

Turbo Codes and Space Communications

Sam Dolinar, Dariush Divsalar, and Fabrizio Pollara¹

*Communications Systems and Research Section
Jet Propulsion Laboratory, California Institute of Technology
Pasadena, CA 91109
FAX: (818) 354-6825. E-mail: sam@shannon.jpl.nasa.gov*

ABSTRACT

This paper describes a class of parallel and serial concatenated codes called turbo codes, which achieve near-Shannon-limit error correction performance with low decoding complexity. The exceptional performance of these codes is explained, together with the basic structure of the encoders and decoders. These new codes are currently under consideration by CCSDS to define a new coding standard for space telemetry.

I. INTRODUCTION

It is just over a quarter century since the launch of NASA's Pioneer 9 spacecraft on the first deep-space mission that relied on error-correcting codes to enhance scientific data return. The success of that first coding system spawned a continuous sequence of improvements in channel coding. Concatenated coding systems comprising an inner convolutional code and an outer Reed-Solomon code have been successfully used in deep-space missions for over a decade and have been incorporated in CCSDS standards for space telemetry systems. However, in order to obtain the highest coding gains, the decoder at the ground station is such a complex device that further improvements within this class of codes are extremely expensive.

Progress in the development of new, more powerful codes has been hampered because good codes are hard to find, hard to analyze, and hard to decode. Coding theorists have traditionally developed codes with a lot of structure, which leads to feasible decoders. But coding theory suggests that codes chosen "at random" should perform very well if their block size is large enough. The challenge to find practical decoders for "nearly random," large codes had not been seriously considered until recently.

This paper discusses a new answer to these problems: a class of parallel and serial concatenated codes, often called "turbo" codes, which achieve near-Shannon-limit error correction performance with low decoding complexity. The exceptional performance of these codes is explained, together with the basic structure of the encoders and decoders. These new codes are currently under consideration by CCSDS to define a new coding standard for space telemetry.

II. TURBO CODES

In 1993 a new class of concatenated codes called turbo codes was introduced [1]. Turbo codes outperform the most powerful codes known to date, but more importantly they are much simpler to decode.

A. Encoder and Decoder Structures

A turbo encoder is a combination of two simple recursive convolutional encoders, each using a small number of states. For a block of k information bits, each constituent code generates a set of parity bits. The turbo code consists of the information bits and both sets of parity, as shown in Fig. 1.

The key innovation is an interleaver P , which permutes the original k information bits before encoding the second code. If the interleaver is well-chosen, information blocks that correspond to error-prone codewords in one code will correspond to error-resistant codewords in the other code. The resulting code achieves performance similar to that of Shannon's random codes. However, random codes approach optimum performance only at the price of a prohibitively complex decoder.

¹This research was carried out at the Jet Propulsion Laboratory (JPL), California Institute of Technology, under a contract with the National Aeronautics and Space Administration (NASA).

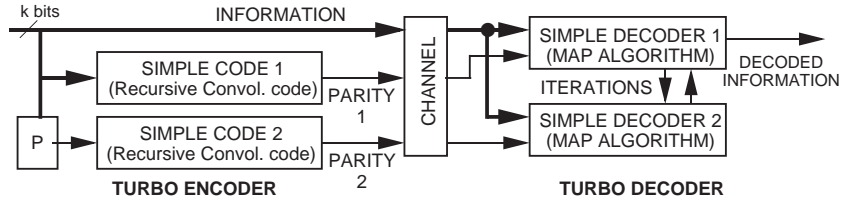


Fig. 1. Structure of turbo encoder and decoder.

Turbo decoding uses two *simple* decoders individually matched to the simple constituent codes. Each decoder sends likelihood estimates of the decoded bits to the other decoder, and uses the corresponding estimates from the other decoder as *a priori* likelihoods. The constituent decoders use the “MAP” (maximum *a posteriori*) bitwise decoding algorithm, which requires the same number of states as the well-known Viterbi algorithm. The turbo decoder iterates between the outputs of the two decoders until reaching satisfactory convergence. The final output is a hard-quantized version of the likelihood estimates of either of the decoders.

B. Performance Simulations

Simulations by many researchers (e.g., see [3]) have shown that turbo codes can achieve small error probabilities using a signal-to-noise ratio (SNR) just slightly higher than the capacity limit, when the turbo code block is very large (10^4 information bits or more). Fig. 2 shows the simulated performance of a family of turbo codes of rates $r= 1/2, 1/3, 1/4$ and $1/6$, constructed for an information block length of 10200 bits. For these results, the decoder made its final decoding decisions after 10 iterations.

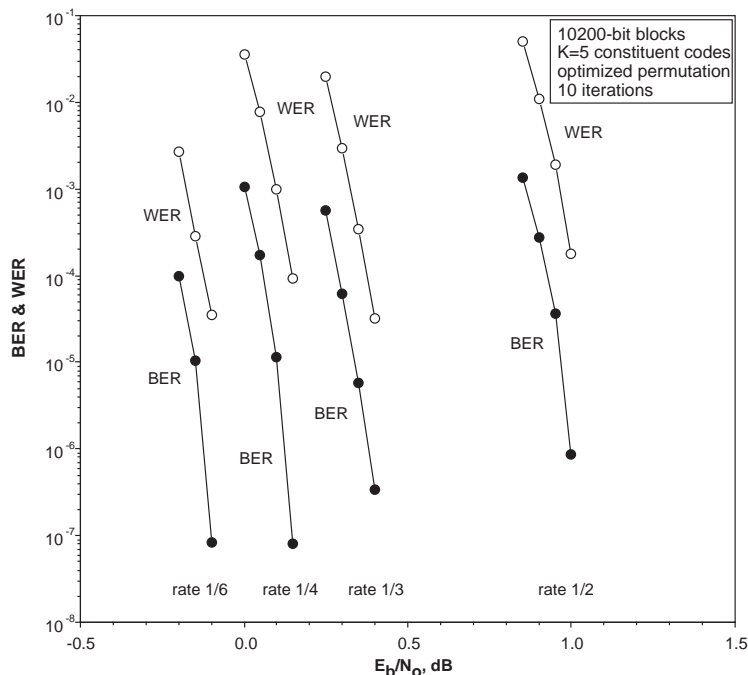


Fig. 2. Performance curves for various turbo codes.

The turbo codes in Fig. 2 are systematic parallel concatenated codes with two recursive convolutional components. The backward connection vector for both component codes is 10011. The forward connection vector for both component codes and rates 1/2 and 1/3 is 11011. The forward connection vectors for rate 1/4 are 10101, 11111 for the first component code, and 11011 for the second component. The forward connection vectors for rate 1/6 are 11111, 11101, 10111 for the first component code, and 11011,

11111 for the second component. Puncturing of every other symbol from each component is necessary for rate 1/2. No puncturing is done for rates 1/3, 1/4, and 1/6. These code rates and component codes are the ones currently recommended for standardization by CCSDS.

To achieve a bit error rate (BER) of 10^{-6} , threshold bit-SNRs (E_b/N_0) of approximately -0.1 dB, $+0.15$ dB, $+0.4$ dB, and $+1.0$ dB, are required by the turbo codes of rates 1/6, 1/4, 1/3, and 1/2, respectively. These same threshold bit-SNRs achieve a codeword error rate (WER) of approximately 10^{-4} . Turbo codes gain a significant performance improvement over the traditional Reed-Solomon and convolutional concatenated codes currently used by JPL. For example, to achieve an overall BER of 10^{-6} with a block length of 8920 bits (depth-5 interleaving of Reed-Solomon codewords), the required bit-SNRs are approximately 0.8 dB, 1.0 dB, and 2.6 dB for JPL's codes consisting of the (255,223) Reed-Solomon code concatenated with the (15,1/6) convolutional code, the (15,1/4) convolutional code, and the (7,1/2) convolutional code, respectively.

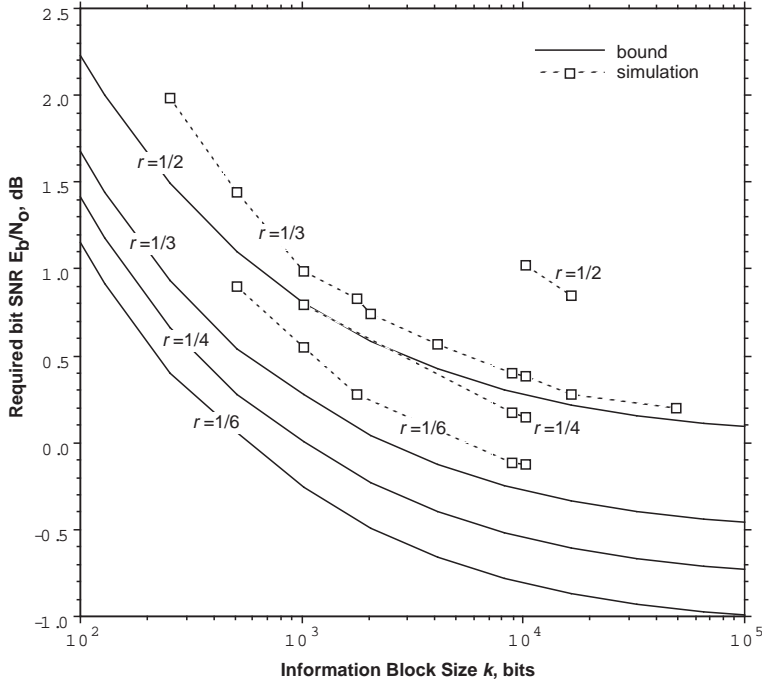


Fig. 3. Turbo code simulated performance as a function of information block size.

Figure 3 shows simulation results for a family of turbo codes of the same rates as those shown in Fig. 2, and built from the same constituent codes, but with varying block sizes. The current recommendation for CCSDS standardization includes block sizes of 1784, 3568, 7136, 8920, and 16384 bits. For each turbo code the graph shows the required E_b/N_0 to achieve a codeword error rate $P_w = 10^{-4}$. The graph also shows for comparison the theoretical lower bounds discussed in the next section.

C. Theoretical Lower Bound on Code Performance

Shannon's sphere packing bound [2] provides a lower limit to the error rate achievable by an arbitrary code of a given block size and code rate. The sphere packing bound is shown in Fig. 4, for the case of equal-energy signals applied to a continuous-input additive white Gaussian noise (AWGN) channel.

The sphere-packing bound would be reached with equality only if the code were a *perfect* code for the continuous-input AWGN channel. Although perfectness is generally an unattainable goal, it can be a good benchmark for code performance. We define the *imperfectness* of a given code as the difference between the code's required E_b/N_0 to attain a given P_w , and the minimum possible E_b/N_0 required to attain the same P_w , as implied by the sphere-packing bound for codes with the same block size k and code rate r . These differences, measured in dB, are shown in Fig. 5 for various codes, with $P_w = 10^{-4}$.

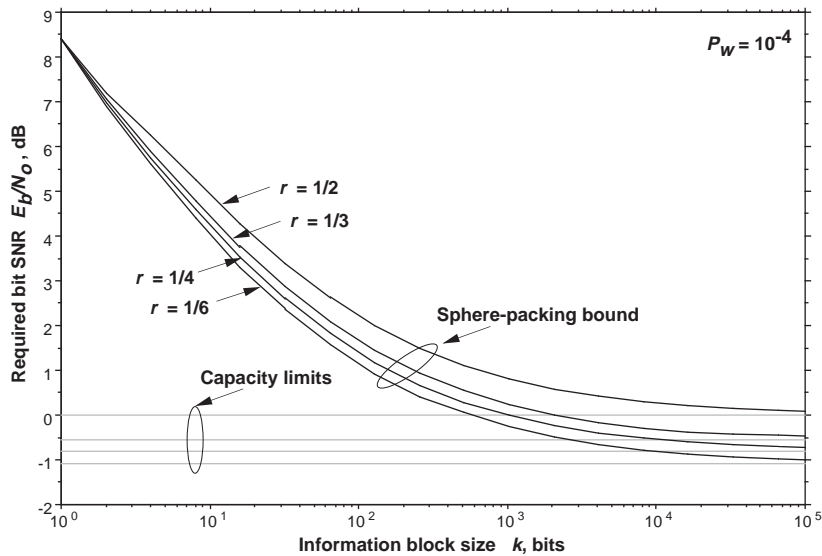


Fig. 4. Minimum required E_b/N_0 implied by the sphere-packing lower bound for codes with varying information block length k and rates $1/6, 1/4, 1/3, 1/2$, operating over a continuous-input AWGN channel and achieving $P_w = 10^{-4}$.

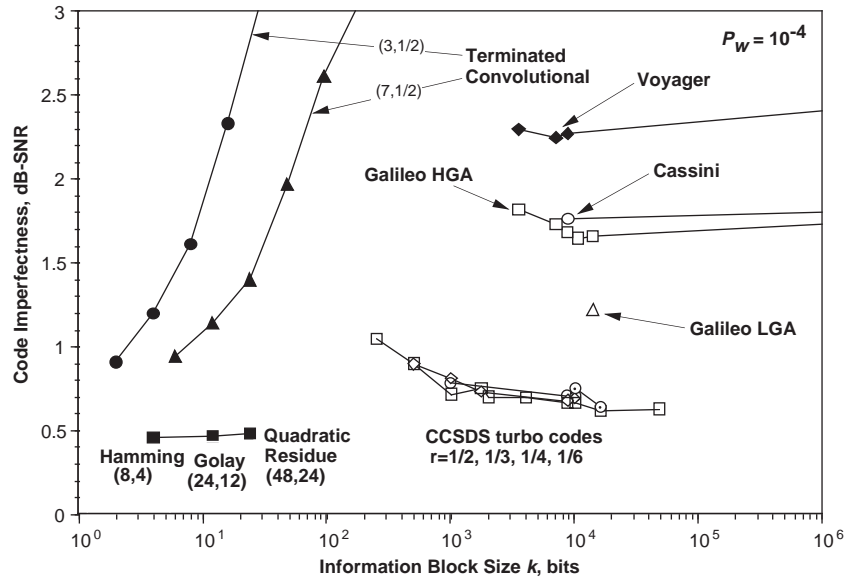


Fig. 5. Code imperfectness relative to the sphere-packing bound, for various codes.

Note that the imperfectness of the turbo codes in Fig. 5 is *uniformly* about 0.7 dB for all four code rates plotted, for all block sizes above 1000 bits. Fig. 5 also shows the imperfectness of various “families” of non-turbo codes for comparison: (1) a family of “best- d_{min} ” codes of rate $1/2$ (with maximum likelihood decoding); (2) two families of rate- $1/2$ convolutional codes terminated to various block lengths; and (3) four families of concatenated codes used in JPL’s deep space missions over the past two decades. The concatenated codes used inner convolutional codes of various rates and constraint lengths, and an outer code block consisting of interleaved (255,223) Reed-Solomon codewords. Each concatenated code family in the figure is obtained by keeping the component codes constant and varying the interleaving depth. One concatenated code (Galileo LGA) used a variable-redundancy (255, x) outer code and 4-stage iterative decoding; results are shown only for a fixed interleaving depth. Fig. 5 shows that JPL’s long codes historically marched toward perfectness in roughly half-dB steps from Voyager to Cassini to Galileo LGA to future missions that will use turbo codes.

D. Theoretical Upper Bound on Code Performance

Upper bounds on the error rate achievable by maximum likelihood decoding of a specific turbo code have been obtained by a union bounding technique [8]. These bounds are expressed in terms of the joint input and output weight distribution of the constituent codes, and they assume random, independently chosen permutations of the input data before each constituent encoding.

The upper bounds on turbo code performance accurately predict the actual turbo decoder's performance in the so-called "error floor" region above the "computational cutoff rate" threshold, below which the bounds diverge and are useless. The bounds prove that the turbo code's performance curve does not stay steep forever as does that of a convolutional/Reed-Solomon concatenated code. When it reaches the error floor, the curve flattens out considerably and from that point onward looks like the performance curve of a weak convolutional code. The error floor is not an absolute lower limit on achievable error rate, but it is a region where the slope of the error rate curve becomes dramatically lower.

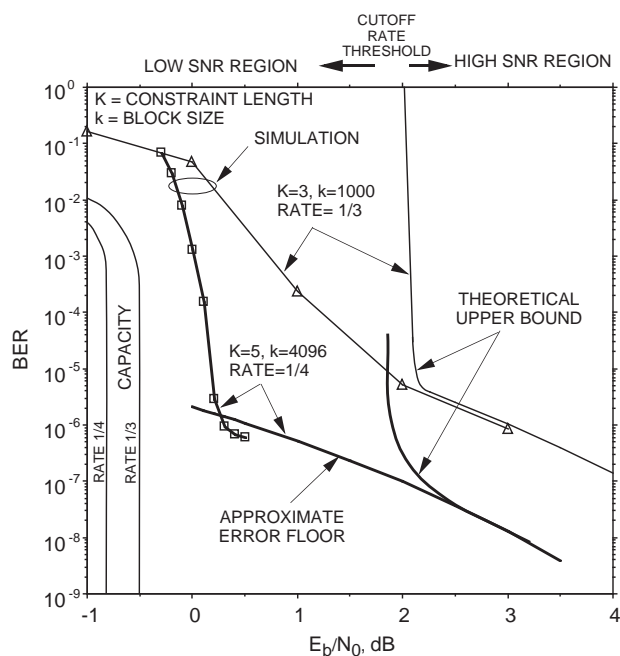


Fig. 6. Illustration of turbo code error floor

Figure 6 gives an illustration of the transition of turbo code performance curves from a steep "waterfall" region into a much flatter "error floor" region for two representative turbo codes. This figure shows the simulated performance of each code, compared with the theoretical upper bound and also with an extrapolation of the theoretical bound that gives a tighter approximation to actual performance on the error floor below the computational cutoff rate threshold. The original turbo codes developed by Berrou *et al* [1] had error floors starting at a BER of about 10^{-5} . By using predictors derived from the theoretical bounds, we have been able to design good turbo codes that lower the error floor to possibly insignificant levels (e.g., as low as 10^{-9} BER).

E. Code Design Considerations

Turbo codes give the system designer vast flexibility to choose any desirable combination of code parameters without sacrificing performance more than intrinsically necessary.

1. **Code Rate and Block Size** — The code rate of the currently recommended CCSDS turbo encoder is selectable from 1/2, 1/3, 1/4, or 1/6, and the block size is selectable from 1784, 3568, 7136, 8920, or 16384 bits. We have seen that a single family of turbo codes encompassing this range of rates and block sizes is *uniformly* nearly perfect (within 0.7 dB) with respect to the theoretical lower bounds shown in Fig. 4. Therefore, many system tradeoffs involving code rate and block size can

be adequately evaluated by reference to the universal theoretical bounds rather than by painstaking simulation of turbo codes of each rate or size.

2. **Constituent Codes** — Effective turbo codes can be constructed from a wide variety of constituents.
 - (a) *Number and Type of Constituent Codes*: Turbo codes with more than two constituent codes are feasible in principle, but have not been well studied (mainly because two-component turbo codes already perform so well). The best performing and best understood constituent codes discovered thus far are recursive convolutional codes, as recommended in the original turbo code paper by Berrou *et al* [1].
 - (b) *Constraint Length*: The currently recommended turbo code is formed from two recursive convolutional codes with constraint length $K = 5$. Higher constraint lengths are more complex to decode, and seem to offer negligible performance improvement. Constituent codes with constraint lengths less than 5 may sometimes be desirable to achieve higher decoding speeds with some sacrifice of performance.
 - (c) *Code Generator Polynomials*: Considerable theory has been developed to guide the choice of constituent code generator polynomials (forward and backward connection vectors). This theory is based on the theoretical upper bounds described in Section II-D. The error floor can be lowered the most if the divisor polynomial (backward connection vector) is primitive.
3. **Permutation** — The best understood permutations for turbo codes are completely random. However, pseudo-random sequences generated by simple randomizing algorithms (such as feedback shift register sequences) do not generate good turbo code permutations. Best performance is achieved by a manually optimized “S-random” permutation [9], but an optimized permutation must be stored in memory because it is infeasible to recompute for every codeword. Berrou [10] developed an algorithm for generating practically computable permutations that perform pretty well.

III. SERIAL CONCATENATION OF INTERLEAVED CODES

The original turbo codes were based on parallel concatenations of constituent codes. Later, similar codes were developed at JPL based on serial concatenation [4]. Serially concatenated codes offer the potential of somewhat better performance than parallel concatenated codes, including lower error floors. To date, there are fewer simulation results for serially concatenated codes, and this is the main reason they are not currently under consideration for CCSDS standardization.

A. Encoder and Decoder Structures

A serially concatenated convolutional code (SCCC) is a serial cascade of two simple convolutional encoders, each using a small number of states. The two constituent codes are the outer and inner codes. For a block of k^O information bits, the outer encoder generates a codeword of k^I bits. These k^I bits are permuted by an interleaver P , and the permuted bits are input to the inner encoder. The output bits of the inner encoder form the codewords of the SCCC, as shown in Fig. 7. For best performance, the inner convolutional code should be recursive, and the outer code can be either recursive or non-recursive..

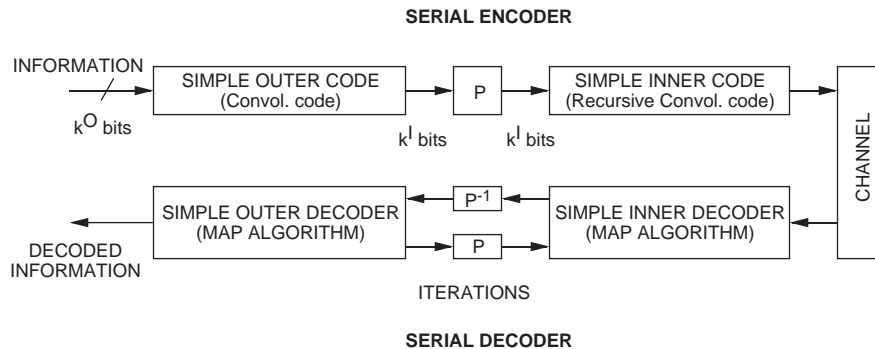


Fig. 7. Structure of encoder and decoder for serially concatenated convolutional codes.

Serial decoding uses two simple decoders individually matched to the simple constituent codes. Each decoder sends to the other decoder likelihood estimates of the k^I bits that form the common interface between the outer and inner codes. As with turbo decoders, the constituent decoders use the MAP (maximum *a posteriori*) bitwise decoding algorithm. The serial decoder iterates between the outputs of the two decoders until reaching satisfactory convergence. The final output is a hard-quantized version of the outer decoder's likelihood estimates of the k^O information bits.

B. Performance Simulations

Simulation results for SCCCs with an input block size of 16384 bits are shown in Fig. 8 for code rates 1/2, 1/3, 1/4, and 1/6. The constituent outer and inner codes are convolutional codes with constraint length $K = 3$. For example, the rate-1/4 serial code was constructed from a nonrecursive rate-1/2 convolutional outer code with forward connections 111, 101, and a recursive rate-1/2 convolutional inner code with backward connections 101 and forward connections 111. Fig. 8 also shows the performance of a rate-1/4 parallel concatenated convolutional code (PCCC) (turbo code) with constraint length $K = 3$ for comparison.

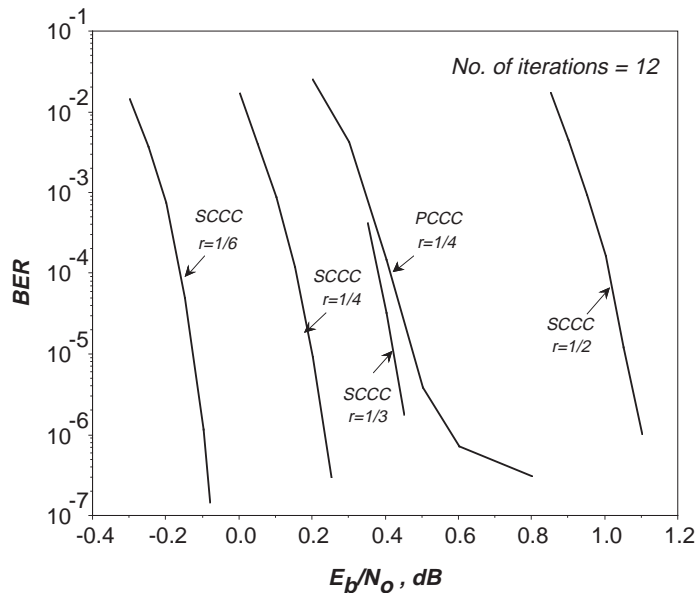


Fig. 8. Example of performance of serially concatenated convolutional codes with 16384-bit blocks and constraint length $K = 3$ constituent codes.

C. Code Performance Bounds

The lower bounds in Section II-C apply to all codes of a given rate and block size, and hence they are equally applicable to codes constructed by serial concatenation. Upper bounds for serially concatenated codes can be evaluated by a union-bounding procedure similar to that used in Section II-D. The upper bounds predict sharply lower error floors for serially concatenated codes than for turbo codes when the minimum distance of the outer code is greater than 2.

D. Additional Code Structures

Recent developments have produced additional code structures amenable to iterative decoding, beyond the simple parallel and serial structures considered here. These new codes include: extensions of parallel and serial concatenated codes to more than two component codes [3], [5]; hybrid and self-concatenated structures [6]; turbo trellis coded modulation for bandwidth efficient transmission [3]; turbo codes with small latency; and very low complexity turbo codes for high speed decoding.

IV. SUMMARY

Turbo codes represent a major paradigm shift in the approach to coding systems for deep space communications. They offer performance improvements of 1 dB or more compared to the codes currently used, and they are more than an order-of-magnitude easier to decode than the most complex of the current codes. Furthermore, they sustain their near-optimum performance over a wide range of fundamental code parameters (such as code rate and block size) that are important to system designers.

Future deep space coding systems will likely be based on families of turbo codes or serially concatenated convolutional codes, with adjustable parameters appropriate for different applications. Such integrated coding families will be an improvement over the present-day hodge-podge of different codes. They will provide better performance, reduce the complexity of decoding, and simplify system integration. To reach this goal, we must continue to study the basic principles governing turbo codes, and to assess the implications of turbo codes on system designs for space communications.

REFERENCES

- [1] C. Berrou, A. Glavieux, and P. Thitimajshima, "Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-Codes", *Proc. of ICC'93*, Geneva, May 1993, pp. 1064-1070.
- [2] C.E. Shannon, "Probability of error for optimal codes in a Gaussian channel", *Bell Syst. Tech. J.*, Vol. 38, pp. 611-656, 1959.
- [3] D. Divsalar and F. Pollara, "On the design of turbo codes," *JPL TDA Progress Report 42-123*, Nov. 15, 1995.
- [4] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Serial Concatenation of Interleaved Codes: Performance Analysis, Design, and Iterative Decoding," *IEEE Transactions on Information Theory*, Vol. 44, no. 3, May 1998.
- [5] S. Benedetto, D. Divsalar, G. Montorsi, and F. Pollara, "Analysis, Design, and Iterative Decoding of Double Serially Concatenated Codes with Interleavers", *IEEE Journal on Selected Areas in Communications*, Vol. 16, no. 2, Feb. 1998, pp. 231-44.
- [6] D. Divsalar, and F. Pollara, "Hybrid Concatenated Codes and Iterative Decoding", *JPL TDA Progress Report 42-130*, Aug. 15, 1997.
- [7] S. Dolinar, D. Divsalar, and F. Pollara, "Code Performance as a Function of Block Size", *JPL TDA Progress Report 42-133*, May 15, 1998.
- [8] D. Divsalar, S. Dolinar, R. J. McEliece, and F. Pollara, "Transfer Function Bounds on the Performance of Turbo Codes," *JPL TDA Progress Report 42-122*, August 15, 1995.
- [9] S. Dolinar, and D. Divsalar, "Weight Distributions for Turbo Codes using Random and Nonrandom Permutations", *JPL TDA Progress Report 42-122*, August 15, 1995.
- [10] C. Berrou, and A. Glavieux, "Near optimum error-correcting coding and decoding: Turbo codes," *IEEE Transactions on Communications*, vol. 44, pp. 1261-1271, Oct. 1996.