Performance of Binary Turbo-Coded 256-ary Pulse-Position Modulation

J. Hamkins[1]

This article compares the performance of binary turbo codes and Reed–Solomon (RS) codes in an optical communications system employing high-order pulse-position modulation (PPM) and an avalanche photodiode (APD) detector. Despite restriction of the turbo codes to binary alphabets and very simple 4- or 8-state encoders, they have coding gains 0.5-dB to 0.7-dB higher than equivalent-rate RS codes when using 256-PPM, employing a typical APD detector, and operating at an end-to-end bit-error rate (BER) of 10^{-5}. BER curves are given for code rates of 1/3 and 1/2, using a variety of turbo and RS coding methods. The simulations use an accurate modeling of APD output statistics instead of typical but less accurate Poisson or Gaussian approximations.

I. Introduction

This article concerns the optical communications system shown in Fig. 1. Information bits are encoded, modulated using 256-ary pulse-position modulation (PPM), transmitted across an optical channel, and received by an avalanche photodiode (APD) detector. For each PPM symbol, the APD produces 256 soft outputs, one for each slot.

A PPM signaling scheme seems to lend itself naturally to a Reed–Solomon (RS) code, whose alphabet size can be easily matched to the PPM order. This results in a one-to-one correspondence between RS code symbols and PPM symbols. Using a maximum count or threshold hard-decision rule [13], a single PPM error translates directly into a single RS-code symbol error. Or, by using threshold or δ-max demodulation [8], an erasure may be declared based on the soft counts in a PPM symbol. This is

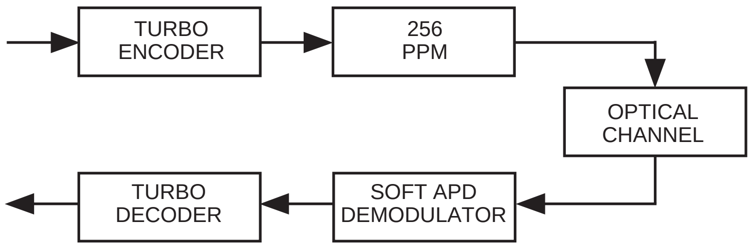

Fig. 1. A turbo-coded optical communication system.

[1] Communications Systems and Research Section.

sometimes referred to as soft-decision RS coding. Since every RS code is a maximum-distance code, it might seem that no more powerful code could be found for this application.

However, one disadvantage of an RS code is that its decoder does not make full use of soft outputs. Even a soft-decision RS decoder does not operate on the soft-demodulator outputs themselves, only on PPM symbols or erasures. Furthermore, this is an inherent limitation of RS codes, because there is no practical method known to modify RS codes to make full use of soft information. This is the motivation for using turbo codes, which already are known to have outstanding performance in other applications and can operate on the soft counts directly.

The initial appearance of turbo codes [4] launched a whirl of research activity, aimed first at merely reproducing the astonishing results (see, e.g., [10]), and then at improving, generalizing, and simplifying the approach (see, e.g., [2,3,5,6,9]). Despite the concerted effort, fundamentally there still is not a complete theoretical understanding of the relationship between the iterative decoder and the maximum-likelihood decoder. Consequently, when maximum-likelihood performance bounds have not provided an accurate estimate of an iterative decoder performance for a particular application or signal-to-noise ratio (SNR) region, simulation results have been an effective substitute. Without a theoretical bit-error rate (BER) curve that closely matches simulations, however, it is important for researchers to generate simulations that accurately model a real system. For this reason, this article uses a very accurate model of APD detector statistics [11].

This article makes two main contributions: (1) a description of how binary turbo codes can be used on a nonbinary optical channel *with full utilization of soft information* and (2) simulation of end-to-end BER performance for baseline RS codes and binary turbo codes using an accurate statistical model of the APD detector. Section II describes the components of Fig. 1; Section III describes the turbo decoder; and Section IV discusses the performance simulations.

II. Turbo-Coded Communication System Components

The primary difficulty of applying a turbo code to an optical channel is that of matching turbo-code symbols to high-order PPM symbols. The optimal PPM order for low SNR space applications can be 256 or higher [7,16], whereas 256-ary turbo codes are a practical impossibility because turbo-decoding complexity is exponential in the symbol alphabet size. At the other extreme, binary turbo codes can be applied to 2-PPM [12], but at the expense of photon efficiency [7] or data rate [16].

As we shall see, the symbol-matching problem is largely a perceived one. Using previously published turbo-decoding algorithms, binary turbo codes can make full use of 256-PPM soft-demodulator outputs by suitably initializing a priori probabilities in the turbo decoder. The performance likely would be better with nonbinary turbo codes, but the binary turbo codes also perform well and represent an improvement over RS codes.

A. Turbo Encoder

A rate-1/3 turbo encoder is shown in Fig. 2. It consists of two systematic rate-1/2 constituent codes and an interleaver of length N. The input $\mathbf{u} = (u_1, \cdots, u_N)$ is a block of independent, uniformly distributed information bits.

In this article, the constituent codes shall be rate 1/2, recursive, systematic convolutional codes. An example of such a convolutional code with termination as described in [10] is shown in Fig. 3. The transfer function of the code shown is $G(D) = [1, (1+D^2)/(1+D+D^2)]$ or, in octal notation, $[1, 5/7]$. The switch in Fig. 3 is in position A for the first N time steps and is in position B for the remaining $m = 2$ steps needed to return to the all-zeroes state, where m is the memory of the code. Branches in the trellis are labeled in the usual way as u/x_0x_{1p}, where u is the input bit, $x_0 = u$ is the systematic output bit, and

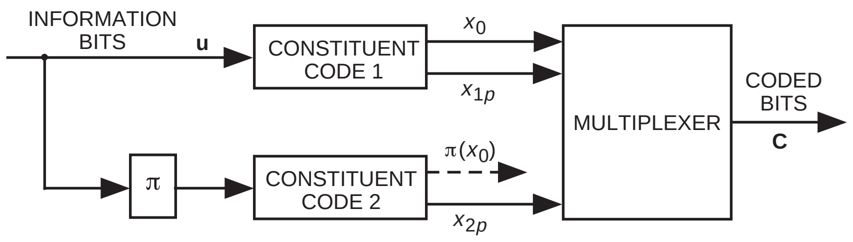

Fig. 2. A rate-1/3 turbo encoder.

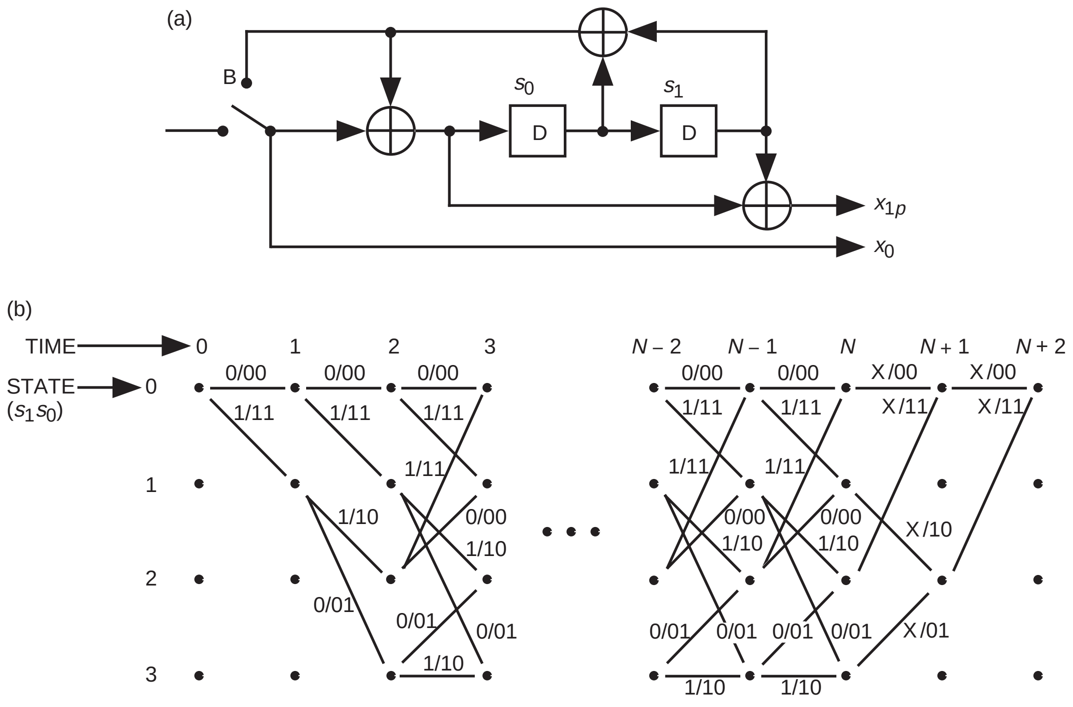

Fig. 3. An example of (a) a convolutional code with trellis termination and (b) the corresponding trellis.

x_{1p} is the parity output bit. During the termination phase, there are no input bits; these are marked as X's on the trellis branches. The output x_0 during this phase is the value of the feedback bit.

The three outputs of the rate-1/3 turbo encoder include the systematic bit, x_0, and the two parity bits, x_{1p} and x_{2p}. The permuted systematic bit from the second convolutional code is not transmitted across the channel. Thus, a total of $n = 3(N + m)$ bits are transmitted for the rate-1/3 code. Or, the bits x_{1p} and x_{2p} may be punctured, resulting in a different code rate. For example, by puncturing x_{1p} on even time steps and x_{2p} on odd time steps, a rate-1/2 code is obtained.

B. Modulator

The simulations use a 256-PPM signaling scheme. The coded bits are grouped into 8-bit sub-blocks and fed to the PPM modulator, which transforms each sub-block into a pulse in one of 256 locations, as shown in Fig. 4. For convenience, we assume n is divisible by 8. Because of the nonbinary signaling, the order in which the coded bits are grouped can affect performance. This will be discussed more in Section IV.

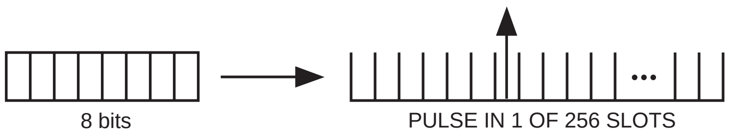

Fig. 4. 256-PPM.

C. Optical Channel

The signal-light intensity generated by the transmitting laser is corrupted by background light, as shown in Fig. 5. The number of photons absorbed by a receiver from an incident optical field of known intensity is a Poisson-distributed random variable [11]. In this article, for simplicity, we assume perfect timing synchronization and no spreading of signal photons across slots, which implies that the number of absorbed photons in each slot is independent of the number of photons absorbed in all other slots. Note that this is not a crucial assumption in the turbo-coding approach taken in this article—if the statistical properties of synchronization error and spreading were known, this could be incorporated into the algorithm. In fact, in systems that exhibit timing jitter or pulse spreading, we believe that the turbo-coding technique described in this article would display even greater performance improvement over a decoder that operates on PPM symbols and erasures only.

D. Demodulator

The detection of weak optical signals is enhanced by a low-noise APD detector that amplifies the electrical current generated by absorbed photons. This is illustrated in Fig. 6, in which the diode symbol represents the more complicated solid-state components of the APD itself; some of the APD parameters are shown in block diagram form. Unfortunately, in addition to amplifying the signal, the APD transforms the simple Poisson distribution of absorbed photons into a much more complicated probability density function (pdf) at the APD output. This pdf has been accurately approximated by Webb [15]. In particular, the probability that m secondary electrons are emitted from the APD in response to the absorption of, on average, $\bar{n}$ primary photons in a slot, is approximated by Webb as

$$P(m|\bar{n}) = \frac{\exp\left[-\dfrac{(m-G\bar{n})^2}{2\bar{n}G^2F\left(1+\frac{m-G\bar{n}}{\bar{n}GF/(F-1)}\right)}\right]}{\sqrt{2\pi\bar{n}G^2F}\left[1+\dfrac{m-G\bar{n}}{\bar{n}GF/(F-1)}\right]^{3/2}} \tag{1}$$

where G is the average APD gain, where F is the excess noise factor given by

$$F = k_{eff}G + \left(2-\frac{1}{G}\right)(1-k_{eff})$$

and where k_{eff} is the ionization ratio. For values of m close to its mean, $G\bar{n}$, Eq. (1) can be approximated by a Gaussian pdf; however, $P(m|\bar{n})$ departs greatly from a Gaussian pdf at both tails, which form the main contribution to error events in decoders [11].

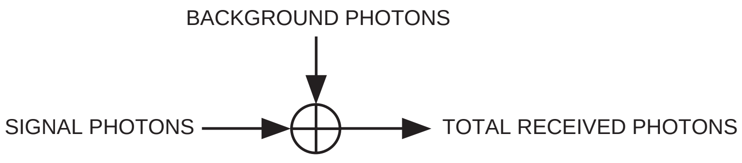

Fig. 5. The optical channel.

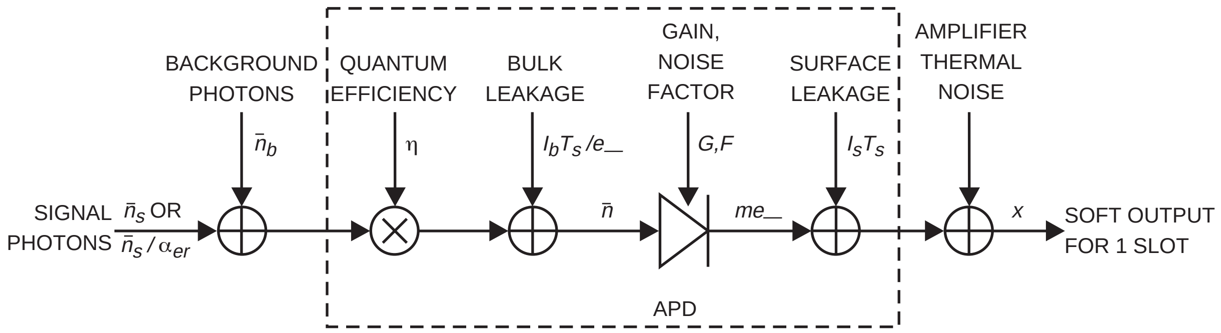

Fig. 6. The soft APD demodulator.

The APD output is the sum of the charge due to Webb-distributed secondary electron emissions and Gaussian-distributed amplifier thermal noise, as shown in Fig. 6. The APD output for a 256-PPM signal consists of 256 soft outputs, each a continuously distributed random variable independent of all other soft outputs.

Denote the pdf of the soft output of a slot by $p_S(\cdot)$ or $p_N(\cdot)$, depending on whether it is a signaling slot or non-signaling slot, respectively. This density is found by using the appropriate value of $\bar{n}$ in Eq. (1) and convolving with a Gaussian distribution representing the amplifier thermal noise.

In a conventional implementation, a hard decision is made at the APD output by choosing the slot with the maximum soft output value. Although intuitive, only recently has this been proven to be the maximum-likelihood rule for detecting uncoded PPM symbols [14]. Or, one may set a fixed threshold and generate an erasure for any PPM word that does not have exactly one slot whose soft output is higher than the threshold [13]. In either case, potentially helpful soft information is lost by a hard decision or erasure decision.

The turbo-decoder implementation described in this article uses the 256 soft outputs directly, and *it does not make any explicit decision regarding which slot contains a pulse.* Instead, an attempt is made to estimate the information bits themselves using an iterative decoder.

III. The Turbo Decoder

A. Soft-Input, Soft-Output (SISO) Structure

We use a turbo decoder structure and notation as in [2] and as shown in Fig. 7. Although this structure was designed with an additive white Gaussian noise (AWGN) channel in mind, part of the beauty of this

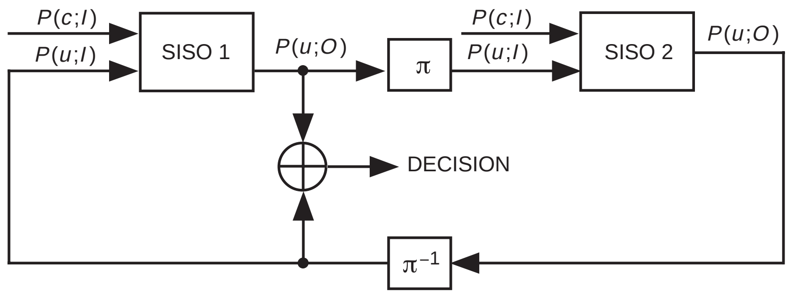

Fig. 7. The turbo decoder.

structure is that each soft-input, soft-output (SISO) module need not know anything about the channel in order to operate—its computation is determined completely by the a priori distribution $P(\mathbf{c}; I)$ and knowledge of the code structure.

B. Calculation of the A Priori Distribution

To properly initialize the turbo decoder, we must determine the a priori probability of codewords for each constituent code.[2] Let $\mathbf{c} = (c_0, c_1)$ denote a possible constituent-code codeword. As in [2], we define $P(\mathbf{c}; I) = \{P_k(\mathbf{c}; I)\}$, for $k = 1, \cdots, N + m$, where for SISO 1,

$$P_k(\mathbf{c}; I) = \Pr[\text{constituent codeword } (x_0, x_{1p}) \text{ is } \mathbf{c} \text{ at time } k | \text{ all soft counts from all symbols}] \tag{2}$$

and for SISO 2,

$$P_k(\mathbf{c}; I) = \Pr[\text{constituent codeword } (\pi(x_0), x_{2p}) \text{ is } \mathbf{c} \text{ at time } k | \text{ all soft counts from all symbols}] \tag{3}$$

Recall that $\pi(x_0)$ is not transmitted; computing $P_k(\mathbf{c}; I)$ requires interleaving the soft counts from the symbol containing x_0. Although there are a total of

$$\frac{n \text{ bits} \times 256 \text{ soft counts/PPM symbol}}{8 \text{ bits/PPM symbol}} = 8n$$

soft counts per block, all soft counts in the conditioning in Eqs. (2) and (3) are irrelevant to computing $P_k(\mathbf{c}; I)$ except those from PPM symbols that contain the kth constituent codeword bits. That is, a soft count does not give information about (x_0, x_{1p}) unless it is a soft count from a slot of a PPM symbol that depends on x_0 or x_{1p}.

To compute $P_k(\mathbf{c}; I)$, there are two cases to consider. We consider SISO 1 here; the analysis is identical for SISO 2, except that (x_o, x_{1p}) is replaced by $(\pi(x_0), x_{2p})$.

Case 1. The bits (x_0, x_{1p}) that form the kth constituent codeword are grouped into two separate PPM symbol epochs, s and t.

In this case, let x_0 be the lth bit of the sth PPM symbol; let x_{1p} be the mth bit of the tth PPM symbol; and let $\mathbf{y} = (y_0, \cdots, y_{255})$ and $\mathbf{z} = (z_0, \cdots, z_{255})$ denote the soft counts from the two transmitted PPM symbols. Then

$$\begin{aligned} P_k(\mathbf{c}; I) &= P_k((c_0, c_1); I) \\ &= P(\text{bit } l \text{ of } s\text{th PPM symbol is } c_0 | \mathbf{y}) \times P(\text{bit } m \text{ of } t\text{th PPM symbol is } c_1 | \mathbf{z}) \qquad (4) \\ &= \left(\sum_{\substack{\mathbf{a}=(a_1,\cdots,a_8) \\ a_l = c_0}} P(\mathbf{a}|\mathbf{y}) \right) \times \left(\sum_{\substack{\mathbf{a}=(a_1,\cdots,a_8) \\ a_m = c_1}} P(\mathbf{a}|\mathbf{z}) \right) \qquad (5) \end{aligned}$$

[2] By "a priori" probability, we mean that there is conditioning on the receiver decision statistics, but no conditioning on the turbo code structure, i.e., for the purposes of computing $P(\mathbf{c}; I)$, the components of $\mathbf{c}$ are assumed to be independent and uniformly distributed. This is the usual meaning in the context of turbo codes; in other contexts, conditioning on receiver statistics would be termed a posteriori.

where Eq. (4) follows by the independence of the soft counts from one PPM symbol to the next, and where in Eq. (5) $\mathbf{a}$ is the 8-bit vector defining a 256-PPM symbol. This is similar to the formulations given in [2, Section III.D] and [9, Section IV.A].

Case 2. The bits (x_0, x_{1p}) that form the kth constituent codeword are grouped into the same PPM symbol epoch s.

In this case, let x_0 be the lth bit of the sth PPM symbol; let x_{1p} be the mth bit of the sth PPM symbol; and let $\mathbf{y} = (y_0, \cdots, y_{255})$ denote the set of soft counts from the sth PPM symbol. Then

$$\begin{aligned} P_k(\mathbf{c}; I) &= P_k((c_0, c_1); I) \\ &= P(\text{bits } l \text{ and } m \text{ of } s\text{th PPM symbol are } c_0 \text{ and } c_1, \text{ respectively}|\mathbf{y}) \\ &= \sum_{\substack{\mathbf{a}=(a_1,\cdots,a_8) \\ a_l = c_0 \\ a_m = c_1}} P(\mathbf{a}|\mathbf{y}) \qquad (6) \end{aligned}$$

In either Case 1 or Case 2, we need to compute the probability that a given 256-PPM symbol $\mathbf{a} = (a_1, \cdots, a_8)$ was transmitted, given its corresponding set of 256 soft counts, $\mathbf{y} = (y_0, \cdots, y_{255})$. In the following, $p(\cdot)$ denotes the pdf of a continuous random variable or vector, and $P(\cdot)$ denotes the probability mass function (pmf) of a discrete random variable or vector. We have

$$\begin{aligned} P(\mathbf{a}|\mathbf{y}) &= \frac{p(\mathbf{y}|\mathbf{a})P(\mathbf{a})}{p(\mathbf{y})}, \text{ by Bayes rule} \\ &= \frac{p(\mathbf{y}|\mathbf{a})P(\mathbf{a})}{\sum_{i=0}^{255} p(\mathbf{y}|\mathbf{a}_i)P(\mathbf{a}_i)} \qquad (7) \\ &= \frac{p(\mathbf{y}|\mathbf{a})}{\sum_{i=0}^{255} p(\mathbf{y}|\mathbf{a}_i)}, \text{ since } P(\mathbf{a}_i) = 1/256 \text{ for all } i \qquad (8) \end{aligned}$$

where, in Eq. (7), we have summed over all possible 8-bit symbols, i.e., $\mathbf{a}_0 = (0, \cdots, 0)$, $\mathbf{a}_1 = (0, \cdots, 0, 1)$, and so on. Each of these symbols is a priori (i.e., without conditioning on soft counts or code structure) equiprobable.

Each 256-PPM symbol gives rise to one signaling slot and 255 non-signaling slots. By independence of the soft outputs, we may separate $p(\mathbf{y}|\mathbf{a})$ as the product of 255 non-signaling pdfs and 1 signaling pdf:

$$p(\mathbf{y}|\mathbf{a}) = p_S(y_\mathbf{a}) \prod_{i=0, i\neq \mathbf{a}}^{255} p_N(y_i) = \frac{p_S(y_\mathbf{a})}{p_N(y_\mathbf{a})} \prod_{i=0}^{255} p_N(y_i) \qquad (9)$$

where $p_S(\cdot)$ and $p_N(\cdot)$ are defined in Section II.D, and where we have used $\mathbf{a}$ as either an 8-bit vector or as an index equal to the number it represents in binary. Letting the likelihood function for the ith slot within the jth PPM symbol be denoted by $L_i^{(j)} = [p_S(y_i)]/[p_N(y_i)]$ (where it is understood that the soft counts y_i are from the jth PPM symbol), and plugging Eq. (9) into Eq. (8), it follows that, for the soft counts $\mathbf{y}$ from the sth PPM symbol,

$$P(\mathbf{a}|\mathbf{y}) = \frac{L_{\mathbf{a}}^{(s)} \prod_{i=0}^{255} p_N(y_i)}{\sum_{i=0}^{255}\left(L_i^{(s)} \prod_{k=0}^{255} p_N(y_k)\right)} = \frac{L_{\mathbf{a}}^{(s)}}{\sum_{i=0}^{255} L_i^{(s)}} \tag{10}$$

Thus, the a priori probability that a given 256-PPM symbol is **a**, given 256 soft outputs, is the ratio of the **a**th likelihood function to the sum of all likelihood functions of all symbol values. This may be used in Eqs. (5) and (6) to compute the a priori probabilities:

$$P_k(\mathbf{c};I) = P_k((c_0,c_1);I) = \begin{cases} \left(\dfrac{\sum\limits_{\substack{\mathbf{a}=(a_1,\cdots,a_8)\\ a_l=c_0}} L_{\mathbf{a}}^{(s)}}{\sum\limits_{i=0}^{255} L_i^{(s)}}\right) \times \left(\dfrac{\sum\limits_{\substack{\mathbf{a}=(a_1,\cdots,a_8)\\ a_m=c_1}} L_{\mathbf{a}}^{(t)}}{\sum\limits_{i=0}^{255} L_i^{(t)}}\right) & \text{if } c_0 \text{ and } c_1 \text{ are in different PPM symbols} \\ \dfrac{\sum\limits_{\substack{\mathbf{a}=(a_1,\cdots,a_8)\\ a_l=c_0,a_m=c_1}} L_{\mathbf{a}}^{(s)}}{\sum\limits_{i=0}^{255} L_i^{(s)}} & \text{if } c_0 \text{ and } c_1 \text{ are in the same PPM symbol} \end{cases} \tag{11}$$

It is important that Eq. (11) not be substantially harder to compute as compared with the usual hard decision of choosing the maximum soft output. The computation of the a priori likelihoods L_i is extremely fast with the use of a lookup table for $P_S(x)/P_N(x)$. Alternatively, L_i may be approximated by a Gaussian pdf of the appropriate mean and variance, with some loss in performance. Since the denominator of Eq. (11) does not depend on **a**, it need only be computed once per 256-PPM symbol.

C. Turbo-Decoder Iterations

Given the a priori probability distributions as calculated in the previous subsection, the turbo decoder operates in the usual way. We briefly review the computation made. Details not given here are contained in [1,2,4,10].

Computation of the extrinsic information for the information bit u at time k involves taking the sum, over all edges of the trellis that correspond to that value of u at time k, of the probability of the starting state of the edge times the probability of the edge according to the a priori distribution times the probability of the ending state of the edge. Following the notation of [2], the extrinsic information is given by

$$P_k(u;O) = H_u \sum_{e:u(e)=u} A_{k-1}[s^S(e)] P_k[c(e);I] B_k[s^E(e)] \tag{12}$$

where H_u is a normalization constant that forces $\sum_u P_k(u;O) = 1$, and where the forward recursion for the A's and backward recursion for the B's [1] are given by

$$A_k(s) = \sum_{e:s^E(e)=s} A_{k-1}[s^S(e)]\, P_k[u(e);I]\, P_k[c(e);I] \text{ for } k = 1,\cdots,n \tag{13}$$

$$B_k(s) = \sum_{e:s^S(e)=s} B_{k+1}[s^E(e)]\, P_{k+1}[u(e);I]\, P_{k+1}[c(e);I] \text{for } k = n-1,\cdots,0 \tag{14}$$

with initial conditions

$$A_0(s) = \begin{cases} 1 & s = S_0 \\ 0 & \text{otherwise} \end{cases}$$

and

$$B_n(s) = \begin{cases} 1 & s = S_n \\ 0 & \text{otherwise} \end{cases}$$

The turbo decoder block diagram is shown in Fig. 7. The steps in the operation of the turbo decoder are as follows:

(1) Compute the a priori information using the soft APD outputs and Eq. (11).

(2) Let $i \leftarrow 1$.

(3) For SISO i, compute the A's by the forward recursion in Eq. (13).

(4) For SISO i, compute the B's by the backward algorithm in Eq. (14).

(5) For SISO i, compute the extrinsic information $P(u;O)$ using Eq. (12).

(6) If ($i = 1$), then interleave the extrinsic information. Otherwise, deinterleave the extrinsic information.

(7) Let $i \leftarrow 3 - i$.

(8) Let iteration $\leftarrow$ iteration + 0.5.

(9) If (iteration < number of iterations to do), go to Step (3). Otherwise, go to Step (10).

(10) Add extrinsic output from SISO 1 to deinterleaved extrinsic output from SISO 2. Use zero threshold to determine bit values.

IV. Performance Simulations

Using the method of the previous section, simulations were conducted for turbo-coded 256-PPM and a typical APD detector, in a shot-noise-limited scenario. Table 1 lists the parameters that were used in the simulations. The parameters that varied include the code rate, the constituent-code generator polynomials and the corresponding number of decoding states, and the average number of absorbed signal photons. Metrics were computed using a multiplicative method, as opposed to an additive method using additional log() and exp() calls. As the errors occur in clusters, with most blocks containing no errors and some blocks containing many errors, the simulations were continued until 100 error-containing blocks were processed (with a maximum of 163 million bits processed per simulation). This usually implied the simulation of 10,000 or more bit errors.

RS performance was determined by simulating the soft counts using a program to generate Webb deviates, computing the resulting PPM symbol-error rate (SER) when the maximum-likelihood rule is used [14], and using a formula for the performance of RS code based on that SER [13]. Erasures were not considered.

Figures 8 and 9 show the performance of coded 256-PPM as a function of input SER for coding rates of 1/3 and 1/2, respectively. Parts (a) and (b) of each figure show the same simulation data in two different ways. Since the turbo decoder does not make an explicit determination of PPM symbols, the input SER technically is not defined for turbo codes; the SERs used in the plots are those that would have occurred had a maximum-likelihood rule been used to make PPM symbol decisions. The three curves for turbo codes represent the 4-state noninterleaved PPM, 4-state interleaved PPM, and 8-state interleaved

Table 1. Parameters used in turbo-coded PPM simulations.

Parameter	Value
Turbo encoder	
Overall code rate	1/3 or 1/2
Encoders	
Generator	5/7 or 17/15 (octal)
Constraint length	3 or 4, respectively
Interleaver length	16,382
Block size	16,384
Channel type	PPM
PPM modulator	
PPM order	256
Slot duration	2.000×10^{-8}
Optical channel	
Average signal photons/slot	20 to 30
Average background photons/slot	10
APD	
Gain	80
Ionization ratio	0.007
Noise temperature	300
Noise equivalent bandwidth	2.500×10^{7}
Load resistance	1.466×10^{5}
Extinction ratio	∞
Bulk leakage current	4.000×10^{-14}
Surface leakage current	2.000×10^{-9}
Turbo decoder	
Iterations	10
Metrics	Multiplicative method
Conventions	$\exp(-1000) = 0$
	$\exp(1000) = \mathrm{Inf}$
Maximum bits to process	163,820,000 (10,000 blocks)

PPM simulations. By noninterleaved PPM, we mean that the bits at the output of the turbo encoder are grouped into 8-bit blocks directly and PPM modulated. This order is

$$(x_0, x_{1p}, x_{2p}, x_0, x_{1p}, x_{2p}, \cdots, x_0, x_{1p}, x_{2p})$$

For the interleaved PPM, the order of the bits has been partially randomized as follows. Starting with a bit order of

$$(x_0, \cdots, x_0, x_{1p}, \cdots, x_{1p}, x_{2p}, \cdots, x_{2p})$$

the interleaver π—the same interleaver used in the turbo encoder itself—is used to permute each third of the block. That is,

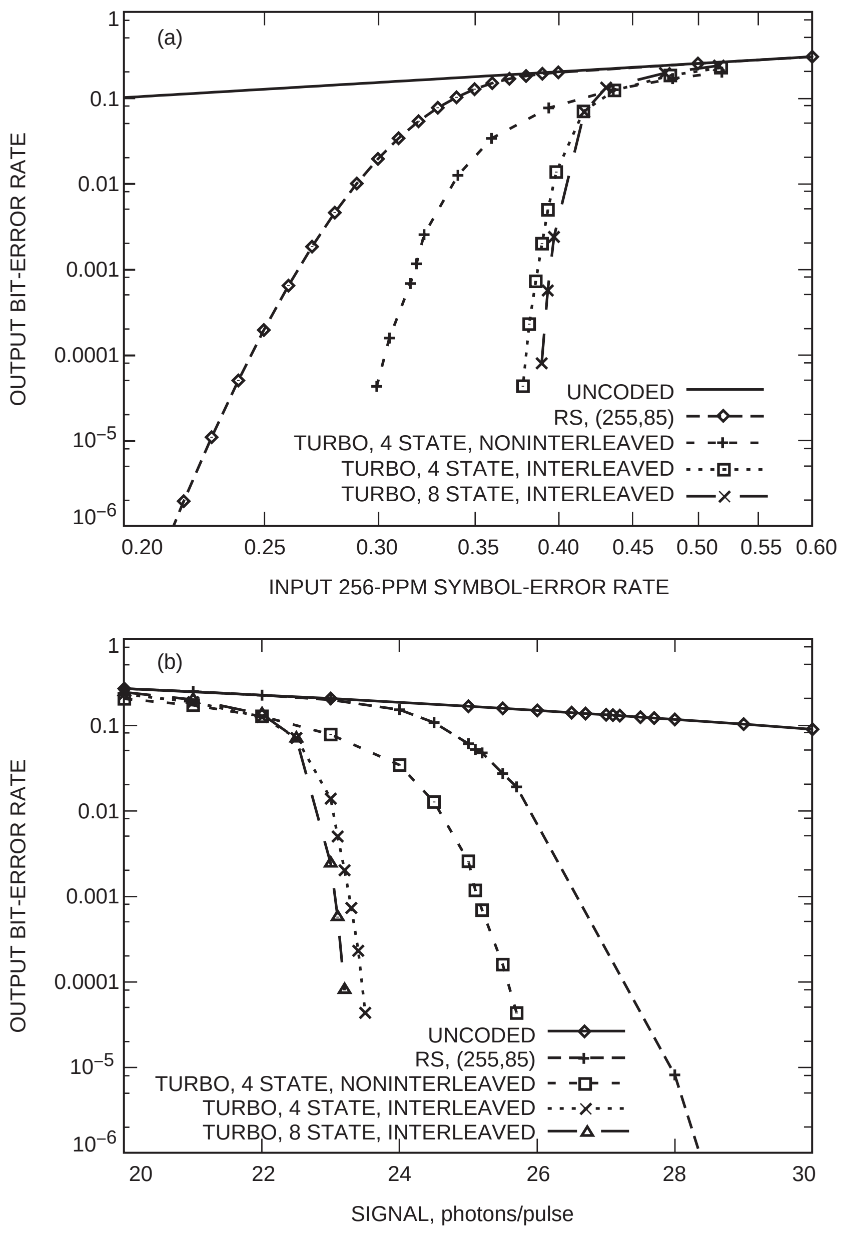

Fig. 8. The performance of rate-1/3 codes for 256-PPM with an APD detector as a function of (a) input SER and (b) signal photons per pulse.

$$(x_0, \cdots, x_0) \rightarrow (\pi(x_0), \cdots, \pi(x_0))$$

$$(x_{1p}, \cdots, x_{1p}) \rightarrow (\pi(x_{1p}), \cdots, \pi(x_{1p}))$$

$$(x_{2p}, \cdots, x_{2p}) \rightarrow (\pi(x_{2p}), \cdots, \pi(x_{2p}))$$

The effect of the interleaver before the modulator is to spread out related bits into different PPM symbols. Thus, when chance yields misleading soft information, the effect is spread across the entire block, instead of

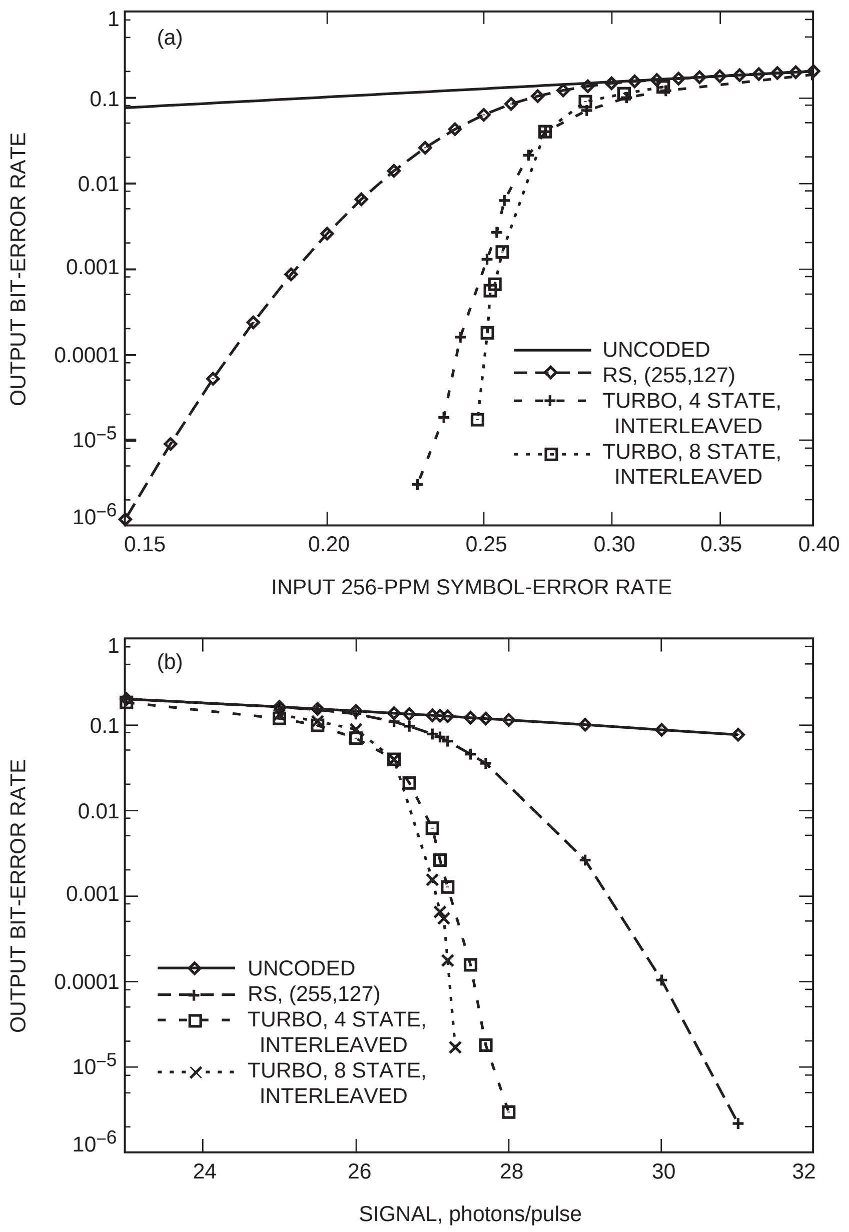

Fig. 9. The performance of rate-1/2 codes for 256-PPM with an APD detector as a function of (a) input SER and (b) signal photons per pulse.

clustered together. Reusing the same interleaver saved memory in running the simulations. Using a different, longer interleaver potentially could yield better results, but this was not investigated.

From Figs. 8 and 9, we may calculate the coding gain. Unlike the AWGN channel, whose performance can be characterized with a single SNR parameter, there are multiple parameters that affect performance on an optical channel. As such, the coding gain will depend on each of these parameters and no general statement can be made of the coding gain of one coded system over another. The coding gain must be stated as the gain for a particular choice of operating parameters.

We define the coding gain as the savings in signal photons per information bit for one system over another. At an output BER of 10^{-5} and using the parameters given in Table 1, the coding gain of the rate-1/3 binary turbo code is $10\log_{10}(27.3/23.3) = 0.7$ dB higher than the (256,85) RS code. The rate-1/2 binary turbo code has a coding gain $10\log_{10}(30.6/27.3) = 0.5$ dB above that of a (256,127) RS code.

V. Conclusions

For code rates of 1/2 and 1/3, binary turbo codes outperform the RS codes for PPM signals received by a soft APD detector. At an output BER of 10^{-5}, the improvement in coding gain is about 0.5 dB for rate-1/2 codes and 0.7 dB for rate-1/3 codes. While this article was limited to these two rates and specific operating parameters, we expect similar performance improvements for other code rates.

The computational complexity of the turbo-decoder implementation is modest, involving relatively simple 4- or 8-state trellises. The interleaver necessitates large storage capacity, but turbo-coding research is developing algorithmic methods to use interleavers; perhaps such a technique can be used in this application to alleviate the need to store the interleaver.

There are a number of areas for improvement in the turbo codes and in which future research is warranted:

(1) *Code rate optimization.* This article has made little attempt to determine the optimal code rate, because only rates 1/2 and 1/3 were considered. Overall, the rate-1/2 codes had better performance in terms of required photons per information bit. We expect that the optimal code rate is between 1/2 and 7/8.

(2) *Nonbinary turbo codes.* Nonbinary turbo codes can be expected to improve over the binary turbo codes presented here. There is a limit to the alphabet size because of computational complexity considerations, but 4-ary or 8-ary turbo decoders are still tractable and might significantly improve performance.

(3) *Modulator interleaver design.* We may optimize the design of the interleaver that feeds the modulator. No attempt was made to optimize it in this article.

(4) *Imperfect symbol synchronization or pulse spreading.* In these scenarios, the soft counts are likely to be smeared across multiple slots. If this behavior can be modeled mathematically, it is an ideal scenario in which to take full advantage of the soft counts using a turbo decoder. In addition, turbo codes can be combined with trellis-coded modulation.

Acknowledgments

The author acknowledges Meera Srinivasan for helpful discussions and for providing C code for generating Webb deviates, and Dariush Divsalar for helpful discussions about turbo codes.

References

[1] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Trans. Inform. Theory*, vol. IT-20, pp. 284–287, March 1974.

[2] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “A Soft-Input Soft-Output Maximum A Posteriori (MAP) Module to Decode Parallel and Serial Concatenated Codes,” *The Telecommunications and Data Acquisition Progress Report 42-127, July–September 1996*, Jet Propulsion Laboratory, Pasadena, California, pp. 1–20, November 15, 1996.
http://tmo.jpl.nasa.gov/tmo/progress_report/42-127/127H.pdf

[3] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, “Self-Concatenated Codes With Self-Iterative Decoding for Power and Bandwidth Efficiency,” *IEEE Int. Symp. Inform. Theory*, Massachusetts Institute of Technology, Cambridge, Massachusetts, p. 177, 1998.

[4] C. Berrou, A. Glavieux, and P. Thitimajshima, “Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes,” *Proc. IEEE Int. Conf. on Communications*, Geneva, Switzerland, pp. 1064–1070, May 1993.

[5] S. Benedetto and G. Montorsi, “Design of Parallel Concatenated Convolutional Codes,” *IEEE Trans. Commun.*, vol. 44, no. 5, pp. 591–600, May 1996.

[6] A. S. Barbulescu and S. S. Pietrobon, “Rate Compatible Turbo Codes,” *Electronics Letters*, vol. 31, no. 7, pp. 535–536, March 1995.

[7] C.-C. Chen, “Figure of Merit for Direct-Detection Optical Channels,” *The Telecommunications and Data Acquisition Progress Report 42-109, January–March 1992*, Jet Propulsion Laboratory, Pasadena, California, pp. 136–151, May 15, 1992.
http://tmo.jpl.nasa.gov/tmo/progress_report/42-109/109L.PDF

[8] D. Divsalar, R. M. Gagliardi, and J. H. Yuen, “PPM Performance for Reed–Solomon Decoding Over an Optical–RF Relay Link,” *IEEE Trans. Commun.*, vol. COM-32, no. 3, pp. 302–305, March 1984.

[9] D. Divsalar and F. Pollara, “On the Design of Turbo Codes,” *The Telecommunications and Data Acquisition Progress Report 42-123, July–September 1995*, Jet Propulsion Laboratory, Pasadena, California, pp. 99–121, November 15, 1995.
http://tmo.jpl.nasa.gov/tmo/progress_report/42-123/123D.pdf

[10] D. Divsalar and F. Pollara, “Turbo Codes for Deep-Space Communications,” *The Telecommunications and Data Acquisition Progress Report 42-120, October–December 1994*, Jet Propulsion Laboratory, Pasadena, California, pp. 29–39, February 15, 1995.
http://tmo.jpl.nasa.gov/tmo/progress_report/42-120/120D.pdf

[11] F. M. Davidson and X. Sun, “Gaussian Approximation Versus Nearly Exact Performance Analysis of Optical Communication Systems With PPM Signaling and APD Receivers,” *IEEE Trans. Commun.*, vol. 36, no. 11, pp. 1185–1192, November 1988.

[12] K. Kiasaleh, “Turbo-Coded Optical PPM Communication Systems,” *Journal of Lightwave Technology*, vol. 16, no. 1, pp. 18–26, January 1998.

[13] G. S. Mecherle, *Maximized Data Rate Capability for Optical Communication Using Semiconductor Devices With Pulse Position Modulation*, Ph.D. Thesis, University of Southern California, Los Angeles, California, May 1986.

[14] V. Vilnrotter, M. Simon, and M. Srinivasan, *Maximum Likelihood Detection of PPM Signals Governed by an Arbitrary Point Process Plus Additive Gaussian Noise*, JPL Publication 98-7, Jet Propulsion Laboratory, Pasadena, California, April 1998.

[15] P. P. Webb, R. J. McIntyre, and J. Conradi, "Properties of Avalanche Photodiodes," *RCA Review*, vol. 35, pp. 234–278, June 1974.

[16] J. Hamkins, "The Capacity of Avalanche Photodiode-Detected Pulse-Position Modulation," *The Telecommunications and Mission Operations Progress Report 42-138, April–June 1999*, Jet Propulsion Laboratory, Pasadena, California, pp. 1–19, August 15, 1999.
http://tmo.jpl.nasa.gov/tmo/progress_report/42-138/138A.pdf