-1535, Nov. 1963.

[7] C. S. Güntürk, J. C. Lagarias, and V. A. Vaishampayan,"On the Robustness of Single-Loop Sigma-Delta Modulation," IEEE Trans. Inform. Th., vol. 47, No. 5, pp. 1735-1744, July 2001.

[8] A. N. Karanicolas, H.-S. Lee and K. L. Bacrania, "A 15-b 1-Msample/s Digitally Self-Calibrated Pipeline ADC", IEEE Journal of Solid-State Circuits, vol. 28, No. 12, pp. 1207-1215, Dec. 1993.

[9] A. N. Kolmogorov and V. M. Tikhomirov, "ϵ-entropy and ϵ-capacity of sets in function spaces," *Uspehi*, 14(86):3-86, 1959 (also: AMS Translations, Series 2, Vol. 17, 1961, pp. 277-364).

[10] S. H. Lewis and P. R. Gray, "A Pipelined 5-Msample/s 9-bit Analog-to-Digital Converter," IEEE Journal of Solid State Circuits, vol. sc-22, No. 6, pp. 954-961, Dec. 1987.

[11] Y. Meyer, "Wavelets and Operators," Cambridge University Press, 1992.

[12] W. Parry, "On β-expansions of real numbers," Acta. Math. Acad. Sci. Hungar., vol. 15, pp. 95-105, 1964.

[13] V. Komornik and P. Loreti, "Subexpansions, superexpansions and uniqueness properties in non-integer bases," Period. Math. Hungar. 44 (2002), no. 2, pp. 197–218.

[14] K. Dajani and C. Kraaikamp, "From greedy to lazy expansions and their driving dynamics," Expo. Math., 20, (2002), no. 4, pp. 315–327.