A Generalized Pre-Processor for Block and Convolutionally Coded Signals

V. Vilnrotter,[1] C. Lee,[1] and N. Lay[1]

The concept of a generalized data pre-processor for preliminary estimates of block and convolutionally encoded symbols is described and evaluated. Preliminary data estimates are envisioned to have application in removing binary phase-shift-keyed (BPSK) modulated data from the received carrier prior to phase estimation, particularly in low signal-to-noise ratio (SNR) environments where squaring losses impose a limit on estimator performance. Reception of weak signals from distant spacecraft, emergency-mode communications, and signals generated by small antenna elements of a larger array provide practical examples of where this technique can be used to improve overall system performance. Examples illustrating the potential improvements in carrier recovery are evaluated using both analysis and system-level simulation. The simulations include realistic block and convolutionally coded signals of the type often encountered in practice, such as extended Hamming and extended Bose–Chaudhuri–Hocquenghem (BCH) codes, along with short constraint-length convolutional codes generally employed in turbo codes. It is shown that received signal power can be reduced by as much as 6 dB when this technique is employed, implying a similar reduction in required transmitter power or antenna area on the spacecraft.

I. Introduction

The novel concept presented and evaluated in this article addresses the phase-estimation problem under conditions of very low signal-to-noise ratio (SNR) characteristic of distant spacecraft, emergency-mode communications, and reception of turbo-coded signals where symbol SNRs of -6 dB are routinely encountered. This technique may also provide significantly improved phase estimates for the individual elements of a large antenna array, such as the 400-element array of 5-meter antennas recently proposed for the DSN, where symbol SNRs as low as -31 dB may be encountered.

The concept of an information-reduced maximum a posteriori (MAP) phase estimator for binary phase-shift-keyed (BPSK) modulated signals has been presented in a previous article [1], where it was shown that dramatic reductions in squaring-loss performance can be achieved by partially removing the data modulation from the carrier before attempting to estimate the phase. With this approach, a partially reconstructed carrier forms the input to a MAP carrier phase estimation loop designed to accommodate

[1] Communications Systems and Research Section.

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.

arbitrary data transition densities. The performance of this coupled system consisting of a pre-processor and a phase estimator has been evaluated for the case of uncoded and block-orthogonal coded signals in [2], where improvements of approximately 5 dB in squaring loss have been demonstrated at symbol SNRs as low as −10 dB. Here we extend these earlier results to algebraic block codes and convolutional codes in order to determine system performance when realistic codes are observed.

The system concept is illustrated in Fig. 1. First the data sequence is estimated using a symbol-sequence estimator or data pre-processor, and then the estimated data sequence $\hat{m}(t-\Delta)$ is multiplied with a delayed version of the sampled waveform. Even if the data estimates contain some errors, the original data modulation will be largely removed, leaving only a residual error sequence on the carrier with a necessarily lower transition rate than the original signal. This partially reconstructed carrier forms the input to a MAP phase estimator whose structure depends on the transition rate at its input, as described in [1,2]: with equal transition probabilities, the structure reduces to the familiar hyperbolic tangent loop, whereas if the transition probability approaches zero, the in-phase arm in the MAP phase estimator is completely eliminated, resulting in a conventional phase-locked loop. The MAP phase estimator supplies the pre-processor with an estimate of phase, $\hat{\theta}$, whereas the pre-processor provides an estimate of the residual transition rate, $\hat{p}$, to the MAP carrier-tracking loop. Thus, it was shown that ideal phase-locked loop performance can be achieved even at low symbol SNRs if the effects of the data modulation can be removed.

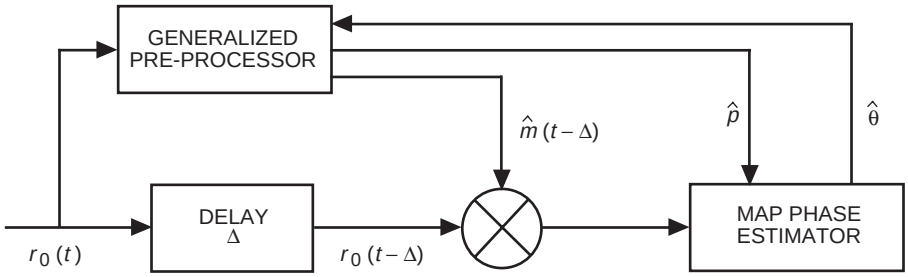

Fig. 1. Block diagram of the coupled data pre-processor MAP phase-estimation concept.

II. Signal and Noise Models

Consider a modulated sinusoidal signal received in the presence of additive narrowband noise,

$$r_0(t) = \sqrt{2P}m(t)\cos\big(\omega_0 t + \theta(t)\big) + n(t) \tag{1}$$

where $m(t)$ represents amplitude modulation, $\theta(t)$ is a slowly varying random phase process, and $\omega_0 = 2\pi f_0$, where f_0 is the carrier frequency. The narrowband representation of the additive noise waveform is

$$n(t) = \sqrt{2}\,n_c(t)\cos(\omega_0 t\,) - \sqrt{2}\,n_s(t)\sin(\omega_0 t\,) \tag{2}$$

where $n_c(t)$ and $n_s(t)$ are statistically independent, stationary Gaussian random processes, each with two-sided spectral level $S_n(f) = N_0/2$ and single-sided bandwidth W assumed to be much less than f_0. Defining

$$\left.\begin{aligned} N_c(t) &= n_c(t)\cos\theta(t) + n_s(t)\sin\theta(t) \\ N_s(t) &= -n_c(t)\sin\theta(t) + n_s(t)\cos\theta(t) \end{aligned}\right\} \tag{3}$$

and using simple trigonometric identities, Eq. (2) can be rewritten in a form where the total noise has been decomposed into a component that is in phase with the signal and phase-noise-modulated carrier, plus a component in quadrature with it:

$$n(t) = \sqrt{2}\, N_c(t) \cos\big(\omega_0 t + \theta(t)\big) - \sqrt{2}\, N_s(t) \sin\big(\omega_0 t + \theta(t)\big) \tag{4}$$

Note that this transformation does not change the statistics of the noise provided the phase process remains constant over the observation interval.

Expanding the sinusoidal terms in Eq. (3) as complex exponentials, Eq. (1) can be rewritten as

$$\tilde{r}_0(t) = \sqrt{2P} m(t) \frac{e^{j(\omega_0 t + \theta(t))} + e^{-j(\omega_0 t + \theta(t))}}{2} + \tilde{n}(t) \tag{5a}$$

where now

$$\tilde{n}(t) = \sqrt{2}\, N_c(t) \frac{e^{j(\omega_0 t + \theta(t))} + e^{-j(\omega_0 t + \theta(t))}}{2} - \sqrt{2}\, N_s(t)\, \frac{e^{j(\omega_0 t + \theta(t))} - e^{-j(\omega_0 t + \theta(t))}}{2j} \tag{5b}$$

Let $\hat{\theta}(t)$ be an estimate of the phase process $\theta(t)$. When this phase estimate is applied to a local oscillator (LO), the resulting complex function can be represented as

$$\tilde{s}_{\mathrm{LO}}(t) \equiv \sqrt{2}\, e^{-j(\omega_0 t + \hat{\theta}(t))} \tag{6}$$

The received waveform can be frequency translated (or downconverted) without loss to complex baseband by multiplying the received and local signals and complex-low-pass filtering the result (denoted CLP) to eliminate the double-frequency terms, yielding the complex baseband waveform

$$\tilde{r}(t) = \tilde{r}_0(t)\, \tilde{s}_{\mathrm{LO}}(t)|_{\mathrm{CLP}} = \sqrt{P}\, m(t)\, e^{j\varphi(t)} + \big\{N_c(t) + jN_s(t)\big\} e^{j\varphi(t)} \tag{7}$$

where $\varphi(t) \equiv \theta(t) - \hat{\theta}(t)$. If the phase estimate is suitably close to the true phase, so that $\varphi(t) \approx 0 \quad \forall t$, then the downconversion is considered coherent; otherwise, it is partially coherent.

III. Coherent Detection

Assuming the coherent condition holds, the received signal reduces to

$$\tilde{r}(t) = \Big\{\sqrt{P}\, m(t) + N_c(t)\Big\} + jN_s(t) \tag{8}$$

Note that the signal is contained entirely in the real part; therefore, only the real part of the complex baseband signal needs to be observed when the coherent condition holds.

The real part of Eq. (8) is the real function

$$r(t) = \sqrt{P}\, m(t) + N_c(t) \equiv s_m(t) + N_c(t) \tag{9}$$

where $E\,N_c(t) = 0$ with two-sided spectral level $S_n(f) = N_0/2$ and single-sided bandwidth W. Assume the noise bandwidth, W, is much greater than the signal bandwidth, B, and expand the received waveform in an orthonormal series. If $W \gg B$ Hz, then the noise autocorrelation function is of the form

$$R(t_1 - t_2) \equiv E\big\{N_c(t_1)\, N_c(t_2)\big\} = N_0 W \frac{\sin\big(2\pi W(t_1 - t_2)\big)}{2\pi W(t_1 - t_2)} \cong \frac{N_0}{2}\delta(t_1 - t_2) \tag{10}$$

In order to expand the signal and noise processes in a series with uncorrelated (for Gaussian noise, independent) coefficients, we make use of the Karhounen–Loeve (K-L) expansion, and select as our basis the set of functions $\{\psi_i(t)\}$ that are the eigenfunctions of Eq. (10), satisfying the integral equation

$$\int_{-\infty}^{\infty} R(t_1 - t_2)\, \psi_i(t_2)\, dt_2 = \lambda_i\, \psi_i(t_1) \tag{11}$$

This integral equation is solved by the well-known sampling functions

$$\psi_i(t) = \sqrt{2W}\ \frac{\sin(2\pi W t - i\pi)}{2\pi W t - i\pi} \tag{12}$$

with $\lambda_i = N_0/2$ for $0 \le i \le 2WT; 0$ otherwise. When operating on any waveform bandlimited to W Hz, these sampling functions effectively yield samples of the waveform at time instants $i/2W$:

$$\int_{-\infty}^{\infty} r(t)\, \psi_i(t)\, dt = \frac{1}{\sqrt{2W}}\, r\left(\frac{i}{2W}\right) \equiv r_i = s_{m,i} + n_i \tag{13}$$

Therefore, if the observation interval is suitably great (so that $2WT \gg 1$), the received random waveforms can be represented using a K-L expansion with uncorrelated coefficients as

$$r(t) = \sum_{i=0}^{2WT} r_i\, \psi_i(t)\, dt = \sum_{i=0}^{2WT} s_{m,i}\psi_i(t) + \sum_{i=0}^{2WT} n_i\psi_i(t) \tag{14}$$

where $s_{m,i} \cong \int_0^T s_m(t)\ \psi_i(t)\, dt$ represents samples of the signal, and similarly n_i represents samples of the noise.

IV. Maximum-Likelihood Detection

In the above derivation, the probability density of each noise sample was assumed to be Gaussian of the form

$$p_{N_c}(n_i) = \frac{1}{\sqrt{2\pi}\sigma} e^{-n_i^2/2\sigma^2} \tag{15}$$

Because the sampling functions are approximate solutions of the integral Eq. (11), different samples are uncorrelated and hence independent for the case of Gaussian processes. The probability density of a vector of noise samples can therefore be expressed as the product of the individual densities:

$$p_{\mathbf{N_c}}(\mathbf{n}) = \prod_{k=0}^{2WT} p_{N_c}(n_k) = \frac{1}{(2\pi\sigma^2)^{(2WT+1)/2}}\, e^{-\sum_{k=0}^{2WT} n_k^2/2\sigma^2} \tag{16}$$

The probability density of the received vector, conditioned on the signal waveform $m(t)$, can be obtained from Eq. (16) directly by noting that $n(t) = r(t) - s_m(t)$; therefore,

$$p_r\big(\mathbf{r}|s_m(t)\big) = p_{N_c}\big(\mathbf{r} - \mathbf{s}_m|s_m(t)\big) = \frac{1}{\big(2\pi\sigma^2\big)^{(2WT+1)/2}}\, e^{-\sum_{k=0}^{2WT} (r_k - s_{m,k})^2/2\sigma^2} \tag{17}$$

where $\mathbf{r}$ and $\mathbf{s}_m$ are vectors of coefficients representing the received and signal waveforms. In decision theory, this conditional density is called the likelihood function and is the quantity to be maximized over the set of messages for maximum-likelihood detection. If $s_m(t)$ represents one of M equally likely messages, the likelihood of the observed vector is evaluated for each of the M possible messages, and the one with the greatest numerical value is selected as our best estimate of the transmitted message, given the observed vector of samples. Since maximization of a monotone-increasing function of the likelihood will provide the same result, it is convenient to maximize the natural logarithm of the conditional density, yielding the log-likelihood function

$$\begin{aligned}\ln p\big(\mathbf{r}|s_m(t)\big) &= -\frac{2WT+1}{2}\,\ln\big(2\pi\sigma^2\big) - \sum_k \frac{(r_k - s_{m,k})^2}{2\sigma^2} \\ &= -\frac{2WT+1}{2}\,\ln\big(2\pi\sigma^2\big) - \sum_k \frac{r_k^2}{2\sigma^2} - \sum_k \frac{s_{m,k}^2}{2\sigma^2} + \sum_k \frac{r_k s_{m,k}}{\sigma^2}\end{aligned} \tag{18}$$

The first two terms on the right-hand side do not depend on m and hence cannot be used to differentiate between the signals. The third term does depend on m but only through the sum of squares of the coefficients. If the sum is independent of the message (i.e., equal-energy signals), then this term can also be ignored. Assuming equal-energy signals and multiplying through by the common scale factor yields the decision function

$$\Lambda_m = \sum_k r_k s_{m,k} = \mathbf{r}\cdot\mathbf{s}_m, \quad m = 1, 2, \cdots, M \tag{19}$$

which is seen to be the inner product of the mth signal and received vectors. Thus, that signal is selected whose correlation with the received vector is the greatest. By substituting the integral form of the samples and letting $2WT = K$, this can also be written in terms of the original time functions as

$$\begin{aligned}\Lambda_m &= \sum_{k=0}^{K} r_k s_{m,k} = \sum_{k=0}^{K} \int_0^T r(t_1)\psi_k(t_1)dt_1 \int_0^T s_m(t_2)\psi_k(t_2)dt_2 \\ &= \int_0^T dt_1 \int_0^T dt_2\, r(t_1)s_m(t_2) \sum_{k=0}^{K} \psi_k(t_1)\psi_k(t_2) \propto \int_0^T dt_1 \int_0^T dt_2\, r(t_1)s_m(t_2)R(t_1 - t_2) \\ &\cong \frac{N_0}{2}\int_0^T dt_1 \int_0^T dt_2\, r(t_1)s_m(t_2)\delta(t_1 - t_2) = \frac{N_0}{2}\int_0^T r(t)s_m(t)dt\end{aligned} \tag{20}$$

Here we made use of Mercer's theorem to substitute the correlation function for the sum of sampling-function products in the second line. Therefore, it can be seen that the mth log-likelihood function can be equivalently expressed as the correlation of the received waveform with the mth signal waveform. The continuous version of the log-likelihood function often yields an accurate representation of densely sampled waveforms—that is, waveforms that are sampled much faster than the inverse of the signal bandwidth.

V. Information Transfer and Performance of Binary Sequences

The above derivation holds for arbitrary M-ary BPSK-modulated signals in additive Gaussian noise and demonstrates that, for the case of equilikely signals, best performance is obtained by correlating the received waveform with all possible realizations of the signal. However, not all signal sets yield the same detection performance nor provide the same information transfer to the receiver, as we now demonstrate.

Consider the case of BPSK modulation, where the amplitude of the received signal is modulated with a sequence of real binary symbols taking on the value $+\sqrt{P}$ or $-\sqrt{P}$ in each consecutive τ second time interval. Thus, if the mth transmitted symbol consists of a sequence of K binary symbols represented by the K-dimensional vector $\mathbf{x} = (x_{m,1},\, x_{m,2}, \cdots, x_{m,K})$, then the modulation can be expressed as

$$m(t) = \begin{cases} x_{m,1} & 0 \le t < \tau \\ x_{m,2} & \tau \le t < 2\tau \\ \quad \vdots & \\ x_{m,N} & (N-1)\tau \le t < N\tau \end{cases} \tag{21}$$

where $x_{m,i} = \pm 1$. For signals of the form described in Eq. (21), the log-likelihood function can now be rewritten as

$$\Lambda_m = \int_0^T r(t) s_m(t) dt = \sum_{k=1}^{K} \int_{(k-1)\tau}^{k\tau} r(t) s_m(t) dt = \sum_{k=1}^{K} \rho_{m,k} \tag{22}$$

which is recognized as a sum of Gaussian random variables $\rho_{m,k}$ over k. Therefore, the mth log-likelihood function is a Gaussian random variable with mean value determined by the correlation between the received and locally generated signals, and variance equal to the sum of the component variances: $\sigma_\Lambda^2 = K\, N_0 \tau/2$.

The maximum amount of information that can be conveyed by K binary symbols is K bits; this is achieved when all possible 2^K sequences are used to modulate the carrier. At the other extreme, if the information symbol is simply repeated K times, then only 1 bit of information is transmitted in $K\tau$ seconds. It is instructive to examine the error performance of these two extreme cases, since these serve as useful bounds to the information content and word-error performance of commonly used sequences.

A. Symbol-Error Performance with Maximal Information Transfer

Consider a sequence of K consecutive uncoded BPSK symbols, where each symbol lasts for τ seconds. For example, if $K = 1$, two distinct mean values are possible for each received symbol, namely $\pm\sqrt{P}\tau$, but only 1 bit of information is transferred. If $K = 2$, then 2 bits of information are transferred; now the possible mean vectors are $(-\sqrt{P}\tau, -\sqrt{P}\tau), (-\sqrt{P}\tau, \sqrt{P}\tau), (\sqrt{P}\tau, -\sqrt{P}\tau)$, and $(\sqrt{P}\tau, \sqrt{P}\tau)$. The extension to longer sequences is straightforward; for example, one of the possible received sequences of length eight is $(\sqrt{P}\tau, -\sqrt{P}\tau, \sqrt{P}\tau, \sqrt{P}\tau, -\sqrt{P}\tau, -\sqrt{P}\tau, -\sqrt{P}\tau)$. It is clear that, for any K, exactly 2^K codewords can be constructed in this manner.

First, consider the case of $K = 1$. Only two hypotheses are possible in this case, denoted by H_1 and H_2, occurring with a priori probabilities $P(H_1)$ and $P(H_2)$. To compute receiver performance, suppose that one of the two messages has been received, and compute the probability of correct detection conditioned on the reception of that message. It is implicitly assumed that a replica of the received waveform, $r(t)$, has been stored in a form suitable for repeated processing. With $r(t) = \sqrt{P} + N_c(t), 0 \le t < \tau$ and $s_1(t) = 1, 0 \le t < \tau$, the receiver computes Λ_1, which is seen to be a Gaussian random variable with mean $\sqrt{P}\tau$ and variance $N_0\tau/2$. Next, the receiver multiplies the same received waveform by $s_2(t) = -1, 0 \le t < \tau$ and computes Λ_2. Note that in this example Λ_2 is exactly equal to the negative of Λ_1.

Since by assumption the first hypothesis is true, the correct binary symbol is selected if $\Lambda_1 > \Lambda_2$, or, equivalently, if $\Lambda_1 - \Lambda_2 > 0$, in which case the receiver declares that hypothesis H_1 is true. However, since $\Lambda_2 = -\Lambda_1$, it follows that the mean and variance of the difference are $E(\Lambda_1 - \Lambda_2) = 2\sqrt{P}\,\tau$ and $\text{var}(\Lambda_1 - \Lambda_2) = 4 \times (N_0\tau/2) = 2N_0\tau$. The probability of a correct decision, given that H_1 is true, is simply the probability that a Gaussian random variable with mean $2\sqrt{P}\tau$ and variance $2N_0\tau$ exceeds zero:

$$\Pr(\Lambda_1 - \Lambda_2 > 0 | H_1) = \int_0^\infty \frac{1}{\sqrt{2\pi(2N_0\tau)}} e^{-(x-2\sqrt{P}\tau)^2/2(2N_0\tau)} dx \tag{23}$$

With the change of variables $y = (x - 2\sqrt{P}\tau)/\sqrt{2N_0\tau}$ and $dy = dx/\sqrt{2N_0\tau}$, Eq. (23) yields

$$\Pr\left(\Lambda_1 - \Lambda_2 > 0 | H_1\right) = \int_{-2\sqrt{P\tau/2N_0}}^\infty \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dy = 1 - Q\left(\sqrt{\frac{2P\tau}{N_0}}\right) = 1 - Q\left(\sqrt{\frac{2E_s}{N_0}}\right) \tag{24}$$

where $Q(x) \equiv \int_x^\infty (1/\sqrt{2\pi}) e^{-y^2/2} dy$.

The probability of correct decision is the average (over the a priori probabilities) of the conditional probability of being correct: $P(C) = P(C|H_1)P(H_1) + P(C|H_2)P(H_2)$. For equilikely hypotheses, $P(H_1) = P(H_2) = 1/2$; hence, for this case, the average probability of correct decoding is $P(C) = P(C|H_1) = 1 - Q\left(\sqrt{2E_s/N_0}\right)$, while the average probability of error is $P(E) = 1 - P(C) = Q\left(\sqrt{2E_s/N_0}\right) \equiv p$.

Next, consider the probability of selecting the correct K sequence when all possible realizations are allowed. We observe that for $K = 2$ the possible signal vectors occupy all possible corners of a square in the two-dimensional signal space spanned by the vectors $(1, 1), (1, -1)$. In general, the signal vectors occupy every corner of a hypercube of dimension K. As shown in [3], the probability of correctly decoding a vector of length K when all possible realizations are allowed is given by $P(C) = (1 - p)^K$, which is seen to be the probability of not making any binary symbol errors in K independent trials, and where the probability of a binary symbol error is $p \equiv Q(\sqrt{2E_s/N_0})$. It follows that the probability of vector error, defined as the probability of not selecting the correct received vector, is $P(E) = 1 - (1 - p)^K$. A graph of this vector-error probability as a function of E_s/N_0 is shown in Fig. 2, labeled as "all possible sequences," for $K = 8$ and 16, in the low-SNR region of operation.

An interesting conclusion follows from the observation that the probability of a correct vector decision is equivalent to the probability of selecting a vector whose Hamming distance from the received information vector is zero. Similarly, the probability of selecting a vector that is a Hamming distance one from the received vector is $K\,p(1 - p)^{K-1}$, that of selecting a vector a distance two from the transmitted vector is $\binom{K}{2} p^2 (1 - p)^{K-2}$, while the general term is

$$\Pr(k \text{ symbol errors in a vector of length } K) = \binom{K}{k} p^k (1-p)^{K-k} \tag{25}$$

Evidently, the errors are governed by the binomial distribution; therefore, the average number of errors in a vector of length K is Kp. Alternatively, it follows that the average probability of a binary symbol error is $P(SE) = p$. Since p is also the probability of making a binary symbol error when decoding individual binary symbols one at a time, this result confirms the observation that vector decoding of all possible binary sequences is equivalent to bit-by-bit decoding.

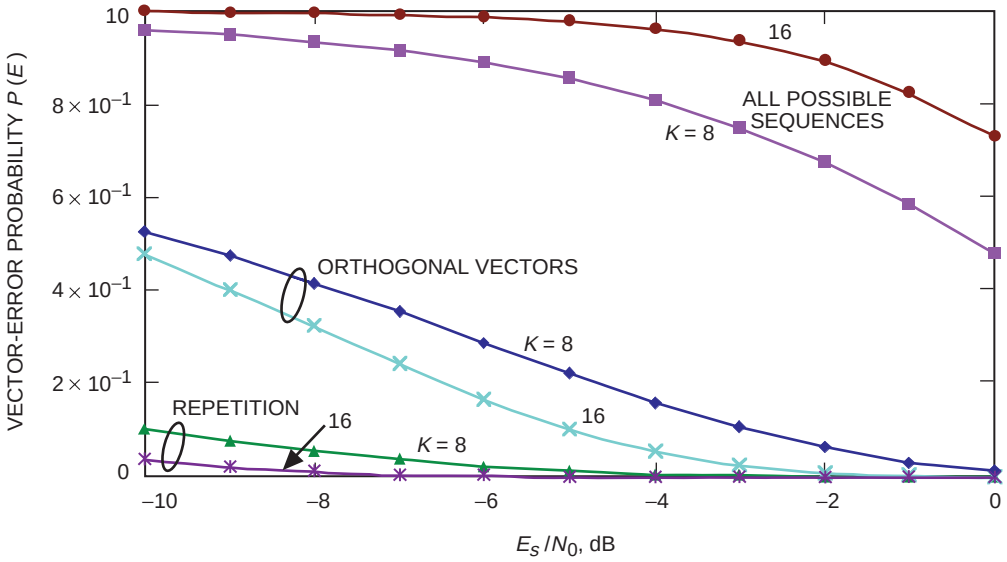

Fig. 2. Probability of vector error for the case of all possible sequences, orthogonal vectors, and repetition vectors.

B. Repetition Coding: Lower Bound on Symbol-Error Probability

Suppose that instead of sending all possible sequences, we simply repeat each binary symbol K times. This scheme generates exactly two mean vectors of length K, namely $(\sqrt{P}\tau, \sqrt{P}\tau, \cdots, \sqrt{P}\tau)$ and $(-\sqrt{P}\tau, -\sqrt{P}\tau, \cdots, -\sqrt{P}\tau)$. The amount of information transferred in $K\tau$ seconds is only 1 bit, however, since the energy of each information bit has been increased by a factor of K, decoding performance has been greatly improved. Specifically, the probability of a vector error now becomes $P(E) = Q(\sqrt{2KE_s/N_0})$, which represents a dramatic improvement over the performance of the maximal information transfer vectors, as can be seen in Fig. 2.

Note that for this case the probability of a binary symbol error, $P(SE)$, is equivalent to the probability of a vector error, $P(E)$, since grouping into blocks of K binary symbols does not change the average probability of symbol error (if a vector error occurs, all K binary symbols will be in error; however, each correctly decoded vector also yields K correctly decoded binary symbols, hence grouping blocks of symbols has no effect on the average number of errors). Therefore, $P(SE) = P(E) = Q\left(\sqrt{2KE_s/N_0}\right)$ for the case of repetition coding.

C. Performance of Orthogonal Block-Coded Modulation

There are many practical encoding schemes that transfer more than 1 bit of information per vector and at the same time achieve reasonable error probabilities. Perhaps the simplest among these is orthogonal block coding, where $\log_2 K$ information bits are encoded onto K orthogonal vectors. For example, vectors of length 8 carry 3 bits of information; length 32 contain 5 bits; and so on. An accepted technique

for generating orthogonal codewords is to use the sign convention defined by the rows of a Hadamard matrix of suitable dimension. There is a Hadamard matrix of order equal to any power of two. As in [2], a Hadamard matrix of order K can be constructed from one of order $K/2, K = 4, 8, 16, \cdots$, using Sylvester's construction, as

$$\mathbf{H}_K = \begin{bmatrix} \mathbf{H}_{K/2} & \mathbf{H}_{K/2} \\ \mathbf{H}_{K/2} & -\mathbf{H}_{K/2} \end{bmatrix} \tag{26}$$

For example, a Hadamard matrix of order four is

$$\mathbf{H}_4 = \begin{bmatrix} 1, & 1 & 1, & 1 \\ 1, & -1, & 1, & -1 \\ 1, & 1, & -1, & -1 \\ 1, & -1, & -1, & 1 \end{bmatrix} \tag{27}$$

The encoder assigns each sequence of $\log_2 K$ information bits in some predetermined order to different rows of the Hadamard matrix. One possible assignment for the case $K = 4$ is the following:

$$(1,1) \rightarrow (1,1,1,1)$$
$$(1,0) \rightarrow (1,-1,1,-1)$$
$$(0,1) \rightarrow (1,1,-1,-1)$$
$$(0,0) \rightarrow (1,-1,-1,1)$$

If the first codeword is received, the mean-value vector at the receiver is of the form ($\sqrt{P}\tau, \sqrt{P}\tau, \sqrt{P}\tau, \sqrt{P}\tau$); let the reception of this vector be denoted by hypothesis H_1. When correlated with $s_1(t)$ as defined in Eq. (19), the mean value of the first log-likelihood function becomes $K\sqrt{P}\tau$. Since noise samples from disjoint time intervals are uncorrelated, the variance of the first log-likelihood function is the sum of the component variances, namely $KN_0\tau/2$. Because the codewords are orthogonal, the mean value of every other log-likelihood function is zero; however, its variance does not change. Therefore, when H_1 is true, $E(\Lambda_1) = K\sqrt{P}\tau,\ \text{var}\ (\Lambda_1) = KN_0\tau/2$. For every other log-likelihood function, $E(\Lambda_k) = 0,\ \text{var}\ (\Lambda_k) = KN_0\tau/2;\ k \neq 1$.

The received noise-corrupted codeword is correlated with every row of the Hadamard matrix, and the row corresponding to the greatest correlation coefficient is selected as the maximum-likelihood codeword decision. In other words, the correct codeword is selected if the correlation result corresponding to the correct codeword exceeds all others; for the case of additive Gaussian noise, the probability of correct detection given H_1, $P(C|H_1)$, can be expressed as

$$P(C|H_1) = \int_{-\infty}^{\infty} \frac{dy}{\sqrt{2\pi \dfrac{KN_0\tau}{2}}}\, e^{-(y-K\sqrt{P}\tau)^2/2(KN_0\tau/2)} \left[\int_{-\infty}^{y} \frac{dx}{\sqrt{2\pi \dfrac{KN_0\tau}{2}}}\, e^{-x^2/2(KN_0\tau/2)} \right]^{K-1} \tag{28}$$

which is also equal to the average probability of correct detection, $P(C)$, when all hypotheses are equally likely (that is, when $P(H_i) = (1/K),\ i = 1, 2, \cdots, K$). The probability of selecting the wrong vector is $P(E) = 1 - P(C)$, also shown in Fig. 2 for the cases $K = 8$ and 16. Note that the probability of vector error actually decreases with increasing K for orthogonal vectors; this occurs because here we are keeping

the binary symbol SNR, instead of the total vector SNR, fixed. Since longer vectors have greater energy, performance improves with vector length.

For orthogonal vectors, the probability of a binary symbol error, $P(SE)$, can be related to the probability of correct vector detection as $P(SE) = [K/2(K-1)]P(E)$. Binary symbol-error probabilities for orthogonal codewords of lengths 8 and 16 are shown in Fig. 3 as functions of the binary symbol SNR E_s/N_0, in the low-SNR region, along with binary symbol-error probabilities for all possible sequences and for repetition vectors. Note that the binary symbol-error probability for all possible sequences remains p regardless of the value of K, but changes with K for both the orthogonal and repetition vectors.

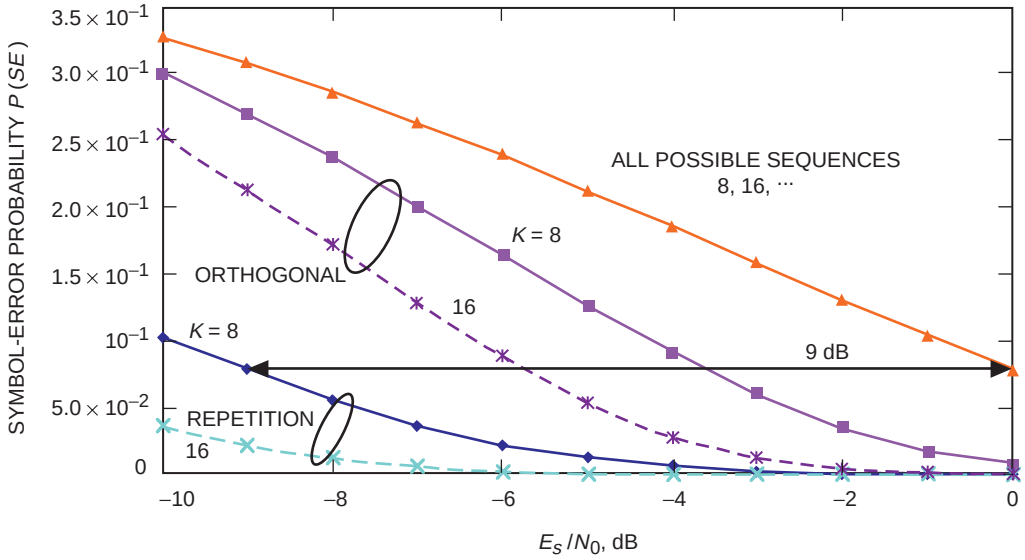

Fig. 3. Probability of binary symbol error for the case of all possible sequences, orthogonal vectors, and repetition vectors.

D. Algebraic Block Codes

The process of encoding and decoding information using algebraic block codes is treated extensively in the literature. For our interest here, we note that pre-processing by means of maximum-likelihood block decoding as defined in Eq. (19) requires the generation and storage of all possible channel codewords for subsequent correlation with the received noise-corrupted codeword. This suggests incorporating an encoder into the pre-processor, which often consists of nothing more than a generator matrix or generator polynomial, plus the arithmetic operations needed to generate the codewords. Complexity and storage requirements tend to grow with block length and may become prohibitive for large blocks, due to possible exponential growth; however, optimum detection is always achieved. Despite the complexity, decoding appears feasible with currently available dedicated high-speed hardware at moderate bit rates, for moderate-sized dictionaries consisting of perhaps thousands to millions of codewords. Since squaring loss is most pronounced in the low-SNR region at low symbol rates, it appears that decoding based on the maximum-likelihood approach may not present a serious problem for deep-space applications.

E. Convolutional Codes

It would be advantageous if the same maximum-likelihood block decision approach described above could also be applied to convolutional codes, since that would enable the generalization of the pre-processor structure to most commonly employed encoding schemes, including turbo codes. An additional

benefit of processing short blocks of convolutionally encoded symbols is that the user can effectively match the vector lengths to channel conditions, ensuring essentially stationary statistics over each block. However, this requires the generation of all possible sequences of a given length for correlating with the received waveform. We begin by describing the partitioning of convolutionally encoded binary sequences into blocks, and the operations for generating the complete set of codewords needed for maximum-likelihood detection.

Convolutional encoders generate channel symbols by sampling the contents of serially connected shift registers in a predetermined manner and at a predetermined rate. This operation is illustrated in Fig. 4, which represents a three-register, rate 1/2 encoder, with taps represented by the generating polynomials $p = 1\,1\,1$ and $q = 1\,0\,1$. We are interested in decoding the output of the convolutional encoder with the maximum-likelihood block decoding strategy described above. One reason for this approach is that, in a time-varying channel, where the phase of the signal changes with time, there is an implied coherence time over which the phase remains essentially constant. Therefore, the user can select the block length to match channel conditions, thus ensuring detection for each block.

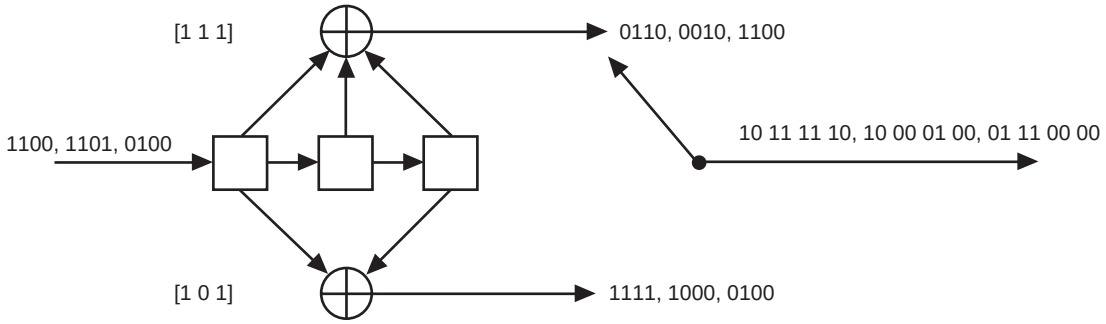

Fig. 4. Block codeword representation of convolutional codes (each block codeword is separated by a comma).

The information sequence shown in Fig. 4 has been separated into blocks of 4 bits; since this is a rate 1/2 encoder, each block of 4 information symbols gives rise to a block of 8 output symbols. If initially the three registers contained zeros, represented by 000, then shifting the first information block through 1 bit at a time (in our example, 0100, with the rightmost bit entering first) yields the output block 01 11 00 00. Since there is a one-to-one mapping from the input space to the output space, the total number of output symbols equals the total number of input symbols, which for this case is $2^4 = 16$.

It is evident from the encoder structure that channel-symbol blocks generated by the convolutional encoder depend not only on the current information bits but also on the state the registers were left in by the trailing bits of the last information block. After every symbol of the first block has been clocked through, the shift registers contain 010. When the first bit of the second information block is clocked in, the contents of the rightmost shift register is clocked out; for our example, the shift registers now contain 101, where the leftmost register contains the first symbol of the second information block, and the middle and rightmost registers contain the trailing symbols of the first information block. The rightmost 2 bits determine the state of the register and clearly affect the next output block; even if exactly the same information bits were to be reentered, the output block would be different. Since there are $2^2 = 4$ states possible in our example, the total number of codewords this three-stage encoder can generate after the first input block is shifted through is $16 \times 4 = 64$.

VI. Simplified Pre-Processor Structure

The correlation operation specified by the maximum-likelihood detection algorithm requires knowledge of every codeword generated by the encoder. This approach does not make use of the algebraic structure inherent in block codes to simplify decoder design, but rather employs the encoder to generate a copy of every codeword at the receiver. An immediate simplification to the pre-processor structure results from the observation that, for BPSK-modulated signals, correlation with all possible codewords is equivalent to correlation with a suitable orthogonal basis, followed by matrix multiplication to reconstruct the codewords. Since only K basis vectors are needed to span a K-dimensional signal space, while the number of possible codewords could grow exponentially with K, this approach often results in substantial simplifications to the processing algorithms and hardware. For block codes, the equivalence of the direct and simplified approaches can be demonstrated as follows.

Codewords for block codes of a given length consist of both information and check bits. The binary values (0,1) generated by the encoder are modulated onto the carrier as BPSK symbols via the mapping $0 \rightarrow -1,\ 1 \rightarrow 1$ (or vice-versa). After this mapping, a valid BPSK-modulated codeword of length K, C_i, selected from the set of codewords $\{C\}$, takes the form $C_i = (c_{i1}, c_{i2}, \cdots, c_{iK})$, where the elements of the vector are $c_{ik} = \pm 1$. Since each codeword is K-dimensional, it can be represented by a linear superposition of basis functions that span the same K-dimensional space. Any orthogonal subset of the set of BPSK sequences of length K will suffice; however, we shall select the rows of the K-dimensional Hadamard matrix as our basis because of their close connection to some algebraic codes, such as the Reed–Muller codes, and because the rows of a Hadamard matrix are orthogonal. Assuming that $K = 2^L$, where L is some integer, we can therefore represent the ith codeword of an arbitrary block code as

$$C_i' = \sum_{j=1}^{K} a_{ij} H_j \tag{29}$$

where $H_j = (h_{j1}, h_{j2}, \cdots, h_{jK}), h_{jk} = \pm 1$, and $a_{ij} = C_i H_j^T$, where the vector H_j is equal to the jth row of the K-dimensional Hadamard matrix. Since a common scale factor applied to all codewords does not affect the decision, it is not necessary to normalize the coefficients a_{ij} in this application.

It is useful to define the time function corresponding to the jth row of the Hadamard matrix as

$$h_j(t) \equiv \begin{cases} h_{j1} & 0 \le t < \tau \\ h_{j2} & \tau \le t < 2\tau \\ & \vdots \\ h_{jK} & (K-1)\tau \le t < K\tau \end{cases}, \quad h_{jk} = \pm 1 \tag{30}$$

We can write the time-function representation of the mth codeword as

$$\left.\begin{aligned} s_m(t) &= \sum_{j=1}^{K} s_{mj}^{(h)} h_j(t) \\ s_{mj}^{(h)} &= \int_0^T s_m(t) h_j(t)\, dt \end{aligned}\right\} \tag{31}$$

Given that the ith signal was received (that is, given that H_i is true), the mth log-likelihood function can now be expressed as

$$\Lambda_m = \int_0^T r(t)s_m(t)\,dt = \int_0^T r(t)\sum_{j=1}^{K} s_{mj}^{(h)} h_j(t)\,dt$$

$$= \sum_{j=1}^{K} s_{mj}^{(h)} \int_0^T r(t)h_j(t)\,dt = \sum_{j=1}^{K} r_j^{(h)} s_{mj}^{(h)} = \mathbf{r}^{(h)} \cdot \mathbf{s}_m^{(h)} \tag{32}$$

where the last equality follows directly from the projection of the received signal onto the jth Hadamard basis function: $r_j^{(h)} = \int_0^T r(t)h_j(t)\,dt$.

Suppose there are $N \geq K$ codewords in a block code. The vector of log-likelihood function values, $\bar{\Lambda} = (\Lambda_1, \Lambda_2, \cdots, \Lambda_N)$, needed for the maximum-likelihood decision, can now be obtained as a simple matrix multiplication of the codeword coefficient matrix with the transpose of the received coefficient vector,

$$\Lambda^T = \mathbf{S}^{(h)} \left(\mathbf{r}^{(h)}\right)^T \tag{33}$$

where

$$\mathbf{S}^{(h)} \equiv \begin{bmatrix} s_{11}^{(h)} & s_{12}^{(h)} & \cdots & s_{1K}^{(h)} \\ s_{21}^{(h)} & s_{22}^{(h)} & \cdots & s_{2K}^{(h)} \\ & & \vdots & \\ s_{N1}^{(h)} & s_{N2}^{(h)} & \cdots & s_{NK}^{(h)} \end{bmatrix}$$

and $\mathbf{r}^{(h)} = (r_1^{(h)}, r_2^{(h)}, \cdots, r_K^{(h)})$. Implicit in this solution is the assumption that the codeword coefficient matrix is known; however, this is not a serious restriction since the coefficient matrix can always be generated at the pre-processor by using the block code's encoder to generate all possible codewords, followed by a Hadamard decomposition to obtain the coefficients. A block diagram of the simplified pre-processor structure corresponding to Eq. (33) is illustrated in Fig. 5.

The extension to convolutional codes, processed in blocks as described above, is straightforward. The first block generated by the encoder with initialized registers is a sequence of binary symbols of length K and hence is subject to the Hadamard decomposition for a K-dimensional space. The second and subsequent blocks produce more vectors proportional to the total number of states, but still span the same K-dimensional space. Therefore, all channel blocks generated by convolutional encoders can also be reconstructed from the Hadamard coefficients by means of simple matrix multiplication.

VII. Simulation Results

A. Pre-Processor Structure

The pre-processor structure shown in Fig. 5 was simulated using the signal processing workstation (SPW). In the coherent mode of operation (that is, $\phi(t) \cong 0$), this pre-processor performs a decomposition of the received codewords using orthogonal Hadamard basis functions as described above, and generates a vector of coefficients of length K for each received block. Next, the coefficient vector is multiplied by a

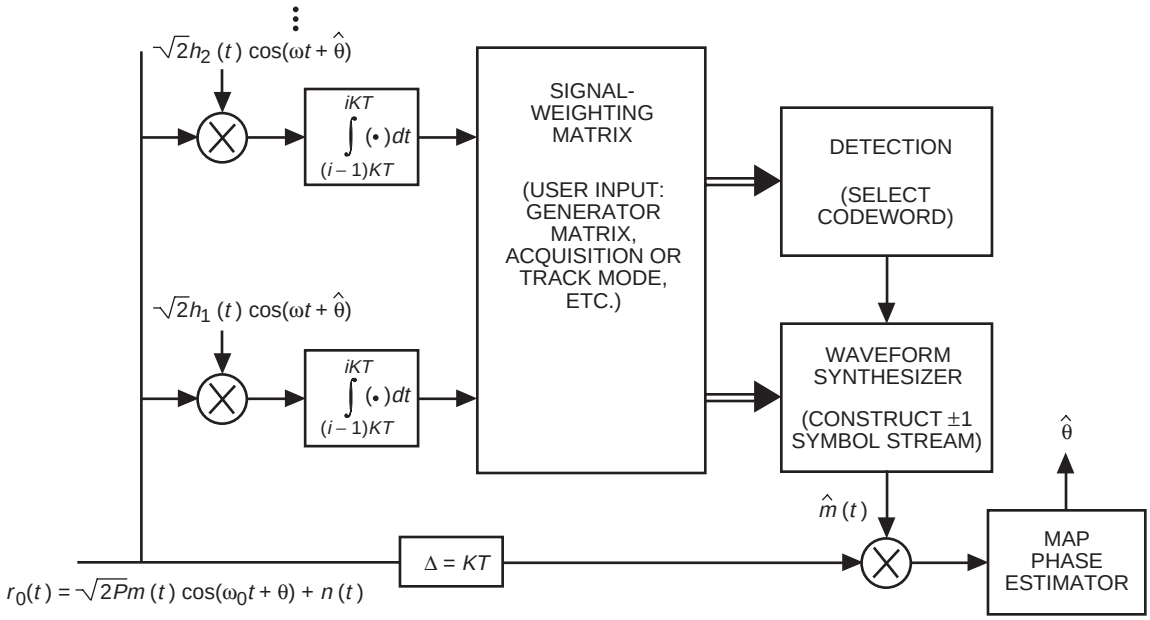

Fig. 5. Block diagram of the simplified pre-processor, using Hadamard decomposition and matrix reconstruction.

signal-weighting matrix to generate the vector of log-likelihood functions. The largest component of this decision vector is identified, and its index is used to reconstruct the decoded codeword.

Note that the basic pre-processor structure remains essentially the same for any code of a given block length, except for the signal-weighting matrix, which is unique to each code. The signal-weighting matrix is pre-computed for each coding scheme by first generating the set of codewords and then multiplying the Hadamard matrix by the coefficient matrix representing all possible codewords.

In the simulations, perfect symbol timing was assumed, and only block lengths that were powers of two were simulated. The sampling rate was set to correspond to 500,000 samples/second, and an equivalent symbol rate of 20,000 symbols/second was used, yielding 25 samples/symbol. In addition to orthogonal block codes, an extended Hamming code, an extended Bose–Chaudhuri–Hocquenghem (BCH) code, and a three-stage convolutional code were simulated and evaluated in the low-SNR region.

B. Performance of the Pre-Processor with Algebraic Block Codes

Extended Hamming codes can be created from a Hamming generator matrix, G, by appending a column of ones to create a new generator matrix, G'. A particular codeword vector, $\mathbf{C}$, can be generated by multiplying a block of data symbols, $\mathbf{x}$ (row vector), with the generator matrix, G': $\mathbf{C} = \mathbf{x}\,G'$, using modulo 2 arithmetic. In our example, a data vector consisting of 4 symbols is multiplied together with a 4×8 generator matrix to yield a codeword vector of length 8, which is a power of two and hence matches the input requirements of the simulated pre-processor. In the pre-processor, all possible information sequences are multiplied with the generator matrix to create every codeword, from which the coefficient matrix can be constructed using the Hadamard decomposition described above.

BCH codewords were created by convolving each information sequence with a generator polynomial. In the simulations, the generator polynomial $g(x) = [1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 1\ 1\ 1]$ was implemented, which represents a (15,5,3) BCH code. In order to create codeword vectors whose lengths were powers of two, an extended BCH code was created by adding a parity check bit to each codeword.

The symbol-error performance of the pre-processor operating on extended Hamming and extended BCH algebraic codes as a function of E_s/N_0 is shown in Figs. 6 and 7, where the theoretical performance

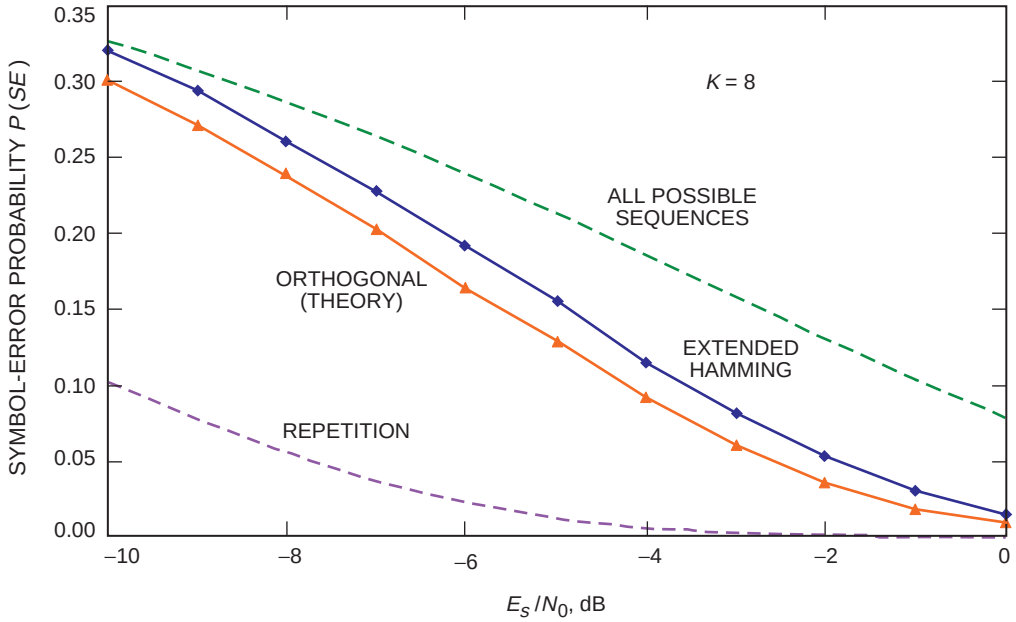

Fig. 6. Probability of binary symbol error for length-8 orthogonal and extended Hamming codes (all-possible-sequences and repetition vectors are included for reference).

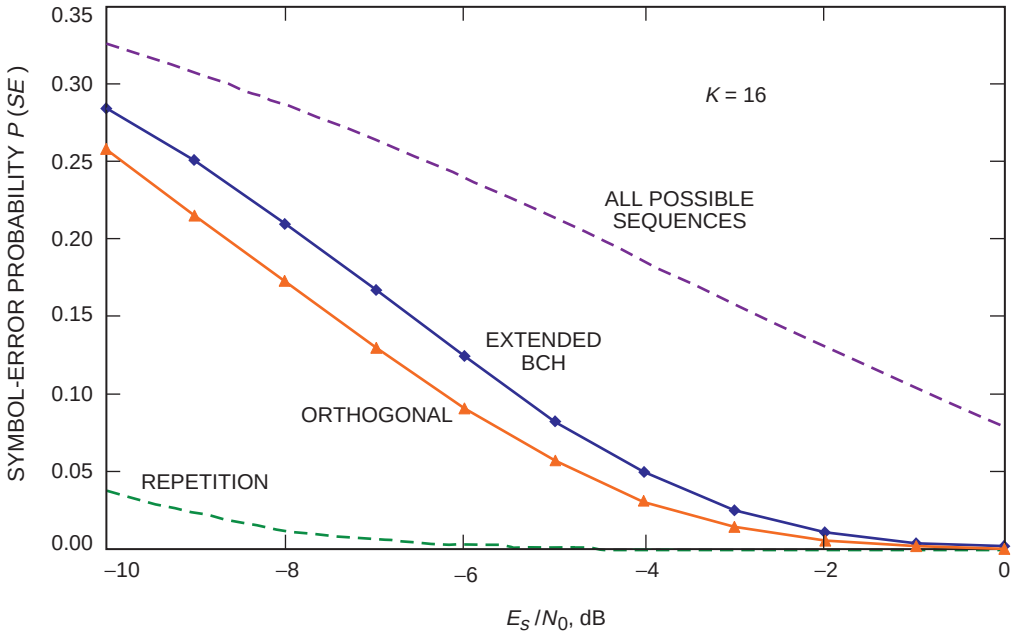

Fig. 7. Probability of binary symbol error for length-16 orthogonal and extended BCH codes (all-possible-sequences and repetition vectors are included for reference).

of the appropriate orthogonal block code was also included for reference. Note that both algebraic codes perform somewhat worse than the corresponding orthogonal block code in terms of symbol-error probability; however, the algebraic codes convey more information than orthogonal block codes due to the larger number of codewords in their dictionary. Symbol-by-symbol decoding, represented by the dashed curve labeled "all possible sequences," appears to perform nearly as well as the algebraic codes at very low SNRs (-10 dB E_s/N_0); however, no improvement in squaring-loss performance over a conventional

tan-hyperbolic phase-locked loop is possible in this case because the pre-processor's symbol estimates are completely correlated with the symbol estimates in the in-phase arm of the loop, see [1, Appendix A].

C. Performance of the Pre-Processor with Convolutional Codes

Convolutional codes were generated for the simulation using a rate 1/2 code with generator polynomials $p = [1\ 1\ 1]$ and $q = [1\ 0\ 1]$. To generate a list of all possible codewords, a block-code representation of convolutional codes described in Section V.E has been used.

Two block-decoding schemes for convolutionally encoded data have been simulated. First, decoder performance is bounded by assuming that the state of the registers is known after each input block; hence, the pre-processor needs to correlate the received waveform with only 16 codewords (for our example). This is a useful bound, but not a practical decoding scheme because in reality the states are not known and, therefore, must be estimated along with the codewords; incorrect decoding results in loss of state information, which means the pre-processor generates the wrong codeword set for correlating with the received signal. This, in turn, leads to incorrect decoding of subsequent codewords, with little chance of ever recovering the correct state. This lower bound on pre-processor symbol-error rate is illustrated in Fig. 8 by the curve labeled "pre-processor (known state bound)" and represents performance attainable only when the states of the convolutional encoder are perfectly known.

The second scheme makes no attempt to estimate the state of the encoder; hence, the pre-processor must correlate the received waveform with all possible codewords associated with every state, which for our case becomes $16 \times 4 = 64$ codewords. Symbol-error performance of this approach is also shown in Fig. 8, as the curve labeled "pre-processor (all states)." Its symbol-error performance is approximately 1.5 dB worse than that of the known state bound, and in fact closely matches the performance of symbol-by-symbol decoding represented by the "all possible sequences" curve. However, it must be remembered

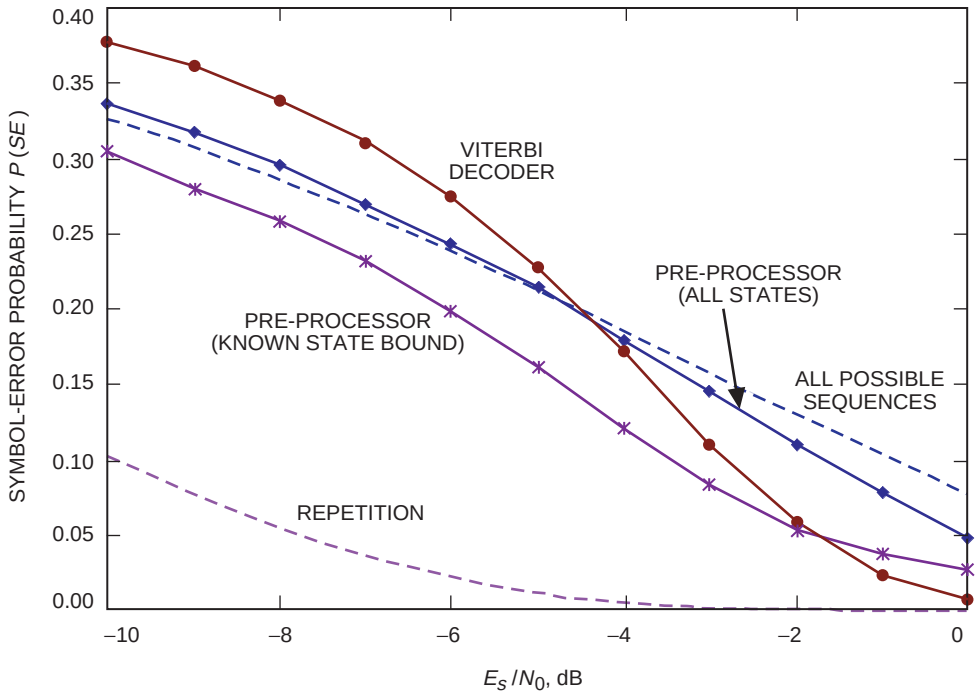

Fig. 8. Probability of binary symbol error for convolutionally coded symbols, block-length-8 decoding (Viterbi decoder and all-possible-sequences and repetition vectors are also included for reference).

that the block-decoding scheme reduces the correlation between the output of the symbol integrator in the in-phase arm of the MAP phase estimator and the pre-processor's symbol estimates (which are now based on a block decision), thus enabling squaring-loss reduction, whereas symbol-by-symbol estimation does not [1, Appendix A]. Although this block pre-processor approach attains optimal performance for the expanded set of all possible codewords, it may become prohibitively complex for long registers and block lengths due to the great number of codewords that must be generated and processed.

Finally, the symbol-error performance of a conventional Viterbi decoder of the type generally used in the DSN has also been included for comparison, labeled "Viterbi decoder" in Fig. 8. In order to evaluate the Viterbi decoder in the same framework as the pre-processor, it was operated with a fixed delay of 8 symbols. Symbol-error performance of the Viterbi decoder operated in this mode is seen to be worse than that of the block pre-processor at SNRs less than -4 dB. Thus, when channel dynamics dictate operation with relatively short delays, the block pre-processor appears to have a clear advantage over the conventional Viterbi decoder operating with fixed delay, in the region of very low signal-to-noise ratios.

It is believed that relatively simple block algorithms can be found for convolutional codes whose performance falls between the two extreme cases discussed above. For example, carrying along the state associated with the selected codeword (the codeword corresponding to the greatest log-likelihood function), plus the state associated with the next most likely codeword, and so on, up to some number that represents a reasonable compromise between complexity and performance, may provide a practical way to avoid catastrophic errors with only a modest increase in complexity.

D. Performance of the Pre-Processor-Aided MAP Phase Estimator

The performance of the entire combined system, consisting of the MAP phase estimator and pre-processor as shown in Fig. 1, has been evaluated by means of SPW simulations. Coded symbol streams were generated as described above, delayed along one path and applied to the pre-processor along the other. After the blocks were decoded, the symbol estimates were applied to the delayed waveform in order to remove the data and thus reduce transitions, and the resulting reconstructed carrier was then applied to the MAP phase estimator along with an estimate of the transition rate measured in real time. The MAP phase estimator then provided an estimate of the instantaneous carrier phase to the pre-processor needed for coherent operation, completing the outer loop. Each symbol was sampled 25 times, and Gaussian noise samples of the proper variance were added to each sample to establish the desired symbol SNR. The MAP phase estimation loop was operated at two different loop bandwidths (5 Hz or 250 Hz), providing either high loop SNR (36 dB) at $E_s/N_0 = 0$ dB for the orthogonal and block-coded symbols, or low SNR (20 dB) at $E_s/N_0 = 0$ dB for some of the convolutionally coded results. The results of a large number of simulations were averaged, where each simulation was long enough to ensure that the entire system reached steady state before any measurements were taken.

The simulations for the convolutional block-decoding schemes were set up so that the loop SNR of the MAP phase estimation loop was 20 dB when the symbol SNR was $E_s/N_0 = 0$ dB. As the symbol SNR decreases, the squaring loss increases, and therefore the loop SNR decreases proportionally. It is generally accepted that, for operational tracking of a BPSK-modulated signal, a Costas-loop SNR of 16 dB must be maintained to limit radio loss to acceptable levels. Note from Fig. 9 that the conventional Costas loop suffers a 4 dB squaring loss when $E_s/N_0 = -4.8$ dB, while the polarity-type Costas loop reaches the 4 dB squaring-loss limit when $E_s/N_0 = -4$ dB; this means that in our example neither loop qualifies for operational tracking of BPSK signals weaker than -4 to -5 dB. However, in the same example, the loop SNR of the combined pre-processor MAP phase estimator actually remains above 17 dB as long as the symbol SNR exceeds -10 dB; this represents at least a 6 dB advantage over Costas loops currently used for tracking suppressed-carrier signals. This very formidable advantage would enable extending the range of a given spacecraft by 3 dB, could be used to reduce spacecraft telecommunications power requirements by 6 dB, or could reduce the diameter of spacecraft or ground antennas by a factor of two.

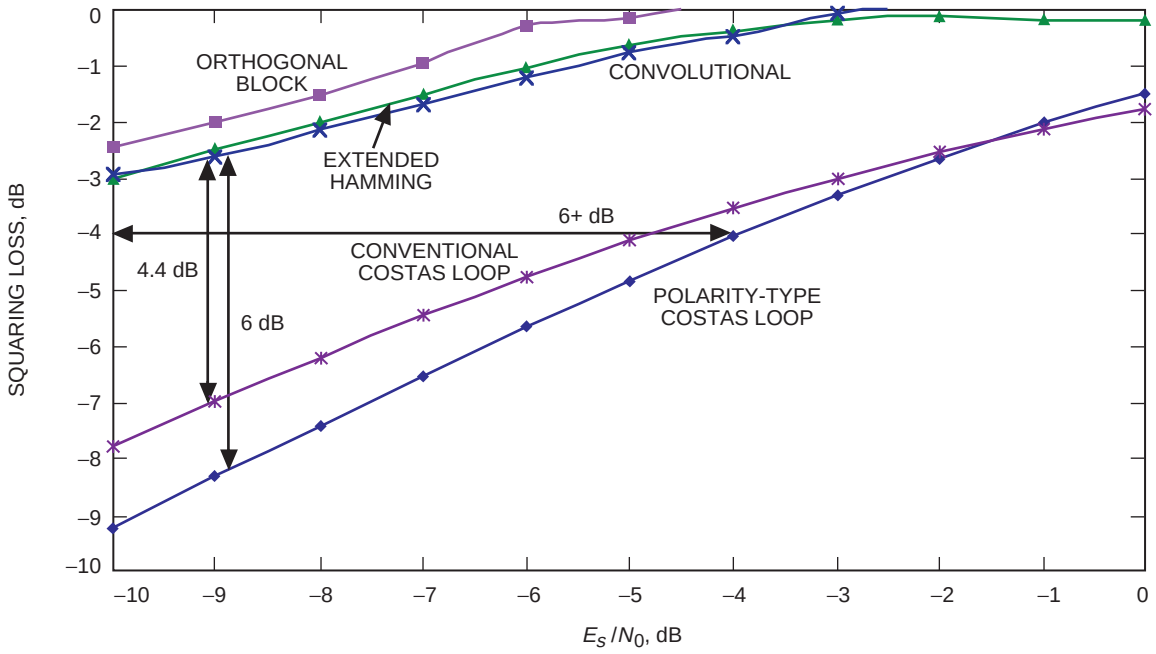

Fig. 9. Comparison of squaring-loss improvement with orthogonal, extended Hamming, and convolutionally coded signals over classical Costas loops (block length 8).

VIII. Summary and Conclusions

A pre-processor structure based on maximum-likelihood principles and applicable to carrier reconstruction of BPSK-modulated signals has been developed and evaluated by means of analysis and SPW simulation. The results of previous investigations restricted to block-orthogonal-modulated signals [2] have been extended to include algebraic block codes as well as convolutionally coded signals. It has been demonstrated that the pre-processor provides useful estimates of the data that can be used to reduce the transitions in a BPSK-modulated waveform, thus effectively reconstructing the carrier and enabling greatly improved phase estimation in the low-SNR regime. It was shown that, when the reconstructed carrier is applied to a MAP phase estimation loop designed for arbitrary transition probabilities, squaring loss in the low-SNR region can be reduced by 4 to 6 dB, enabling loop operation at 7 to 8 dB lower values of E_s/N_0 than with conventional Costas loops. This gain can be used to more than double the useful range of a spacecraft, to reduce spacecraft telecommunications power requirements by a factor of five, or to reduce the diameter of either the spacecraft or ground antennas by a factor of two. In addition to the example presented here, effective carrier reconstruction may also prove valuable in radio-science applications with suppressed-carrier signals, enabling accurate phase measurements without the need to divert precious telemetry signal power to an unmodulated (residual) carrier component.

References

[1] M. K. Simon and V. A. Vilnrotter, "Iterative Information-Reduced Carrier Synchronization Using Decision Feedback for Low SNR Applications," *The Telecommunications and Data Acquisition Progress Report 42-130, April–June 1997*, Jet Propulsion Laboratory, Pasadena, California, pp. 1–21, August 15, 1997. http://tmo.jpl.nasa.gov/tmo/progress_report/42-130/130A.pdf

[2] V. Vilnrotter, A. Gray, and C. Lee, "Carrier Synchronization for Low Signal-to-Noise Ratio Binary Phase-Shift-Keyed Modulated Signals," *The Telecommunications and Mission Operations Progress Report 42-139, July–September 1999*, Jet Propulsion Laboratory, Pasadena, California, pp. 1–16, November 15, 1999. http://tmo.jpl.nasa.gov/tmo/progress_report/42-139/139I.pdf

[3] J. M. Wozencraft and I. M. Jacobs, *Principles of Communication Engineering*, New York: John Wiley and Sons, 1965.