54-61
64184
p. 12

N96-16642

Transfer Function Bounds on the Performance of Turbo Codes

D. Divsalar, S. Dolinar, and F. Pollara
Communications Systems and Research Section

R. J. McEliece
California Institute of Technology
and
Communications Systems and Research Section

In this article, we apply transfer function bounding techniques to obtain upper bounds on the bit-error rate for maximum-likelihood decoding of turbo codes constructed with random permutations. These techniques are applied to two turbo codes with constraint length 3 and later extended to other codes. The performance predicted by these bounds is compared with simulation results. The bounds are useful in estimating the "error floor" that is difficult to measure by simulation, and they provide insight on how to lower this floor. More refined bounds are needed for accurate performance measures at lower signal-to-noise ratios.

I. Introduction

Simulations have shown that turbo codes can produce low error rates at astonishingly low signal-to-noise ratios if the information block is large and the permutations are selected randomly [3,4]. In addition to simulations, it is also useful to have theoretical bounds that establish decoder performance in the range where obtaining sufficient data from simulations is impractical.

In this article, we apply transfer function bounding techniques to obtain upper bounds on the bit-error rate for maximum-likelihood decoding of turbo codes constructed with random permutations. The premise for these bounds is the same as for the usual transfer function bounds applied to standard convolutional codes [7]. The error probability is upper bounded by a union bound that sums contributions from error paths of different encoded weights. The state diagram of the code is used to enumerate the paths of each possible weight.

The transfer function bounds for turbo codes differ from the usual transfer function bounds for convolutional codes in several respects. For turbo codes, these bounds require a term-by-term joint enumerator for all possible combinations of input weights and output weights of error events, even for bounds on the word error rate. Second, because turbo codes are block codes, it is crucial to accurately enumerate "compound" error events that can include more than one excursion from the all-zero state during the fixed block length, as shown in Fig. 1. Third, since explicit results are intractable for any particular randomly chosen permutation, the bound is developed as a random coding bound. Finally, the bounds are derived as upper bounds on the bit-error rate of a maximum-likelihood decoder operating on

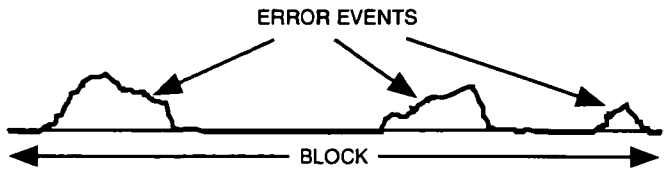

Fig. 1. Example of "compound" error events.

turbo-encoded data, whereas the iterative turbo-decoding procedure is not guaranteed to converge to the maximum-likelihood codeword. On the other hand, since the turbo decoder attempts to minimize the bit-error rate rather than the word-error rate, it can in some circumstances yield a slightly lower bit-error rate than a maximum-likelihood decoder.

Transfer function bounds for turbo codes were first published by Benedetto and Montorsi [1,2], but our computation method allows for more accurate numerical results. Our algorithm uses a short recursion formula to calculate the necessary transfer function coefficients efficiently for large block lengths. We have found that the bound "diverges" (i.e., becomes useless) at low E_b/N_0, as does the corresponding type of bound for standard convolutional codes.

Our approach in this article is first to explicitly develop the bounds for two exemplary turbo codes with constraint length 3. One code is the same reported in [2]. The second code should be superior to the first according to the heuristic arguments of [5]; it is presented to underscore the reasons why one code should outperform the other. Only after developing the two examples in detail do we extend the theory to additional codes.

II. Turbo Code Examples

In this section, we introduce the two particular turbo codes that will be used throughout this article to develop the bounds.

A. Encoder Diagrams

Figures 2(a) and 2(b) depict the two exemplary turbo encoders. Both encoders produce one uncoded output stream $\mathbf{x}_0$ and two encoded parity streams $\mathbf{x}_1, \mathbf{x}_2$, for an overall code rate of 1/3. The parity streams come from simple recursive convolutional encoders with constraint length $K = 3$ (i.e., memory $m = 2$). For the first code, the parity sequences both correspond to a ratio of generator polynomials g_a/g_b, where $g_a(D) = 1 + D + D^2$ and $g_b(D) = 1 + D^2$. For the second code, the parity sequences both correspond to g_b/g_a. Representing g_a as octal 7 and g_b as octal 5, the two codes are denoted by (1, 7/5, 7/5) and (1, 5/7, 5/7), respectively. The notation explicitly shows the method of generating each of the three output streams, one uncoded and two parity. We refer to these three separate rate-1 components of the code as *code fragments*. The turbo code is a *parallel concatenation* of its code fragments. By parallel concatenation, we simply mean adjoining several pieces of a codeword to form the full codeword.

Figures 2(a) and 2(b) show each code fragment preceded by a permutation, π_0, π_1, or π_2. This seemingly needless complication is introduced here for symmetry and to facilitate random coding arguments presented later. For these two examples, the only permutation relevant to the construction of the overall turbo code is the *relative* permutation $\pi_1^{-1}\pi_2$ or $\pi_2^{-1}\pi_1$ between the inputs $\mathbf{u}_1$ and $\mathbf{u}_2$. In practice, the permutations π_0 and π_1 are identities (i.e., no permutation). Each of the encoders in Figs. 2(a) and 2(b) is used to generate a $(3(N+2), N)$ block code, where N is the information block length. Following the information bits, an additional 2 "tail bits" are appended in order to drive the encoder to the all-zero state at the end of the block. The termination method described in [3] can be used.

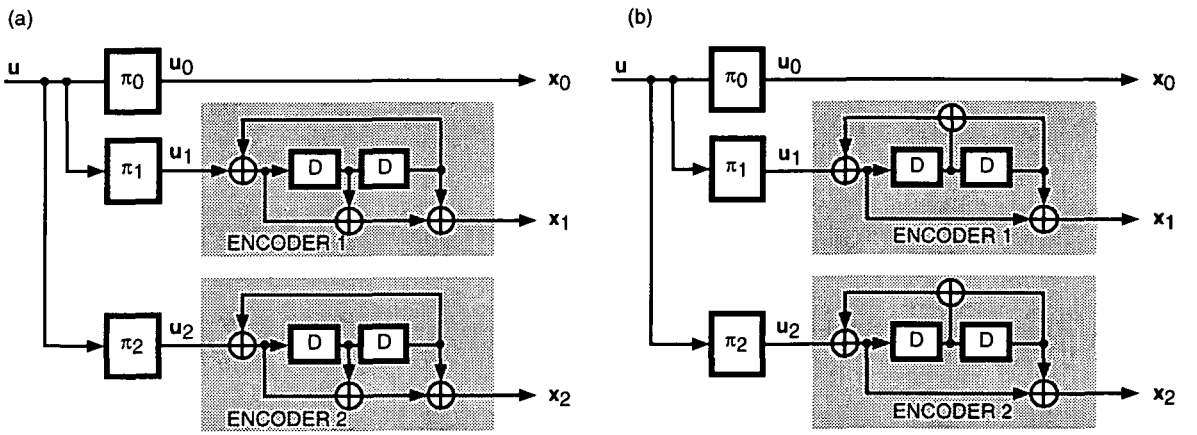

Fig. 2. Two examples of turbo encoders: (a) the (1, 7/5, 7/5) code and (b) the (1, 5/7, 5/7) code.

B. State Diagrams

Each of the two exemplary turbo encoders is a four-state device. Figures 3(a) and 3(b) show the state transition diagrams for the nontrivial code fragments, (g_a/g_b) and (g_b/g_a), respectively. In this diagram, each transition between states is labeled by the input information bit and the corresponding output encoded bit. It is convenient to replace each edge label in Figs. 3(a) and 3(b) with a monomial $L^l I^i D^d$, where l is always equal to 1, and i and d are either 0 or 1, depending on whether the corresponding input and output bits are 0 or 1, respectively. Then the information in the state transition diagrams can be summarized by state transition matrices $\mathbf{A}(L, I, D)$, where

$$\mathbf{A}_{7/5}(L, I, D) = \begin{pmatrix} L & LID & 0 & 0 \\ 0 & 0 & LD & LI \\ LID & L & 0 & 0 \\ 0 & 0 & LI & LD \end{pmatrix} \tag{1}$$

for the (7/5) code fragment, and

$$\mathbf{A}_{5/7}(L, I, D) = \begin{pmatrix} L & LID & 0 & 0 \\ 0 & 0 & LI & LD \\ LID & L & 0 & 0 \\ 0 & 0 & LD & LI \end{pmatrix} \tag{2}$$

for the (5/7) code fragment.

C. Input–Output Weight Enumerator

For a given code fragment, defined by a state diagram with 2^m states as in Section II.B, denote by $t(l, i, d)$ the number of paths of length l, input weight i, and output weight d, starting and ending in state 0^m, with the proviso that the last m edges in the path are "termination" edges and have neither length nor input weight. The corresponding transfer function (generating function) is defined by

$$T(L, I, D) = \sum_{l \geq 0} \sum_{i \geq 0} \sum_{d \geq 0} L^l I^i D^d t(l, i, d) \tag{3}$$

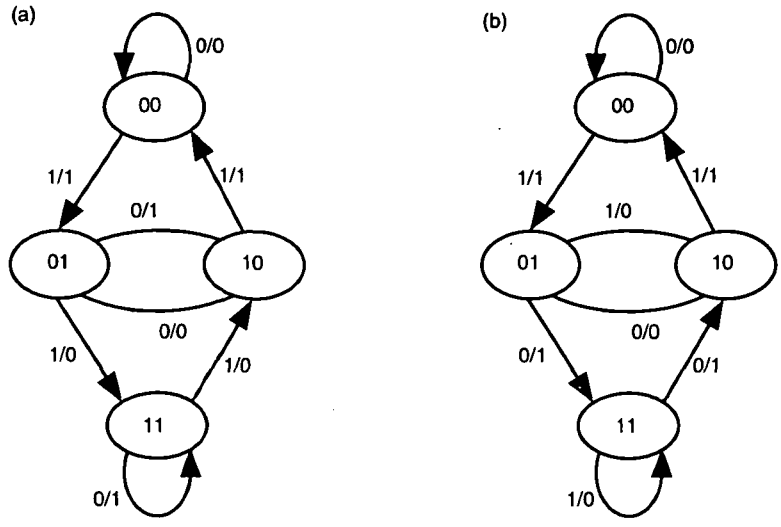

Fig. 3. State diagrams of two code fragments: (a) the (7/5) code fragment and (b) the (5/7) code fragment.

Using the method described in Section 4.7 of [8], we find that $T(L, I, D)$ is the $(0^m, 0^m)$ entry in the matrix

$$(\mathbf{I} + \mathbf{A}(L, I, D) + \mathbf{A}(L, I, D)^2 + \mathbf{A}(L, I, D)^3 + \cdots)\mathbf{A}(1, 1, D)^m \tag{4}$$

The factor $\mathbf{A}(1, 1, D)^m$ takes care of the termination edges. Since $\mathbf{I} + \mathbf{A} + \mathbf{A}^2 + \mathbf{A}^3 + \cdots = (\mathbf{I} - \mathbf{A})^{-1}$, it follows from Eq. (4) that

$$T(L, I, D) = \left[(\mathbf{I} - \mathbf{A}(L, I, D))^{-1} \cdot \mathbf{A}(1, 1, D)^m\right]_{0^m, 0^m} \tag{5}$$

Using Eq. (5) (approximately, by omitting the termination factor $\mathbf{A}(1, 1, D)^m$), we find that the transfer functions for the (7/5) and (5/7) code fragments are

$$T_{7/5}(L, I, D) \approx \frac{1 - LD - L^2D + L^3\left(D^2 - I^2\right)}{1 - L(1 + D) + L^3\left(D + D^2 - I^2 - I^2D^3\right) - L^4\left(D^2 - I^2 - I^2D^4 + I^4D^2\right)} \tag{6}$$

and

$$T_{5/7}(L, I, D) \approx \frac{1 - LI - L^2I - L^3\left(D^2 - I^2\right)}{1 - L(1 + I) - L^3\left(D^2 - I - I^2 + I^3D^2\right) + L^4\left(D^2 - I^2 - I^2D^4 + I^4D^2\right)} \tag{7}$$

respectively. Note that, in this approximation, $T_{7/5}(L, I, D) = T_{5/7}(L, D, I)$, i.e., the roles of input weight i and output weight d are reversed for the two code fragments.

If we multiply both sides of Eq. (6) by the denominator of the right-hand side, and take the coefficient of $t(l, i, d)$ of both sides of the resulting equation, we obtain the following recursion determining $t_{7/5}(l, i, d)$ for the (7/5) code fragment, for $l \geq 0, i \geq 0, d \geq 0$:

$$
\begin{aligned}
t(l,i,d) &= t(l-1,i,d-1) + t(l-1,i,d) \\
&\quad + t(l-3,i-2,d-3) + t(l-3,i-2,d) - t(l-3,i,d-2) - t(l-3,i,d-1) \\
&\quad + t(l-4,i-4,d-2) - t(l-4,i-2,d-4) - t(l-4,i-2,d) + t(l-4,i,d-2) \\
&\quad + \delta(l,i,d) - \delta(l-1,i,d-1) - \delta(l-2,i,d-1) + \delta(l-3,i,d-2) - \delta(l-3,i-2,d)
\end{aligned}
$$

where $\delta(l,i,d) = 1$ if $l = i = d = 0$ and $\delta(l,i,d) = 0$ otherwise, and with the initial conditions that $t(l,i,d) = 0$ if any index is negative.

Similarly, $t_{5/7}(l,i,d)$ for the (5/7) code fragment can be evaluated by the recursion

$$
\begin{aligned}
t(l,i,d) &= t(l-1,i-1,d) + t(l-1,i,d) \\
&\quad + t(l-3,i-3,d-2) - t(l-3,i-2,d) - t(l-3,i-1,d) + t(l-3,i,d-2) \\
&\quad - t(l-4,i-4,d-2) + t(l-4,i-2,d-4) + t(l-4,i-2,d) - t(l-4,i,d-2) \\
&\quad + \delta(l,i,d) - \delta(l-1,i-1,d) - \delta(l-2,i-1,d) - \delta(l-3,i,d-2) + \delta(l-3,i-2,d)
\end{aligned}
$$

again with the understanding that $t(l,i,d) = 0$ if any index is negative.

III. Union Bounds on Word and Bit-Error Probabilities

In this section, we use the input–output weight enumerators $t(l,i,d)$ for the various code fragments to obtain a union bound on the probabilities of word error and bit error, assuming an additive white Gaussian noise channel with channel symbol signal-to-noise ratio E_s/N_0.

A. Derivation of the Bound

As depicted in Figs. 2(a) and 2(b), the turbo code is constructed as a parallel concatenation of its three code fragments, each preceded by a random permutation of the input information bits $\mathbf{u}$. The randomly chosen permutations π_0, π_1, and π_2 transform the input sequence $\mathbf{u}$ into three permuted sequences $\mathbf{u}_0$, $\mathbf{u}_1$, and $\mathbf{u}_2$, each having the same Hamming weight as the original sequence $\mathbf{u}$. Since the turbo code has block length N, there are $t_{7/5}(N,i,d)$ codeword fragments of input weight i and output weight d from the two (7/5) code fragments of the (1, 7/5, 7/5) code, and $t_{5/7}(N,i,d)$ codeword fragments of input weight i and output weight d from the two (5/7) code fragments of the (1, 5/7, 5/7) code.

Denote by $p(d|i)$ the conditional probability of producing a codeword fragment of weight d given a randomly selected input sequence of weight i. Then[1]

$$
p(d|i) = \frac{t(N,i,d)}{\sum_{d'} t(N,i,d')} = \frac{t(N,i,d)}{\binom{N}{i}}
$$

[1] The summation $\sum_d t(N,i,d)$ equals the total number of codewords of information weight i, $\binom{N}{i}$. However, this is not exact if $t(N,i,d)$ is computed according to the approximation that does not account for the termination edges.

The conditional probability distributions $p_{7/5}(d|i)$ and $p_{5/7}(d|i)$ are plotted in Figs. 4(a) and 4(b) for the two code fragments (7/5) and (5/7), for block length $N = 100$. For the uncoded fragment, $p_1(d|i) = \delta(i, d)$.

Note from Figs. 4(a) and 4(b) that the (7/5) code fragment admits only even input weights i, while the (5/7) code fragment has only even output weights d. The vertical scale of Fig. 4(b) is twice that of Fig. 4(a) to reflect the concentration of probability into even-only output weights for the (5/7) fragment. The figures also show for reference a binomial probability distribution for 100 trials with probability 1/2 (in the case of Fig. 4(b), the reference "binomial" distribution is twice the binomial probability for even weights only).

In both Figs. 4(a) and 4(b), the conditional probability distribution $p(d|i)$ approaches the binomial reference for moderate to large values of input weight i, indicating a more or less random distribution of output weights d. In contrast, the skewed distributions for low values of i are what differentiate the two code fragments from each other and the corresponding overall turbo codes from purely random codes. In particular, notice how the $p(d|i)$ distribution for input weight $i = 2$ is more skewed toward lower output weights d for the (7/5) fragment than for the (5/7) fragment.

If the permutations are selected randomly and independently, the probability $\tilde{p}(d_0, d_1, d_2|i)$ that any input sequence $\mathbf{u}$ of weight i will be mapped into codeword fragments of weights d_0, d_1, and d_2 is

$$\tilde{p}(d_0, d_1, d_2|i) = p_1(d_0|i)p_{7/5}(d_1|i)p_{7/5}(d_2|i)$$

for the (1, 7/5, 7/5) code and

$$\tilde{p}(d_0, d_1, d_2|i) = p_1(d_0|i)p_{5/7}(d_1|i)p_{5/7}(d_2|i)$$

for the (1, 5/7, 5/7) code. The conditional probability that a maximum-likelihood decoder will prefer a particular codeword of total weight $d = d_0 + d_1 + d_2$ to the all-zero codeword is $Q(\sqrt{2dE_s/N_0})$, where $Q(\cdot)$ is the complementary unit variance Gaussian distribution function. Thus, the codeword error probability P_w is upper bounded as follows:

$$P_w = \sum_{i=1}^{N} \text{Prob [error event of weight } i] \leq \sum_{i=1}^{N} \binom{N}{i} \mathrm{E}_{d|i} \left\{ Q\left(\sqrt{\frac{2dE_s}{N_0}} \right) \right\} \qquad (8)$$

where the conditional expectation $\mathrm{E}_{d|i}\{\cdot\}$ is over the probability distribution $\tilde{p}(d_0, d_1, d_2|i)$. Similarly, the information bit-error probability P_b is upper bounded by

$$P_b = \sum_{i=1}^{N} \frac{i}{N} \text{Prob [error event of weight } i] \leq \sum_{i=1}^{N} \frac{i}{N} \binom{N}{i} \mathrm{E}_{d|i} \left\{ Q\left(\sqrt{\frac{2dE_s}{N_0}} \right) \right\} \qquad (9)$$

The error probabilities P_w and P_b bounded in Eqs. (8) and (9) are averages (over randomly chosen permutations) of the word and bit error probabilities achieved by any particular turbo code with specified permutations π_0, π_1, π_2.

B. Evaluation of the Bound for the Examples

Figures 5(a) and 5(b) show the bounds on P_b for the (1, 7/5, 7/5) code and the (1, 5/7, 5/7) code for various block lengths N. Note that the transition from a well-behaved, useful, low P_b bound into

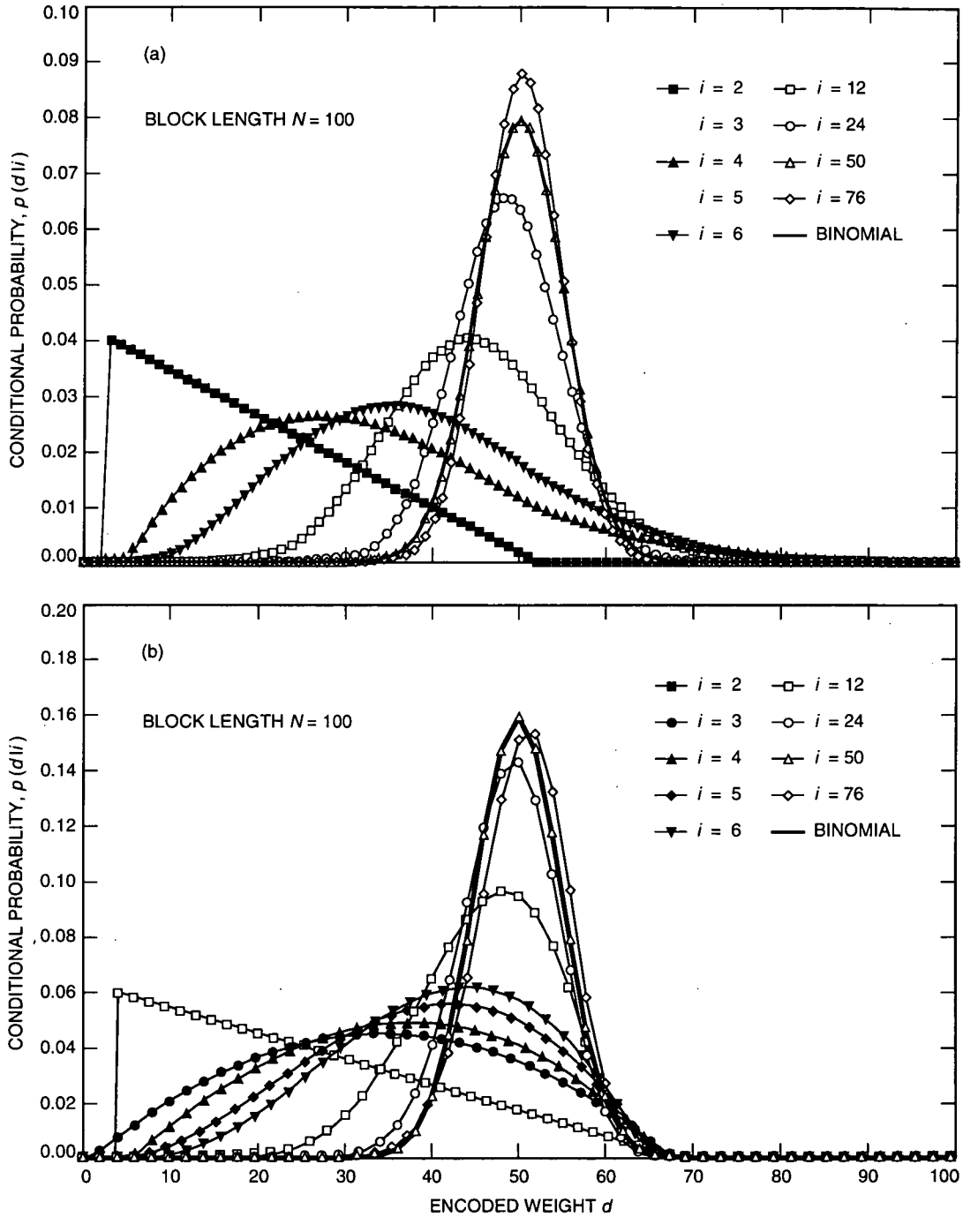

Fig. 4. Conditional probability distribution $p(d \mid i)$ for output weight d given input weight i: (a) code fragment (7/5) and (b) code fragment (5/7).

a diverged, useless bound greater than 1 occurs very abruptly if the block length N (i.e., permutation length) is large. The abrupt transition occurs roughly when the information bit signal-to-noise ratio E_b/N_0 drops below the threshold determined by the computational cutoff rate R_0, i.e., when $E_s/N_0 = rE_b/N_0 < -\ln(2^{1-r}-1)$ for a code with rate r [9]. This behavior mimics that of similar bounds applied to totally random codes, which turbo codes resemble.

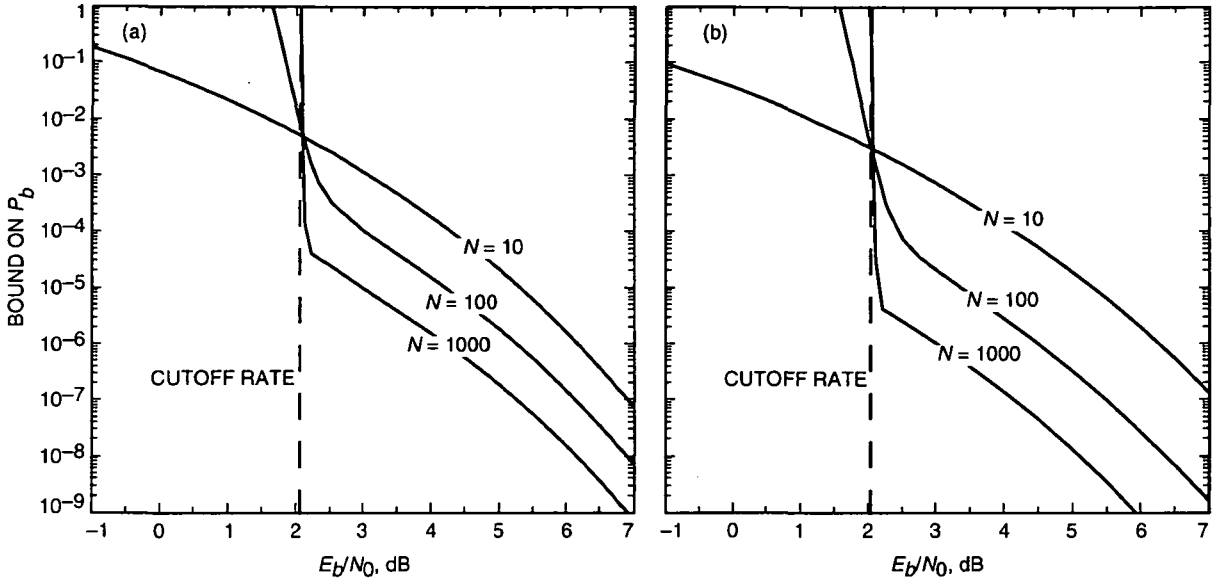

Fig. 5. Bounds on P_b for various block lengths N: (a) the (1, 7/5, 7/5) code and (b) the (1, 5/7, 5/7) code.

In computing these bounds, we discovered and overcame two distinct types of pitfalls. First, there is an inherent numerical precision problem that we have solved for block lengths up to about 1000. Second, for low E_b/N_0, there is a sinister "false convergence" region where the bound seems to have converged to an unchanging value, but after remaining at this constant value for many terms, it suddenly diverges to a useless probability bound greater than 1. Figure 6 illustrates this false convergence behavior. For this figure, the summation in the union bound expression in Eq. (9) is truncated after i^+ terms on the assumption that higher-order terms in i will not contribute to the sum. For large block lengths (e.g., $N = 400$ or $N = 1000$ in the figure), this assumption seems to be validated because the cumulative summation becomes almost totally flat after a small fraction i^+/N of all the terms. However, when i^+/N reaches about 0.2, the cumulative summation starts increasing rapidly before saturating at a much higher level than the first plateau. This illustration of the false convergence behavior is for $E_b/N_0 =$ 2.00 dB, which is just barely below the R_0 threshold of 2.03 dB for rate 1/3 codes. The effect is even more dramatic if E_b/N_0 is decreased further. On the other hand, when E_b/N_0 is above the R_0 threshold, false convergence is not a problem. Figure 7 shows how quickly and truly the summation converges when $E_b/N_0 = 2.50$ dB. The curves in Fig. 7 are also plotted versus the fraction i^+/N in order to show the full range of i^+ for all values of N simultaneously. This demonstrates that the second plateau observed in Fig. 6 is absent at the higher value of E_b/N_0. However, it is apparent from Fig. 7 that only a handful of terms (roughly $i \leq 10$) are needed for convergence in this case, and this is almost independent of the value of N.

When we first attempted to evaluate these bounds, we were fooled by the false convergence region and were computing error rates low enough to contradict Shannon's limit! After we learned how to properly evaluate the bounds, we found that other researchers (e.g., [2]) were unaware of the intricacies of the divergence and had computed the bounds inaccurately at low E_b/N_0. However, in the next section, we will see that this divergence is an artifact of the bound; the error rate from an actual turbo decoder does not diverge at the R_0 threshold.

C. Comparison of the Bounds With Turbo Decoder Simulation Results

Figure 8 compares the computed bounds for the (1, 7/5, 7/5) and (1, 5/7, 5/7) turbo codes with simulated turbo decoder bit-error rates. We observe that, above the R_0 threshold of 2.03 dB, the simulated turbo decoder bit-error rate closely matches the error rate predicted by the bound. Below this threshold,

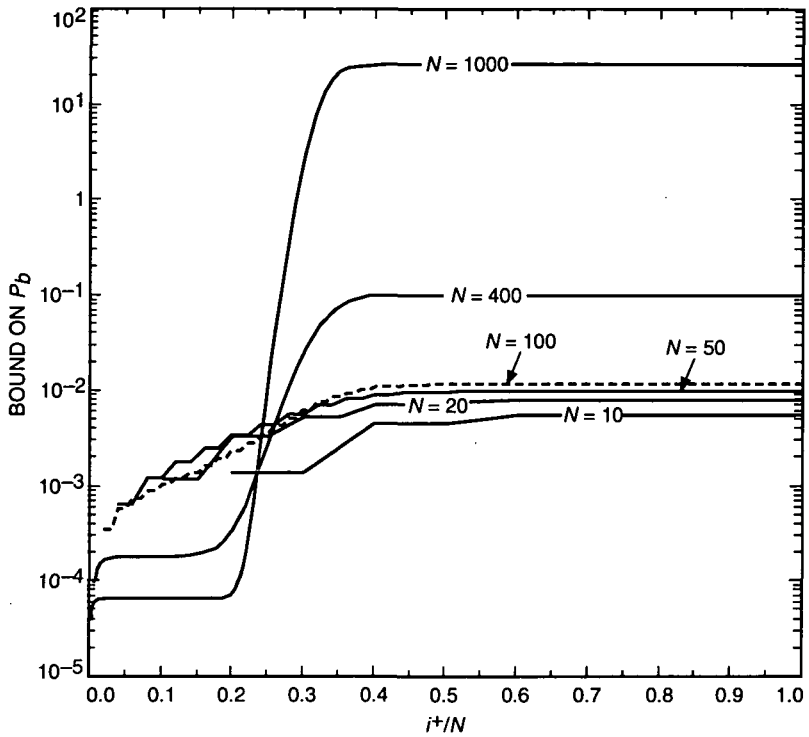

Fig. 6. False convergence behavior at E_b/N_0 = 2.00 dB (just below the R_0 threshold of 2.03 dB).

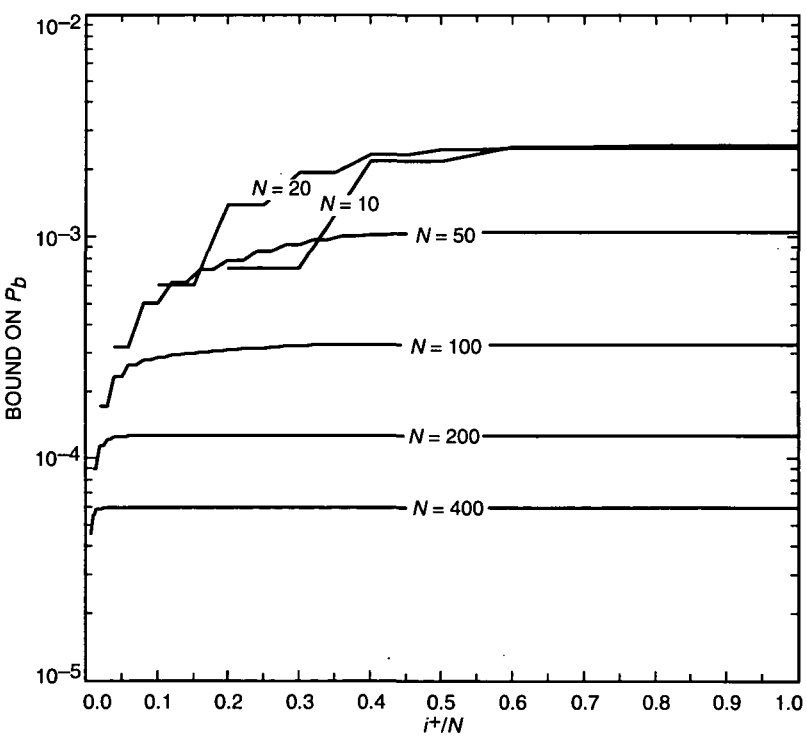

Fig. 7. Convergence at E_b/N_0 = 2.50 dB (comfortably above the R_0 threshold of 2.03 dB).

the turbo decoder experiences its own region of "divergence" wherein its performance deteriorates rapidly because its iterative decoding algorithm frequently fails to converge. However, this "divergence" is far less steep than that experienced by the bound, and it occurs well below the R_0 threshold, allowing turbo decoders to operate in the region between the limit determined by channel capacity and that determined by R_0.

We observe in Fig. 8 the relative performance of the two exemplary turbo codes and the relative performance of the same codes with different block lengths. The (1, 5/7, 5/7) code is clearly superior to the (1, 7/5, 7/5) code in the region where the bound accurately predicts turbo decoder performance. This is consistent with the heuristic arguments presented in [5]. However, when E_b/N_0 is low enough that the decoder's iterative algorithm stops being effective, the two codes perform similarly, and the heuristic arguments do not apply. By comparing the results for block lengths $N = 100$ and $N = 1000$, we also see how the performance of turbo decoders at high E_b/N_0 is dramatically improved by increasing the length of the random permutation.

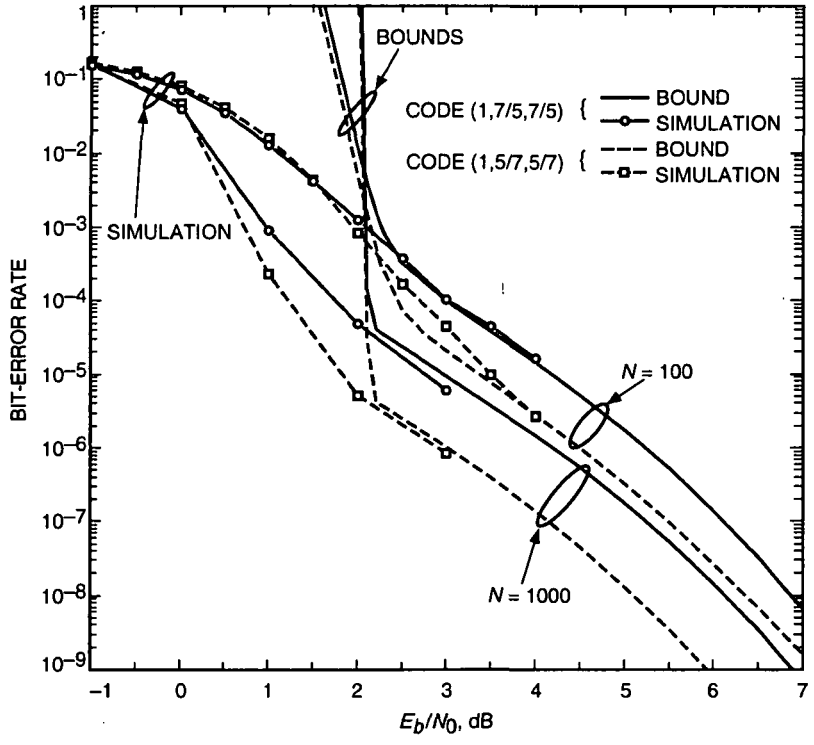

Fig. 8. Transfer function bound versus simulated bit-error rates for the (1, 7/5, 7/5) and (1, 5/7, 5/7) codes.

D. The Turbo Decoder Error Floor

Unfortunately, the region where turbo codes have offered astounding performance is below the computational cutoff rate threshold, so at first glance the bounds appear to be of dubious utility. Nevertheless, our work has some immediate applications and suggests some refinements. For E_b/N_0 above the computational cutoff rate threshold, we believe that the bound is not only meaningful but that it essentially tells the whole story, i.e., the bit-error rate predicted by the bound is accurately achieved both by a maximum-likelihood decoder and by a turbo decoder. This is demonstrated by the confluence of the simulation and bound performance curves in Fig. 8 at high E_b/N_0. In this region, evaluation of the bound requires only a few terms in the summation, and its behavior is predictable from the more heuristic analysis about the relationships of weight distributions, permutations, and the number of codes reported

in [5]. This low-slope region of the bound establishes the elusive "error floor" that several researchers, including ourselves, have noted but have had difficulty establishing clearly via simulations, because the error floor for large block-length turbo codes is too low to simulate accurately.

The error floor is actually not flat, but instead is a low-slope region of the performance curve, wherein the turbo decoder's error rate decreases very slowly with increasing E_b/N_0. The slope of the error floor is limited by the weakness of the turbo code's simple constituent codes, but the position of the error floor can be lowered by increasing the permutation length N. Empirically, the error floor appears to be extrapolatable backwards through the computational cutoff rate barrier to some (as yet undetermined) lower E_b/N_0 where it finally stops being an accurate predictor of turbo code performance. Furthermore, the extrapolated error floor in this region appears to be computable as the "false convergence" plateau we noted earlier. Thus, we surmise that, for some values of E_b/N_0 below the R_0 threshold, the false convergence region of the bound actually corresponds to a true convergence region for predicting turbo decoder performance: even though the bound diverges, the portion of the bound based only on low-weight input sequences is still a useful predictor of performance.

IV. Results for Other Codes

Our results thus far have been developed with respect to the two exemplary turbo codes for concreteness. The theory obviously generalizes to arbitrary turbo codes constructed as parallel concatenations of code fragments. The enumerators $t(l, i, d)$ must be evaluated for each code fragment, and the union bound is obtained as a summation over products of independent enumerators. Some results for rate 1/3 and 1/4 codes with constraint lengths 3 and 4 are shown in Fig. 9 comparing the bounds with the Shannon limits for these rates.

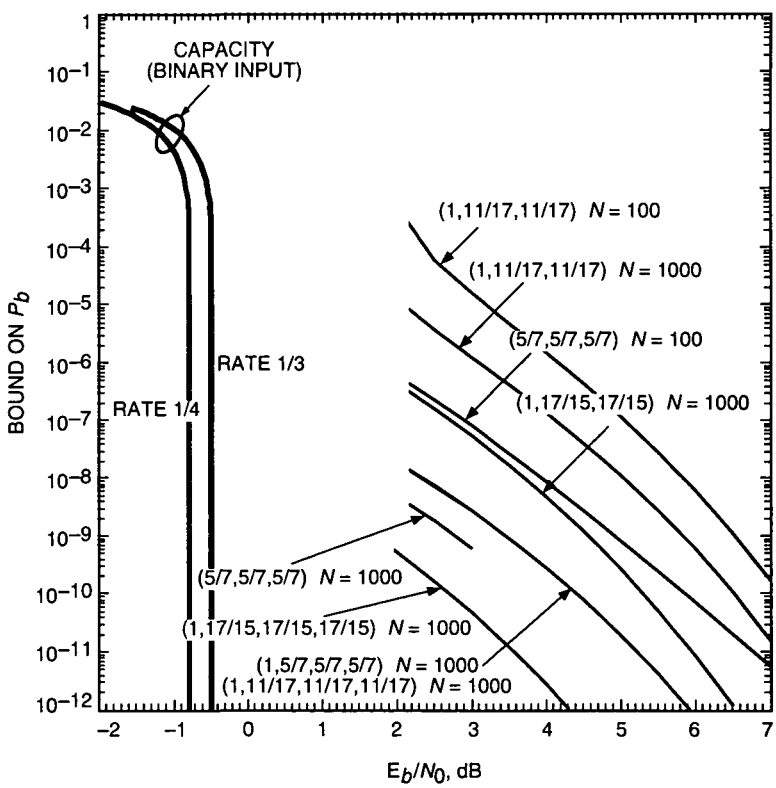

Fig. 9. Transfer function bounds for other codes.

V. Conclusion and Further Work

One of the most important lessons we have learned is that the divergence properties of these bounds for turbo codes appear to be the same as those of similar bounds applied to random codes. This observation leads us to try to adapt known bounding techniques that diverge at capacity, rather than the computational cutoff rate, when applied to random codes. Foremost among the candidate techniques we propose to evaluate are the Gallager bound and the bound based on the "code geometry function" [6]. Preliminary work is encouraging, but is currently limited by extreme numerical computation barriers when the block length is large. However, we are developing analytical approximations that are valid asymptotically as the block length gets larger.

References

[1] S. Benedetto and G. Montorsi, "Average Performance of Parallel Concatenated Block Codes," *Electronics Letters*, vol. 31, no. 3, pp. 156–158, February 2, 1995.

[2] S. Benedetto and G. Montorsi, "Performance Evaluation of Turbo-Codes," *Electronics Letters*, vol. 31, no. 3, pp. 163–165, February 2, 1995.

[3] D. Divsalar and F. Pollara, "Turbo Codes for Deep-Space Communications," *The Telecommunications and Data Acquisition Progress Report 42-120, October–December 1994*, Jet Propulsion Laboratory, Pasadena, California, pp. 29–39, February 15, 1995.

[4] D. Divsalar and F. Pollara, "Multiple Turbo Codes for Deep-Space Communications," *The Telecommunications and Data Acquisition Progress Report 42-121, January–March 1995*, Jet Propulsion Laboratory, Pasadena, California, pp. 66–77, May 15, 1995.

[5] S. Dolinar and D. Divsalar, "Weight Distributions for Turbo Codes Using Random and Nonrandom Permutations," *The Telecommunications and Data Acquisition Progress Report 42-122, April–June 1995*, Jet Propulsion Laboratory, Pasadena, California, pp. 56–65, August 15, 1995.

[6] S. Dolinar, L. Ekroot, and F. Pollara, "Improvements on Probability of Error Bounds for Block Codes on the Gaussian Channel," *Proceedings 1994 International Symposium on Information Theory*, Trondheim, Norway, p. 243, June 27–July 1, 1994.

[7] A. J. Viterbi and J. K. Omura, *Principles of Digital Communication and Coding*, New York: McGraw-Hill, 1979.

[8] R. P. Stanley, *Enumerative Combinatorics*, Monterey, California: Wadsworth & Brooks/Cole, 1986.

[9] S. Benedetto, E. Biglieri, and V. Castellani, *Digital Transmission Theory*, Englewood Cliffs, New Jersey: Prentice-Hall, Inc., 1987.