ADVANCED MODULATION AND CODING STRATEGIES FOR FADING CHANNEL COMMUNICATIONS

FINAL REPORT

MCS MILLER COMMUNICATIONS SYSTEMS LTD.

ADVANCED MODULATION AND CODING STRATEGIES FOR FADING CHANNEL COMMUNICATIONS

FINAL REPORT

Contract No.: 36001-7-3579/01-ST
MCS File No.: 8832
SSC File No.: 04ST.36001-7-3579
Date: April 28, 1988

Prepared by: _____
P. Idez for M. Moher

Approved by: _____
B. A. Mazur

SUBMITTED BY:
MCS MILLER COMMUNICATIONS SYSTEMS LTD.
300 Legget Drive,
Kanata, Ontario,
Canada K2K 1Y5

TABLE OF CONTENTS

1.0 INTRODUCTION

This study consisted of three parts. The first part involved determining the parameters and evaluating the performance of a demodulator (for CPM signals), that is based on maximum likelihood detection principles for reception over Rayleigh fading channels. The second part involved evaluating the performance of concatenating a simple 8-state TCM modem with a Reed-Solomon code for the reliable transmission of data over Rician fading channels. The third part of the study was to prepare a software documentation manual for the entire software package, including software that was developed under previous contracts.

An approach based upon the maximum likelihood detection of constant envelope modulation (CPM) schemes, transmitted over flat-fading Rayleigh channels, had been studied previously (in work funded by Transport Canada) for the detection of Aviation Binary Phase Shift Keying transmitted over Rician fading channels [1]. One of the purposes of the first part of this study was to determine the performance of this detection approach when applied to a CPM scheme transmitted over a Rayleigh fading channel. While the proposed approach is applicable to virtually any CPM scheme, the example scheme chosen here was one that had been recently developed by researchers at the University of Toronto, known as N32FM [2]. The performance for the proposed approach was compared to that of differential detection preceded by linear phase lowpass filtering (assuming a complex baseband implementation).

The second part of this study was to determine, using computer simulation, the performance of concatenating a simple 8-state trellis code [3] with a Reed-Solomon (RS)

code, for the reliable transmission of data over fading channels. It is desirable that the symbols for the RS code be 8-bit symbols (i.e. elements of GF(256)). Another desirable characteristic of the RS code is that the number of information bits in a packet be a multiple of the 96 bit INMARSAT signal unit. An example of such a code is the (240,180) shortened RS code (shortened from (255,195)). This code is a rate 3/4 code, that contains 15 signal units of data, and is capable of correcting any packet with no more than 30 bytes in error! Even better performance can be achieved if the TCM decoder can pass suitable erasure information to the RS decoder. The error correction capability (i.e., BER performance) of the above scheme was evaluated by computer simulation. However, almost all of the remaining uncorrectable errors can be detected by the RS decoder. The error detection capability of the code was also estimated.

In the past few years MCS has developed a massive software package, under funding from the Government of Canada, for determining the fading-channel performance of advanced modulation and coding schemes. While this software has been documented on a contract-by-contract basis, no single up-to-date comprehensive documentation exists. The third part of this study was to prepare a comprehensive "Software User's Guide".

2.0 MAXIMUM LIKELIHOOD DETECTION OF CPM SIGNALS

In this section, we discuss techniques for the maximum likelihood sequence estimation (MLSE) of continuous phase modulated signals. The simulated performance of the derived technique is compared to differential detection over a number of different Rayleigh fading channels.

2.1 CPM Signals and N32FM Pulse Shaping

Continuous phase modulated signals are strong candidates for power-limited communications systems such as mobile radio because of their constant envelope properties. In addition, proper choice of the modulating waveform will result in a quite narrow transmit spectrum with remarkably low out-of-band energy. This is also important in such applications as mobile radio, which are subject to the "near-far" problem. In this section we will briefly describe CPM signals, and, in particular, the N32FM pulse shape.

The generic form for a CPM signal is

$$x(t) = A \cos(2\pi f_c t + \phi(t) + \phi_o) \tag{2.1}$$

where f_c is the carrier frequency and ϕ_o is a constant phase offset. The information bearing excess phase function $\phi(t)$ can be expressed as

$$\phi(t) = 2\pi h \int_{-\infty}^{t} \sum_{m} a_m f(s-mT)\, ds \tag{2.2}$$

where a_m is a data symbol, f(t) is the frequency pulse and h is the modulation index. The following normalization constraint is used to define the modulation index h.

$$\int_{-\infty}^{\infty} f(t)\, dt = \frac{1}{2} \tag{2.3}$$

From [2], we have that the pulse shape f(t), that satisfies both Nyquist's second and third criteria for no intersymbol interference, has a frequency response given by

$$N_{32}(f) = \begin{cases} \dfrac{1 + T|f|\ (\mathrm{sinc}(2fT)-1)}{\mathrm{sinc}(fT)} & |f| \leq \dfrac{1}{T} \\ 0 & |f| \geq \dfrac{1}{T} \end{cases} \tag{2.4}$$

where T is the symbol period. Using this pulse shape with FM modulation produces a scheme called N32FM. The spectral density of N32FM can be shown to fall off asymtotically at the rate of $1/|f-f_c|^6$, due to zero crossings at time constants which are odd multiples of T/2 [2]. This is superior to most other CPM pulse shapes where the asymptotic spectrum is proportional to $1/|f-f_c|^4$ or more. The modulation index is usually chosen as h=1/2 so that the CPM signal may be coherently demodulated [2].

Note that when truncating the N32FM pulse in an FIR implementation one must be careful to choose the truncation length such that the Nyquist second and third criteria are still satisfied.

Nyquist's third criteria is embodied in the constraint

$$\int_{\frac{(2k-1)T}{2}}^{\frac{(2k+1)T}{2}} f(t)\,dt = \begin{cases} \frac{1}{2} & k = 0 \\ 0 & k \neq 0 \end{cases} \tag{2.5}$$

Substituting this into the excess phase function $\phi(t)$, equation (2.2), and setting $t = \frac{(2n+1)T}{2}$, we obtain

$$\phi_n := \phi\left(\frac{(2n+1)T}{2}\right) = 2\pi h \sum_{m} \frac{a_m}{2}$$

$$= \frac{\pi}{2} \sum_{m=1}^{n} a_m. \tag{2.6}$$

where $a_m = \pm 1$ and $h = 1/2$. Thus we have

$$\phi_n = \begin{cases} k\pi & n \text{ even} \\ \left(\frac{k+1}{2}\right)\pi & n \text{ odd} \end{cases} \tag{2.7}$$

for some $k \in z$ (k is a function of n). If we form the difference $\phi_n - \phi_{n-1}$, we can recover the data sequence (a_m).

In practice, this is not as easy as described above. If one coherently demodulates the signal then one is left with the excess phase function $\psi(t) = \cos \phi(t) + j \sin \phi(t)$ in complex baseband representation. This still has the property

$$\psi\left(\frac{(2n+1)T}{2}\right) = \begin{cases} \pm 1 & n \text{ even} \\ \pm j & n \text{ odd} \end{cases} \tag{2.8}$$

and can be used with differential decoding in the absence of noise. However, if noise is present it will distort

the above situation due to the attendant filters. The problem of doing optimum filtering under these circumstances is investigated in [4].

In our situation, however, we are interested in performing differential detection of the N32FM waveform, which is, in fact, a simpler problem, given the above observations.

Given the rapid fall off of the spectrum of N32FM, intuition would suggest that using a rectangular bandpass filter followed by a standard differential detection circuit would provide close to optimal differential detection. The bandpass filter should be just wide enough to pass the modulated signal without significant distortion (this should include any modulation due to fading). For example, a bandpass filter which passes 99% of the signal energy should be more than sufficiently wide. (Note that it was required for this study that we test fade rates up to 0.3 times the bit rate.) With this formulation the signal processing is almost identical to standard differential detection as shown in Figure 2.1.

For N32FM, performance with differential detection should be close to ideal differential detection with 2-DPSK. There would seem to be no advantage to going to advanced noncoherent detection schemes such as those discussed in [5] and [6], using modified discriminator detection. These schemes are intended for the case of correlative coding, and for those cases where the phase differential between adjacent symbols is less than $\pi/2$. Neither of these situations apply here.

2.2 MLSE of CPM Signals over Rayleigh Fading Channels

In [1], a technique is developed for the maximum likelihood detection of a data sequence corresponding to a constant

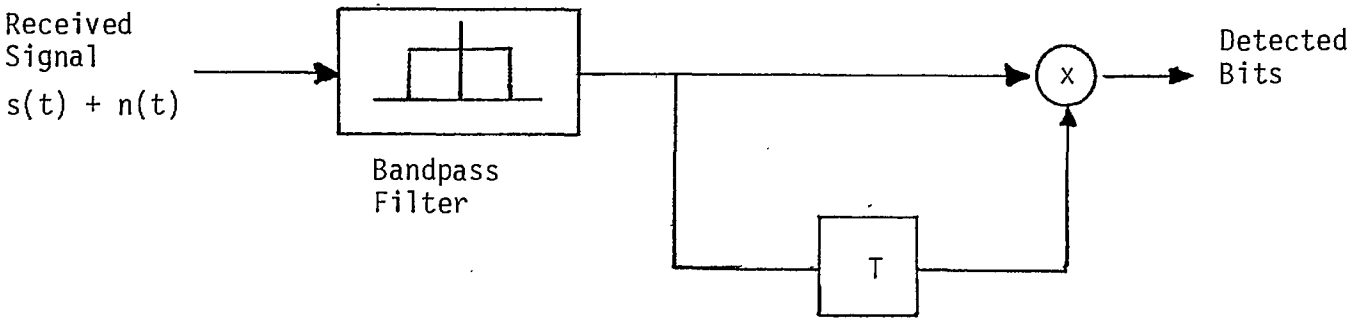

Figure 2.1 A differential detection scheme for CPM with a frequency pulse shape satisfying Nyquist's second and third criteria.

envelope signal transmitted over a Rayleigh fading channel. The technique is based on the observation that, if one multiplies the received signal by the conjugate of the transmitted signal, then the resulting signal is a Gaussian process which is independent of the data and has a spectrum which is determined solely by the fading process and the noise process. If the spectrum of this Gaussian process can be modeled as an all-pole filter, then maximum likelihood detection is equivalent to choosing the data sequence (transmitted signal) which minimizes the squared errors of the corresponding forward linear predictor.

Mathematically, we wish to choose the transmitted signal $x(t,I)$, which is a function of the data sequence $I = (\ldots, a_0, a_1, a_2, \ldots)$, that minimizes

$$J'(I) = \int_{-\infty}^{\infty} \left[\int_{-\infty}^{t} h(t-s)\, \overline{x(s,I)}\, y(s)\, ds \right]^2 dt \qquad (2.9)$$

where $y(t)$ is the received sequence and $h(t)$ is the linear prediction error filter corresponding to the fading plus noise process. (All signals are assumed to be in complex baseband representation.) In the discrete time or sampled scenario in which we will be working this becomes

$$J(I) = \sum_{j=-\infty}^{\infty} \left[\sum_{i=0}^{\infty} h_i\, \overline{x_{j-i}(I)}\, y_{j-i} \right]^2 \qquad (2.10)$$

Under the assumption of a causal frequency pulse of finite duration, it is possible to perform the minimization of (2.10) using a dynamic programming approach. That is, we assume the frequency pulse shape $f(t)$ satisfies

$$f(t) = 0 \quad \text{for } t<0 \text{ and } t>\ell T \tag{2.11}$$

where T is the symbol period. Under this condition, the transmitted CPM signal, in complex baseband representation, is

$$x(t,I) = A \exp\{j\, 2\pi h \int_{-\infty}^{t} \sum_{m=n-\ell}^{n} a_m f(s-mT)ds\} \tag{2.12}$$

for $(n-1)T \leq t \leq nT$. Clearly, $x(t,I)$ (and the sampled version $x_i(I)$) only depends on the data sequence up to a_n, that is, on $I_n = (\ldots, a_{n-2}, a_{n-1}, a_n)$.

Assume that, in the sampled version, there are r samples per symbol period. Then define

$$J_n(I) = \sum_{j=-\infty}^{nr} \Big[\sum_{i=0}^{\infty} h_i \overline{x_{j-i}(I)}\, y_{j-i} \Big]^2 \tag{2.13}$$

$$= \sum_{j=-\infty}^{(n-1)r} \Big[\sum_{i=0}^{\infty} h_i \overline{x_{j-i}(I)}\, y_{j-i} \Big]^2 + \sum_{j=(n-1)r+1}^{nr} \Big[\sum_{i=0}^{\infty} h_i \overline{x_{j-i}(I)}\, y_{j-i} \Big]^2 \tag{2.14}$$

There are several observations to be made about equations (2.13) and (2.14). The first is that, because of (2.12), J_n depends only upon the data symbols up to and including the n'th, that is, $J_n(I) = J_n(I_n)$. Secondly, we note that the first summation of (2.14) is simply $J_{n-1}(I_{n-1})$. Thus (2.14) can be written as

$$J_n(I_n) = J_{n-1}(I_{n-1}) + B_n(I_n) \tag{2.15}$$

where $B_n(I_n)$ corresponds to the branch metric in the Viterbi decoding algorithm

$$B_n(I_n) = \sum_{j=(n-1)r+1}^{nr} \left[\sum_{i=0}^{\infty} h_i \overline{x_{j-i}(I_n)}\, y_{j-i} \right]^2 \qquad (2.16)$$

In theory, J(I) can be determined by calculating $J_n(I_n)$ using (2.15). The practicality of this approach, however, depends on the complexity of the branch metric (2.16). If we assume a linear predictor of order N, then

$$h_i = 0 \quad \text{for } i<0 \text{ and } i>N+1 \qquad (2.17)$$

Under this assumption, consider the j = nr term of

$$B_n(I_n) = \sum_{j=(n-1)r+1}^{nr} \left[\sum_{i=0}^{N} h_i \overline{x_{j-i}(I_n)}\, y_{j-i} \right]^2 \qquad (2.18)$$

For $0 \leq i \leq r-1$, equation (2.12) shows that x_{nr-i} depends on $(a_{n-\ell}, a_{n-\ell+1}, \ldots, a_n)$. In general, for $kr \leq i \leq (k+1)r-1$, x_{nr-i} depends on $(a_{n-\ell-k}, \ldots, a_{n-k})$. If we assume the predictor order $N = pr-1$, for some p, and consider all the possible j in equation (2.18), we see that $B_n(I_n)$ depends only on the bits $(a_{n-\ell-p}, a_{n-\ell-p+1}, \ldots, a_n)$. That is, the branch metric at stage n depends only on the $\ell+p+1$ most recent bits. Because of this, the program can be set up as a dynamic programming problem with $2^{\ell+p}$ states. The metric cumulative function for state p is given by

$$J_n^p = \min_q \{J_n^q + B_n(q,p)\} \qquad (2.19)$$

where $B_n(q,p)$ is the branch metric from state q to p, that is, from $(a_{n-\ell-p}, \ldots, a_{n-1})$ to $(a_{n-\ell-p+1}, \ldots, a_n)$. The optimum sequence histories which lead to a given state are stored, and the decision for the most likely bit in a given position is made after the sequences converge.

2.3 Implementation of MLSE of CPM Over Rayleigh Fading Channels

In the previous section, the theoretical details of MLSE of CPM signals over Rayleigh fading channels were explained and it was found that they result in a trellis (finite state machine) with $2^{\ell+p}$ states, where ℓ is the duration of the frequency pulse shape in symbol periods and p is the duration of the linear predictor used to model the fading spectrum, also in symbol periods. The implementation complexity is proportional to the number of states. It is also proportional to the complexity of computing a branch metric, which is also a function of the order of the linear predictor. To make an implementation practical, we wish to minimize the complexity as much as possible.

We first tackled the problem of minimizing p, the duration of the frequency pulse. We have chosen the N32FM waveform as the frequency pulse shape because of its superior spectral properties. We chose $\ell=3$, as the minimum number of symbol periods to which this pulse shape could be truncated. The corresponding pulse shapes and spectrums after FM modulation are shown in Figure 2.2. Clearly, there is little loss in the spectral properties with this approach. Using a shorter pulse shape, i.e., $\ell=2$, resulted in significant spectral distortion.

Secondly, we looked at the predictor order. Linear prediction, as we have described it, is equivalent to

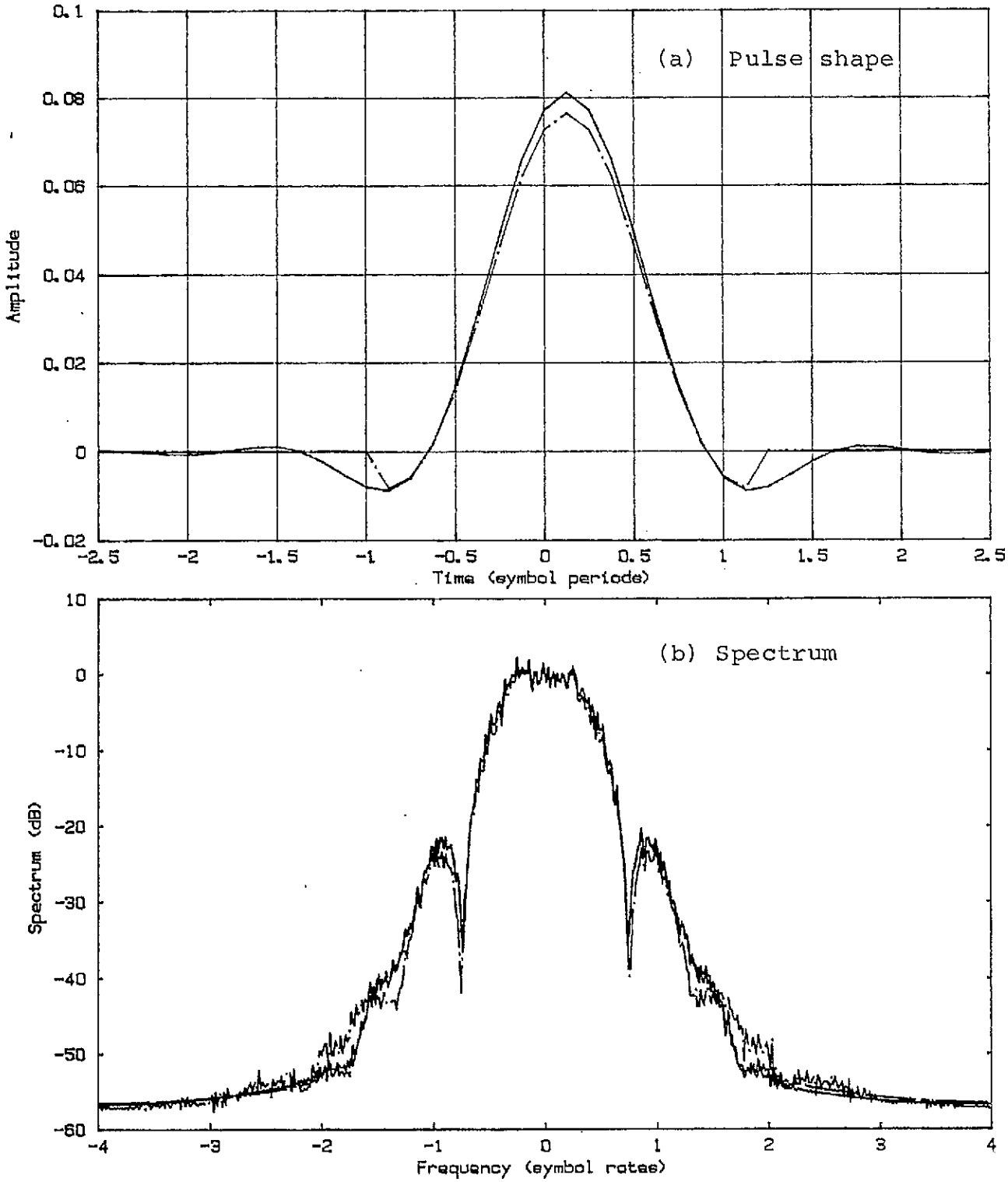

Figure 2.2: Pulse shape and FM modulated spectrum of N32FM waveform
(————ℓ=∞, —— - ——ℓ=3, —— . ——ℓ=2)

modeling the fading and noise spectrum with an all-pole filter [8]. The predictor order, rp-1 in the notation of Section 2.2, is the number of poles used in this model. Clearly, we would like to maximize the number of poles to maximize the accuracy of the model. In practice, this will be limited by the accuracy to which the fading spectrum is known. On the other hand, one wishes to minimize p to reduce the implementation complexity. This would seem to leave the option of increasing the sampling rate r. However, increasing r reduces the signal to noise ratio; thus this parameter should also be minimized. The sampling rate, however, must be large enough to avoid causing significant distortion to the constant envelope properties of the received signal (in an ideal channel). The choice r=2 was selected as the best compromise between maximizing the signal to noise ratio and minimizing the signal distortion. Complexity constraints limit the value of p to four or less.

If we choose p=3, then the corresponding trellis diagram would have $2^{\ell+p} = 64$ states; and similarly, 128 states result with $\ell=4$. An approach we have taken to simplify this is illustrated in Figure 2.3 for the case $\ell=3$. In this figure we have defined a window representing the received signal samples over which each term (linear prediction error) of the branch metric is calculated. For the case r=2, there are two such terms and the corresponding windows are shown (see equation (2.16)). Clearly, in this case ($\ell=3$, p=3) the branch metric is influenced by six bits. However, the influence of the sixth bit appears to be quite minor; it does not affect the calculation of the first term of the branch metric at all, and only influences the last sample of the second term. For this reason, it was felt that the influence of the sixth bit, or the $(\ell+p)$'th bit in general, could be ignored, and as a result the complexity is reduced by a factor of 2.

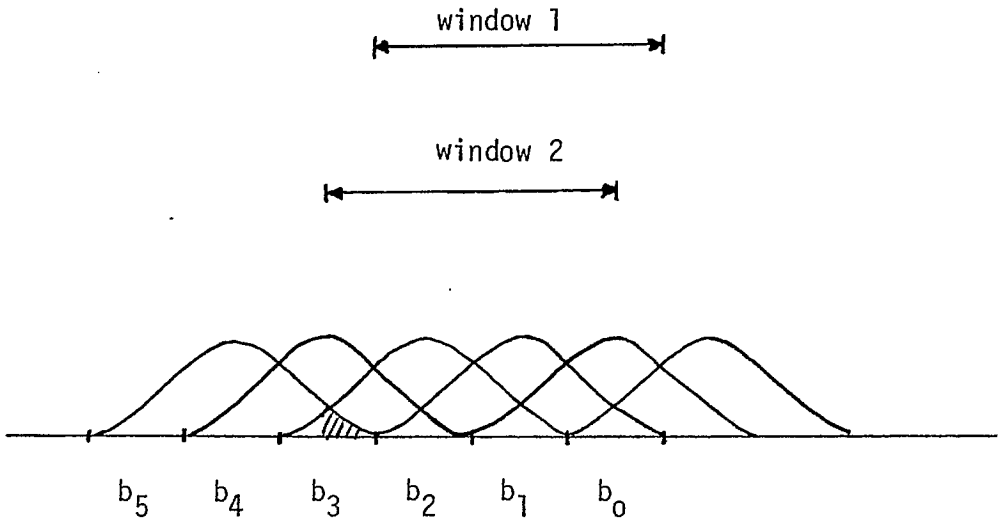

Figure 2.3 Illustration of bits affecting a branch metric calculation. ($r=2$, $\ell=3$, $p=3$)

Under the above assumptions, the modulated sequences corresponding to the $2^{\ell+p-1}$ bit sequences (representing the different trellis states) can be generated, and multiplied by the linear predictor coefficients to form a bank of matched filters. Note that these are r matched filters for each state, corresponding to the r terms of each branch metric calculation. These matched filters are all finite impulse response (FIR) filters with pr coefficients, and are thus quite simple to implement.

In Figure 2.4, we illustrate how well a 5'th and 7'th order predictor matches the fading spectrum when the fading bandwidth to bit rate is .01, .05, 0.1 and 0.3 (r=2). The fading spectrum is assumed to have a 10% root raised cosine spectral shape and E_b/N_o = 10 dB. Visually, there is a reasonable match between them, and there is no apparent significant advantage of the 7'th order predictor over the 5'th order.

Given the above definition of state, and the corresponding branch metric calculation, MLSE is performed using the standard dynamic programming algorithm, analogous to Viterbi detection of convolutional codes.

2.4 MLSE Detection of CPM Signals With Coding

When CPM data is protected by coding in conjunction with interleaving, true MLSE must use one gigantic trellis which accounts for the combined effects of the channel and the coding. Such an approach has been taken to the MLSE of data which is protected by forward error correction coding, and transmitted over a channel with intersymbol interference [7]. For technical reasons (the correlation matrix is data dependent and does not possess a square root), this approach cannot be applied here. A theoretical

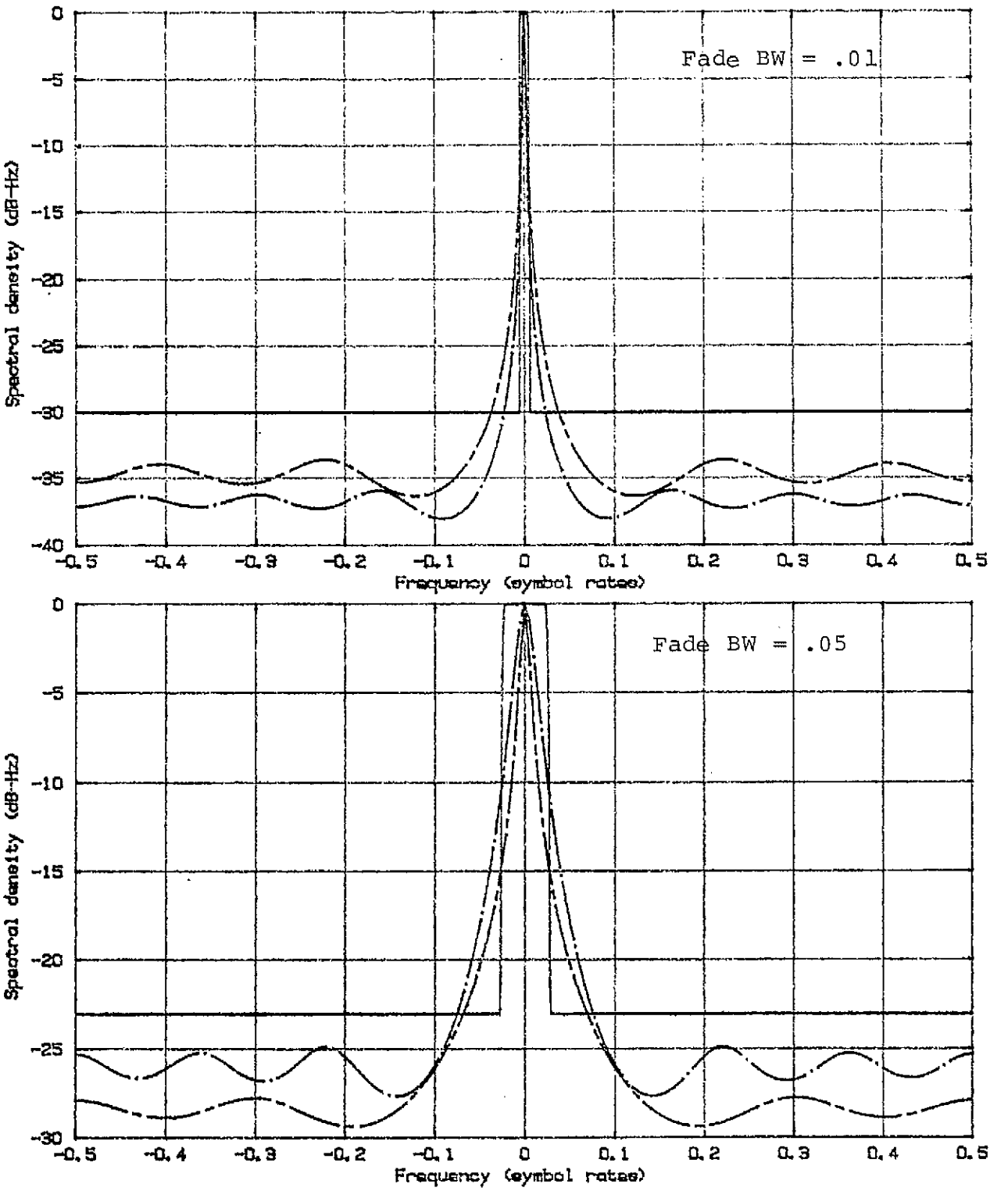

Figure 2.4(a): Composite spectrum and 5'th and 7'th order predictor spectrums
(——— composite spectrum, —— - —— 7'th
—— . —— 6'th)

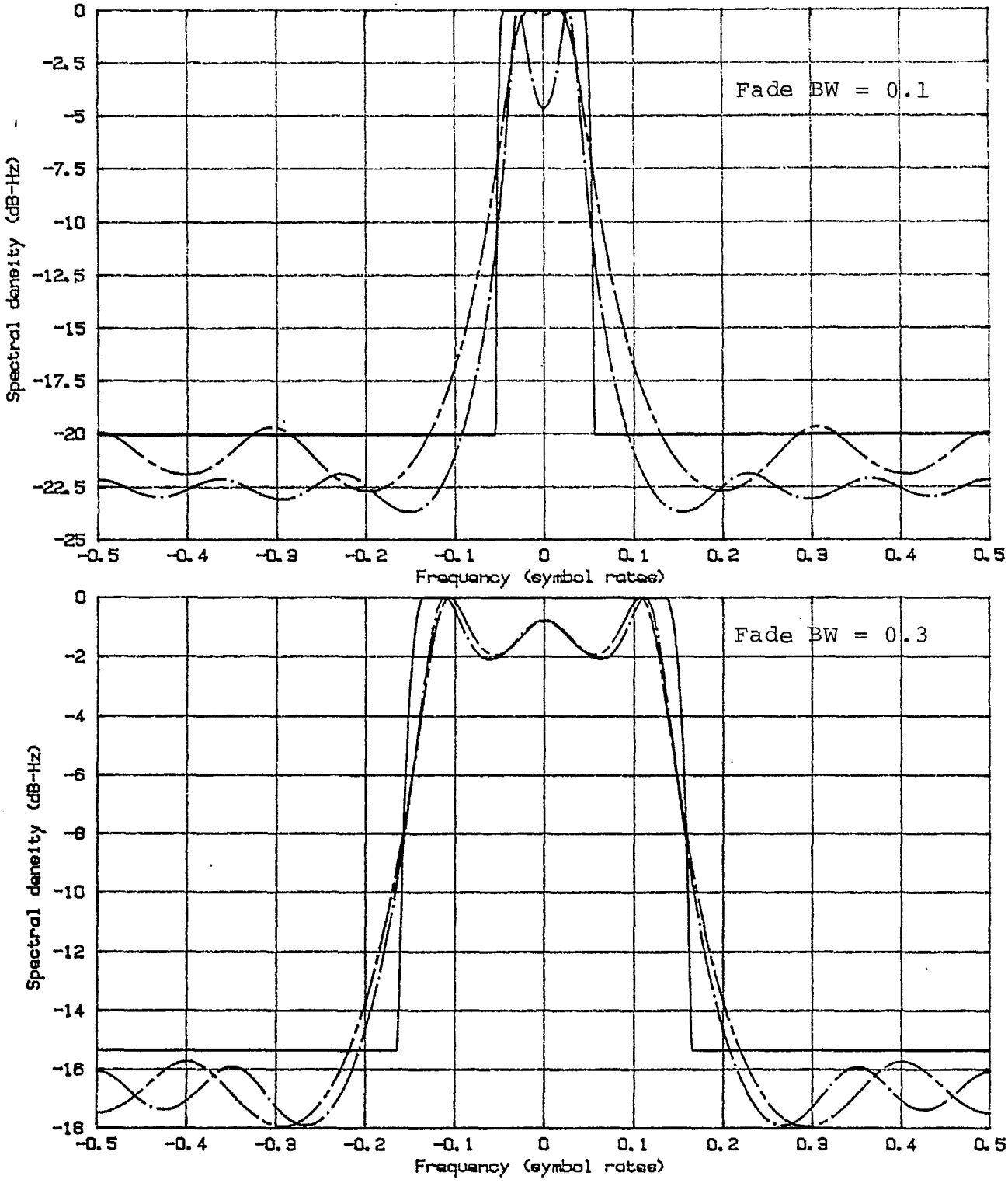

Figure 2.4(b): Composite spectrum and 5'th and 7'th order predictor spectrums
(—— composite spectrum, —— - —— 7'th,
——.—— 5'th)

approach is possible but, in general, the resulting implementation is too complex to be of practical use in these circumstances. Instead the obvious simplification is proposed, that is, separating the detection and decoding processes into two separate trellises of manageable size. This simplification will result in some loss in performance relative to true MLSE, but it is believed that if the right information is passed between the two trellises this loss can be minimized.

2.4.1 Soft Decision Techniques

As previously mentioned, minimizing the implementation complexity of MLSE when coding has been applied seems to require the separation of the detection and decoding trellises with an exchange of information between the two. Considerable effort has focused on the best information or "soft decisions" to exchange between the trellises. However, we believe that we have not yet found the best definition of a soft decision.

The following methods showed some improvement over simple hard decisions:

(i) The first method was based on the bit sequences (histories) of the most likely paths through the trellis, as determined by the Viterbi decoding algorithm. By looking at a point before these paths merged, it was thought that the distribution of 1s and 0s would be indicative of the confidence in the decision at that point. However, it was found that, with this detection technique, the sequence paths merged quite quickly, within two or three bits of the system memory. As a result, the soft decisions produced by this technique were quantized quite

coarsely, typically to levels of (-1, -½, 0, ½, 1). Consequently, the performance of this technique was only slightly better than hard decisions.

(ii) The second method was based on the observation that in the derivation of the maximum likelihood sequence estimator for coded data, the objective function being minimized is inversely weighted by the received noise spectral density [9, p. 296]. Because the noise spectral density is usually constant, this proportionality is ignored. This derivation also implicitly assumes a constant received energy per bit, a situation which is not true under fading channel conditions. This motivated the approach of exchanging hard decisions between the two trellises but also providing channel state information which was used as a weighting factor in the Viterbi decoding. The channel state information (received energy per symbol) is directly estimated using the envelope of the received signal. The received envelope is a reasonably accurate estimate of the channel state because the average signal to noise ratio is quite high in the cases of interest. This technique performed the best of the soft decision techniques investigated and results are given in Section 2.6.

(iii) The third method investigated was based on the predictor errors produced by the detection algorithm. The difficultly with this approach is that at the time the predictor errors are produced, the actual bit decision is unknown, as well as the actual path arriving at that decision. Consequently, we compared the minimum predictor error over all paths arriving at a given bit

decision to the minimum predictor error over all paths arriving at the opposite decision. The difference between these two minimums was the soft decision information.

This method performed only slightly better than the hard decision approach.

(iv) The last method considered was motivated in part by the third method but more by the following. Consider the implementation when one does true MLSE, that is, the detection and decoding trellises are not separated. Assume the interleaving depth is n, and q_i is the state of the i'th multiplexed decoder, then the overall trellis state, Q, is the combination (product space) of the individual states

$$Q = (q_1, q_2, \ldots, q_n)$$

At each step in the trellis, only one of the q_i are altered and these are altered in sequence. The branch metric calculation of equation (2.18) is still the same when the full trellis is used; the only difference is in the manner that x_j depends on I_n. This dependency is illustrated in Figure 2.5 for the case p=3 and ℓ=3.

Each bit in the sequences depends on a different decoder, in a cyclical manner which repeats every n bits (assuming bit interleaving as opposed to symbol interleaving). In general, the number of possible states Q is immense. If, for example, a constraint length 7 convolutional code is used, then the number of states in each decoder is $|q_i| = 2^6$, and the total number of states is $|Q| = (2^6)^n$, where n is the

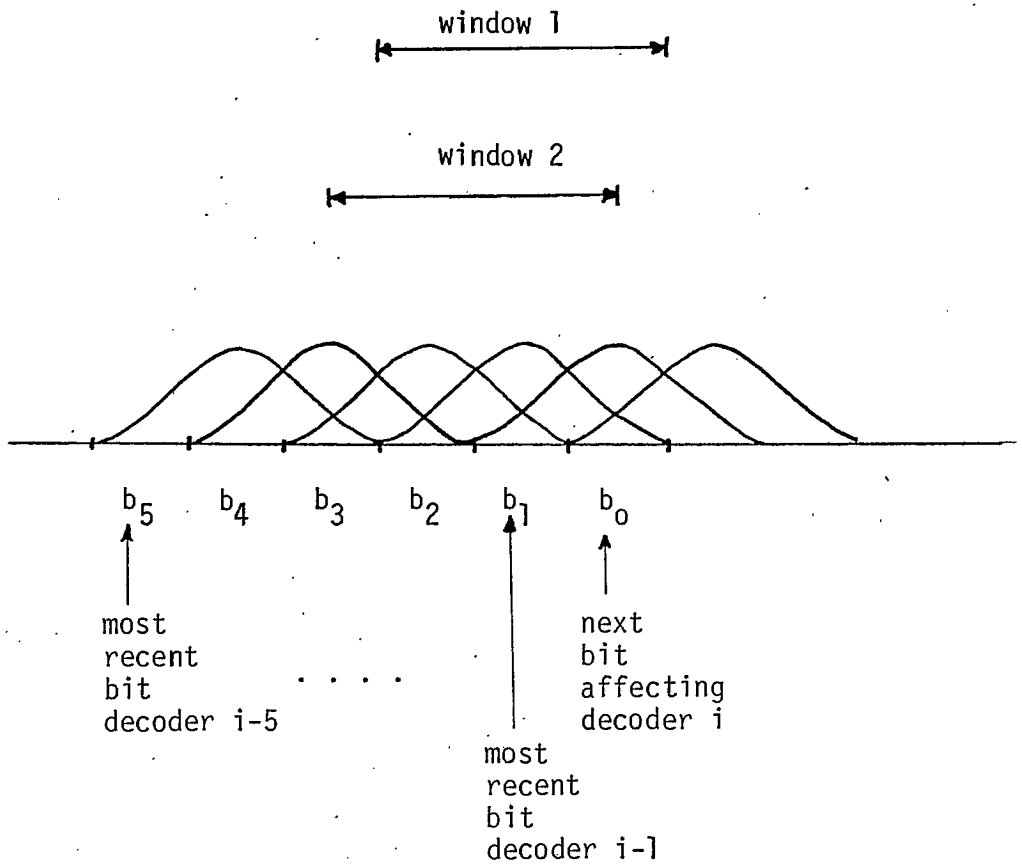

Figure 2.5 Illustration of the division of bits among multiplexed codecs. ($r=2$, $\ell=3$, $p=3$)

interleaving depth. Updating this number of cumulative metrics at each step is clearly not feasible and justifies the separation of detection and decoding trellises. (Note that the number of branch metrices to be calculated at each step is much less than $|Q|$. It is $2^{\ell+p}$, where $2^{\ell+p}$ is the number of different coded bit sequences affecting the window.)

The suggested approach for generating a soft decision for bit i is, given the MLSE of the bits surrounding i (the merged sequence in the Viterbi decoder of the CPM signal) assume the surrounding bits are correct and generate the prediction errors corresponding to bit i and an inverted bit i (see Figure 2.6). These two prediction errors are then used as explicit branch metrics in the Viterbi decoding algorithm of the convolutional code, not just as a "soft decision".

This approach and several variations of it have been tried but none provide reliable detection under noisy channel conditions. This technique, however, is intuitively felt to be close to the optimum approach with a split trellis. Further investigation of this approach is strongly recommended, although time constraints have prevented our doing so here.

2.5 Performance Without Coding

The performance of both differentially detected CPM (DCPM) and maximum likelihood detection of CPM was estimated through computer simulation. The results are shown in Figure 2.7 for the case of uncoded data. Also included for

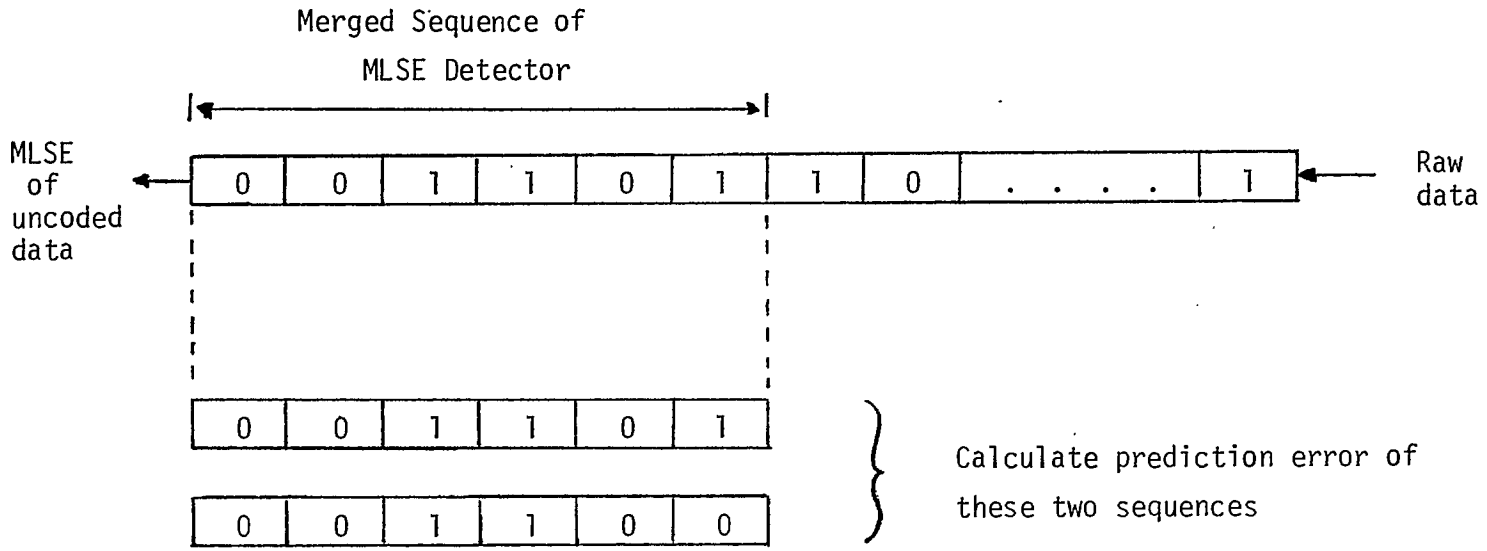

Figure 2.6 Illustration of fourth soft decision approach.

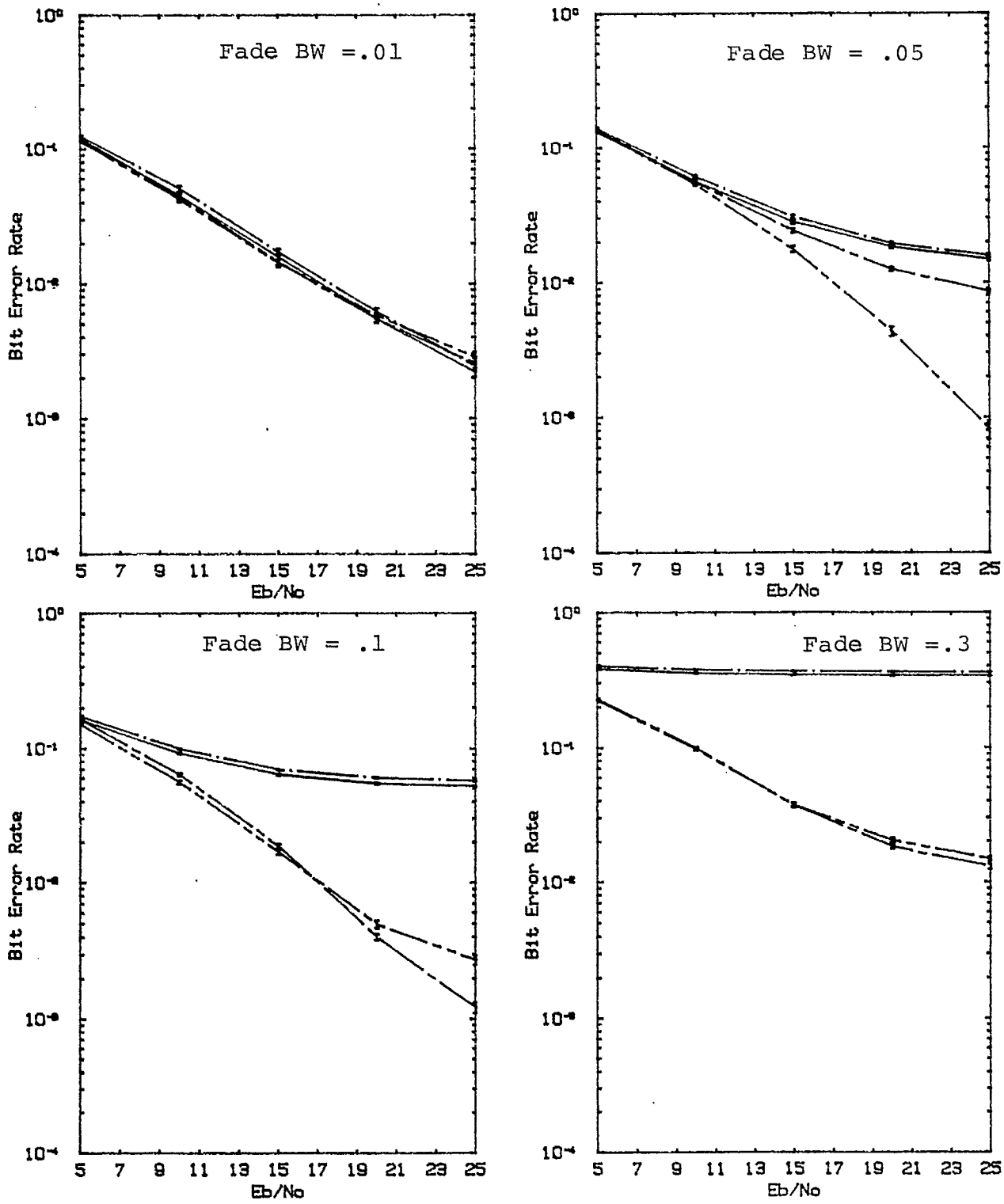

Figure 2.7: Performance over Rayleigh fading channel without coding

———— 2-DPSK

——.—— DCPM

— - — MLSE with 6'th order predictor

— -- — MLSE with 8'th order predictor

comparison and validation purposes is the performance of differentially detected binary PSK (2-DPSK). All results are for Rayleigh fading channels, with one-sided fading bandwidths of 0.01, 0.05, 0.1 and 0.3 times the channel bit rate. The fading spectrum is that described in [10].

Looking at these results in order of increasing fading bandwidth the following observations can be made. When the ratio of fading bandwidth to bit rate, η, is 0.01, there is very little difference between any of the techniques. In this case, the performance with 2-DPSK matches closely the theoretical performance curve for slow-fading Rayleigh channels [9, p. 491]. Performance with DCPM is slightly poorer than with 2-DPSK and this difference can be attributed to filtering losses. Performance of the MLSE technique is very slightly better than 2-DPSK and DCPM at lower E_b/N_o and slightly worse at high E_b/N_o. This is true for both a 6'th and an 8'th order predictor. The poorer performance at high E_b/N_o is because the predictors have been selected for an E_b/N_o of 10 dB. In general, at this low fading rate, there is very little difference between any of the methods.

As the fading parameter η is increased to 0.05 through 0.3 we see a gradual degradation of the performance of 2-DPSK and DCPM. When η = 0.3 these two techniques are almost useless, providing an error rate between 0.3 and 0.4. On the other hand, performance with the MLSE technique remains remarkably good considering the channel conditions. There is some degradation in performance as the fading bandwidth increases, but even when η=0.3, MLSE in conjunction with a moderate complexity coding algorithm could provide reliable performance. The slight variations and discrepancies seen between the 6'th and 8'th order predictor results are due to the fact that both are optimized to an E_b/N_o of 10 dB.

Traditional techniques for using these faster fading channels are to employ FSK modulation with noncoherent detection and diversity. Diversity is used to make the fast fading channel look like a series of slow fading channels. The advantage of the MLSE technique is that it does not require the bandwidth expansion needed to implement diversity.

2.6 Performance With Coding

The performance of 2-DPSK, DCPM and MLSE of CPM in conjunction with a constraint length 7 convolutional code was also estimated through simulation. The conditions were the same as in the tests without coding. The fading parameter η is defined as the ratio of the one-sided fading bandwidth to the channel bit rate. The results for 2-DPSK and DCPM, with hard and soft decisions being passed on to the Viterbi decoder for the convolutional code, are shown in Figure 2.8. The soft decision used for both these methods was the output of the delay and multiply portion of the differential detector. In all but the fastest fading rate case, coding improves performance substantially. Soft decisions offer a considerable performance advantage over hard decisions. As before, there is a slight degradation of performance of differential CPM relative to DPSK. In general, performance degrades as the fading rate increases, and, as one can see, when $\eta = 0.3$ coding is quite ineffectual. The only exception to this is that soft decision performance with a fading parameter of 0.01 seems to be slightly worse than with a fading parameter of 0.05. This is believed to be due to less than optimum interleaving in the former case. The interleave factor was 10 in all examples and interleaving was performed on a subsymbol basis.

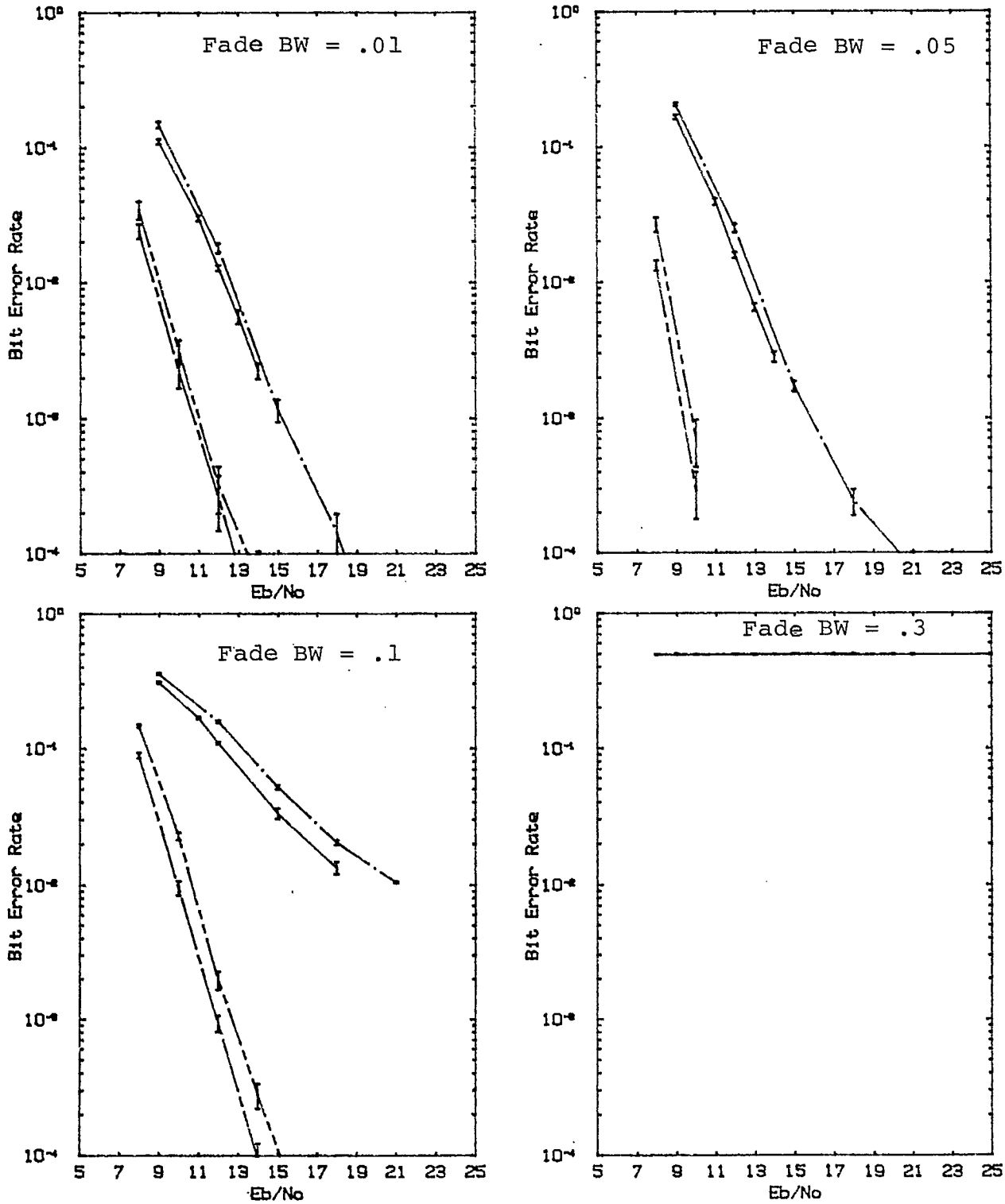

Figure 2.8: Performance over Rayleigh fading channel with coding.

———— 2-DPSK with hard decisions
——.—— DCPM with hard decisions
—— - —— 2-DPSK with soft decisions
—— -- —— DCPM with soft decisions

In Figure 2.9, the performance of MLSE with coding is shown. As one would expect from the uncoded results, the hard decision performance is better than that of both DCPM and DPSK, and the difference becomes more dramatic as the fading bandwidth increases. The "soft decision" technique used was hard decisions in conjunction with channel state information, as described in Section 2.4. This soft decision technique does offer a significant advantage over hard decisions, but it does not perform as well as soft decision DPSK on the slower fading channels. It is thought that a better choice of soft decision may eliminate this difference (see Section 2.4).

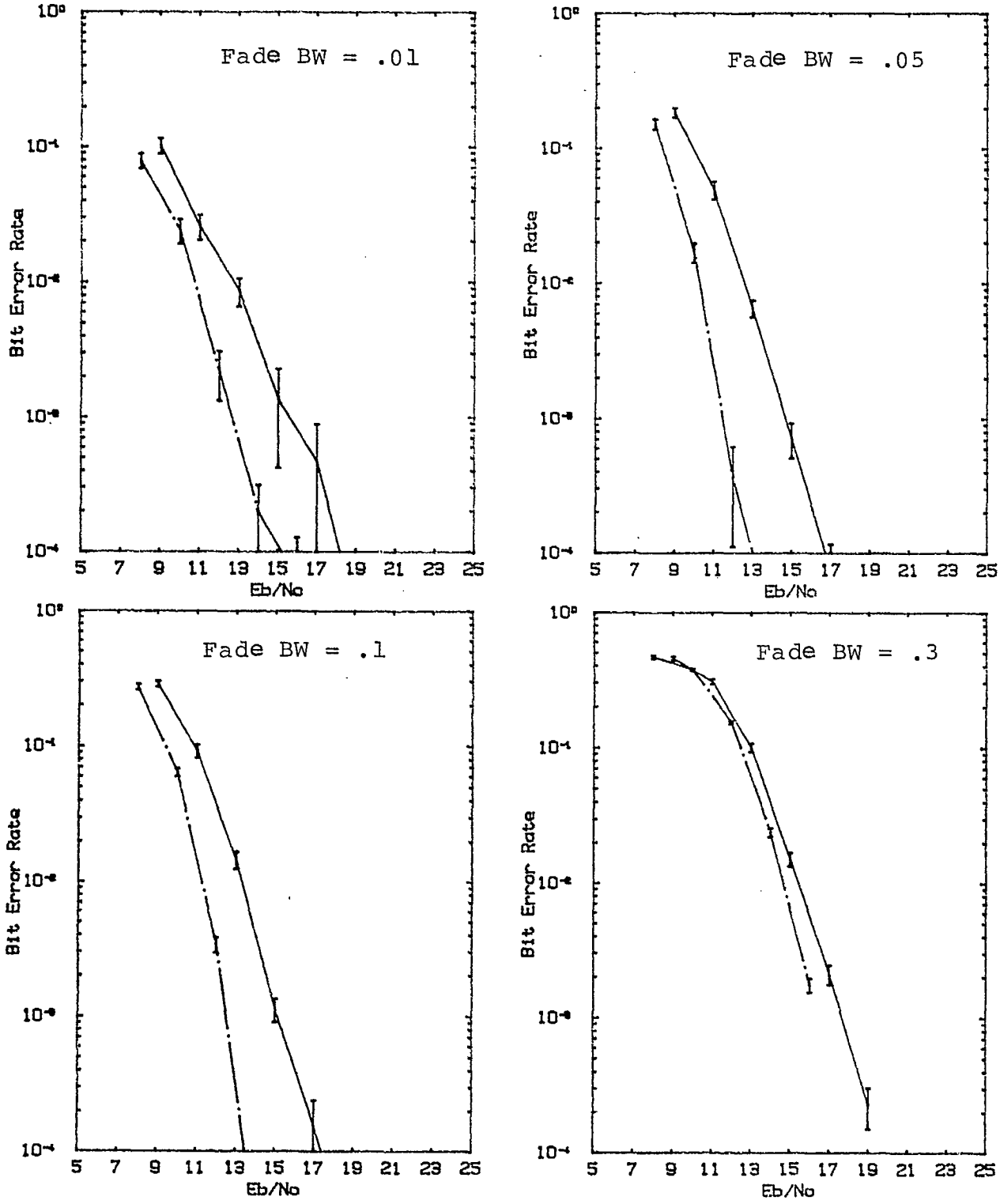

Figure 2.9: Performance over Rayleigh fading channel with coding.

———— MLSE (6'th order) with hard decisions

—— . —— MLSE (6'th order) with soft decisions (channel state information)

3.0 CONCATENATED REED-SOLOMON/TRELLIS CODE PERFORMANCE

The second part of this study was to determine, using computer simulation, the performance that results when an 8-state trellis code is concatenated with a Reed-Solomon (RS) code for reliable transmission of data over fading channels. The fading channels being considered here were Rician fading channels, with fading bandwidths which would be typical of land-mobile or aeronautical satellite communications. The trellis code considered was an 8-state trellis code which independently codes the inphase and quadrature channels to provide a 16 point QAM constellation. Interleaved with the code sequence is a known pilot sequence, which is used to provide coherent detection and gain control at the receiver. The details of this coding and modulation strategy may be found in [14].

While the trellis coding and modulation scheme described above does provide good performance, it does not provide any error detection capabilities. Concatenating a Reed-Solomon code with the above scheme can improve performance and also provide error detection capabilities. Because of potential applications, it is desirable that the symbols for the RS code be 8-bit symbols, that is, elements of the Galois field GF(256). Another desirable characteristic of the RS code is that the number of information bits in a packet be a multiple of the 96-bit INMARSAT signal unit. This motivates the choice of the (240,180) shortened RS code (shortened from (255,195)). This is a rate 3/4 code that contains 15 signal units of data, and is capable of correcting a codeword with up to 30 bytes in error.

This section is divided into three subsections. The first discusses the implementation of the RS coding scheme. The

second presents the performance of the combined concatenated coding scheme. The third section looks at the error detection capability of the code.

3.1 Reed-Solomon Code Implementation

The block diagram of Figure 3.1 shows the position of the Reed-Solomon code in the overall coding and modulation strategy. The RS(240,180) code is the outer code in this scenario and it takes a block of 8x180 = 1440 information bits and codes it up to 8x240 = 1920 code bits. The output symbols of the RS code are interleaved on a subsymbol basis, that is, the RS symbols are read into a buffer, one symbol (8 bits) per row. The interleaving depth (the number of symbols in the buffer) matches the interleaving depth (number of codecs) in the multiplexed trellis code.

The trellis code is interleaved (or multiplexed) because these codes perform best when the errors are independent. Interleaving breaks up sequences of burst errors caused by the fading channel and results in better trellis code performance. On the other hand, an error event at the output of a trellis codec is usually a burst of errors. The interleaving also breaks up these error sequences and makes them look like uncorrelated errors. A RS code, however, is ideally suited to burst errors because it functions on a per symbol basis. An error burst of 8 bits will cause one or at most two Reed-Solomon symbols to be in error. The code we have chosen can correct up to 30 such symbol errors. Interleaving the RS code on the subsymbol basis, as shown in Figure 3.1, means that not only does one get the benefits of interleaving for the trellis codec, but de-interleaving after the trellis code means that errors will be grouped together in bursts which is the ideal situation for the RS decoder.

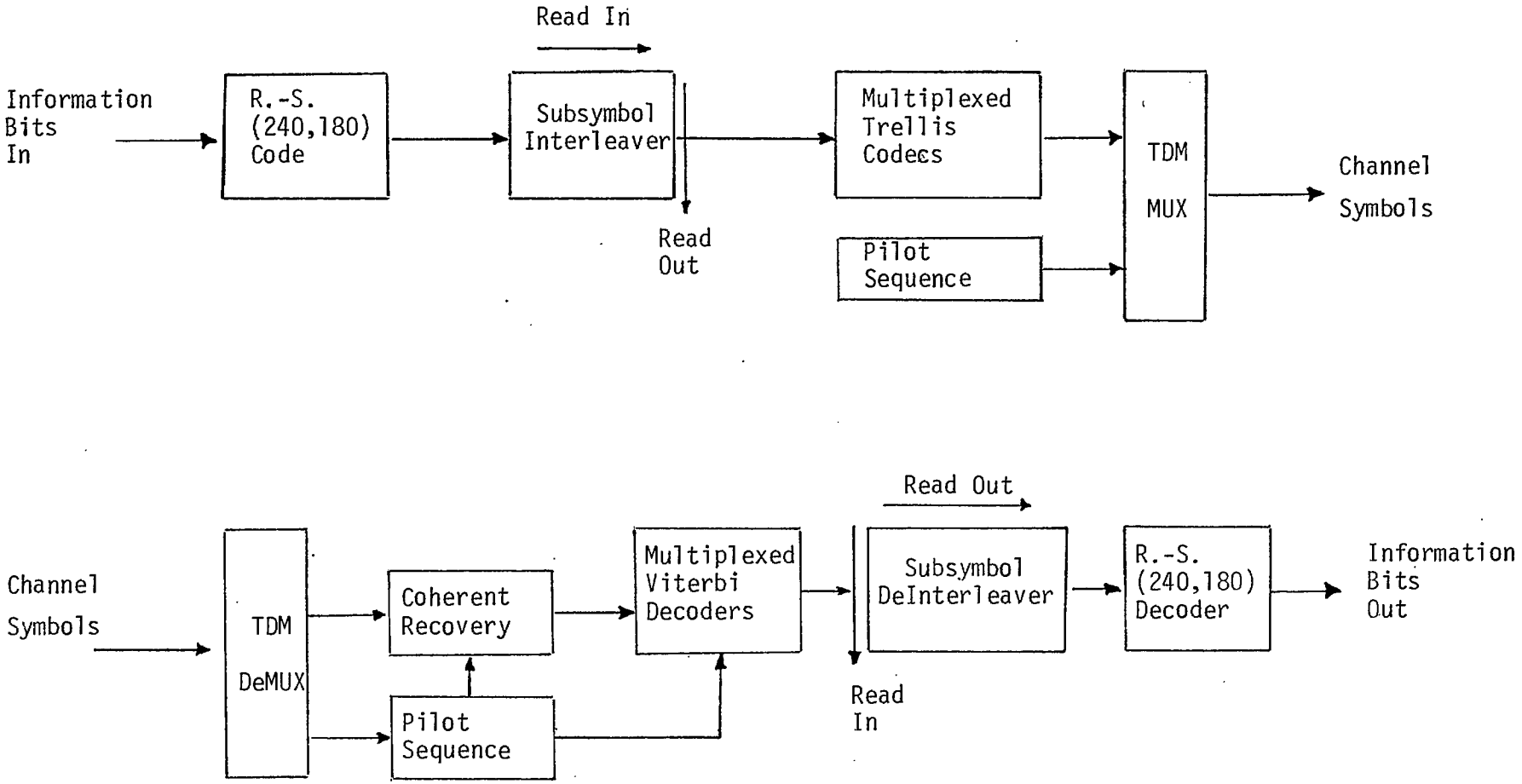

Figure 3.1 Illustration of concatenated coding scheme with subsymbol interleaving matched to the interleaving depth of the trellis code.

3.1.1 The Reed-Solomon Coding Algorithm

The RS class of codes is a subclass of the Bose-Chaudhuri-Hocquenghem (BCH) class of codes. The BCH codes are multiple-error-correcting block codes and are among the more important coding techniques for three reasons; they are well understood, there are good codes in this class; and there are practical implementations of these codes.

The RS codes are maximum distance BCH codes. They have the property that the block length divides the multiplicative order of the symbol alphabet. That is, if there are q different symbols, then the blocklength is q-1. In our case $q=2^8$ and the block length is 255.

The steps to implementing the coding algorithm for a RS(q-1,k) code, where k is the number of information symbols, are as follows:

(i) define a generator polynomial g(y) by

$$g(y) = \prod_{i=1}^{2t} (y - \beta^i) \qquad (3.1)$$

where t is the number of symbol errors the code must be capable of correcting and β is a primitive element of the field GF(q). The number t must satisfy the constraint

$$q-1-k-2t = 0 \qquad (3.2)$$

(ii) information bits are coded into blocks of k symbols in GF(q) (i.e., eight information bits per symbol in GF(256)) and form the information polynomial

$$i(y) = \sum_{j=0}^{k-1} s_j \, y^j \tag{3.3}$$

where (s_j) are the information symbols.

(iii) the codeword polynomial is then given by

$$c(y) = i(y) * g(y) \tag{3.4}$$

$$= \sum_{j=0}^{q-1} c_j \, y^j \tag{3.5}$$

The coefficients (c_j) are the code symbols to be transmitted.

All of the computations in the above algorithm must use the arithmetic of GF(q). If β is a primitive element of this field, then all elements of this field except 0 (the additive identity) can be represented as β^k for some k, and

$$\beta^k * \beta^\ell = \beta^{(k+\ell) \bmod (q-1)} \tag{3.6}$$

or using β as a logarithmic base

$$k * \ell = k + \ell \bmod (q-1) \tag{3.7}$$

where * on the left hand side is multiplication in GF(q) and + on the right hand side is addition in the integers. While this appears to be the simplest notation, it means that addition in GF(q) is not simple to compute.

It was found that the simplest and fastest way to implement addition was to create an addition table and save it in memory. Note that with the logarithmic notation described above, the multiplicative identity is denoted by 0, and the additive identity must be denoted by some other element, say $-\infty$. Also note that when one writes expressions such as $(y - \beta^j)$ in equation (3.1), one means $(id.y - \beta^j)$ where id is the multiplicative identity, (that is, 1 is not the multiplicative identity).

To implement a Reed-Solomon code shortened by m information symbols, one follows the same procedure as above except that the information bits are used to form (k-m) information symbols in step (ii), and the remaining information symbols (higher terms in i(y)) are set to zero (the additive identity).

The resulting codeword has a zero coefficient for the m highest terms and only the q-1-m lower coefficients are transmitted.

3.1.2 The Reed-Solomon Decoding Algorithm

The approach chosen to decode the received codeword is the Petersen-Gorenstein-Zierler (PGZ) algorithm [11]. While this algorithm is not as fast as the Berlekamp-Massey algorithm or variations of it, it is a simple, understandable algorithm which is readily implemented and is flexible. The algorithm is flexible in the sense that it can be easily adapted to other Reed-Solomon (or BCH) codes.

To implement this decoder requires the following steps:

(i) determine the syndromes (S_j) of the received codeword by

$$S_j = c(\beta^j) \quad j = 1, \ldots, 2t \tag{3.8}$$

that is, evaluate the received codeword polynomial $c(y)$ at the zeros β^j of the generator polynomial.

(ii) solve the system of equations

$$\begin{bmatrix} S_1 & S_2 & S_3 & \cdots & S_\nu \\ S_2 & S_3 & S_4 & \cdots & S_{\nu+1} \\ S_3 & S_4 & S_5 & \cdots & S_{\nu+2} \\ \cdot & \cdot & \cdot & & \cdot \\ \cdot & \cdot & \cdot & & \cdot \\ S_\nu & S_{\nu+1} & S_{2\nu+2} & \cdots & S_{2\nu-1} \end{bmatrix} \begin{bmatrix} \Lambda_\nu \\ \Lambda_{\nu-1} \\ \Lambda_{\nu-2} \\ \cdot \\ \cdot \\ \Lambda_1 \end{bmatrix} = \begin{bmatrix} -S_{\nu+1} \\ -S_{\nu+2} \\ -S_{\nu+3} \\ \cdot \\ \cdot \\ -S_{2\nu} \end{bmatrix} \tag{3.9}$$

for (Λ_i), the coefficients of the error locator polynomial $\Lambda(y)$

$$\begin{aligned} \Lambda(y) &= \Lambda_\nu y^\nu + \Lambda_{\nu-1} y^{\nu-1} + \ldots + \Lambda_1 y + 1 \\ &= \prod_{j=1}^{\nu} (1 - y X_j) \end{aligned} \tag{3.10}$$

where X_j are the error locations (in GF(q)). In the above ν is clearly the number of errors. In practice this is unknown, and one solves (3.9) by beginning with the assumption $\nu=t$ (the maximum number of errors that can be corrected) and decreasing it until the matrix on the left hand side of (3.9) is non-singular. The system of equations (3.9) was solved by performing an LU decomposition using Galois field arithmetic.

(ii) the zeros of the error locator polynomial $\Lambda(y)$, which are the inverted error locations X_j^{-1}, are found using a Chien search [11], that is, by trial and error. Since we are dealing with a finite field, this is quite simple to do.

(iii) once the error locations are known, the error magnitudes y_j can be determined using the Forney algorithm [11]

$$y_j = \frac{\Omega(X_j^{-1})}{X_j^{-1}\ \Lambda'(X_j^{-1})} \tag{3.11}$$

where $(\ )'$ means the formal derivative and $\Omega(x)$ is the error evaluator polynomial given by

$$\Omega(x) = S(x)\ \Lambda(x) \bmod x^{2t} \tag{3.12}$$

where

$$S(x) = \sum_{j=1}^{2t} S_j\ x^j \tag{3.13}$$

(iv) the appropriate corrections are made to the received codeword. Then the corrected received codeword is divided by the generator polynomial to recover the information polynomial.

The above algorithm is relatively straightforward to implement and executes quite quickly. With this approach there are three criteria on which to reject a codeword (packet) for having too many errors. They are:

- an error position of zero (the additive identity) is found,
- an error magnitude of zero is found, or
- the remainder after dividing the corrected codeword polynomial by the generator polynomial is not zero.

We will show later that these are very strong error detection criteria.

If a shortened RS code is used then the received codeword is simply assumed to have zeros as the higher order coefficients, and the algorithm is executed as above. The shortened RS code has even stronger error detection capabilities, because an error position which falls in the m higher order symbols, which are known to be zero, is a further rejection criterion.

3.1.3 Reed-Solomon Decoding With Erasures

With each RS codeword there is associated k information symbols and 2t parity symbols, although these are not, in general, arranged in a systematic form. The 2t parity symbols allow the correction of up to t errors. In a sense, t parity symbols are used to determine the error locations and the remaining t are used to determine the error magnitudes. If some of the error locations are known (erasure information), then the code can be used to correct more than t errors. In fact, if the location of ε errors (erasures) are known, then the code can correct these errors and $(2t-\varepsilon)/2$ other errors whose locations are unknown. (We must necessarily have $\varepsilon \leq 2t$.)

To implement the decoding algorithm with erasures we first note that the set of equations (3.9) is equivalent to forming the product, $S(x)\,\Lambda(x)$, of the syndrome polynomial

and the error locator polynomial, and setting terms of degree greater than ν equal to zero ($\nu \leq t$). To implement erasure decoding we write

$$\Lambda(x) = \Lambda^u(x)\ \Lambda^k(x)$$

where $\Lambda^u(x)$ is the error locator polynomial corresponding to unknown error locations, and $\Lambda^k(x)$ is the error locator polynomial corresponding to known error locations. We then form

$$\begin{aligned} S^e(x) &= S(x)\ \Lambda^k(x) \\ &= \left(\sum_{j=1}^{2t} S_j\, x^j\right)\left(\sum_{\ell=0}^{\varepsilon} \Lambda_\ell^k\, x^\ell\right) \\ &= \sum_{j=1}^{2t+\varepsilon} S_j^e\ x^j \end{aligned} \tag{3.14}$$

where

$$S_j^e = \sum_{\ell=0}^{j} \Lambda_\ell^k\, S_{i-\ell} \tag{3.15}$$

and where $\Lambda_0^k = 1$, and $\Lambda_\ell^k = 0$ if $\ell \geq \varepsilon$. We then set the terms of $S^e(x)\ \Lambda(x)$ of order greater than ν equal to zero to obtain the set of equations

$$\begin{bmatrix} S^e_{\varepsilon+1} & S^e_{\varepsilon+2} & \cdots & S^e_{\nu} \\ S^e_{\varepsilon+2} & S^e_{\varepsilon+3} & \cdots & S^e_{\nu+1} \\ \vdots & & & \vdots \\ S^e_{\nu} & S^e_{\nu+1} & \cdots & S^e_{2\nu-\varepsilon-1} \end{bmatrix} \begin{bmatrix} \Lambda^u_{\nu-\varepsilon} \\ \Lambda^u_{\nu-\varepsilon-1} \\ \vdots \\ \Lambda^u_1 \end{bmatrix} = \begin{bmatrix} -S^e_{\nu+1} \\ -S^e_{\nu+2} \\ \vdots \\ -S_{2\nu-\varepsilon} \end{bmatrix} \tag{3.16}$$

This set of equations is solved for the remaining error locations (up to $(2t-\varepsilon)/2$ of them). The remainder of the algorithm is then executed as in the case with no erasure information. (The theory behind this may be found in [11], [12], [13].)

3.2 Performance of the Reed-Solomon Code

In this section we look at the performance, estimated through computer simulation, of the concatenated coding scheme described in Section 3.1. This scheme uses a (240,180) shortened Reed-Solomon outer code and an 8-state QAM trellis inner code. The performance of the trellis code alone has been documented in [14], and the results here are mainly concerned with the combined performance. In Table 3.1 we present the fraction of codeblocks rejected as a function of the noise and fading parameters for a 60 Hz fading channel. The fading varies from the static channel case ($K=-\infty$) to the severe fading case ($K = 0$ dB). Over the range of E_b/N_o, the percentage of blocks rejected ranges from 0 to 100%. It should be pointed out that in the table 0% rejected implies 0 out of the 896 codeblocks simulated were rejected. For comparison purposes, Table 3.2 provides the average output bit error of the inner

E_b/N_o	Rician K Factor (dB) -∞	-10	-5	0
11	.0	.0	.0	.0
9	.0	.0	.0	.0
7	.002	.008	.036	.006
6	.63	.66	.60	.15
5	1.0	1.0	.99	.62

Table 3.1: Fraction of codeblocks rejected in a 60 Hz Rician fading channel for various noise and fading parameters.

E_b/N_o	Rician K Factor (dB) -∞	-10	-5	0
11	.0	$1x10^{-5}$	$6x10^{-5}$	$2x10^{-4}$
9	$1x10^{-4}$	$4x10^{-4}$	$1x10^{-3}$	$1x10^{-3}$
7	$8x10^{-3}$	$1x10^{-2}$	$1x10^{-2}$	$8x10^{-3}$
6	$4x10^{-2}$	$4x10^{-2}$	$4x10^{-2}$	$2x10^{-2}$
5	$1x10^{-1}$	$1x10^{-1}$	$8x10^{-2}$	$4x10^{-2}$

Table 3.2: Average output error rate of inner trellis code which produced the results in Table 3.1.

trellis code under these conditions. It was found that the performance of the Reed-Solomon code was strictly a function of the input error rate. It did not explicitly depend on the fading channel parameters in any other way. The critical point in the performance curve appears to occur at an input bit error rate of 1.0×10^{-2} to the Reed-Solomon decoder. Below this error rate, almost all codeblocks are corrected and accepted; above this, error rate performance deteriorates rapidly.

In all of the examples simulated, all code blocks were either successfully decoded or rejected. There were no falsely decoded code blocks! This will be explained in Section 3.4.

In Table 3.3 the codeblock rejection rates over a 120 Hz fading channel are provided. The corresponding average output error rates of the inner trellis code are provided in Table 3.4. The major effect of the faster fading rate is a worsening of the average output error rate of the inner trellis code. The critical point in the performance of the Reed-Solomon code is, again, when the output error rate of the trelis code is 10^{-2}. Increasing the fading rate simply increases the E_b/N_o at which this point is reached.

3.3 Performance of the Reed-Solomon Code With Erasure Information

As explained in Section 3.1.3, the correction capabilities of a code can be enhanced if some information about the location of errors can be provided. In this case the erasure information would be an output of the inner trellis code.

E_b/N_o	Rician K Factor (dB) -∞	-10	-5	0
11	.0	.0	.0	.0
9	.0	.0	.0	.0
7	.079	.12	.22	.079
6	.95	.95	.92	.59
5	1.0	1.0	1.0	.97

Table 3.3: Fraction of codeblocks rejected in a 120 Hz Rician fading channel for various noise and fading parameters.

E_b/N_o	Rician K Factor (dB) -∞	-10	-5	0
11	.0	3×10^{-5}	2×10^{-4}	8×10^{-4}
9	3×10^{-4}	1×10^{-3}	2×10^{-3}	4×10^{-3}
7	1.8×10^{-2}	2.1×10^{-2}	2.4×10^{-2}	1.7×10^{-2}
6	6×10^{-2}	6×10^{-2}	6×10^{-2}	4×10^{-2}
5	1×10^{-1}	1×10^{-1}	8×10^{-2}	7×10^{-2}

Table 3.4: Average output error rate of inner trellis code which produced the results in Table 3.3.

There is one tradeoff involved in the use of erasure information. This tradeoff is unknowingly declaring a symbol to be erased when one actually has detected it correctly. This will increase the number of errors and can result, if the error rate is high enough, in either rejection of the codeblock or false decoding.

Seemingly, the most logical source for this erasure information is the channel state information generated as part of the trellis code demodulator [14]. In particular, it was decided to use the estimated channel gain as an erasure criterion. If the channel gain fell below a certain threshold, the corresponding symbol was declared erased. This left the problem of setting the threshold. This latter problem was solved by simulating performance using different threshold settings. In Figure 3.2, the code block rejection rate is plotted as a function of this threshold setting for various channel scenarios. In all cases, the unfaded channel gain is unity. We observe that, except for very low threshold settings, including erasure information can significantly increase the rejection rate. There were no falsely decoded blocks in all of the simulated blocks.

The conclusion is that the number of erasures is very sensitive to the threshold setting. Either very few erasures occur, or enough erasures occur to cause a block rejection. One might expect this to be the case, because the channel gain information (or any type of erasure criterion) obtained from the trellis code is provided on a bit basis and one bad bit is enough to corrupt a RS symbol.

To more explicitly show the effects of this type of erasure criterion, simulation scenarios, identical to those which

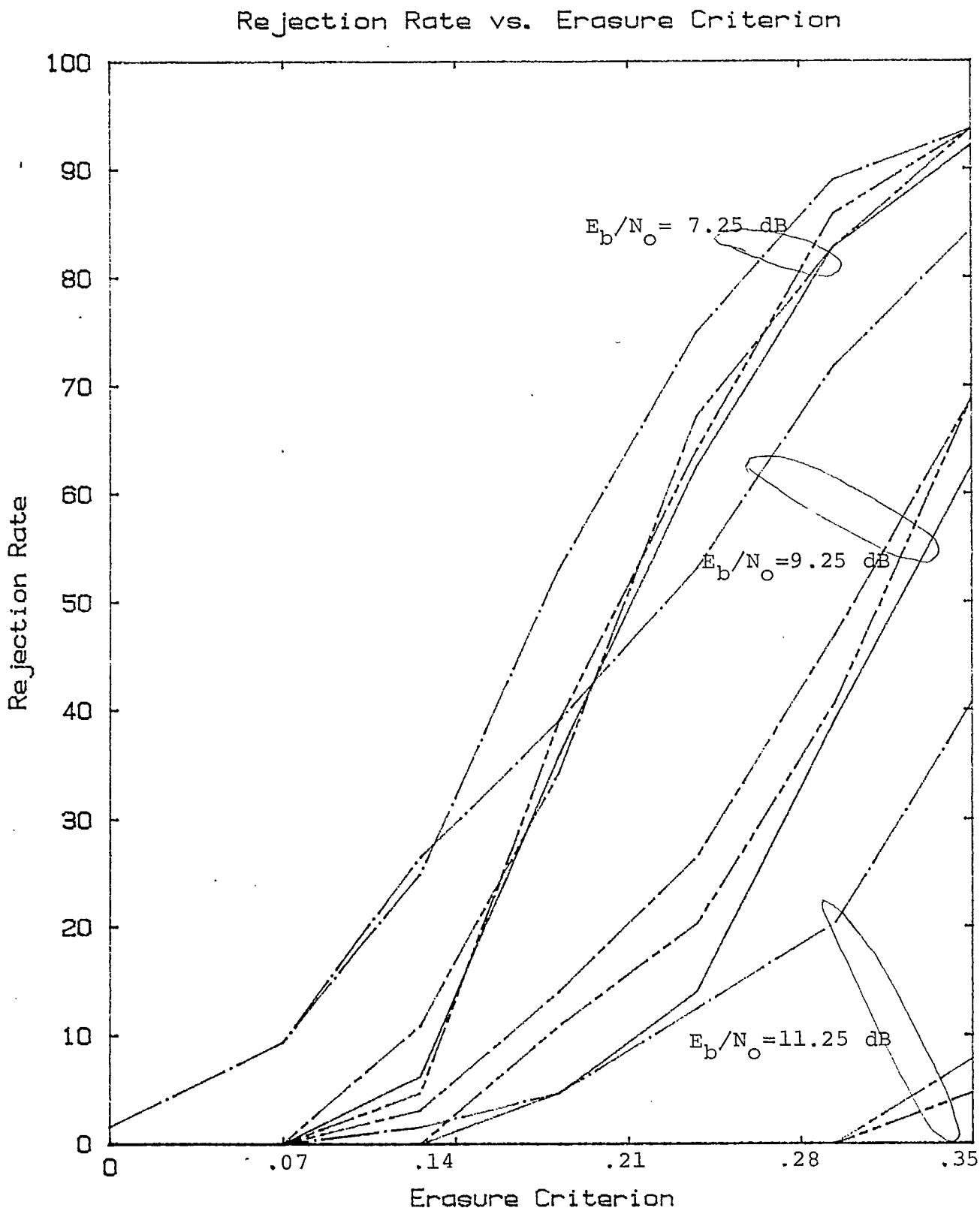

Figure 3.2: Rejection rate of block code as a function of erasure threshold under various channel conditions (———K=-∞, —— -- ——K=-10 dB, —— - ——K=-5 dB, ——— . ———K=0 dB)

produced Table 3.1, were run with a threshold setting of 0.07. The results are shown in Table 3.5. Comparing these results to Table 3.1 we see that the only effect is a slight worsening of the rejection rates under the poorer channel conditions.

From this we conclude that this erasure criterion is of little benefit. Other erasure criteria which are not as sensitive to a single bit fading but more to a symbol fade could provide better performance, but these were not tested.

3.4 Probability of False Decoding

When decoding block codes three outcomes are possible. The code block can be correctly decoded, a codeblock can be rejected because an internal consistency check indicates that there are two many errors for the code to correct, or the codeblock can be falsely decoded. The simulation results obtained with the (240,180) shortened RS code only produced the first two outcomes; no false decodings occurred. The purpose of this section is to estimate the probability of false decoding.

Reed-Solomon codes are popular because they are optimum for their structure. They are maximum distance codes. To determine the probability of false decoding for the above code means getting a handle on the weight distribution of the code. Fortunately we have the following results for maximum distance codes.

The weight distribution of a maximum distance (n,k) code over GF(q) is given by [11, p. 434]

E_b/N_o	Rician K Factor (dB)			
	$-\infty$	-10	-5	0
11	.0	.0	.0	.0
9	.0	.0	.001	.004
7	.002	.011	.12	--
6	.63	.68	.72	.40
5	1.0	1.0	.995	.84

Table 3.5: Fraction of codeblocks rejected in a 60 Hz Rician fading channel with an erasure threshold of .07 and various noise and fading parameters.

$$A_\ell = \begin{cases} 1 & \ell = 0 \\ 0 & \ell = 1, \ldots, d^*-1 \\ \binom{n}{\ell} \sum_{j=0}^{\ell-d^*} (-1)^j \binom{\ell}{j} (q^{\ell-d^*+1}-1) & \ell \geq d^* \end{cases} \quad (3.17)$$

where A_ℓ is the number of codewords of weight ℓ and d^* is the minimum distance. We also have the result that the number of error patterns of weight h that are at a distance s from a particular codeword of weight ℓ is [11, p. 441]

$$N(\ell,h;s) = \sum_{\substack{0 \leq i \leq n \\ 0 \leq j \leq n \\ i+2j+h=s+\ell}} \binom{n-\ell}{j+h-\ell} \binom{\ell}{i} \binom{\ell-i}{j} (q-1)^{j+h-1} (q-2)^i \quad (3.18)$$

With these results we find that the probability of a decoding error is [11, p. 441]

$$P_e = \sum_{h=0}^{n} \left(\frac{P}{q-1}\right)^h (1-P)^{n-h} \sum_{s=0}^{t} \sum_{\ell=1}^{n} A_\ell \, N(\ell,h;s) \quad (3.19)$$

where P is the probability of a symbol error.

These formulae apply to the (255,195) Reed-Solomon code, but because the (240,180) shortened code has effectively 15 extra parity check bits, its performance should be even better. Thus, equation (3.19) was evaluated to upper bound the probability of a false decoding. Note that because the probabilities in equation (3.19) are very small, while the numbers A_ℓ and N are very large, this evaluation had to be performed using extended precision arithmetic.

The results are shown in Table 3.6 for the (255,195) code, along with the results for simpler codes with proportionate error correction capabilities. The probability of a false decoding for this code is less than 10^{-33}, that is, essentially zero. The smallness of these numbers may make one suspicious of the numerical accuracy of the results, but the software has been compared to results in the literature [15] for the (31,23) code and they match exactly. We therefore feel confident in these results.

P	Probability of False Decoding			
	RS(31,23)	RS(63,47)	RS(127,97)	RS(255,195)
.6	$1.5.10^{-2}$	7.3×10^{-6}	1.5×10^{-13}	3.0×10^{-34}
.5	1.4×10^{-2}	6.4×10^{-6}	1.3×10^{-13}	2.7×10^{-34}
.4	1.2×10^{-2}	5.5×10^{-6}	1.1×10^{-13}	2.3×10^{-34}
.2	6.0×10^{-3}	3.1×10^{-6}	7.1×10^{-14}	1.4×10^{-34}
.1	1.2×10^{-3}	3.6×10^{-6}	6.5×10^{-15}	5.6×10^{-36}
.01	6.6×10^{-8}	1.7×10^{-14}	3.0×10^{-27}	2.6×10^{-59}

Table 3.6: Probability of false decoding versus probability of input symbol error for various Reed-Solomon codes.

4.0 SUMMARY AND CONCLUSIONS

The three objectives of this study were the software implementation and simulation of maximum likelihood detection of continuous phase modulated signals over Rayleigh fading channels, simulating the performance over Rician fading channels of a (240,180) shortened Reed-Solomon code concatenated with a 8-state trellis code, and the preparation of a software user's guide for the complete software package.

Under the reasonable assumptions of finite frequency pulse duration for the CPM signal and a finite duration linear predictor for the composite fading and noise spectrum, it was shown that MLSE of CPM signals over Rayleigh flat fading channels could be implemented using the Viterbi algorithm. It was also shown that the branch metric calculations required in the Viterbi algorithm could be implemented as a bank of FIR filter and square circuits. The resulting algorithm, while not simple in terms of complexity, could be implemented using existing digital signal microprocessor technology for lower data rates, that is, less than 5 kbps. The performance of this algorithm was estimated through computer simulation. The results showed that MLSE was comparable in performance to more conventional techniques such as 2-DPSK and differentially detected CPM when the fading bandwidth to bit rate ratio was low, less than .01. However, at higher fading rates, MLSE performed remarkably better. In fact, at a fading bandwidth to bit rate ratio of 0.3, these conventional techniques were totally unreliable, while MLSE, in conjunction with a constraint length 7 rate 1/2 convolutional code, could provide very good data performance.

With regard to the second task, it was found that the concatenated coding scheme provided very good performance. Over Rician fading channels with K-factors ranging from $-\infty$ to 0 dB and for 60 and 120 Hz fading, the concatenated coding scheme was found to provide perfect performance (no block errors or rejections) for E_b/N_o ratios greater than 8 dB. It was also found that the Reed-Solomon coding scheme has very powerful error detection properties, with the probability of falsely decoding a block being less than 10^{-33}. With this powerful coding scheme, it was found that the obvious approach to providing erasure information was not beneficial; the major effect being to increase the rejection rate.

The software users' guide, for the LINKSIM simulation program which was used to generate most of the above performance results, is included as a separate document.

REFERENCES

[1] M. Moher, "A-BPSK with Convolutional Coding", MCS Final Report No. 8811, SSC File No.: 660ER-7-0003/96, November 1987.

[2] B. Sayar and S. Pasupathy, "Nyquist 3 Pulse Shaping in Continuous Phase Modulation", IEEE Trans. on Comm., vol. COM-35, pp. 57-66, January 1987.

[3] J. H. Lodge and M. L. Moher, "Time Diversity for Mobile Satellite Channels Using Trellis Coded Modulation", GLOBECOM'87.

[4] P. Galko and S. Pasupathy, "Linear Receivers for Correlatively Coded MSK", IEEE Trans. Comm., vol. COM-33, pp. 338-347, April 1985.

[5] S. Bellini et al, "Noncoherent Detection of Tamed Frequency Modulation", IEEE Trans. Comm., vol. COM-32, pp. 218-224, March 1984.

[6] S. Bellini and G. Tartara, "Efficient Discriminator Detection of Partial-Response Continuous Phase Modulation", IEEE Trans. Comm., vol. COM-33, pp. 883-886, August 1985.

[7] G. Ungerboeck, "Adaptive Maximum-Likelihood Receiver for Carrier-modulated Data-transmission Systems", IEEE Trans. Comm., vol. COM-22, pp. 624-636, May 1974.

[8] J. Makhoul, "Linear Prediction: A Tutorial Review", Proc. IEEE, vol. 63, pp. 561 - 580, April 1975.

[9] J. G. Proakis, Digital Communications, McGraw-Hill, 1983.

[10] J. H. Lodge et al, "A Comparison of Data Modulation Techniques for Land Mobile Satellite Channels", IEEE Trans. Veh. Tech., vol. VT-36, pp. 28-35, February 1987.

[11] R. E. Blahut, Theory and Practice of Error Control Codes, Addison Wesley, 1983.

[12] G. C. Clark and J. B. Cain, Error-Correction Coding for Digital Communications, Plenum Press, 1981.

[13] R. McEliece, The Theory of Information and Coding, Encyclopedia of Mathematics and Its Applications, Addison-Wesley, 1977.

[14] M. Moher, "Time Diversity Using Trellis Coded Modulation Schemes", MCS Final Report No. 8726, SCC File No.: 27ST-36001-6-3547, January 1987.

[15] Z. Huntoon and A. Michelson, "On the Computation of the Probability of Post-Decoding Error Events for Block Codes", IEEE Trans. Inf. Th., vol. IT-23, pp. 399-403, May 1977.