Reduced Complexity Decoding of Coded Pulse-Position Modulation Using Partial Statistics

B. Moision[1] and J. Hamkins[1]

We show the storage requirements for soft-in soft-out (SISO) decoding of pulse-position modulation may be reduced by discarding a substantial subset of the slot likelihoods while suffering neglibible loss in performance in a serially concatenated, iteratively decoded scheme. We illustrate optimum processing of the remaining subset under a choose the maximum rule and show that a simplified trellis may be used for SISO decoding.

I. Introduction

The deep-space optical link operates efficiently at high peak-to-average power ratios [1,2], or low duty-cycles, which may be achieved by modulating the data using M-ary pulse-position modulation (PPM), in which $\log_2 M$ bits choose the location of a single pulsed slot in an M-slot symbol. For average-power-constrained links, efficient operation is typically achieved for large orders M. For example, the Mars Laser Communications Demonstration, a deep-space optical link that will fly on the Mars Telecommunications Orbiter in 2009, plans to use $M \in \{32, 64, 128\}$.[2] To efficiently achieve low bit-error rates, the modulation is concatenated with an error-correcting code. Near-capacity performance has been demonstrated for coded PPM by including soft-PPM demodulation in iterative decoding [3,4].

In the iterative soft-decision decoding process, the likelihood each slot contains a pulse is computed by the receiver and transmitted to the decoder, where it is stored for the duration of the iterative decoding process. However, storing these likelihoods can be prohibitively expensive for the large orders and short slot widths required by deep-space links. We show that the storage requirements can be reduced by discarding a substantial subset of the likelihoods while suffering virtually no loss in performance. We illustrate optimum processing of the remaining subset and show that a simplified trellis may be used for iterative decoding.

Two channel models will be considered: a Gaussian model, which may be used to model the output of an avalanche photodiode (APD) detector, and the Poisson model, which may be used to model photon-counting devices, such as a photomultiplier tube (PMT). We choose these models to simplify presentation

[1] Communications Architectures and Research Section.

[2] *MLCD Mars Lasercom Terminal To Ground Terminals Interface Control Document*, MLCD ICD2 (internal document), National Aeronautics and Space Administration, December 2004.

The research described in this publication was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

and facilitate comparisons with known results. More accurate models exist for the APD and PMT, such as the McIntyre–Conradi [5,6], Webb+Gaussian [7], and Gamma models [8]; however, the conclusions drawn from the Gaussian and Poisson model carry over to the more complex models, and we anticipate relative losses on the same order.

The article is organized as follows. In Section III, we derive channel likelihoods under a 'choose the maximum' rule. In Section IV, we show the performance of iterative decoding with partial statistics. In Section V, we quantify the complexity reduction and demonstrate that decoding may be accomplished with a time-varying, reduced-complexity trellis.

II. Preliminaries

The coded modulation system considered in this article is illustrated in Fig. 1. User data are encoded by the serial concatenation of outer code $\mathcal{C}_o$ and inner code $\mathcal{C}_i$ through a bit interleaver $\prod$. Throughout we use U, X, A, C, Y to denote random variables and u, x, a, c, y their realizations. We use $\mathbf{C}, \mathbf{Y}$ and $\mathbf{c}, \mathbf{y}$ to denote vectors of random variables and their realizations. We write $\mathbf{c} = (\mathbf{c}_1, \cdots, \mathbf{c}_N)$, where each $\mathbf{c}_k = (c_{k,1}, \cdots, c_{k,M})$ is a vector. We also write $\mathbf{I}$ to denote a random set of indices and let $\mathcal{I}$ denote a realization of $\mathbf{I}$.

The user input is $\mathbf{U} = (\mathbf{U}_1, \cdots, \mathbf{U}_K)$, a K-vector of binary p-vectors, i.e., a pK-bit vector. $\mathbf{U}$ is encoded by a rate-p/q outer code $\mathcal{C}_o$ to produce $\mathbf{X} = (\mathbf{X}_1, \cdots, \mathbf{X}_K)$, a K-vector of binary q-vectors. These qK bits are permuted using Π, yielding another qK-vector, $\mathbf{A} = (\mathbf{A}_1, \cdots, \mathbf{A}_K)$. We denote the length of the bit interleaver by $|\Pi| = qK$. $\mathbf{A}$ is then encoded by the inner code $\mathcal{C}_i$ consisting of a rate-1 code whose outputs are mapped to an M-ary PPM constellation. That is, $\mathbf{A}$ is encoded by $\mathcal{C}_i$ to yield $\mathbf{C} = (\mathbf{C}_1, \cdots, \mathbf{C}_N)$, where each $\mathbf{C}_k$, $1 \le k \le N$, is a binary M-vector with a single non-zero entry.

We refer to $\mathbf{C}$ as a *codeword*, to each $\mathbf{C}_k$ as a *PPM symbol*, and to each component of $\mathbf{C}_k = (C_{k,1}, \cdots, C_{k,M})$ as a *slot*. If $C_{k,j} = 1$, we say slot j of the kth PPM symbol is a *pulsed slot* or *signal slot*, and we write $\mathbf{C}_k = \mathbf{c}^{(j)}$ to denote the binary M-vector corresponding to this jth PPM symbol; if $C_{k,i} = 0$, it is a *nonpulsed slot* or *noise slot*. There are $N = |\Pi|/\log_2 M$ symbols, or $NM = |\Pi|M/\log_2 M$ slots per codeword.

This article considers two rate-1 codes as part of the inner code: the identity, and a binary accumulator, i.e., a $1/(1+D)$ mapping. We refer to the inner code that is formed by the concatenation of a binary accumulator and PPM mapping as accumulate-PPM (APPM). An APPM encoder can be described by a 2-state, $2M$-edge trellis, with each stage corresponding to one PPM symbol.

The kth transmitted coded PPM symbol, binary M-vector $\mathbf{C}_k = (C_{k,1}, \cdots, C_{k,M})$, is received as a noisy version $\mathbf{Y}_k = (Y_{k,1}, \cdots, Y_{k,M})$. We assume the $Y_{k,i}$ are conditionally independent given $\mathbf{C}_k$. We let $F_n(y)$ and $F_s(y)$ denote the conditional cumulative distribution functions of Y in noise and signal slots, respectively, evaluated at y, and let $f_n(y)$ and $f_s(y)$ denote the probability density (or mass) functions. That is, $F_n(y) = \text{Prob}(Y \le y | C = 0)$. We also define $F_n(y^-) = F_n(y) - \text{Prob}(Y = y | C = 0) = \text{Prob}(Y < y | C = 0)$, and define $F_s(y^-)$ similarly.

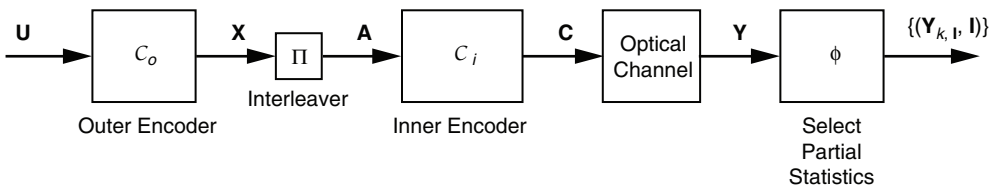

Fig. 1. The constrained storage channel model.

III. Partial Statistics

To realize the gains of iterative decoding algorithms nominally requires computation of a likelihood of the form

$$p_{\mathbf{Y}_k|\mathbf{C}_k}\left(\mathbf{y}_k|\mathbf{c}^{(j)}\right) = f_{Y|C}(y_{k,j}|1)\prod_{i=1,i\neq j}^{M} f_{Y|C}(y_{k,i}|0) = \frac{f_{Y|C}(y_{k,j}|1)}{f_{Y|C}(y_{k,j}|0)}\prod_{i=1}^{M} f_{Y|C}(y_{k,i}|0) = K\frac{f_{Y|C}(y_{k,j}|1)}{f_{Y|C}(y_{k,j}|0)} \quad (1)$$

where K does not depend on j. Thus, for each symbol the M slot-likelihood ratios are sufficient for computing the required conditional likelihoods. However, high data rates, large values of M, and large interleavers can make likelihood storage and processing prohibitively expensive. The set of channel likelihoods corresponding to one codeword requires $fM|\Pi|/\log_2(M)$ bits, where f is the number of bits used to represent fixed or floating-point values, typically 4–8. For example, with $M = 64, f = 6, |\Pi| = 16384$, the storage per codeword is 131 kilobytes. Multiple codewords may need to be stored concurrently to accommodate delays in iterative decoding and any parallelization of the decoding algorithm. For comparison, the storage required for each codeword of a hard-decision (n, k) Reed–Solomon decoder, which requires only the location of the slot with the maximum count per symbol, or $n\log_2 M$ bits, would be approximately 47 bytes per codeword, using the convention $n = M-1 = 63$. However, a Reed–Solomon decoder with these parameters requires 2–3 dB additional signal power [4].

To reduce the complexity of iterative decoding, in this section we describe a method to discard the majority of the channel likelihoods and to effectively operate the decoder using only the remainder. For example, in Section IV we illustrate an example of the case described above, where the storage required for one codeword of likelihoods may be reduced from 131 kilobytes to 38 kilobytes, and the number of operations to execute the forward–backward algorithm on the inner code, which comprises the majority of operations for iterative decoding, may be reduced by 29 percent, with negligible degradation in performance.

A. The Channel-Likelihood Selector Function, ϕ

Suppose P of the M elements of each PPM symbol $\mathbf{Y}_k$, as well as their indices $\mathbf{I}$, are made available to the receiver. As illustrated in Fig. 1, ϕ denotes the rule for choosing the P samples, $\phi : \mathbf{y}_k \to (\mathbf{y}_{k,\mathcal{I}}, \mathcal{I})$, $\mathcal{I} \subset \{1, \cdots, M\}, |\mathcal{I}| = P$, where $(\mathbf{y}_{k,\mathcal{I}}, \mathcal{I}) = \{(y_{k,j}, j)|j \in \mathcal{I}\}$ is the vector of retained samples and indices. For example, if ϕ is the rule 'choose the largest,' $P = 2$, and $\mathbf{y}_k = (0.9, 0.2, 0.8, 0.3, 0.1, 0.2, 0.0, 0.1)$, then $\phi(\mathbf{y}_k) = \{(0.9, 1), (0.8, 3)\}$, i.e., the largest two values of $\mathbf{y}_k$ are saved, along with their positions in $\mathbf{y}_k$. The mapping ϕ reduces the dimensionality of the received information. By allowing $\mathcal{I}$ to be a function of the observation $\mathbf{y}_k$, we will see that the reduction may be implemented with negligible performance loss.

Considering the decision rule as part of the channel, the optimal rule ϕ^* is the one that maximizes the channel capacity, i.e., the one that yields the most information about the codewords,

$$\phi^* \stackrel{\text{def}}{=} \underset{\phi, p_{\mathbf{C}}}{\operatorname{argmax}} I\big(\mathbf{C}; \phi(\mathbf{Y})\big) \quad (2)$$

where $I\big(\mathbf{C}; \phi(\mathbf{Y})\big)$ is the mutual information between $\mathbf{C}$ and $\phi(\mathbf{Y})$, and $p_{\mathbf{C}}$ is the distribution on $\mathbf{C}$. The optimization of the decision rule ϕ is not addressed in this article. We assume ϕ is fixed and address methods of utilizing the available information to estimate the transmitted data.

The conditional likelihoods

$$p_{\phi(\mathbf{Y}_k)|\mathbf{C}_k}(\phi(\mathbf{y}_k)|\mathbf{c}_k) = p_{\mathbf{Y}_{k,\mathcal{I}}|\mathbf{C}_{k,\mathcal{I}}}(\mathbf{y}_{k,\mathcal{I}}|\mathbf{c}_{k,\mathcal{I}})p_{\mathbf{I}|\mathbf{C}_k,\mathbf{Y}_{k,\mathcal{I}}}(\mathcal{I}|\mathbf{c}_k, \mathbf{y}_{k,\mathcal{I}}) \quad (3)$$

are a sufficient statistic for the maximum a posteriori estimation of $\mathbf{C}$ given $\phi(\mathbf{Y})$, and will serve as inputs to an iterative decoding algorithm. The term $p_{\mathbf{Y}_{k,\mathcal{I}}|\mathbf{C}_{k,\mathcal{I}}}$ is the likelihood that would result if the input were mapped to a P-dimensional constellation, and the term $p_{\mathbf{I}|\mathbf{C}_k,\mathbf{Y}_{k,\mathcal{I}}}$ is an adjustment to reflect the likelihood of the subset selection $\mathcal{I}$.

Choosing the maximum element is the maximum-likelihood symbol decision rule for several channels of interest [9]. As a heuristic extension, we let ϕ choose the P largest elements of $\mathbf{y}_k$.

For a continuous-output channel, we have

$$p_{\mathbf{I}|\mathbf{C}_k,\mathbf{Y}_{k,\mathcal{I}}}(\mathcal{I}|\mathbf{c}^{(j)},\mathbf{y}_{k,\mathcal{I}}) = \text{Prob}(\max \mathbf{Y}_{k,\bar{\mathcal{I}}} \leq \psi | \mathbf{C}_k = \mathbf{c}^{(j)}, \Psi = \psi)$$

where $\Psi = \min \mathbf{Y}_{k,\mathcal{I}}$, i.e., the minimum of the components of the vector $\mathbf{Y}_{k,\mathcal{I}}$, and $\bar{\mathcal{I}} = \{1,\cdots,M\}/\mathcal{I}$. Then

$$\begin{aligned}\text{Prob}(\max \mathbf{Y}_{k,\bar{\mathcal{I}}} \leq \psi | \mathbf{C}_k = \mathbf{c}^{(j)}) &= \begin{cases} F_n(\psi)^{M-P}, & j \in \mathcal{I} \\ F_n(\psi)^{M-P-1} F_s(\psi), & j \notin \mathcal{I} \end{cases} \\ &= F_n(\psi)^{M-P-1} F_s(\psi) \times \begin{cases} g(\psi), & j \in \mathcal{I} \\ 1, & j \notin \mathcal{I} \end{cases} \end{aligned} \tag{4}$$

where

$$g(\psi) \stackrel{\text{def}}{=} \frac{F_n(\psi)}{F_s(\psi)}$$

For a discrete-output channel, ties for the P largest elements may occur with positive probability. As shown in Appendix A, this results in the modified probability

$$p_{\mathbf{I}|\mathbf{C}_k,\mathbf{Y}_{k,\mathcal{I}}}(\mathcal{I}|\mathbf{c}^{(j)},\mathbf{y}_{k,\mathcal{I}}) = \sum_{t=0}^{M-P} \frac{1}{\binom{v+t}{t}} \binom{M-P}{t} F_n(\psi^-)^{M-P-t-1} F_s(\psi^-) f_n(\psi)^t \tag{5}$$

$$\times \begin{cases} g(\psi-), & j \in \mathcal{I} \\ \dfrac{M-P-t}{M-P} + \dfrac{t}{M-P} g(\psi^-) L(\psi), & j \notin \mathcal{I} \end{cases} \tag{6}$$

where $v = |\{y_{k,i} = \psi | i \in \mathcal{I}\}|$, the number of slots that tie the minimum. As an approximation, Eq. (4) may be used in place of Eq. (6), which has the effect of neglecting the probability of multiple slots of a symbol tying for the Pth largest value.

B. Poisson Channel

The output of a photon-counting detector, e.g., a photomultiplier tube, may be modeled as a Poisson process, with mean n_b in a noise slot and $n_s + n_b$ in a signal slot,

$$p_{Y|C}(k|1) = \frac{(n_s + n_b)^k e^{-(n_s+n_b)}}{k!}$$

$$p_{Y|C}(k|0) = \frac{n_b^k e^{-n_b}}{k!}$$

In the special case $n_b = 0$, the PPM channel reduces to an M-ary erasure channel with

$$p_{\mathbf{C}_k|\mathbf{Y}_k}\left(\mathbf{c}^{(j)}|\mathbf{y}_k\right) = \begin{cases} 1, & y_{k,j} > 0 \\ 1/M, & \mathbf{y}_k = \mathbf{0} \\ 0, & y_{k,i} > 0, i \neq j \end{cases}$$

$$p_{\mathbf{Y}_k|\mathbf{C}_k}\left(\mathbf{y}_k|\mathbf{c}^{(j)}\right) = M p_{\mathbf{Y}_k}(\mathbf{y}_k) p_{\mathbf{C}_k|\mathbf{Y}_k}\left(\mathbf{c}^{(j)}|\mathbf{y}_k\right)$$

Factoring out $p_{\mathbf{Y}_k}$, which is not a function of $\mathbf{c}^{(j)}$, the likelihoods depend only on whether photons are observed—not on their number. Hence we may always take $P = 1$ in zero background, where it is sufficient to store the location (or absence) of observed photons. In the remainder, we address the case $n_b > 0$.

For $n_b > 0$, a soft-decision demodulator would nominally compute and store likelihoods

$$p_{\mathbf{Y}_k|\mathbf{C}_k}\left(\mathbf{y}_k|\mathbf{c}^{(j)}\right) = \left(\prod_{i=1}^{M} p_{Y|C}(y_{k,i}|0)\right) e^{-n_s} \left(1 + \frac{n_s}{n_b}\right)^{y_{k,j}}$$

for each $\mathbf{c}^{(j)}$. With partial statistics, we have

$$\left.\begin{aligned} p_{\mathbf{Y}_{k,\mathcal{I}}|\mathbf{C}_{k,\mathcal{I}}}(y_{k,\mathcal{I}}|\mathbf{c}_{k,\mathcal{I}}) &= \prod_{i\in\mathcal{I}} p_{Y|C}(y_{k,i}|0) \times \begin{cases} 1, & j \notin \mathcal{I} \\ e^{-n_s}\left(1 + \frac{n_s}{n_b}\right)^{y_{k,j}}, & j \in \mathcal{I} \end{cases} \\ p_{\phi(\mathbf{Y}_k)|\mathbf{C}_k}(\phi(\mathbf{y}_k)|\mathbf{c}^{(j)}) &= \sum_{t=0}^{M-P} K_t \times \begin{cases} \frac{M-P-t}{M-P} + \frac{t}{M-P} g(\psi^-) L(\psi), & j \notin \mathcal{I} \\ e^{-n_s} g(\psi^-)\left(1 + \frac{n_s}{n_b}\right)^{y_{k,j}}, & j \in \mathcal{I} \end{cases} \\ g(\psi) &= \frac{\sum_{k=0}^{\psi} \frac{n_b^k}{k!}}{e^{-n_s} \sum_{k=0}^{\psi} \frac{(n_b+n_s)^k}{k!}} \end{aligned}\right\} \quad (7)$$

where K_t denotes a constant relative to $\mathbf{c}^{(j)}$. If we ignore the possibility of ties, we may use Eq. (4) to obtain an approximation for Eq. (7):

$$p_{\phi(\mathbf{Y}_k)|\mathbf{C}_k}\left(\phi(\mathbf{y}_k)|\mathbf{c}^{(j)}\right) \approx K \times \begin{cases} 1, & j \notin \mathcal{I} \\ e^{-n_s} g(\psi) \left(1 + \dfrac{n_s}{n_b}\right)^{y_{k,j}}, & j \in \mathcal{I} \end{cases} \tag{8}$$

A second approximation is arrived at by artificially forming a full statistics observation by assigning the missing observables $\mathbf{Y}_{k,\mathcal{I}}$ a stand-in value equal to the mean of the noise slot. In this way, Eq. (1) becomes

$$p_{\phi(\mathbf{Y}_k)|\mathbf{C}_k}\left(\phi(\mathbf{y}_k)|\mathbf{c}^{(j)}\right) \approx K \times \begin{cases} 1, & j \notin \mathcal{I} \\ \left(1 + \dfrac{n_s}{n_b}\right)^{y_{k,j} - n_b}, & j \in \mathcal{I} \end{cases} \tag{9}$$

C. Gaussian Channel

The output of an APD may be modeled as Gaussian when the number of signal photons incident on the detector is large [7], yielding, after normalization,

$$p_{Y|C}(y|1) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-(y-1)^2}{2\sigma^2}\right)$$

$$p_{Y|C}(y|0) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(\frac{-y^2}{2\sigma^2}\right)$$

On observation of $\mathbf{y}_k$, a soft-decision demodulator nominally computes the likelihoods

$$p_{\mathbf{Y}_k|\mathbf{C}_k}\left(\mathbf{y}_k|\mathbf{c}^{(j)}\right) = \frac{1}{(2\pi\sigma^2)^{M/2}} \exp\left(\frac{-1}{2\sigma^2} \sum_{i=1}^{M} y_{k,i}^2\right) \exp\left(\frac{2y_{k,j} - 1}{2\sigma^2}\right)$$

With partial statistics we have

$$\left.\begin{aligned} p_{\mathbf{Y}_{k,\mathcal{I}}|\mathbf{C}_{k,\mathcal{I}}}(\mathbf{y}_{k,\mathcal{I}}|\mathbf{c}_{k,\mathcal{I}}) &= \frac{1}{(2\pi\sigma^2)^{P/2}} \exp\left(\frac{-1}{2\sigma^2} \sum_{i \in \mathcal{I}} \mathbf{y}_{k,i}^2\right) \times \begin{cases} 1, & j \notin \mathcal{I} \\ \exp\left(\dfrac{2y_{k,j} - 1}{2\sigma^2}\right), & j \in \mathcal{I} \end{cases} \\ p_{\phi(\mathbf{Y}_k)|\mathbf{C}_k}\left(\phi(\mathbf{y}_k)|\mathbf{c}^{(j)}\right) &= K \times \begin{cases} 1, & j \notin \mathcal{I} \\ g(\psi) \exp\left(\dfrac{2y_{k,j} - 1}{2\sigma^2}\right), & j \in \mathcal{I} \end{cases} \\ g(\psi) &= \frac{\operatorname{erfc}\left(\dfrac{-\psi}{\sigma\sqrt{2}}\right)}{\operatorname{erfc}\left(\dfrac{-(\psi - 1)}{\sigma\sqrt{2}}\right)} \end{aligned}\right\} \tag{10}$$

where

$$\operatorname{erfc}(t) \stackrel{\text{def}}{=} \frac{2}{\sqrt{\pi}} \int_t^\infty e^{-y^2}\, dy$$

and K is constant relative to $\mathbf{c}^{(j)}$.

Computing likelihoods via Eq. (10) requires a table lookup or computation in order to determine $g(\psi)$. This may be eliminated by replacing $g(\psi)$ with an estimate independent of ψ. For example, we may replace $g(\psi)$ in Eq. (10) by its mean, $E\big(g(\Psi)\big)$:

$$p_{\phi(\mathbf{Y}_k)|\mathbf{C}_k}\left(\phi(\mathbf{y}_k)|\mathbf{c}^{(j)}\right) \approx K \times \begin{cases} 1, & j \notin \mathcal{I} \\ E\big(g(\psi)\big) \exp\left(\dfrac{2y_{k,j}-1}{2\sigma^2}\right), & j \in \mathcal{I} \end{cases} \tag{11}$$

Using this simplified likelihood requires pre-computation of $E\big(g(\Psi)\big)$. The distribution of Ψ for a PPM channel is derived in Appendix B.

IV. Iterative Decoding Performance

Figures 2 and 3 illustrate the bit- and word-error rates for a coded system with $M = 64$ on the Poisson channel with $n_b = 1$. Approximations (8) and (9) are used with variable number of statistics P. The outer code is the 4-state, rate-3/5, $d_{\text{free}} = 4$ convolutional code from [10], and the inner code is APPM. The interleaver is a 16410-bit spread interleaver, and iterations are stopped when the bit estimates from decoding the outer code form a codeword of the outer code. The error floor observed is due to this stopping rule. It may be removed by using another rule, e.g., [3]. We observe less than 0.05-dB loss relative to using full statistics with either approximation and $P \geq 8$.

Figures 4 and 5 illustrate bit-error rates for $M = 16$ and $M = 256$ on the Gaussian channel. Equation (10) is used for channel likelihoods. The outer code is the rate-1/2, 4-state code with generator matrix $G(D) = \left[1\ (1 + D^2)/(1 + D + D^2)\right]$. The inner code is APPM in each case. The interleaver is a 4096-bit randomly generated interleaver. Decoding uses fixed 8 iterations. We observe less than 0.05-dB loss relative to using full statistics with $P \geq 8$ in both cases.

Figure 6 illustrates results using Eqs. (10) and (11). The outer code in Fig. 6 is the rate-1/2, 4-state code with generator matrix $G(D) = \left[1\ (1 + D^2)/(1 + D + D^2)\right]$. The inner code is $M = 256$ PPM. The simulations use 8 iterations and a 4096-bit spread interleaver. We see negligible degradation using the approximate likelihood.

The ratio P/M at which we see negligible loss decreases with M. Hence, the utility of using partial statistics increases with M. We conjecture that the ratio P/M required for a fixed loss is also decreasing with the noise variance (in the limiting cases with $n_b = 0$ on the Poisson channel or $\sigma^2 = 0$ on the Gaussian channel, we may take $P = 1$).

V. Complexity

In Section III, we illustrated that a small fraction of the statistics may be used in iterative decoding while suffering a small performance degradation. In this section, we quantify the storage and complexity gains from using partial statistics.

On observation of $\mathbf{y}_k$, the P largest elements of $\mathbf{y}_k$ are found and their corresponding likelihoods computed and stored. The largest P elements may be found by scanning each one of the M observations

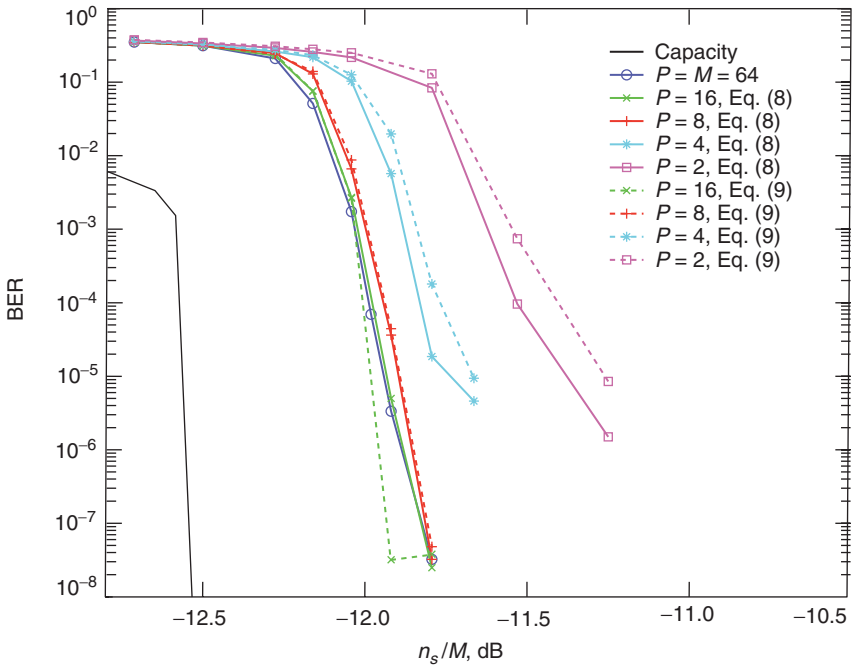

Fig. 2. Bit-error rate, Poisson channel, partial statistics using Eq. (8) (solid) or Eq. (9) (dashed), $n_b = 1$, $M = 64$ APPM, $P \in \{2, 4, 8, 16, 64\}$.

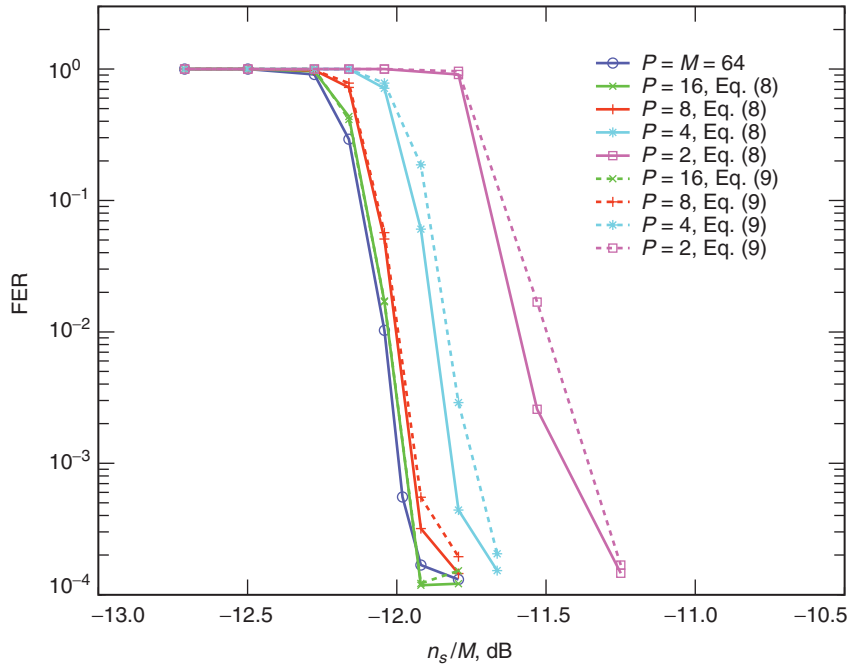

Fig. 3. Word-error rate, poisson channel, partial statistics using Eq. (8) (solid) or Eq. (9) (dashed), $n_b = 1$, $M = 64$ APPM, $P \in \{2, 4, 8, 16, 64\}$.

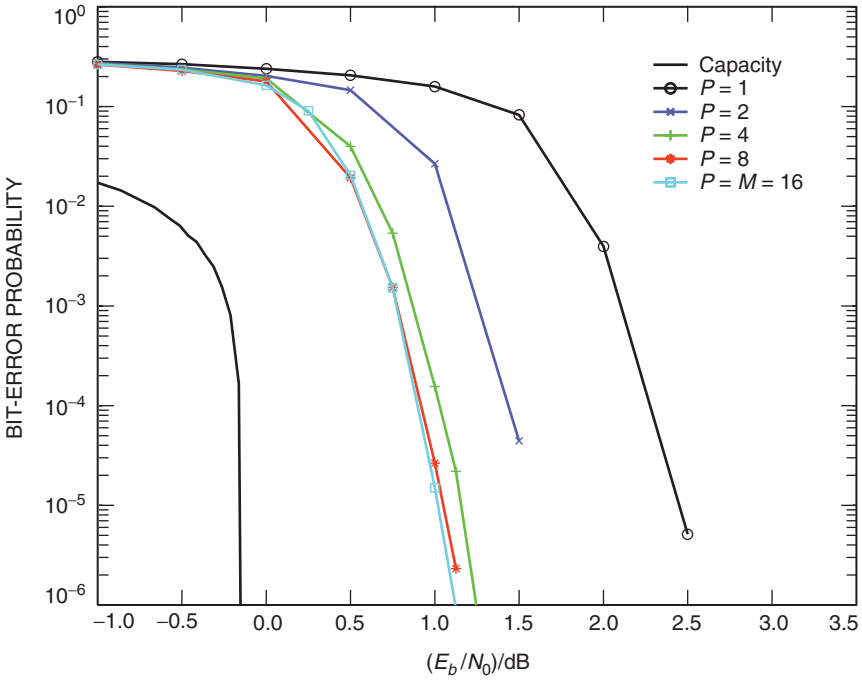

Fig. 4. Bit-error rate, Gaussian channel, $M = 16$ APPM, $P \in \{1, 2, 4, 8, 16\}$.

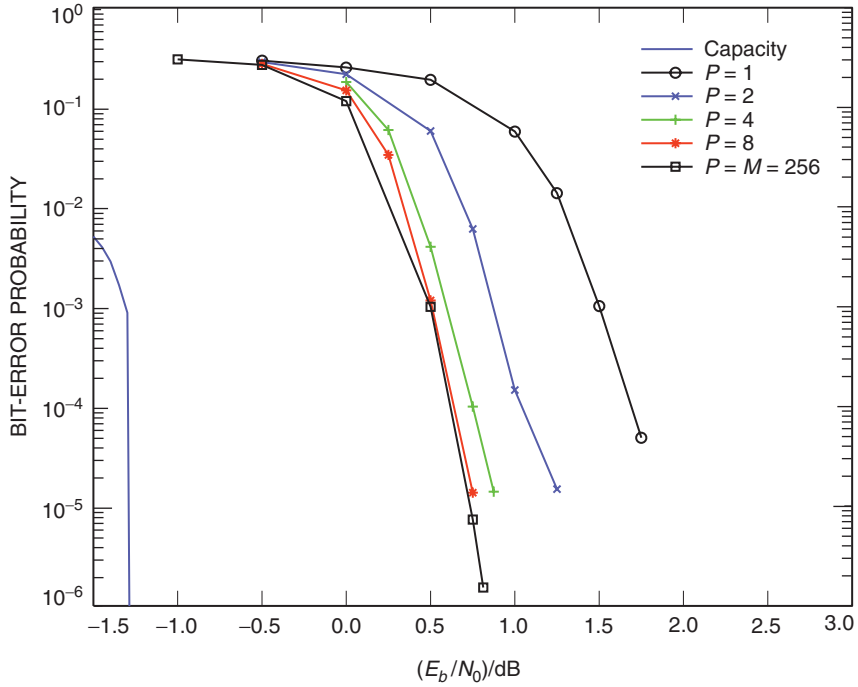

Fig. 5. Bit-error rate, Gaussian channel, $M = 256$ APPM, $P \in \{1, 2, 4, 8, 256\}$.

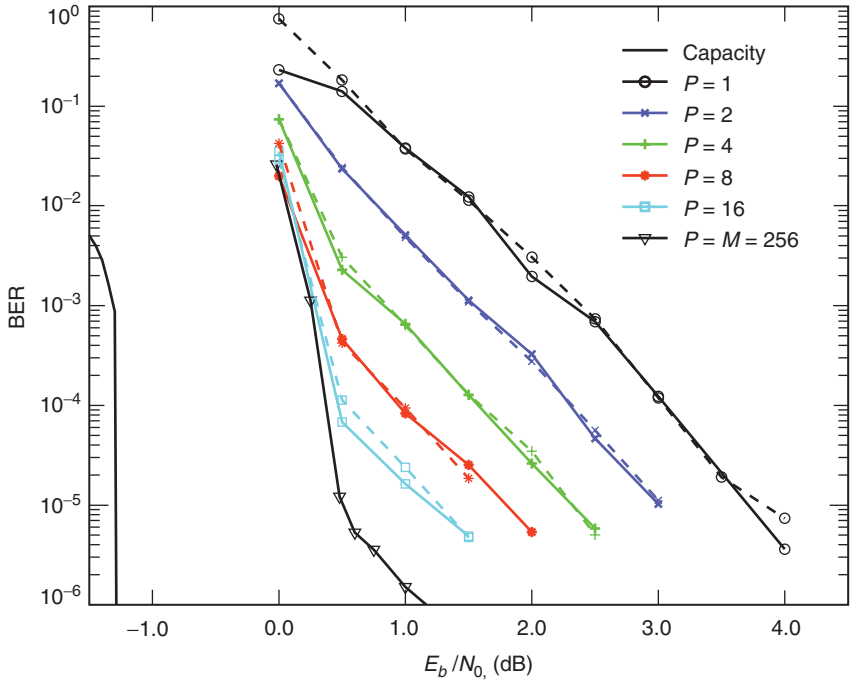

Fig. 6. Comparison of partial statistics using Eq. (10) (solid) or Eq. (11) (dashed) in Eq. (9), $M = 256$ APPM, $P \in \{1, 2, 4, 8, 16\}$.

and either discarding it or adding it to a heap of the largest elements observed thus far. With a heap size limited to P, this takes $O(M \log P)$ time using standard techniques [11]. Each likelihood computation requires one exponentiation and, in the Gaussian case, one multiplication. Only P likelihoods need be computed for partial statistics.

This yields a trade-off in sorting cost versus likelihood computation. A full assessment of this trade-off depends on the implementation of the algorithm and the platform. For example, if additions, multiplications, and exponentiations all require similar computation time, sorting costs will likely dominate. If exponentiations cost much more than multiplications, which cost more than additions, than likelihood computation will likely dominate. However, in either case, the net cost of decoding will be dominated by the implementation of the forward–backward algorithm, not the likelihood sorting and computation. The major gain of partial statistics lies in the reduced storage requirements.

Table 1 lists the storage required per received coded PPM symbol with full and partial statistics. Recall f denotes the number of bits used to represent likelihoods. In addition to storing the likelihoods, we must save the addresses of the P largest values, for which we assess a cost of $P \log_2 M$ bits.

Table 1. Channel-likelihood storage requirements.

Statistic method	Storage, bits	
	Address	Likelihood
Full	—	Mf
Partial	$P \log_2 M$	Pf

Accessing the values may proceed as follows. Let `edgeptr` be a pointer to the list of the indices of the P saved likelihoods, `likelihoodptr` a pointer to the corresponding likelihoods, and `default` $= K$, where K is defined in Eq. (8) or Eq. (10). Assume likelihoods for the kth trellis stage are accessed sequentially. The likelihood of the `e`th coded PPM symbol, `pci`, may be accessed by the following C code:

```
if (e==*edgeptr){
     pci = *likelihoodptr;
     edgeptr ++;
     likelihoodptr ++;
} else
     pci = default;
},
```

which we assume takes negligible additional time compared to that required to access likelihoods with full statistics.

The storage ratio for full versus partial likelihoods is

$$\frac{\text{storage for } P \text{ likelihoods}}{\text{storage for } M \text{ likelihoods}} = \frac{P}{M}\left(1 + \frac{\log_2 M}{f}\right)$$

illustrated in Fig. 7 for $f = 6$. Let $f = 6$ and consider the pairs (M, P) illustrated in Figs. 2, 4, and 5 for which we observe negligible performance loss. For $M = 256, P = 8$, partial statistics require 0.06 times the storage as full. For $M = 64, P = 8$, partial statistics require 0.25 times the storage as full. For $M = 16, P = 8$, partial statistics require 0.83 times the storage as full. These points are plotted in Fig. 7.

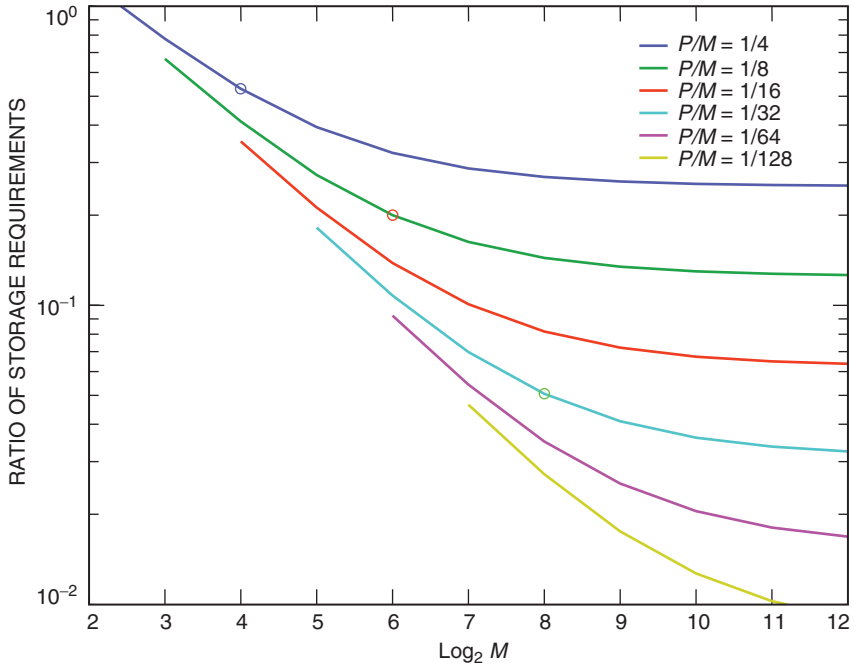

Fig. 7. Ratio of storage required for partial APPM trellis relative to full trellis.

A. Forward–Backward Algorithm, Full Statistics

In order to assess decoding complexity, we first describe an implementation of the forward–backward algorithm when all available statistics are used. Several more extensive discussions of the algorithm and its use in iterative decoding may be found in the literature, e.g., [12,13].

For either the inner or outer code, we formally describe the trellis by its set of states $\mathcal{V}$, and its set of directed, labeled edges $\mathcal{E}$. Each edge $e \in \mathcal{E}$ has an initial state $i(e)$, a terminal state $t(e)$, an input label $\mathbf{a}(e)$, and an output label $\mathbf{c}(e)$. Consider the inner code. Encoding proceeds by following a path through the graph and reading off the output edge labels as follows. Let s_{k-1} be the state at time $k-1$, and e the unique edge with $i(e) = s_{k-1}$ and $\mathbf{a}(e) = \mathbf{a}_k$. Then $\mathbf{e}_k = e$, $\mathbf{c}_k = \mathbf{c}(e)$, and $s_k = t(e)$. We use the notation $\mathbf{a}_{i:j} \stackrel{\text{def}}{=} (\mathbf{a}_i, \mathbf{a}_{i+1}, \cdots, \mathbf{a}_j)$ to denote a subsequence. For the outer code, the input is $\mathbf{u}$ and the output is $\mathbf{x}$, but in all other respects the trellis concept and notation are the same.

The symbol sequence $\mathbf{y} = (\mathbf{y}_1, \cdots, \mathbf{y}_N)$, a noisy version of $\mathbf{c}$, is observed. Given the observation $\mathbf{y}$, and an a priori estimate of $A_{k,i}$ via $p_{\mathbf{A}_{k,i}}(\cdot)$, we desire to compute maximum-likelihood estimates, or extrinsic information,

$$p_{\mathbf{Y}|A_{k,i}}(\mathbf{y}|a) = \frac{1}{p_{A_{k,i}}(a)} p_{A_{k,i},\mathbf{Y}}(a, \mathbf{y})$$

$$= \frac{1}{p_{A_{k,i}}(a)} \sum_{e \in \mathcal{E}: a(e)_i = a} p_{E_k,\mathbf{Y}}(e, \mathbf{y}) \tag{12}$$

which we will compute by determining

$$\lambda_k(e) \stackrel{\text{def}}{=} p_{E_k,\mathbf{Y}}(e, \mathbf{y}) \tag{13}$$

for each edge in the trellis. To this end, let

$$\alpha_k(s) \stackrel{\text{def}}{=} p_{S_k,\mathbf{Y}_{1:k}}(s, \mathbf{y}_{1:k})$$

$$\beta_k(s) \stackrel{\text{def}}{=} p_{\mathbf{Y}_{k+1:N}|S_k}(\mathbf{y}_{k+1:N}|s)$$

$$\gamma_k(e) \stackrel{\text{def}}{=} p_{E_k,\mathbf{Y}_k|S_{k-1}}(e, \mathbf{y}_k|s) \tag{14}$$

$$= p_{\mathbf{A}_k}\big(\mathbf{a}(e)\big) p_{\mathbf{Y}_k|\mathbf{C}_k}\big(\mathbf{y}_k|\mathbf{c}(e)\big) \tag{15}$$

and factor Eq. (13) as

$$\lambda_k(e) = \alpha_{k-1}\big(i(e)\big)\gamma_k(e)\beta_k\big(t(e)\big) \tag{16}$$

The following recursive equations are used to compute the α's and β's:

$$\alpha_k(s) = \sum_{e:t(e)=s} \alpha_{k-1}\big(i(e)\big)\gamma_k(e) \tag{17}$$

$$\beta_k(s) = \sum_{e:i(e)=s} \beta_{k+1}\big(t(e)\big)\gamma_{k+1}(e) \tag{18}$$

Table 2 lists the computation and storage complexity per trellis stage to execute one iteration of the forward–backward algorithm. We count each addition and multiplication as a single operation, and assess no storage cost for the $p_{\mathbf{Y}|A_{k,i}}$, to isolate the complexity of the forward–backward algorithm. Note that the product $\alpha_{k-1}\big(i(e)\big)\gamma_k(e)$ may be computed once for use in Eqs. (16) and (17). We assume the entire codeword of β_k's is computed and stored; the α_k's are computed and immediately discarded at each stage after being used to compute the λ_k. Hence, we assess no storage requirements for the α_k's, as only one stage is ever stored. Similarly, we assess no storage cost for $p_{\mathbf{A}_k}$ and λ_k, since these may be used once computed. Since $\mathbf{a}_k$ is a binary vector, we must form products for the input edge labels—$(\log_2 M)$-ary products in our case. We assume these are formed by taking one $(\log_2 M)$-ary product and finding the product that differs in one location via a multiplication and division.

Table 2. Computational complexity per trellis stage.

Computation	Operations	Storage, bits
$p_{\mathbf{A}_k}\big(\mathbf{a}(e)\big)$	$\log_2 M + 2M$	—
Eq. (15) $\gamma_k(e)$	$\lvert\mathcal{E}\rvert$	$f\lvert\mathcal{E}\rvert$
Eq. (18) $\beta_k(s)$	$2\lvert\mathcal{E}\rvert - \lvert\mathcal{V}\rvert$	$f\lvert\mathcal{V}\rvert$
Eq. (17) $\alpha_k(s)$	$2\lvert\mathcal{E}\rvert - \lvert\mathcal{V}\rvert$	—
Eq. (16) $\lambda_k(e)$	$\lvert\mathcal{E}\rvert$	—
Eq. (12) $p_{\mathbf{Y}\lvert A_{k,i}}(\mathbf{y}\lvert a)$	$\lvert\mathcal{E}\rvert \log_2 M$	—

B. Forward–Backward Algorithm, Partial Trellis

Each iteration, the forward–backward algorithm with partial statistics begins by computing the modified form of Eq. (15),

$$\gamma_k(e) = p_{\mathbf{A}_k}\big(\mathbf{a}(e)\big)p_{\phi(\mathbf{Y}_k)|\mathbf{C}_k}\big(\phi(\mathbf{y}_k)|\mathbf{c}(e)\big)$$

for each edge in the trellis, which changes each iteration as $p_{\mathbf{A}_k}(\cdot)$ is updated by the outer code. However, with partial statistics, for each k there are only $P+1$ distinct values of $p_{\phi(\mathbf{Y}_k)|\mathbf{C}_k}\big(\phi(\mathbf{y}_k)|\mathbf{c}(e)\big)$. One can take advantage of this and use a reduced-complexity time-varying trellis with $|\mathcal{V}|$ states and at most $|\mathcal{V}|(P+|\mathcal{V}|)$ edges. We'll show this may be used to reduce the number of operations in executing the algorithm, albeit at the cost of addressing a time-varying trellis.

Let $\mathcal{J}_k(r,s,\xi)$ be the collection of parallel edges with channel likelihood ξ,

$$\mathcal{J}_k(r,s,\xi) \stackrel{\text{def}}{=} \big\{e | i(e)=r, t(e)=s, p_{\phi(\mathbf{Y}_k)|\mathbf{C}_k}\big(\phi(\mathbf{y}_k)|\mathbf{c}(e)\big) = \xi\big\}$$

Form a *partial* trellis by replacing the edges in $\mathcal{J}_k(r,s,\xi)$ with a single edge $g(r,s,\xi)$ with initial state r and terminal state s. Let $\mathcal{E}'_k$ be the collection of modified edges. The kth stage of the partial trellis has

state set $\mathcal{V}$ and edge set $\mathcal{E}'_k$. For clarity, we will use g to denote edges in $\mathcal{E}'_k$ and e to denote edges in $\mathcal{E}$. We write $e \in g(r, s, \xi)$ or $e \in g$ as shorthand for $e \in \mathcal{J}_k(r, s, \xi)$.

With P statistics, the partial trellis will have $|\mathcal{E}'| \leq |\mathcal{V}|(P + |\mathcal{V}|)$. Each $g \in \mathcal{E}'_k$ is traversed if $\mathbf{a}_k \in \bigcup_{e \in g} a(e)$; hence we define

$$p_{\mathbf{A}_k}\big(\mathbf{a}(g)\big) \stackrel{\text{def}}{=} p_{G_k|S_{k-1}}\big(g|i(g)\big)$$

$$= \sum_{e \in g} p_{\mathbf{A}_k}\big(\mathbf{a}(e)\big) \tag{19}$$

and for $g(r, s, \xi)$ put, analogous to Eq. (14),

$$\gamma_k(g) \stackrel{\text{def}}{=} p_{G_k, \mathbf{Y}_k|S_{k-1}}(g, \mathbf{y}_k|r)\xi$$

$$= p_{\mathbf{A}_k}\big(\mathbf{a}(g)\big)\xi$$

We proceed to compute $\lambda_k(e)$ for each edge in the partial trellis. Note that the α's and β's are the same whether computed on the partial or full trellis (using partial statistics in both cases) and that for $e \in g$,

$$\gamma_k(e) = \gamma_k(g) \frac{p_{\mathbf{A}_k}\big(\mathbf{a}(e)\big)}{p_{\mathbf{A}_k}\big(\mathbf{a}(g)\big)}$$

hence,

$$\lambda_k(e) = \lambda_k(g) \frac{p_{\mathbf{A}_k}\big(\mathbf{a}(e)\big)}{p_{\mathbf{A}_k}\big(\mathbf{a}(g)\big)} \tag{20}$$

Hence, after computing the λ's on the partial trellis, we may compute the bit likelihoods $p_{\mathbf{Y}|A_{k,i}}$ as

$$p_{\mathbf{Y}|A_{k,i}}(\mathbf{y}|0) = \frac{1}{p_{A_{k,i}}(0)} \sum_{e: e \in \mathcal{E}, a_{k,i}(e)=0} \lambda_k(e)$$

$$= \frac{1}{p_{A_{k,i}}(0)} \sum_{g \in \mathcal{E}'_k} \frac{\lambda_k(g)}{p_{\mathbf{A}_k}\big(\mathbf{a}(g)\big)} \sum_{e \in g, \mathbf{a}_{k,i}(e)=0} p_{\mathbf{A}_k}\big(\mathbf{a}(e)\big) \tag{21}$$

There are at most $|\mathcal{V}|^2$ edges in $\mathcal{E}'_k$ with $|\mathcal{J}_k| > 1$. Hence, Eq. (21) requires no more than $|\mathcal{E}| \log_2 M + |\mathcal{V}|^2 + |\mathcal{V}|^2 \log_2 M$ operations (for each $e \in \mathcal{E}$ either $\lambda_k(e) = \lambda_k(g)$ or it belongs to one of the $|\mathcal{V}|^2$ sums corresponding to a g with $|\mathcal{J}_k| > 1$). Table 3 lists the computation and storage complexity to execute one iteration of the forward–backward algorithm on the partial trellis. The $p_{\mathbf{A}_k}\big(\mathbf{a}(e)\big)$ and $p_{\mathbf{A}_k}\big(\mathbf{a}(g)\big)$ are used at the start of the algorithm in Eqs. (19) and (15), and at the end in Eq. (21). Hence, we may trade off operations for storage. Here we assume the $p_{\mathbf{A}_k}\big(\mathbf{a}(e)\big)$ are recomputed but the $p_{\mathbf{A}_k}\big(\mathbf{a}(g)\big)$ are not, doubling the operational cost for $p_{\mathbf{A}_k}\big(\mathbf{a}(e)\big)$ and assessing no storage cost. In our motivating example, the code is APPM, so that $|\mathcal{V}| = 2$ and $|\mathcal{E}| = 2M$. Figures 7 and 8 illustrate the storage and

operational requirements per APPM trellis stage for the partial trellis relative to the full trellis for various ratios P/M.

Three points (M, P) are plotted: $(256, 8), (64, 8), (16, 4)$. They correspond to performance results illustrated in this article where we have observed negligible performance degradation. For $M = 256$, $P = 8, f = 6$, the partial trellis operates with 31 percent fewer operations and 95 percent less storage. For $M = 64, P = 8, f = 6$, the partial trellis operates with 29 percent fewer operations and 80 percent less storage. For $M = 16, P = 8, f = 6$, the partial trellis operates with 18 percent fewer operations and 47 percent less storage.

Table 3. Computation cost per partial trellis stage.

Computation		Operations	Storage, bits
	$p_{\mathbf{A}_k}\big(\mathbf{a}(e)\big)$	$2(\log_2 M + 2M)$	—
Eq. (19)	$p_{\mathbf{A}_k}\big(\mathbf{a}(g)\big)$	$\lvert\mathcal{V}\rvert(M - P)$	$f\lvert\mathcal{V}\rvert^2$
Eq. (15)	$\gamma_k(g)$	$\lvert\mathcal{E}'\rvert$	$f\lvert\mathcal{E}'\rvert$
Eq. (18)	$\beta_k(s)$	$2\lvert\mathcal{E}'\rvert - \lvert\mathcal{V}\rvert$	$f\lvert\mathcal{V}\rvert$
Eq. (17)	$\alpha_k(s)$	$2\lvert\mathcal{E}'\rvert - \lvert\mathcal{V}\rvert$	—
Eq. (13)	$\lambda_k(g)$	$\lvert\mathcal{E}'\rvert$	—
Eq. (21)	$p_{A_{k,i}\lvert\mathbf{Y}}$	$\lvert\mathcal{E}\rvert\log_2 M + (1 + \log_2 M)\lvert\mathcal{V}\rvert^2$	—

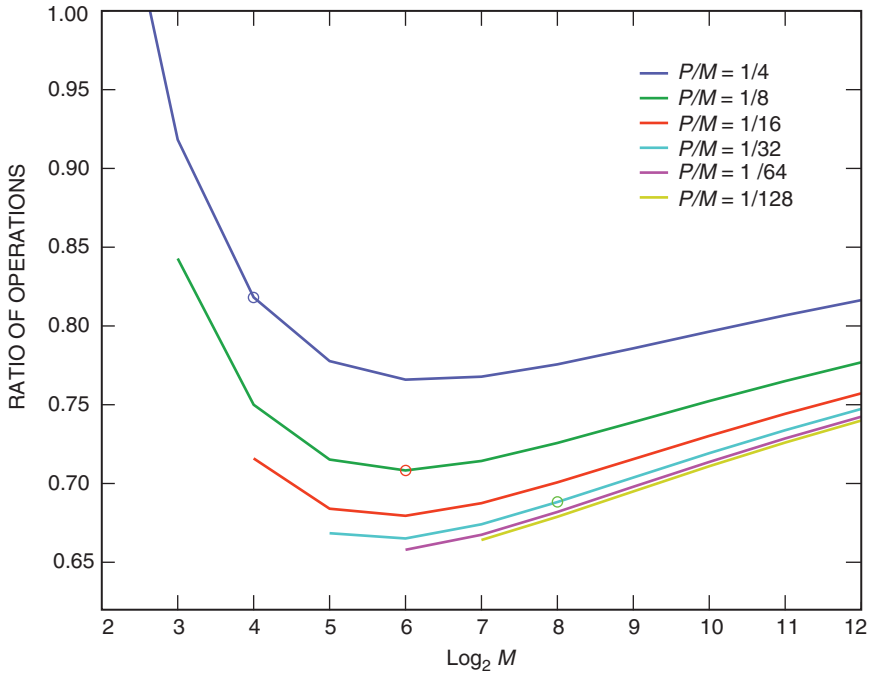

Fig. 8. Ratio of operations required for partial APPM trellis relative to full trellis.

An exact trade-off will depend on implementation details beyond the scope of this article, but we expect the complexity reduction due to the reduced number of edges will be on the same order. However, the reduced number of computations may be outweighed by the complexity of implementing a time-varying trellis.

VI. Conclusions

For coded modulation schemes that use a high PPM order and iterative demodulation, the storage required for the channel likelihoods and the complexity of decoding PPM may be a bottleneck in implementing iterative decoding. We have shown that a small subset of the channel likelihoods may be used with negligible degradation, by setting the remainder of the likelihoods to an appropriate constant. In addition, a reduced-complexity trellis may be used for the forward–backward algorithm. The reduced trellis requires fewer operations and storage, but must be time-varying.

The problem of generating soft information with partial statistics may be thought of as estimating the distances from the coded sequences to the received sequence when only P of the M coordinates of each $\mathbf{y}_k$ are known. We've observed that the loss relative to having knowledge of all coordinates may be kept negligibly small provided the coordinates may be chosen as a function of the observations.

References

[1] R. G. Lipes, "Pulse-Position-Modulation Coding as Near-Optimum Utilization of Photon Counting Channel with Bandwidth and Power Constraints," *The Deep Space Network Progress Report 42-56, January and February 1980*, Jet Propulsion Laboratory, Pasadena, California, pp. 108–113, April 15, 1980.
http://tmo.jpl.nasa.gov/tmo/progress_report2/42-56/56N.PDF

[2] A. D. Wyner, "Capacity and Error Exponent for the Direct Detection Photon Channel–Part I," *IEEE Transactions on Information Theory*, vol. 34, no. 6, pp. 1449–1461, November 1988.

[3] B. Moision and J. Hamkins, "Coded Modulation for the Deep-Space Optical Channel: Serially Concatenated Pulse-Position Modulation," *The Interplanetary Network Progress Report*, vol. 42-161, Jet Propulsion Laboratory, Pasadena, California, pp. 1–25, May 15, 2005.
http://ipnpr.jpl.nasa.gov/progress_report/42-161/161T.pdf

[4] B. Moision and J. Hamkins, "Deep-Space Optical Communications Downlink Budget: Modulation and Coding," *The Interplanetary Network Progress Report 42-154, April–June 2003*, Jet Propulsion Laboratory, Pasadena, California, pp. 1–28, August 15, 2003.
http://ipnpr.jpl.nasa.gov/tmo/progress_report/42-154/154K.pdf

[5] R. J. McIntyre, "The Distribution of Gains in Uniformly Multiplying Avalanche Photodiodes: Theory," *IEEE Transactions on Electron Devices*, vol. ED–19, no. 6, pp. 703–713, June 1972.

[6] J. Conradi, "The Distribution of Gains in Uniformly Multiplying Avalanche Photodiodes: Experimental," *IEEE Transactions on Electron. Devices*, vol. ED–19, no. 6, pp. 713–718, June 1972.

[7] S. Dolinar, D. Divsalar, J. Hamkins, and F. Pollara, "Capacity of Pulse-Position Modulation (PPM) on Gaussian and Webb Channels," *The Telecommunications and Mission Operations Progress Report 42-142, April–June 2000*, Jet Propulsion Laboratory, Pasadena, California, pp. 1–31, August 15, 2000. http://tmo.jpl.nasa.gov/tmo/progress_report/42-142/142H.pdf

[8] H. H. Tan, "A Statistical Model of the Photomultiplier Gain Process with Applications to Optical Pulse Detection," *The Telecommunications and Data Acquisition Progress Report 42-68, January and February 1982*, Jet Propulsion Laboratory, Pasadena, California, pp. 55–67, April 15, 1982. http://tmo.jpl.nasa.gov/tmo/progress_report/42-68/68H.PDF

[9] V. Vilnrotter, M. Simon, and M. Srinivasan, "Maximum Likelihood Detection of PPM Signals Governed by Arbitrary Point-Process Plus Additive Gaussian Noise," *IEE Electronics Letters*, vol. 35, no. 14, pp. 1132–1133, July 1999.

[10] D. G. Daut, J. W. Modestino, and L. D. Wismer, "New Short Constraint Length Convolutional Code Constructions for Selected Rational Rates," *IEEE Transactions on Information Theory*, vol. IT-28, no. 5, pp. 794–800, September 1982.

[11] W. H. Press, S. A. Teukolsky, W. T. Vetterling, and B. P. Flannery, *Numerical Recipes in C*, 2nd. edition, New York: Cambridge University Press, 1992.

[12] 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

[13] L. R. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal Decoding of Linear Codes for Minimizing Symbol Error Rate," *IEEE Transactions on Information Theory*, vol. 20, pp. 284–287, March 1974.

Appendix A
The Probability of Index Selection on the Discrete-Output Channel

Given $\mathbf{c}_k$ was transmitted, and a subset $\mathbf{y}_{k,\mathcal{I}}$ of the received vector is received, the probability that subset $\mathcal{I}$ corresponds to the selected index set under a 'choose the maximum' rule is computed here. We define

$$\Psi \stackrel{\text{def}}{=} \min_{i\in\mathcal{I}} Y_{k,i}$$

$$V \stackrel{\text{def}}{=} |\{Y_{k,i} = \psi : i \in \mathcal{I}\}|$$

$$T \stackrel{\text{def}}{=} |\{Y_{k,i} = \psi : i \notin \mathcal{I}\}$$

We have

$$p_{\mathbf{I}|\mathbf{C}_k,\mathbf{Y}_{k,\mathcal{I}}}(\mathcal{I}|\mathbf{c}^{(j)},\mathbf{y}_{k,\mathcal{I}}) = \text{Prob}(I=\mathcal{I},|\mathbf{C}_k=\mathbf{c}^{(j)},\Psi=\psi,V=v) \tag{A-1}$$

$$= \text{Prob}(I=\mathcal{I},\max\mathbf{Y}_{k,\bar{\mathcal{I}}}\le\psi|\mathbf{C}_k=\mathbf{c}^{(j)},\Psi=\psi,V=v) \tag{A-2}$$

$$= \sum_{t=0}^{M-P}\text{Prob}(I=\mathcal{I},\max\mathbf{Y}_{k,\bar{\mathcal{I}}}\le\psi,T=t|\mathbf{C}_k=\mathbf{c}^{(j)},\Psi=\psi,V=v) \tag{A-3}$$

In Eq. (A-1), we used the fact that $\mathbf{y}_{k,\mathcal{I}}$ affects the probability only insofar as it defines Ψ and V; in Eq. (A-2), we included an event that is necessarily satisfied if $\mathcal{I}$ is selected; and in Eq. (A-3), we sum the joint probability over all possibilities of T.

When $j \in \mathcal{I}$, each term of Eq. (A-3) denotes the conditional probability that exactly t noise slots take on value ψ, that the other $M-P-t$ are strictly lower, and that index set $\mathcal{I}$ is chosen, given that the signal and nonsignal slot positions are known, that the smallest slot in $\mathcal{I}$ takes on value ψ, and that v slots in $\mathcal{I}$ have value ψ. The probability of the correct index set under these conditions is $1/\binom{v+t}{t}$, and so we may rewrite Eq. (A-3) as

$$p_{\mathbf{I}|\mathbf{C}_k,\mathbf{Y}_{k,\mathcal{I}}}(\mathcal{I}|\mathbf{c}^{(j)},\mathbf{y}_{k,\mathcal{I}}) = \sum_{t=0}^{M-P}\frac{1}{\binom{v+t}{t}}\binom{M-P}{t}F_n(\psi^-)^{M-P-t}f_n(\psi)^t \tag{A-4}$$

When $j \notin \mathcal{I}$, we consider the events $y_{k,j} < \psi$ and $y_{k,j} = \psi$ separately, to obtain

$$p_{\mathbf{I}|\mathbf{C}_k,\mathbf{Y}_{k,\mathcal{I}}}(\mathcal{I}|\mathbf{c}^{(j)},\mathbf{y}_{k,\mathcal{I}}) = \sum_{t=0}^{M-P}\Big[\text{Prob}(\mathcal{I},\max\mathbf{Y}_{k,\bar{\mathcal{I}}}\le\psi,T=t,y_{k,j}<\psi|\mathbf{C}_k=\mathbf{c}^{(j)},\Psi=\psi,V=v)$$

$$
\begin{aligned}
&\quad + \text{Prob}(\mathcal{I}, \max \mathbf{Y}_{k,\bar{\mathcal{I}}} \le \psi, T = t, y_{k,j} = \psi | \mathbf{C}_k = \mathbf{c}^{(j)}, \Psi = \psi, V = v) \Big] \\
&= \sum_{t=0}^{M-P} \frac{1}{\binom{v+t}{t}} \left[\binom{M-P-1}{t} F_n(\psi^-)^{M-P-t-1} F_s(\psi^-) f_n(\psi)^t \right. \\
&\quad \left. + I_{\{t \ge 1\}} \binom{M-P-1}{t-1} F_n(\psi^-)^{M-P-t} f_s(\psi) f_n(\psi)^{t-1} \right] \\
&= \sum_{t=0}^{M-P} \frac{1}{\binom{v+t}{t}} \binom{M-P}{t} F_n(\psi^-)^{M-P-t-1} F_s(\psi^-) f_n(\psi)^t \\
&\quad \times \left(\frac{M-P-t}{M-P} + \frac{t}{M-P} g(\psi^-) L(\psi) \right) \qquad \text{(A-5)}
\end{aligned}
$$

where $L(\psi) \stackrel{\text{def}}{=} f_s(\psi)/f_n(\psi)$, and we have used the relation

$$
\binom{M-P}{t} = \frac{M-P}{M-P-t} \cdot \binom{M-P-1}{t} = \frac{M-P}{t} \cdot \binom{M-P-1}{t-1}
$$

Putting Eqs. (A-4) and (A-5) together, we obtain

$$
p_{\mathbf{I}|\mathbf{C}_k, \mathbf{Y}_{k,\mathcal{I}}} \left(\mathcal{I} | \mathbf{c}^{(j)}, \mathbf{y}_{k,\mathcal{I}} \right) = \sum_{t=0}^{M-P} \frac{1}{\binom{v+t}{t}} \binom{M-P}{t} F_n(\psi^-)^{M-P-t-1} F_s(\psi^-) f_n(\psi)^t \qquad \text{(A-6)}
$$

$$
\times \begin{cases} g(\psi-), & j \in \mathcal{I} \\ \dfrac{M-P-t}{M-P} + \dfrac{t}{M-P} g(\psi^-) L(\psi), & j \notin \mathcal{I} \end{cases} \qquad \text{(A-7)}
$$

As a check, note that the $t = 0$ term of Eq. (A-7) reduces to an analogue of Eq. (4).

Appendix B

Distribution of $\Psi = \min\{\mathbf{Y}_j | j \in \mathcal{I}\}$

Let Ψ be the Pth largest element of the set $\mathbf{Y} = \{N_1, N_2, \cdots, N_{M-1}, S\}$, where the N_i are independent and identically distributed (IID) with density $p_N(n)$ and distribution $F_N(n)$, and S, the signal slot, has density $p_S(s)$ and distribution $F_S(s)$. We desire the density of Ψ. The distribution is

$$
\begin{aligned}
F_\Psi(\psi) &= \text{Prob}[P\text{th largest element of } \mathbf{Y} \le \psi] \\
&= \sum_{i=0}^{P-1} \text{Prob}[\text{exactly } i \text{ elements of } \mathbf{Y} \text{ are greater than } \psi] \\
&= \sum_{i=0}^{P-1} \binom{M-1}{i-1} F_N(\psi)^{M-i} \big(1 - F_N(\psi)\big)^{i-1} \big(1 - F_S(\psi)\big) \\
&\quad + \binom{M-1}{i} F_N(\psi)^{M-i-1} \big(1 - F_N(\psi)\big)^{i} F_S(\psi) \qquad \text{(B-1)}
\end{aligned}
$$

The density follows directly by differentiating Eq. (B-1).