TDA Progress Report 42-108 February 15, 1992

S/4-32

P-15

N92-24315

JJ570450

Node Synchronization Schemes for the Big Viterbi Decoder

K.-M. Cheung, L. Swanson, and S. Arnold
Communications Systems Research Section

The Big Viterbi Decoder (BVD), currently under development for the DSN, includes three separate algorithms to acquire and maintain node and frame synchronization. The first measures the number of decoded bits between two consecutive renormalization operations (renorm rate), the second detects the presence of the frame marker in the decoded bit stream (bit correlation), while the third searches for an encoded version of the frame marker in the encoded input stream (symbol correlation). This article gives a detailed account of the operation, as well as performance comparison, of the three methods.

I. Introduction

This article summarizes the node synchronization (abbreviated as node sync) study for the the Big Viterbi Decoder (BVD) currently under development for the DSN. The function of node sync is to align incoming channel symbols with the code trellis used by a Viterbi decoder and ensure proper phasing between the decoded bit clock and the received symbol clock. The proper phasing of the two clocks is crucial to the operation of the decoder and must be established before synchronized decoding starts, and monitored thereafter.

The present NASA standard concatenated code uses a (7,1/2) convolutional code as its inner code and an 8-bit (255,223) Reed-Solomon (RS) code as its outer code. This system achieves a bit error rate (BER) of 10^{-6} at a bit signal-to-noise ratio (SNR) of 2.53 dB. Recent code search efforts [1,2] show that SNR improvement of up to 2 dB is possible by using a constraint-length 15 convolutional code as the inner code. To demonstrate this performance improvement, an experimental (15,1/4) convolutional encoder was implemented on the Galileo spacecraft, and a programmable convolutional decoder, the BVD hardware prototype, was completed and tested for codes with constraint lengths up to 15 and code rates of $1/N$, $N = 2, 3, \ldots, 6$ [3]. Good node and frame synchronization schemes are essential to the realization of the aforementioned performance offered by long constraint length codes, which are expected to operate at very low SNR [the nominal operational SNR for the (15,1/4) convolutional code concatenated with the (255,223) RS code is E_b/N_0 = 0.9 dB, or equivalently $E_s/N_0 = -5.7$]. Since the outer RS code does not affect the node sync operations of the system, all subsequent discussions in this article address only node sync of the BVD, and SNR always means E_b/N_0 as seen by the BVD.

The BVD provides three separate algorithms to acquire and maintain node synchronization. Each of the three schemes operates in two modes: the acquisition mode, when the decoder is first activated or when a loss-of-sync is detected, and the tracking mode, when the decoder has acquired sync and tries to maintain it. A false declaration of loss-of-sync during tracking, when the decoder is actually in-sync, is a severe offense that causes loss of valuable data and should be avoided. However, when the decoder truly loses sync, it should detect it and correct for it as soon as possible. A general DSN requirement during tracking is for no false loss-of-sync for 24 hr. For Galileo data rates, 134.4 kbits/sec and 115 kbits/sec, and a frame size of 5120 bits, the DSN requirement can be guaranteed by keeping the probability of false declaration of loss-of-sync below 4.4×10^{-7}.

Each of the three schemes is designed to individually meet the DSN requirements of node sync acquisition and tracking, so that the overall proposed system performance, which is based on a combination of the three schemes, is guaranteed to meet the DSN's requirements.

Scheme 1 measures the number of decoded bits between two consecutive renormalizations (renorm rate). It is a continuous operation and assumes no knowledge of the data format. This method can be used for node sync only. Schemes 2 and 3 require that an a priori known bit pattern, the frame marker, be inserted at regular intervals, known as frames, in the bit stream. Scheme 2 detects the presence of the frame marker in the decoded bit stream (bit correlation), while scheme 3 searches for an encoded version of the frame marker in the encoded input stream (symbol correlation). Schemes 2 and 3 are used for both node sync and frame sync. In the acquisition mode, a statistic x is measured at all possible locations of the frame marker, and the observed values of x are used to identify likely locations of the marker. Frame-to-frame verification (as in scheme 2) or integration over several frames (as in scheme 3) is used to reduce the probability of miss and the probability of false alarm at the expense of longer acquisition time. In the tracking mode the statistic x is monitored from frame to frame at the selected marker location to verify continued sync (flywheeling).

The three schemes have their respective advantages and disadvantages. The approach taken in the BVD is to use a combination of the three methods so that overall system performance is optimized. Previous research [4] indicates that each algorithm generates sync statistics that might be more reliable than others in a certain operational environment (e.g., SNR, frame size, operational mode, etc.). The overall node sync performance depends on the selection of the right combination of algorithms under the given operational environment.

The node sync analysis of the aforementioned three schemes involves many different user-chosen system parameters, such as code, frame size, frame-marker size, operating SNR, integration time, and so on. It is impossible to describe all the analysis results for many different combinations of parameters. In this article, the authors only present analysis results for the (15,1/4) code and the (15,1/6) code, with frame length $B = 8960$ bits, frame-marker size $L = 32$ bits, and acquisition time and tracking time (for schemes 2 and 3) equal to 4 frames. The authors have developed some software modules for the three schemes, which, when given the system parameters, generate the required thresholds and the corresponding synchronization performances.

By using the global node-sync procedure suggested in Section V, node sync will be acquired in less than 40,000 bits with a probability of 0.996, and true loss-of-sync will be detected in less than 40,000 bits with a probability of 0.99998.

The rest of the article is organized as follows: Section II describes the operations of scheme 1 and introduces a maximum-likelihood method to estimate the optimal threshold. Sections III and IV describe schemes 2 and 3 and their respective methods to determine the thresholds. Section V gives the performance comparisons of the three node sync schemes, Section VI discusses the robustness of the three schemes with respect to an uncertain SNR estimation, and finally Section VII discusses unresolved issues.

II. Renormalization Rate Scheme (Scheme 1)

This method measures the observable x, the number of bits between two consecutive state metric-renormalization events (renorms), or the mean number of bits between n consecutive renorms (the total number of bits between n consecutive renorms divided by n), and compares x with a preselected threshold. A similar scheme that measures the renorm counts within a long but fixed integration time was analyzed in [9] for the (7,1/2) code.

The renormalization-rate scheme starts with an arbitrary channel-symbol offset and tests the renormalization rate to detect node sync. If the system is determined to be out-of-sync, the offset is changed and the bit clock is changed by one channel symbol. This may need to occur as many as N times. Let H_0 be the hypothesis that the code is in-sync, and H_1 that the code is not. It is observed from the BVD software simulations that the conditional probability density function (pdf) $f(x|H_0)$ is approximately Gaussian with mean m_0 and variance σ_0^2. Similarly, the conditional pdf $f(x|H_1)$ is approximately Gaussian with mean m_1 and variance σ_1^2. That is,

$$f(x|H_0) \approx \frac{1}{\sqrt{2\pi}\sigma_0} \exp^{\frac{(x-m_0)^2}{2\sigma_0^2}}$$

and

$$f(x|H_1) \approx \frac{1}{\sqrt{2\pi}\sigma_1} \exp^{\frac{(x-m_1)^2}{2\sigma_1^2}}$$

These approximated pdf's are especially good when n is large (Central Limit Theorem). Then m_0, m_1, σ_0, and σ_1 can be estimated through software simulation of the metric renorm of the BVD. By assuming $f(x|H_0)$ and $f(x|H_1)$ to be truly Gaussian, the probability of miss, P_M, which is the likelihood that the node sync scheme detects out-of-sync when the system is actually in-sync, is given by

$$P_M = P(x < T|H_0) = \int_{-\infty}^{T} f(x|H_0)dx$$

where the statistic x is compared with a threshold T. Similarly, the probability of false alarm, P_F, which is the likelihood that the node sync scheme detects in-sync when the system is actually out of sync, is given by

$$P_M = P(x > T|H_1) = \int_{T}^{\infty} f(x|H_1)dx$$

In the acquisition mode, node sync is achieved when the observable x is greater than or equal to the acquisition threshold T_1 at the correct location for which H_0 is true, and is smaller than T_1 at the $N-1$ locations for which H_1 is true. By assuming the ideal autocorrelation property of the frame sync marker, one has

$$P_{\text{acq}} = (1 - P_F)^{N-1}(1 - P_M)$$

Tables 1 and 2 tabulate the optimal threshold and its corresponding P_{acq} for various SNR's for the (15,1/4) code and the (15,1/6) code, respectively. Notice that for a given E_b/N_0, T_1 remains the same for different values of n, but P_{acq} falls off rapidly as n increases. The required acquisition time of this scheme, in the worst case, is nN renorm counts, since this scheme requires n renorms for each trial and there are at most N trials for a $1/N$ code. This corresponds to less than 40,000 bits in the worst case.

In the tracking mode, in order to maintain the false loss-of-sync probability $1 - P_{tk}$ below 4.4×10^{-7}, the threshold T_2 is chosen so that

$$1 - P_{tk} = P(x < T_2|H_0) \leq 4.4 \times 10^{-7}$$

The corresponding probability of detecting true loss-of-sync is then given by

$$P(x < T_2|H_1) = \int_{-\infty}^{T_2} f(x|H_1)dx$$

Tables 3 and 4 tabulate the threshold required to maintain the false loss-of-sync probability below 4.4×10^{-7}, and the corresponding probability of detecting true loss-of-sync for the (15,1/4) code and the (15,1/6) code, respectively.

III. Decoded Bit-Correlation Scheme (Scheme 2)

The decoded bit-correlation scheme is the DSN's current method for frame sync. This scheme is now available in the BVD to detect frame sync as well as node sync, because when frame sync is achieved, node sync naturally follows. The frame sync acquisition probabilities and their optimal thresholds for the (15,1/4) code at various SNR's were given in [8]. In this section, the authors evaluate the node sync acquisition and tracking probabilities and their corresponding optimal thresholds for the (15,1/4) code as well as the (15,1/6) code. The bit-correlation scheme alone is not an effective way for acquiring node sync. Node sync has to be achieved before the Viterbi decoder can operate properly, and this scheme operates on the decoded bit stream. To acquire node sync with this method, one needs to detect node sync by performing bit correlation on the B locations of a frame for as many as N possible channel symbol offsets. This may require as many as N frames to detect sync (and one more frame to verify sync), and such a long acquisition time does not meet DSN requirements. This scheme should only be used to verify schemes 1 and 3, that is, to check for the existence of the frame sync marker in the decoded bit stream, after node sync is established by scheme 1 or 3. The subsequent discussion assumes that correct node sync has already been acquired by schemes 1 and/or 3, and evaluates the frame sync acquisition performance of this scheme to see how well it can be used to verify schemes 1 and 3. In the tracking mode, this scheme gives a performance comparable to the other two schemes, and should be considered on an equal basis with the other two schemes.

Scheme 2 compares the true sync word marker to a 32-bit segment of decoded bits. Those bits found to be in disagreement are counted. This count is then compared with a predetermined threshold T_1 optimized for a given SNR. If the number of disagreements is greater than T_1, those 32 bits are rejected as the sync word. Otherwise, the 32 bits are recorded as a sync word candidate. During the acquisition mode, successive one-bit shifts of 32-bit decoded signal segments are compared with the true sync word until the threshold test is passed at the same location in two consecutive frames. In this article, the authors wish to acquire sync with next-frame verification within four frames. Once sync has been declared, the scheme enters into the tracking mode, testing for the sync word is done in a "flywheeling" fashion, that is, by comparing the counts with a preset threshold T_2 only at the presumed frame-marker position. If two consecutive frames fail the threshold test, the scheme will abort the flywheeling operation and switch to the acquisition mode.

The decoded data bits were generated by using the Little Viterbi Decoder (LVD) [5], a hardware decoder that decodes a bit stream at a rate of 60 bps for the (15,1/4) convolutional code, and at a rate of 40 bps for the (15,1/6) code. The received symbols fed into the decoder represent encoded symbols generated from the all-zero information bit sequence corrupted by additive white Gaussian noise (AWGN). The LVD generated enough data to ensure that 100 error bursts were produced for each SNR of interest. For each SNR tested, the decoded bits were subjected to the threshold test for possible threshold values T_1, where $0 \leq T_1 \leq 10$. For a random 32-bit window of decoded bits, this test determines whether the number of decoded bit errors in the observed window exceeds the given threshold. A count is maintained of the number of 32-bit windows with more than T_1 errors. The 32-bit window is then shifted to the right by one bit until all possible 32-bit segments have been tested.

Due to the slow speed of the LVD, the authors only obtained enough data to do the sync analysis for the (15,1/4) code and the (15,1/6) code in the SNR range from 0 to 1 dB. Analysis in the higher SNR range probably requires the use of the BVD. This would involve modifying the software and hardware of the BVD, and that is something the authors would like to do in the near future.

In the decoded bit-correlation scheme, the acquisition probability P_{acq} and the false loss-of-sync probability 1-P_{tk} in the tracking mode are functions of the probability of miss, P_M, and the probability of false alarm, P_F. P_M is the likelihood that the sync word is not detected in the decoded bit stream, which is $P(x \geq T|$ sync marker in the current 32-bit window), where T is a preselected threshold. P_F is the likelihood that the sync word is falsely detected in an incorrect position in the decoded bit stream, which is $P(x < T|$ sync marker not in the current 32-bit window). From the above discussion P_M can be estimated from the LVD error data as

$$P_M = \frac{\text{number of 32-bit windows where the number of errors exceeds } T}{\text{number of 32-bit windows tested within a given file}}$$

Assuming that random data P_F are given by [6],

$$P_F = \sum_{k=0}^{T} \binom{32}{k} 2^{-32}$$

Note that P_M depends on the code and the SNR, but P_F does not. Figures 1 and 2 give the P_M versus P_F curves for the (15,1/4) code and the (15,1/6) code, respectively.

In the acquisition mode, the probability of acquiring sync correctly with next-frame verification within four frames can be approximated for small values of P_M and P_F by [4]

$$P_{\text{acq}} \approx 1 - 3P_M^2 - \frac{B-1}{2} P_F^2$$

where $B = 8960$ is the length of a frame in bits. Tables 5 and 6 give the optimal threshold together with the corresponding P_{acq} for the (15,1/4) code and the (15,1/6) code, respectively. It is observed that in both cases the optimal threshold remains relatively constant for a wide range of SNR.

In the tracking mode, the node sync scheme is required to maintain the probability of false loss-of-sync declaration below 4.4×10^{-7} and to keep a reasonably high probability of true loss-of-sync declaration. On the basis of their extensive simulation results, the authors propose to declare a loss-of-track when five consecutive frames fail the threshold test. To satisfy the stringent DSN requirement of keeping the false loss-of-sync probability below 4.4×10^{-7}, while at the same time keeping a high probability of detecting true loss-of-track, this is equivalent to finding the threshold T_2 so that

$$P_M^5 \leq 4.4 \times 10^{-7}$$

Tables 7 and 8 give the threshold values that achieve the above-required false loss-of-sync probability, together with the corresponding probability of detecting true loss-of-sync during a track, which is $(1 - P_F)^5$, for the (15,1/4) code and the (15,1/6) code, respectively.

IV. Symbol Correlation Scheme (Scheme 3)

The frame-sync acquisition probabilities (scheme 2) and their optimal thresholds for the (7,1/2) code at various SNR's were evaluated in [4]. In this section, the authors evaluate the node-sync acquisition and tracking probabilities and their corresponding optimal threshold of the (15,1/4) code as well as the (15,1/6) code. To describe the node-sync scheme based on channel-symbol measurements, the notation used in [4] is employed. Due to the low SNR operation range of the BVD, this node-sync scheme requires multiple-frame integration, and differs slightly from the frame-sync scheme proposed in [4]. To make this article more self-contained, the authors take into account the multiple-frame integration feature and rederive some of the equations needed for the node-sync analysis. The incoming data bit stream b_i includes both true data bits and sync marker bits λ_i. To simplify subsequent discussion, the encoded sync pattern is assumed to have the ideal autocorrelation property. The data bit stream is packaged into data frames b_i, $i = 1, \ldots, B$, of B bits each, and L sync marker bits λ_i, $i = 1, \ldots, L$, are included in every data frame. The data bit stream is convolutionally encoded by a rate $1/N$, constraint-length K convolutional encoder. The encoded channel symbol stream s_i, $i = 1, \ldots, S$, is likewise partitioned into frames of $S = NB$ symbols each, and each frame includes a set of $M = N(L - K + 1)$ sync marker symbols m_i, $i = 1, \ldots, M$, that are totally determined by the sync marker bits λ_i, $i = 1, \ldots, L$. The remaining $N(B - L + K - 1)$ symbols in each frame depend solely on the true data bits or else on a combination of true data bits and sync marker bits.

The channel symbols are assumed to have constant magnitude s (i.e., $s_i = \pm s$), and they are received through an AWGN channel with noise samples n_i, $i = 1, \ldots, S$, with zero mean and variance σ^2. The ratio $\rho = s^2/\sigma^2$ is an SNR parameter. In terms of ρ, the channel symbol SNR is $E_s/N_0 = \rho/2$, and the bit SNR is $E_b/N_0 = N\rho/2$. The received symbols r_i, $i = 1, \ldots, S$, are passed through a maximum likelihood convolutional decoder (Viterbi decoder) to obtain the decoded bits d_i, $i = 1, \ldots, B$.

Two factors contribute to the difficulties of synchronization using channel symbols. First, the long constraint-length, low-rate codes are designed to operate in a low SNR environment in which the channel symbols are severely corrupted by noise. Second, an inherent drawback of this scheme is that only $L - K + 1$ bits of frame marker are usable in the correlation. The encoded symbols corresponding to the other $K - 1$ bits depend on the previous contents of the encoder shift register. To recover enough SNR for the correlation, integration over j ($j \geq 1$) frames is needed at low SNR and/or when L is small. This is equivalent to increasing the SNR by a factor of j in a one-frame symbol correlation. The sync time, however, is also increased by a factor of j.

Let x be the symbol correlation statistic defined by[1]

$$x = \sum_{i=1}^{M} m_i r_i$$

The BVD symbol-correlation scheme works as follows: When the decoder is first activated, or when a loss-of-sync is detected, the sync system initiates the acquisition mode. The statistic x is measured at all S possible locations of the frame marker, each integrated over j_1 frames, and the observed values of x are compared with a programmable threshold T_1 to identify likely locations of the marker. The statistic x is said to pass the threshold test if $x \geq T_1$. Sync is declared if only one value of x passes the threshold test and the rest fail. If all S values fail the threshold test, or when two or more values of x pass the threshold test, the decoder aborts the sync process and starts a new sync search.

After the decoder has acquired sync, the sync system initiates the tracking mode. The statistic x is tested against a preselected threshold T_2 only at the presumed marker position integrated over j_2 frames. Again, x is said to pass the threshold test if $x \geq T_2$. If x fails the

[1] This "positive correlation" statistic differs slightly from the "negative correlation" statistic used in [4], but it is equivalent in performance and it corresponds to the actual statistic measured by the BVD.

threshold test, the out-of-sync hypothesis is declared, and the sync system switches to acquisition mode.

The BVD actually has the capability of using two thresholds $T_{1,H}$ and $T_{1,L}$ (acquisition mode) or $T_{2,H}$ and $T_{2,L}$ (tracking mode), where $T_{1,L} = -T_{1,H}$ and $T_{2,L} = -T_{2,H}$. The lower threshold $T_{1,L}$ or $T_{2,L}$ is used to identify the sync marker in the case of bit inversion, which results from the operation of the telemetry receiver and cannot be easily overcome. In this article, only single thresholds T_1 (acquisition mode) and T_2 (tracking mode) in the normal non-bit-inversal case are considered. The results are easily extendable to the case of double thresholds.

The general performance expressions in this article are derived for arbitrary combinations of the parameters K, N, L, M, B, and S. The following is the performance analysis of this scheme.

Let x be integrated over j frames. The observed value of this statistic should be near jMs^2 if r_i, $i = 1, \ldots, M$, contains the marker, and otherwise should be near 0. It is therefore natural to compare the observed values of x with a preselected threshold T to make tentative yes–no decisions about the location of the marker, according to whether x falls below or exceeds T. That is

$$x \underset{\text{out-of-sync}}{\overset{\text{in-sync}}{\gtreqless}} T$$

If x falls below T when x is measured at the true position of the marker, the rule gives a tentative decision that misses the sync-marker location. Conversely, if x is at least T when x is not measured at the true marker position, then the decision causes a false detection of sync or false alarm. As in scheme 2, the effectiveness of the sync system can be measured by two competing measures: the probability of miss (P_M) and the probability of false alarm (P_F), which are both functions of T and are defined as

$$P_M = \text{Prob}[x \geq T | \text{sync marker in the current } M \text{ – symbol window}]$$

and

$$P_F = \text{Prob}[x < T | \text{sync marker not in the current } M \text{ – symbol window}]$$

By measuring x at the marker after integrating over j frames, the channel-symbol correlation statistic x can be expressed as

$$x = \sum_{i=1}^{M} m_i(jm_i + n_i^{(j)}) = \sum_{i=1}^{M} u_i$$

where $jm_i + n_i^{(j)}$ is the ith received symbol integrated over j frames, $n_i^{(j)}$ is $N(0, j\sigma^2)$, u_i is $N(js^2, js^2\sigma^2)$, and x/σ^2 is $N(jMs^2/\sigma^2, jMs^2/\sigma^2) = N(jM\rho, Mj\rho)$. Thus, the miss probability P_M is calculated simply as

$$P_M(T, j\rho) = Q\left(\frac{jM\rho - \frac{T}{\sigma^2}}{\sqrt{Mj\rho}}\right)$$

Away from the marker, the correlation statistic x is a sum of conditionally Gaussian random variables, some with zero mean and some with nonzero mean:

$$x = \sum_{i=1}^{M-w} (js^2 + m_i n_i^{(j)}) + \sum_{i=1}^{w} (-js^2 + m_i n_i^{(j)})$$

$$= \sum_{i=1}^{M-w} u_i + \sum_{i=1}^{w} v_i$$

where $i = 1, \ldots, w$ represents the indices of the encoded symbols that differ from the marker symbols; v_i is $N(-js^2,\ js^2\sigma^2)$; and x/σ^2 is $N(j(M-2w)s^2/\sigma^2,\ jMs^2/\sigma^2) = N(j(M-2w)\rho,\ jM\rho)$. The false alarm probability is obtained by averaging the conditional Gaussian probability distribution for x over the discrepancy weight distribution Prob[w],

$$P_F(T, j\rho) = \sum_{w=0}^{M} \text{Prob}[w] Q\left(\frac{2wj\rho - jM\rho + \frac{T}{\sigma^2}}{\sqrt{jM\rho}}\right)$$

where Prob[w] $\approx 2^{-M}\binom{M}{w}$ [7]. Note that both $P_M(T, j\rho)$ and $P_F(T, j\rho)$ depend on the symbol SNR ρ and the integration interval j only in terms of the product $j\rho$, which is the effective symbol SNR after integration over j frames.

Figures 3 and 4 show $P_M(T, j\rho)$ versus $P_F(T, j\rho)$ for the symbol-correlation statistic (frame-marker length $L =$ 32 bits and integration interval $j = 4$) for the (15,1/4) code and the (15,1/6) code, respectively. Note that doubling the integration time is equivalent to increasing the SNR by a factor of 2. Thus, for example, integrating over 4 frames at 0.0 dB has the same performance as integrating over 2 frames at 3.0 dB.

In the acquisition mode, the statistic x is checked at all possible locations of the frame sync marker after integration over j_1 frames. The observed values of x are compared with a preset threshold T_1. Sync is declared if only one of S values of x passes the threshold test ($x \geq T$) and the rest fail ($x < T$). With the assumption that the sync marker sequence has an ideal autocorrelation property, the statistics of x are independent from location to location, and P_M and P_F, as functions of threshold T_1 and integration time j_1 frames, determine the probability of acquisition P_{acq} integrated over j_1 frames as follows:

$$P_{\text{acq}}(T_1, j_1\rho) = \left(1 - P_F(T_1, j_1\rho)\right)^{S-1}\left(1 - P_M(T_1, j_1\rho)\right)$$

An optimal normalized threshold $M\rho - T_1/j_1\sigma^2$ that maximizes $P_{\text{acq}}(T_1, j_1\rho)$ can be evaluated for any given combination of SNR, code rate, frame length, frame-marker size, and integration time. Tables 9 and 10 give the optimal normalized thresholds $M\rho - T_1/j_1\sigma^2$ together with their corresponding P_{acq} with $j_1 = 4$ and $B = 8960$ for the (15,1/4) code and the (15,1/6) code, respectively.

In the tracking mode, the statistic x is integrated over j_2 frames, and is then compared with a preset threshold T_2 only at the presumed frame-marker position (flywheeling). If $x > T_2$, the in-sync hypothesis is assumed, otherwise the decoder declares out-of-sync. Tables 11 and 12 give the normalized thresholds $M\rho - T_2/j_2\sigma^2$ ($j_2 = 4$) and their corresponding probability of detecting true loss-of-sync during a track $1 - P_F(T_2, j_2\rho)$ for the (15,1/4) code and the (15,1/6) code, respectively, that produce a false loss-of-sync probability $P_M(T_2, j_2\rho)$ of 4.4×10^{-7}, with a respective frame marker length of 32 bits and frame length of 8960 bits.

V. Comparisons of the Three Schemes

The proposed BVD approach on node sync is to use a combination of the three methods so that the overall system performance is optimized. This section describes the advantages and disadvantages of the three node sync schemes. This section also suggests a global node sync procedure, which is based on the three node-sync algorithms described above, to declare in-sync and loss-of-sync, both in the acquisition mode and in the tracking mode.

From this study, scheme 1 gives the best node sync performance, both in the acquisition mode and tracking mode. However, a poor estimate of E_b/N_0 can degrade its performance quite rapidly. Scheme 2 alone, as mentioned in Section III, is not an effective method to acquire node-sync due to its long acquisition time. It is, however, more robust than schemes 1 and 3 and can be used to verify and confirm the other two schemes. In fact, when this scheme is used to verify node sync, it performs better than scheme 3 for $E_b/N_0 \geq 0.4$ dB. Scheme 3 gives a reasonably good node-sync performance, though not as good as scheme 1. However, scheme 3 is more robust than scheme 1 in the presence of poor E_b/N_0 estimation. Also, scheme 2 has the extra capability to detect symbol inversion, and it is the only scheme that can perform node sync without convolutional decoding.

Based on the above analysis, the authors propose the following global node sync procedure: In the acquisition mode, node sync is acquired when any two schemes declare sync. That is, when any two schemes pass the threshold tests (threshold values are given in Tables 1, 2, 5, 6, 9, and 10). In the tracking mode, a loss of track is declared when two out of three schemes fail the threshold tests (threshold values are given in Tables 3, 4, 7, 8, 11, and 12). This node sync procedure, which is obtained through computer simulations and modeling analysis, can be refined and improved when the node sync experiments are performed using the actual BVD hardware.

VI. Robustness of the Three Schemes With Respect to SNR

The three node sync schemes discussed above share one common feature: They all measure an observable x and compare it with a threshold T, which is a function of the operating SNR. Knowledge of the operating SNR is required for the three schemes to work properly. A good estimate of SNR is difficult to obtain, especially at a low SNR. However, the authors observe that in the above node sync schemes, either the threshold values change very little for a fairly wide range of SNR (schemes 2 and 3), or the node sync schemes operate so well that any threshold values corresponding to a $\pm$1-dB range of SNR about the operating SNR can provide a synchronization performance that satisfies the DSN requirements (schemes 1 and 3) of fast node sync acquisition and very low false declaration of sync probability. Thus, an SNR estimation with $\pm$1-dB accuracy is sufficient to guarantee the node sync schemes of the BVD to work properly. However, if the SNR estimation is off by more than 1 dB, the node-sync performance can degrade rapidly. Figures 5 and 7 give the

(15,1/4) code-acquisition performance of schemes 1 and 3, respectively, by using the threshold value at 0.5 dB when the true SNR ranges from 0–1.5 dB. Figure 6 gives the (15,1/4) code-acquisition performance of scheme 2 by using the 0.5-dB threshold value when the true SNR ranges from 0–0.8 dB.

VII. Unresolved Issues

(1) Though accurate SNR estimation is not needed to ensure proper node synchronization of the BVD, techniques for SNR estimation, particularly at low SNR operation range, deserve more study as they are required in such other aspects of the BVD operation as choosing the proper stepsize for channel symbol quantization.

(2) In all three schemes, it is possible to avoid SNR estimation and at the same time preserve a good sync acquisition performance by using the best-match approach instead of the threshold approach. The threshold approach, and thus SNR estimation, is still required in the tracking mode. The best-match approach requires storing all the observables that correspond to different synchronization offsets and picking the sync location that gives the most favorable observable. This increases the hardware and software complexity to various degrees (with respect to the threshold implementation), depending on the synchronization schemes. The best-match strategy is, in fact, equivalent to maximum-likelihood synchronization estimation, and can be shown to give better performance than the threshold strategy. Mathematical analysis and performance simulation of the three schemes using the best-match strategy are tedious and will not be given in this article.

The issue of implementation complexity is discussed as follows: Scheme 1 has only N different synchronization offsets and requires storing N values and choosing among N values. The increase in complexity is minimal. In fact, the current BVD prototype uses the best-match strategy in software to perform initial node sync acquisition in testing. However, best-match strategy for schemes 2 and 3 is not available in the BVD. Schemes 2 and 3 perform node sync as well as frame sync, and the use of the best-match strategy can enhance both node sync and frame sync. Since there are B possible sync offset locations in the decoded bit stream, scheme 2 requires a buffer of size B. Also scheme 2 requires the ability to pick in real time the most favorable observable among an array of B values. These functions do not yet exist in the current BVD frame-synchronization subsystem. Scheme 3 is similar to scheme 2. It requires a buffer of size NB. However, since scheme 3 allows channel symbol integration over a few frames, this buffer exists in the current frame-synchronization subsystem and can be tapped into to perform the best-match function. Thus, the three sync schemes can be modified as suggested above to avoid SNR estimation and at the same time improve acquisition performance, since the best-match approach is in fact a maximum-likelihood strategy.

(3) The current DSN method for frame sync, which is a bit-correlation scheme, uses a "test and verify" strategy for frame sync acquisition. That is, the unique sync candidate from any frame is chosen as the sync location if and only if it is verified once in the next succeeding frame by repassing the threshold test at the corresponding location within that frame. This scheme requires making two separate and independent frame sync decisions, one frame after the other. Intuitively, this scheme is inferior to the scheme that takes into account the bit-correlation information of two successive frames before making a decision on frame sync acquisition. One can call this scheme an "integrate and test" strategy. This strategy, in a rough sense, is equivalent to performing decoded bit correlation by using a frame marker of size $2L$ over a frame length of size $2B$. Table 13 tabulates the optimal threshold and its corresponding P_{acq} for various SNR for the (15,1/4) code by using the new decoded bit-correlation scheme. Figure 8 compares the P_{acq} performances between the old "test and verify" strategy and the new "integrate and test" strategy. The new strategy is uniformly better than the old scheme by about a factor of 2 on the probability of not acquiring sync within four frames. The additional hardware required is a buffer of size B, as there are B possible sync locations and the scheme requires storing a temporary correlation observable for each location in every other frame. On the other hand, this scheme does not require the logic to perform frame-to-frame verifications.

(4) If SNR estimation cannot be achieved to within ± 1-dB accuracy, either more sophisticated node-sync schemes [as suggested in issue (2)] should be used, or a 64-bit frame marker should be used. Figure 9 gives the (15,1/4) code-acquisition performance of scheme 2 using an 0.8-dB threshold value and a 64-bit sync marker when the true SNR ranges from 0–0.8 dB, and Fig. 10 gives the (15,1/4) code-acquisition performance of scheme 3 by using a 2-dB threshold value and a 64-bit sync marker when the true SNR ranges from 0–4 dB.

References

[1] J. H. Yuen and Q. D. Vo, "In Search of a 2-dB Coding Gain," *TDA Progress Report 42-83*, vol. July–September 1985, Jet Propulsion Laboratory, Pasadena, California, pp. 26–33, November 15, 1985.

[2] S. Dolinar, "A New Code for Galileo," *TDA Progress Report 42-93*, vol. January–March 1988, Jet Propulsion Laboratory, Pasadena, California, pp. 83–96, May 15, 1988.

[3] J. Statman, G. Zimmerman, F. Pollara, and O. Collins, "A Long Constraint Length VLSI Viterbi Decoder for the DSN," *TDA Progress Report 42-95*, vol. July–September 1988, Jet Propulsion Laboratory, Pasadena, California, pp. 134–142, November 15, 1988.

[4] S. Dolinar and K. Cheung, "Frame Synchronization Methods Based on Channel Symbol Measurements," *TDA Progress Report 42-98*, vol. April–June 1989, Jet Propulsion Laboratory, Pasadena, California, pp. 121–137, August 15, 1989.

[5] C. Lahmeyer and K. Cheung, "Long Decoding Runs for Galileo's Convolutional Codes," *TDA Progress Report 42-95*, vol. July–September 1988, Jet Propulsion Laboratory, Pasadena, California, pp. 143–146, November 15, 1988.

[6] L. Swanson, *A Comparison of Frame Synchronization Methods,* JPL Publication 82-100, Jet Propulsion Laboratory, Pasadena, California, December 15, 1982.

[7] K. Cheung, "The Weight Distribution and Randomness of Linear Codes," *TDA Progress Report 42-97*, vol. January–March 1989, Jet Propulsion Laboratory, Pasadena, California, pp. 208–215, August 15, 1989.

[8] S. Arnold and L. Swanson, "Frame Synchronization for the Galileo Code," *TDA Progress Report 42-104*, vol. October–December 1990, Jet Propulsion Laboratory, Pasadena, California, pp. 211–218, February 15, 1991.

[9] U. Cheng, "Node Synchronization of Viterbi Decoders Using State Metrics," *TDA Progress Report 42-94*, vol. April–June 1988, Jet Propulsion Laboratory, Pasadena, California, pp. 201–209, August 15, 1988.

Table 1. (15,1/4) code-acquisition performance of scheme 1.

SNR, dB	Renorm counts, n	Threshold, T_1	$1 - P_{acq}$
0.1	1	1749	7.69×10^{-10}
0.1	2	1746	$< 1.00 \times 10^{-16}$
0.5	1	1814	7.99×10^{-15}
0.5	2	1806	$< 1.00 \times 10^{-16}$
1.0	1	1901	$< 1.00 \times 10^{-16}$
1.0	2	1894	$< 1.00 \times 10^{-16}$
1.5	1	2001	$< 1.00 \times 10^{-16}$
1.5	2	1985	$< 1.00 \times 10^{-16}$
2.0	1	2106	$< 1.00 \times 10^{-16}$
2.0	2	2100	$< 1.00 \times 10^{-16}$
2.5	1	2221	$< 1.00 \times 10^{-16}$
2.5	2	2184	$< 1.00 \times 10^{-16}$
3.0	1	2305	$< 1.00 \times 10^{-16}$
3.0	2	2275	$< 1.00 \times 10^{-16}$

Table 3. (15,1/4) code-tracking performance of scheme 1.

SNR, dB	Renorm counts, n	Threshold, T_2	$1 - P$ (detect true loss-of-track)
0.1	1	1812	$< 1.00 \times 10^{-16}$
0.1	2	1886	$< 1.00 \times 10^{-16}$
0.5	1	1995	$< 1.00 \times 10^{-16}$
0.5	2	2087	$< 1.00 \times 10^{-16}$
1.0	1	2172	$< 1.00 \times 10^{-16}$
1.0	2	2256	$< 1.00 \times 10^{-16}$
1.5	1	2527	$< 1.00 \times 10^{-16}$
1.5	2	2616	$< 1.00 \times 10^{-16}$
2.0	1	2927	$< 1.00 \times 10^{-16}$
2.0	2	3040	$< 1.00 \times 10^{-16}$
2.5	1	3456	$< 1.00 \times 10^{-16}$
2.5	2	3558	$< 1.00 \times 10^{-16}$
3.0	1	4047	$< 1.00 \times 10^{-16}$
3.0	2	4173	$< 1.00 \times 10^{-16}$

Table 2. (15,1/6) code-acquisition performance of scheme 1.

SNR, dB	Renorm counts, n	Threshold, T_1	$1 - P_{acq}$
0.0	1	756	5.44×10^{-4}
0.0	2	754	7.30×10^{-7}
0.5	1	797	7.26×10^{-7}
0.5	2	792	5.09×10^{-13}
1.0	1	855	6.26×10^{-12}
1.0	2	849	$< 1.00 \times 10^{-16}$
1.5	1	886	$< 1.00 \times 10^{-16}$
1.5	2	882	$< 1.00 \times 10^{-16}$
2.0	1	952	$< 1.00 \times 10^{-16}$
2.0	2	949	$< 1.00 \times 10^{-16}$
2.5	1	965	$< 1.00 \times 10^{-16}$
2.5	2	960	$< 1.00 \times 10^{-16}$
3.0	1	1065	$< 1.00 \times 10^{-16}$
3.0	2	1054	$< 1.00 \times 10^{-16}$

Table 4. (15,1/6) code-tracking performance of scheme 1.

SNR, dB	Renorm counts, n	Threshold, T_2	$1 - P$ (detect true loss-of-track)
0.0	1	717	3.88×10^{-1}
0.0	2	753	8.25×10^{-8}
0.5	1	795	1.10×10^{-7}
0.5	2	838	$< 1.00 \times 10^{-16}$
1.0	1	906	$< 1.00 \times 10^{-16}$
1.0	2	953	$< 1.00 \times 10^{-16}$
1.5	1	1020	$< 1.00 \times 10^{-16}$
1.5	2	1078	$< 1.00 \times 10^{-16}$
2.0	1	1172	$< 1.00 \times 10^{-16}$
2.0	2	1225	$< 1.00 \times 10^{-16}$
2.5	1	1334	$< 1.00 \times 10^{-16}$
2.5	2	1407	$< 1.00 \times 10^{-16}$
3.0	1	1548	$< 1.00 \times 10^{-16}$
3.0	2	1619	$< 1.00 \times 10^{-16}$

Table 5. (15,1/4) code-acquisition performance of scheme 2.

SNR, dB	Threshold, T_1	$1 - P_{acq}$
0.0	6	8.59×10^{-3}
0.1	6	5.97×10^{-3}
0.2	6	3.29×10^{-3}
0.3	5	1.44×10^{-3}
0.4	5	7.00×10^{-4}
0.5	5	4.60×10^{-4}
0.6	5	2.60×10^{-4}
0.7	5	1.10×10^{-4}
0.8	4	5.00×10^{-5}

Table 6. (15,1/6) code-acquisition performance of scheme 2.

SNR, dB	Threshold, T_1	$1 - P_{acq}$
0.1	5	1.79×10^{-3}
0.3	5	4.94×10^{-4}
0.5	5	1.27×10^{-4}
0.7	4	3.35×10^{-5}
0.9	4	7.90×10^{-6}
1.1	3	1.70×10^{-6}

Table 7. (15,1/4) code-tracking performance of scheme 2.

SNR, dB	Threshold, T_2	$1 - P$ (detect true loss-of-track)
0.0	8	2.77×10^{-2}
0.1	7	8.38×10^{-3}
0.2	7	8.38×10^{-3}
0.3	7	8.38×10^{-3}
0.4	7	8.38×10^{-3}
0.5	7	8.38×10^{-3}
0.6	7	8.38×10^{-3}
0.7	7	8.38×10^{-3}
0.8	7	8.38×10^{-3}

Table 8. (15,1/6) code-tracking performance of scheme 2.

SNR, dB	Threshold, T_2	$1 - P$ (detect true loss-of-track)
0.1	7	8.38×10^{-3}
0.3	6	2.14×10^{-3}
0.5	6	2.14×10^{-3}
0.7	6	2.14×10^{-3}
0.9	6	2.14×10^{-3}
1.1	6	2.14×10^{-3}

Table 9. (15,1/4) code-acquisition performance of scheme 3.

SNR, dB	Threshold, T_1	$1 - P_{acq}$
0.0	8.7	3.69×10^{-3}
0.5	9.8	2.06×10^{-3}
1.0	11.0	1.12×10^{-3}
1.5	12.3	6.01×10^{-4}
2.0	13.7	3.15×10^{-4}
2.5	15.2	1.62×10^{-4}
3.0	16.9	8.21×10^{-5}

Table 10. (15,1/6) code-acquisition performance of scheme 3.

SNR, dB	Threshold, T_1	$1 - P_{acq}$
0.0	10.1	7.14×10^{-4}
0.5	11.4	3.21×10^{-4}
1.0	12.8	1.38×10^{-4}
1.5	14.4	5.74×10^{-5}
2.0	16.1	2.30×10^{-5}
2.5	17.9	8.90×10^{-6}
3.0	19.8	3.34×10^{-6}

Table 11. (15,1/4) code-tracking performance of scheme 3.

SNR, dB	Threshold, T_2	$1 - P$ (detect true loss-of-track)
0.0	14.8	1.90×10^{-5}
0.5	15.6	6.32×10^{-6}
1.0	16.6	2.04×10^{-6}
1.5	17.5	6.50×10^{-7}
2.0	18.6	2.00×10^{-7}
2.5	19.7	6.00×10^{-8}
3.0	20.8	1.00×10^{-8}

Table 12. (15,1/6) code-tracking performance of scheme 3.

SNR, dB	Threshold, T_2	$1 - P$ (detect true loss-of-track)
0.0	14.8	1.59×10^{-6}
0.5	15.6	3.50×10^{-7}
1.0	16.6	7.00×10^{-8}
1.5	17.5	1.00×10^{-9}
2.0	18.6	$< 1.00 \times 10^{-16}$
2.5	19.7	$< 1.00 \times 10^{-16}$
3.0	20.8	$< 1.00 \times 10^{-16}$

Table 13. (15,1/4) code-acquisition performance of the new scheme.

SNR, dB	Threshold, T_1	$1 - P_{acq}$
0.0	13	3.97×10^{-3}
0.2	13	1.28×10^{-3}
0.4	12	3.29×10^{-4}
0.6	12	7.55×10^{-5}
0.7	12	3.69×10^{-5}
0.8	11	1.40×10^{-5}

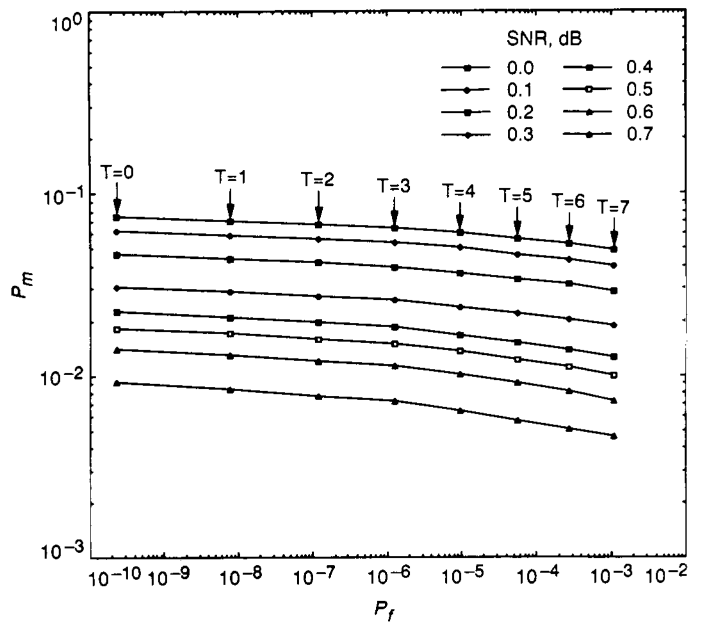

Fig. 1. P_m versus P_f (frame length = 8960 bits).

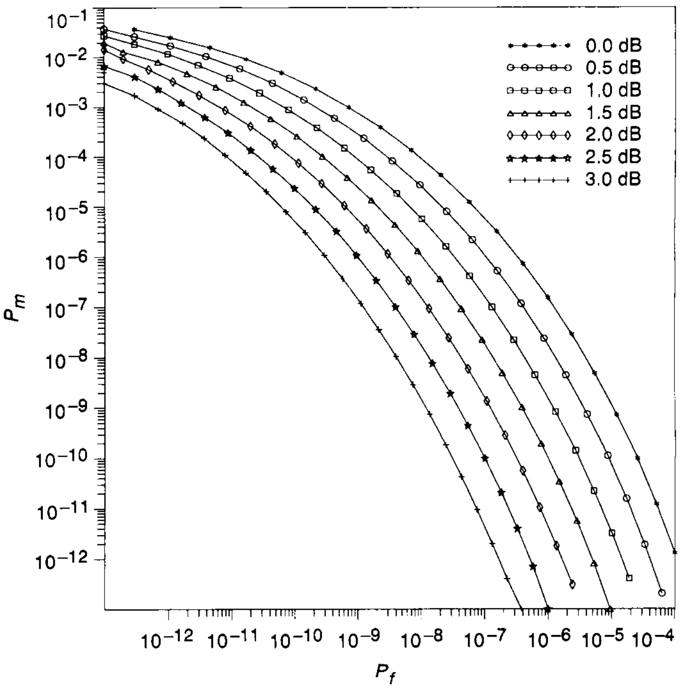

Fig. 3. P_m versus P_f for the (15,1/4) code (J = 4).

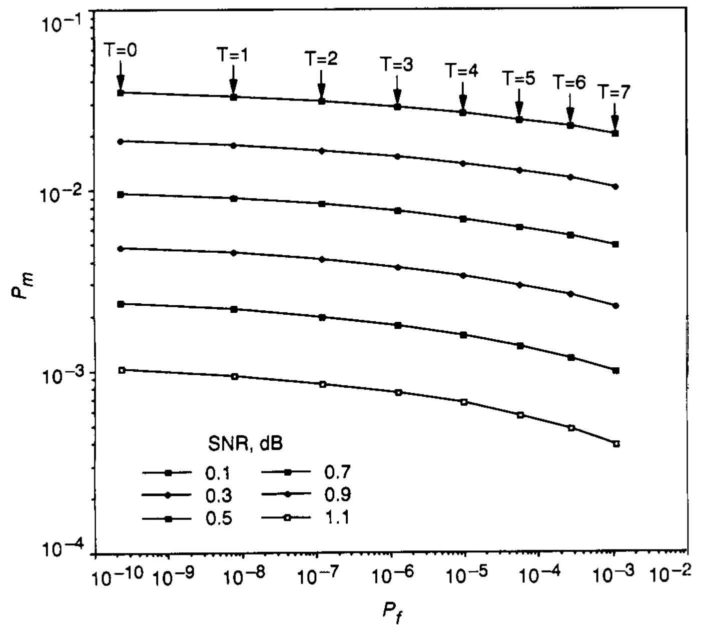

Fig. 2. P_m versus P_f (frame length = 8960 bits).

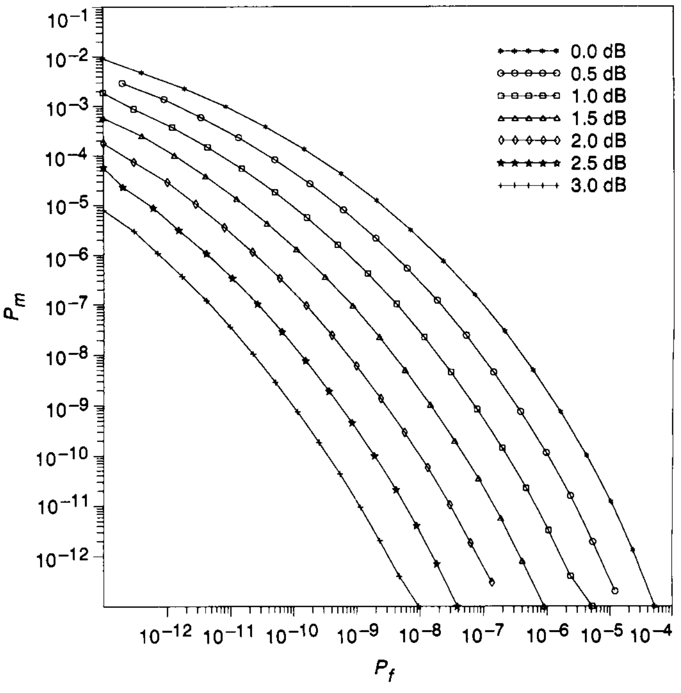

Fig. 4. P_m versus P_f for the (15,1/6) code (J = 4).

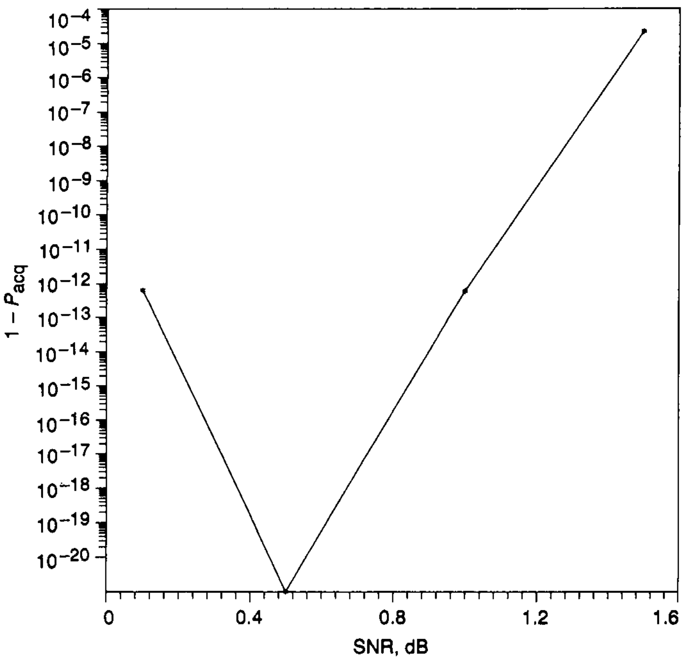

Fig. 5. (15,1/4) code-acquisition performance of scheme 1 using a fixed threshold from 0–1.5 dB.

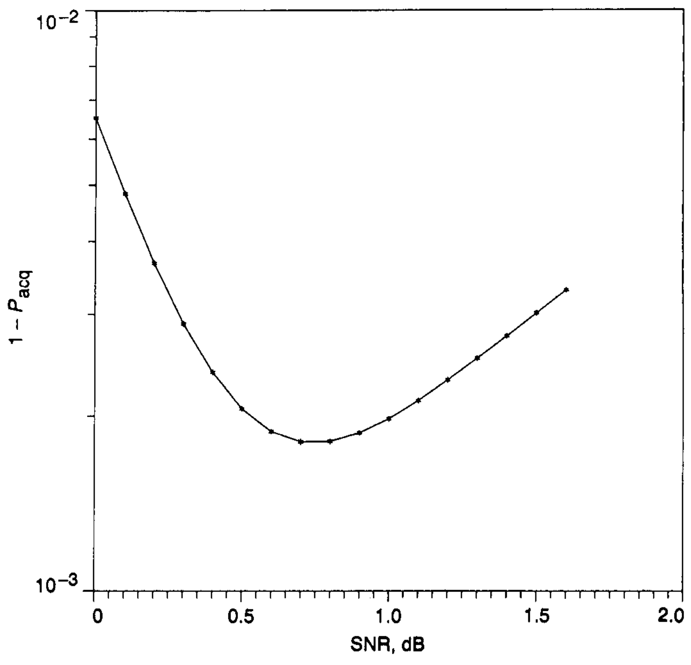

Fig. 7. (15,1/4) code-acquisition performance of scheme 3 using a fixed threshold from 0–1.5 dB.

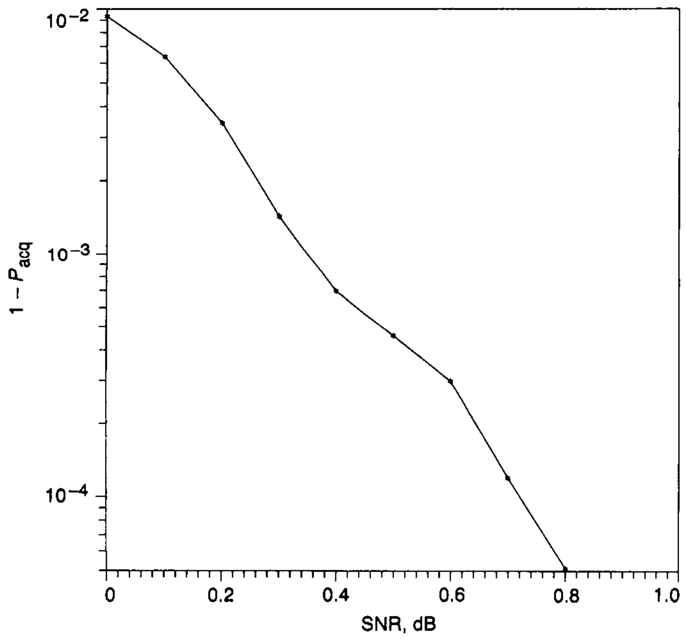

Fig. 6. (15,1/4) code-acquisition performance of scheme 2 using a fixed threshold from 0–0.8 dB.

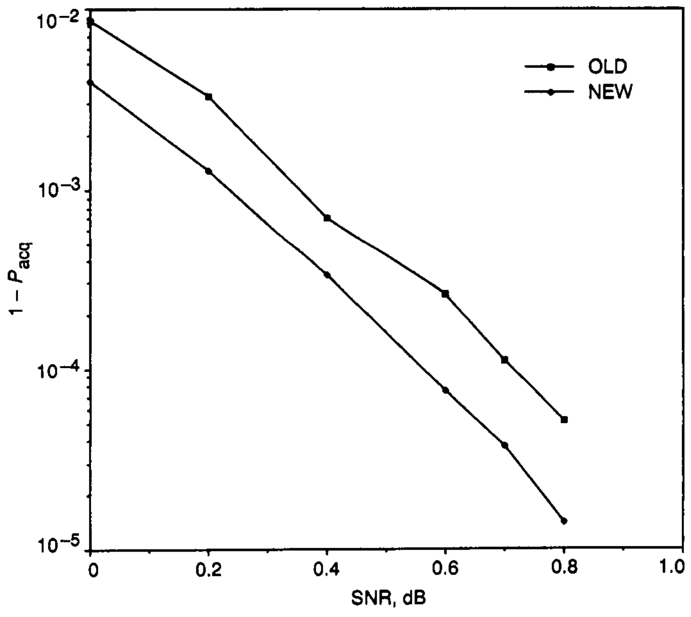

Fig. 8. P_{acq} of the old and new strategies.

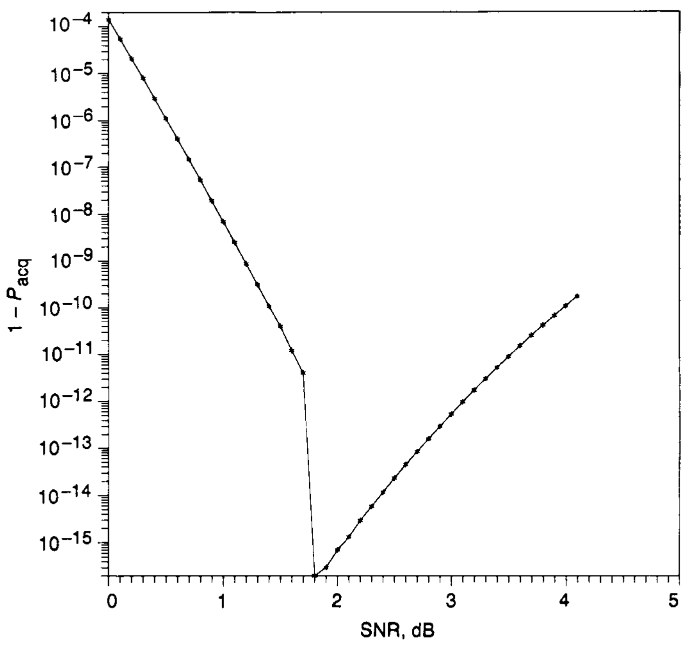

Fig. 9. (15,1/4) code-acquisition performance of scheme 3 using a fixed threshold from 0–4 dB (64-bit sync marker).

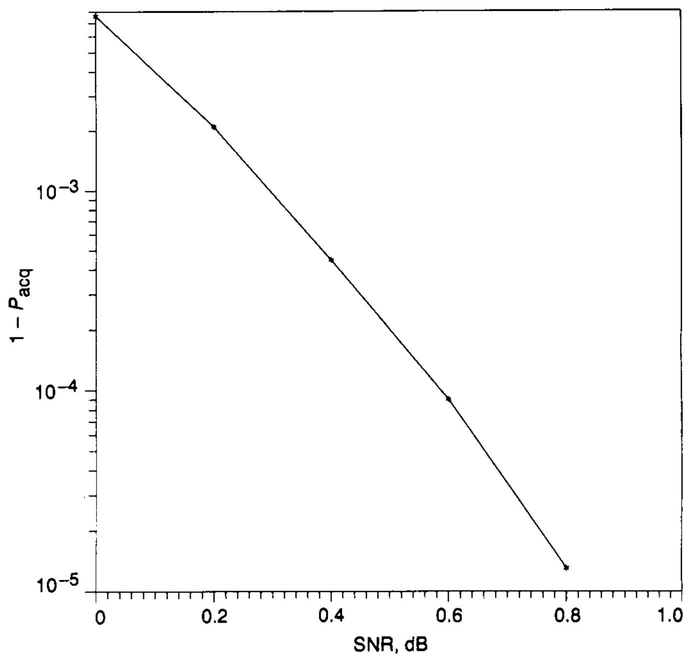

Fig. 10. (15,1/4) code-acquisition performance of scheme 2 using a fixed threshold from 0–0.8 dB (64-bit sync marker).