Anytime coding on the infinite bandwidth AWGN channel: A sequential semi-orthogonal code

Anant Sahai
Department of Electrical Engineering and Computer Sciences
267 Cory Hall UC Berkeley, CA 94720-1770
e-mail: `sahai@eecs.berkeley.edu`

Abstract — **It is well known that orthogonal coding can be used to approach the Shannon capacity of the power-constrained AWGN channel without a bandwidth constraint. In this paper, I show that it is possible to do this using a semi-orthogonal variation of pulse position modulation that is sequential in nature — bits can be "streamed out" without having to buffer up blocks of bits at the transmitter. ML decoding of the code results in an exponentially small probability of error as a function of tolerated receiver delay and thus eventually a zero probability of error on every transmitted bit. Just as orthogonal coding is related to the idea of capacity per unit cost, this code suggests an interesting sequential version of that problem.**

I. Introduction

Shannon's capacity theorem is arguably the greatest accomplishment in communications theory. Together with the source-channel separation theorem, it established the fundamental role of bits in communication. Unfortunately, the random coding proof is non-constructive in that does not give a construction for any explicit code. This has led to the proverb: "Almost all codes are "good" codes except for the the ones that we can think of."[10] However, there is a channel for which explicit non-random constructions for capacity-achieving codes exist: the continuous-time AWGN channel with an input power constraint and no bandwidth constraint.

We model the noise process as white with intensity $\frac{N_0}{2}$. The capacity of the channel is most naturally expressed in terms of energy per bit and is given by:

$$E_b > N_0 \ln 2 \tag{1}$$

which means that reliable communication is possible if the normalized energy per-bit exceeds $\ln 2$. When viewed in terms of bits per unit time, it means that reliable communication requires:

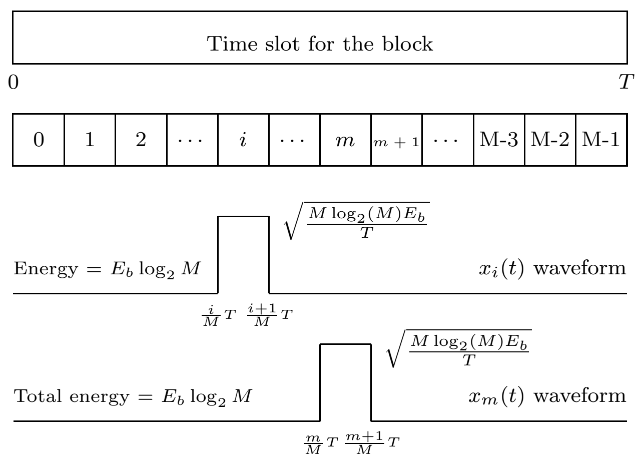

Figure 1: Pulse position modulation illustrated in the block coding framework.

$$R < C_\infty = \frac{P}{N_0}\log_2 e \tag{2}$$

where P represents the allowed power per unit time.

For such channels, it is further known that orthogonal signaling can be used to achieve data rates arbitrarily close to capacity. One such orthogonal signaling scheme is pulse position modulation (PPM) as depicted in figure 1. Suppose we have M distinct messages that we want to be able to transmit during one time slot of duration T. To communicate message $0 \le m \le M-1$, the transmitter sends out a burst of its allocated transmit power during the time slot $[\frac{m}{M}T, \frac{m+1}{M}T]$ and is silent during the rest of the time slots. Since the time-slots are disjoint, the waveforms $x_m(t)$ corresponding to different messages m are necessarily orthogonal over the interval $[0, T]$.

The receiver can do simple maximum-likelihood detection by having a bank of M matched-filters that correlate the received signal $Y(t)$ with the M disjoint waveforms. The result of this correlation can be considered as Z_i for message i and due to the AWGN assumption, can be

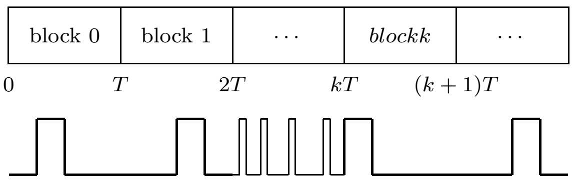

Figure 2: One block after another where each block uses an orthogonal code.

modeled as:

$$Z_i = \begin{cases} N_i & \text{if } i \neq m \\ \sqrt{2E_b \log_2 M} + N_m & \text{if } i = m \end{cases} \quad (3)$$

where the N_i are iid standard zero-mean unit-variance[1] Gaussian random variables, E_b represents the normalized energy per bit, and m represents the true message sent. $\log_2 M$ represents the number of bits to be communicated.

In this case, ML decoding consists of finding the time-slot with the highest Z_i. Classical analysis of this channel [3] shows that the orthogonal code with maximum-likelihood (ML) decoding has a probability of block error that goes to zero exponentially with block duration:

$$P_e \leq K e^{-T E_{orth}(R)} \quad (4)$$

where K is a rate dependent constant and

$$E_{orth}(R) = \begin{cases} (\frac{C_\infty}{2} - R)\ln 2 & \text{if } 0 \leq R \leq \frac{C_\infty}{4} \\ (\sqrt{C_\infty} - \sqrt{R})^2 \ln 2 & \text{if } \frac{C_\infty}{4} < R < C_\infty \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

For a situation in which bits arrive from the source at regular intervals, the traditional view involves buffering up a block of $\log_2 M$ bits, and then sending out the block of data1 using orthogonal signaling while we wait for the next block of data bits to arrive at the encoder. Because there is no bandwidth constraint, the duration of signalling can be made as small as we like and hence essentially all the end-to-end delay can be attributed to the buffering at the encoder.[2]

It is natural to ask if the placement of the delay at the encoder is fundamental or whether it can be moved to the decoder. In the following section, I give a sequential version of pulse position modulation that has a "semi-orthogonality" property in that the waveforms corresponding to different possible bitstreams are orthogonal beyond the point where they first differ in any bit's value. Analysis of this scheme shows that it is possible to shift the delay entirely to the decoder without any loss of reliability. Furthermore, the encoder naturally has the "anytime" or "delay-universal" property in that the target delay does not need to be known to the encoder. The same encoder works for all values of delay. A side effect of this property is that every bit is eventually perfectly known at the decoder. I close with some discussions of extensions and implications of this result. In particular, this result seems to extend naturally to the capacity per unit cost framework where it suggests a sequential version of the problem. Eventually, I hope that this might help us better understand the role of encoder burstiness and decoder delay in reliable communication.

II. The semi-orthogonal code

In this section, I motivate and give a sequential version of pulse position modulation.

II.A Motivation: zero-rate coding by repetition

Suppose that we were facing a binary symmetric channel. We all know that the repetition code is an excellent code at zero-rate. To achieve a target reliability, we just repeat a bit as many times as needed. To achieve perfect reliability, we just repeat it infinitely often. Now, suppose that bits were arriving at the encoder regularly, but we did not care about the delay in decoding them at the decoder nor about any increase in delay with bit position. One strategy to communicate reliably would be as follows:

1. Initialize $i = 1$
2. Output every bit B_1^i
3. Increment i
4. goto step 2.

This would result in an output stream $B_1; B_1, B_2; B_1, B_2, B_3; B_1, B_2, B_3, B_4; \ldots$ where the semicolons are used to denote the points of time at which we start repeating bits again and the commas mark the times between channel uses. It is clear that if the decoder waits long enough, it will get enough repetitions of any individual bit to achieve its target reliability.

While this code has zero rate and has increasing required delay with time, it is possible to modify this scheme for use on the infinite bandwidth AWGN channel with finite rate, finite energy per bit, and fixed delays.

II.B Repeated/refined pulse position modulation

Suppose that bits are arriving every τ seconds and we are allowed E_b energy per bit. Rather than waiting to build up a buffer of $\log_2 M$ bits and then signaling, suppose we instead spent our E_b energy immediately but used it to "repeat" the value of every bit received so far as in the scheme of section II.A. The scheme is illustrated in figure 3. It "refines" the information as time goes one.

[1]The $2E_b$ term is there to achieve this normalization of the noise.

[2]This is different in the traditional picture as applied to a DMC or any other finite degree-of-freedom channel. In that picture, T must be of a certain length and hence the delay is essentially split between the encoder and the decoder resulting in an end-to-end delay of around $2N - 1$ channel uses for a block-code of size N.

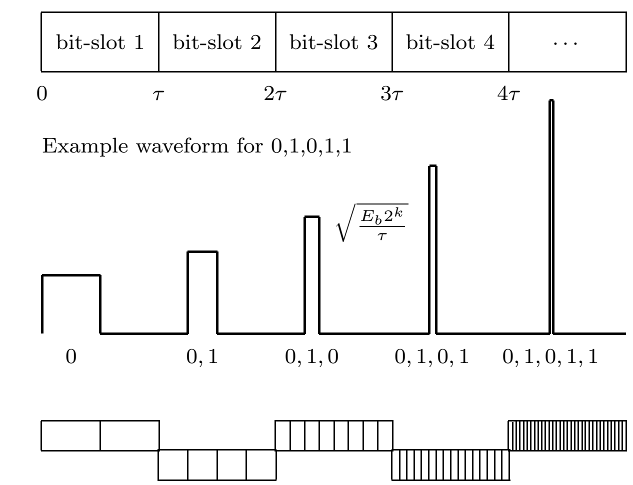

Figure 3: Repeated pulse position modulation illustrated. The time slots are on the top and the possible sub-slots are on the bottom.

Lemma II.1 *The repeated pulse position modulation code is semi-orthogonal.*

If x is the waveform corresponding to the bitstream b, and x' is the waveform corresponding to the bitstream b', then $x([(j-1)\tau, T]$ is orthogonal to $x'([(j-1)\tau, T]$ whenever there exists a bit position $i \leq j$ for which $b_i \neq b'_i$.

Proof: This is a simple consequence of the orthogonality of traditional pulse-position modulation. Since the underlying data bits differ somewhere at or before j, then over each time-slot after $(j-1)\tau$ the signals x and x' have disjoint support. $\square$

Within the time slot $[(k-1)\tau, k\tau]$, we divide it into 2^k disjoint sub-slots and put all our energy (E_b) into the sub-slot that corresponds to the realization of B_1^k that we have seen so far at the encoder. If our target rate is $R = \frac{1}{\tau}$ and our target average power per unit time is P, then by using $E_b = \frac{P}{R}$, it is clear that this scheme meets our target power constraint — not just when we average over the realizations of the incoming bits B, but for every possible sequence of bits. Figure 4 illustrates the natural tree structure of this code.

The decoder is assumed to have a target delay of $d\tau$ seconds and to be interested in estimating the value of the bits with that delay. In order to study asymptotic behavior, we are interested in the case of d large but finite.

III. Analysis of P_e

Because this is a sequential encoding scheme that is going to be used with finite delay, the relevant error-event is a bit-error, not a block error. We want the probability of bit-error to go to zero with delay. However, whenever we are dealing with bit-errors, it is important to specify which bit we are talking about. We will consider a code to achieve reliability $E_a(R)$ if there exists a rate-dependent constant K' so that

$$P(\hat{B}_i \neq B_i) \leq K' e^{-d\tau E_a(R)}$$

for every $i > 0$.

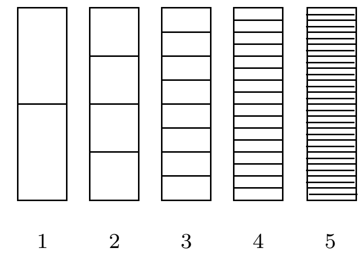

Figure 4: The sub-slots in each timeslot are refinements of the sub-slots in the previous timeslot. This gives rise to a natural tree structure and the semi-orthogonality property of the code.

Our focus will be on ML decoding. To get $\hat{B}_i$, the decoder has access to the received waveform $Y(t)$ over the interval $t \in [0, (i+d)\tau]$. Since we are not assuming an a-priori joint distribution over the B_i, we will follow the following strategy:

- For every possible bit-sequence $\check{b}_1^{i+d}$, compute the log-likelihood $\ln p(Y([0,(i+d)\tau]) = y([0,(i+d)\tau])|B_1^{i+d} = \check{b}_1^{i+d})$. By the white nature of the noise, we know:

$$\begin{aligned} &\ln p(Y([0,(i+d)\tau]) = y([0,(i+d)\tau])|B_1^{i+d} = \check{b}_1^{i+d}) \\ &\quad = \sum_{j=1}^{i+d} \ln p_Y(y([(j-1)\tau, j\tau])|B_1^j = \check{b}_1^j) \end{aligned} \tag{6}$$

- Pick the most likely sequence $\tilde{B}_1^{i+d}$ and emit its i-th position. In the white-Gaussian case, this will reduce to a picking the bit-sequence that results in the minimum Euclidean distance between waveforms.

It is important to note that the decisions are not remembered in this decoder. In principle, it recomputes the ML path each time.

III.A Suffix-error analysis

Suppose that a genie gave the decoder access to the correct value of the bits B_1^{i-1}. Since we know the truth before time $(i-1)\tau$ and (6) tells us that the log-likelihood is additive across time, we only need to consider $\ln p(Y([(i-1)\tau, (i+d)\tau]) = y([(i-1)\tau, (i+d)\tau])|B_1^{i+d} = \check{b}_1^{i+d})$. The total duration of the relevant signal is thus $(d+1)\tau$.

The only way that we can get an error is if one of the 2^d bitstreams with $\check{b}_i \neq b_i$ has a larger likelihood than the true stream. By Lemma II.1, the true waveform is orthogonal to all the false waveforms under consideration. Recalling that the analysis of probability of block-error calculation in [3] used only the union bound and the fact that the true waveform was orthogonal to each of the false ones[3], we can immediately apply (4) to see that

$$P(\hat{B}_i \neq B_i \text{ at delay } d | B_1^{i-1} \text{ known}) \leq K e^{-(d+1)\tau E_{orth}(R)} \tag{7}$$

III.B Dealing with the uncertain prefix

The actual decoder does not know B_1^{i-1}. However, we can bound the error probability as follows:

$$\begin{aligned} & P(\hat{B}_i \neq B_i \text{ at delay } d) \\ \leq & \sum_{j=0}^{i-1} P(\hat{B}_{i-j} \neq B_{i-j} \text{ at delay } d+j | B_1^{i-1-j} \text{ known}) \end{aligned} \tag{8}$$

since to make an error at bit i the most likely sequence has to differ from the true sequence at i or earlier. The regular union bound then gives us (8) with the earlier positions having correspondingly increased delays.

Combining (8) with (7) gives us:

$$\begin{aligned} & P(\hat{B}_i \neq B_i \text{ with delay } d) \\ \leq & \sum_{j=0}^{i-1} K e^{-(d+1+j)\tau E_{orth}(R)} \\ < & K\left(\sum_{j=0}^{\infty} e^{-(j+1)\tau E_{orth}(R)}\right) e^{-d\tau E_{orth}(R)} \\ = & \frac{K}{e^{\tau E_{orth}(R)} - 1} e^{-d\tau E_{orth}(R)} \end{aligned}$$

which gives us the desired result with $E_a(R) = E_{orth}(R)$ and no dependence on the bit position i and delay d.

Theorem III.1 *The repeated pulse position modulation code under maximum-likelihood decoding for individual bits with delay $d\tau$ achieves the orthogonal coding error exponent for every delay and bit position.*

$$P(\hat{B}_i \neq B_i \text{ with delay } d) \leq K' e^{-d\tau E_{orth}(R)} \tag{9}$$

An immediate consequence of theorem III.1 is that this code achieves zero probability of error on every bit in the limit of large delays. The limit here is purely at the decoder rather than being over encoder-decoder pairs. As such, it shows more clearly what the nature of reliable communication can be over the infinite-bandwidth channel. Using the language of anytime reliability [5], theorem III.1 establishes an explicit lower bound on the anytime reliability of the infinite-bandwidth AWGN channel.

[3]The analysis in [3] proceeded by approximating the error event by the union of two events: that the noise in the direction of the true codeword is large and the event that a false codeword beats the true codeword conditioned on the fact that the noise in the direction of the true codeword is small. The straight union bound over all the false codewords was then used for the second event. By adjusting what is meant by "large/small," the appropriate probabilities, the two terms could be matched in the exponent and gave the exponential bound.

IV. Interpretations and extensions

I have constructed a new sequential variation on orthogonal coding for the infinite-bandwidth AWGN channel called repeated pulse position modulation. (RPPM) This code has the special property that it pushes all the delay to the decoder and moreover it is delay-universal in that the decoder can choose any delay that it might like and attain exponentially good reliability with that delay choice. Thus, it is an explicit construction that complements the random coding arguments in [5, 7]. The existence of feedback-free anytime codes allows us to better understand the performance of systems that have access to noisy feedback as recently shown in [6] as well as approach the problem of interactive computations over noisy channels.[8]

The code as described is a pulse-position modulation variation that ends up requiring unboundedly large peak amplitudes while meeting a hard average power constraint. Because all that is required is orthogonality on each time slot, we can use any orthogonal signaling we prefer. In particular, we could use signals that have constant amplitude and just change abruptly in phase. So this type of signaling can be done while meeting a hard amplitude constraint.

So, is this a practical tool for data communication? It seems likely that the answer to this question is negative. Orthogonal signaling can be very wasteful of bandwidth and the RPPM scheme given here actually uses ∞ bandwidth which is never available in practice. The goal here is rather to refine our understanding of the role of delay in reliable communication and the tradeoffs possible between the encoder and decoder. It provides another extreme point balancing block-codes on the other side. As such, it is best to consider its theoretical rather than practical implications.

IV.A Capacity per unit cost

Verdu's capacity per unit cost framework [9] is the natural generalization of the infinite-bandwidth power-constrained AWGN channel. To see how the main result of this paper translates, we first can reinterpret the error calculation so far as a function of the number of bits intervening rather than the time delay and rate. Apply the substitutions $E_b = \frac{P}{R}$, $\tau = \frac{1}{R}$ to (2), (9), and (5) to get:

$$P(\hat{B}_i \neq B_i \text{ with } d \text{ bits intervening}) \leq K' e^{-dE_{orth}(\frac{E_b}{N_0})} \tag{10}$$

where $E_{orth}(\frac{E_b}{N_0}) =$

$$\begin{cases} (\frac{E_b}{N_0}(\frac{1}{2\ln 2}) - 1)\ln 2 & \text{if } \frac{E_b}{N_0} > 4\ln 2 \\ (\sqrt{\frac{E_b}{N_0}(\frac{1}{\ln 2})} - 1)^2 \ln 2 & \text{if } \ln 2 < \frac{E_b}{N_0} \leq 4\ln 2 \\ 0 & \text{otherwise} \end{cases} \tag{11}$$

With this done, we no longer require the infinite-bandwidth assumption as long as we are interested only in the probability of error going to zero as a function of intervening energy. This gives rise to the natural "sequential" version of Verdu's capacity per unit cost framework:

- There is a zero-cost channel input available.
- The encoder gets access to the bits one at a time and can use any number of channel uses it likes.
- The total cost of the channel uses so far must be less than E_b times the number of bits the encoder has received.
- The decoder wants to estimate the values of all the bits, but is willing to wait until the encoder has spent a certain extra amount dE_b more than it had when it first got access to the desired bit.

As the encoding strategy in [9] is a kind of pulse position modulation, the repeated pulse-position-modulation strategy described here translates directly into that framework.[4] All we have to do is interpret the sub-slots depicted in figure 3 as individual channel uses. The semi-orthogonality property of Lemma II.1 translates directly into a semi-disjoint support property. We will use the ML detector in the natural form and (6) continues to apply as appropriately interpreted. For error analysis aiming at an analogue of (7), we can bound ML performance by the suboptimal, yet simple, hypothesis-testing based decoder in [9]. Since the error analysis in [9] relies only on the disjointness of the true codeword to each false one individually, all the arguments given here extend directly.[5] All that is required for the prefix-argument of section III.B to work is that the probability of error go to zero exponentially in d. Although this is not explicit in the stated proof in [9], it does indeed hold.[6]

There is only one new consideration: integer effects. In the continuous amplitude situation, we could measure out any desired energy and pour it into a disjoint time period. For a DMC with an input cost constraint, there might be no way to hit the desired cost with a single input and so some way of time-sharing between inputs is essential.

It is here that we can see a role for a finite additional "delay" being imposed at the encoder. If the encoder decides to "burst" its output, it can do so by buffering up bits (spending no channel cost) until it has L of them ready to go, and is willing to spend LE_b cost to do the incremental encoding. By letting L get large, LE_b gets big enough to smooth out any integer constraint.[7] However, the probability of error analysis given earlier continues to hold and the probability of L-burst error will go to zero exponentially at the correct rate with respect to cost increments.

In the capacity per unit cost formulation, this scheme or finitely truncated versions of it might actually have practical consequences. Consider a sensor network with very little energy available for long-range communication, but also very little data to send. If the data is going to be used by some application, the sensor may not know what the acceptable "delay" is. By using an anytime or delay-universal scheme like the one presented here, a sensor might be able to leave that choice to the decoder.

IV.B Exploring the fundamental role of bursting

In the previous section, we sketched how we could use bursting at the encoder to smooth out integer-effects regarding the input cost. As such, the role of the bursting-induced delay at the encoder was different from the role of the delay at the decoder. If we think about a rate/cost and reliability region, the encoder delay was used to make a certain (rate/cost, reliability) pair achievable. Once that was done, it was the decoder delay that appeared essentially in the reliability calculations.

It remains to be seen whether this situation is in fact generic. Analysis based on the block-coding paradigm can not easily distinguish between the encoder and decoder. Long blocks are used to average over all the stochastic uncertainty in the system. Two examples where different effects all come together are:

- Writing on dirty paper.[1] Here, the advance knowledge of the interference is getting longer as the block-length increases. Initial work marrying this scheme to that given in [4] leads us to suspect (though it is far from certain) that bursting/delay at the encoder can be used to get the capacity up while the decoding delay is used to get the probability of error down.
- Fading wideband channels. Here, being able to burst the signal allows us to: amortize the cost of channel estimation over many bits and to ignore parts of the channel variation through

[4] We will use the zero-cost channel input in most places, except for the position that corresponds to all the bits so far. There we use the appropriate more expensive channel input.

[5] For a given prefix, the true suffix has d expensive channel inputs at the appropriate places. The false suffices all have the zero cost channel input in that place. Similarly, they all have their expensive channel inputs where the true suffix has zeros. Thus the pairwise competition works.

[6] Look at the set of types that we we will accept as representing that a "pulse" is present at the appropriate set of channel inputs. The threshold corresponds to how much of a margin around the type P_0 we are going to accept. All that matters is that there is a margin and so the probability of missing the true "pulse" is exponentially small with d. The Stein's lemma argument already gives us an exponentially small probability of false alarm using the union bound. By choosing the threshold to equalize the exponents, we have a single exponent. The decoder analyzed in [9] might not give the best possible exponent, but all we need here is that it does give us an exponent. To see why is it not the best, the reader is encouraged to carry out this analysis for the AWGN case.

[7] Having a budget of LE_b allows us to choose the appropriate mix of channel inputs in a sub-slot that together cost less than LE_b but have close to the desired divergence.

time/frequency. It will be interesting to see if the techniques/ideas here can allow us to establish that bursting at the encoder and processing at the receiver also can have complementary roles — with encoder bursting used to get the appropriate amortization/ignoring effects while decoder delay gets the probability of error as low as we like.

This exploration might have practical significance since it could allow us to make better use of channels where bursting is restricted to some finite values too small to allow for the desired probability of error. Currently the main practical tool we have for such situations is concatenated coding [2] where we can use a limited lookahead/bursty inner-code together with a more robustly error-correcting outer code. The idea here is the same, but where the outer code is sequential in nature and the inner-code does not push the error probability to be small, but rather "improves the channel" by exploiting side-information, training-data, or any other special features of the problem.

Acknowledgments

Thanks to Pramod Viswanath for suggesting that there might be a connection to the capacity per unit cost formulation.

References

[1] M Costa, "Writing on dirty paper," *IEEE Transactions on Information Theory*, 29:439–441, May 1983.

[2] G. D. Forney, Jr., *Concatenated Codes.* Cambridge, MA: M.I.T. Press, 1966.

[3] R.G. Gallager, *Information Theory and Reliable Communication.* New York, NY: John Wiley and Sons, 1971.

[4] T. Liu and P. Viswanath, "Opportunistic orthogonal writing on dirty paper," SPCOM 04.

[5] A. Sahai, "Any-time Information Theory," PhD Dissertation, MIT February 2001.

[6] A. Sahai and T. Simsek, "On the variable-delay reliability function of discrete memoryless channels with access to noisy feedback," *IEEE Workshop on Information Theory*, Nov 2004

[7] A. Sahai and S. K. Mitter, "Source coding and channel requirements for unstable processes." To be submitted to *IEEE Transactions on Information Theory*, February 2005.

[8] L. Schulman, "Coding for Interactive Communication." *IEEE Trans. on Information Theory*, Vol 42, pp. 1745-1756, November 1996.

[9] S. Verdú, "On Channel Capacity per Unit Cost," *IEEE Trans. Information Theory*,36:1019-1030, Sep 1990.

[10] Jack Wolf, "An Introduction to Turbo Codes: The Quintessential Channel Coding Technique," lecture given at MIT LIDS, March 2000.