Communications in the Observation Limited Regime

by

Manish Bhardwaj

B.A.Sc. (Honors), Nanyang Technological University (1997)
S.M., Massachusetts Institute of Technology (2001)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Doctor of Philosophy in Computer Science and Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

June 2009

Author .....
Department of Electrical Engineering and Computer Science
May 22, 2009

Certified by .....
Anantha Chandrakasan
Joseph F. and Nancy P. Keithley Professor of Electrical Engineering
Thesis Supervisor

Accepted by .....
Terry P. Orlando
Chairman, Department Committee on Graduate Theses

Communications in the Observation Limited Regime

by

Manish Bhardwaj

Submitted to the Department of Electrical Engineering and Computer Science
on May 22, 2009, in partial fulfillment of the
requirements for the degree of
Doctor of Philosophy in Computer Science and Engineering

Abstract

We consider the design of communications systems when the principal cost is observing the channel, as opposed to transmit energy per bit or spectral efficiency. This is motivated by energy constrained communications devices where sampling the signal, rather than transmitting or processing it, dominates energy consumption. We show that sequentially observing samples with the maximum *a posteriori* entropy can reduce observation costs by close to an order of magnitude using a (24,12) Golay code. This is the highest performance reported over the binary input AWGN channel, with or without feedback, for this blocklength.

Sampling signal energy, rather than amplitude, lowers circuit complexity and power dissipation significantly, but makes synchronization harder. We show that while the distance function of this non-linear coding problem is intractable in general, it is Euclidean at vanishing SNRs, and *root* Euclidean at large SNRs. We present sequences that maximize the error exponent at low SNRs under the peak power constraint, and under all SNRs under an average power constraint. Some of our new sequences are an order of magnitude shorter than those used by the 802.15.4a standard.

In joint work with P. Mercier and D. Daly, we demonstrate the first energy sampling wireless modem capable of synchronizing to within a ns, while sampling energy at only 32 Msamples per second, and using no high speed clocks. We show that traditional, minimum distance classifiers may be highly sensitive to parameter estimation errors, and propose robust, computationally efficient alternatives. We challenge the prevailing notion that energy samplers must accurately shift phase to synchronize with high precision.

Thesis Supervisor: Anantha Chandrakasan
Title: Joseph F. and Nancy P. Keithley Professor of Electrical Engineering

Acknowledgments

I am indebted to Anantha for his support, encouragement, the freedom to pursue problems, and the collaborative lab environment from which this work emerged. I am grateful to my committee members Dave Forney and Lizhong Zheng for their feedback and encouragement. An extra thank you to Dave, who taught me communications design and coding. I am also thankful to Anant Sahai at Berkeley for his encouragement in pursuing the observation limited setting.

Thanks to my collaborators Pat and Denis! I learnt from both of them, and the thesis is better as a result. Thanks to Vivienne, Maryam, Rana, Amrita and Naveen for being willing movie and dinner partners. Thanks to the many friends who reviewed or helped with this work, including Joyce, Julius, Maryam, Raj and Sashi. Thanks to Rex, Eugene, and Raj — my first friends at MIT! Thank you Kush and Lindi, and Kathleen! Thank you Bill, for your deep commitment to public service, and grinding it out in Bihar!

I would not have made it without Creighton's unwavering friendship, or the love and encouragement of my brother Ashu. How do I thank my parents, Renu and Manoj, for their sacrifices, hard work and love that went in raising Ashu and me? I hope we can lead lives worthy of their love and toil.

This work was funded by DARPA HI-MEMS program (Contract # FA8650-07-C-7704), and in part an IBM Research Fellowship, and a grant from the NSF.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

From an information theorist's perspective, communications design is the quest to equal Shannon's ultimate bound on the rate of reliable communication. From a practitioner's perspective, it is building systems that meet specific cost and performance criteria. Sometimes these goals are perfectly aligned, as in deep-space communications. Bandwidth and receiver processing power are virtually unlimited, and the designer's main concern is the severely power limited transmitter on the spacecraft. Massey reckons that every dB in coding gain saved expeditions in the 1960s an estimated $1 million [38]!

The huge explosion in wireless terminals has made battery lifetime a critical measure of communications systems performance [11]. Systems designers have a finite bucket of energy, and care about the *total* electronics energy consumed in the communications chain for every bit that is reliably conveyed. When distances or spectral efficiencies are large, the radiated power dominates signal processing power (the latter usually dominated by radio-frequency circuits). In such cases, capacity achieving schemes also maximize battery lifetime [57]. However, such prescriptions fail for a rapidly growing class of short-range, low-spectral-efficiency wireless links, such as recently standardized by 802.15.4a[29]. In such systems, the radiated power may be less than 0.1 mW whereas analog and radio-frequency front-ends dissipate 10s of mW. Hence, the traditional theoretical focus on the transmitted power does not yield the most battery efficient solution. Both the circuits and information theory communi-

ties have recognized this mismatch, and issued the same prescription. On the circuits front, it has been recognized that analog signal processing consists of an irreducible component, independent of data rate. Hence, prior work has prescribed increasing data rates to amortize this fixed cost [56, 16]. This is done by increasing the spectral efficiency via higher order modulation, and discarding available degrees of freedom, which is contrary to classical information theory. As constellation sizes grow, the rate dependent component of signal processing power overcomes the irreducible bias, and further increases are not beneficial. Discarding available channel uses makes the transmit signal peakier. Massaad *et al.* have proved that when the capacity problem is reformulated to account for a power dissipation component that is independent of radiated power, bursty transmission that does not use all available degrees of freedom is indeed capacity achieving [37].

While useful, these prescriptions may have limited mileage in practice. Peaky transmissions are problematic from both a circuit and regulatory viewpoint. For instance, pulsed 15.4a UWB transmitters already operate at the peak power limit imposed by semiconductor technology [58]. Also, supporting multiple amplitude levels may significantly increase the cost of the transmitter and receiver, negating the very premise of these solutions. What options are available to a practitioner when further increases in peak power or higher modulation orders are not possible, but plenty of spare degrees of freedom are available? Are there other approaches that allow trading degrees of freedom or transmit power for better battery efficiency?

We will argue that fundamentally different insights and tradeoffs are enabled by considering a channel that looks like deep-space, but with the role of the transmitter and receiver interchanged. Consider the communications system in figure 1-1.

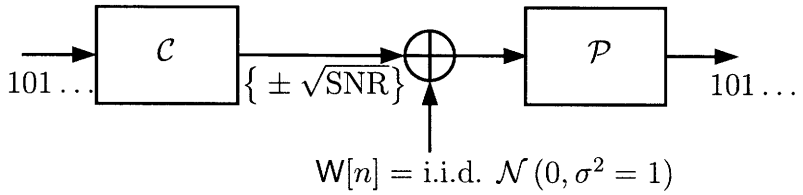

Figure 1-1: Setup of the coding under observation cost problem.

Information is conveyed over a discrete-time, binary-input AWGN channel. The

system uses a code $\mathcal{C}$ and BPSK modulation. A procedure $\mathcal{P}$ observes noisy channel outputs and infers the information bits. The observation cost is equal to the *expected* number of samples observed by $\mathcal{P}$ to decode an information bit.

Problem 1.1 (Coding under receive cost constraints)**.** What choice of $\mathcal{C}$ and $\mathcal{P}$ minimizes the expected cost per information bit under a specified SNR and BER constraint?

Note that the transmitter can send as many coded bits as it desires. What matters is that the receiver judiciously pick the samples it observes. The next chapter deals with this problem in detail. We will see that "receiver" oriented capacity is unchanged, and a conventional capacity achieving code can realize the minimum receive cost. But, the performance versus complexity landscape is dramatically altered, and simple codes with adaptive channel sampling outperform traditional ones. The techniques developed here draw from the theory of experiment design, where the number of observations are minimized under a specified reliability constraint by carefully picking from a set of available experiments. These techniques can be applied to different modulation and coding schemes. In comparison, the second part of this thesis considers a particular technique to reduce observation cost — sampling energy rather than amplitude. The key problem turns out to be synchronizing such an energy sampling receiver, and we device new sequences to address this. The last part of our thesis is joint work with P. Mercier and D. Daly, where we demonstrate a single chip, energy sampling wireless modem that incorporates our new synchronization techniques.

In the remainder of this chapter, we discuss sources of energy dissipation in wireless transceivers in detail to see when the formulation above applies. We end the chapter with a preview of our contributions. Three chapters then follow, dealing with coding under observation costs, synchronization of energy sampling receivers, and the energy sampling wireless modem. We end with conclusions and the outlook for this work.

1.1 Modeling Energy Consumption

The problem of minimizing electronics energy consumption reduces to that of minimizing observation if two conditions are satisfied. First, sampling the received signal should be the dominant source of energy consumption in the communications link. Second, sampling energy must be proportional to the number of samples taken. We now study when these assumptions are valid. Figure 1-2 shows the makeup of an example wireless transmitter and receiver.

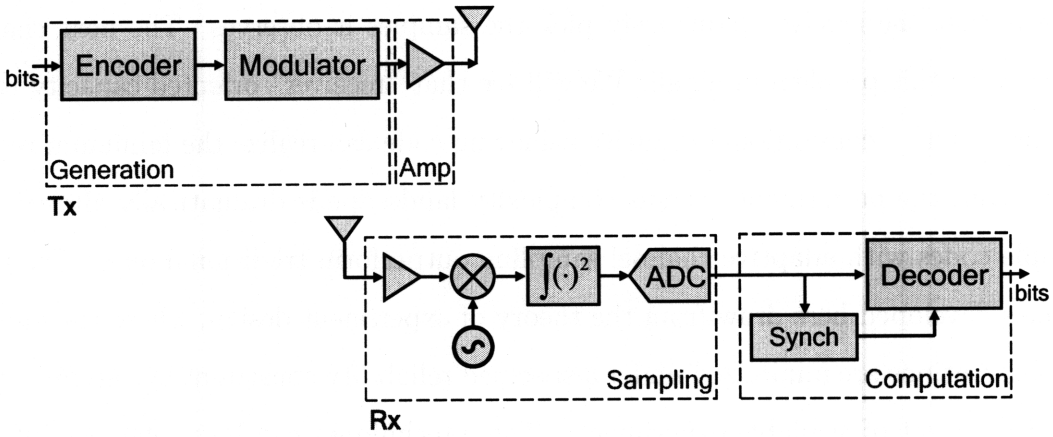

Figure 1-2: Structure of a wireless transmitter and a non-coherent receiver.

We divide transmit electronics into signal generation and amplification. The receiver is divided into signal sampling and the subsequent computation. We use $\mathcal{E}_{\mathrm{A}}$ to denote the *electronics* energy consumed by component *A per information bit.* Hence, $\mathcal{E}_{\mathrm{tx}} = \mathcal{E}_{\mathrm{gen}} + \mathcal{E}_{\mathrm{amp}}$ and $\mathcal{E}_{\mathrm{rx}} = \mathcal{E}_{\mathrm{samp}} + \mathcal{E}_{\mathrm{comp}}$.

1.1.1 Transmit versus Receive Energy

Of the many factors that determine how $\mathcal{E}_{\mathrm{tx}}$ and $\mathcal{E}_{\mathrm{rx}}$ compare, the regulatory limit on output power is usually the governing one. When a high output (radiated) power is permitted, as in wireless LANs, the transmitter is likely to dominate due to $\mathcal{E}_{\mathrm{amp}}$. When regulatory limits are tight, as in UWB systems, the receiver, with its significantly more challenging signal conditioning and processing, dominates consumption.

As an example, Lee, Wentzloff, and Chandrakasan have demonstrated a UWB

system with $\mathcal{E}_{tx}$=43 pJ/bit and $\mathcal{E}_{rx}$=2.5 nJ/bit, i.e. the receiver dominates energy consumption by 60x [33, 58].

1.1.2 Sampling versus Processing Energy

If $\mathcal{E}_{rx} \gg \mathcal{E}_{tx}$, the next question is whether $\mathcal{E}_{samp} \gg \mathcal{E}_{comp}$. This comparison is difficult because notions of complexity, and sources of energy consumption are often markedly different in analog and digital circuits. One prevailing view is that the energy efficiency of digital circuits scales more aggressively than that of analog circuits as technology progresses. Thus the $\mathcal{E}_{samp}/\mathcal{E}_{comp}$ ratio can be expected to grow with time.

The receiver of Lee *et al.* quoted above is among the most energy efficient at the data rates of interest to us [33]. Also, virtually all its energy is consumed in sampling. Hence, 2.5nJ/bit is indicative of the limits of energy efficient sampling in current technology. This may be compared with the 0.18 nJ/bit consumed by a recently reported 64-to-256 state, reconfigurable Viterbi decoder [2]. This illustrates that $\mathcal{E}_{samp} \gg \mathcal{E}_{comp}$ is a tenable assumption, provided the receiver bears a 'reasonable' computational burden.

1.1.3 Circuit Startup Time

Whether $\mathcal{E}_{samp}$ is proportional to the duration of sampling depends on certain important non-idealities. There are physical and architectural constraints on how fast a sampler can turn on and off. These fixed costs are unrelated to the duration of observation. Cho and Chandrakasan have studied the impact of large startup times on the energy efficiency of sensor nodes, and proposed a new carrier synthesis scheme to reduce it by 6x [15].

Lee *et al.* have demonstrated a turn-on time of 3 ns in a system with a minimum observation duration of 30 ns. Hence, a proportional model would be appropriate for this receiver. Note that such rapid turn-on is possible due to the carrierless nature of their receiver.

In summary, we detailed a model for energy consumption in wireless devices and the conditions under which sampling cost is a good proxy for the electronics energy consumed per bit.

1.2 Thesis Contributions

Here is a preview of our contributions.

- We have proposed a new class of communications systems where the receiver adaptively samples the channel to minimize observation costs. While motivated by short-range, low-data-rate, pulsed UWB systems, this is, we hope, a useful addition to a modem designer's toolbox, since it allows trading degrees of freedom for reduced system complexity in a manner that is fundamentally different compared with previous approaches. We hope that future wireless standards will incorporate a "battery emergency" mode that allows the basestation to sacrifice bandwidth to enable a terminal to conserve energy.

- A byproduct of this work is a practical illustration of the power of channel feedback. If the choice made by our adaptive sampling receiver is conveyed to the transmitter via a noiseless feedback link it is possible to breach the cutoff rate over the AWGN channel using the (24,12,8) Golay code. This is the shortest code by an order of magnitude compared with all previously reported schemes, with or without feedback. It is also 7x shorter than Shannon's sphere-packing bound on information blocksize, and illustrates the gains possible when atypical channel behavior is exploited via a maximum entropy sampling scheme.

- We have formally analyzed the problem of synchronization over the energy sampling channel. We have shown that while the distance function of this non-linear coding problem is intractable in general, it is Euclidean at vanishing SNRs, and *root* Euclidean at large SNRs. We have designed sequences that maximize the error exponent at low SNRs under the peak power constraint, and under all SNRs under an average power constraint. These sequences are

often an order of magnitude shorter than the best previous ones. We believe our work merits another look at the sequences currently used by the 802.15.4a standard.

- In joint work with P. Mercier and D. Daly, we have demonstrated the first energy sampling wireless modem capable of synchronizing to within a ns while sampling energy at only 32 Msamples per second, and using no high speed clocks. In addition to the new synchronization sequences, this required developing computationally efficient VLSI classifiers that are robust to parameter estimation errors. We have demonstrated the inadequacy of traditional, minimum distance classifiers for this purpose. It is our hope that this modem will challenge the prevailing notion that energy samplers must accurately shift phase to synchronize with high precision.

Chapter 2

Coding Under Observation Constraints

We will now study the coding problem introduced in the last chapter. We discuss the connection between this problem and coding for channels with feedback. This is followed by a discussion of the sequential probability ratio test (SPRT) as a simple example of the significant gains possible via intelligent sampling techniques. The more general problem of optimally sampling linear block codes is then discussed. We present simulation results that demonstrate the effectiveness of proposed sampling techniques in reducing observation cost. This is followed by an evaluation of the energy, hardware, and bandwidth costs of the proposed schemes. We end with a comparison of our proposed receivers with those that achieve similar receive cost gains using conventional coding techniques.

2.1 Preliminaries

The adjective 'conventional' refers to a fixed length code that is decoded using a ML criterion. In other words, a codeword is transmitted exactly once, the receiver looks at the entire codeword and uses a ML decoder. Our setup is the binary input, discrete time AWGN channel introduced previously (figure 1-1). Given a (n, k) block code, $\mathcal{C}$, we define the *code* rate, $r_{\mathcal{C}} = k/n$, and 'SNR per bit', $E_b/N_0 = \text{SNR}/(2r_{\mathcal{C}})$. The code

rate must be distinguished from the *transmit* rate, $r_{\text{tx}} = k/n'$, where n' is the total number of bits transmitted for k information bits. In conventional systems, $r_{\text{tx}} = r_C$. The r.v. N denotes the number of samples observed prior to stopping. The *receive* rate, r_{rx}, is defined analogously to the transmit rate thus, $r_{\text{rx}} = k/\mathrm{E}[\mathsf{N}]$. Note that, $r_{\text{rx}} \geq r_{\text{tx}}$. The *receive cost per bit* is defined analogously to its transmit counterpart E_b/N_0, thus, $c_{\text{bit}} = \text{SNR}/(2r_{\text{rx}})$. For conventional systems, $c_{\text{bit}} = E_b/N_0$, allowing easy comparison of receive costs. Note that while stating problem 1.1, we equated cost with the number of samples observed. Our definition here is broader because it allows comparing solutions at different SNRs. The observation cost problem is thus one of minimizing c_{bit} under some combination of p_{b}, r_{rx}, and SNR constraints.

2.2 Fundamental Limits, Relation to Feedback

Shannon's capacity theorem states that for reliable communication,

$$\frac{E_b}{N_0} \geq \frac{2^{2r} - 1}{2r}$$

where r is the information rate in bits/channel use or bits/dimension. This limit also applies to receive costs as defined above,

$$c_{\text{bit}} \geq \frac{2^{2r_{\text{rx}}} - 1}{2r_{\text{rx}}}$$

To see why this is so, note that for every system with a certain receive cost, c_{bit}, there exists a system with feedback that can achieve the same transmit cost, $E_b/N_0 = c_{\text{bit}}$. We just convey the bit that the receiver would like to sample next to the transmitter via the noiseless feedback link. Since feedback does not increase the capacity of discrete memoryless channels [46], the limit on receive cost follows.

A feedback link allows exploiting atypical channel behavior — the receiver can stop early under a favorable draw of the noise process. Receive cost schemes share this trait. They sample just enough to reach the desired reliability. That said, the two problems are not identical. As we just demonstrated, every receive cost strategy

corresponds naturally to a feedback coding strategy. But, a feedback coding strategy does not necessarily translate to a receive cost strategy. Feedback schemes that have full knowledge of channel outputs can "look over the decoder's shoulder" and instruct the receiver to stop sampling once the right result is inferred [45]. This is not possible in our setting, where the transmitter has no knowledge of channel outputs. An example of a feedback scheme that does have a receive cost analogue is decision feedback [21], and we will see that this turns out to be asymptotically optimum under certain conditions.

2.3 Traditional Coding

When an uncoded system operating at a specified SNR cannot achieve a desired error-rate, the simplest option is to use a repetition code, i.e., repeat symbols and then average them at the receiver. A more efficient technique is to use a code. Consider for instance, the setup in figure 1-1 operating at a SNR of 10.5 dB. To achieve a BER of 10^{-6}, we must repeat every symbol twice, i.e., use a (2,1) repetition code. We could also use a simple $(4, 3)$ parity-check code which also achieves a BER of around 10^{-6} at this SNR. Hence, we use 4 symbols for every 3 bits instead of 6 — a savings, or coding gain, $\gamma_{\mathcal{C}}(10^{-6})$, of 1.5x (1.7 dB).

A valid question at this juncture is — why not use a capacity achieving scheme like Turbo or LDPC codes with a suitable iterative decoder to achieve the lowest possible receive cost? The reason, elaborated in the following sections, is that the receive problem is governed by a performance versus complexity landscape that is often dramatically different and more favorable when compared with conventional coding.

2.4 The SPRT with Uncoded Data

Fixed length repetition codes offer no gain. However, substantial reduction in observation is possible if we use a decoder with variable stopping times. Consider then a

transmitter that repeats its information bit indefinitely. As Wald demonstrated in the broader context of sequential binary hypothesis testing, the SPRT is the optimum sampling strategy [53, 54].

Definition (SPRT). Consider that $m \geq 1$ observations, $\mathsf{y}_0, \mathsf{y}_1, \ldots, \mathsf{y}_{m-1}$, have been made thus far. The SPRT is defined by the following rule,

$$\text{If } \sum_{i=0}^{m-1} \ell(y_i) = \sum_{i=0}^{m-1} \ln \frac{p(y_i \mid \theta_1)}{p(y_i \mid \theta_2)} \begin{cases} \geq A > 0 & \text{Accept } \theta_1 \\ \leq B < 0 & \text{Accept } \theta_2 \\ \in [B, A] & \text{Continue sampling} \end{cases}$$

where A, B are thresholds, and θ_1, θ_2 are the binary hypotheses.

The SPRT is thus a simple extension of the well-known likelihood ratio test. Instead of observing a fixed number of samples, we stop when the magnitude of log-likelihood exceeds a threshold that reflects the desired confidence. The SPRT is optimum in the strongest possible sense.

Theorem 2.1 (The SPRT is optimum (Wald [55])). *Assume that a SPRT yields error probabilities* $\varepsilon_1, \varepsilon_2$ *and expected sample sizes* $\mathrm{E}_{\theta_1}[\mathsf{N}], \mathrm{E}_{\theta_2}[\mathsf{N}]$ *under the two hypotheses, respectively. Then,* any *sequential procedure* $\mathcal{P}'$ *that realizes* $(\varepsilon_1' \leq \varepsilon_1, \varepsilon_2' \leq \varepsilon_2)$ *obeys* $\mathrm{E}_{\theta_1}[\mathsf{N}'] \geq \mathrm{E}_{\theta_1}[\mathsf{N}]$ *and* $\mathrm{E}_{\theta_2}[\mathsf{N}'] \geq \mathrm{E}_{\theta_2}[\mathsf{N}]$.

Since computing log-likelihoods reduces to summation for the case of i.i.d. Gaussian observations, implementing the SPRT is trivial for the AWGN channel (figure 2-1). The SPRT requires about a third of the observations used by a fixed length repetition code. This is a significant saving with virtually zero baseband costs. As one might expect, we do not need to repeat the bit indefinitely. There is virtually no performance loss by limiting observations to slightly more than that required by a fixed-length receiver using repetition codes (more on this in a later section). So, how does the SPRT compare with conventional coding techniques? Figure 2-2 plots the receive costs of the 'rate (r_{rx}) 1/2 SPRT' and the popular fixed-length

rate ($r_{rx} = r_{tx} = r_C$) 1/2, constraint length, $K = 7$, convolutional code with octal generators [133 171].

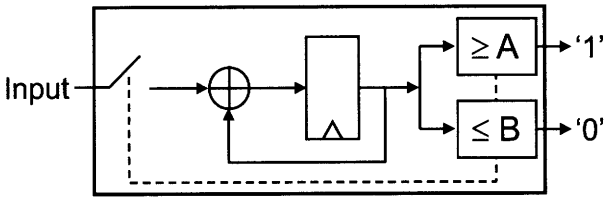

Figure 2-1: The SPRT as decoder for uncoded transmission over the binary AWGN channel.

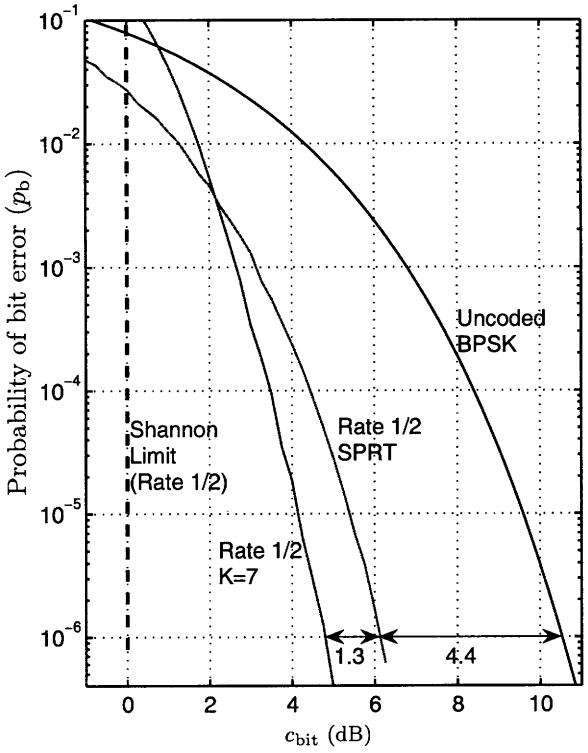

Figure 2-2: Receive costs of the rate 1/2 SPRT and $K = 7$ convolutional code.

The (2,1,$K = 7$) code yields a gain of 5.7 dB. The SPRT achieves a gain of about 4.4 dB. Stated in receive cost terms, the (2,1,7) code reduces observation to roughly a quarter, and the SPRT to roughly a third of that needed by a conventional repetition code operating at the same SNR. However, the SPRT has a hardware complexity that is several orders of magnitude lower than the Viterbi decoder for the (2,1,7) code — 10s of gates compared to 10s of thousands [34]. This example illustrates the dramatic

difference in performance versus complexity when coding under receive, rather than transmit cost constraints.

What is the maximum gain possible via the SPRT? The asymptotic expansion of the expected number of samples yields [14],

$$\mathrm{E}_\theta\,[\mathsf{N}] \doteq \frac{-\ln \varepsilon_\theta}{\mathcal{D}(\mathrm{P}_0 \parallel \mathrm{P}_1)} \quad \text{as } r_{\mathrm{rx}} \to 0 \tag{2.1}$$

where $f(x) \doteq g(x)$ as $x \to a$ implies $\lim_{x\to a} f(x)/g(x) = 1$, P_i are the likelihood functions of channel symbols, and ε_θ is the probability of error under hypothesis θ. We can now compute the asymptotic *sequential* gain, γ_{s}, by comparing the SPRT's error exponent with that of a fixed-length repetition code. For the binary AWGN channel, we have,

$$\gamma_{\mathrm{s}} = \frac{\mathcal{D}(\mathcal{N}(-a,\sigma^2) \parallel \mathcal{N}(a,\sigma^2))}{\mathcal{D}(\mathcal{N}(0,\sigma^2) \parallel \mathcal{N}(a,\sigma^2))} = 4 \equiv 6 \text{ dB} \tag{2.2}$$

Note that the gain is independent of the underlying SNR. Simulations show that $\gamma_{\mathrm{s}}\,(10^{-6})$ for a receive rate of $1/10$ is 5.1 dB, about 0.7 dB better than that for rate $1/2$, and 0.9 dB short of the asymptotic limit.

A similar analysis can be applied to the binary symmetric channel with probability of error p,

$$\gamma_{\mathrm{s}} = \frac{\mathcal{D}(\mathcal{B}(p) \parallel \mathcal{B}(1-p))}{\mathcal{D}(\mathcal{B}(\frac{1}{2}) \parallel \mathcal{B}(1-p))} = \begin{cases} 2 & \text{for } p \to 0 \\ 4 & \text{for } p \to \frac{1}{2} \end{cases}$$

Here, $\mathcal{B}(p)$ is the $(p, 1-p)$ Bernoulli distribution. The gain varies from 3 dB for almost noiseless channels to 6 dB for very noisy ones.

2.5 Sampling Block Codes When $r_{\mathrm{rx}} \to 0$

The natural way to improve the SPRT's receive cost is to encode the transmitted data.

Problem 2.1 (Sampling Block Codes). Consider a transmitter than sends infinite

copies of a codeword drawn from $\mathcal{C}$, a (n, k) code. What sampling procedure minimizes c_{bit} under a r_{rx} and p_{b} constraint?

The problem of choosing from several available coded bits to infer the transmitted codeword is one of optimum experiment design, i.e., classification under a maximum sample size constraint when several types of observations (experiments) are available (see Chernoff's monograph for an accessible treatment [14]). To get some perspective on the intractability of such problems, consider that the optimum solution is not known even for the simpler problem of generalizing the SPRT to more than two hypotheses, with no choice of experiments [20]! We will devote this section to a procedure that is asymptotically optimum (i.e., as $r_{\text{rx}} \to 0$), and the one that follows to schemes that work well with a moderate sample size.

2.5.1 Chernoff's Procedure A

Chernoff initiated the study of asymptotically optimum design and we now state his central result [13, 14]. In what follows, the set of experiments is denoted by $\mathscr{E} = \{e\}$. These are 'pure' experiments that form the basis for randomized experiments, whose set we denote by $\mathscr{E}^*$. Each element in $\mathscr{E}^*$ corresponds to a convex composition of the underlying experiments. Likelihoods must be conditioned on not only the underlying hypothesis, but also the experiment. We will use $\mathrm{P}_{\theta;e} \triangleq p(y \mid \theta; e)$. We will also use a more compact notation for K-L divergence, $\mathcal{D}_e(\theta_i, \theta_j) \triangleq \mathcal{D}(\mathrm{P}_{\theta_i;e} \parallel \mathrm{P}_{\theta_j;e})$.

Definition (Procedure A). Suppose, without any loss of generality, that the ML estimate, $\hat{\Theta} = \theta_0$, after m observations, $\boldsymbol{y} = \{y_0, y_1, \ldots, y_{m-1}\}$, have been made. Then,

I. Pick $e(m+1) = \arg \sup_{e \in \mathscr{E}^*} \inf_{\theta \neq \theta_0} \mathcal{D}_e(\theta_0, \theta)$

II. Terminate when $\min_{j \neq 0} \ell_0(\boldsymbol{y}) - \ell_j(\boldsymbol{y}) > a$.

Here, $\ell_i(\boldsymbol{y})$ is the log-likelihood of of hypothesis θ_i, and a is some suitably chosen positive threshold. Thus, roughly speaking, Procedure A picks the experiment that

maximizes the minimum K-L distance between the most likely hypothesis and the remaining ones. Procedure A yields an expected sample size that generalizes (2.1),

$$\mathrm{E}_\theta[\mathsf{N}] \doteq \frac{-\ln \varepsilon_\theta}{\mathcal{D}_\theta} \quad (\max\ \varepsilon_\theta \to 0) \tag{2.3}$$

$$\text{where } \mathcal{D}_\theta \triangleq \sup_{e \in \mathscr{E}^*} \inf_{\theta' \neq \theta} \mathcal{D}_e(\theta, \theta') \tag{2.4}$$

Theorem 2.2 (Procedure A is Asymptotically Optimum [13, 20]). *Every sequential procedure satisfies,*

$$\mathrm{E}_\theta[\mathsf{N}] \geq \frac{-\ln \varepsilon_\theta}{\mathcal{D}_\theta}(1 + o(1)) \quad (\max\ \varepsilon_\theta \to 0)$$

Note that, unlike the SPRT, Procedure A is guaranteed to be optimum only in an order-of-magnitude sense.

2.5.2 Application to Sampling Codes

We denote the codewords of a block code $\mathcal{C}$ by $\{\boldsymbol{c}_i,\ i = 1, 2, \ldots, M\}$, and the bits of $\boldsymbol{c}_i$ by $c_{ij},\ j = 1, 2, \ldots, n$. The hypothesis θ_i corresponds to $\boldsymbol{c}_i$ being transmitted, and the experiment $e_j,\ j = 1, 2, \ldots, n$, is defined as observing the channel output corresponding to j^{th} bit.

Suppose next that m observations have been made and $\hat{\Theta} = \theta_1$, i.e., codeword 1 is the most likely. Then, we define a K-L distance matrix induced by codeword 1,

$$\boldsymbol{D}_{ij}^{\theta_1} = \mathcal{D}_{e_j}(\theta_1, \theta_i) = \begin{cases} \mathcal{D}(\mathrm{P}_0 \parallel \mathrm{P}_1) & \text{if } c_{1j} \neq c_{ij} \\ 0 & \text{otherwise} \end{cases}$$

It follows that if $\mathcal{C}$ is linear, then $\left\{\boldsymbol{D}^{\theta_i}\right\}$ are isomorphic under row permutations (and equal to a weighted version of the codebook minus the all zero codeword). Hence, for linear codes, the sampling strategy is independent of the currently most likely codeword.

Lemma 2.1. *Uniformly sampling all coordinates is strictly sub-optimum if deleting*

or replicating certain coordinates increases the coding gain of the resulting codebook.

The proof is straightforward and omitted. We believe the converse to be true[1].

Conjecture 2.1. *Uniform sampling is the optimum asymptotic strategy for 'good', linear block codes, i.e., codes for which no set of coordinate deletions or replications can increase the coding gain.*

Uniform sampling achieves,

$$\mathrm{E}[\mathsf{N}] \doteq \frac{-\ln \varepsilon}{\frac{d}{n}\mathcal{D}(\mathrm{P}_0 \parallel \mathrm{P}_1)}$$

Normalizing this to the expected observations per information bit gives,

$$\frac{\mathrm{E}[\mathsf{N}]}{k} \doteq \frac{-\ln \varepsilon}{\gamma_c \mathcal{D}(\mathrm{P}_0 \parallel \mathrm{P}_1)}$$

Hence, the overall asymptotic reduction in sampling costs compared with a fixed length repetition code is $\gamma_c \gamma_s$, i.e., a product of the coding and sequential gain. Note that the feedback analogue of uniform sampling would be *bitwise* decision feedback.

2.6 Sampling Block Codes When $r_{\mathrm{rx}} \nrightarrow 0$

In this section, we discuss procedures better suited to moderate sample sizes, and derive a new scheme to sample codes.

2.6.1 Alternatives to Procedure A

Procedure A suffers from two key drawbacks when the expected number of observations is small. First, it relies on a ML estimate to pick an experiment, and this estimate can be very unreliable in the initial phases of observation. This leads to a poor choice of experiments and wasted samples. Second, in picking an experiment that maximizes the minimum distance between the ML and other hypotheses,

[1] We are tightening a proof by contradiction.

it completely ignores the *a-posteriori* probability of those hypotheses. Thus, a highly unlikely hypothesis might dictate the choice of experiment.

In what follows, we denote the *a-posteriori* probability of a hypothesis by Π_θ, i.e., $\Pi_\theta \triangleq \Pr[\Theta = \theta \mid \{y\}]$, where Θ is the true state of nature. Note that we use $\{y\}$ to denote a *set* of observations from possibly different experiments. This is to distinguish it from $\boldsymbol{y}$ which refers to samples from the same experiment.

Blot and Meeter proposed *Procedure B* which incorporates reliability information in distance calculations [7, 39]. Assuming m observations have been made, the next experiment is picked thus,

$$e(m+1) = \arg \max_{e \in \mathscr{E}} \sum_{\theta} \Pi_\theta \mathcal{D}_e(\hat{\Theta}, \theta) \qquad \text{[Procedure B]}$$

where $\hat{\Theta}$ is the ML hypothesis. Box and Hill proposed a metric that factors in posteriors instead of relying solely on the ML estimate [8],

$$e(m+1) = \arg\max_{e \in \mathscr{E}} \sum_{\theta'} \sum_{\theta} \Pi_{\theta'} \Pi_\theta \left[\mathcal{D}_e(\theta', \theta) + \mathcal{D}_e(\theta, \theta')\right] \quad \text{[Box-Hill]}$$

Note that in cases where K-L distances commute, the Box-Hill procedure can be written as,

$$e(m+1) = \arg\max_{e \in \mathscr{E}} \sum_{\theta'} \Pi_{\theta'} \sum_{\theta} \Pi_\theta \mathcal{D}_e(\theta', \theta)$$

and hence is a straightforward generalization of Procedure B. Chernoff proposed *Procedure M* that weighs posteriors more carefully [13],

$$\arg\max_{e \in \mathscr{E}} \sum_{\theta'} \Pi_{\theta'} \left[\sum\nolimits_{\theta \neq \theta'} \Pi_\theta \mathcal{D}_e(\theta', \theta) \Big/ \sum\nolimits_{\theta \neq \theta'} \Pi_\theta\right] \quad \text{[Procedure M]}$$

It is interesting to note that while these procedures yield better results than Procedure A for practical sample sizes, they are either known to be asymptotically suboptimum, or optimum only under certain constraints [14].

2.6.2 Application to Sampling Codes

The transmitted codeword and its bits are denoted by the r.v.s $\mathbf{c}$ and c_j, $j = 1, 2, \ldots, n$, respectively. Similarly, $(\hat{\mathbf{c}})_j$ denotes a bit of the ML codeword, which must be distinguished from $\hat{c}_j$ which denotes the ML estimate of a bit. We denote codeword and bit posteriors by Π_i and π_j respectively, i.e.,

$$\begin{aligned} \Pi_i &\triangleq \Pr[\mathbf{c} = \boldsymbol{c}_i \mid \{y\}] && i = 1, 2, \ldots, M \\ \pi_j &\triangleq \Pr[\mathsf{c}_j = 1 \mid \{y\}] = \sum\nolimits_{i \in U_j(1)} \Pi_i && j = 1, 2, \ldots, n \end{aligned}$$

where $U_j(x)$ is the set of indices of all codewords whose j^{th} bit is equal to x.

A sampling procedure assigns a distance metric, μ_j, to every coordinate j, and picks the one with the largest metric. We have seen that Procedure A for linear codes leads to uniform sampling, i.e., $\mu_j^{(\mathrm{A})}$ are identical. Procedure B yields,

$$\mu_j^{(\mathrm{B})} = \Pr\left[\mathsf{c}_j \neq (\hat{\mathbf{c}})_j \mid \{y\}\right] = \begin{cases} \pi_j & \text{if } (\hat{\mathbf{c}})_j = 0 \\ 1 - \pi_j & \text{otherwise} \end{cases}$$

Note that if the *a-posteriori* probability of the ML codeword exceeds $1/2$,

$$\mu_j^{(\mathrm{B})} = \min(\pi_j, 1 - \pi_j)$$

which implies that Procedure B picks the bit with the maximum *a-posteriori* entropy. This matches our intuition that the most uncertain bit yields the most information. The more symmetric distance metric of the Box-Hill procedure yields an explicit maximum entropy prescription,

$$\mu_j^{(\mathrm{BH})} = 2\pi_j(1 - \pi_j)$$

Chernoff's Procedure M yields a cumbersome metric,

$$\mu_j^{(\mathrm{M})} = \pi_j \sum\nolimits_{i \in U_j(0)} \frac{\Pi_i}{1 - \Pi_i} + (1 - \pi_j) \sum\nolimits_{i \in U_j(1)} \frac{\Pi_i}{1 - \Pi_i}$$

which reduces to the Box-Hill metric as the ML estimate becomes more reliable,

$$\mu_j^{(\mathrm{M})} \rightarrow \mu_j^{(\mathrm{BH})} \text{ as } \max\ \Pi_i \rightarrow 1$$

In summary, all three procedures (eventually) prescribe observing the bit with the largest *a-posteriori* entropy. We call this *maximum entropy* (ME) sampling.

There is some precedence of using bit reliabilities in the context of hybrid ARQ schemes for iterative decoders. Shea proposed retransmitting the most unreliable *information* bits in order to help the decoder to converge [48]. In our context, this scheme would essentially reduce to the SPRT. Mielczarek and Krzymien have recently proposed a more elaborate metric to label specific bits "non-convergent" based on forward and backward parameters in the BCJR algorithm [41]. Their scheme does not reduce blocklengths or receiver complexity compared to previous ones, but does reduce the amount of feedback required.

2.7 Examples of Sampling to Reduce Receive Cost

In this section, we report the simulated performance when the uniform and ME strategies are used to sample block codes. In order to quantify the gain due to exploiting atypical channel behavior, we will compare the blocklengths required by our schemes with the lower bound imposed by Shannon's sphere packing bound. We use the numerical techniques in the paper by Dolinar, Divsalar and Pollara [19], which follow Shannon's original derivation [47]. Shannon's derivation permits *perfect* spherical codes with no constraint on the alphabet. Hence, the bounds are slightly optimistic for our binary input channel.

We begin with a simple (7,3,4) dual Hamming code. Figure 2-3 plots the receive costs of this and other r_{rx}= 1/2 codes.

The key observation is that sampling a trivial (7,3) code with $r_{\mathrm{rx}} = 1/2$ using the ME criterion achieves the same receive cost as the much stronger, rate 1/2, $K = 7$ convolutional code. Figure 2-4 plots the performance for a 1/10 rate. ME sampling

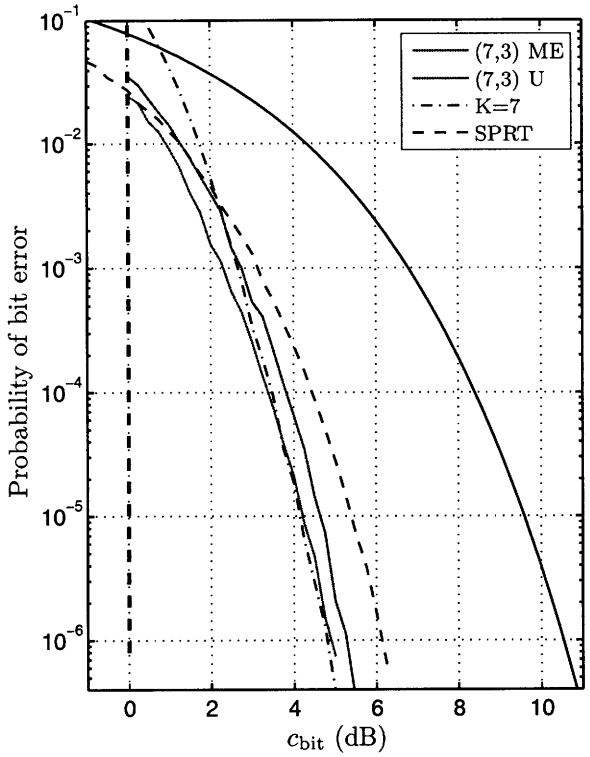

Figure 2-3: Performance of the $r_{\text{rx}} = 1/2$, (7,3) code using ME and uniform sampling (U).

has a gain of close to 7 dB, and outperforms uniform sampling by about 0.9 dB. A gain of 7 dB is about the limit of what is possible with convolutional codes using ML (Viterbi) decoding. For instance, increasing the constraint length of a rate 1/2 code from 7 to 9 improves the gain from 5.7 to 6.5 dB. A rate 1/4 constraint length $K = 9$ with octal generators [463 535 733 745] has a gain of roughly 7 dB. Further rate reductions are unlikely to buy much. Higher gains would require sequential decoding of large constraint length codes [24]. In summary, our receiver achieves, with practically zero baseband processing, the same *receive* cost as the strongest Viterbi decoded convolutional code.

For our next example, we consider the (24,12,8) Golay code, chosen for its remarkable gain at a small blocklength, and its efficient trellis representations. Figure 2-5 plots the receive cost performance of the Golay code at $r_{\text{rx}} = 1/2$. When decoded using the ME criterion, the Golay code achieves a gain of 7.7 dB. This is 0.3 dB from the cutoff limit, which is unprecedented, with or without feedback, for $k = 12$.

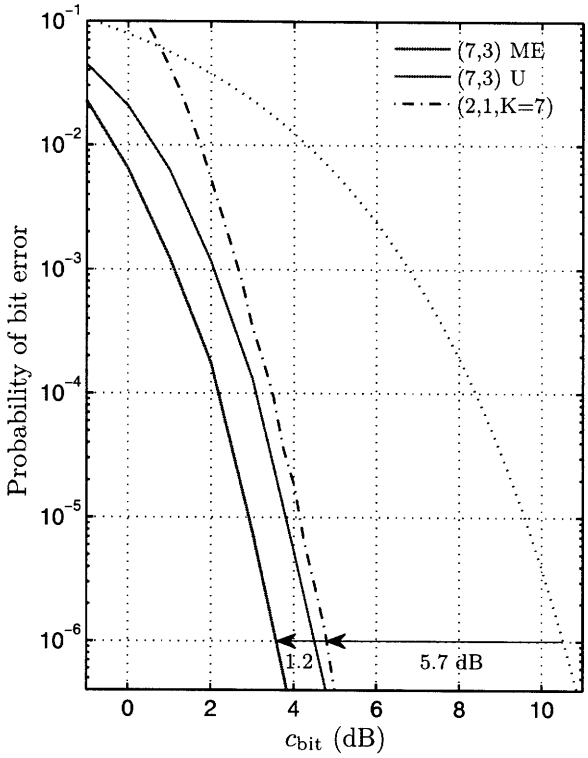

Figure 2-4: Performance of the $r_{\text{rx}} = 1/10$, (7,3) code using ME and uniform sampling.

(Note that a fixed-length Golay code has a gain of about 4.5 dB.) The ME criterion outperforms uniform sampling by 0.8 dB, which is significant in this regime. The sphere packing limit on block size at this E_b/N_0 and performance is $k \geq 56$. For comparison, a 3GPP2, rate $\approx 1/2$, (768,378) Turbo code achieves a gain of 8.5 dB at a frame error rate (FER) of 10^{-3} [26]. Crozier *et al.* have reported a (264,132) Turbo code with a gain of about 7.3 dB at a FER of 10^{-4}.

Other examples include:

- *Convolutional*: A sequentially decoded convolutional code with $K = 41$ with a gain of 7.5 dB [42]. Assuming that 5 constraint lengths are sufficient to realize most of this gain, this would be a block size of $k \approx 200$.

- *Concatenated*: A concatenated code with an outer RS(255,223) code, and an inner rate 1/2, $K = 7$ code. An interleaving depth of just 2 achieves a gain of roughly 7.5 dB. The effective rate is 0.43, and the block size is 3568 bits. (Forney's doctoral thesis is the original reference for concatenated codes [23].

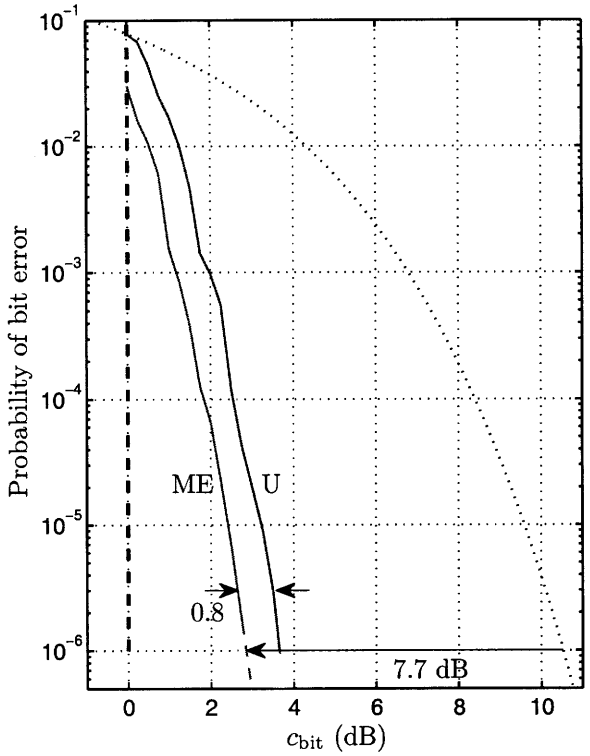

Figure 2-5: Performance of the Golay code at $r_{\text{rx}} = 1/2$.

The example used here is drawn from [19]).

- *LDPC*: Tong *et al.* report a (160,80) LDPC code with a gain of roughly 7.3 dB [51]. They also report a (160,80) random binary code with a gain of roughly 7.8 dB. The authors comment that traditional belief propagation incurs a large performance loss (2 dB), and use ML decoding instead, which is computationally prohibitive, limiting the practical utility of these codes.

- *Hybrid-ARQ*: Rowitch *et al.* use a high rate, (256,231) BCH, outer code and a Turbo code based on a (3,1,5) recursive systematic convolutional (RSC) code (a *rate compatible punctured Turbo (RCPT)* code). They are about 0.2 dB better than the cutoff limit for $r_{\text{rx}} = 1/2$ and $k = 231$ [44]. They also report that an ARQ scheme based on Hagenauer's rate compatible punctured convolutional (RCPC) codes, which uses a (3,1,7) mother code and $k = 256$, is right at the cutoff limit for a receive rate of $1/2$.

Clearly, ME sampling allows significant reductions in blocklength over conven-

tional codes, including those that use feedback (hybrid ARQ). If we are prepared to lower the receive rate, the advantage is even more pronounced. Figure 2-6 shows the performance when Golay codes are sampled with a receive rate of 1/6.

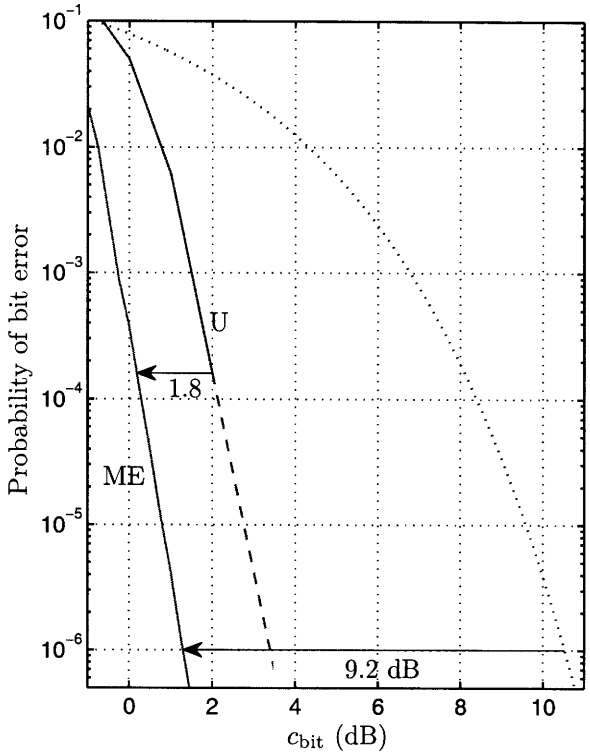

Figure 2-6: Sampling the Golay code with receive rate 1/6.

The gain using ME is 9.2 dB, 1.8 dB higher than uniform sampling, and 0.4 dB better than the rate 1/6 cutoff limit. The sphere packing lower bound for this rate and performance is $k \geq 81$. Hence, we are close to a seventh of the bound, as opposed to a fifth for rate 1/2. Increasing the expected number of observations expands the class of atypical channel behavior that sequential schemes exploit, resulting in higher gains.

For comparison, Xilinx Inc. reports a 3GPP2, rate 1/5 Turbo code with block-length $k = 506$ that achieves a gain of around 9 dB [30]. Gracie *et al.* calculate the gain of a rate 1/3, $k = 378$ code from the same family to be 9.1 dB at a FER of 10^{-3} [26]. RCPC and RCPT ARQ schemes are ill-equipped to exploit low rates and perform poorly at rate 1/6 [44]. Our scheme outperforms these by about 2 dB.

2.8 Practical Considerations

In this section we determine the energy, hardware, and bandwidth costs of our sampling strategies.

2.8.1 VLSI Costs

We would like the digital baseband implementation of our sampling strategies to consume a small fraction of the energy required to sample the channel. To see if this is the case, we compute the number of arithmetic operations required, and estimate the energy consumed to execute these in current semiconductor technology. This approach has the obvious shortcoming that arithmetic is not the only source of energy consumption. For instance, conveying data, especially over long interconnect, can incur substantial energy costs. However, such estimates are useful in eliminating clearly infeasible approaches, and in declaring others provisionally feasible. It helps that our proposed solutions do not require large memories, interleavers, or switches, as is the case for iterative decoders. This makes our designs more local, and helps reduce interconnect costs.

We assume that the front-end consumes 1 nJ of energy per channel sample [33, 17], and samples are quantized to 3 bits. For a receive rate 1/6 Golay system, we make 72 observations on average, which requires 7 additional bits on top of the 3 bit channel sample, for a total of 10 bits. Hence, we expect codeword log-likelihoods (LLs) to be no wider than this. Simulations by J. Kwong in a 65 nm semiconductor process suggest that 1 nJ allows about 40,000 additions of 10 bit operands (at 0.5 Volt). Measured silicon results by Mercier *et al.* for a modem in a 90 nm process confirm that these are the right order of magnitude [40]. Ideally, we would like the digital baseband energy dissipation to be limited to 0.1 nJ. This permits at most 4000 10b adds *per channel sample.* We now discuss the cost of several systems in increasing order of baseband complexity.

We begin with systems that can implement brute-force versions of uniform and ME sampling at negligible cost. Recall that uniform sampling terminates when the

difference between the two largest codeword LLs exceeds a threshold. Hence, for a $(n, k = \log_2(M))$ block code, the brute-force strategy requires M additions *per sample* to update the codeword LLs, and another $M - 1$ adds to find the difference between the winner and runner up, for a total of $2M - 1$. As an example, for the (7,3) Hamming code, this is 15 adds per sample, which is clearly insignificant. The (24,12) Golay code on the other hand would require 8191 adds, which exceeds our energy budget by 2x.

Consider next a brute-force implementation of ME sampling. Table 2.1 shows the computation required to compute bit *a posteriori* probability (APP).

Operations per sample	MUL	ADD
Multiply $\mathbf{\Pi}$ with $e^{\pm y_{i^*}}$	M	-
Normalize $\mathbf{\Pi}$	-	$M-1$
If $\max \mathbf{\Pi} > a$ terminate	-	$M-1$
Calculate $\pi_i = \sum_{j \in U_i(1)} \Pi_j$	-	$nM/2$
Pick $i^* = \arg\max_i \lvert\pi_i - 1/2\rvert$	-	$2n-1$

Table 2.1: Computation required per sample for a brute-force implementation of the ME algorithm. We use $\mathbf{\Pi}$ to denote the vector of the $M = 2^k$ codeword posteriors, and i^* for the index of the observed bit.

The table does not include the cost of implementing the exponentiation via an 8-entry lookup table. The (7,3) code sampled using brute-force ME would need about 8 multiplies and 110 adds per sample — again, insignificant compared with the cost of sampling.

Consider next a system that uniformly samples the Golay code. As we've seen, the brute-force approach requires 8191 adds per sample. An alternative is to consider trellis representations of codes (see [24, Chap. 10] for a thorough, tutorial introduction to the general topic.) We use Lafourcade and Vardy's scheme for Viterbi decoding of the Golay code [32], which has essentially the same complexity as Forney's original scheme [22]. The main modification we need is to produce both the winner and the runner up to determine if termination is warranted. We call this the "2-Viterbi" algorithm.

Lafourcade *et al.* consider the Golay trellis with three 8-bit sections with state

complexity profile $\{1, 64, 64, 1\}$ and branch complexity profile $\{128, 1024, 128\}$. Ignoring branch metric computation, a conventional Viterbi algorithm would require $2|E| - |V| + 1$ operations (additions and comparisons) to produce the ML path. This would be 2431 for the trellis above. This may be trimmed to 1087 by making the following observation. Roughly put, when a pair of states share two branches that are complements (i.e. $\boldsymbol{c}$ and $\overline{\boldsymbol{c}}$), we can halve the adds and the comparisons used in the ACS operation. Finally, the branch metrics may be computed via a slight refinement of a standard Gray code technique, and this requires 84 adds per section. Hence, the standard Viterbi algorithm may be executed using 1087+3x84 or 1339 adds. We have computed that the 2-Viterbi variation requires 2685+3x84 or 2937 adds.

A more careful analysis reveals that average case costs are lower. We run the 2-Viterbi after every channel observation. If the bit corresponding to this symbol lies, say, in the third trellis section, we do not have to perform computations for the first two sections. Since we sample bits uniformly, the average number of observations turns out to be about 1852, as opposed to 2937. Next, we always observe the k systematic bits of a block code. Hence, when running a $r_{\text{rx}} = 1/2$ uniform sampling scheme, we incur the 2-Viterbi cost for only half the samples. Hence, the cost per sample is 926 adds. This suggests that we must approach the full cost as $r_{\text{rx}} \to 0$. But another saving is possible at low rates. When running at say, $r_{\text{rx}} = 1/6$, we make $6k$ observations on average. Suppose we observe three channel samples at a time. This incurs a receive cost loss of only $6k/(6k + 3)$, which is less than 0.2 dB for $k = 12$, but cuts arithmetic costs to a third, i.e., 2937/3 or 979 adds per sample (not 1852/3, since partial updates are not possible when the sampled bits span more than one section). Note that 1000 adds is a quarter of our energy budget, and this suggests that uniformly sampling the Golay code is energy feasible.

The hardware cost of this scheme would depend on the throughput and latency constraints. A fully parallel implementation would require 2937 sets of 10 bit adders, which, assuming 5 logic gates per single bit adder is not trivial — about 146K gates for adders alone. However, such an implementation could run at an extremely high clock rate, essentially the inverse of the delay of a 30b adder. In current technology,

this can easily approach a GHz or more (depending on the voltage used). Since this is higher than what many systems require, it allows us to reduce the area proportionally. Also, note that observing sets of m bits at a time, reduces throughput requirements by the same factor.

Finally, consider a system that use ME sampling. One way to implement such sampling is to run the BCJR algorithm on the code's trellis [4]. The coordinate ordering for the Golay code that matches the Muder bound is well known, and yields a 24 section trellis with $|V| = 2686$ states, and $|E| = 3580$ branches [24]. The max-log-MAP approximation allows replacing the multiply-accumulate (MAC) operations in BCJR with add-compare-select (ACS) operations. Hence, the forward and backward passes have the same computational complexity as a Viterbi pass, which requires $2|E| - |V| + 1$, or 4475 operations. Since these passes are run after making a single channel observation, we need only partial updates, and the total cost for two passes is only 4475, rather than 2x4475, adds. Unfortunately, the cost of computing the log bit APPs after completing the forward-backward pass is rather high. For a section with e edges, we need e adds each for the branches labeled '0' and '1' respectively (for a binary trellis), and a further $e/2 - 1$ comparisons per group to find the maximum in each. This is followed by one final subtraction. Hence, we need about $3|E|$, or about 10,740 adds. This brings the total to 15,215 adds, which is 4x our energy budget.

This computation may be substantially reduced by running a max-log-MAP algorithm on the *tail-biting* trellis of a Golay code, which has $|V| = 192$ states and $|E| = 384$ edges [10]. While the trellis is an order-of-magnitude more compact, the MAP algorithm needs to be modified to avoid limit cycles or pseudocodewords [3]. Madhu and Shankar have proposed an efficient MAP decoder that avoids these problems by tracking sub-trellises of the tail-biting Golay trellis [36]. Under a plausible assumption about convergence, we have estimated that it takes about five trellis passes, and 2124 adds to compute all the bit APPs. Again, for a rate 1/2 code, this reduces to 1062. We can reduce these further by running the MAP algorithm after groups of observations, though the performance loss in this case is not as easily determined as for uniform sampling. From a hardware perspective, the simplest im-

plementation of this scheme would require about 12 kBits of storage and 6300 gates for the adders. Such an implementation could be run at a frequency limited only by the delay of 32 full adders, again allowing GHz clock rates.

2.8.2 Impact of Non-Zero Transmit Rates

Practical systems cannot afford zero transmit rates. We now determine the maximum transmit rate which preserves most of the gain of our sampling schemes.

The issue of limiting sequential tests to a maximum number of observations was considered by Wald, and his observation essentially answers the question above. Wald considered a sequential test to determine if the unknown mean of a Gaussian r.v. with known variance exceeded a specified threshold. It was shown that limiting the SPRT's observation to that of a fixed length test, resulted in double the error rate of the latter [54]. A doubling of error is insignificant in the Gaussian context. A mere fraction of a dB can compensate for this increase when error rates are low.

Proposition 2.1 (Transmit Rate Design Guide). *Sequential strategies over the AWGN channel require* $(1+\varepsilon)$ *more samples than a fixed-length strategy with identical performance.*

For a SPRT based receiver, this means the transmit rate is marginally lower than that of a repetition coded receiver. Consider next a coded system with uniform sampling. E.g., a (24,12), $r_{\text{rx}} = 1/6$ receiver. A fixed length Golay code provides 4.5 dB of gain. Our scheme provides about 7.4 dB of gain, or 3 dB (2x) more than fixed length Golay. Hence, we require a transmit rate slightly lower than $(1/2)(1/6)$. Simulations show that a rate of $1/14$ is indistinguishable from a zero transmit rate.

The idea carries over to ME sampling with the caveat that sampling is intermittent. Having sampled a bit, the receiver might have to wait for the next codeword to observe the next desired bit. If a ME scheme selected bit indices independently, we would observe 2 bits per codeword on average. Hence, the transmit rate is reduced by a factor of $2/n$ compared with uniform sampling schemes. The bit indices are not i.i.d. in practice, and simulations for the Golay code show that only 1.6 bits are observed on

average for $r_{rx} = 1/6$. This ME scheme is about 4.8 dB better than Golay (about 3x). Hence, the transmit rate required is slightly lower than (1.6/24)(1/3)(1/6) or about 1/270. A repetition coded system would have a rate of (1/9)(1/6) or 1/54. Hence, we expand bandwidth by 5x over a repetition coded system to deliver a receive cost that is 9x lower.

The discussion above assumes a lossless sampling strategy. We can trade off transmit rate for some degradation in receive cost. As an example, we can force the ME algorithm to make some minimum number of observations for each codeword (pick not just the bit with the highest entropy, but also the runner-up, etc.) The algorithm could also factor in the "finite" horizon to improve error rates. For instance, we can ensure that there are no "unspent" observations when the transmission is about to end, and so on.

2.9 Comparisons with Current Receivers

What are the best options available to a system designer faced with the receive cost problem? Are the proposed receivers more suitable than existing, fixed-length ones? The designer must consider several factors, among them, baseband energy and hardware costs, whether samplers can turn on/off rapidly enough to support ME schemes, permissible bandwidth sacrifice, receive rate constraints, and typical message sizes, etc. We have demonstrated that the new class of receivers appear feasible from an energy and silicon real estate perspective. We expect that both these measures will be substantially improved by tuning the algorithms. We now discuss the tradeoffs at three gain settings.

The new approach seems advantageous if a gain of 6-7 dB is required. Provided the sampler can turn on and off sufficiently fast, a (7,3) ME $r_{rx} = 1/10$ system provides 7 dB of gain at virtually zero hardware cost, and a bandwidth sacrifice of roughly 2x compared with a repetition coded system. If on-off time is an issue, a gain of 6 dB is possible by uniformly sampling a (7,3) code with $r_{rx} = 1/10$, with a transmit rate that is 1.6x *greater* than a repetition code.

The prescription is less clear cut in the 7-8 dB gain regime, where the choice is between uniformly sampling a Golay code, or using the strongest convolutional code that may be Viterbi decoded with reasonable effort. For instance, uniformly sampling Golay at $r_{\text{rx}} = 1/6$ has a gain of about 7.5 dB. As discussed earlier, a rate 1/4 constraint length $K = 9$ code has a gain of roughly 7 dB. Assuming 5 constraint lengths suffice to preserve most of this gain, we may use a terminated convolutional code with $k \approx 45$. Both solutions can meet the strict digital baseband energy dissipation limits. Both can achieve throughput in the GHz range, though the (2,1,9) code can run at roughly three times the Golay's speed (it has only one ACS in the critical path, while we run 2-Viterbi on the Golay, which increases delay). The Golay design is extremely compact. The trellis has only three sections, with a state complexity limited to 64. The only really compute intensive section is the middle one, and even there, the raw branch complexity of 1024 is reduced to 512 by exploiting complementary codewords. Its memory requirements are also substantially lower (about an order of magnitude) than the terminated convolutional code's. A control oriented application where very short commands are used (8-10 bits) may incur a high energy cost if forced to use larger payloads, and might prefer the Golay solution. An application where the sampler runs in the GHz range might have difficulty meeting latency constraints using 2-Viterbi on Golay (i.e. the decision to terminate sampling might come too late), and might be better served by the (2,1,9) solution. Note that the transmit rate required for the uniform sampling solution is about 2-2.5x *higher* than a repetition code, but half that of the (2,1,9) solution.

The final design point is the ME sampled Golay code at receive rate 1/6, with a gain of 9.2 dB. Benkeser *et al.* have recently reported a VLSI efficient Turbo decoder for 3G wireless that can realize similar gains [5, 6]. (The authors do not report the block size, but we estimate that $k \approx$ 400–500 should suffice.) Benkeser's solution achieves 10.8 Mbps with 6 iterations, using 44K gates and 122 kbits of memory. ME sampling implemented on the tail-biting Golay trellis would require 5 passes of the trellis. We postulate that running the ME on the compact tail-biting trellis would result in a significantly more local VLSI solution, and translate to lower energy and

hardware. The chief barrier to adoption of our proposed receiver would be the ability of the sampler to shut on/off quickly. Another factor to consider would be the 5x bandwidth sacrifice compared with a repetition coded solution.

2.10 Summary

We turn the information theoretic problem on its head and ask what coding techniques work best when cost is dominated by sampling the received signal. This formulation is of interest in short-range, low-data-rate communications devices that must be extremely energy efficient. We propose a sequential scheme that samples the bit with the maximum *a posteriori* entropy at every step, and show that it allows reduction of blocklengths by an order-of-magnitude over traditional coding schemes, including those with feedback.

Chapter 3

Synchronization by Sampling Energy

In this chapter our focus is receivers that sample signal energy rather than amplitude. It is well known that non-coherent systems, i.e., those agnostic to signal phase, simplify sampling. The phase noise, and frequency offset specifications of the carrier are relaxed. Hence, carrier generation requires simpler hardware and lower energy. In fact, Lee et al. report a pulsed ultra-wideband (UWB) implementation that mixes the signal with itself, eliminating the carrier entirely [33]. Energy sampling systems reduce sampling costs further by essentially decimating the non-coherent output. This allows the analog-to-digital converters to run at a lower rate, simplifying hardware and reducing electronics power.

While these sampling techniques reduce the electronics energy required per sample, say, $\mathcal{E}_{\text{samp}}$, they increase the number of samples required to maintain performance. In systems that use binary signaling, like pulsed-UWB, the loss of phase necessitates orthogonal signaling, with the accompanying SNR degradation. Filtering a signal devoid of its phase introduces an additional degradation that increases with the length of the filter. Hence, such schemes are only viable when the reduction in $\mathcal{E}_{\text{samp}}$ is greater than the increase in the samples required. UWB energy sampling receivers, as reported by Lee et al., and Daly et al. suggest that the tradeoff is beneficial [17].

The packet length in UWB energy sampling systems today is often dominated by

the preamble required to synchronize the receiver. This is partly because such systems are often used in telemetry-type applications with very short payloads [17]. Also, accurate synchronization may be required to allow triangulating receiver location. Hence, scenarios where the preamble is an order of magnitude longer than the payload are not implausible.

While the significant synchronization overhead in UWB energy sampling receivers is well understood [43, 49], no work we know of has addressed the fundamental question: what are the limits of synchronization over the energy sampling (ES) channel, and what sequences might achieve those limits? Current thinking, as reflected by the recent IEEE 802.15.4a standardization process, appears to be that sequences that work well over coherent channels (maximum length sequences and variants) must also be optimum for the ES channel [29].

We pose the problem of synchronization over the ES channel formally, and show how error exponents may be calculated when the SNR $\rightarrow 0$ or ∞. It becomes clear that, unlike the coherent case, there is no single optimum synchronization sequence for a given length. Rather, the optimum sequence is a function of SNR. We construct sequences for the low and high SNR regimes that significantly outperform existing ones.

After outlining our model in the next section, we will briefly study classical sequences (i.e. optimum when sampling amplitude). This will be followed by an analysis of two existing families for energy sampling. The first is the family of *periodic* sequences. We will derive the error exponent for this family, and comment on the currently used classifier. The second is the family of oversampled Hadamard (O-H) sequences. This will be followed by developing three new families, the first two of which have large Euclidean distances, and are suitable for low SNR operation. The first is the *interpolated* Hadamard (I-H) sequence family. The second is the family that maximizes Euclidean distance — maximum squared distance (MSD) sequences. The final new family is that of *walking* sequences, which significantly improve the root Euclidean distance, and are suitable for high SNR operation. This will be followed by simulations comparing the performance of these families.

We denote vectors in boldface, e.g., $\boldsymbol{x}$, $\boldsymbol{y}_i$, etc. The j^{th} element of a vector is denoted via $(\cdot)_j$, e.g., $(\boldsymbol{x})_j$ or $(\boldsymbol{y}_i)_j$ which we usually write as x_j or $y_{i,j}$. Random variables are sans-serif. Hence x is a deterministic scalar, and y_i a random vector.

3.1 Preliminaries

We introduce the energy sampling channel, synchronization as a coding problem, and distance properties of synchronization codebooks. Note that synchronization is a continuous time (CT), *estimation* problem. In what follows, we treat it as a discrete-time (DT), *classification* problem. We do this for several reasons. Although the DT version resolves delays only up to half a symbol period, this is sufficient for many pulsed UWB applications, where this corresponds to an error of at most 1 ns. This allows tagging a location to within a foot. Next, our approach holds, at least theoretically, even if sub-symbol resolution is desired. All that is necessary is changing the codebook. In fact, we have used a DT setup to synchronize using Gaussian pulses. The codebook is more sophisticated, but the framework holds. This is possible because the CT case may be transformed to a DT one by the appropriate change of basis. This is well understood to be the sinc function basis for the coherent case. The transformation is not exact for squared Gaussian processes, but Urkowitz has shown that a DT treatment is valid for unaliased signals [52].

3.1.1 Energy Sampling Channel, Integrated Codebooks

The real, discrete-time, energy sampling (ES) channel is described by (figure 3-1),

$$\mathsf{y}_k = \sum_{j=kP}^{kP+(P-1)} (x_j + \mathsf{w}_j)^2 \qquad k = 0, 1, 2, \ldots$$

where P is the period of integration, x_j are the transmitted *symbols*, y_k are the received *samples*, and w_j are i.i.d. Gaussian r.v.s distributed as $\mathcal{N}(0, \sigma^2 = 1)$.

We restrict the input to on-off signaling; i.e., $x_i = \alpha c_i$, where $c_i \in \{0, 1\}$, and α is the peak signal amplitude. We may refer to α^2 as the "peak SNR per symbol."

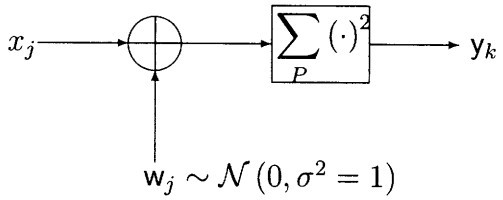

Figure 3-1: The discrete-time, energy sampling channel

In most of the chapter, we will assume that the signal is peak-constrained; in a final section we will investigate an average power constraint. Note that the most realistic formulation of the problem would consider both constraints *simultaneously*, but this work starts off with a separate analysis.

The received samples are chi-square r.v.s with P degrees of freedom[1],

$$y_k \sim \chi^2_P\left(s_k^2\right)$$

and centrality parameter,

$$s_k^2 = \sum_{j=kP}^{kP+(P-1)} x_j^2 \tag{3.1}$$

Thus, only the number of ones in a codeword $\boldsymbol{c}$ during the integration period, $t_k(\boldsymbol{c}) = \sum_{kP}^{kP+(P-1)} c_j$, affects the statistics of the channel output.

Consider a binary codebook $\mathcal{C}$, with M codewords, $\boldsymbol{c}_i$, $i \in [0, M-1]$, of length n. If n is a multiple of the integration period P, then each codeword, $\boldsymbol{c}$, yields an integrated counterpart, $\boldsymbol{t}(\boldsymbol{c})$, with $n_{\mathrm{r}} = n/P$ elements, $t_k(\boldsymbol{c})$, $k \in [0, n_{\mathrm{r}} - 1]$. We say that $\boldsymbol{c}$ has a symbol length of n, and a sample length of n_{r}. The *integrated* codebook $\mathcal{T}(\mathcal{C})$ has M codewords $\boldsymbol{t}_i \triangleq \boldsymbol{t}(\boldsymbol{c}_i)$ of length n_{r}. It follows from (3.1) that sample likelihoods when $\boldsymbol{c}_i$ is transmitted are given by,

$$y_k \sim \chi^2_P\left(t_{i,k}\ \mathrm{SNR}\right)$$

Hence, receiver statistics are completely determined by the integrated codebook $\mathcal{T}(\mathcal{C})$

[1]Appendix 3.9.1 reviews the notation and properties of chi-square r.v.s.

and the SNR.

3.1.2 Synchronization Codes

In our work, synchronization refers to the problem of inferring an unknown channel delay. Communications systems infer this delay by transmitting a known preamble to the random, message bearing payload. The preamble is partitioned into shorter sequences that are repeated. This is done for several reasons. First, this reduces hardware costs. For instance, as we shall see later, operation at a low SNR with $P = 16$ might require $n_{\mathrm{r}} \approx 7000$. This is an order larger than what is considered feasible, cost-wise, from a VLSI point of view. For instance, the 802.15.4a standard mandates $n_{\mathrm{r}} = 31$, with an optional mode with $n_{\mathrm{r}} = 127$. Thus, smaller sequences are repeated with a start-frame-delimiter (SFD) at the end for disambiguation. The resulting sequences are longer than the optimum, unconstrained sequence, but the approach keeps hardware costs manageable. Another reason for repetition is that synchronization is preceded by an automatic gain control (AGC) operation. The initial part of the sequence is lost as the system finds the optimum gain. Also, the loss is variable in length, depending on the SNR. Hence, the synchronizer starts in the "middle" of a sequence. Finally, repeating sequences is useful for estimating certain other channel impairments like frequency offsets, etc. As a result of sequence repetition, the synchronization problem is best modeled as inferring the unknown *cyclic* shift of a sequence, say $\boldsymbol{c}$, transmitted over a channel. Thus, synchronization is a coding problem where the codebook is the circulant matrix of $\boldsymbol{c}$, i.e., it consists of the n cyclic shifts of $\boldsymbol{c}$,

$$\mathcal{C}(\boldsymbol{c}) \triangleq \left\{\boldsymbol{c}_i = \boldsymbol{c}^{(i)},\, i \in [0, n-1]\right\}$$

where the $(\cdot)^{(r)}$ operator denotes a right shift by r places, i.e.,

$$\left(\boldsymbol{c}^{(r)}\right)_j = c_{(j-r) \bmod n}$$

It is often useful to express the index of a codeword $\boldsymbol{c}_i$ thus,

$$i \triangleq jP + \ell \qquad (0 \leq \ell < P)$$

We call $\ell = i \bmod P$, the *phase*, and $j = \lfloor i/P \rfloor$, the *sample* shift (as opposed to ordinary, or symbol shift). We define the *subsampled-sequence* or simply subsequence, $\boldsymbol{q}_k(\boldsymbol{c})$, as the sequence obtained by downsampling $\boldsymbol{c}$ by a factor P, starting at symbol c_k,

$$(\boldsymbol{q}_k(\boldsymbol{c}))_j = c_{jP+k} \qquad j \in [0, n_{\mathrm{r}} - 1],\ k \in [0, P-1] \tag{3.2}$$

Hence, there are P subsequences, not necessarily unique. We express a sequence in terms of its subsequences thus,

$$\boldsymbol{c} = \boldsymbol{q}_0(\boldsymbol{c}) \cdot \boldsymbol{q}_1(\boldsymbol{c}) \cdot \boldsymbol{q}_2(\boldsymbol{c}) \cdots \boldsymbol{q}_{P-1}(\boldsymbol{c}) \tag{3.3}$$

We sometimes abbreviate $\boldsymbol{q}_k(\boldsymbol{c})$ to $\boldsymbol{q}_k$. The reader may verify that (3.3) yields the following elementary relationships ($0 \leq \ell < P$),

$$\begin{aligned}
\boldsymbol{c}^{(1)} &= \boldsymbol{q}_{P-1}^{(1)} \cdot \boldsymbol{q}_0 \cdot \boldsymbol{q}_1 \cdots \boldsymbol{q}_{P-2} \\
\boldsymbol{c}^{(\ell)} &= \boldsymbol{q}_{P-\ell}^{(1)} \cdots \boldsymbol{q}_{P-1}^{(1)} \cdot \boldsymbol{q}_0 \cdots \boldsymbol{q}_{P-\ell-1} \\
\boldsymbol{c}^{(P)} &= \boldsymbol{q}_0^{(1)} \cdot \boldsymbol{q}_1^{(1)} \cdot \boldsymbol{q}_2^{(1)} \cdots \boldsymbol{q}_{P-1}^{(1)} \\
\boldsymbol{c}^{(jP)} &= \boldsymbol{q}_0^{(j)} \cdot \boldsymbol{q}_1^{(j)} \cdot \boldsymbol{q}_2^{(j)} \cdots \boldsymbol{q}_{P-1}^{(j)} \\
\boldsymbol{c}^{(jP+\ell)} &= \boldsymbol{q}_{P-\ell}^{(j+1)} \cdots \boldsymbol{q}_{P-1}^{(j+1)} \cdot \boldsymbol{q}_0^{(j)} \cdots \boldsymbol{q}_{P-\ell-1}^{(j)}
\end{aligned} \tag{3.4}$$

It follows from definitions that,

$$\boldsymbol{t}\,(\boldsymbol{c}) = \sum_{k=0}^{P-1} \boldsymbol{q}_k(\boldsymbol{c}) \tag{3.5}$$

Synchronization performance is determined by the SNR and the integrated codebook $\mathcal{T}(\mathcal{C}(\boldsymbol{c}))$, which we sometimes abbreviate to $\mathcal{T}(\boldsymbol{c})$, or $\mathcal{T}(\boldsymbol{c})$. Note that integrating

a codeword shifted by $r = jP$ is the same as shifting the integrated result by j, i.e.,

$$\boldsymbol{t}_{jP+\ell} = \boldsymbol{t}\left(\boldsymbol{c}^{(jP+\ell)}\right) = \boldsymbol{t}_{\ell}^{(j)} \tag{3.6}$$

Hence, $\mathcal{T}(\boldsymbol{c})$ is completely characterized by its first P codewords, and the other codewords are shifts of these.

We now illustrate these definitions via a toy sequence,

$$\boldsymbol{c} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \end{bmatrix}$$

of length 6. Then,

$$\mathcal{C}(\boldsymbol{c}) \triangleq \begin{bmatrix} \boldsymbol{c}_0 \\ \boldsymbol{c}_1 \\ \boldsymbol{c}_2 \\ \boldsymbol{c}_3 \\ \boldsymbol{c}_4 \\ \boldsymbol{c}_5 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}$$

Assume further that $P = 2$, which gives $n_{\mathrm{r}} = n/P = 3$. Then,

$$\mathcal{T}_2(\boldsymbol{c}) \triangleq \mathcal{T}_2(\mathcal{C}(\boldsymbol{c})) \triangleq \begin{bmatrix} \boldsymbol{t}_0 \\ \boldsymbol{t}_1 \\ \boldsymbol{t}_2 \\ \boldsymbol{t}_3 \\ \boldsymbol{t}_4 \\ \boldsymbol{t}_5 \end{bmatrix} = \begin{bmatrix} 2 & 0 & 0 \\ 1 & 1 & 0 \\ 0 & 2 & 0 \\ 0 & 1 & 1 \\ 0 & 0 & 2 \\ 1 & 0 & 1 \end{bmatrix}$$

Also, $\boldsymbol{q}_0(\boldsymbol{c}_1) = [0\,1\,0]$, and $\boldsymbol{q}_1(\boldsymbol{c}_1) = [1\,0\,0]$.

3.1.3 Elementary Distance Properties

We denote Hamming distances by d_{H}. We also define,

$$d_{\mathrm{sq}}(\boldsymbol{t}, \boldsymbol{t}') = \|\boldsymbol{t} - \boldsymbol{t}'\|^2$$
$$d_{\mathrm{rt}}(\boldsymbol{t}, \boldsymbol{t}') = \left\|\sqrt{\boldsymbol{t}} - \sqrt{\boldsymbol{t}'}\right\|^2 \triangleq \sum_j \left(\sqrt{t_j} - \sqrt{t'_j}\right)^2$$

We call these metrics the squared distance, and root distance (squared). As we shall see in the next section, the squared distance is significant at low SNRs, while the root distance is the appropriate function at high SNRs. The codebook minimum distance is defined as usual,

$$d(\mathcal{V}) = \min_{\boldsymbol{v} \neq \boldsymbol{v}' \in \mathcal{V}} d(\boldsymbol{v}, \boldsymbol{v}')$$

where $\mathcal{V}$ may be a binary or integrated codebook, and d is the relevant distance.

We will find the following definitions useful. When two codewords differ at a coordinate, the ordered pair of differing symbols is called a *difference pair*, and the set of all difference pairs is called a *difference set*. For instance, for the codewords,

$$\begin{bmatrix} \boldsymbol{t}_0 \\ \boldsymbol{t}_1 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 2 & 0 & 2 \\ 0 & 1 & 0 & 1 & 1 & 2 \end{bmatrix}$$

the difference set is $\{(1,0),(0,1),(2,1),(0,1)\}$. A difference pair is said to be of the *type* (x, y) if it is either (x, y) or (y, x).

We now derive some elementary bounds on these distances.

Lemma 3.1. *For any two adjacent integrated codewords, $\boldsymbol{t}, \boldsymbol{t}' \in \mathcal{T}(\boldsymbol{c})$,*

$$d_{\mathrm{rt}}(\boldsymbol{t}, \boldsymbol{t}') \leq d_{\mathrm{sq}}(\boldsymbol{t}, \boldsymbol{t}') = d_{\mathrm{H}}(\boldsymbol{t}, \boldsymbol{t}') = d_{\mathrm{H}}\left(\boldsymbol{q}, \boldsymbol{q}^{(1)}\right) \tag{3.7}$$

for some subsequence $\boldsymbol{q}$ of $\boldsymbol{c}$.

Proof of lemma 3.1. The result is a corollary of the property that all difference pairs of adjacent codewords are of the type $(k, k-1)$, $k \in \mathbb{N}$, and hence the Hamming

distance is equal to the squared distance. Also, $\left|\sqrt{x}-\sqrt{y}\right| \leq |x-y|$ which proves the root and squared distance inequality.

Recall (3.4),

$$\begin{aligned}\boldsymbol{c}^{(jP+\ell)} &= \boldsymbol{q}_{P-\ell}^{(j+1)} \cdots \boldsymbol{q}_{P-1}^{(j+1)} \cdot \boldsymbol{q}_{0}^{(j)} \cdots \boldsymbol{q}_{P-\ell-2}^{(j)} \boldsymbol{q}_{P-\ell-1}^{(j)} \\ \boldsymbol{c}^{(jP+\ell+1)} &= \boldsymbol{q}_{P-\ell-1}^{(j+1)} \cdot \boldsymbol{q}_{P-\ell}^{(j+1)} \cdots \boldsymbol{q}_{P-1}^{(j+1)} \cdot \boldsymbol{q}_{0}^{(j)} \cdots \boldsymbol{q}_{P-\ell-2}^{(j)}\end{aligned}$$

It follows from (3.5) that,

$$\begin{aligned}\boldsymbol{t}_{jP+\ell+1} - \boldsymbol{t}_{jP+\ell} &= \boldsymbol{q}_{P-\ell-1}^{(j+1)} - \boldsymbol{q}_{P-\ell-1}^{(j)} \\ &= \boldsymbol{s}^{(1)} - \boldsymbol{s}\end{aligned} \tag{3.8}$$

where

$$\boldsymbol{s} \triangleq \boldsymbol{q}_{P-\ell-1}^{(j)}$$

Hence, distances between adjacent integrated codewords depend only on one subsequence, which we call the *active* subsequence. We also call $d_{\mathrm{H}}\left(\boldsymbol{s}, \boldsymbol{s}^{(1)}\right)$ the *adjacent distance* of $\boldsymbol{s}$. Since $\boldsymbol{s}$ is binary, it follows from (3.8) that all difference pairs are of the type $(k, k-1)$. For a difference set of this form, clearly,

$$d_{\mathrm{sq}}\left(\boldsymbol{t}, \boldsymbol{t}'\right) = d_{\mathrm{H}}\left(\boldsymbol{t}, \boldsymbol{t}'\right) = d_{\mathrm{H}}\left(\boldsymbol{s}, \boldsymbol{s}^{(1)}\right) = d_{\mathrm{H}}\left(\boldsymbol{q}_{P-\ell-1}, \boldsymbol{q}_{P-\ell-1}^{(1)}\right)$$

where the last equality holds since shifting a sequence does not alter its adjacent distance. Also,

$$d_{\mathrm{rt}}\left(\boldsymbol{t}, \boldsymbol{t}'\right) \leq d_{\mathrm{sq}}\left(\boldsymbol{t}, \boldsymbol{t}'\right)$$

is always true, regardless of adjacency constraints. The lemma follows. □

Note also that all three distance measures of integrated codebooks are bounded thus,

$$d(\mathcal{T}(\boldsymbol{c})) \leq \min\left(n_{\mathrm{r}} - 1, \frac{2w(\boldsymbol{c})}{P}\right) \quad (n_{\mathrm{r}} \text{ odd}) \tag{3.9}$$

where $w(\boldsymbol{c})$ denotes the weight of sequence $\boldsymbol{c}$.

3.1.4 Performance Analysis

Consider the binary hypothesis testing problem $\Theta = \theta_{i=0,1}$, where Θ denotes the true state of nature, and the θ_i are the equiprobable hypotheses. The observed r.v. has likelihoods,

$$\mathsf{y} \sim p_j(y) \triangleq p_{\mathsf{y}|\theta_j}(y \mid \theta_j),\ j = 0 \text{ or } 1$$

Then, the maximum-likelihood (ML) classifier, $\hat{\Theta}$, that observes m i.i.d. r.v.s $\mathsf{y}_0, \mathsf{y}_1, \ldots, \mathsf{y}_{m-1}$ distributed as above, yields a probability of error that is bounded thus,

$$p_{\text{err}} \triangleq \Pr\left(\hat{\Theta}(\boldsymbol{y}) \neq \Theta\right) \leq e^{-m\rho_{\mathsf{y}}} \tag{3.10}$$

where ρ_{y} is the *Chernoff exponent* given by the Chernoff bound,

$$\rho_{\mathsf{y}} = -\ln \min_{\lambda<0} \mathrm{E}_0\, e^{\lambda \ell(y)}$$

where, $\ell(y)$ is the log-likelihood ratio,

$$\ell(y) = \ln \frac{p_0(y)}{p_1(y)}$$

and E_i denotes the expectation under θ_i. Equation (3.10) is useful for bounding system performance. Moreover, it is exponentially tight, i.e., the vanishing error rate is guaranteed to decrease exponentially with m with a rate *identical* to the exponent. Hence, an experiment that yields twice the exponent requires half as many samples to achieve identical asymptotic performance. Since our concern is minimizing sampling, the exponent is of fundamental importance.

Consider next the classification problem when one of two codewords, $\boldsymbol{c}$ or $\boldsymbol{c}'$, of length n may be transmitted over the ES channel. The ML classifier observes the received vector $\mathbf{y}$ of length $n_{\mathrm{r}} = n/P$. We denote the exponent for this test by $\rho(\boldsymbol{c}, \boldsymbol{c}'; P, \mathrm{SNR}) \triangleq \rho_{\mathbf{y}}$. Then, the *codebook exponent* is defined thus,

$$\rho(\mathcal{C}; P, \mathrm{SNR}) = \min_{\boldsymbol{c} \neq \boldsymbol{c}' \in \mathcal{C}} \rho(\boldsymbol{c}, \boldsymbol{c}'; P, \mathrm{SNR})$$

We can now bound the probability of codeword error when a ML scheme infers which of the M codewords of $\mathcal{C}$ was transmitted,

$$p_{\text{err}} \leq (M-1)e^{-\rho(\mathcal{C};P,\,\text{SNR})}$$

The problem of finding the codebook exponent for general SNR appears to be intractable because of the chi-square distribution. However, we have derived an exponentially accurate approximation.

Theorem 3.1. *The exponent of discriminating codewords, $\boldsymbol{c}$ and $\boldsymbol{c}'$ transmitted over the ES channel is*[2]*,*

$$\rho(\boldsymbol{c},\boldsymbol{c}';P,\text{SNR}) \doteq d_{\text{sq}}\left(\boldsymbol{t}(\boldsymbol{c}),\,\boldsymbol{t}(\boldsymbol{c}')\right)\frac{\text{SNR}^2}{16P} \qquad (\text{SNR}\to 0) \tag{3.11}$$

and,

$$\rho(\boldsymbol{c},\boldsymbol{c}';P,\text{SNR}) \doteq d_{\text{rt}}\left(\boldsymbol{t}(\boldsymbol{c}),\,\boldsymbol{t}(\boldsymbol{c}')\right)\frac{\text{SNR}}{8} \qquad (\text{SNR}\to \infty) \tag{3.12}$$

The proof is included in appendix 3.9.1.

Corollary 3.1. *The codebook exponent is given by,*

$$\rho(\mathcal{C};P,\text{SNR}) \doteq d_{\text{sq}}\left(\mathcal{T}(\boldsymbol{c})\right)\frac{\text{SNR}^2}{16P} \qquad (\text{SNR}\to 0)$$

and,

$$\rho(\mathcal{C};P,\text{SNR}) \doteq d_{\text{rt}}\left(\mathcal{T}(\boldsymbol{c})\right)\frac{\text{SNR}}{8} \qquad (\text{SNR}\to \infty)$$

Note that the exponent in the limit of infinite SNR is independent of the integration period. However, we will see that the exponent in both regimes varies as n_{r}/P because of combinatorial constraints. In light of corollary 3.1, we will seek sequences

[2] $f(x) = o(g(x))$ $(x \to a)$ implies $\lim_{x\to a} f(x)/g(x) = 0$, while $f(x) \doteq g(x)$ $(x \to a)$ implies $\lim_{x\to a} f(x)/g(x) = 1$.

that maximize the squared or root distance for a given n and P, rather than trying directly to optimize the codeword exponent as a function of SNR.

3.2 Prior Work

For the case $P = 1$, the integrated codebook $\mathcal{T}$ is identical to the binary codebook $\mathcal{C}$. Hence, the synchronization problem is to maximize the minimum distance of a binary circulant matrix. This is a well studied problem, and we summarize the key results. For a sequence of length n,

$$d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{c})\right) \leq \left\lfloor \frac{n+1}{2} \right\rfloor \tag{3.13}$$

The result follows from elementary properties of difference sets [28]. Furthermore, odd lengths that achieve (3.13) are necessarily of the form $n \equiv -1 \mod 4$. An odd length sequence that meets the bound (3.13) corresponds to a *cyclic Hadamard difference set*, and is termed a Hadamard sequence. Such a sequence is equidistant from all its cyclic shifts, and has weight $(n \pm 1)/2$ depending on whether a majority of ones or zeros is chosen. The existence of such sets for arbitrary n remains open, though computer searches support a conjecture due to Golomb.

Conjecture 3.1 (Golomb [25]). *Hadamard sequences exist for lengths,* $n \equiv -1 \mod 4$, *if and only if,*

$$n = \begin{cases} 2^k - 1 & k \in \mathbb{N}, \text{ or} \\ p & p \text{ a prime, or} \\ p(p+2) & p, p+2 \text{ are both primes} \end{cases}$$

There is a constructive proof of the *if* part. For instance, sequences corresponding to $n = 2^k - 1$ are the well known *maximum length* or m-sequences, and the construction relies on primitive polynomials in GF(2). Construction for the other cases relies on properties of quadratic residues. As of September 2006, the conjectured *only if* part

had been verified for all but seven open cases for $n < 10000$ [25].

The focus of our work is to generalize the notion of good sequences for $P > 1$. While prior work is replete with receivers and synchronization techniques over the ES channel, it generally assumes that Hadamard sequences (and their ternary extensions) are optimum. The two approaches popular today use periodic pulse trains, or upsampled Hadamard sequences. We briefly discuss these now.

3.2.1 Periodic Sequences

A popular synchronization scheme for the ES channel uses alternating bursts of 1s and 0s followed by an aperiodic suffix,

$$\boldsymbol{c} = [\underbrace{\boldsymbol{u}\,\boldsymbol{u}\,\ldots\,\boldsymbol{u}}_{M\cdot P \text{ repetitions}}\ \ \boldsymbol{v}]$$

where,

$$\boldsymbol{u} = [\underbrace{1\,1\,1\ldots1}_{P \text{ symbols each}}\ \underbrace{0\,0\,0\ldots0}_{P \text{ symbols each}}] \tag{3.14}$$

and $\boldsymbol{v}$ is the suffix required to ensure non-zero integrated distance. The receiver processes the periodic pattern independently of the suffix. Hence, it can only resolve the cyclic shift modulo the period ($2P$). Once this shift is inferred, the receiver realigns its sampling phase with the sequence $\boldsymbol{u}$ and begins a search for the suffix. The suffix thus resembles a SFD.

The receiver processes the periodic pattern by partitioning it into P sections, each with M repetitions of $\boldsymbol{u}$. Each section is sampled with a different initial sampling phase, i.e., samples for section m are obtained thus,

$$\mathsf{y}_k = \boldsymbol{t}\left(\left((\boldsymbol{x}+\mathbf{w})^2\right)^{(m)}\right) = \sum_{j=P.k-m}^{P.k+(P-1)-m} (x_j + \mathsf{w}_j)^2 \qquad k = 0, 1, 2, \ldots$$

(Non-causality is managed by inserting delays and extra samples.) Every section produces two metrics by adding the odd and even samples, respectively. The receiver picks the highest of the $2P$ metrics to infer the shift (Test A). It may be seen that the

metrics are chi-square r.v.s with MP degrees of freedom, and a centrality parameter of the form $s^2 = jM\text{SNR}$, where $j \in [0, P]$. An error occurs if the largest sample does not correspond to the r.v. with the largest centrality parameter ($s^2 = MP\text{SNR}$). Prior work has relied on numerical techniques to compute the performance of this test [43]. We now bound the error exponents for vanishing and infinite SNR.

Consider the binary hypothesis test where two independent chi-square r.v.s $\chi^2_{MP}(M(P-1)\text{SNR})$ and $\chi^2_{MP}(MP\text{SNR})$ are presented in one of the two possible orders with equal probability. A ML classifier is used to infer the order (Test B). It may be shown that the probability of error of this test is strictly smaller than that of Test A[3]. The exponent of test B is readily calculated via theorem 3.1,

$$\rho_{\text{y}} \doteq 2\frac{(MP\text{SNR} - M(P-1)\text{SNR})^2}{16MP} = \frac{2M}{16P}\text{SNR}^2 \triangleq \frac{d_{\text{sq}}}{16P}\text{SNR}^2 \qquad (\text{SNR} \to 0) \tag{3.15}$$

in the low SNR regime, where the factor of 2 reflects that we have access to two observations with identical information. Hence,

$$d_{\text{sq}} = 2M = \frac{n_{\text{r}}}{P}$$

since the number of samples in the periodic portion is $2MP$. In the high SNR regime, the exponent for test B is,

$$\rho_{\text{y}} \doteq 2\frac{(\sqrt{MP\text{SNR}} - \sqrt{M(P-1)\text{SNR}})^2}{8} \triangleq \frac{d_{\text{rt}}}{8}\text{SNR} \qquad (\text{SNR} \to \infty)$$

which gives,

$$d_{\text{rt}} = 2\left(\sqrt{P} - \sqrt{P-1}\right)^2 M \leq \frac{2M}{2.9P} = \frac{n_{\text{r}}}{2.9P^2} \qquad (P > 1) \tag{3.16}$$

We will see that this is a factor of about $6P$ worse than sequences we will introduce

[3]Consider the alternate test A.1 where only these two r.v.s are compared and an error is declared if the maximum does not correspond to the larger centrality parameter. Clearly, A.1's probability of error is lower than that of test A. Also, Test B is no worse than than A.1 since it uses a ML scheme to tag the observations, rather than just picking the maximum.

later.

Note that test A has access to $2P$ r.v.s, but by using the max criterion, it essentially behaves as if it looks at only 2 (test B). Hence, its exponent is degraded by a factor proportional to P. Using a better test which incorporates phase information can rectify this, but a key problem remains. This synchronization strategy does not tell us how to pick the aperiodic suffix $\boldsymbol{v}$. This would be okay if $\boldsymbol{v}$ were short compared to the sequence, and ad-hoc choices would work, but neither is true.

One variation on periodic sequences is a paper by Park et al., who modulate the amplitude of the pulse train to improve performance [31]. Aside from the practical difficulty of doing so, it is hard to gauge precisely what gains such modulation offers.

3.2.2 Oversampled Sequences

The recent IEEE 802.15.4a standard uses sequences with the structure [29],

$$\tilde{\boldsymbol{c}} = \boldsymbol{s} \cdot \mathbf{0} \cdot \mathbf{0} \cdots \mathbf{0}$$

where $\mathbf{0}$ is the all-zero subsequence. In other words, $\tilde{\boldsymbol{c}}$ is obtained by oversampling $\boldsymbol{s}$ by a factor of P. We denote the sample length of $\tilde{\boldsymbol{c}}$ by $\tilde{n}_{\mathrm{r}}$. Clearly, $\mathcal{T}(\tilde{\boldsymbol{c}})$ has zero distance due to its all-zero subsequences. We will see how this is managed shortly. From (3.4), we have,

$$\tilde{\boldsymbol{c}}_{jP+\ell} = \tilde{\boldsymbol{c}}^{(jP+\ell)} = \underbrace{\mathbf{0} \cdot \mathbf{0} \cdots \mathbf{0}}_{\ell} \cdot \boldsymbol{s}^{(j)} \cdot \underbrace{\mathbf{0} \cdot \mathbf{0} \cdots \mathbf{0}}_{P-\ell-1}$$

and thus,

$$\tilde{\boldsymbol{t}}_{jP+\ell} = \boldsymbol{s}^{(j)} \tag{3.17}$$

Hence, all integrated codewords with the same sample shift are identical. 15.4a compliant receivers discriminate between shifts of $\tilde{\boldsymbol{c}}$ by sweeping over all sampling

phases. In other words, the transmitted sequence is,

$$\boldsymbol{c} = [\underbrace{\tilde{\boldsymbol{c}} \quad \tilde{\boldsymbol{c}} \quad \ldots \quad \tilde{\boldsymbol{c}}}_{P \text{ repetitions}}]$$

and the corresponding integrated codeword is,

$$\begin{aligned} \boldsymbol{t} &= \left[\boldsymbol{t}\left(\tilde{\boldsymbol{c}}\right) \quad \boldsymbol{t}\left(\tilde{\boldsymbol{c}}^{(1)}\right) \quad \boldsymbol{t}\left(\tilde{\boldsymbol{c}}^{(2)}\right) \quad \ldots \quad \boldsymbol{t}\left(\tilde{\boldsymbol{c}}^{(P-1)}\right)\right] \\ &= \left[\tilde{\boldsymbol{t}}_0 \quad \tilde{\boldsymbol{t}}_1 \quad \tilde{\boldsymbol{t}}_2 \quad \ldots \quad \tilde{\boldsymbol{t}}_{P-1}\right] \end{aligned}$$

which is different from the usual, fixed-sampling-phase codeword, $\boldsymbol{t}(\boldsymbol{c})$. Note that while $\boldsymbol{t}$ above may also be generated by shifting the phase in the transmitter, the resulting sequence $\boldsymbol{c}$ would not be periodic. Also, $n_{\mathrm{r}} = P\tilde{n}_{\mathrm{r}}$ samples. Similarly, the integrated codeword corresponding to codeword $\boldsymbol{c}_i$ is,

$$\boldsymbol{t}_i = \left[\tilde{\boldsymbol{t}}_i \quad \tilde{\boldsymbol{t}}_{i+1} \quad \tilde{\boldsymbol{t}}_{i+2} \quad \ldots \quad \tilde{\boldsymbol{t}}_{i+P-1}\right], \quad i \in [0, n_{\mathrm{r}} - 1]$$

The integrated codebook is a Toeplitz-like matrix (with constant entries along the positive, rather than negative slope diagonals),

$$\mathcal{T} = \begin{bmatrix} \boldsymbol{t}_0 \\ \boldsymbol{t}_1 \\ \boldsymbol{t}_2 \\ \vdots \\ \boldsymbol{t}_i \\ \vdots \\ \boldsymbol{t}_{n_{\mathrm{r}}-1} \end{bmatrix} = \begin{bmatrix} \tilde{\boldsymbol{t}}_0 & \tilde{\boldsymbol{t}}_1 & \ldots & \tilde{\boldsymbol{t}}_{P-1} \\ \tilde{\boldsymbol{t}}_1 & \tilde{\boldsymbol{t}}_2 & \ldots & \tilde{\boldsymbol{t}}_P \\ \tilde{\boldsymbol{t}}_2 & \tilde{\boldsymbol{t}}_3 & \ldots & \tilde{\boldsymbol{t}}_{P+1} \\ & & \vdots & \\ \tilde{\boldsymbol{t}}_i & \tilde{\boldsymbol{t}}_{i+1} & \ldots & \tilde{\boldsymbol{t}}_{i+P-1} \\ & & \vdots & \\ \tilde{\boldsymbol{t}}_{\tilde{n}_{\mathrm{r}}-1} & \tilde{\boldsymbol{t}}_0 & \ldots & \tilde{\boldsymbol{t}}_{P-2} \end{bmatrix}$$

Note that $\tilde{\boldsymbol{t}}_i = \tilde{\boldsymbol{t}}_{i \bmod \tilde{n}_\mathrm{r}}$ since $\tilde{\boldsymbol{t}}$ has length $\tilde{n}_\mathrm{r}$. This explains the indices of the last row. We may now use (3.17) to further clarify the structure of our codebook,

$$\mathcal{T} = \begin{bmatrix} \boldsymbol{t}_0 \\ \boldsymbol{t}_1 \\ \boldsymbol{t}_2 \\ \vdots \\ \boldsymbol{t}_P \\ \boldsymbol{t}_{P+1} \\ \vdots \\ \boldsymbol{t}_{n_\mathrm{r}-2} \\ \boldsymbol{t}_{n_\mathrm{r}-1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{s} & \boldsymbol{s} & \dots & \boldsymbol{s} & \boldsymbol{s} \\ \boldsymbol{s} & \boldsymbol{s} & \dots & \boldsymbol{s} & \boldsymbol{s}^{(1)} \\ \boldsymbol{s} & \boldsymbol{s} & \dots & \boldsymbol{s}^{(1)} & \boldsymbol{s}^{(1)} \\ \vdots & & & & \\ \boldsymbol{s}^{(1)} & \boldsymbol{s}^{(1)} & \dots & \boldsymbol{s}^{(1)} & \boldsymbol{s}^{(1)} \\ \boldsymbol{s}^{(1)} & \boldsymbol{s}^{(1)} & \dots & \boldsymbol{s}^{(1)} & \boldsymbol{s}^{(2)} \\ \vdots & & & & \\ \boldsymbol{s}^{(\tilde{n}_\mathrm{r}-1)} & \boldsymbol{s}^{(\tilde{n}_\mathrm{r}-1)} & \dots & \boldsymbol{s} & \boldsymbol{s} \\ \boldsymbol{s}^{(\tilde{n}_\mathrm{r}-1)} & \boldsymbol{s} & \dots & \boldsymbol{s} & \boldsymbol{s} \end{bmatrix}$$

Note that the integrated codebook is binary valued. Hence, both the squared and root distance are equal to the Hamming distance. If $\boldsymbol{s}$ is a Hadamard sequence then,

$$d_\mathrm{H}\left(\boldsymbol{s}^{(r)}, \boldsymbol{s}^{(r' \neq r)}\right) = \frac{\tilde{n}_\mathrm{r}+1}{2}$$

and inspection of $\mathcal{T}$ shows that,

$$d_\mathrm{sq} = d_\mathrm{rt} = d_\mathrm{H} = \frac{\tilde{n}_\mathrm{r}+1}{2} = \frac{n_\mathrm{r}+P}{2P}$$

Hence, if we were to use asymptotic measures as a guide, oversampled Hadamard, or O-H sequences, appear preferable to periodic sequences in the low SNR regime. They have a marginal squared distance advantage, while doing away with the suffix. They have a significant root distance advantage — at least a factor of $3P/2$ — compared with periodic sequences (3.16), which again indicates their superiority in the high SNR regime. For this reason, we use the performance of O-H sequences as our baseline. In the next two sections, we will introduce new sequences suitable for operation at low SNRs.

3.3 Interpolated Sequences

We call a sequence *interpolated* if it can be shifted to the canonical form,

$$\boldsymbol{c} = \boldsymbol{s} \cdot \boldsymbol{s} \cdots \boldsymbol{s} \tag{3.18}$$

where all its subsequences, $\boldsymbol{q}_k(\boldsymbol{c})$, are equal to a *generator* sequence $\boldsymbol{s}$. In other words, $\boldsymbol{c}$ is obtained by repeating every bit in $\boldsymbol{s}$, P times (hence the nomenclature). A natural question is whether good generators produce good sequences. We now summarize our findings.

Definition. Given a binary sequence, say $\boldsymbol{u}$, its *shift-and-sum* sequence is,

$$\boldsymbol{u}_+ \triangleq \boldsymbol{u} + \boldsymbol{u}^{(1)}$$

Note that $\boldsymbol{u}_+$ is, in general, ternary.

Definition. We say that the generator, $\boldsymbol{s}$, of an interpolated sequence, $\boldsymbol{c}$, *preserves distance* if

$$d_{\mathrm{H}}\left(\mathcal{T}(\boldsymbol{c})\right) = d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{s})\right)$$

Lemma 3.2. *For an interpolated sequence $\boldsymbol{c}$ with generator $\boldsymbol{s}$,*

$$d_{\mathrm{H}}\left(\mathcal{T}(\boldsymbol{c})\right) \leq d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{s})\right)$$

Theorem 3.2. *A generator always preserves distance when P is odd.*

Theorem 3.3. *A generator preserves distance for even P if and only if it has the same distance as its shift-and-sum sequence, i.e.,*

$$d_{\mathrm{H}}\left(\mathcal{T}(\boldsymbol{c})\right) = d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{s})\right) \Longleftrightarrow d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{s}_+)\right) = d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{s})\right) \qquad (P \text{ even})$$

Theorem 3.4. *m-sequences preserve distance for all P.*

Conjecture 3.2. *Hadamard sequences preserve distance for all P.*

We now prove these results. For the interpolated sequence $\boldsymbol{c}$ in (3.18),

$$\boldsymbol{t}_0 = \sum \boldsymbol{q}_k(\boldsymbol{c}) = P\boldsymbol{s}$$

and (3.6) gives,

$$\boldsymbol{t}_{rP} = \boldsymbol{t}_0^{(r)} = P\boldsymbol{s}^{(r)}$$

Hence,

$$d_{\mathrm{H}}\left(\mathcal{T}(\boldsymbol{c})\right) \leq \min_{r,r'} d_{\mathrm{H}}\left(\boldsymbol{t}_{rP}, \boldsymbol{t}_{r'P}\right) = \min_{r,r'} d_{\mathrm{H}}\left(\boldsymbol{s}^{(r)}, \boldsymbol{s}^{(r')}\right) = d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{s})\right)$$

and lemma 3.2 follows. Also,

$$d_{\mathrm{H}}\left(\boldsymbol{t}_{rP}, \boldsymbol{t}_{r'P}\right) = d_{\mathrm{H}}\left(\boldsymbol{s}^{(r)}, \boldsymbol{s}^{(r')}\right) \geq d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{s})\right)$$

Thus, codewords with zero phase achieve a distance $\geq d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{s})\right)$. As we now show, this is true even if only one of the codewords has zero phase. Recall (3.4),

$$\boldsymbol{c}^{(rP+\ell)} = \boldsymbol{q}_{P-\ell}^{(r+1)} \cdots \boldsymbol{q}_{P-1}^{(r+1)} \cdot \boldsymbol{q}_0^{(r)} \cdots \boldsymbol{q}_{P-\ell-1}^{(r)}$$

where the phase $\ell < P$, as usual. Hence, a codeword, say $\boldsymbol{u}$, of an interpolated sequence has the form,

$$\boldsymbol{u} = \underbrace{\boldsymbol{s}^{(r+1)} \cdot \boldsymbol{s}^{(r+1)} \cdots \boldsymbol{s}^{(r+1)}}_{\ell \text{ subsequences}} \cdot \underbrace{\boldsymbol{s}^{(r)} \cdot \boldsymbol{s}^{(r)} \cdots \boldsymbol{s}^{(r)}}_{P-\ell \text{ subsequences}}$$

for some r. Assume that $\boldsymbol{u}$ has non-zero phase, i.e., $\ell \neq 0$. The integrated codeword $\boldsymbol{t}$ corresponding to $\boldsymbol{u}$ is,

$$\boldsymbol{t} = \ell\, \boldsymbol{s}^{(r+1)} + (P - \ell)\boldsymbol{s}^{(r)}$$

Then,

$$\boldsymbol{t}_{kP} - \boldsymbol{t} = P\boldsymbol{s}^{(k)} - \left(\ell\, \boldsymbol{s}^{(r+1)} + (P-\ell)\boldsymbol{s}^{(r)}\right)$$

At each coordinate, say j, where $\boldsymbol{s}^{(r)}$ and $\boldsymbol{s}^{(r+1)}$ are unequal, i.e. one of them is zero,

$$\left|\ell\, s_j^{(r+1)} + (P-\ell)s_j^{(r)}\right| < P$$

since $0 < \ell < P$. Hence, $\boldsymbol{t}_{kP} - \boldsymbol{t}$ must be non-zero at j. But there are at least $d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{s})\right)$ such coordinates, and hence,

$$d_{\mathrm{H}}\left(\boldsymbol{t}_{kP}, \boldsymbol{t}\right) \geq d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{s})\right)$$

when $\boldsymbol{t}$ has non-zero phase. We now consider the final case where both phases are non-zero. Define,

$$\boldsymbol{u} \neq \boldsymbol{u}' = \underbrace{\boldsymbol{s}^{(r'+1)} \cdot \boldsymbol{s}^{(r'+1)} \cdots \boldsymbol{s}^{(r'+1)}}_{\ell' \text{ subsequences}} \cdot \underbrace{\boldsymbol{s}^{(r')} \cdot \boldsymbol{s}^{(r')} \cdots \boldsymbol{s}^{(r')}}_{P-\ell' \text{ subsequences}}$$

with $\ell' \neq 0$ and its corresponding integrated codeword $\boldsymbol{t}'$. Then,

$$\boldsymbol{t} - \boldsymbol{t}' = \ell\, \boldsymbol{s}^{(r+1)} + (P-\ell)\boldsymbol{s}^{(r)} - \left(\ell' \boldsymbol{s}^{(r'+1)} + (P-\ell')\boldsymbol{s}^{(r')}\right) \tag{3.19}$$

Assume for now that all the sample shifts $r, r+1, r'$, and $r'+1$ are distinct, and rewrite the expression above as,

$$\boldsymbol{t} - \boldsymbol{t}' = \ell\, \boldsymbol{w} + (P-\ell)\boldsymbol{x} - (\ell' \boldsymbol{y} + (P-\ell')\boldsymbol{z})$$

Consider the coordinates where $\boldsymbol{w}$ and $\boldsymbol{y}$ differ (at least $d_{\mathrm{H}}\left(\mathcal{C}(\boldsymbol{s})\right)$). We say that a coordinate j is *lost between* $\boldsymbol{w}$ and $\boldsymbol{y}$ if $w_j \neq y_j$ but $t_j = t'_j$. We now show that coordinates can only be lost if $P = \ell + \ell'$. Consider the case when $w_j = 1$ and $y_j = 0$. If x_j is also 1, we cannot lose the coordinate since $P - \ell' < P$. If x_j is zero, then $\ell = P - \ell'$. A similar argument applies with $w_j = 0$ and $y_j = 1$. We can similarly prove that coordinates can be lost between $\boldsymbol{x}$ and $\boldsymbol{y}$ only if $\ell = \ell'$. Hence, if the

Hamming distance $d_{\mathrm{H}}(\boldsymbol{t}, \boldsymbol{t}')$ is smaller than both $d_{\mathrm{H}}(\boldsymbol{w}, \boldsymbol{y})$ and $d_{\mathrm{H}}(\boldsymbol{x}, \boldsymbol{y})$, the two conditions must hold simultaneously, i.e., $P = 2\ell = 2\ell'$, which is impossible since P is odd. The cases where sample shifts overlap may be proved similarly, and theorem 3.2 follows.

We now consider even P. An interpolated sequence may have lower distance than its generator only if $P = 2\ell = 2\ell'$. (This also holds when sample shifts overlap.) In this case (3.19) may be simplified to,

$$\begin{aligned} \boldsymbol{t} - \boldsymbol{t}' &= \frac{P}{2}\left(\boldsymbol{s}^{(r+1)} + \boldsymbol{s}^{(r)} - \left(\boldsymbol{s}^{(r'+1)} + \boldsymbol{s}^{(r')}\right)\right) \\ &= \frac{P}{2}\left(\boldsymbol{s}_{+}^{(r)} - \boldsymbol{s}_{+}^{(r')}\right) \end{aligned} \tag{3.20}$$

Note that r and r' must be distinct when $\ell = \ell'$ else $\boldsymbol{t}$ and $\boldsymbol{t}'$ refer to the same codeword. It follows that,

$$d_{\mathrm{H}}(\boldsymbol{t}, \boldsymbol{t}') = d_{\mathrm{H}}\left(\boldsymbol{s}_{+}^{(r)}, \boldsymbol{s}_{+}^{(r' \neq r)}\right)$$

Since r and r' may be chosen arbitrarily, theorem 3.3 follows.

We now prove that m-sequence generators preserve distance. First, note that, for $x_i \in \{0, 1\}$,

$$|x_0 \pm x_1 \pm x_2 \pm x_3| \geq x_0 \oplus x_1 \oplus x_2 \oplus x_3$$

where $\oplus$ is the binary exclusive-or operation. This inequality holds since the RHS is 1 only if an odd number of x_is are 1, in which case the LHS can never be zero for any assignment of signs. From (3.20) we have,

$$\begin{aligned} \left|t_j - t_j'\right| &\geq \left|s_j^{(r+1)} + s_j^{(r)} - \left(s_j^{(r'+1)} + s_j^{(r')}\right)\right| \\ &\geq s_j^{(r)} \oplus s_j^{(r+1)} \oplus s_j^{(r')} \oplus s_j^{(r'+1)} \end{aligned}$$

Hence,

$$d_{\mathrm{H}}(\boldsymbol{t}, \boldsymbol{t}') \geq w\left(\boldsymbol{s}^{(r)} \oplus \boldsymbol{s}^{(r+1)} \oplus \boldsymbol{s}^{(r')} \oplus \boldsymbol{s}^{(r'+1)}\right) = w\left(\boldsymbol{s}^{(g)}\right)$$

for some shift g, which follows from the linearity of m-sequences. The weight of an m-sequence with more 1s than 0s is equal to its Hamming distance (both $(n_r+1)/2$), and the result follows.

Hadamard sequences are not linear in general. However, we have verified that their sum-and-shift sequences have the same distance as the sequences themselves for sample lengths up to 1023. This led us to conjecture (3.2). Note that several (non-trivially) distinct Hadamard sequences may exist for certain lengths. We have verified the conjecture only for one sequence at each length.

We now illustrate these ideas with an example. The length 15 sequence,

$$\boldsymbol{s} = [1\,0\,1\,0\,1\,0\,0\,1\,0\,0\,1\,0\,0\,1\,1]$$

achieves $d_H(\mathcal{C}(\boldsymbol{s})) = 6$, the best possible for a non-Hadamard sequence of this length. The interpolated sequence for $P = 3$,

$$\boldsymbol{c} = [\underline{1\,1\,1}\ \underline{0\,0\,0}\ \underline{1\,1\,1}\ \underline{0\,0\,0}\ \underline{1\,1\,1}\ \underline{0\,0\,0}\ \underline{0\,0\,0}\ \underline{1\,1\,1}\ \underline{0\,0\,0}\ \underline{0\,0\,0}\ \underline{1\,1\,1}\ \underline{0\,0\,0}\ \underline{0\,0\,0}\ \underline{1\,1\,1}\ \underline{1\,1\,1}]$$

achieves a Hamming distance of 6, as theorem 3.2 guarantees for odd P. To see if $\boldsymbol{s}$ is distance preserving for even P, we construct its shift-and-sum sequence.

$$\boldsymbol{s}_+ = \boldsymbol{s} + \boldsymbol{s}^{(1)} = [2\,1\,1\,1\,1\,1\,0\,1\,1\,0\,1\,1\,0\,1\,2]$$

The pair,

$$\begin{bmatrix} \boldsymbol{s}_+ \\ \boldsymbol{s}_+^{(3)} \end{bmatrix} = \begin{bmatrix} 2\,1\,1\,1\,1\,1\,0\,1\,1\,0\,1\,1\,0\,1\,2 \\ 0\,1\,2\,2\,1\,1\,1\,1\,1\,0\,1\,1\,0\,1\,1 \end{bmatrix}$$

proves that $d_H(\mathcal{C}(\boldsymbol{s}_+)) \leq 5$. Hence, $\boldsymbol{s}$ does not preserve Hamming distance for even P. To see this, construct the interpolated sequence for $P = 4$. Then, the pair,

$$\begin{bmatrix} \boldsymbol{t}_2 \\ \boldsymbol{t}_{14} \end{bmatrix} = \begin{bmatrix} 4\,2\,2\,2\,2\,2\,0\,2\,2\,0\,2\,2\,0\,2\,4 \\ 0\,2\,4\,4\,2\,2\,2\,2\,2\,0\,2\,2\,0\,2\,2 \end{bmatrix}$$

has a Hamming distance of 5.

We call a sequence with a Hadamard sequence generator, an *interpolated Hadamard sequence*, or *I-H sequence*. If conjecture 3.2 is true then,

$$d_{\mathrm{H}}\left(\mathcal{T}(\boldsymbol{c})\right) = d_{\mathrm{sq}}\left(\mathcal{T}(\boldsymbol{c})\right) = \frac{n_{\mathrm{r}}+1}{2}$$

for a length n_{r} Hadamard generator. To see why the squared distance is equal to the Hamming distance, note that the squared distance of a codebook cannot be smaller than its Hamming distance. Also, the adjacent squared distance of a codebook is equal to the adjacent distance of its subsequences (3.7). The adjacent distance of a Hadamard sequence is the same as its Hamming distance, and the property follows. For comparison, a O-H sequence has,

$$d_{\mathrm{H}} = d_{\mathrm{sq}} = \frac{n_{\mathrm{r}}+P}{2P}$$

Hence, I-H sequences offer a squared distance gain approaching P for $n_{\mathrm{r}} \gg P$. Also, since squared distance cannot exceed $n_{\mathrm{r}}-1$ for odd n_{r}, I-H sequences are within 3 dB of the best possible squared distance sequences.

Lemma 3.3. *Define the* minimum different sample *between codewords* $\boldsymbol{t}, \boldsymbol{t}'$ *thus,*

$$t_{\min}(\boldsymbol{t}, \boldsymbol{t}') \triangleq \min_{j \,|\, t_j \neq t'_j} \max(t_j, t'_j)$$

If, for adjacent codewords $\boldsymbol{t}, \boldsymbol{t}' \in \mathcal{T}(\boldsymbol{c})$,

$$t_{\min}(\boldsymbol{t}, \boldsymbol{t}') > 1$$

then,

$$d_{\mathrm{rt}}\left(\boldsymbol{t}, \boldsymbol{t}'\right) \leq \frac{d_{\mathrm{H}}\left(\boldsymbol{t}, \boldsymbol{t}'\right)}{2.9\, t_{\min}(\boldsymbol{t}, \boldsymbol{t}')} \tag{3.21}$$

Proof of lemma 3.3. The function $f(x) = \left(\sqrt{x} - \sqrt{x-1}\right)^2$, $x \geq 1$, decreases monotonically in x. Also, it may be shown that,

$$\frac{1}{4x} \;\leq\; f(x) \;\leq\; \frac{1}{2.9x}, \quad x \geq 2 \tag{3.22}$$

When $\boldsymbol{t}, \boldsymbol{t}' \in \mathcal{T}(\boldsymbol{c})$ are adjacent, all difference pairs are of the type $(x, x-1)$. Hence,

$$\begin{aligned} d_{\mathrm{rt}}(\boldsymbol{t}, \boldsymbol{t}') &= \sum_j \left(\sqrt{t_j} - \sqrt{t'_j}\right)^2 \\ &\leq \sum_{t_j \neq t'_j} \frac{1}{2.9 \max(t_j, t'_j)} \qquad \textit{if } \min_j \max(t_j, t'_j) > 1 \\ &\leq \frac{d_{\mathrm{H}}(\boldsymbol{t}, \boldsymbol{t}')}{2.9 \left(\min_j \max(t_j, t'_j \neq t_j)\right)} \end{aligned}$$

□

Theorem 3.5. *If $\boldsymbol{c}$ is an interpolated Hadamard sequence, then,*

$$d_{\mathrm{rt}}(\mathcal{T}(\boldsymbol{c})) < \frac{n_{\mathrm{r}} + 1}{2.9P} \tag{3.23}$$

Proof. If $\boldsymbol{c}$ is an interpolated sequence with generator $\boldsymbol{s}$, then,

$$\boldsymbol{c}^{(\ell)} = \underbrace{\boldsymbol{s}^{(1)} \cdot \boldsymbol{s}^{(1)} \cdots \boldsymbol{s}^{(1)}}_{\ell \text{ subsequences}} \cdot \underbrace{\boldsymbol{s} \cdot \boldsymbol{s} \cdots \boldsymbol{s}}_{P-\ell \text{ subsequences}} \qquad (\ell < P)$$

and hence,

$$\boldsymbol{t}_\ell = \ell\, \boldsymbol{s}^{(1)} + (P - \ell)\boldsymbol{s}$$

If P is even, set $\ell = P/2$. Then,

$$\boldsymbol{t}_{P/2} = \frac{P}{2}\left(\boldsymbol{s} + \boldsymbol{s}^{(1)}\right) \tag{3.24}$$

$$\boldsymbol{t}_{P/2+1} = \boldsymbol{t}_{P/2} + \left(\boldsymbol{s}^{(1)} - \boldsymbol{s}\right) \tag{3.25}$$

Note that (3.24) implies that if $\boldsymbol{s}$ and $\boldsymbol{s}^{(1)}$ differ at a coordinate, then $\boldsymbol{t}_{P/2} = P/2$ at that coordinate. Hence, it follows from (3.25) that all difference pairs between $\boldsymbol{t}_{P/2}$ and $\boldsymbol{t}_{P/2+1}$ are of the type,

$$\left(\frac{P}{2}, \frac{P}{2} \pm 1\right)$$

The result follows from lemma 3.3. The case for odd P may be similarly proved. □

A corollary is that the root distance of a O-H sequence is at least 1.5x greater than that of an I-H sequence of the same length (the bound (3.23) is fairly tight for small P). The advantage grows close to 2x for large P since $\lim_{x\to\infty} f(x) = (4x)^{-1}$, and it can be shown that,

$$d_{\mathrm{rt}}\left(\mathcal{T}(\boldsymbol{c})\right) \approx \frac{n_{\mathrm{r}}+1}{4P} \qquad (P \geq 5)$$

3.4 Maximum Squared Distance Sequences

In this section, we will construct sequences with good squared distance. In discussing such sequences, we will find the following characterization of sequence complexity useful. The minimum number of distinct subsequences of a sequence, with the minimum being taken over all shifts of the sequence, is called its *dimension.* We exclude all-zero subsequences from the count. Thus, both O-H and I-H sequences are one-dimensional.

A sequence is a MSD if it maximizes the squared distance for given n_{r} and P. Hence, any sequence that achieves a squared distance of $n_{\mathrm{r}} - 1$ (n_{r} is assumed odd unless specified otherwise) is a MSD (3.9). We call such sequences *strong* MSDs, or just strong.

3.4.1 Simple, Strong Sequences

Recall that the Hamming and squared distance between adjacent integrated codewords is equal to the adjacent distance of the active subsequence (3.7). Hence, all subsequences of a strong sequence have an adjacent distance of $n_{\mathrm{r}} - 1$. This is only possible if they have alternating ones and zeros with exactly one repeated bit, i.e., they belong to the set,

$$\boldsymbol{S}_{\mathrm{alt}} = \left\{ \boldsymbol{s}_{\mathrm{alt}}^{(r)},\ \overline{\boldsymbol{s}}_{\mathrm{alt}}^{(r)} \right\}, \quad r \in [0, n_{\mathrm{r}} - 1] \tag{3.26}$$

where,

$$\boldsymbol{s}_{\mathrm{alt}} = [1\,0\,1\,0\,\ldots 1\,0\,0]$$

is a sequence of length n_r, and $\overline{s}$ denotes binary inversion. Since $\boldsymbol{S}_{\text{alt}}$ has $2n_r$ elements, a brute-force approach must examine $(2n_r)^P$ sequences to establish if a strong sequence exists. This is polynomial in n_r, as opposed to the set of all possible sequences, which is exponential[4].

Lemma 3.4. *An interpolated sequence cannot be strong.*

Proof. Let $\boldsymbol{c}$ be an interpolated sequence with sample length n_r, and generator, $\boldsymbol{s}$. Consider the following codewords,

$$\begin{bmatrix} \boldsymbol{c}_0 \\ \boldsymbol{c}_1 \\ \boldsymbol{c}_{n_r-1} \end{bmatrix} = \begin{bmatrix} \boldsymbol{s} \cdot \boldsymbol{s} \cdots \boldsymbol{s} \cdot \boldsymbol{s} \\ \boldsymbol{s}^{(1)} \cdot \boldsymbol{s} \cdots \boldsymbol{s} \cdot \boldsymbol{s} \\ \boldsymbol{s} \cdot \boldsymbol{s} \cdots \boldsymbol{s} \cdot \boldsymbol{s}^{(n_r-1)} \end{bmatrix}$$

Then,

$$\boldsymbol{t}_1 - \boldsymbol{t}_{n_r-1} = \boldsymbol{s}^{(1)} - \boldsymbol{s}^{(n_r-1)}$$

Since $\boldsymbol{s}$ is binary, it follows that,

$$d_{\text{sq}}\left(\boldsymbol{t}_1, \boldsymbol{t}_{n_r-1}\right) = d_{\text{H}}\left(\boldsymbol{s}^{(1)}, \boldsymbol{s}^{(n_r-1)}\right) = d_{\text{H}}\left(\boldsymbol{s}, \boldsymbol{s}^{(2)}\right)$$

since the distance depends only on the relative shift. If $\boldsymbol{c}$ is strong, $\boldsymbol{s}$ must be drawn from $\boldsymbol{S}_{\text{alt}}$. It may be shown that all sequences in $\boldsymbol{S}_{\text{alt}}$ have an alternate adjacent distance of 2. The lemma follows. □

Hence, strong sequences are necessarily two dimensional. We now consider the simplest 2–D sequences.

Definition. We call a 2–D sequence *simple*, if a suitable shift may be expressed in the form,

$$\boldsymbol{c} = \boldsymbol{s} \cdot \boldsymbol{s} \cdots \boldsymbol{s} \cdot \tilde{\boldsymbol{s}}$$

Searching for a strong sequence that is simple requires testing only $4n_r^2$ candidates. We now present our findings about the existence and construction of such sequences.

[4]The complexity of calculating the distance of a sequence is polynomial in n_r, P. Hence, we focus on the number of sequences to be tested.

Theorem 3.6. *A strong, simple sequence can exist for n_r, P only if,*

$$P \geq \left\lceil \sqrt{2(n_r - 7)} \right\rceil, \quad n_r \geq 11$$

The complete proof is in appendix 3.9.2. Since sequences in $\boldsymbol{S}_{\text{alt}}$ have very poor alternate adjacent distance, there are codeword pairs in $\mathcal{T}(\boldsymbol{c})$ with very poor Hamming distance. Such pairs rely on a few difference pairs of the type $(0, P/2)$ to achieve $d_{sq} = n_r - 1$. Hence, P must be on the order of $\sqrt{n_r}$.

Conjecture 3.3. *For all odd $n_r \geq 7$, and $P \geq P_{\text{lim}}(n_r)$, there exists a simple, strong sequence of the form,*

$$\boldsymbol{c} = \boldsymbol{s}_{alt} \cdot \boldsymbol{s}_{alt} \cdots \boldsymbol{s}_{alt} \cdot \boldsymbol{s}_{alt}^{(r(n_r))} \tag{3.27}$$

where P_{lim} and r are functions of n_r.

Note that r is independent of P. Hence, the same structure may be used for all $P \geq P_{\text{lim}}$. Theorem 3.6 implies that $P_{\text{lim}}(n_r) \geq \left\lceil \sqrt{2(n_r - 7)} \right\rceil$. We have verified this conjecture for all $n_r \equiv -1 \mod 4$, $n_r \leq 275$ and $P \leq 16$. For n_r of this form, we found that,

$$r(n_r) = 2 \left\lfloor \frac{n_r + 1}{8} \right\rfloor \tag{3.28}$$

works without exception. (Several values of r work, so $r(n_r)$ may be one of several functions.)

Lemma 3.5. *A suitable shift of the sequence $\boldsymbol{c}$ in* (5.1) *can always be expressed as a periodic sequence,*

$$\boldsymbol{c}^{(g)} = [\underbrace{\boldsymbol{u}\,\boldsymbol{u}\,\ldots\,\boldsymbol{u}}_{M \text{ repetitions}}\ \boldsymbol{v}]$$

with $\boldsymbol{u}$ as defined in (3.14),

$$\boldsymbol{u} = [\underbrace{1\,1\,1\ldots1}_{P \text{ symbols each}}\ \underbrace{0\,0\,0\ldots0}]$$

and,

$$M = \frac{n_r - 1}{2} - \left\lceil \frac{r(n_r)}{2} \right\rceil$$

The result follows from the definition of $\boldsymbol{s}_{\text{alt}}$ and is omitted. Hence, good sequences do have periodic prefixes, at least when P is large enough — on the order of $\sqrt{n_{\text{r}}}$ — and we are operating in the low SNR regime (where squared distance is meaningful). Current systems that use periodic sequences have poor performance because of their decoding scheme, and ad-hoc selection of $\boldsymbol{v}$. As noted earlier, using the maximum energy criterion to discriminate between possible phases completely disregards the structure of the integrated codebook. This leads to a performance loss proportional to P. The current practice of picking $\boldsymbol{v}$ to be a Barker or other pseudo-random sequence can also significantly diminish the squared distance.

As an example, we now construct a simple, strong sequence for $n_{\text{r}} = 11$. For this length, theorem 3.6 limits $P \geq 3$. Computer searches prove that $P_{\text{lim}}(11) = 5$. For $P = 5$, the sequence,

$$\boldsymbol{c} = \boldsymbol{s}_{\text{alt}} \cdot \boldsymbol{s}_{\text{alt}} \cdot \boldsymbol{s}_{\text{alt}} \cdot \boldsymbol{s}_{\text{alt}} \boldsymbol{s}_{\text{alt}}^{(2)}$$

is strong. We can shift $\boldsymbol{c}$ to demonstrate its periodic prefix,

$$\boldsymbol{c}^{(-10)} = [\boldsymbol{u}\ \boldsymbol{u}\ \boldsymbol{u}\ \boldsymbol{u}\ \boldsymbol{v}]$$

where,

$$\boldsymbol{u} = [1\,1\,1\,1\,1\,0\,0\,0\,0\,0]$$

and,

$$\boldsymbol{v} = [\underline{0\,0\,0\,0\,1}\ \underline{1\,1\,1\,1\,0}\ \underline{0\,0\,0\,0\,0}]$$

While $\boldsymbol{c}$ has $d_{\text{sq}} = 10$, its Hamming distance is only 4. This should be compared with an I-H sequence of the same length, for which both distances are 6.

3.4.2 Non-Simple MSDs

When no simple sequences are strong, we may test some, or all the $(2n_{\text{r}})^P$ candidate sequences to establish if *any* strong sequence exists. For instance, no simple sequence is strong for $n_{\text{r}} = 15, P = 5$. However, a computer search produced the following, 75

bit long, strong, 3–D sequence,

$$\boldsymbol{c} = \boldsymbol{s}_{\text{alt}} \cdot \boldsymbol{s}_{\text{alt}} \cdot \overline{\boldsymbol{s}}_{\text{alt}}^{(9)} \cdot \overline{\boldsymbol{s}}_{\text{alt}}^{(13)} \cdot \overline{\boldsymbol{s}}_{\text{alt}}^{(13)}$$

This sequence can also be expressed as a periodic sequence, with a SFD that is close to half the length. As another example, there is no simple, strong sequence for $n_{\text{r}} = 19$ and $P = 6$. However, the following 114 bit long, 2–D sequence is strong,

$$\boldsymbol{c} = \boldsymbol{s}_{\text{alt}} \cdot \boldsymbol{s}_{\text{alt}} \cdot \boldsymbol{s}_{\text{alt}} \cdot \overline{\boldsymbol{s}}_{\text{alt}}^{(12)} \cdot \overline{\boldsymbol{s}}_{\text{alt}}^{(12)} \cdot \overline{\boldsymbol{s}}_{\text{alt}}^{(12)}$$

Expressed as a periodic sequence, the SFD is again about half the length.

Search complexity increases drastically if the MSD for a n_{r}, P tuple is not strong. We can generalize the set $\boldsymbol{S}_{\text{alt}}$ in (3.26) thus,

$$\boldsymbol{S}_d = \left\{ \boldsymbol{s} \mid d_{\text{H}}\left(\boldsymbol{s}, \boldsymbol{s}^{(1)}\right) = d \right\}$$

i.e. the set of all sequences with adjacent distance d. Note that $\boldsymbol{S}_d$ is empty for odd values of d. Else,

$$|\boldsymbol{S}_d| = 2\binom{n_{\text{r}}}{d}$$

since a sequence in $\boldsymbol{S}_d$ must have bit transitions at exactly d out of n_{r} positions. The doubling is because inversion does not affect membership. If our squared distance target is d_{sq}, then all subsequences must belong to the set,

$$\boldsymbol{S}_{\geq d_{\text{sq}}} = \bigcup_{d_{\text{sq}} \leq d \leq n_{\text{r}}-1} \boldsymbol{S}_d$$

and the number of candidate sequences to test is,

$$\left|\boldsymbol{S}_{\geq d_{\text{sq}}}\right|^P$$

It is desirable to set our distance target as a fixed fraction of n_{r}. For instance, I-H sequences achieve a squared distance of roughly $n_{\text{r}}/2$. But, it may be shown that any

fixed fraction (excluding trivial ones) leads to an exponential number of sequences (in n_r) in $\boldsymbol{S}_{\geq d_{sq}}$. Hence, for large n_r, such a search is as infeasible as an exhaustive one. The problem of constructing a sequence with squared distance that is guaranteed to exceed a specified fraction of n_r remains unsolved.

Table 3.1 summarizes the results of our computer searches over a range of parameters. It is interesting to note that we could always find sequences such that $d_{sq}(n_r) \geq \lfloor 3n_r/4 \rfloor$. The longest sequence in the table is the 1008 bit long simple, strong sequence for $n_r = 63, P = 16$. The searches were not exhaustive. So it is possible that these distances will be improved, and simpler sequences found. The sequences corresponding to entries in the table are described in appendix A.

$P \backslash n_r$	11	15	19	23	27	31	63
2	8	12	14	18	20	24	
3	8	12	16_3	18	22	24_3	
4	8	12	16	20	24_3	26	$32^\dagger$
5	**10**	$\mathbf{14}_3$	16	20	24	28	
6	**10**	**14**	$\mathbf{18}_2$	$\mathbf{22}_3$	24	28	
7	**10**	**14**	**18**	$\mathbf{22}_2$	$\mathbf{26}_2$	$\mathbf{30}_3$	
8	**10**	**14**	**18**	**22**	**26**	$\mathbf{30}_2$	60
16	**10**	**14**	**18**	**22**	**26**	**30**	**62**

Table 3.1: Best known squared-distances for given n_r, P. Sequences in bold are strong. Subscripts indicate sequence dimension. All sequences without a subscript are simple. (†No new sequences are listed for n_r=63 and $P < 8$. The distance achieved by I-H sequences is quoted instead.)

3.4.3 Distance Comparison

Recall that O-H sequences yield,

$$d_{sq} = \frac{n_r + P}{2P} \approx \frac{n_r}{2P} \quad (n_r \gg P)$$

and, I-H sequences have,

$$d_{sq} = \frac{n_r + 1}{2} \approx \frac{n_r}{2} \quad (n_r \gg 1)$$

Hence, for a given P, strong MSDs achieve a squared distance gain of $2P$ and 2 over O-H and I-H sequences respectively (for sufficiently large n_{r}). Note that MSDs are not designed with root distance in mind. We estimate that strong sequences have roughly double the root distance of I-H sequences, or equivalently, about the same root distance as O-H sequences.

3.5 Walking Sequences

Our focus in this section is constructing sequences with good root distances. Recall that difference pairs for adjacent integrated codewords are of the type $(k, k-1)$, and that,

$$\left(\sqrt{k}-\sqrt{k-1}\right)^2 \leq \frac{1}{2.9k}, \quad k \geq 2$$

Hence, one (0,1) pair contributes about six times as much as a (1,2) pair, and, nine times as much as a (2,3) pair, etc. This motivates exploring sequences where most adjacent difference pairs are of type (0,1).

Definition. We define *root integer* distance, thus,

$$d_{\mathrm{Z}}\left(t, t'\right) = \left\lfloor \left(\sqrt{t}-\sqrt{t'}\right)^2 \right\rfloor$$

and by extension,

$$d_{\mathrm{Z}}\left(\boldsymbol{t}, \boldsymbol{t}'\right) = \sum_j d_{\mathrm{Z}}\left(t_j, t'_j\right)$$

and,

$$d_{\mathrm{Z}}\left(\mathcal{T}(\boldsymbol{c})\right) \triangleq \min_{\boldsymbol{t},\boldsymbol{t}' \in \mathcal{T}(\boldsymbol{c})} d_{\mathrm{Z}}\left(\boldsymbol{t}, \boldsymbol{t}'\right)$$

Note that $d_{\mathrm{rt}}\left(\mathcal{T}(\boldsymbol{c})\right) \geq d_{\mathrm{Z}}\left(\mathcal{T}(\boldsymbol{c})\right)$. We will henceforth abbreviate *root integer* to *integer* distance.

Corollary 3.2. *The integer distance between adjacent integrated codewords is the number of type (0,1) difference pairs.*

Theorem 3.7. *If a sequence* $\boldsymbol{c}$ *has,*

$$d_{\mathrm{Z}}\left(\mathcal{T}(\boldsymbol{c})\right) \geq 2t, \quad t \in \mathbb{N}$$

then its sample length is bounded thus,

$$n_{\mathrm{r}} \geq t(P+1) \tag{3.29}$$

Proof. The proof relies solely on studying adjacent distances. Since the Hamming distance between every pair of adjacent codewords is at least $2t$, every subsequence must have a weight of at least t, and hence the sequence weight must be no less than Pt. We now prove that we need as many coordinates to accommodate this weight. Recall that the Hamming distance between adjacent codewords is determined solely by the adjacent distance of the active subsequence. Consider the difference pairs between the active subsequence and its shift. A 1 bit in the subsequence might show up twice, once, or not at all in the difference set, depending on whether it has zero, one, or two neighbors that are 1s. We call the first kind *strong* and the second *weak*. As an example, if,

$$\boldsymbol{s} = [1_a\ 1_b\ 1_c\ 0\ 1_d\ 0]$$

(the tags allow distinguishing 1s), then, when $\boldsymbol{s}$ is active,

$$\begin{aligned}
\boldsymbol{t}_i - \boldsymbol{t}_{i+1} &= \boldsymbol{t}_i + \boldsymbol{s}^{(1)} - \boldsymbol{s} \\
&= \boldsymbol{t}_i + [(0 - 1_a)\ (1_a - 1_b)\ (1_b - 1_c)\ (1_c - 0)\ (0 - 1_d)\ (1_d - 0)] \\
&= \boldsymbol{t}_i + [-1_a\ 0\ 0\ 1_c\ -1_d\ 1_d]
\end{aligned}$$

Bits a and c are weak, and bit d is strong. Bit b does not participate in any difference pairs. Consider the case when the weight is exactly Pt. Then, all 1s must be strong since every subsequence must realize a distance of $2t$ using t 1s. However, no two subsequences can have a strong 1 at the same coordinate and still generate difference pairs of type (0,1). This is because the integrated symbol at that coordinate would

be 2, and the only possible difference pair would be (2,1). Hence, we need as many coordinates as there are 1s, i.e., Pt. A weight greater than Pt allows weak 1s, which may share coordinates with weak 1s from other subsequences. However, it may be shown that a weak 1 can share its coordinate with at most one other weak 1. But we need at least two weak 1s to make up for every strong 1 we eliminate. Hence, sharing does not allow reduction in the total coordinates required. We thus have Pt coordinates where exactly one subsequence is a 1. Hence, every integrated codeword has at least Pt 1s. Some codeword must also have at least t zeros to realize $2t$ difference pairs of type (0,1). The result follows. □

In the rest of the section, we will explore sequences that approach the bound (3.29). Note that such sequences need not achieve the maximum root distance for a given length. However, we will show that they achieve substantially better root distance than all previously discussed sequences. We will limit ourselves to sequences that yield *only* (0,1) difference pairs.

Definition. A sequence, $\boldsymbol{c}$, is *P-sparse* if, (i) Its integrated codebook $\mathcal{T}(\boldsymbol{c})$ has only 0 or 1 entries, and, (ii) It is minimum weight, i.e., it realizes a distance of $2t$ using the minimum possible weight Pt.

Corollary 3.3. *A sequence is* P-sparse *only if there are at least* $P-1$ *zeros between any two 1s.*

Note that the binary constraint implies that all distance measures (Hamming, squared, root, and integer) are identical for sparse sequences. Also, all distances are even, since the integrated codewords have constant weight.

A sparse sequence of sample length n_{r} may be specified by a *location* tuple, $\boldsymbol{\ell} \in \{0, 1, \ldots, P\}^{n_{\mathrm{r}}}$ as follows. Divide the sequence into n_{r} successive groups of P bits each. Then ℓ_j is the location of a '1' in the j^{th} group, with $\ell_j = 0$ if the group is all zeros. For instance, we will see later that,

$$\boldsymbol{c} = [\underline{0\,0}\ \ \underline{1\,0}\ \ \underline{0\,1}\ \ \underline{0\,0}\ \ \underline{0\,0}]$$

is 2-sparse. The location tuple for $P = 2$ is,

$$\boldsymbol{\ell}(\boldsymbol{c}) = (0, 1, 2, 0, 0)$$

Lemma 3.6. *Every positive element of a location tuple of a sparse sequence is strictly greater than its left neighbor.*

Proof. Clearly, a positive element cannot be smaller than its left neighbor else there will be less than $P - 1$ zeros between the corresponding 1s. The minimum weight criterion precludes positive neighbors that are equal. To see why, note that $\ell_j = k \neq 0$ implies that subsequence $\boldsymbol{q}_{k-1}$ has a 1 at coordinate j. Since all 1s must be strong, they cannot have 1s as neighbors, and the result follows. □

Note that minimizing the length of a sparse sequence is equivalent to minimizing the number of zeros, since the 1s occupy exactly tP coordinates. Lemma 3.6 implies that any time a positive element in the location tuple is not greater than its positive left neighbor (i.e., the closest left neighbor that is positive), we need at least one intervening zero. This suggests that the following class of sequences might be desirable.

Definition. If every positive element (except the element 1) in the location tuple of a sparse sequence is greater than its positive left neighbor, we say the sequence *walks*, or that it is a *walking* sequence (abbreviated w-sequence).

The nomenclature arises since the 1s in successive groups of P bits appear to drift rightward when the sequence is parsed from left to right. Note that a w-sequence is completely characterized by the locations of its positive elements. Hence, there are

$$\binom{n_{\mathrm{r}}}{tP}$$

candidate w-sequences that we must test for a distance of $2t$. Assuming that the number of required zeros in a w-sequence scales at most linearly with t, i.e., $n_{\mathrm{r}}(t, P) \leq$

$t(P + c_0)$ for some constant c_0, we may show via Stirling's approximation that[5],

$$\binom{n_r}{tP} = 2^{O(t \log P)} \tag{3.30}$$

This is significantly smaller than the total number of sequences of comparable length,

$$2^{n_r P} \geq 2^{tP^2}$$

since $n_r \geq tP$ (3.29). We will see that for large P, there appears to be no penalty in restricting sequences with good integer distance to w-sequences.

We now illustrate construction of sparse sequences for small t and P. In what follows, we will sometimes invoke the following Johnson bound [35] which applies to constant weight binary codes,

$$A(n_r, w, 2t) \leq \frac{n_r(n_r - 1)(n_r - 2) \cdots (n_r - w + t)}{w(w - 1)(w - 2) \cdots (w - t)}$$

where $A(n_r, w, 2t)$ is the cardinality of a set of constant weight (w) binary codewords that achieve a Hamming distance of $2t$. In our case, $A = n_r P$, the number of codewords, and $w \geq tP$. It may be shown that increasing the weight beyond the minimum relaxes the bound, and hence we set $w = tP$. The smallest n_r that satisfies the resulting inequality is the desired bound. While every integrated codebook of a sparse sequence is a valid constant weight binary code, the reverse is not necessarily true.

Consider sparse (not necessarily walking) sequences with $t = 1$. The bound (3.29) gives, $n_r \geq P + 1$. This may be tightened to,

$$n_r \geq P + 3 \tag{3.31}$$

To see why, we claim that for $t = 1$,

$$\binom{n_r}{P} \geq n_r P \tag{3.32}$$

[5]We use $f(x) = O(g(x))$ to indicate that there exist constants x_0 and $c_0 > 0$ such that $f(x) < c_0 g(x)\ \forall\ x > x_0$.

The quantity on the left is the total number of strings of length n_r with exactly P 1s, and that on the right is the number of integrated codewords. No two codewords may be identical, and (3.32) follows, and yields (3.31). The w-sequence,

$$\boldsymbol{\ell}(\boldsymbol{c}) = (1, 2, 3, \ldots, P, 0, 0, 0) \tag{3.33}$$

meets the bound (3.31) with equality for all P. The proof is straightforward and omitted. For example, $P = 2$ yields,

$$\boldsymbol{c} = [\underline{1\,0}\ \ \underline{0\,1}\ \ \underline{0\,0}\ \ \underline{0\,0}\ \ \underline{0\,0}]$$

and,

$$\mathcal{T}(\boldsymbol{c}) = \begin{bmatrix} 1 & 1 & 0 & 0 & 0 \\ 1 & 0 & 1 & 0 & 0 \\ 0 & 1 & 1 & 0 & 0 \\ 0 & 1 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 \\ 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 \end{bmatrix}$$

which has a distance 2.

Consider next the case $t = P = 2$. The bound (3.29) yields $n_r \geq 6$. The Johnson bound yields $n_r \geq 9$. Exhaustive searches prove that n_r cannot equal 9. There are, however, several solutions of sample length 10. E.g,

$$\boldsymbol{\ell}(\boldsymbol{c}) = (1, 2, 0, 0, 0, 0, 1, 0, 2, 0)$$

i.e.,

$$\boldsymbol{c} = [\underline{1\,0}\ \ \underline{0\,1}\ \ \underline{0\,0}\ \ \underline{0\,0}\ \ \underline{0\,0}\ \ \underline{0\,0}\ \ \underline{1\,0}\ \ \underline{0\,0}\ \ \underline{0\,1}\ \ \underline{0\,0}]$$

achieves a distance of 4.

As a final example, consider $t = 3$ and $P = 2$. The bound (3.29) yields 9, the Johnson bound is 11, and the best known constant weight code with $d = 2t = 6$, $w = tP = 6$, and $A(n_{\mathrm{r}}, d, w) \geq Pn_{\mathrm{r}}$ has length 13 [9]. Computer searches show that the shortest sparse sequence for these parameters has length 15. One such sequence is,

$$\boldsymbol{\ell}(\boldsymbol{c}) = (1, 2, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 2, 0, 0) \tag{3.34}$$

and $\boldsymbol{c}$ has integrated distance 6. These examples seem to suggest that it might be possible to construct sparse sequences of length $t(P + 3)$ with distance $2t$. We will introduce a sub-family of w-sequences before presenting a conjecture to this effect.

Definition. Suppose that a w-sequence with distance $2t$ and length n_{r} has $t \mid n_{\mathrm{r}}$. Suppose further that the location tuple of this sequence (or some suitable shift) may be divided into t successive blocks of equal length, such that each block has an identical number of zeros. We call such a sequence a *block* w-sequence.

A block w-sequence with n_{z} zeros per block may be specified by a set of t n_{z}-tuples that identify the locations of the zeros in the corresponding block,

$$\mathcal{B}(\boldsymbol{c}) = (\boldsymbol{b}_1, \boldsymbol{b}_2, \ldots, \boldsymbol{b}_t)$$

The length 15 sequence in (3.34) is a block sequence,

$$\boldsymbol{\ell}(\boldsymbol{c}) = (\underbrace{1, 2, 0, 0, 0}_{\text{Block } 1}, \underbrace{0, 0, 1, 2, 0}_{2}, \underbrace{1, 0, 2, 0, 0}_{3})$$

and may be specified thus,

$$\mathcal{B}(\boldsymbol{c}) = \big((3, 4, 5), (1, 2, 5), (2, 4, 5)\big)$$

Conjecture 3.4. *There exists a block w-sequence of length $n_{\mathrm{r}} = t(P + 3)$ for any given t and P, which achieves a distance of $2t$.*

We have verified this conjecture for all $t \leq 8$ and $P \leq 16$. Note that in light of the lower bound (3.29), block w-sequences are optimum integer distance sequences in the limit of large P. Clearly, there are no more than

$$\binom{P+3}{3}^t = O\left(P^{3t}\right) = 2^{O(t \log P)}$$

candidate sequences with block size $P+3$ for given t, P. (The expression overcounts since it ignores shifted representations of the same sequence.) Note that the order of the exponent is identical to that of general w-sequences (3.30), though it may be shown that the actual exponent is smaller. While this is considerably less than the set of all possible sequences, it is still a large number. For instance, for $t = 8$, and $P = 16$, it is about 2^{79} candidate sequences, each $P \cdot t \cdot (P+3) = 2432$ bits long! In practice, two factors substantially mitigate this complexity. First, most of the blocks have the form in (3.33). Next, there exists a threshold $P_{\text{lim}}(t)$, such that the block structure may be reused for any $P \geq P_{\text{lim}}(t)$.

Table 3.2 lists w-sequences with block size $P+3$ and distance $2t$. We use the threshold effect to specify the block structure for all $P \geq P_{\text{lim}}(t)$. Such a construction is only a conjecture for now, but has been verified for $P \leq 16$. Note that the block specifiers $\boldsymbol{b}_i$ are 3-tuples, and all unspecified blocks are equal to,

$$\boldsymbol{b}_I = (P+1, P+2, P+3)$$

or the *identity* block. Sequences when $P < P_{\text{lim}}(t)$ are in appendix B.1.

As an illustration, consider $t = 4, P = 7$. Since $P > P_{\text{lim}}(4) = 4$, the construction in the table applies, and the desired sequence is,

$$\mathcal{B}(\boldsymbol{c}) = \big((8,9,10),(8,9,10),(8,9,10),(5,6,10)\big)$$

with $(8,9,10)$ being the identity block. This is a 280 bit long ($= Pt(P+3)$), sparse sequence with a distance of $2t = 8$. This is within about 0.8 dB of the lower bound (3.29), which evaluates to 224 bits for any sequence with this integer distance, sparse

$t = 1, P \geq 2$	$\boldsymbol{b}_1 = (P{+}1, P{+}2, P{+}3)$
$t = 2, P \geq 3$	$\boldsymbol{b}_2 = (P{-}1, P{+}2, P{+}3)$
$t = 3$ or $4, P \geq 4$	$\boldsymbol{b}_t = (P{-}2, P{-}1, P{+}3)$
$t = 5, P \geq 4$	$\boldsymbol{b}_4 = (P, P{+}2, P{+}3)$, $\boldsymbol{b}_5 = (P{-}3, P{-}2, P{+}3)$
$t = 6, P \geq 4$	$\boldsymbol{b}_5 = (P{-}1, P{+}1, P{+}2), \boldsymbol{b}_6 = (1, P{-}1, P{+}3)$
$t = 7, P \geq 5$	$\boldsymbol{b}_6 = (P{-}4, P{-}2, P{-}1)$, $\boldsymbol{b}_7 = (1, P{-}3, P{+}3)$
$t = 8, P \geq 5$	$\boldsymbol{b}_6 = (P, P{+}2, P{+}3)$, $\boldsymbol{b}_7, \boldsymbol{b}_8$ correspond to $\boldsymbol{b}_6, \boldsymbol{b}_7$ for $t = 7$

Table 3.2: Block w-sequences for arbitrary $P \geq P_{\text{lim}}(t)$, conjectured to achieve a distance of $2t$. Only non-identity blocks are specified.

or otherwise. (The Johnson bound is 217 bits, and hence not as tight.) If $P = 3 < P_{\text{lim}}(4)$, the table in appendix B.1 gives,

$$\mathcal{B}(\boldsymbol{c}) = \big((8,9,10),(8,9,10),(3,5,6),(2,3,6)\big)$$

which is a 72 bit sparse sequence with distance 8, about 1.5 dB away from the bound (3.29), which yields 51 bits.

3.5.1 Distance Comparison

If conjecture 3.4 holds,

$$d_{\text{rt}} = 2t = 2\frac{n_{\text{r}}}{P+3} \approx \frac{2n_{\text{r}}}{P} \quad (P \text{ large})$$

O-H sequences have the best root distance of sequences studied thus far,

$$d_{\text{rt}} = \frac{n_{\text{r}} + P}{2P} \approx \frac{n_{\text{r}}}{2P} \quad (n_{\text{r}} \gg P)$$

Hence, for large t, P, block w-sequences improve root distance by a factor of 4, or 6 dB. Note that the squared distance of block w-sequences is a factor of $P/2$ worse than MSDs, which is unsurprising since these sequences were optimized for root distance.

3.5.2 Almost-block w-sequences

Is it possible to construct w-sequences with a block size less than $P+3$? We showed earlier that there are t, P tuples for which this is impossible. Hence, such sequences would not exist for arbitrary t, P. However, computer searches suggest that for all $t > 1$, and P sufficiently large, we can achieve a block size that approaches $P+2$ as $t \to \infty$.

Conjecture 3.5. *For every $t > 1$, and $P > P_{\lim}(t)$, there exists an* almost-block *w-sequence with distance $2t$, that has $t-1$ blocks of size $P+2$ and one block of size $P+3$.*

The conjecture has been verified for $t \leq 8$ and $P \leq 16$. Almost-block sequences permit root distance improvements of 0.4–0.8 dB compared with block sequences. Table 3.3 presents the construction of almost-block sequences. As an example, the almost-block sequence for $t = 4, P = 7$ is seen from the table to be,

$$\mathcal{B}(\boldsymbol{c}) = \big((8,9,10),(8,9),(6,9),(5,9)\big)$$

which is a 259 bit sequence compared with the 280 bit block sequence discussed earlier.

Table 3.4 summarizes the classes of sequences and the tradeoff between length and search complexity.

3.6 Simulation Results

We now study the simulated performance of sequences in the low and high SNR regimes, with small and large P values. In conventional coding, the difference between asymptotic and measured gains is best understood via the distance spectrum: the frequency of distances from a codeword to the rest. Predicting departures from asymptotic performance is more involved in our setup. First, the codebook geometry changes with SNR. The underlying distance function evolves smoothly from Euclidean

$t=2, P \geq 4$	$\boldsymbol{b}_2 = (P-1, P+2)$
$t=3, P \geq 7$	$\boldsymbol{b}_3 = (P-2, P+2)$
$t=4, P \geq 7$	$\boldsymbol{b}_3 = (P-1, P+2)$,
	$\boldsymbol{b}_4 = (P-2, P+2)$
$t=5, P \geq 8$	$\boldsymbol{b}_4 = (\lceil P/2 \rceil - 1, P-1)$,
	$\boldsymbol{b}_5 = (1, P+2)$
$t=6, P \geq 7$	$\boldsymbol{b}_4 = (P, P+2)$,
	$\boldsymbol{b}_5 = (P-4, P+2)$,
	$\boldsymbol{b}_6 = (1, P+2)$
$t=7, P \geq 7$	$\boldsymbol{b}_5 = (\lceil P/2 \rceil - 1, P-1)$,
	$\boldsymbol{b}_6 = (1, P+2)$,
	$\boldsymbol{b}_7 = (P, P+2)$
$t=8, P \geq 7$	$\boldsymbol{b}_5 = (P, P+2)$,
	$\boldsymbol{b}_6 = (P-2, P+2)$,
	$\boldsymbol{b}_7 = (\lceil P/2 \rceil - 1, P-1)$,
	$\boldsymbol{b}_8 = (1, P+2)$

Table 3.3: Almost-block w-sequences conjectured to achieve a distance of $2t$ (verified for $P \leq 16$). The first block has length $P+3$ and the rest $P+2$. Only non-identity blocks are specified. Sequences for P values not included here are in appendix B.2.

Sequence Type	Bound on Length	Search Complexity
Unconstrained	$\geq t(P+1)+1$	$2^{O(tP^2)}$
Sparse	$\geq t(P+1)+1$	$2^{O(tP \log P)}$
Almost-Block†	$t(P+2)+1$	$2^{O(t \log P)}$
Block †	$t(P+3)$	$2^{O(t \log P)}$

Table 3.4: Bounds on sample length for an integer distance of $2t$, versus search complexity (number of candidate sequences) of various families. The † denotes results that are conjectures.

to root-Euclidean as SNR is swept from 0 to ∞. Hence, the squared and root distance gains may not always apply at moderate SNRs. Also, the integrated codebook is characterized by a distance spectra for *each phase* (not all necessarily unique), instead of just one for the entire codebook, as in linear codebooks.

In what follows, the number of nearest Hamming, squared, and root distance neighbors are denoted by m_{H}, m_{sq}, and m_{rt} respectively. When several spectra exist, we choose one with the highest number of nearest neighbors for the relevant metric.

We study sequences when P=2, the smallest non-trivial value, and P=16, which is indicative of the largest integration periods in current systems [33, 49, 40]. For each case, we compare the SNR required by sequences of a given length, and also lengths required for a given SNR, to achieve identical performance. We use two modes of comparison since the translation of coding gain to a reduction in length is SNR dependent.

Note that we would expect gains from new sequences to increase with P. This is because their advantage is predicated on departures from conventional geometry, and such departures are small for small P (all proposed distance functions are identical for P=1). Second, an increasing number of subsequences appears to offer greater combinatorial opportunity.

3.6.1 Low SNR

Operation at low SNRs requires large sequence lengths. As in practical systems, we overcome this problem by repeating shorter sequences followed by a SFD. Note that this is different from using periodic sequences which repeat a sequence of sample length 2. We estimated the SFD lengths required for the sequences in this section, and found that including the overhead changes our comparisons only marginally. Hence, we do not simulate SFDs.

We start with P=2. Table 3.5 shows the parameters and performance of three sequences: a (n_r=30, d_{sq}=8) O-H, (31,16) I-H, and (31,24) SD[6]. All SNR figures quoted in this section are for a probability of codeword error (CER) of 10^{-4}. Figure 3-2(a) plots the performance of these sequences when each is repeated 16x. The I-H sequence is 1 dB, and the SD sequence 1.7 dB, better than the O-H sequence. We now compare lengths required for equal performance. The I-H and SD sequence require 12 and 9 repetitions, respectively, to match 16 repetitions of the O-H sequence. Hence, the SD sequence requires only about half the length of the O-H sequence.

Next, we compare a (112, d_{sq}=4) O-H, (107,54) I-H and (111,110) MSD sequence for P=16. Table 3.6 summarizes the key parameters of these sequences. Figure 3-2(b)

[6]SD: An abbreviation for sequences with good, but not necessarily maximum, squared distance.

Seq.	n_r	d_{sq}	m_{sq}	d_H	m_H	Rep.	SNR (dB)
O-H	30	8	2	8	2	16	2.6
I-H	31	16	4	16	32	16	1.6
SD	31	24	7	13	1	16	0.9
I-H	As above					12	2.4
SD						9	2.4

Table 3.5: Sequences and their performance in the low SNR regime with P=2. *Rep.* denotes number of times a sequence is repeated.

plots their performance for a repetition of 64x. The squared distance gains predict SNR gains of 5.7 dB and 7.2 dB for the I-H and MSD sequence respectively. (Recall that the exponent is proportional to squared distance, and *square* of the SNR in the low SNR regime.) These track the measured gains of 4.7 dB and 6.6 dB. The table also confirms the primacy of squared, rather than Hamming distance in this regime. The O-H and MSD sequences have identical Hamming distances and neighbors. Yet, the latter has significantly better performance. Finally, the I-H and MSD sequences reduce length by a factor of 4.5x and 9x respectively, when compared with the O-H sequence.

Note that the gains are significantly greater for P=16 when compared with those for P=2. Also, P=16 requires sequence lengths that are an order of magnitude greater than those for P=2. This is consistent with an exponent that varies inversely with P at low SNRs.

Seq.	n_r	d_{sq}	m_{sq}	d_H	m_H	Rep.	SNR (dB)
O-H	112	4	2	4	2	64	4.7
I-H	107	54	3	54	136	64	0.0
MSD	111	110	2	4	2	64	-1.8
I-H	As above					14	4.7
MSD						7	4.7

Table 3.6: Sequences and their performance in the low SNR regime for P=16.

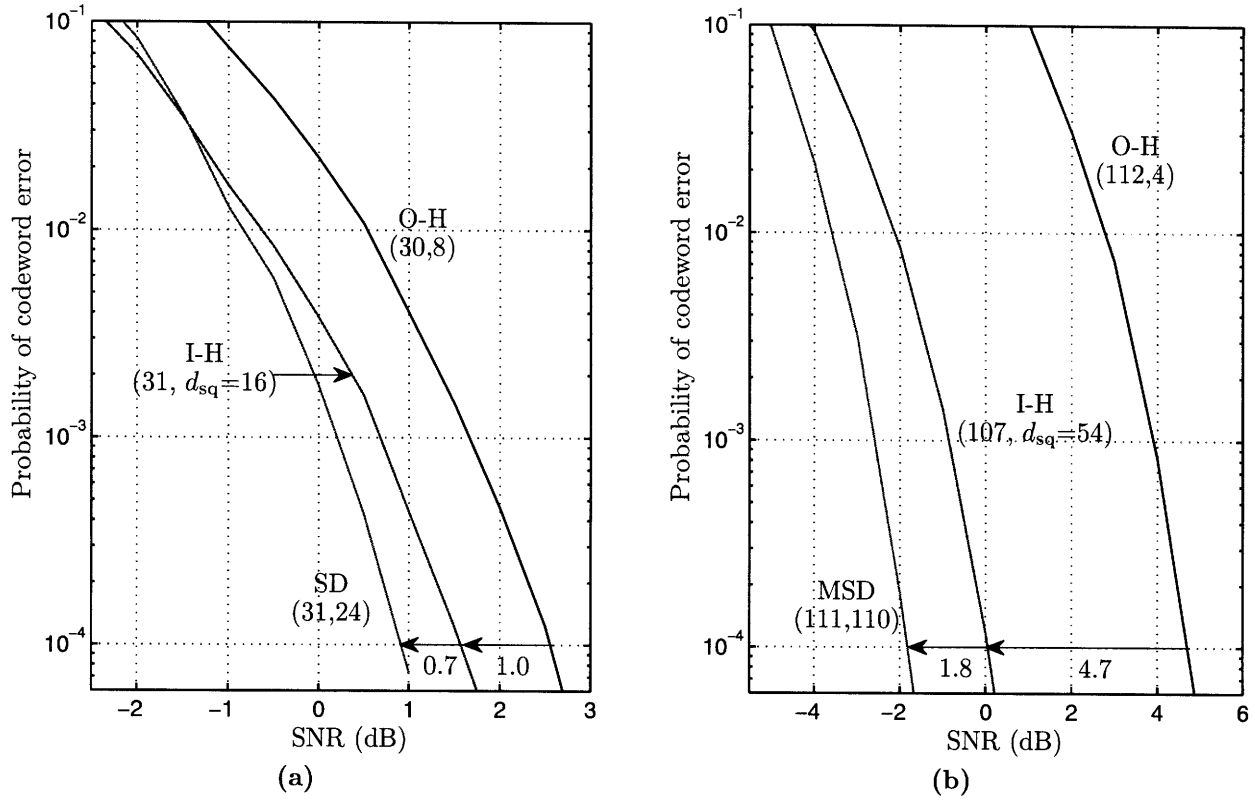

Figure 3-2: Performance in the low SNR regime with (a) P=2, and each sequence being repeated 16x, and, (b) P=16, and every sequence repeated 64x.

3.6.2 High SNR

For P=2, we compare (14, d_{rt}=4) O-H, (15,6) block w-sequence, and (15,6.3) SD sequences (table 3.7, and figure 3-3). Note that the MSD has a marginally better root distance than the w-sequence, though calculations show that they have the same integer distance. This is unsurprising, since all sequence families tend to overlap for short lengths and small values of P. The MSD has a SNR gain of 1.5 dB and reduces length by a factor of 1.4x compared with the O-H sequence.

For P=16, we compare a (109, d_{rt}=3.4) MSD and a (109,12) almost-block w-sequence to a baseline (112,4) O-H sequence (table 3.8, and figure 3-4(a)). Note that the MSD and O-H sequence have identical Hamming distances, and number of root and Hamming neighbors. Yet, despite a *smaller* root distance, the MSD outperforms the O-H sequence by 3.4 dB. This shows that the squared distance criterion continues to be

Seq.	n_r	d_{rt}	m_{rt}	d_{sq}	m_{sq}	d_H	m_H	SNR(dB)
O-H	14	4	2	4	2	4	2	13.1
w-seq.	15	6	14	6	14	6	14	12.4
SD	15	6.3	10	12	7	6	4	11.6
SD	11	4.7	6	8	9	4	2	12.7

Table 3.7: Performance of sequences in the high SNR regime for P=2.

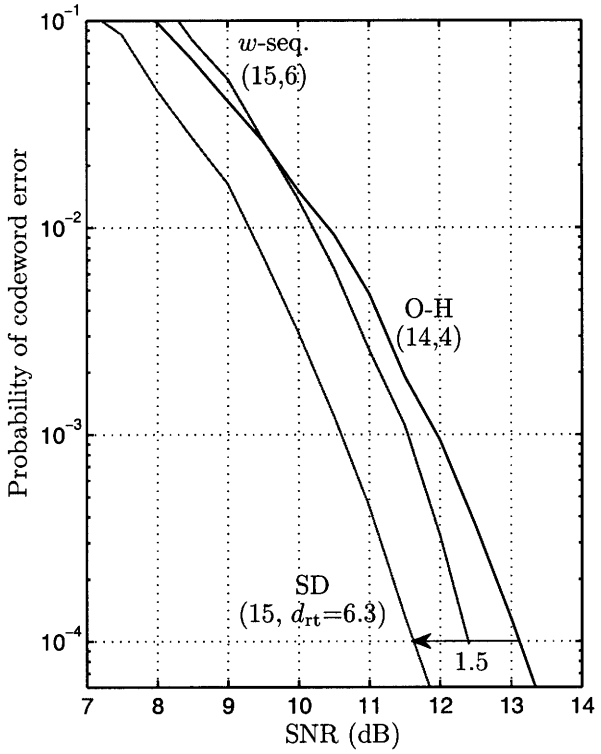

Figure 3-3: Performance in the high SNR regime (P=2).

relevant at these SNRs. This also explains the MSD's 0.9 dB advantage over the w-sequence, despite a root distance that is only about a quarter of the latter. Note however, that the *slopes* of the three error curves qualitatively reflect their root distances. For instance, extrapolating from this figure, we expect the w-sequence to have the best performance in the 10^{-5} to 10^{-6} region.

The sequences in figure 3-4(b) bear this out. The SNR is about 6 dB higher than the previous case, and performance qualitatively tracks root rather than squared distances. A (19, d_{rt}=2) block w-sequence achieves a gain of about 1.1 dB and 3.7

Seq.	n_r	d_{rt}	m_{rt}	d_{sq}	m_{sq}	d_H	m_H	SNR (dB)
O-H	112	4	2	4	2	4	2	15.6
w-seq.	109	12	18	12	18	12	18	13.1
MSD	109	3.4	2	108	2	4	2	12.2
w-seq.	55	6	30	6	30	6	30	15.1
MSD	45	1.4	2	44	2	4	2	15.9

Table 3.8: Sequences and their performance in the high SNR regime for P=16.

dB compared with a (19, d_{rt}=0.6) MSD and (19, d_{rt}=0.3) I-H sequence, respectively (table 3.9). The advantage is not as large as root distances would predict, and this probably points again to the slow transition from the squared to root distance regime (and, perhaps, to a lesser extent, the significantly larger m_{rt} of the w-sequence). The w-sequence is about 2.5x shorter than the baseline (48,2) O-H sequence, which is the shortest sequence in that family for P=16.

Seq.	n_r	d_{rt}	m_{rt}	d_{sq}	m_{sq}	d_H	m_H	SNR (dB)
I-H	19	0.31	2	10	3	10	48	22.2
MSD	19	0.56	2	18	3	4	2	19.6
w-seq.	19	2	34	2	34	2	34	18.5
O-H	48	2	2	2	2	2	2	17.7
I-H	47	0.75	2	24	3	24	76	18.4
MSD	25	0.75	2	24	2	4	2	18.4

Table 3.9: Sequences for the high SNR regime (P=16).

3.6.3 Summary of Improvements

Table 3.10 summarizes the benefits of using new sequences instead of O-H ones. Both the SNR gain and the factor by which length is reduced is shown. Length reductions are quoted for the synchronization sequence by itself, and for the entire packet. Improvements are calculated for two payload sizes – 10 and 100 bits – which represent telemetry-type sensing applications. Pulse position modulation (PPM) is assumed for payload bits, and repetition coding used to achieve a packet error rate

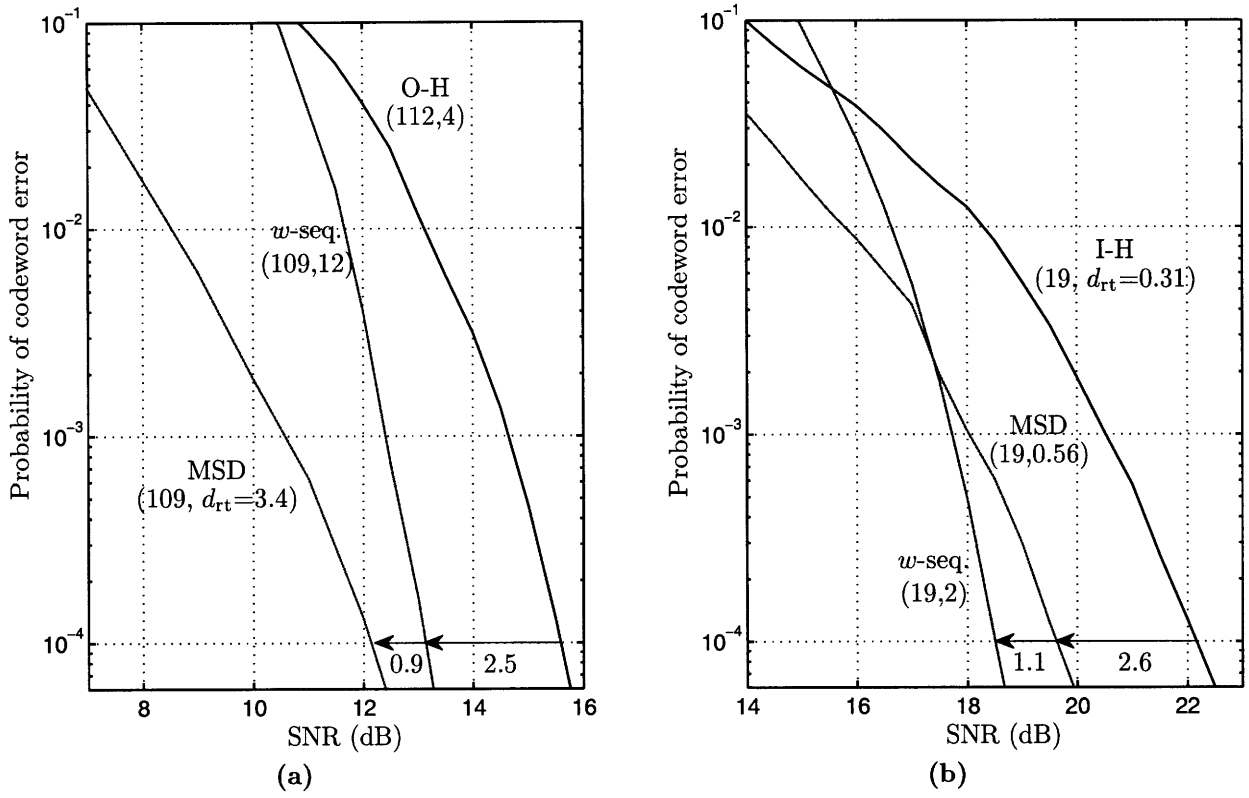

Figure 3-4: Performance for P=16 in the high SNR regime.

(PER) of 10^{-3}. (Ignoring startup costs, more aggressive coding reduces payload size, increasing the impact of shorter preambles). Additional overhead for detection, automatic gain control, SFD, etc. is not included. Note that gains increase with P and with decreasing SNR, and are close to an order of magnitude for $P = 16$ at low SNR.

			Reduction in length (factor)		
SNR	P	Gain (dB)	Synch.	10 bit	100 bit
Low	2	1.7	1.7	1.3	1.0
Low	16	6.5	9.2	8.8	6.4
High	2	1.5	1.3	1.1	1.0
High	16	3.4	2.5	2.0	1.3

Table 3.10: Gains realized when using new sequences in place of O-H sequences.

3.7 Average Versus Peak Power Constraint

The synchronization problem can be stated with a constraint on the average, rather than peak, power.

Problem 3.1. Design of good synchronization sequences:

Given: n, P, p_{av}.

Solution: Binary sequence $\boldsymbol{c}$ of length n, and a symbol SNR.

Constraint on the solution:

$$\frac{w(\boldsymbol{c})\,\text{SNR}}{n} \leq p_{\text{av}} \tag{3.35}$$

Measure of the solution: $\rho\left(\mathcal{C}(\boldsymbol{c}); P, \text{SNR}\right)$.

Problem: Find the solution that maximizes the measure while obeying the constraint.

When SNR $\rightarrow$ 0, the problem of finding the optimum sequence $\boldsymbol{c}_{\text{opt}}$ may be recast thus,

$$\begin{aligned}
\boldsymbol{c}_{\text{opt}}(n) &= \arg \max_{\boldsymbol{c}\in\{0,1\}^n} \rho(\mathcal{C}(\boldsymbol{c}); P, \text{SNR}) \\
&= \arg \max\ d_{\text{sq}}\left(\mathcal{T}(\boldsymbol{c})\right) \frac{\text{SNR}^2}{16P} \\
&= \arg \max\ \frac{d_{\text{sq}}\left(\mathcal{T}(\boldsymbol{c})\right)}{w(\boldsymbol{c})^2}
\end{aligned}$$

since $w(\boldsymbol{c})\,\text{SNR} = np_{\text{av}}$ ((3.35) holds with equality since distances are monotonic in SNR). Recall the elementary distance bound (3.9),

$$\frac{d(\mathcal{T}(\boldsymbol{c}))}{w(\boldsymbol{c})} \leq \frac{2}{P}$$

Also, non-zero distance requires $w(\boldsymbol{c}) \geq P$ (no subsequence may be all zeros). A sparse sequence meets (3.9) with equality. Hence, if a sparse sequence of weight P exists for the specified length, it is optimum. It may be shown that for lengths

$n_r \geq P + 3$, the w-sequence,

$$\ell(\boldsymbol{c}) = (1, 2, 3, \ldots, P, \underbrace{0, 0, \ldots, 0}_{n_r - P \text{ zeros}})$$

satisfies this criterion. We can similarly prove that *any* sparse sequence that exists for the specified length is optimum under the average power constraint as SNR $\rightarrow \infty$.

Note that a O-H sequence of length $n_r = P\tilde{n}_r$ has a weight of $P(\tilde{n}_r \pm 1)/2$, and a distance of $(\tilde{n}_r + 1)/2$. Hence,

$$\frac{d(\mathcal{T}(\boldsymbol{c}))}{w(\boldsymbol{c})} = \frac{\tilde{n}_r + 1}{P(\tilde{n}_r - 1)} \approx \frac{1}{P} \qquad (n_r \gg 1)$$

which is a factor of two worse than the w-sequence above. Note that there is a trivial O-H sequence with $\tilde{n}_r = 1$, which does not incur this loss. However, using such a low weight sequence will likely cause peak power specifications to be violated. (This is also true of a w-sequence of the same weight.) This problem may be avoided at high SNRs by using w-sequences of higher weight, which is not possible with O-H sequences without incurring the 3 dB loss.

3.8 Summary

This work has only initiated an investigation of good synchronization sequences over the ES channel. Incorporating additional constraints will facilitate the adoption of our techniques in practical systems:

- *Multipath*: Synchronization in the presence of multipath is critical for UWB systems.
- *Simultaneous peak and average power constraints.*
- SNR *unknown to the transmitter.*
- *Heterogeneous network*: Sequences might have to work with both coherent receivers, and energy sampling front-ends with different integration periods.

From a theoretical perspective, the original problem of maximizing the exponent for a given SNR is still unsolved. In conventional synchronization, there is a single optimum sequence for a given length. In our problem, there is a *set* of optimum sequences, each of which maximizes the exponent over a certain SNR *interval* (since there are only countably many sequences). This is an interesting set to explore, particularly if it turns out to have a cardinality that is polynomial in n_{r}, P. The following characterization might yield some insights. Consider the set of all sequences that are pareto optimum with respect to the squared and root distances. In other words, no other sequence can strictly improve both distances for a member of this set. This work has only considered the edges of the pareto frontier. Are optimum sequences necessarily on this frontier? Are frontier sequences necessarily optimum? Part of the difficulty in answering these questions is the intractability of the codeword distance (i.e. the exponent) at an arbitrary SNR. A partial ordering of codebook distances does not, however, require an explicit knowledge of distances. For instance, consider the following conjecture.

Conjecture 3.6. *If, for two codeword pairs* $(\boldsymbol{c}, \boldsymbol{c}')$ *and* $(\boldsymbol{b}, \boldsymbol{b}')$,

$$d(\boldsymbol{t}(\boldsymbol{c}), \boldsymbol{t}(\boldsymbol{c}')) > d(\boldsymbol{t}(\boldsymbol{b}), \boldsymbol{t}(\boldsymbol{b}'))$$

for both $d = d_{\mathrm{sq}}$, *and* $d = d_{\mathrm{rt}}$, *then,*

$$\rho(\boldsymbol{c}, \boldsymbol{c}'; \mathrm{SNR}) > \rho(\boldsymbol{b}, \boldsymbol{b}'; \mathrm{SNR})$$

for all SNR.

If true, this might allow proving that some (non-trivial) sequences are strictly inferior to others at all SNRs. Other approaches that exploit the convexity of distance metrics may also be fruitful.

3.9 Selected Proofs

3.9.1 Chernoff Information of χ^2 r.v.s as SNR $\to 0, \infty$

We begin by reviewing chi-square r.v.s and some notation. Most of this material is drawn from [42, Sec. 2.1.4].

If $\mathsf{x}_i \sim \mathcal{N}(\mu_i, \sigma_i^2)$ are independent Gaussian r.v.s, then the r.v.,

$$\mathsf{y} = \sum_{i=0}^{n-1} \mathsf{x}_i^2$$

has a chi-square distribution, $\chi_n^2(s^2)$, with n degrees of freedom, and *centrality parameter*, $s^2 = \sum_i \mu_i^2$. The r.v. y is said to be distributed *centrally* if $s^2 = 0$, and *non-centrally* otherwise. The distribution has moments,

$$\begin{aligned} \mathrm{E}\mathsf{y} &= s^2 + n\sigma^2 \\ \sigma_{\mathsf{y}}^2 &= 2n\sigma^4 + 4ns^2\sigma^2 \end{aligned} \tag{3.36}$$

We assume henceforth that $\sigma^2 = 1$. The pdf of a central χ^2 r.v. is given by,

$$p_{\mathsf{y}}(y) = \frac{1}{2^{n/2}\Gamma(n/2)} y^{\alpha} e^{-\frac{y}{2}}$$

where $\alpha = n/2 - 1$. Non-central r.v.s are distributed thus,

$$p_{\mathsf{y}}(y) = \frac{1}{2}\left(\frac{y}{s^2}\right)^{\alpha/2} e^{-\frac{y+s^2}{2}} I_{\alpha}\left(\sqrt{ys^2}\right) \tag{3.37}$$

where I_α is the modified Bessel function of the first kind.

Recall that in the context of a binary hypothesis testing problem with likelihoods,

$$\mathsf{y} \sim p_i(y) \text{ under hypothesis } \theta_i,\ i \in \{0, 1\}$$

the Chernoff information (C.I.), ρ_{y}, is given by the Chernoff Bound,

$$\rho_{\mathsf{y}} = -\ln \min_{\lambda<0} \mathrm{E}_0\, e^{\lambda \ell(y)}$$

where, $\ell(y)$ is the log-likelihood ratio,

$$\ell(y) = \ln \frac{p_0(y)}{p_1(y)}$$

and E_i denotes the expectation under θ_i.

We now state three propositions related to C.I. when the likelihoods are χ^2 r.v.s,

$$\mathsf{y} \sim \chi^2_P \left(s_i^2 = t_i \mathrm{SNR}\right) \qquad i = 0, 1$$

Theorem 3.8.

$$\mathcal{D}(p_0 \parallel p_1) \doteq \frac{\left(s_0^2 - s_1^2\right)^2}{4P} = \frac{(t_0 - t_1)^2 \mathrm{SNR}^2}{4P} \qquad (\mathrm{SNR} \to 0) \qquad (3.38a)$$

and,

$$\mathcal{D}(p_0 \parallel p_1) \doteq \frac{\left(\sqrt{s_0^2} - \sqrt{s_1^2}\right)^2}{2} = \frac{(\sqrt{t_0} - \sqrt{t_1})^2 \mathrm{SNR}}{2} \qquad (\mathrm{SNR} \to \infty) \qquad (3.38b)$$

Corollary 3.4.

$$\rho_{\mathsf{y}} \doteq \frac{(t_0 - t_1)^2 \mathrm{SNR}^2}{16P} \qquad (\mathrm{SNR} \to 0)$$

and,

$$\rho_{\mathsf{y}} \doteq \frac{(\sqrt{t_0} - \sqrt{t_1})^2 \mathrm{SNR}}{8} \qquad (\mathrm{SNR} \to \infty)$$

Corollary 3.5. *For a vector r.v. with independently distributed χ^2 elements,*

$$\mathbf{y} \sim \chi_P^2\left(\boldsymbol{t}_i \mathrm{SNR}\right) \triangleq \left(\chi_P^2\left(t_{i,0}\mathrm{SNR}\right), \chi_P^2\left(t_{i,1}\mathrm{SNR}\right), \ldots, \chi_P^2\left(t_{i,n-1}\mathrm{SNR}\right)\right) \qquad i = 0, 1$$

we have,

$$\rho_{\mathbf{y}} \doteq \left\|\boldsymbol{t}_0 - \boldsymbol{t}_1\right\|^2 \frac{\mathrm{SNR}^2}{16P} \qquad (\mathrm{SNR} \to 0)$$

and,

$$\rho_{\mathbf{y}} \doteq \left\|\sqrt{\boldsymbol{t}_0} - \sqrt{\boldsymbol{t}_1}\right\|^2 \frac{\mathrm{SNR}}{8} \qquad (\mathrm{SNR} \to \infty)$$

The last corollary does not follow directly from the one before since C.I. is not always additive (unlike K-L divergence).

Proof of theorem 3.8. The proof is a straightforward, if tedious, exercise in Taylor series expansions. A more elegant proof may be constructed for the $\mathrm{SNR} \to 0$ case using a theorem of Chernoff's [12]. We do not include that proof here.

We exclude the trivial case $t_0 = t_1$, and further assume that $t_i \neq 0$ i.e. both distributions are non-central. The case when one of the variables is central follows the same steps.

Plugging the density function (3.37) into,

$$\mathcal{D}(p_0 \parallel p_1) \triangleq \int p_0(y) \ln \frac{p_0(y)}{p_1(y)}\, dy$$

yields,

$$\mathcal{D}(p_0 \parallel p_1) = \frac{\alpha}{2} \ln\left(\frac{t_1}{t_0}\right) - \frac{t_0 - t_1}{2}\mathrm{SNR} + \int p_0(y) \ln \frac{I_\alpha\left(\sqrt{y t_0 \mathrm{SNR}}\right)}{I_\alpha\left(\sqrt{y t_1 \mathrm{SNR}}\right)}\, dy \qquad (3.39)$$

From the series expansion,

$$I_\alpha\left(\sqrt{x}\right) = \left(\frac{x}{4}\right)^{\alpha/2} \sum_{k=0}^{\infty} \frac{\left(\frac{x}{4}\right)^k}{k!\Gamma\left(\alpha+k+1\right)} \qquad (x \geq 0) \tag{3.40}$$

if follows that,

$$I_\alpha\left(\sqrt{x}\right) = \left(\frac{x}{4}\right)^{\alpha/2} \left(\Gamma\left(\alpha+1\right)^{-1}\left(1+a_1x+a_2x^2\right)+o\left(x^2\right)\right) \qquad (x \to 0)$$

where, $a_1 = \dfrac{\Gamma\left(\alpha+1\right)}{4\Gamma\left(\alpha+2\right)}$, and $a_2 = \dfrac{\Gamma\left(\alpha+1\right)}{32\Gamma\left(\alpha+3\right)}$. Using the series expansion for $\ln(1+x)$ we may show that,

$$\ln\left(1+a_1x+a_2x^2+o\left(x^2\right)\right) = a_1x + \left(a_2 - \frac{a_1^2}{2}\right)x^2 + o\left(x^2\right) \quad (x \to 0)$$

It follows that,

$$\ln I_\alpha\left(\sqrt{x}\right) = \frac{\alpha}{2}\ln\frac{x}{4} - \ln\Gamma\left(\alpha+1\right) + a_1x + \left(a_2 - \frac{a_1^2}{2}\right)x^2 + o\left(x^2\right) \qquad (x \to 0)$$

and hence,

$$\begin{aligned}\ln\frac{I_\alpha\left(\sqrt{yt_0\mathrm{SNR}}\right)}{I_\alpha\left(\sqrt{yt_1\mathrm{SNR}}\right)} &= \frac{\alpha}{2}\ln\left(\frac{t_0}{t_1}\right) + a_1(t_0-t_1)\mathrm{SNR}\,y + \left(a_2-\frac{a_1^2}{2}\right)(t_0^2-t_1^2)\mathrm{SNR}^2y^2 + o\left(\mathrm{SNR}^2\right) \\ &\triangleq \frac{\alpha}{2}\ln\left(\frac{t_0}{t_1}\right) + b_1\mathrm{SNR}\,y + b_2\mathrm{SNR}^2y^2 + o\left(\mathrm{SNR}^2\right) \qquad (\mathrm{SNR} \to 0)\end{aligned}$$

where, $b_1 = a_1(t_0-t_1)$, and $b_2 = \left(a_2 - \frac{a_1^2}{2}\right)(t_0^2-t_1^2)$. This gives,

$$\int p_0(y)\ln\frac{I_\alpha\left(\sqrt{yt_0\mathrm{SNR}}\right)}{I_\alpha\left(\sqrt{yt_1\mathrm{SNR}}\right)}\,dy = \frac{\alpha}{2}\ln\left(\frac{t_0}{t_1}\right) + b_1\mathrm{SNR}\,\mathrm{E}_0\,y + b_2\mathrm{SNR}^2\mathrm{E}_0\,y^2 + o\left(\mathrm{SNR}^2\right) \qquad (\mathrm{SNR} \to 0) \tag{3.41}$$

From (3.36),

$$\mathrm{E}_0\,y = P + t_0\mathrm{SNR}$$

$$\mathrm{E}_0\,y^2 = \mathrm{E}_0{}^2y + 2P + 4t_0\mathrm{SNR}$$

Substitution in (3.41) and some simplification yields,

$$\int p_0(y) \ln \frac{I_\alpha\left(\sqrt{yt_0\mathrm{SNR}}\right)}{I_\alpha\left(\sqrt{yt_1\mathrm{SNR}}\right)}\, dy = \frac{\alpha}{2}\ln\left(\frac{t_0}{t_1}\right) + \frac{t_0 - t_1}{2}\mathrm{SNR} + \frac{(t_0 - t_1)^2}{4P}\mathrm{SNR}^2 + o\left(\mathrm{SNR}^2\right)$$

which, with (3.39), yields the desired result (3.38a).

For $\mathrm{SNR} \to \infty$ we use the expansion [1, Sec. 9.7.],

$$I_\alpha(x) = \frac{e^x}{\sqrt{2\pi x}}\left(1 + \sum_{n=1}^{\infty} c_n x^{-n}\right) \qquad (x \text{ large}) \tag{3.42}$$

This gives,

$$\ln I_\alpha(x) = x + o(x) \qquad (x \to \infty)$$

and hence,

$$\begin{aligned}\int p_0(y) \ln \frac{I_\alpha\left(\sqrt{yt_0\mathrm{SNR}}\right)}{I_\alpha\left(\sqrt{yt_1\mathrm{SNR}}\right)}\, dy &= (\sqrt{t_0} - \sqrt{t_1})\sqrt{\mathrm{SNR}}\ \mathrm{E}_0\sqrt{y} + o\left(\sqrt{\mathrm{SNR}}\right) \\ &= (t_0 - \sqrt{t_0 t_1})\mathrm{SNR} + o(\mathrm{SNR})\end{aligned} \qquad (\mathrm{SNR} \to \infty) \tag{3.43}$$

where the last step uses,

$$\mathrm{E}_0\sqrt{y} = \sqrt{t_0\mathrm{SNR}} + o\left(\sqrt{\mathrm{SNR}}\right) \qquad (\mathrm{SNR} \to \infty)$$

(We omit the proof here.) Substituting (3.43) in (3.39), we get,

$$\begin{aligned}\mathcal{D}(p_0 \parallel p_1) &= \frac{\alpha}{2}\ln\left(\frac{t_1}{t_0}\right) - \frac{t_0 - t_1}{2}\mathrm{SNR} + (t_0 - \sqrt{t_0 t_1})\mathrm{SNR} + o(\mathrm{SNR}) \\ &= \frac{\left(\sqrt{t_0} - \sqrt{t_1}\right)^2}{2}\mathrm{SNR} + o(\mathrm{SNR})\end{aligned} \qquad (\mathrm{SNR} \to \infty)$$

and the desired result (3.38b) follows. □

Proof of corollary 3.4. Recall that,

$$\rho_{\mathrm{y}} = \mathcal{D}(q^* \parallel p_0) \triangleq \min_{q \in Q} \mathcal{D}(q \parallel p_0)$$

where,

$$Q = \{q \mid \mathcal{D}(q \parallel p_0) = \mathcal{D}(q \parallel p_1)\}$$

It is known that q^* must be of the form,

$$q^* = p_0^{\lambda} p_1^{1-\lambda} \tag{3.44}$$

for some $\lambda \in [0, 1]$. The converse is also true – any distribution of this form in Q is q^*. Inspection of (3.38a) shows that,

$$q \sim \chi_P^2 \left(s^2 = \frac{s_0^2 + s_1^2}{2} = \frac{t_0 + t_1}{2} \mathrm{SNR} \right) \tag{3.45}$$

yields $\mathcal{D}(q \parallel p_0) \doteq \mathcal{D}(q \parallel p_1)$ for $\mathrm{SNR} \to 0$. Similarly,

$$q \sim \chi_P^2 \left(\sqrt{s^2} = \frac{\sqrt{s_0^2} + \sqrt{s_1^2}}{2} = \frac{\sqrt{t_0} + \sqrt{t_1}}{2} \sqrt{\mathrm{SNR}} \right) \tag{3.46}$$

is equidistant from p_0 and p_1 when $\mathrm{SNR} \to \infty$. All that remains is to check if these equidistant q distributions have the form in (3.44). We now sketch the essential ideas to demonstrate that they do. The series expansion (3.40) yields,

$$I_\alpha \left(\sqrt{ys^2} \right) \propto \left(ys^2 \right)^{\alpha/2} \qquad (s^2 \to 0)$$

Substituting this in the distribution (3.37) gives,

$$p_{\mathsf{y}}(y) \propto y^{\alpha/2} e^{-\frac{1}{2}(y+s^2)}$$

Thus, for $\mathrm{SNR} \to 0$,

$$p_i(y) \propto y^{\alpha/2} e^{-\frac{1}{2}(y+t_i \mathrm{SNR})}$$

and, for the choice of q in (3.45),

$$q(y) \propto y^{\alpha/2} e^{-\frac{1}{2}\left(y + \frac{t_0+t_1}{2} \mathrm{SNR}\right)}$$

It follows that $q \doteq p_0^{1/2} p_1^{1/2}$. Also, $\lambda(\mathrm{SNR}; t_0, t_1) \doteq 1/2$ $(\mathrm{SNR} \to 0)$, regardless of t_0, t_1. A more complete proof of these results keeps a tally of the remainder terms, and is omitted here.

For the large SNR case, (3.42) gives,

$$I_\alpha\left(\sqrt{ys^2}\right) \doteq \frac{e^{\sqrt{ys^2}}}{\sqrt{2\pi\sqrt{ys^2}}} \qquad (s^2 \to \infty)$$

and substitution in (3.37) gives,

$$\begin{aligned} p_{\mathsf{y}}(y) &\propto O\Big(\big(ys^2\big)^{c_0}\Big) e^{-\frac{1}{2}\left(\sqrt{y}-\sqrt{s^2}\right)^2} && (c_0 \text{ constant}) \\ &\propto e^{-\frac{1}{2}\left(\sqrt{y}-\sqrt{s^2}\right)^2\left(1+o(1)\right)} && (s^2, y \to \infty) \end{aligned}$$

Note that, as $s^2 \to \infty$, almost all the probability mass lies at $y \to \infty$. Hence, the polynomial terms $O\Big(\big(ys^2\big)^{c_0}\Big)$ vanish in relation to the exponential. Thus,

$$p_i(y) \propto e^{-\frac{1}{2}\left(\sqrt{y}-\sqrt{s_i^2}\right)^2\left(1+o(1)\right)} \qquad (s^2, y \to \infty)$$

and,

$$q(y) \propto e^{-\frac{1}{2}\left(\sqrt{y}-\frac{\sqrt{s_0^2}+\sqrt{s_1^2}}{2}\right)^2\left(1+o(1)\right)} \qquad (s^2, y \to \infty)$$

for q in (3.46). Note that,

$$\lambda\frac{\left(\sqrt{y}-a_0\right)^2}{2} + (1-\lambda)\frac{\left(\sqrt{y}-a_1\right)^2}{2} = \frac{\left(\sqrt{y}-a_2\right)^2}{2}\left(1+o\left(1\right)\right) \qquad (y \to \infty)$$

holds for $\lambda \doteq 1/2$, and, $a_2 \doteq (a_0 + a_1)/2$. Hence, $q \doteq p_0^{1/2} p_1^{1/2}$. Note that for q in (3.45),

$$\rho_{\mathsf{y}} \doteq \mathcal{D}(q \parallel p_0) \doteq \mathcal{D}(q \parallel p_1) \doteq \frac{\mathcal{D}(p_0 \parallel p_1)}{4}$$

This is also true for q in (3.46), and the result follows. □

Proof of corollary 3.5. Consider the vector r.v. $\mathbf{y} = (\mathsf{y}_1, \mathsf{y}_2, \ldots, \mathsf{y}_n)$ with y_i indepen-

dent, and likelihoods,

$$\mathsf{y}_i \sim p_{i,j}(y) \triangleq p(\mathsf{y}_i \mid \theta_j),\ j \in \{0,1\}$$

The log-likelihood ratios are,

$$\ell_i(y) = \ln \frac{p_{i,0}(y)}{p_{i,1}(y)}$$

Then,

$$\rho_{\mathbf{y}} = -\ln \min_{\lambda<0} \left(\prod_i \mathrm{E}_0\, e^{\lambda \ell_i(y)} \right) \leq -\ln \prod_i \left(\min_{\lambda_i<0} \mathrm{E}_0\, e^{\lambda_i \ell_i(y)} \right) = \sum \rho_{\mathsf{y}_i}$$

with equality only if all the λ_i are identical. As shown above, $\lambda_i \doteq 1/2$, and the result follows from corollary 3.4. □

3.9.2 Bound on Phases in a Simple, Strong MSD Sequence

Theorem. *If a simple sequence of integrated length n_r is strong, then, for odd $n_\mathrm{r} \geq 11$,*

$$P \geq \left\lceil \sqrt{2(n_\mathrm{r} - 7)} \right\rceil$$

Proof. The main idea is to identify a codeword pair that has a Hamming distance which does not scale with n_r. Since a difference pair can contribute a maximum squared distance of P^2, P must be on the order of the root of n_r.

Recall from section 3.4.1 that all subsequences of a strong sequence belong to the set,

$$\boldsymbol{S}_{\mathrm{alt}} = \left\{ \boldsymbol{s}_{\mathrm{alt}}^{(r)},\ \overline{\boldsymbol{s}}_{\mathrm{alt}}^{(r)} \right\}, \quad r \in [0, n_\mathrm{r} - 1]$$

where,

$$\boldsymbol{s}_{\mathrm{alt}} = [1\,0\,1\,0\,\ldots 1\,0\,0]$$

is a sequence of length n_r, and $\overline{\boldsymbol{s}}$ denotes binary inversion. A useful corollary is,

$$\forall\ \boldsymbol{s} \in \boldsymbol{S}_{\mathrm{alt}}, \quad \boldsymbol{s} - \boldsymbol{s}^{(2)} = [-1\,1\,0\,0\,\ldots\,0]^{(r)} \tag{3.47}$$

for some shift r. (This also illustrates that members of $\boldsymbol{S}_{\text{alt}}$ have an alternate adjacent distance of just 2, which explains the poor Hamming distance between certain codeword pairs.) By definition, every *simple* sequence has the form,

$$\boldsymbol{c} = \boldsymbol{s} \cdot \boldsymbol{s} \cdots \boldsymbol{s} \cdot \tilde{\boldsymbol{s}}, \quad \boldsymbol{s} \neq \tilde{\boldsymbol{s}} \tag{3.48}$$

If it is strong, then $\boldsymbol{s}, \tilde{\boldsymbol{s}} \in \boldsymbol{S}_{\text{alt}}$. We can always invert and rotate a simple, strong sequence to derive another simple, strong sequence which has $\boldsymbol{s} = \boldsymbol{s}_{\text{alt}}$ in (3.48) above. If $\boldsymbol{s} = \boldsymbol{s}_{\text{alt}}^{(r)}$ simply rotate left by rP places to get a new $\boldsymbol{s} = \boldsymbol{s}_{\text{alt}}$. If $\boldsymbol{s} = \overline{\boldsymbol{s}}_{\text{alt}}^{(r)}$, invert the sequence after shifting. Since rotations and inversions preserve squared distances, the resulting sequence is also strong.

Now, (3.48) implies,

$$\begin{aligned}
\boldsymbol{c}^{(\ell)} &= \underbrace{\boldsymbol{s}^{(1)} \cdot \boldsymbol{s}^{(1)} \cdots \boldsymbol{s}^{(1)} \cdot \tilde{\boldsymbol{s}}^{(1)}}_{\ell} \cdot \underbrace{\boldsymbol{s} \cdot \boldsymbol{s} \cdots \boldsymbol{s}}_{P-\ell} \\
\boldsymbol{c}^{(\ell+2P)} &= \underbrace{\boldsymbol{s}^{(3)} \cdot \boldsymbol{s}^{(3)} \cdots \boldsymbol{s}^{(3)} \cdot \tilde{\boldsymbol{s}}^{(3)}}_{\ell} \cdot \underbrace{\boldsymbol{s}^{(2)} \cdot \boldsymbol{s}^{(2)} \cdots \boldsymbol{s}^{(2)}}_{P-\ell}
\end{aligned}$$

for $\ell < P$. Hence,

$$\begin{aligned}
\boldsymbol{t}_{\ell} &= (\ell - 1)\boldsymbol{s}^{(1)} + (P - \ell)\boldsymbol{s} + \tilde{\boldsymbol{s}}^{(1)} \\
\boldsymbol{t}_{\ell+2P} &= (\ell - 1)\boldsymbol{s}^{(3)} + (P - \ell)\boldsymbol{s}^{(2)} + \tilde{\boldsymbol{s}}^{(3)}
\end{aligned} \tag{3.49}$$

Consider the case of odd P, and let $\ell = (P+1)/2$. Then (3.49) gives,

$$\begin{aligned}
\boldsymbol{t}_{\ell} &= \frac{P-1}{2}\left(\boldsymbol{s} + \boldsymbol{s}^{(1)}\right) + \tilde{\boldsymbol{s}}^{(1)} \\
\boldsymbol{t}_{\ell+2P} &= \frac{P-1}{2}\left(\boldsymbol{s}^{(2)} + \boldsymbol{s}^{(3)}\right) + \tilde{\boldsymbol{s}}^{(3)}
\end{aligned}$$

and hence,

$$\boldsymbol{t}_{\ell} - \boldsymbol{t}_{\ell+2P} = \frac{P-1}{2}\left(\left(\boldsymbol{s} + \boldsymbol{s}^{(1)}\right) - \left(\boldsymbol{s}^{(2)} + \boldsymbol{s}^{(3)}\right)\right) + \left(\tilde{\boldsymbol{s}}^{(1)} - \tilde{\boldsymbol{s}}^{(3)}\right)$$

Since $\boldsymbol{s} = \boldsymbol{s}_{\text{alt}}$,

$$\left(\boldsymbol{s} + \boldsymbol{s}^{(1)}\right) - \left(\boldsymbol{s}^{(2)} + \boldsymbol{s}^{(3)}\right) = [0\,1\,\underbrace{0\,0\,\ldots\,0}_{P-3\text{ zeros}}\,-1]$$

Also $\tilde{\boldsymbol{s}}^{(1)} - \tilde{\boldsymbol{s}}^{(3)}$ may be evaluated via (3.47). Thus,

$$\boldsymbol{t}_\ell - \boldsymbol{t}_{\ell+2P} = \left(\frac{P-1}{2}\right)[0\,1\;\;0\,0\,\ldots\,0\;\,-1] + [-1\,1\,0\,0\,\ldots 0\,0]^{(r)}$$

for some shift r. It may be verified that $\|\boldsymbol{t}_\ell - \boldsymbol{t}_{\ell+2P}\|^2$ is maximized for $r = 0$ or -1.

$$\max_r \;\|\boldsymbol{t}_\ell - \boldsymbol{t}_{\ell+2P}\|^2 = \left(\frac{P+1}{2}\right)^2 + \left(\frac{P-1}{2}\right)^2 + 1 = \frac{P^2+1}{2} + 1$$

Also,

$$\max_r \;\|\boldsymbol{t}_\ell - \boldsymbol{t}_{\ell+2P}\|^2 \geq \|\boldsymbol{t}_\ell - \boldsymbol{t}_{\ell+2P}\|^2 \geq d_{\text{sq}}\left(\mathcal{T}(\boldsymbol{c})\right) = n_{\text{r}} - 1$$

Hence,

$$\frac{P^2+1}{2} + 2 \geq n_{\text{r}}, \quad P \text{ odd} \tag{3.50}$$

For even P, pick $\ell = P/2$. Then, (3.49) yields,

$$\begin{aligned}
\boldsymbol{t}_\ell &= \left(\frac{P}{2} - 1\right)\left(\boldsymbol{s} + \boldsymbol{s}^{(1)}\right) + \boldsymbol{s} + \tilde{\boldsymbol{s}}^{(1)} \\
\boldsymbol{t}_{\ell+2P} &= \left(\frac{P}{2} - 1\right)\left(\boldsymbol{s}^{(2)} + \boldsymbol{s}^{(3)}\right) + \boldsymbol{s}^{(2)} + \tilde{\boldsymbol{s}}^{(3)}
\end{aligned}$$

and,

$$\begin{aligned}
\boldsymbol{t}_\ell - \boldsymbol{t}_{\ell+2P} &= \left(\frac{P}{2} - 1\right)\left(\left(\boldsymbol{s} + \boldsymbol{s}^{(1)}\right) - \left(\boldsymbol{s}^{(2)} + \boldsymbol{s}^{(3)}\right)\right) + \left(\boldsymbol{s} - \boldsymbol{s}^{(2)}\right) + \left(\tilde{\boldsymbol{s}}^{(1)} - \tilde{\boldsymbol{s}}^{(3)}\right) \\
&= \left(\frac{P}{2} - 1\right)[0\,1\;\;0\,0\,\ldots\,0\;\,-1] + [1\,0\;\;0\,0\,\ldots\,0\;\,-1] + [-1\,1\,0\,0\,\ldots 0\,0]^{(r)} \\
&= \left[1\;\left(\frac{P}{2} - 1\right)\;\;0\,0\,\ldots\,0\;\,-\frac{P}{2}\right] + [-1\,1\,0\,0\,\ldots 0\,0]^{(r)}
\end{aligned}$$

The norm is maximized when $r = -1$,

$$\max_r \;\|\boldsymbol{t}_\ell - \boldsymbol{t}_{\ell+2P}\|^2 = 4 + \left(\frac{P}{2} - 1\right)^2 + \left(\frac{P}{2} + 1\right)^2 = \frac{P^2}{2} + 6 \geq n_{\text{r}} - 1$$

Hence,

$$\frac{P^2}{2} + 7 \geq n_{\mathrm{r}}, \quad P \text{ even}$$

which, with (3.50), completes the proof. The bounds may be tightened by considering additional codeword pairs, which preclude certain values of r during the norm maximization. □

Chapter 4

An Energy Sampling Wireless Modem

We discuss the design of a single-chip, energy sampling, wireless modem with data rates up to 16 Mbps, and a synchronizer that can sustain 16 GOPS (billion operations per second) to synchronize to within $\pm$ 1 ns using the new synchronization sequences developed in the previous chapter. The modem was recently reported in two papers dealing with the RF and analog front-end that samples and quantizes energy [17], and the digital baseband that synchronizes and demodulates [40]. The author developed bit-accurate signal processing algorithms used in the digital baseband, which were implemented on silicon by Patrick Mercier. Denis Daly implemented the energy sampler.

The modem was developed for DARPA's Hybrid Insect Microelectromechanical Systems, or HiMEMS program. The broad goal of the program is "to develop tightly coupled machine-insect interfaces by placing micro-mechanical systems inside the insects during the early stages of metamorphosis" [18]. The modem establishes a wireless link used to control the flight of the hawkmoth *Manduca sexta*. The only source of power on the moth is a harvester that converts insect motion to electrical energy. This imposes severe energy constraints on the modem. Previous work on energy sampling wireless modems suggested their suitability for this application [58, 33].

We will describe the signaling scheme, packet format, and the top level modem

structure in the next section. This will be followed by a detailed discussion of VLSI efficient classifiers (synchronizers). We will end with a description of the chip, measured results, and a summary of contributions.

4.1 Overview

4.1.1 Pulsed Ultra-Wideband Signaling

The FCC defines an Ultra-Wideband (UWB) signal as one with a -10 dB bandwidth exceeding 500 MHz, or a fractional bandwidth (bandwidth/center frequency) exceeding 1/5. The regulatory body opened up the 3.1-10.6 GHz band for UWB signaling in 1999, ushering in a period of remarkable activity in UWB signaling. To reduce the risk of interference to existing devices (including those of operators who had previously purchased spectrum in these bands), UWB signals must obey a noise emission limit of -41.3 dBm/MHz, which amounts to a transmitted power of -13.9 dBm for a 500 MHz signal. Our focus here is single carrier, *pulsed* UWB systems, i.e., those that use amplitudes $\{0, \pm 1\}$ to modulate the UWB pulse. Although our receivers are non-coherent, two phases are still useful. Phase scrambling at the transmitter is a convenient means to disrupt the periodicity of repeated sequences. Left unchecked, such periodicity creates spectral lines that violate regulatory requirements.

4.1.2 Packet Format

Our UWB modem employs a packet format shown in figure 4-1, which is similar to that of the 802.15.4a standard [29].

The packet begins with a preamble which is used to detect the signal and synchronize to it. The preamble is composed of repeated sequences (S_0), followed by a start-frame-delimiter (SFD) composed of K sequences, $U_0, U_1, \ldots, U_{K-1}$, each equal in length to S_0. To keep the silicon cost manageable, our system constrains the sample length, n_r, of S_0 to be 32 or smaller. We also permit at most 74 repetitions of S_0. This includes 10 repetitions where the modem is busy processing and does not sample

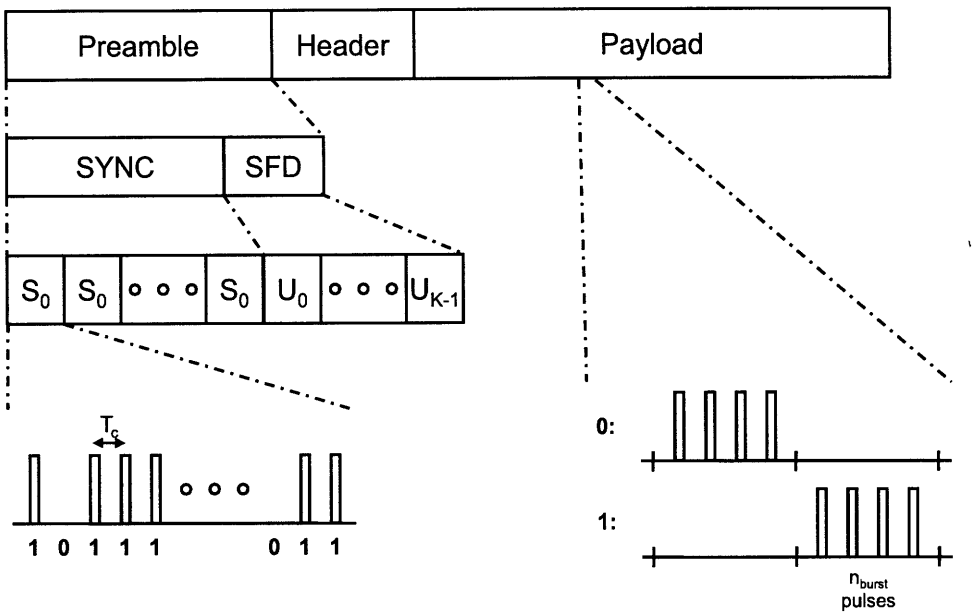

Figure 4-1: Packet format used in energy sampling modem. Pulse phase scrambling and modulation are not shown.

the incoming signal (more on this in a later section). The remaining 64 repetitions allow reliable detection and synchronization at the desired SNR. The SFD sequences $U_i \in \{S_0, S_1 = \mathbf{0}\}$ where $\mathbf{0}$ is the all-zero sequence. Better choices for S_1 exist, but this suffices to achieve the desired error rates, and, simplifies implementation. We support two SFDs. The first is the trivial, $U_0 = \mathbf{0}$, suitable for high SNRs. The other is 5 sequences long, $\mathbf{000}S_0\mathbf{0}$, which, incidentally may be shown to have the same performance as the 8 sequence SFD used in 802.15.4a ($\mathbf{0}S_0\mathbf{0}S_0S_0\mathbf{00}S_0$). The S_0 and U_i sequences use on-off keying (OOK). The pulses are rectangular and 1.95 ns wide, corresponding to a -10 dB bandwidth of 550 MHz. They are separated by a *chip period* (T_c) which is also equal to 1.95 ns in our system. Hence, the pulses are back to back. The transmitter scrambles pulse phase via a PN-sequence to avoid spectral lines.

The header is a 8 bit field that conveys the length of the packet in bytes, and the payload carries the information bits. Both the header and payload use binary pulse position modulation (PPM). The PPM symbol has two slots, and a bit is represented by a set of n_{burst} pulses in the corresponding slot. The system allows the slot time, and n_{burst} to be programmable.

4.1.3 Modem Units

A block diagram of the modem is shown in figure 4-2.

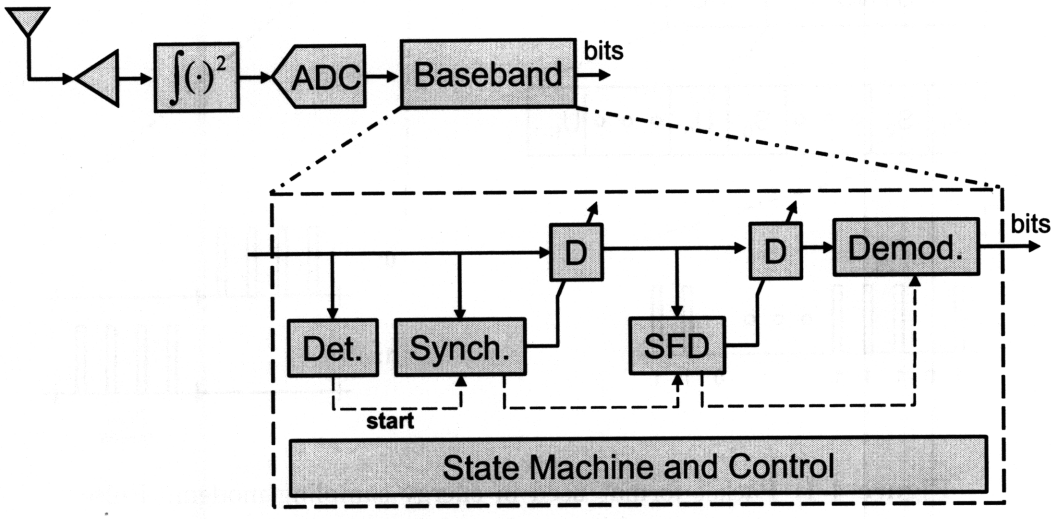

Figure 4-2: Block diagram of the energy sampling modem.

The RF energy sampler uses an integration period of $P = 16$, and a 5 bit analog-to-digital converter (ADC) to quantize energy [17]. The digital baseband has four signal processing units — detector, sample synchronizer, sequence synchronizer (SFD search unit), and demodulator — coordinated by a control unit. The detector is programmed with a threshold that determines the probability of detection and false alarm. The sample synchronizer infers the boundary of the repeated sequences, while the sequence synchronizer infers the start of the payload by searching for the SFD. The demodulator decodes the header to determine packet length, and then demodulates PPM symbols by comparing the energy in the two slots.

Detection and SFD search are classification problems, and use the same hardware as the sample synchronizer. For detection, we require that the largest of the codeword log-likelihoods (LLs) exceed the likelihood of the noise codeword by a threshold. Detection performance may be traded off for lower energy consumption by sub-sampling the codebook.

Searching for the SFD, $U_0 U_1 \ldots U_{K-1}$, corresponds to a classification problem

with the codebook,

$$\begin{bmatrix} S_0 & S_0 & \dots & S_0 & S_0 \\ S_0 & S_0 & \dots & S_0 & U_0 \\ S_0 & S_0 & \dots & U_0 & U_1 \\ & \dots\dots\dots\dots & & & \\ S_0 & U_0 & \dots & U_{K-3} & U_{K-2} \\ U_0 & U_1 & \dots & U_{K-2} & U_{K-1} \end{bmatrix} \tag{4.1}$$

Recall that we restrict U_i to be either S_0, or, $S_1 = \mathbf{0}$. Hence, the LL of each codebook entry may be calculated by adding K LLs of the form, $\log p(\boldsymbol{y} \mid S_i)$. This requires an integrated codebook with just S_0 and S_1.

In summary, a well implemented classifier is the key to the overall efficiency of preamble processing. This is the focus of the next section.

4.2 VLSI Efficient Classifiers

A classifier takes the received vector $\boldsymbol{y}$, and an integrated codebook $\mathcal{T}$ as inputs, computes codeword log-likelihoods (LLs) or suitable approximations, and declares the most likely codeword. Designing a good classifier is an exercise in balancing performance (probability of classification error) and complexity (area, power, etc.) In this section, we study a classifier that strikes a suitable balance when P is large, as in our case.

4.2.1 Suitable Approximations

The maximum-likelihood (ML) classifier is optimum, but costly to compute because of the form of the chi-square distribution. A ML implementation that uses lookups and limits computation to adding sample LLs is possible, though the resulting table sizes are much larger than those required by approximations with comparable performance.

Matched filter (MF) classifiers, i.e. those that pick the codeword with the maximum inner product, $\langle \boldsymbol{y}, \boldsymbol{t} \rangle$, work well at low SNRs, with a performance close to that of ML classifiers. These are suitable when all codewords have the same norm. This

is the case for binary integrated codebooks, e.g., O-H and w-sequences. The loss in MF performance compared with ML increases with SNR.

When codeword norm is not a constant, the minimum distance (MD) classifier that computes $\|\boldsymbol{y} - \mathrm{E}[\mathbf{y} \mid \boldsymbol{t}]\|$ may be used. On substituting the expected value, $\mathrm{E}[\mathbf{y} \mid \boldsymbol{t}] = P + \boldsymbol{t}\mathrm{SNR}$, and using the property that, $\sum_j t_{i,j} = w(\boldsymbol{c})$, i.e. codewords always have the same L_1 norm, it follows that the MD metric is equivalent to $\langle \boldsymbol{y}, \boldsymbol{t} \rangle - \|\boldsymbol{t}\|^2 \mathrm{SNR}/2$. While the MD classifier achieves good performance at low SNRs, it may be extremely sensitive to errors in estimating the SNR[1]. For instance, when using I-H sequences with P=16, a SNR estimation error of about 0.5 dB is enough to render the classifier unusable. Furthermore, this sensitivity is independent of the SNR. Sacrificing link margin does not help. There are two possible solutions. One is to use linear/affine hyperplanes different from those defined by the MD criterion. It may be shown that there exist simple linear hyperplanes that can discriminate between two codewords without regard to SNR (at some cost to performance). The main drawback is that codewords must now be compared pairwise — the most likely codeword cannot be inferred by computing a single metric per codeword, and picking the maximum. The alternative is to use quadratic metrics. We now discuss this option.

When P is large, the central limit theorem motivates treating chi-squared r.v.s as Gaussian,

$$\mathrm{y}_k \;\approx\; \mathcal{N}\left(\mathrm{E}[\mathrm{y}_k \mid t_k], \mathrm{var}[\mathrm{y}_k \mid t_k]\right) \tag{4.2}$$

where $\mathbf{y}$ is, as usual, the received random vector, and $\boldsymbol{t}$ is the transmitted codeword. Thus the LLs may be approximated by,

$$\log p(\boldsymbol{y} \mid \boldsymbol{t}) \approx -\sum_{k=0}^{n_\mathrm{r}-1} \frac{(y_k - \mathrm{E}[\mathrm{y}_k \mid t_k])^2}{2\mathrm{var}[\mathrm{y}_k \mid t_k]} - \frac{1}{2}\sum_{k=0}^{n_\mathrm{r}-1} \log(2\pi \mathrm{var}[\mathrm{y}_k \mid t_k]) \tag{4.3}$$

(The bias term may be ignored with little loss of performance at low error rates.) MD classifiers assume that the signal is Gaussian with equal variance along all dimensions.

[1]It may be shown that a MD classifier is sensitive if and only if there exists a pair of codewords, say, $\boldsymbol{t}$ and $\boldsymbol{t}'$, which lie on the same side of the *linear* hyperplane with the normal $\boldsymbol{t} - \boldsymbol{t}'$. This is only possible if $\boldsymbol{t}$ and $\boldsymbol{t}'$ have different norms, but this is not a sufficient condition. For instance, it may be shown that *no* integrated codebook with $P = 2$ has such a pair, despite unequal norms.

The approximation above accounts for the unequal variances. For lack of a better term, we call this the *unequal variance Gaussian* (UVG) metric. For $P = 16$, this approximation essentially matches the performance of a ML classifier. This is true across SNRs, unlike a MD classifier whose loss grows with SNR. Figure 4-3 illustrates this for a $(n_r = 31, P = 16)$ I-H sequence. Two sets of curves are shown. The first is when the sequence is not repeated, and the other with a repetition of 32x to allow operation at lower SNR. The loss in performance of the UVG classifier compared with the optimum, ML, classifier is about 0.3 dB in the low SNR case (32x) and essentially zero at high SNRs (1x). The MD classifier has a loss of about 0.6 dB (32x) and 0.8 dB (1x). Note that losses are sequence dependent, and these are only illustrations.

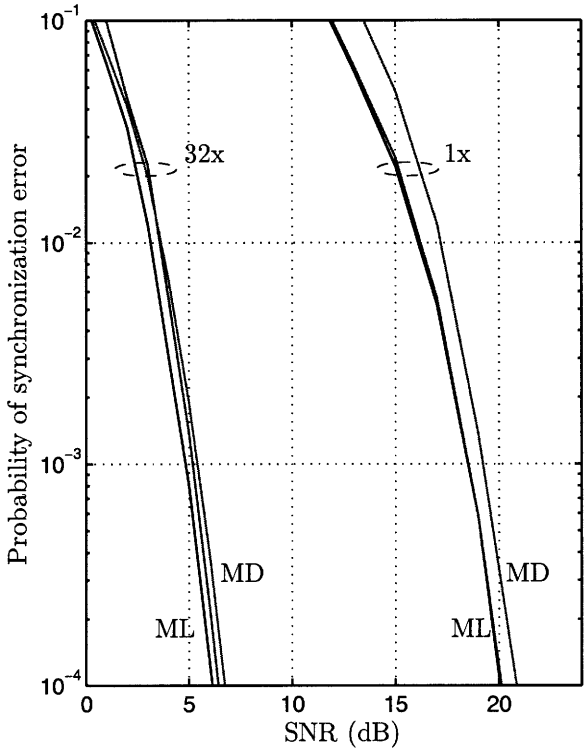

Figure 4-3: The ML, UVG, and MD classifiers for a (31,16) I-H sequence repeated 32x and 1x. The UVG curve lies between ML and MD (32x), and overlaps with ML (1x).

The UVG approximation is preferable to MD since it is more robust to SNR mismatches. Furthermore, this robustness improves with increasing SNR — a desirable system characteristic. Figure 4-4 shows the impact of SNR mismatch for the same I-H sequence. Once again, there are two sets of curves corresponding to 1x and 32x

repetition. Dotted curves show performance of the MD and UVG classifier with no mismatch (i.e. identical to those in figure 4-3). To study the impact of mismatch, we fix the SNR estimate used by the classifier, and sweep the actual SNR seen by the channel. The MD classifier fails catastrophically for differences between true and estimated SNRs that are as small as 0.5 dB. The UVG classifier is seen to be more tolerant at low SNRs (32x), and indifferent to mismatches at high SNRs (1x). All these properties may be rigorously derived by considering the codebook geometry. For instance, in the case of I-H sequences, it may be shown that the MD classifier can only tolerate a linear SNR mismatch factor of about $(P-1)/(P-2)$, for $P > 2$, before catastrophic failure. Hence, the MD classifier's tolerance decreases from about 3 dB ($P = 3$) to 0.3 dB ($P = 16$) when I-H sequences are used.

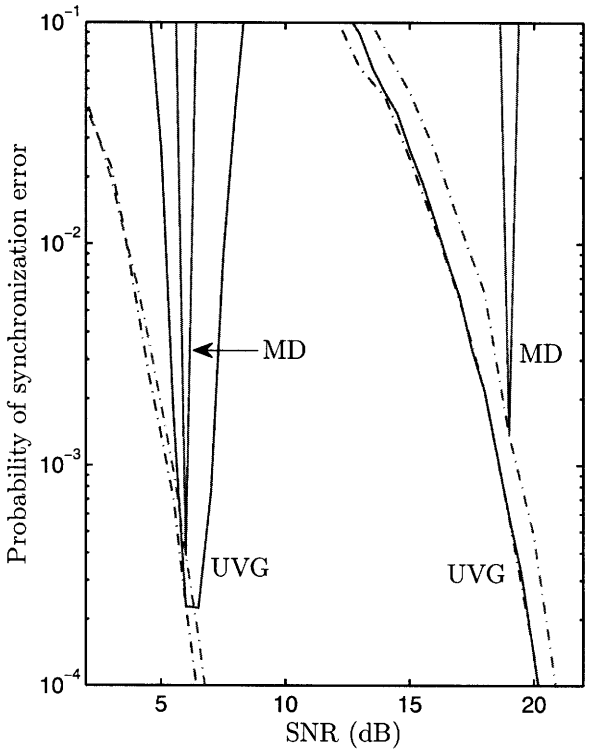

Figure 4-4: Impact of SNR mismatch on the MD and UVG classifiers. Dotted curves denote no mismatch. Curves on the left and right are for 32 and 1 sequence repetition, respectively.

Finally, we will show in the following section that the UVG classifier has essentially the same computational complexity as the MD classifier. This, along with

the robustness, and good performance across SNRs is a strong argument in favor of UVG— when P is large. Note that chi-square r.v.s decay with a linear, rather than a quadratic exponent. Hence, the UVG approximation is mismatched in the tail. This has not been a problem in our system, perhaps because error rates are not very low (on the order of 10^{-4}). Lower error rates, or a different geometry might change this, and designers should be vigilant.

In the next two sections, we will study schemes that reduce the computational burden of the UVG classifier.

4.2.2 Reducing Computation

Following (4.3), we define the decision metric,

$$\beta(\boldsymbol{y}, \boldsymbol{t}) = \sum_{k=0}^{n_r-1} \frac{(y_k - \mu_k)^2}{\sigma_k^2} \tag{4.4}$$

where, $\mu_k = \mathrm{E}[\mathrm{y}_k \,|\, t_k]$, and $\sigma_k^2 = \mathrm{var}[\mathrm{y}_k \,|\, t_k]$. Calculating the metric above takes 2 multiplications and 1 addition *per coordinate*, which we will denote as (2,1). Hence, the computation required for all codewords is $n(2,1)$, where n is the symbol length of the sequence. If R codewords are observed to achieve lower operating SNR, we end up with $Rn(2,1)$ operations (*ops*) per coordinate. Averaging received data and running the classifier just once is significantly cheaper than running the classifier R times and averaging the LLs. To see how this may be done, re-write (4.4) as,

$$\begin{aligned} \beta(\boldsymbol{y}, \boldsymbol{t}) &= \sum_{k=0}^{n_r-1} \left(y_k^2 a_{2,k} + y_k a_{1,k} \right) + a_0(\boldsymbol{t}) \\ &= \langle\, \boldsymbol{y}^2, \boldsymbol{a}_2(\boldsymbol{t}) \,\rangle + \langle\, \boldsymbol{y}, \boldsymbol{a}_1(\boldsymbol{t}) \,\rangle + a_0(\boldsymbol{t}) \end{aligned} \tag{4.5}$$

where, $\boldsymbol{a}_2, \boldsymbol{a}_1$, and a_0 are determined by $\boldsymbol{t}$ and SNR (the latter dependence not shown for brevity). Consider now R repetitions of this sequence. Let $\boldsymbol{y} = [\boldsymbol{y_0}\ \boldsymbol{y_1}\ \ldots\ \boldsymbol{y_{R-1}}]$

where $\boldsymbol{y_r}$ is the received vector for the r-th transmitted sequence. Then,

$$\begin{aligned}\frac{1}{R}\beta(\boldsymbol{y},\boldsymbol{t}) &= \frac{1}{R}\sum_{r=0}^{R-1}\beta(\boldsymbol{y_r},\boldsymbol{t}) \\ &= \left\langle \frac{\sum \boldsymbol{y}_i^2}{R}, \boldsymbol{a}_2(\boldsymbol{t}) \right\rangle + \left\langle \frac{\sum \boldsymbol{y}_i}{R}, \boldsymbol{a}_1(\boldsymbol{t}) \right\rangle + a_0(\boldsymbol{t}) \\ &\triangleq \left\langle \overline{\boldsymbol{y}^2}, \boldsymbol{a}_2(\boldsymbol{t}) \right\rangle + \langle\, \overline{\boldsymbol{y}}, \boldsymbol{a}_1(\boldsymbol{t}) \,\rangle + a_0(\boldsymbol{t})\end{aligned}$$

It takes $R-1$ additions per coordinate to compute $\overline{\boldsymbol{y}}$ (the division by R is usually realized as an arithmetic shift by picking R to be a power of two). It takes R multiplies and $R-1$ adds per coordinate to compute $\overline{\boldsymbol{y}^2}$. Hence, the total for both averages is about $R(1,2)$. Having computed the averages, it takes $n(2,1)$ ops to compute all the metrics. Hence, the overall complexity is $R(1,2)+n(2,1) = (R+2n, 2R+n)$ ops per coordinate. This represents a reduction in multiplications by a factor of $2Rn/(R+2n)$, which is $\geq \min(n, R/2)$. Additions are similarly reduced by a factor $\geq \min(n/4, R/2)$. If $n \gg R$, both operations are reduced by a factor close to R. Put differently, the averaging cost is insignificant in this case, and the overall cost is essentially identical to that of a system that does not repeat sequences.

A further reduction is possible as follows[2]. It may be shown that,

$$\boldsymbol{a}_1(\boldsymbol{t}) = -\frac{1}{2} - P\boldsymbol{a}_2(\boldsymbol{t}) \tag{4.6}$$

for any $\boldsymbol{t}$. Hence,

$$\frac{1}{R}\beta(\boldsymbol{y},\boldsymbol{t}) = \left\langle \overline{\boldsymbol{y}^2} - P\overline{\boldsymbol{y}}, \boldsymbol{a}_2(\boldsymbol{t}) \right\rangle - \frac{1}{2}\sum_j \overline{y_j} + a_0(\boldsymbol{t})$$

Since the second term is independent of $\boldsymbol{t}$, this allows the following equivalent metric to be used,

$$\beta'(\boldsymbol{y},\boldsymbol{t}) = \left\langle \overline{\boldsymbol{y}^2} - P\overline{\boldsymbol{y}}, \boldsymbol{a}_2(\boldsymbol{t}) \right\rangle + a_0(\boldsymbol{t}) \tag{4.7}$$

It takes (1,1) ops per coordinate to compute $\overline{\boldsymbol{y}^2} - P\overline{\boldsymbol{y}}$ once the averages have been

[2]This transform was discovered after the chip was built, and could not be incorporated.

computed. It then takes $n(1,1)$ ops to compute the inner product in (4.7) for all codewords — the same complexity as that of a MF. This transformation cuts the number of multiplications in half compared to realizing the quadratic expression (4.5). The final count is $R(1,2)+(1,1)+n(1,1) \approx (n+R, n+2R)$ ops per coordinate, as opposed to a direct implementation that uses $Rn(2,1)$ ops.

Implementing the β metric efficiently requires pre-computing and storing coefficients. Note that $a_{2,k}$ and $a_{1,k}$ are determined by t_k, which is an integer in $[0, P]$. Hence, we may store the integrated codebook, and a table of $P+1$ coefficients for each of a_2 and a_1, respectively. Also, $a_0\left(\boldsymbol{t}^{(j)}\right) = a_0(\boldsymbol{t})$, and hence only P a_0 coefficients need to be stored. The codebook is completely characterized by the first P codewords. Hence, storing the codebook requires $P\,n_{\mathrm{r}}\,\lceil \log_2(P+1) \rceil$ bits. For example, if $P = 16, n = 512, b_{\text{coeff}} = 12$, the codebook requires 2480 bits and coefficient tables require 600 bits. Since coefficients are SNR dependent, a typical implementation must store several such tables, depending on the total dynamic range, and the system's tolerance to SNR estimation error. For instance, a system with a range of 30 dB which can tolerate ± 1.5 dB of SNR error would require 10 tables.

4.2.3 Distributing Computation

The building block of a UVG classifier carries out the computation in (4.5) (or (4.7)). We call this the LL unit (LLU). Every hardware clock cycle, the LLU accepts a received sample y_k, and the corresponding table entries, and computes the sample LL. A codeword LL is produced every n_{r} cycles.

The first architectural decision is the number of LLUs required, which determines the latency of the classifier. Higher latencies translate to larger temporary buffers to store the incoming signal while the classifier is busy. An alternative is to simply discard the signal (permissible only in the preamble), and hence sacrifice degrees of freedom, while the unit is busy. Also, a lower latency, or increased parallelism allows reducing circuit voltages, which often enables significant reductions in energy consumed [50]. The latency to compute LLs for the entire codebook is $n\,n_{\mathrm{r}}/n_{\mathrm{LLU}}$ clock cycles, where n_{LLU} is the number of available LLUs. The front-end produces an

energy sample every clock cycle, thus sampling a complete sequence every n_{r} cycles. Hence, the latency in terms of sequence durations is n/n_{LLU}. An implementation that can sacrifice some part of the preamble to manage latency typically limits the loss to, say, n_{Loss} sequences. Hence, $n_{\mathrm{LLU}} \geq n/n_{\mathrm{Loss}}$. Our implementation uses $n = 512$ and achieves $n_{\mathrm{Loss}} = 4$ using 128 LLUs.

The next question is a suitable organization of these units. For instance, every unit may store a local copy of the coefficients. This reduces the interconnect costs to conveying the received vector (or averages), but significantly increases the storage costs. Consider an alternative where we group units into P *phase blocks.* Units in a phase block share a common set of coefficients. LLUs in the ℓ^{th} group compute $\beta\left(\boldsymbol{y}, \boldsymbol{t}_\ell^{(j)}\right)$. These are identical to the metrics $\beta\left(\boldsymbol{y}^{(-j)}, \boldsymbol{t}_\ell\right)$. However, the VLSI cost of the two expressions is typically not equivalent. In our implementation using the β metric, the choice was between rotating two sets of 12 bit wide coefficients, or $\overline{\boldsymbol{y}^2}$ and $\overline{\boldsymbol{y}}$, which are 9 bits each. The latter is more efficient, both in energy and storage.

Further optimizations may be possible. For instance, we do not currently share computation across codewords. A good trellis representation, if one exists for our codebooks, would permit that. Another possibility is to relay pre-computed products, rather than the raw data and coefficient vectors. For instance, every clock cycle, a central unit may compute $P+1$ products of the form $\beta(y_k, t)$ and relay these to LLUs. The LLUs then select the desired product based on their codeword and accumulate it. Such a scheme requires only $P+1$, rather than $2n$ multiplications per coordinate — a savings of close to n_{r}. However, this should be weighed against the increase in interconnect and multiplexing costs by a factor of $P+1$.

4.3 Chip Details

The author implemented a bit-accurate digital baseband in Matlab and C. A floating point ML classifier was implemented to serve as a performance benchmark. A fixed point UVG classifier was then developed to enable an efficient VLSI implementation, while limiting the performance loss compared with the benchmark. Patrick Mercier

ported the fixed point representation in Matlab to Verilog, a hardware description language, and established the bit-level equivalence of the two versions. He also carried out the physical design of the digital baseband, and interfaced it to the energy sampler developed by Denis Daly on the same chip. The chip was fabricated in a 90 nm CMOS process, and the digital baseband occupies 2.55 mm^2. More details are included in [40, 17]. A die photograph is shown in figure 4-5.

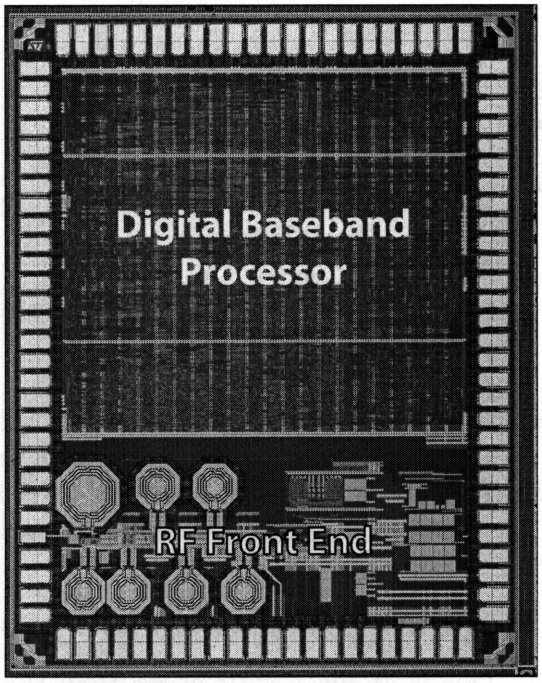

Figure 4-5: A single-chip energy sampling modem.

4.4 Measurements

The equivalence of the digital baseband on the chip to the fixed-point Matlab implementation was established via a mode that allows feeding known inputs directly to the baseband (i.e. bypassing the RF energy sampler). The outputs were then shown to match Matlab bit for bit. Figure 4-6 shows the overall performance of the digital baseband when using length 31 I-H sequences repeated 16 times. It may be seen that the overall implementation loss of the silicon implementation compared with an unquantized, ML classifier is less than 2 dB.

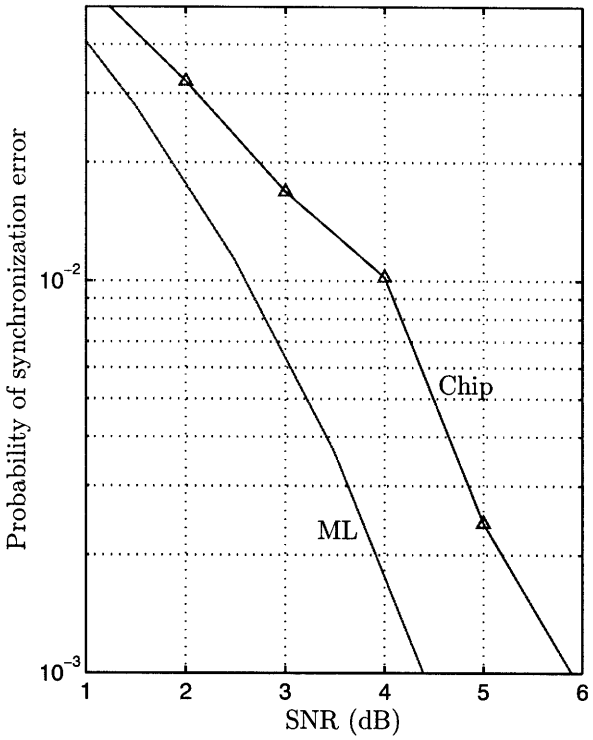

Figure 4-6: Synchronization error rate measured on the chip (courtesy Patrick Mercier), compared with an unquantized ML implementation (simulated). SNR is the ratio of average power to noise, and not SNR per symbol.

The sensitivity of the overall modem was -76 dBm for a data rate of 16 Mbps at a BER of 10^{-3} [17]. This is about 1.5 dB worse than the theoretically predicted value (assuming an unquantized front-end). The discrepancy is probably a result of, among other factors, unaccounted off-chip losses, imperfect synchronization, and measurement resolution.

4.5 Summary

In this chapter we discussed the design of an energy sampling wireless modem capable of accurate synchronization using our new sequences. We showed that the minimum distance classification criterion may suffer from extreme sensitivity to SNR estimation errors. A Gaussian approximation that takes the unequal variance along signal dimensions into account overcomes this problem, without increasing the computa-

tional complexity. This is the first UWB solution to achieve high synchronization precision without changing the sampling phase of the receiver via a high speed clock. We hope that this will dispel the widely held notion that such phase adjustment is a requirement for accurate time of arrival estimation, and lead to more efficient solutions.

Chapter 5

Conclusions

We begin with a list of open problems, and conclude with the outlook for this work.

5.1 Open Problems

5.1.1 Coding

Problem 5.1. Good codes for the energy sampling (ES) channel.

Description: A corollary of theorem 3.1 is that coding gain over the ES channel is given by,

$$\gamma_c \to \begin{cases} d(k/n)^2 & (\text{SNR} \to 0) \\ d(k/n) & (\text{SNR} \to \infty) \end{cases}$$

As an example, in a coherent system, the rate $1/2$, $K = 7$ convolutional code with octal generators [133 171] yields 5.7 dB gain at a BER of 10^{-6}. Using approximations for the probability of bit error (and not asymptotic arguments), we estimate that the gain over a complex ES channel with $P = 16$ is just 2.5 dB. Using higher rate codes should improve gain, but will require longer blocklengths. Good solutions would allow improving 802.15.4a coding performance.

Problem 5.2. Closing the gap to capacity over the observation limited (OL) channel.

Description: We are currently around 2 dB from capacity when sampling the (24,12) Golay code using the maximum entropy (ME) criterion. What are the best

codes and sampling techniques to close this gap? According to the sphere packing bound, a perfect rate 1/6 spherical code with $k = 500$ would be about a dB from capacity (at a codeword error rate of around 10^{-3}). This suggests, based on our work, that $k \approx 70$ should suffice over the OL channel. What are good candidates? A rate-compatible punctured convolutional code (RCPC) where the mother code has a gain of, say, 6.5–7 dB might be an option [27]. Implementing the BCJR algorithm efficiently, as incremental channel observations are made, would be an interesting exercise. See section 2.7 for a discussion of short Turbo and LDPC codes of Crozier *et al.* and Tong *et al.*, which might also be viable.

Problem 5.3. (Conjecture) Let $A(n, d, w)$ be the cardinality of the largest set of constant weight binary strings of length n, minimum pairwise distance d, and weight w. Then,

$$A(5t, 2t, 2t) \geq 10t \qquad t \in \mathbb{N}$$

Description: This is a special case of problem 5.10 for $P = 2$, since integrated codebooks of block w-sequences are constant weight binary codebooks. We have verified the conjecture for $t \leq 8$. The codes for $t = 7, 8$ are the only ones known for the corresponding n, d, w parameters. They have a coding gain of about 4 dB, which is reasonable given the ease of decoding (their codebooks are the union of two circulant matrices.) See Sloane's website, http://www.research.att.com/~njas/codes/Andw/ for the best known constant weight codes, and also [9].

5.1.2 Information Theory

Problem 5.4. Capacity of the energy sampling (ES) channel.

Description: What is the capacity of the channel in figure 3-1? Energy sampling is now part of the 802.15.4a standard, and to the best of our knowledge, this is an open problem. Our work on error exponents as the SNR $\rightarrow 0, \infty$ may help (theorem 3.1).

Problem 5.5. Sphere packing bound for the observation limited (OL) channel.

Description: Given a rate, Shannon bounded the information blocklength (k) required to achieve a certain reliability over the AWGN channel by essentially calcu-

lating the probability that a n-dimensional Gaussian r.v. falls outside a hypercone with solid angle 2^{-k} [19]. E.g., for a gain of 9.2 dB, rate 1/6, and codeword error rate of around 10^{-4}, this evaluates to $k \geq 81$. A k of 12 is possible on the OL channel by sampling the Golay code using the ME criterion. What is the analogous sphere packing bound for the OL channel?

Problem 5.6. (Conjecture) If, for two codeword pairs $(\boldsymbol{c}, \boldsymbol{c}')$ and $(\boldsymbol{b}, \boldsymbol{b}')$,

$$d(\boldsymbol{t}(\boldsymbol{c}), \boldsymbol{t}(\boldsymbol{c}')) > d(\boldsymbol{t}(\boldsymbol{b}), \boldsymbol{t}(\boldsymbol{b}'))$$

for both $d = d_{\mathrm{sq}}$, and $d = d_{\mathrm{rt}}$, then,

$$\rho(\boldsymbol{c}, \boldsymbol{c}'; \mathrm{SNR}) > \rho(\boldsymbol{b}, \boldsymbol{b}'; \mathrm{SNR})$$

for all SNR.

Description: See section 3.1 for notation. If true, this would be a major step toward finding sequences which maximize the exponent for a given SNR, rather than vanishing or infinite SNRs. It would imply that such sequences always lie on the $(d_{\mathrm{sq}}, d_{\mathrm{rt}})$ Pareto frontier (see problem 5.12).

5.1.3 Sequence Design

Problem 5.7. Given the sample length of a sequence, n_{r} (odd), find the smallest integration period (P) for which a strong sequence exists, i.e., $d_{\mathrm{H}}(\mathcal{T}(\boldsymbol{c})) = n_{\mathrm{r}} - 1$ (see section 3.1 for notation.)

Description: We have shown that for a strong sequence to be simple,

$$P \geq \left\lceil \sqrt{2(n_{\mathrm{r}} - 7)} \right\rceil, \quad n_{\mathrm{r}} \geq 11$$

What is corresponding lower bound if we allow sequences that are not simple? Table 3.1 tabulates strong sequences that are not simple.

Problem 5.8. (Conjecture) There always exists a sequence $\boldsymbol{c}$ such that,

$$d_{\text{sq}}\left(\mathcal{T}(\boldsymbol{c})\right) \geq \left\lceil \frac{3n_{\text{r}}}{4} \right\rceil$$

Description: Truth would imply that it is always possible to improve the squared distance of an interpolated Hadamard sequence by a factor of about 1.5, i.e. roughly 1.8 dB. The conjecture is suggested by table 3.1.

Problem 5.9. (Conjecture 3.3) For all odd $n_{\text{r}} \geq 7$, and $P \geq P_{\text{lim}}(n_{\text{r}})$, there exists a simple, strong sequence of the form,

$$\boldsymbol{c} = \boldsymbol{s}_{\text{alt}} \cdot \boldsymbol{s}_{\text{alt}} \cdots \boldsymbol{s}_{\text{alt}} \cdot \boldsymbol{s}_{\text{alt}}^{(r(n_{\text{r}}))} \tag{5.1}$$

where P_{lim} and r are functions of n_{r}.

Description: See section 3.4.1 for details.

Problem 5.10. (Conjecture 3.4) There exists a block w-sequence of length $n_{\text{r}} = t(P+3)$ for any given t and P, which achieves a distance of $2t$.

Description: If true, it proves that we can always achieve a root-distance gain of 4x compared with oversampled Hadamard sequences (802.15.4a-like codes). It also yields a new family of constant weight codes, and establishes the lower bound,

$$A(t(P+3), 2t, Pt) \geq tP(P+3) \qquad t \in \mathbb{N}$$

where $A(n, d, w)$ is the cardinality of the largest set of constant weight binary strings of length n, minimum pairwise distance d, and weight w.

Problem 5.11. What sequences maximize the squared or root distance in the presence of multipath?

Description: Studying sequence performance under multipath is critical for use in real world scenarios. The high SNR case (root distance criterion) might be easier to tackle first because sequences that are sparse are optimum in the AWGN case, and this will minimize intersymbol interference (ISI).

Problem 5.12. Are there families other than strong sequences that are Pareto optimum with respect to the $(d_{\text{sq}}, d_{\text{rt}})$ criteria (figure 5-1)?

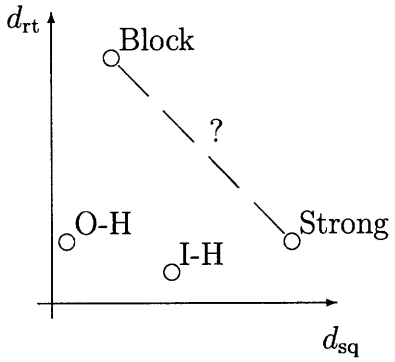

Figure 5-1: What does the Pareto frontier look like?

Description: A sequence is Pareto optimum with respect to the $(d_{\text{sq}}, d_{\text{rt}})$ criteria if no other sequence can simultaneously improve both these distances. The set of all Pareto optimum sequences forms the Pareto frontier. If the conjecture in problem 5.6 is true, it would establish the fundamental importance of the frontier in constructing sequences that are optimum for a given SNR: such sequences will lie on the frontier (the converse is not necessarily true.) Strong sequences maximize d_{sq} and hence lie on the frontier. Block sequences likely lie close to the frontier (at least for large P). Little else is known about the frontier. One quick contribution would be to plot the frontier for small n_{r} and P values, using computer searches, to get some qualitative insight into the more general problem.

5.1.4 VLSI

Problem 5.13. A digital baseband that achieves 9x reduction in observation via the (24,12,8) Golay code.

Description: The baseband would set a record for the shortest blocklength for this gain, by an order of magnitude, compared with any reported communications system. The challenge is an efficient VLSI architecture for computing bit APPs on a tail-biting trellis of the Golay code. Key references are [10, 36].

Problem 5.14. Demonstrate a 10x reduction in VLSI costs of the energy sampling synchronizer reported by Mercier *et al.* [40].

Description: Requires development of new linear classifiers that are robust to SNR mismatch, and the corresponding VLSI architecture. Our work allows high precision synchronization without high speed clocks, which would enable cost sensitive 802.15.4a chipsets to offer locationing. (Such chipsets would typically not require high precision synchronization for the payload.)

5.2 Outlook

We hope that the theme of adaptive sampling — well known to statisticians — will find systematic use in modem design, beyond the examples in this thesis. The general ideas are straightforward:

1. Quit computation when the results are reliable enough.
2. Add on some extra observations for the case when the channel is bad and the metric needs extra samples to reach the reliability threshold.
3. Whenever possible, pick observations using the Box-Hill, or Blot-Meeter criterion, rather than uniformly.

These techniques are probably used in an ad-hoc manner today, but diligent use may allow trading degrees of freedom for greatly reduced complexity in battery constrained devices.

As for specific observation cost coding techniques, we feel confident that reducing observation by 4-5x is best done by using a small (7,3) code sampled uniformly or using maximum entropy. We hope that future standards will consider "battery emergency modes" where bandwidth is traded off to conserve battery charge using simple adaptive sampling. We believe that the main barrier to realizing higher gains (8-9x) is the ability to rapidly turn a sampler on/off. Currently, this seems practical only for non-coherent systems.

Our work on new synchronization sequences suggests that there are considerable gains for energy sampling receivers operating at low SNR and moderate integration periods. Should energy sampling systems continue to grow in importance — their ultra low cost is an important attraction — the best way for 802.15.4a devices to use these new sequences would be in a special, extreme energy efficiency mode. This might offer vendors a way to, at least temporarily, distinguish themselves from others. The experience of 802.11 WLANs shows that standards compliant solutions tend to quickly converge to a common cost and performance point. Successful vendors in that space have offered extra modes to differentiate their offerings. An important unknown in our sequence design work is how multipath changes our relative advantage over current schemes. Our work on classifiers robust to parameter mismatch will hopefully aid adoption, should it occur.

Appendix A

Tables of Sequences with Good Squared Distance

The table of best known squared distances is repeated for ease of reference.

$P\backslash n_r$	11	15	19	23	27	31	63
2	8	12	14	18	20	24	
3	8	12	16_3	18	22	24_3	
4	8	12	16	20	24_3	26	$32^\dagger$
5	**10**	$\mathbf{14}_3$	16	20	24	28	
6	**10**	**14**	$\mathbf{18}_2$	$\mathbf{22}_3$	24	28	
7	**10**	**14**	**18**	$\mathbf{22}_2$	$\mathbf{26}_2$	$\mathbf{30}_3$	
8	**10**	**14**	**18**	**22**	**26**	$\mathbf{30}_2$	60
16	**10**	**14**	**18**	**22**	**26**	**30**	**62**

Table A.1: Best known squared-distances for given n_r, P. Sequences in bold are strong. Subscripts indicate sequence dimension. All sequences without a subscript are simple.

All simple and strong sequences are of the form,

$$\boldsymbol{c} = \boldsymbol{s}_{\text{alt}} \cdot \boldsymbol{s}_{\text{alt}} \cdot \ldots \boldsymbol{s}_{\text{alt}} \cdot \boldsymbol{s}_{\text{alt}}^{(r)}$$

where,

$$\boldsymbol{s}_{\text{alt}} = [1\ 0\ 1\ 0\ \ldots\ 1\ 0\ 0]$$

and,

$$r = 2\left\lfloor \frac{n_r + 1}{8} \right\rfloor$$

The following tables cover the remaining cases. Sequences are identified by the (n_r, P, d_{sq}) tuple. Table A.2 covers the construction of strong MSDs which are not simple. Table A.3 covers all sequences that are not strong (but may be MSD).

(15,5,14)	$\{0, 0, \overline{9}, \overline{13}, \overline{13}\}$
(19,6,18)	$\{0, 0, 0, \overline{12}, \overline{12}, \overline{12}\}$
(23,6,22)	$\{0, 0, 15, 15, \overline{16}, \overline{16}\}$
(23,7,22)	$\{0, 0, 0, 0, 0, \overline{16}, \overline{16}\}$
(27,7,26)	$\{0, 0, 0, 0, \overline{14}, \overline{14}, \overline{14}\}$
(31,7,30)	$\{0, 0, 0, \overline{23}, \overline{29}, \overline{29}, \overline{29}\}$
(31,8,30)	$\{0, 0, 0, 0, 0, \overline{22}, \overline{22}, \overline{22}\}$

Table A.2: Sequences that are strong but not simple. All subcodes are shifts, and sometimes inversions of s_{alt}. Subcodes are listed in order from 0 to $P-1$, and specified by the right shift, with an overbar for inversion.

Table A.3: Sequences that are not strong. Subcodes q_i are specified by the location of 1s.

(11,2,8)	q_0={1, 3, 5, 8} q_1={1, 3, 5, 7}
(11,3,8)	q_0=q_1={1, 3, 5, 7} q_2={1, 3, 5, 6, 8}
(11,4,8)	q_0=q_1=q_2={1, 3, 5, 7} q_3={1, 3, 5, 6, 8}
(15,2,12)	q_0={1, 3, 5, 7, 10, 12, 13} q_1={1, 3, 5, 7, 9, 11, 12}
(15,3,12)	q_0=q_1={1, 3, 5, 7, 9, 11} q_2={1, 3, 5, 7, 8, 10, 11, 13}
(15,4,12)	q_0=q_1=q_2={1, 3, 5, 7, 9, 11} q_3={1, 3, 5, 7, 10, 12}
(19,2,14)	q_0={1, 3, 5, 7, 9, 11, 13, 14} q_1={1, 3, 5, 7, 9, 10, 12, 13, 17}
(19,3,16)	q_0={1, 3, 5, 7, 9, 11, 12, 14, 16, 17} q_1={1, 3, 5, 7, 9, 11, 13, 15} q_2={1, 3, 5, 7, 9, 11, 13, 15, 16}
(19,4,16)	q_0=q_1=q_2={1, 3, 5, 7, 9, 11, 13, 15} q_3={1, 3, 5, 7, 9, 11, 12, 14, 16}
(19,5,16)	q_0=q_1=q_2=q_3={1, 3, 5, 7, 9, 11, 13, 15} q_4={1, 3, 5, 7, 9, 11, 12, 14, 16}
(23,2,18)	q_0={1, 3, 6, 8, 10, 12, 14, 15, 18, 19, 21} q_1={1, 3, 5, 7, 9, 11, 13, 15, 17, 18}
(23,3,18)	q_0=q_1={1, 3, 5, 7, 9, 11, 13, 15, 17} q_2={1, 3, 5, 7, 9, 11, 14, 16, 17, 20}
(23,4,20)	q_0=q_1=q_2={1, 3, 5, 7, 9, 11, 13, 15, 17, 19}

Continued on next page

Table A.3 – continued from previous page

	q_3={1, 3, 5, 7, 9, 11, 13, 14, 16, 18, 19, 22}
(23,5,20)	q_0=q_1=q_2=q_3={1, 3, 5, 7, 9, 11, 13, 15, 17, 19} q_4={1, 3, 5, 7, 9, 11, 13, 16, 18, 20}
(27,2,20)	q_0={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 20} q_1={1, 3, 5, 7, 8, 10, 12, 13, 15, 18, 19, 20, 23, 26}
(27,3,22)	q_0={1, 2, 4, 6, 8, 10, 12, 14, 16, 19, 21, 22, 25} q_1=q_2={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21}
(27,4,24)	q_0={1, 3, 5, 7, 9, 11, 13, 15, 17, 20, 22, 23, 25} q_1=q_2={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23} q_3={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 22, 24}
(27,5,24)	q_0=q_1=q_2=q_3={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23} q_4={1, 3, 5, 7, 9, 11, 13, 15, 17, 18, 20, 22, 24}
(27,6,24)	q_0=q_1=q_2=q_3=q_4={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23} q_5={1, 3, 5, 7, 9, 11, 13, 15, 17, 18, 20, 22, 24}
(31,2,24)	q_0={1, 4, 7, 9, 10, 12, 14, 16, 19, 20, 21, 23, 25, 27, 28, 30} q_1={2, 4, 6, 8, 9, 11, 14, 16, 18, 19, 20, 23, 25, 27, 28, 30}
(31,3,24)	q_0={1, 3, 5, 7, 8, 9, 11, 13, 15, 18, 20, 21, 23, 24, 26, 29, 30} q_1={2, 4, 6, 8, 10, 13, 16, 18, 19, 22, 23, 25, 28, 30} q_2={1, 2, 4, 6, 8, 10, 12, 13, 15, 17, 18, 20, 24, 26, 29}
(31,4,26)	q_0=q_1=q_2={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25} q_3={1, 3, 5, 7, 9, 11, 13, 15, 17, 20, 22, 24, 25, 28}
(31,5,28)	q_0={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 22, 24, 26, 28} q_1=q_2=q_3=q_4={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27}
(31,6,28)	q_0=q_1=q_2=q_3=q_4={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27} q_5={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 22, 24, 26, 28}
(63,8,60)	q_0=q_1=q_2=q_3=q_4=q_5=q_6={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 45, 47, 49, 51, 53, 55, 57, 59}

Continued on next page

Table A.3 – continued from previous page

	$\boldsymbol{q}_7$={1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 39, 41, 43, 46, 48, 50, 52, 54, 56, 58, 60}

Appendix B

Tables of Walking Sequences

B.1 Block w-sequences

$t = 2, P = 2$	$\boldsymbol{b}_2$=(1,3,5)
$t = 3, P = 2$	$\boldsymbol{b}_2$=(1,2,5), $\boldsymbol{b}_3$=(2,4,5)
$t = 3, P = 3$	$\boldsymbol{b}_2$=(3,5,6), $\boldsymbol{b}_3$=(2,3,6)
$t = 4, P = 2$	$\boldsymbol{b}_3$=(1,2,4), $\boldsymbol{b}_4$=(1,3,5)
$t = 4, P = 3$	$\boldsymbol{b}_3$=(3,5,6), $\boldsymbol{b}_4$=(2,3,6)
$t = 5, P = 2$	$\boldsymbol{b}_3$=(1,3,4), $\boldsymbol{b}_4$=(1,2,3), $\boldsymbol{b}_5$=(1,4,5)
$t = 5, P = 3$	$\boldsymbol{b}_4$=(2,3,6), $\boldsymbol{b}_5$=(1,4,6)
$t = 6, P = 2$	$\boldsymbol{b}_3$=(1,4,5), $\boldsymbol{b}_4$=(2,3,4), $\boldsymbol{b}_5$=(1,2,3), $\boldsymbol{b}_6$=(1,3,5)
$t = 6, P = 3$	$\boldsymbol{b}_5$=(1,2,4), $\boldsymbol{b}_6$=(1,3,6)
$t = 7, P = 2$	$\boldsymbol{b}_3$=(2,3,4), $\boldsymbol{b}_4$=(1,4,5), $\boldsymbol{b}_5$=(1,2,5), $\boldsymbol{b}_6$=(1,2,4), $\boldsymbol{b}_7$=(1,3,5)
$t = 7, P = 3$	$\boldsymbol{b}_5$=(2,5,6), $\boldsymbol{b}_6$=(1,3,4), $\boldsymbol{b}_7$=(1,2,6)
$t = 7, P = 4$	$\boldsymbol{b}_5$=(4,6,7), $\boldsymbol{b}_6$=(1,5,7), $\boldsymbol{b}_7$=(2,3,7)
$t = 8, P = 2$	$\boldsymbol{b}_4$=(2,4,5), $\boldsymbol{b}_5$=(1,3,4), $\boldsymbol{b}_6$=(1,2,5), $\boldsymbol{b}_7$=(1,2,3), $\boldsymbol{b}_8$=(1,4,5)
$t = 8, P = 3$	$\boldsymbol{b}_5$=(3,5,6), $\boldsymbol{b}_6$=(2,5,6), $\boldsymbol{b}_7$=(1,3,4), $\boldsymbol{b}_8$=(1,2,6)
$t = 8, P = 4$	$\boldsymbol{b}_6$=(3,6,7), $\boldsymbol{b}_7$=(1,4,5), $\boldsymbol{b}_8$=(1,2,7)

Table B.1: Table of block w-sequences. Please refer to section 3.5 for an explanation of the notation.

B.2 Almost-block w-sequences

$t = 3, P = 4$	$\boldsymbol{b}_2$=(2,6)
$t = 3, P = 5$	$\boldsymbol{b}_2$=(4,7), $\boldsymbol{b}_3$=(2,7)
$t = 3, P = 6$	$\boldsymbol{b}_3$=(2,8)
$t = 4, P = 4$	$\boldsymbol{b}_2$=(2,5), $\boldsymbol{b}_3$=(1,6), $\boldsymbol{b}_4$=(1,6)
$t = 4, P = 5$	$\boldsymbol{b}_2$=(5,7), $\boldsymbol{b}_3$=(2,5), $\boldsymbol{b}_4$=(1,7)
$t = 4, P = 6$	$\boldsymbol{b}_3$=(3,8), $\boldsymbol{b}_4$=(5,8)
$t = 5, P = 4$	$\boldsymbol{b}_2$=(2,6), $\boldsymbol{b}_4$=(4,6), $\boldsymbol{b}_5$=(4,6)
$t = 5, P = 5$	$\boldsymbol{b}_3$=(4,7), $\boldsymbol{b}_4$=(2,4), $\boldsymbol{b}_5$=(1,7)
$t = 5, P = 6$	$\boldsymbol{b}_3$=(5,8), $\boldsymbol{b}_4$=(4,8), $\boldsymbol{b}_5$=(5,8)
$t = 5, P = 7$	$\boldsymbol{b}_3$=(7,9), $\boldsymbol{b}_4$=(1,9), $\boldsymbol{b}_5$=(4,9)
$t = 6, P = 4$	$\boldsymbol{b}_2$=(3,6), $\boldsymbol{b}_3$=(4,6), $\boldsymbol{b}_4$=(1,3), $\boldsymbol{b}_5$=(1,2), $\boldsymbol{b}_6$=(1,6)
$t = 6, P = 5$	$\boldsymbol{b}_4$=(3,7), $\boldsymbol{b}_5$=(3,5), $\boldsymbol{b}_6$=(1,7)
$t = 6, P = 6$	$\boldsymbol{b}_4$=(5,8), $\boldsymbol{b}_5$=(4,6), $\boldsymbol{b}_6$=(1,8)
$t = 7, P = 4$	$\boldsymbol{b}_3$=(4,6), $\boldsymbol{b}_4$=(3,6), $\boldsymbol{b}_5$=(3,6), $\boldsymbol{b}_6$=(1,4), $\boldsymbol{b}_7$=(1,6)
$t = 7, P = 5$	$\boldsymbol{b}_4$=(5,7), $\boldsymbol{b}_5$=(1,4), $\boldsymbol{b}_6$=(1,7), $\boldsymbol{b}_7$=(3,7)
$t = 7, P = 6$	$\boldsymbol{b}_4$=(6,8), $\boldsymbol{b}_5$=(5,8), $\boldsymbol{b}_6$=(3,6), $\boldsymbol{b}_7$=(1,8)
$t = 8, P = 4$	$\boldsymbol{b}_4$=(2,4), $\boldsymbol{b}_5$=(1,5), $\boldsymbol{b}_6$=(1,6), $\boldsymbol{b}_8$=(3,6)
$t = 8, P = 5$	$\boldsymbol{b}_5$=(3,7), $\boldsymbol{b}_6$=(2,5), $\boldsymbol{b}_7$=(1,3), $\boldsymbol{b}_8$=(1,7)
$t = 8, P = 6$	$\boldsymbol{b}_5$=(4,8), $\boldsymbol{b}_6$=(6,8), $\boldsymbol{b}_7$=(2,5), $\boldsymbol{b}_8$=(1,8)

Table B.2: Table of almost-block w-sequences of length $n_r = t(P + 2) + 1$. The first block ($\boldsymbol{b}_1$) has three zeros and the rest have two. Unspecified blocks are identity (see section 3.5.2).

Bibliography

[1] M. Abramowitz and I.A. Stegun. *Handbook of Mathematical Functions with formulas, graphs, and mathematical tables.* Courier Dover Publications, Ninth edition, 1970.

[2] M. A. Anders, S. K. Mathew, S. K. Hsu, R. K. Krishnamurthy, and S. Borkar. A 1.9 Gb/s 358 mW 16–256 State Reconfigurable Viterbi Accelerator in 90 nm CMOS. *IEEE Journal of Solid-State Circuits*, 43(1):214–222, 2008.

[3] J. B. Anderson and S. M. Hladik. An optimal circular Viterbi decoder for the bounded distance criterion. *IEEE Transactions on Communications*, 50(11):1736–1742, 2002.

[4] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv. Optimal decoding of linear codes for minimizing symbol error rate. *IEEE Transactions on Information Theory*, 20(2):284–287, 1974.

[5] C. Benkeser, A. Burg, T. Cupaiuolo, and Q. Huang. A 58mW 1.2 sq. mm. HSDPA Turbo Decoder ASIC in 0.13um CMOS. In *International Solid-State Circuits Conference (ISSCC)*, pages 264–265, 2008.

[6] C. Benkeser, A. Burg, T. Cupaiuolo, and Q. Huang. Design and Optimization of an HSDPA Turbo Decoder ASIC. *IEEE Journal of Solid-State Circuits*, 44(1):98–106, 2009.

[7] W.J. Blot and D.A. Meeter. Sequential experimental design procedures. *Journal of the American Statistical Association*, 68(343):586–593, 1973.

[8] G. E. P. Box and W. J. Hill. Discrimination among mechanistic models. *Technometrics*, 9(1):57–71, 1967.

[9] A.E. Brouwer, J.B. Shearer, N.J.A. Sloane, and W.D. Smith. A new table of constant weight codes. *IEEE Transactions on Information Theory*, 36(6):1334–1380, November 1990.

[10] A. R. Calderbank, G. D. Forney, Jr., and A. Vardy. Minimal tail-biting trellises: the Golay code and more. *IEEE Transactions on Information Theory*, 45(5):1435–1455, 1999.

[11] A. P. Chandrakasan, R. Amirtharajah, S. Cho, J. Goodman, G. Konduri, J. Kulik, W. Rabiner, and A. Wang. Design considerations for distributed microsensor systems. In *Proceedings of the IEEE Custom Integrated Circuits Conference (CICC)*, pages 279–286, 1999.

[12] H. Chernoff. A measure of asymptotic efficiency for tests of a hypothesis based on the sum of observations. *The Annals of Mathematical Statistics*, pages 493–507, 1952.

[13] H. Chernoff. Sequential design of experiments. *Annals of Mathematical Statistics*, 30:pp. 755–770, 1959.

[14] H. Chernoff. Sequential analysis and optimal design. In *8th Regional Conference in Applied Mathematics, Sponsored by National Science Foundation - Conference Board of the Mathematical Sciences*, New Mexico State University, 1972.

[15] S.H. Cho and A. P. Chandrakasan. A 6.5 GHz CMOS FSK modulator for wireless sensor applications. In *Symposium on VLSI Circuits*, pages 182–185, 2002.

[16] S. Cui, A.J. Goldsmith, and A. Bahai. Energy-constrained modulation optimization. *Wireless Communications, IEEE Transactions on*, 4(5):2349–2360, Sept. 2005.

[17] D. C. Daly, P. P. Mercier, M. Bhardwaj, A. L. Stone, J. Voldman, R. B. Levine, J. G. Hildebrand, and A. P. Chandrakasan. A pulsed UWB receiver SoC for insect motion control. In *International Solid State Circuits Conference*, pages 200–201, February 2009.

[18] Defense Advanced Research Projects Agency (DARPA). HiMEMS Program.

[19] S. Dolinar, D. Divsalar, and F. Pollara. Code performance as a function of block size. *JPL TDA Progress Report*, 42(133), 1998.

[20] V.P. Dragalin, A.G. Tartakovsky, and V.V. Veeravalli. Multihypothesis sequential probability ratio tests (i): Asymptotic optimality. *IEEE Transactions on Information Theory*, 45 (7):2448–2461, Nov., 1999.

[21] G. D. Forney, Jr. Exponential error bounds for erasure, list, and decision feedback schemes. *IEEE Transactions on Information Theory*, 14(2):206–220, 1968.

[22] G. D. Forney, Jr. Coset Codes – Part II: Binary Lattices and Related Codes. *IEEE Transactions on Information Theory*, 34(5):1152–1187, Sep 1988.

[23] G. D. Forney, Jr. *Concatenated Codes*. Sc.D. thesis, Masschusetts Institute of Technology, March, 1965.

[24] G. D. Forney, Jr. 6.451, Principles of Digital Communications – II. *MIT OpenCourseWare Lecture Notes*, Spring, 2005.

[25] S.W. Golomb. On the classification of cyclic Hadamard sequences. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*, 89(9):2247–2253, 2006.

[26] K. Gracie and M.H. Hamon. Turbo and turbo-like codes: principles and applications in telecommunications. *Proceedings of the IEEE*, 95(6):1228–1254, 2007.

[27] J. Hagenauer. Rate-compatible punctured convolutional codes (RCPC codes) and their applications. *IEEE Transactions on Communications*, 36(4):389–400, 1988.

[28] M. Hall Jr. *Combinatorial Theory.* John Wiley & Sons, Inc. New York, NY, USA, 1998.

[29] IEEE. Approved Draft Amendment to IEEE Standard-PART 15.4: Wireless Medium Access Control (MAC) and Physical Layer (PHY) Specifications for Low-Rate Wireless Personal Area Networks (LR-WPANs). *IEEE Approved Std P802.15.4a/D7*, January 2007.

[30] Xilinx Incorporated. 3GPP2 Turbo Decoder v2.1, Product Specification (DS275).

[31] S.Y. Jung and D.J. Park. Design of preamble signal for synchronization with UWB non-coherent energy detection receiver. In *IEEE International Conference on Ultra-Wideband*, pages 464–468, 2005.

[32] A. Lafourcade and A. Vardy. Optimal sectionalization of a trellis. *IEEE Transactions on Information Theory*, 42(3):689–703, 1996.

[33] F. S. Lee and A. Chandrakasan. A 2.5nJ/b 0.65V 3-to-5GHz subbanded UWB receiver in 90nm CMOS. In *International Solid State Circuits Conference*, pages 116–117, February 2007.

[34] C.C. Lin, Y.H. Shih, H.C. Chang, and C.Y. Lee. Design of a power-reduction Viterbi decoder for WLAN applications. *IEEE Transactions on Circuits and Systems I*, 52(6):1148–1156, 2005.

[35] F. J. MacWilliams and N. J. A. Sloane. *The Theory of Error-Correcting Codes.* North-Holland Amsterdam, 1977.

[36] A. S. Madhu and P. Shankar. Approximate MAP decoding on tail-biting trellises. In *International Symposium on Information Theory (ISIT)*, pages 1325–1328, Sept. 2005.

[37] P. Massaad, M. Medard, and L. Zheng. Impact of processing energy on the capacity of wireless channels. In *International Symposium on Information Theory and its Applications (ISITA)*, October 2004.

[38] J.L. Massey. Deep-space communications and coding: A marriage made in heaven. *Advanced Methods for Satellite and Deep Space Communications, Lecture Notes on Control and Information Sciences*, 182:1–17, 1992.

[39] D. Meeter, W. Pirie, and W. Blot. A comparison of two model-discrimination criteria. *Technometrics*, 12(3):457–470, August 1970.

[40] P. P. Mercier, M. Bhardwaj, D. C. Daly, and A. P. Chandrakasan. A 0.55V 16Mb/s 1.6mW non-coherent IR-UWB digital baseband with 1ns synchronization accuracy. In *International Solid State Circuits Conference*, pages 252–253, February 2009.

[41] B. Mielczarek and W.A. Krzymien. Adaptive hybrid ARQ systems with BCJR decoding. *IEEE Transactions on Vehicular Technology*, 57(3):1606–1619, May 2008.

[42] John G. Proakis. *Digital Communications.* McGraw-Hill, 4th edition, 2000.

[43] A. Rabbachin and I. Oppermann. Synchronization analysis for UWB systems with a low-complexity energy collection receiver. In *International Workshop on Ultra Wideband Systems*, pages 288–292, 2004.

[44] D. N. Rowitch and L. B. Milstein. On the performance of hybrid FEC/ARQ systems using rate compatible punctured turbo (RCPT) codes. *IEEE Transactions on Communications*, 48(6):948–959, 2000.

[45] A. Sahai. Why block length and delay are not the same thing? (Submitted to the IEEE Transactions on Information Theory), 2006.

[46] C. Shannon. The zero error capacity of a noisy channel. *IEEE Transactions on Information Theory*, 2(3):8–19, 1956.

[47] C.E. Shannon. Probability of error for optimal codes in a Gaussian channel. *Bell Syst. Tech. J*, 38(3):611–656, 1959.

[48] J. M. Shea. Reliability-based hybrid ARQ. *Electronics Letters*, 38(13):644–645, 2002.

[49] L. Stoica, A. Rabbachin, H. Repo, S. Tiuraniemi, and I. Oppermann. An ultra-wideband system architecture for tag based wireless sensor networks. *IEEE Transactions on Vehicular Technology*, 54(5):1632–1645, 2005.

[50] V. Sze, R. Blazquez, M. Bhardwaj, and A. Chandrakasan. An energy efficient sub-threshold baseband processor architecture for pulsed ultra-wideband communications. In *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, volume 3, pages 908–911, May 2006.

[51] S. Tong, D. Lin, A. Kavcic, L. Ping, and B. Bai. On the Performance of Short Forward Error-Correcting Codes. *IEEE Communications Letters*, 11(11):880, 2007.

[52] H. Urkowitz. Energy detection of unknown deterministic signals. *Proceedings of the IEEE*, 55(4):523–531, 1967.

[53] A. Wald. Sequential method of sampling for deciding between two courses of action. *Journal of the American Statistical Association*, 40(231):277–306, 1945.

[54] A. Wald. *Sequential Analysis.* John Wiley, New York, 1947.

[55] A. Wald and J. Wolfowitz. Optimum character of the sequential probability ratio test. *The Annals of Mathematical Statistics*, 19(3):326–339, 1948.

[56] A.Y. Wang, S.H. Cho, C.G. Sodini, and A.P. Chandrakasan. Energy efficient modulation and MAC for asymmetric RF microsensor systems. In *International Symposium on Low Power Electronics and Design*, pages 106–111, 2001.

[57] A.Y. Wang and C.G. Sodini. On the energy efficiency of wireless transceivers. In *IEEE International Conference on Communications (ICC)*, volume 8, pages 3783–3788, June 2006.

[58] D. Wentzloff and A. Chandrakasan. A 47pJ/pulse 3.1-to-5GHz All-Digital UWB Transmitter in 90nm CMOS. In *International Solid State Circuits Conference*, pages 118–119, February 2007.