Turbo Codes for Noncoherent FH-SS With Partial Band Interference

Joseph H. Kang, *Student Member, IEEE,* and Wayne E. Stark, *Fellow, IEEE*

***Abstract*—In this paper, turbo codes are investigated in a slow frequency-hopped spread spectrum (FH-SS) system with partial-band jamming. In addition, full-band thermal noise is present. The channel model is that of a partial-band jammer in which a fraction of the frequency band is jammed and the remaining fraction is unjammed. This paper focuses on the implemention and performance of a modified turbo decoder for this model. We refer to the knowledge that each transmitted bit is jammed as channel state information. We consider cases of known or unknown channel state and variable number of bits per hop. Our approach is to modify the calculation of branch transition probabilities inherent in the original turbo decoder. For the cases with no side information and multiple bits per hop, we iteratively calculate channel state estimates. Analytical bounds are derived and simulation is performed for noncoherent demodulation. The performance of turbo codes is compared with a Reed–Solomon and a concatenated code comprised of a convolutional inner code and Reed–Solomon outer code.**

***Index Terms*—Concatenated coding, frequency hop communication, spread spectrum communication.**

I. Introduction

TURBO codes are an exciting new channel coding scheme that achieve data communication at signal-to-noise ratios close to the Shannon limit. The results published in the inaugural paper by Berrou *et al*. [1] were so good that they were met with much skepticism by the coding community. Since then, however, these results have been reproduced and even improved [2]. Consequently, much of the present research is focused on applying turbo codes to different systems.

The encoder described in [1] is formed using a parallel concatenation of two or more component encoders. Note that serial concatenated codes have been proposed [3], but are not considered in this paper. If the input block has N information bits, the encoded bit stream is made up of the uncoded data bits and the parity bits of the component encoders. The key element of the encoder is the use of an interleaver which permutes the information sequence and then uses this as the input to the second component encoder. In general, this permutation allows low weight outputs of the first component encoder to result in high weight outputs of the second component encoder. Thus, the combination of the encoders might contain favorable distance properties, even if each component encoder does not.

Paper approved by S. S. Pietrobon, the Editor for Coding Theory and Techniques of the IEEE Communications Society. Manuscript received December 15, 1997; revised March 26, 1998 and June 8, 1998.

The authors are with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor, MI 48109 USA.

Publisher Item Identifier S 0090-6778(98)08118-5.

It is well known that a randomly chosen code of sufficiently large block length is capable of approaching channel capacity [4]. In general, however, the complexity of maximum likelihood decoding such a code increases exponentially with block length up to the point where decoding becomes physically unrealizable. The encoder mimics random codes by making use of a large random interleaver. While turbo coding performance also improves for increasing interleaver lengths, the decoding complexity grows only linearly, making the decoding of large block lengths possible. Note that a turbo decoder does not perform maximum likelihood decoding directly, but attempts to achieve maximum likelihood decoding in an iterative way. The original turbo decoder [1] used two MAP algorithm decoders. There are other less complex algorithms that can be used in place of the MAP algorithm for each decoder such as SOVA and Max-Log-MAP [5]. However, because these other algorithms are suboptimal, they reduce the complexity of decoding at the cost of performance. Hence, for the purposes of this paper, we will consider the turbo decoder where each component decoder uses the MAP algorithm to calculate *a posteriori* likelihood estimates for each bit.

The potential of turbo codes can be best exemplified by its successful application to deep space communications. Using MAP decoders, turbo codes (16 state constituent codes, overall rate $1/2$) were shown to outperform the concatenated code of the Voyager, Galileo, and Cassini missions. The gain, for instance, over the Voyager code (rate $1/2$, constraint length 7 convolutional code concatenated with a $(255\,223)$ Reed–Solomon code) is approximately 1.5 dB at a BER of 10^{-5}. The primary difference is that the turbo code has greater decoding complexity than the Voyager code.

Thus, we arrive at the primary disadvantage of using turbo codes with the MAP algorithm. Decoding complexity for turbo codes is proportional to the block length, the number of decoding iterations, and the constraint length of the constituent codes. The MAP decoder is approximately four times more complex than the Viterbi algorithm and this must be iterated several times. It is known that turbo code performance generally increases with interleaver or block length [6]. In fact, Berrou *et al.* showed a bit error rate of 10^{-5} at 0.7 dB, but needed to use 18 decoding iterations and a block length of 65 536 bits. The large amount of computation required for turbo decoding explains why much of the current research is focused on reduced complexity decoders [5], [7]–[9].

In packet data communications, the use of error correction codes plays a key role in achieving low packet error rates. When transmitting speech, large processing delay is

unacceptable and higher error rates are tolerable. In data communication, low error rates are more important and delay at the decoder is more acceptable. Therefore, it would appear that turbo codes are possible candidates for packet data communications.

One packet data network of interest is slow-frequency hop radios. There has been considerable interest in enhancing slow frequency hop radios so they can be integrated into a packet radio network [10], [11]. One such enhancement would be improved error correction coding. While research on the application of Reed–Solomon codes and concatenated (Reed–Solomon and convolutional) codes to frequency-hopped spread spectrum (FH-SS) systems in partial-band interference have shown promising results [10], [12], it would be interesting to see how well turbo codes perform.

Turbo codes were first considered for an FH-SS system in [13] where performance was analyzed versus spectral efficiency. The model, however, did not include partial-band interference or memory. In [14], turbo codes were considered for coherent FH-SS with partial-band interference. In this paper, we investigate the performance of turbo codes in noncoherent FH-SS with partial-band interference.

The outline of this paper is as follows. In Section II, the system model including details of the FH-SS model is described and turbo codes are briefly reviewed. In Section III, the modifications to the turbo decoder necessary for FH-SS are described. Analytical performance bounds are derived in Section IV. In Section V, simulation results are presented and compared to the performance of other well-known coding techniques. In Section VI, the numerical results of our analytical bounds are discussed. Finally, we discuss the potential of applying turbo codes to practical FH-SS systems in our conclusion of Section VII.

II. System Model

A. Transmitter

The encoder is formed by concatenating two constituent codes in parallel and separating the codes by an interleaver. As in the original work by Berrou *et al.* [1], the constituent codes are recursive systematic convolutional codes. The encoder takes as input the data sequence $d_k \epsilon \{0,1\}$ of length N and then outputs three streams: the data bits d_k, the parity bits $p_{1,k}$ of the first component encoder with input d_k, and the parity bits $p_{2,k}$ of the second component encoder with interleaved d_k as input. The modulation considered is binary frequency shift keying (BFSK). The resultant signal is frequency hopped. The hopping patterns of the FH-SS system are modeled as sequences of independent random variables uniformly distributed over the allowable frequency range.

B. Channel

It is assumed that there exists an on–off jammer that will evenly distribute its power over a fraction ρ of the frequency range. Thus, transmission occurs over a channel that includes full-band thermal noise with double-sided power spectral density $N_0/2$ and partial band interference with double-sided power spectral density $N_J/2\rho$ which covers a fraction ρ of the band. As a result, there are essentially two channel states: jammed and unjammed. The probability of hopping to a jammed state is ρ, and the probability of hopping to an unjammed state is $1-\rho$. It is assumed that the jammer stays on for the entire duration of the hop if it is jammed at all. Let $(y_{1,k}, y_{2,k}, y_{3,k})$ be the outputs of the channel and let $z_{i,k}$ represent the channel state for $y_{i,k}$. Jammed and unjammed states correspond to $z_{i,k}=1$ and $z_{i,k}=0$, respectively.

C. Original Turbo Decoder

Turbo decoding is an iterative procedure which makes use of the MAP algorithm. The derivation of this algorithm has been well documented in previous papers [1], [15], [16]. Let S_k be the state of the first encoder at time k, and let L_k be the log likelihood ratio (LLR) of the *a posteriori* probabilities. Then

$$L_k = \log \frac{\Pr(d_k = 1 \mid \underline{y}_1, \underline{y}_2)}{\Pr(d_k = 0 \mid \underline{y}_1, \underline{y}_2)}. \tag{1}$$

The branch transition probabilities used by the MAP algorithm are calculated as

$$\begin{aligned}\gamma_i(y_{1,k}, y_{2,k}, m', m) = {} & p(y_{1,k} \mid d_k = i, S_k = m, S_{k+1} = m') \\ & \cdot p(y_{2,k} \mid d_k = i, S_k = m, S_{k+1} = m') \\ & \cdot p(S_{k+1} = m' \mid d_k = i, S_k = m) \\ & \cdot p(d_k = i \mid S_k = m)\end{aligned} \tag{2}$$

where $p(S_{k+1} = m' \mid d_k = i, S_k = m) = 1$ if bit i is associated with the given state transition and equals zero if it is not. $p(d_k = i \mid S_k = m) = p(d_k = i)$ depends on the *a priori* probabilities of the information bits d_k. It can be shown that

$$L_k = L'_k + L_{1,k} + L_k^{(2)} \tag{3}$$

where

$$L'_k = \log \frac{p(d_k = 1)}{p(d_k = 0)} \tag{4}$$

$$L_{1,k} = \log \frac{p(y_{1,k} \mid d_k = 1)}{p(y_{1,k} \mid d_k = 0)}. \tag{5}$$

$L_k^{(2)}$ is termed the *extrinsic* portion of the log likelihood ratio. It is used as *a priori* information for the second MAP (MAP2) decoder. The concept of extrinsic information is important in that it prevents information introduced by one of the component decoders to be passed back to that component decoder. Let $L^{(2)}$ and $L^{(3)}$ correspond to the extrinsic information generated by the MAP1 and MAP2 decoders, respectively. Assuming equally likely transmitted data, the combined MAP LLR for bit d_k is

$$\tilde{L}_k = L_{1,k} + L_k^{(2)} + L_k^{(3)}. \tag{6}$$

Thus, the decoding algorithm is as follows. Initially, $L'_k = 0$. The output of MAP1 is $L_{1,k} + L_k^{(2)}$. For MAP2, we let $L'_k = L_k^{(2)}$ (i.e., we let the extrinsic information generated by MAP1 become the *a priori* information for MAP2). The output of MAP2 is $\tilde{L}_k$. Similarly on the next iteration, we let

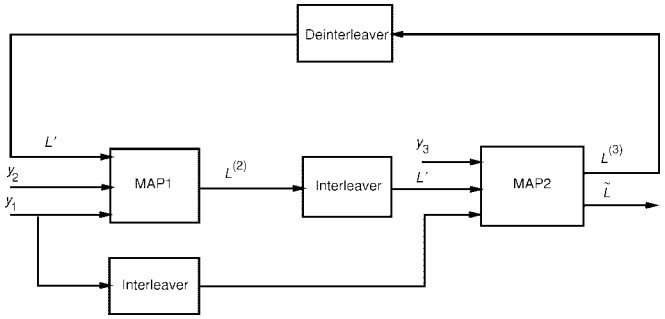

Fig. 1. Block diagram of the turbo decoder.

$L'_k = L_k^{(3)}$. This process is iterated and eventually converges to some low bit error rate (BER), where the final bit decision uses (6). The structure of the turbo decoder is shown in Fig. 1. Note that because the log likelihood ratio is the ratio of *a posteriori* information bit probabilities, the turbo decoding process can be viewed as the iterative improvement of *a posteriori* information bit probabilities. We will use this when we consider the turbo decoder for FH-SS systems.

III. Turbo Decoder for FH-SS Systems

The turbo decoding algorithm is dependent on what information is available to the turbo decoder. We examine the cases where knowledge of the channel state (i.e., jammed or unjammed) is either available or unavailable to the decoder. The case of known channel state will be referred to as side information (SI). In addition, the cases of independent and identically distributed (IID) transmission (i.e., one bit per hop) and transmission over a channel with memory (i.e., h bits per hop) are considered.

For all cases, the power spectral densities of the channel noise, $N_0/2$ and $N_J/2\rho$, are assumed to be known to the decoder. These values are necessary to compute the branch transition probabilities in (2). Note, however, that exact SNR values are not necessary to achieve good decoding performance. By [17] and as well by our own investigation, the performance of turbo codes is not sensitive to SNR mismatch. Thus, low complexity SNR estimation algorithms can be implemented to achieve similar performance to the case where the SNR is exactly known.

A. FH-SS Without Memory

In this section, we consider the case of one bit per hop. If the channel state is unknown, then the modified turbo decoder for IID FH-SS needs only to adapt the calculation of branch transition probabilities. More specifically, (2) is calculated using

$$p(y_{j,k} \mid d_k = i, S_k = m, S_{k+1} = m') = \sum_{z=0}^{1} p(y_{j,k} \mid d_k = i, S_k = m, S_{k+1} = m', z_{j,k} = z) \cdot p(z_{j,k} = z). \quad (7)$$

If the channel state is known, we treat the side information as information received by the decoder. Thus, for branch transition probability computations, we calculate the joint conditional probability of the channel output $y_{j,k}$ and the appropriate channel state $z_{j,k}$ as

$$p(y_{j,k}, z_{j,k} = z \mid d_k = i, S_k = m, S_{k+1} = m') = p(y_{j,k} \mid d_k = i, S_k = m, S_{k+1} = m', z_{j,k} = z) \cdot p(z_{j,k} = z). \quad (8)$$

Having described our modifications for the one bit per hop case, we will move on to the more interesting case with multiple bits per hop.

TABLE I
Structure of the Frequency Hopper

HOP	1	$c_{1,1}$	$c_{2,L+1}$	$c_{3,2L+1}$	$c_{1,3L+1}$	...
.		.	.	.	.	...
.		.	.	.	.	...
HOP	L	$c_{1,L}$	$c_{2,2L}$	$c_{3,3L}$	$c_{1,4L}$	...
HOP	$L+1$	$c_{2,1}$	$c_{3,L+1}$	$c_{1,2L+1}$	$c_{2,3L+1}$	...
.		.	.	.	.	...
.		.	.	.	.	...
HOP	$2L$	$c_{2,L}$	$c_{3,2L}$	$c_{1,3L}$	$c_{2,4L}$	...
HOP	$2L+1$	$c_{3,1}$	$c_{1,L+1}$	$c_{2,2L+1}$	$c_{3,3L+1}$	...
.		.	.	.	.	...
.		.	.	.	.	...
HOP	$3L$	$c_{3,L}$	$c_{1,2L}$	$c_{2,3L}$	$c_{3,4L}$	...

B. FH-SS With Memory

In this section, we discuss our modification to the turbo decoder for the case of multiple bits per hop. First, we detail the design of the hopping structure. The hopping structure is important for cases with memory because unlike the IID case where jammed hops affect single bits, jammed hops now affect multiple bits. For instance, if $c_{1,k}$, $c_{2,k}$, and $c_{3,k}$ are the coded bits which correspond to information bit d_k, then it would be beneficial to send these bits over separate hops. If they were sent over the same hop and that hop was jammed, it would be difficult to decode the information bit correctly. Because the convolutional encoders display memory (i.e. $c_{2,k}$ is dependent on $c_{1,k}$, $c_{1,k-1}$, $c_{1,k-2}$, and so on), it makes sense for similar reasons to separate consecutive coded bits by as much as possible. Using $L = N/h$ where h is the number of bits per hop and N is the number of information bits per packet, the structure of the hopper is shown in Table I.

For cases with multiple bits per hop and no side information, we attempt to compensate for the lack of side information by generating estimates of the channel. Our approach is to calculate *a posteriori* probabilities for each channel state $p(z_k \mid \underline{y}_1, \underline{y}_j)$ and send this information in addition to $p(d_k \mid \underline{y}_1, \underline{y}_j)$ between decoders. Thus, information bit estimates and channel state estimates can be iteratively improved. The use of channel estimates to calculate branch transition probabilities should lead to improved error rates.

Note that the structure of the hopper shown in Table I allows channel estimates to be calculated in a manner similar to the way information bit estimates are calculated in the original turbo decoder. For instance, each MAP decoder in the original turbo decoder calculates *a posteriori* probabilities of the information bits given two of the three received observation vectors.

This information is passed to the next MAP decoder which uses this information as *a priori* information bit probabilities. Similarly, by using the structure of the hopper in Table I, two-thirds of the relevant observation sequence is available to each MAP decoder so that *a posteriori* probabilities for each hop can be calculated. This information can be passed between MAP decoders and be used as *a priori* channel state probabilities.

We define $\underline{R}$ as the vector of received channel outputs that is available to the MAP decoder, $\underline{R}_k$ as the subset of $\underline{R}$ that has been received over a given hop with state z_k, and $\tilde{\underline{R}}_k$ as the subset of $\underline{R}$ that has not been received over the hop with state z_k. Thus, $\underline{R} = \underline{R}_k \cup \tilde{\underline{R}}_k$ where $\underline{R}_k = (\underline{R}_{k,1}, \cdots, \underline{R}_{k,h})$. The calculation of state estimates is shown below

$$p(z_k = z \mid \underline{R}) = \frac{p(\underline{R} \mid z_k = z) \cdot p(z_k = z)}{p(\underline{R})} \tag{9}$$

$$= \frac{p(\underline{R}_k \mid z_k = z) \cdot p(\tilde{\underline{R}}_k \mid z_k = z) \cdot p(z_k = z)}{p(\underline{R})} \tag{10}$$

$$= p(\underline{R}_k \mid z_k = z) \cdot p(z_k = z) \cdot \frac{p(\tilde{\underline{R}}_k)}{p(\underline{R})} \tag{11}$$

$$= p(\underline{R}_k \mid z_k = z) \cdot p(z_k = z) \cdot K \tag{12}$$

$$\approx p(\underline{R}_k \mid z_k = z) \cdot \tilde{p}(z_k = z \mid \underline{R}_k) \cdot K \tag{13}$$

where K is a normalizing factor chosen to make the probability density function sum to one and $\tilde{p}(z_k = z \mid \underline{R}_k)$ is the channel estimate provided by the previous MAP decoder.

In the above equation $\tilde{p}(z_k = z \mid \underline{R}_k)$ is used in place of $p(z_k = z)$ to take advantage of the state estimate provided by the previous MAP decoder. In order to compute each state estimate, we must calculate the conditional joint probabilities of $\underline{R}_k$ in (14). This can be calculated by performing total probability on $p(\underline{R}_k \mid z_k = z)$ over the respective coded bits

$$p(\underline{R}_k \mid z_k = z) = \sum_{\underline{c}_k} p(\underline{R}_k \mid \underline{c}_k = \underline{c}, z_k = z) p(\underline{c}_k = \underline{c}) \tag{14}$$

where $\underline{c}_k$ represents the vector of coded bits respective to $\underline{R}_k$.

This will require *a priori* probability knowledge of the coded bits. But, the MAP decoders are already sending log likelihood ratios that give $p(d_k = d \mid \underline{R})$. Using this information, $p(c_{i,k} = c) \approx p(c_{i,k} = c \mid \underline{R})$ can be calculated. Thus, as the MAP decoders refine their estimates of $p(d_k = d \mid \underline{R})$, estimates of $p(z_k \mid \underline{R})$ are also getting more refined.

Note that as h, the number of bits per hop, increases, the complexity of directly computing $p(\underline{R}_k \mid z_k = z) = p(R_{k,1}, \cdots, R_{k,h} \mid z_k = z)$ rises exponentially. To overcome this problem, we developed the following recursion.

1) Initial Case:

$$p(z_k = z \mid R_{k,1}) = \frac{p(R_{k,1} \mid z_k = z) \cdot p(z_k = z)}{p(R_{k,1})} \tag{15}$$

$$= \sum_{c=0}^{1} p(R_{k,1} \mid z_k = z, c_{k,1} = c) \cdot p(c_{k,1} = c) \cdot \frac{p(z_k = z)}{p(R_{k,1})} \tag{16}$$

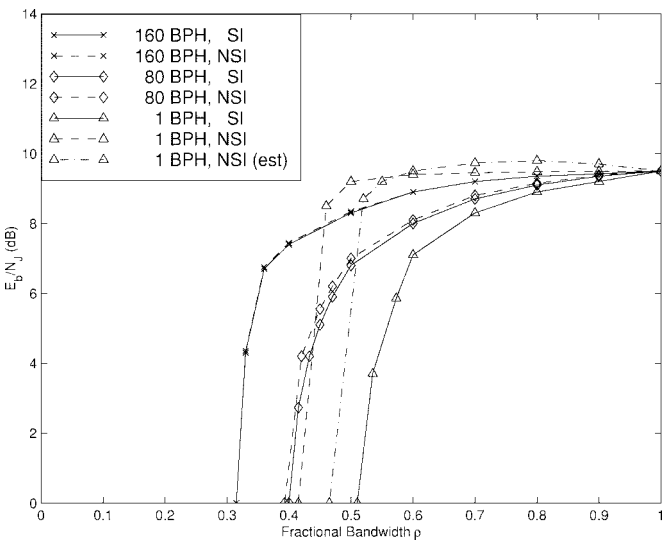

Fig. 2. Simulation results of turbo codes in noncoherent FH-SS.

$$\approx \sum_{c=0}^{1} p(R_{k,1} \mid z_k = z, c_{k,1} = c) \cdot p(c_{k,1} = c) \cdot \frac{\tilde{p}(z_k = z \mid \underline{R}_k)}{p(R_{k,1})} \tag{17}$$

where $\tilde{p}(z_k = z \mid \underline{R}_k)$ is the channel state estimate of the previous MAP decoder.

2) Recursion:

$$p(z_k = z | R_{k,1}, R_{k,2}, \cdots, R_{k,i-1}, R_{k,i}) = \frac{p(R_{k,i} | z_k = z, R_{k,1}, .., R_{k,i-1}) p(z_k = z | R_{k,1}, \cdots, R_{k,i-1})}{p(R_{k,i} | R_{k,1}, \cdots, R_{k,i-1})}$$

$$\approx \frac{p(R_{k,i} | z_k = z) \cdot p(z_k = z | R_{k,1}, \cdots, R_{k,i-1})}{p(R_{k,i} | R_{k,1}, \cdots, R_{k,i-1})} \tag{18}$$

$$\approx \frac{p(R_{k,i} \mid z_k = z) \cdot p(z_k = z \mid R_{k,1}, \cdots, R_{k,i-1})}{p(R_{k,i} \mid R_{k,1}, \cdots, R_{k,i-1})} \tag{19}$$

Note that (19) is an approximation since $R_{k,i}$ is lightly correlated with $R_{k,i-1}, \cdots, R_{k,1}$.

Once the state estimates have been computed, they are ready to be used in a turbo decoder. We use state estimates in a manner analogous to the way that a turbo decoder uses information bit estimates. When there is no SI, the *a priori* state probabilities are replaced by the *a posteriori* state probabilities for branch transition probability calculations. As in the IID case, the appropriate *a priori* probability is used for cases with side information. Thus, for the MAP1 decoder with $j = 1, 2$

$$p(y_{j,k} \mid d_k = i, S_k = m, S_{k+1} = m') = \sum_{z=0}^{1} p(y_{j,k} \mid d_k = i, S_k = m, S_{k+1} = m', z_{j,k} = z) \cdot p(z_{j,k} = z) \tag{20}$$

$$\approx \sum_{z=0}^{1} p(y_{j,k} \mid d_k = i, S_k = m, S_{k+1} = m', z_{j,k} = z) \cdot p(z_{j,k} = z \mid \underline{y}_1, \underline{y}_3)) \tag{21}$$

when there is no side information and

$$p(y_{j,k}, z_{j,k} = z \mid d_k = i, S_k = m, S_{k+1} = m') = p(y_{j,k} \mid d_k = i, S_k = m, S_{k+1} = m', z_{j,k} = z) \cdot p(z_{j,k} = z) \tag{22}$$

when there is side information.

IV. Analytical Bounds

It is often impractical to achieve simulated results for extremely low BER's. As a result, bounds are often calculated. Here, we invoke the Union–Bhattacharyya bound to obtain an upper bound on the probability of error.

Turbo codes are linear, so without loss of generality, we will assume that the all-zeros codeword was transmitted. If A_d is the weight enumerator of the code, $P_2(d)$ is the pairwise error probability between the all-zeros codeword and a codeword of weight d, and D is the Bhattacharyya parameter where $P_2(d) \leq D^d$, then the bound for an (n, k) block code is

$$P_{\text{word}} \leq \sum_{d=d_{\min}}^{n} A_d P_2(d) \tag{23}$$

$$\leq \sum_{d=d_{\min}}^{n} A_d D^d. \tag{24}$$

It was shown in [18] that with noncoherent reception, optimal decoding with side information leads to

$$D = \begin{cases} \left[\displaystyle\int_0^\infty u e^{-u^2/2} I_0^{1/2}(u\sqrt{2E_s/N_J})\, du\right]^2, & E_s/N_J < 2.87 \\ \dfrac{1.424}{E_s/N_J}, & E_s/N_J \geq 2.87. \end{cases}$$

Square-law combining is a suboptimal method of decoding in additive white Gaussian noise (AWGN), but has been shown to have an approximate performance loss of 0.14 dB for reasonable SNR's [18]. Because square-law combining is suboptimal, an upper bound on its performance will also be an upper bound to the performance of optimal decoding. The Bhattacharyya parameter for square-law combining in worst case jamming is $(\Gamma = E_s/N_J)$

$$D = \begin{cases} \left[\dfrac{1}{1-\lambda^2} \exp\left\{-\dfrac{\lambda\Gamma}{(1+\lambda)}\right\}\right], & \Gamma < 3 \\ \dfrac{4e^{-1}}{\Gamma}, & \Gamma \geq 3 \end{cases} \tag{25}$$

$$\lambda = \frac{\sqrt{(2+\Gamma)^2 + 8\Gamma} - (2+\Gamma)}{4}. \tag{26}$$

The only known way to exactly calculate A_d is via an exhaustive search involving all possible input sequences. One solution is to calculate an average upper bound by computing an average weight function over all possible interleaving schemes [19]. If the average weight function is defined as

$$\bar{A}_d = \sum_{i=0}^{k} \binom{k}{i} p(d \mid i) \tag{27}$$

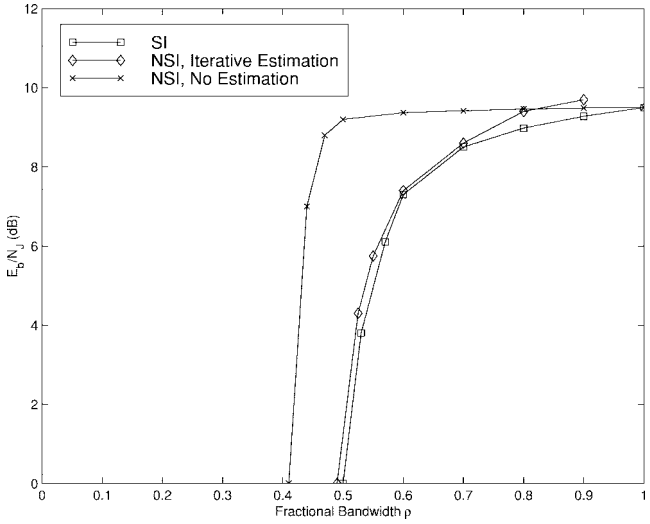

Fig. 3. Simulation results for 20 bits per hop.

where $p(d \mid i)$ is the probability that an interleaving scheme maps an input weight of i to produce a codeword of total weight d, and $\binom{k}{i}$ is the number of input frames with weight i. An algorithm for calculating $p(d \mid i)$ was described in [19]. Thus,

$$\bar{P}_{\text{word}} \leq \sum_{d=d_{\min}}^{n} \sum_{i=1}^{k} \binom{k}{i} p(d \mid i) D^d. \tag{28}$$

V. Simulation Results

For all simulations, the component encoders are rate $1/2$ recursive systematic convolutional encoders with memory four and octal generators $(37, 21)$. The packet size is 1760 information bits and the number of decoder iterations is five. A helical interleaver [20] is used to guarantee trellis termination. The SNR of the full-band thermal noise is set to 20 dB. Cases with memory are simulated using 20, 80, and 160 bits per hop (BPH).

Fig. 2 shows the plot of minimum E_b/N_J needed to achieve a packet error rate (PER) of 10^{-3} for a given ρ. As would be expected, the cases with side information (SI) performed better than their counterparts without SI (NSI). The SI and NSI curves only meet when $\rho = 1.0$. In this case, all states are jammed, so side information does not provide any additional information.

Notice that the graphs in Fig. 2 exhibit a tradeoff as the number of bits per hop increases. For any memory, no SI case, where channel states are iteratively estimated, performance is obviously upper bounded by the memory SI case. Because channel state estimates will improve if the number of bits per hop increases, we should be able to get arbitrarily close to the corresponding memory, SI result by increasing the memory. This is shown in Fig. 2 by examining the performance differences between corresponding SI and no SI cases for variable bits per hop. For memory, no SI cases, we were able to calculate reliable state information. These state estimates provided useful information which in turn aided the decoding process.

The downside of increasing the memory can be seen by analyzing the SI cases in Fig. 2. For all SI cases, knowledge of the channel state nulls out the advantage of more effective estimation. As a result, one might expect the performance of SI cases to be similar. However, as shown in Fig. 2, this is clearly not the case. The performance of SI cases degrades as the memory increases, but it uses fewer total number of hops (thereby making direct comparisons unfair). Thus, while increased memory leads to improved channel estimates in cases without SI, the upper bound for the performance of no SI cases effectively decreases. This tradeoff implies that for the NSI case with state estimation, there should be exist an optimal number of bits per hop that yields best performance. For $\rho \leq 0.7$ we found this value to be on the order of 20.

Due to inaccurate state estimation for large values of ρ, there is not a dwell interval length that is optimal for all values of ρ. To see this, first consider the results of the 1 bit per hop (IID) case in Fig. 2. While it is expected that the SI case outperforms the NSI case without state estimation, it is interesting to compare these results with the IID NSI case when state estimation is performed. For low values of ρ, the NSI case with state estimation yields better results than the NSI case without estimation. Even with a single bit of observation, state estimation is valuable if the disparity between the SNR's of the estimated states ($N_0/2$ and $N_J/2\rho$) is large. For larger values of ρ, however, it becomes difficult to distinguish between the two states. Inaccurate state estimates lead to an improper weighting in branch transition probability calculations and thus, overall performance degrades.

Next, consider the results for the 20 BPH case. In Fig. 3, simulation results for three decoding cases are presented: SI, NSI with iterative state estimation, and NSI with no state estimation. Note that for cases without SI, state estimation is beneficial for $\rho \leq 0.8$. At higher values of ρ, however, it becomes difficult to discern between the two channel states. Thus, channel state estimates become inaccurate and yield worse performance relative to the decoder which does not use state estimation. However, because the NSI, no state estimation decoder shows a loss of less than 1 dB with respect to the SI case for $\rho > 0.8$, one solution is to use a hybrid decoder which decides whether or not to calculate state estimates based on a threshold at $\rho = 0.8$

In addition, note that when state estimates are computed, the 20 BPH case performs more poorly than even the IID case for high values of ρ. Clearly, more observations will lead to more reliable state estimates. However if these estimates are inaccurate, they adversely affect the decoder calculations for multiple bits. Thus for each value of ρ, state estimation is beneficial only when there is a sufficient number of observations to guarantee a reliable estimate.

While we have discussed the results obtained when using five turbo decoding iterations, it is interesting to note the performance of the decoder after each iteration. The fast convergence of the turbo decoder is exhibited in Table II where the 160 BPH $(h = 160)$ cases both use $\rho = 0.8$, $E_b/N_J = 9.5$ dB, the 1 BPH NSI case without state estimation uses $\rho = 0.7$, $E_b/N_J = 9.5$ dB, and the 1 BPH SI case uses $\rho = 0.7$, $E_b/N_J = 8.5$ dB.

TABLE II
PERFORMANCE OF TURBO CODES BY ITERATION

PER	h=160, NSI	h=160, SI	h=1, NSI	h=1, SI
1 iter.	3.3E−01	2.8E−01	5.2E−01	4.6E−01
2 iter.	2.2E−03	1.8E−03	2.6E−03	2.1E−03
3 iter.	1.1E−03	1.1E−03	1.3E−03	1.2E−03
4 iter.	8.4E−04	8.5E−04	8.8E−04	8.6E−04
5 iter.	8.4E−04	8.4E−04	8.8E−04	8.6E−04

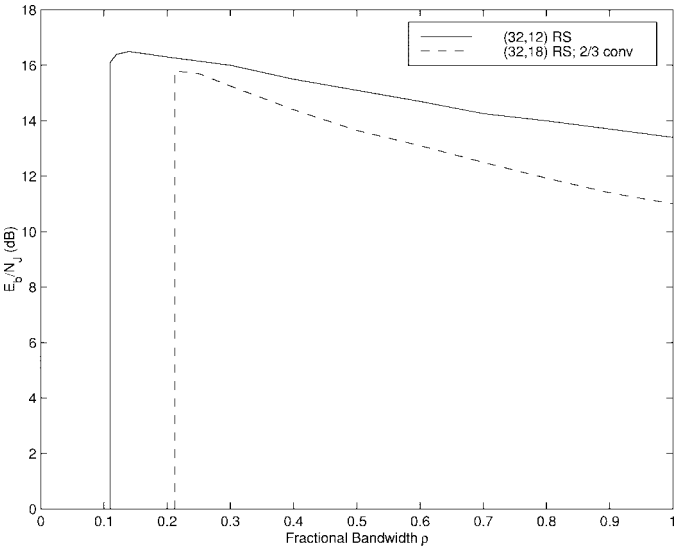

Fig. 4. Performance of other codes in noncoherent FH-SS.

We have shown that it is possible to bridge the gap between cases with and without side information by iteratively computing state estimates for a large number of bits per hop. However, the performance improvements may still be unsatisfactory. In the case of noncoherent reception, one possible improvement would be to perform phase estimation. If the phase during a hop is assumed to move very slowly and an orthogonal signal set is used, phase estimation can be performed using a method analogous to state estimation. In this manner, joint decoding and phase tracking can be achieved.

In order to gauge our results, we refer to the application of other coding methods to FH-SS systems. In particular, Pursley and Frank investigated the use of the Reed–Solomon code and the Reed–Solomon/convolutional concatenated code (RS-CC) in [10]. For the Reed–Solomon code with no inner code, they used a $(32, 12)$ errors-only RS code over $GF(2^5)$ with noncoherent reception and 27 codewords per packet. Thus, the code had rate $3/8$ and the total number of information bits per packet was 1620. There were 135 binary symbols per dwell period. For the concatenated code, they used a $(32, 18)$ RS code for the outer code and a rate $2/3$, constraint length 6 convolutional code with erasure threshold $\gamma = 3$ for the inner code. In addition, the convolutional code used soft decisions. The dwell interval spanned 159 binary symbols and there were 20 codewords per packet. The overall rate of the code was $3/8$ and the number of information bits per packet was 1800. Requiring a PER of 10^{-3}, these results are shown in Fig. 4.

Because the turbo code system with 160 BPH matches the parameters shown in Fig. 4 quite closely, we will use

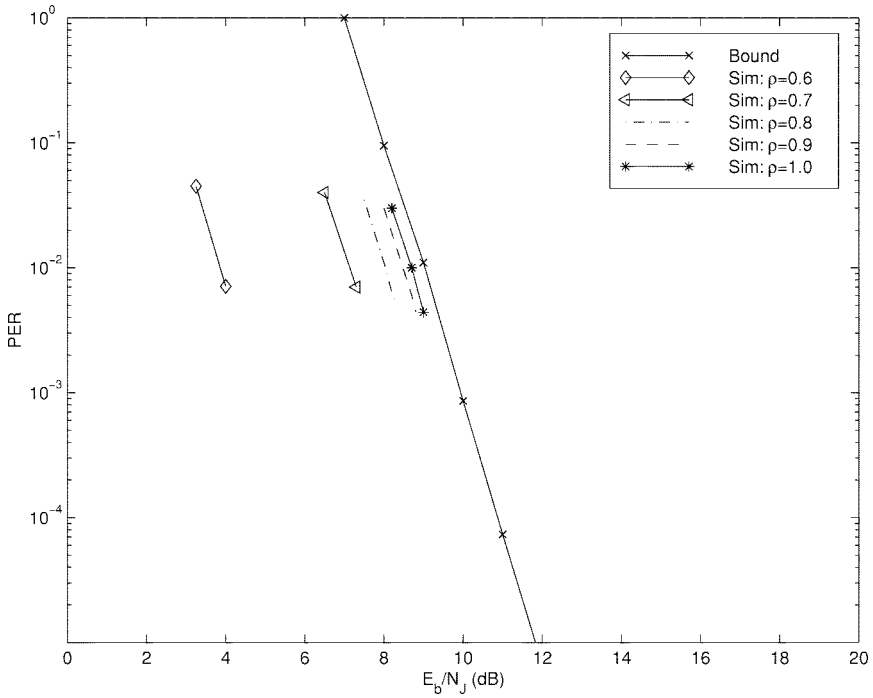

Fig. 5. Numerical results of turbo codes in noncoherent reception with side information.

this system for comparison. Notice the large performance difference between the turbo code and the coding schemes of Fig. 4. Comparing the worst-case performance of the turbo code system without side information with the RS and RS-CC systems, the turbo code system shows a gain of 6.9 and 6.3 dB, respectively. Note that to achieve worst-case performance for turbo codes, the jammer needs to jam the entire band ($\rho = 1$), while for RS and RS-CC codes, the jammer needs to jam only a small fraction of the band (about $\rho = 0.12$ and $\rho = 0.22$, respectively). Another performance measure of FH-SS systems is ρ^*, the minimum fractional jamming bandwidth required to induce any decoding errors. In this case, turbo codes save about 0.1 and 0.2, respectively.

While turbo codes seem to significantly outperform the RS and RS-CC codes, it is unfair to directly compare these results. First, the coding rates of the coding systems are different: $1/3$ for the turbo code and $3/8$ for the RS and RS-CC codes. However, this minor difference in code rate is not sufficient to explain the contrast in performance. Another difference is in the treatment of the memory case without side information. For turbo codes, we exploit the memory of the channel by estimating channel states and using this information in our branch transition probabilitiy calculations. For the RS-CC code, Pursley and Frank also use the memory to predict which hops have been jammed, but they do so in a very different way. The goal is to declare an erasure if the jamming in a dwell interval is sufficiently severe that many of the RS symbols are likely to be in error. The metric they use to estimate the channel is the Hamming distance between the binary code sequence chosen by the Viterbi decoder and the binary sequence that results from making hard decisions on the output of the demodulator. The resulting distance represents their estimate of the number of errors produced by the demodulator γ. The different methods at which the turbo code and RS-CC systems treat the uncertainty of the channel state makes it difficult to compare the performance of the systems. The final difference between the coding systems is that the turbo decoder is more computationally complex. Even with the recursions of the MAP decoders, these calculations are iterated many times. Thus, we arrive at the familiar tradeoff between computational complexity and performance.

VI. Numerical Results: Bounds

Fig. 5 shows the numerical results for noncoherent reception when side information is available to the receiver. In addition, some simulation results are included for reference. Note that the Bhattacharrya parameter D was calculated assuming worst case jamming. As shown in Fig. 5, the average upper bound calculated for noncoherent reception is tight for $\rho = 1.0$, the value which yields worst case jamming.

VII. Conclusion

We have shown that there exists great potential for noncoherently demodulated turbo codes in frequency-hop spread spectrum systems by analyzing cases with 1, 20, 80, and 160 bits per hop, and either with or without channel state side information. The case with 1 BPH and side information was uniformly superior relative to the other cases. We also witnessed performance degradation for the SI cases as the number of bits per hop increased due to the change in effective block length. However, as the length of the dwell period increased, performance differences between the SI and no SI cases tended to diminish due to effective channel state estimation. Finally, we compared our simulation to other coding schemes and found the comparison to be favorable toward turbo codes.

While turbo codes which use component MAP decoders are effective in reducing the E_b/N_J required to achieve a given packet error probability, there is considerable computational cost. Before turbo codes can be integrated into a packet radio network or any practical data communications system, the computational complexity needs to be reduced while minimizing performance losses. The work in this paper exemplifies the error correction power of turbo codes. Future research should investigate the application of lower complexity turbo decoders to FH-SS systems.

REFERENCES

[1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon limit error-correcting coding and decoding," in *Proc. ICC'93*, May 1993, pp. 1064–1070.

[2] C. Berrou, "Some clinical aspects of turbo codes," in *Int. Symp. Turbo Codes*, Sept. 1997, pp. 26-31.

[3] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial concatenation of interleaved codes: Performance analysis, design, and iterative decoding," *TDA Progress Rep.*, 42-126, JPL, Aug. 1996.

[4] T. Cover and J. Thomas, *Elements of Information Theory*. New York: Wiley-Interscience, 1991.

[5] P. Robertson, E. Villebrun, and P. Hoeher, "A comparison of optimal and sub-optimal MAP decoding algorithms operating in the log domain," in *Proc. ICC'95*, June 1995, pp. 1009-1013.

[6] S. Benedetto and G. Montorsi, "Design of parallel concatenated convolutional codes," *IEEE Trans. Comm.*, vol. 44, May 1996.

[7] C. Berrou, P. Adde, E. Angui, and S. Faudeil, "A low complexity soft-output Viterbi decoder architecture," in *Proc. ICC'93*, May 1993, pp. 737-740.

[8] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Algorithm for continuous decoding of turbo codes," *Electron. Lett.*, vol. 32, no. 4, Feb. 1996.

[9] S. Pietrobon, "Implementation and performance of a turbo/MAP decoder," *Int. J. Satellite Comm.*, to appear.

[10] C. Frank and M. Pursley, "Concatenated coding for frequency-hop spread-spectrum with partial-band interference," *IEEE Trans. Commun.*, vol. 44, pp. 377-387, Mar. 1996.

[11] J. Mathis et al., "Final design plan: Sincgars packet switch overlay," *SRI Tech. Rep.*, SRI Project no. 1244, July 1986.

[12] M. Pursley and W. Stark, "Performance of Reed–Solomon coded frequency-hop spread-spectrum communications in partial-band interference," *IEEE Trans. Commun.*, vol. COM-33, pp. 767-774, Aug. 1985.

[13] U. Fiebig and P. Robertson, "Soft decision decoding in fast frequency hopping systems with convolutional codes and turbo codes," in *Proc. ICC'96*, June 1996, pp. 968–973.

[14] J. Kang and W. Stark, "Turbo codes for coherent FH-SS with partial band interference," in *Proc. MILCOM'97*, Nov. 1997, pp. 5–9.

[15] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. Inform. Theory*, vol. 20, pp. 284–287, Mar. 1974.

[16] P. Robertson, "Illuminating the structure of code and decoder for parallel concatenated recursive systematic (turbo) codes," in *Proc. GLOBECOM'94*, Dec. 1994, pp. 1298–1303.

[17] T. Summers and S. Wilson, "SNR mismatch and online estimation in turbo decoding," *IEEE Trans. Commun.*, vol. COM-46, pp. 421–423, Apr. 1998.

[18] W. Stark, "Coding for frequency-hopped spread-spectrum communications with partial-band interference—Part II: Coded performance," *IEEE Trans. Commun.*, vol. COM-33, pp. 1045–1052, Oct. 1985.

[19] D. Divsalar, S. Dolinar, F. Pollara, and R. McEliece, "Transfer function bounds on the performance of turbo codes," TDA Progress Rep. 42-122, JPL, Aug. 1995.

[20] A. Barbulescu and S. Pietrobon, "Terminating the trellis of turbo codes in the same state," *Electron. Lett.*, vol. 31, pp. 22–23, Nov. 1995.

Joseph H. Kang (S'96) received the B.S. and M.Eng. degrees in electrical engineering from the Massachusetts Institute of Technology (MIT) Cambridge, both in 1995. He is currently working toward the Ph.D. degree in electrical engineering at the University of Michigan in Ann Arbor.

He has conducted research on radar scattering with the Research Lab of Electronics and video compression with the Media Lab, both at MIT. In addition, he was an intern with the Telecommunications Division at Daewoo Corp. His current research interests include error control coding, modulation techniques, and channel estimation for spread spectrum communications.

Wayne E. Stark (S'77–M'78–SM'94–F'98) received the B.S. (with highest honors), M.S., and Ph.D. degrees in electrical engineering from the University of Illinois, Urbana, in 1978, 1979, and 1982, respectively.

Since September 1982 he has been a faculty member in the Department of Electrical Engineering and Computer Science at the University of Michigan, Ann Arbor where he is currently Professor. From 1984–1989 he was Editor for Communication Theory of the IEEE Transactions on Communication in the area of Spread-Spectrum Communications. His research interests are in the areas of coding and communication theory, especially for spread-spectrum and wireless communication networks.

Dr. Stark was involved in the planning and organization of the 1986 International Symposium on Information Theory which was held in Ann Arbor, Michigan. He was selected by the National Science Foundation as a 1985 Presidential Young Investigator. He is pricipal investigator of a Army Research Office Multidisciplinary University Research Initiative project on Low Energy Mobile Communications. He is a member of Eta Kappa Nu, Phi Kappa Phi, and Tau Beta Pi.