Analysis of Structure and Performance of Turbo Codes over Additive White Gaussian Noise Channels: Practical Design and Implementation

A.Anglin-Jaffe and M.A.Ambroze

Fixed and Mobile Communications, University of Plymouth, Plymouth, UK
e-mail: M.Ambroze@plymouth.ac.uk

Abstract

This paper is concerned with the design and effectiveness of a family of error correction codes known as Turbo codes. Turbo codes are correction codes used to eliminate errors from communications networks. This family of code has error performance levels approaching the theoretical limit outlined by Shannon (1948). These codes are applied in the practical setting of error correction in deep space communication. This paper reviews the current literature associated with turbo coding, then outlines and evaluates a practical design scheme for a turbo code. Modifications to the code to provide better error performance are described and the results generated are discussed.

Keywords

Error Correcting Code, Convolutional Code, Turbo Code, S Interleaver

1 Introduction

In 1948, Claude Shannon (1948) proved that by properly encoding information, any errors caused by channel noise could be reduced to an arbitrarily small value, virtually eliminating the problem of noise as long as the rate of information was less than the capacity of the channel. This was the foundation of the field of Information Theory, and laid the groundwork for perfect communication over imperfect channels. However, Shannon's work was all theoretical. He gave no indication of the actual code that would provide these values. Since that time, researchers have been working towards providing codes that begin to approach this "Shannon limit" (Dolinar and Divsalar 1995; Berrou et al 1993). These codes have evolved from relatively simple algebraic codes, up to the most current limit approaching codes. These codes approach Shannon's limit, even at very high levels of noise interference (Vucetic et al. 2007).

The purpose of this paper is to evaluate turbo codes, one of the near capacity achieving types of code. Practical programming techniques are used to create a convolutional coding scheme in addition to a Turbo coding scheme and these methods are compared. Modifications to the Turbo code are described, which involve both changing the style of the interleaver and changing the component codes. An analysis follows as to how these changes affect the codes' performance.

2 Turbo Codes

In the early 1990's Berrou, Glavieux and Thitimajshima (1993) presented a paper on a new line of codes that produced error rates closer to the Shannon limit than ever before, with far less complexity than the codes being produced at the time. These "turbo codes" have an interesting encoder structure consisting of the parallel concatenation of two convolutional codes separated by an interleaver. This interleaver rearranges the data before it enters the second convolutional encoder. After transmission, the information is decoded by two separate iterative decoders using a-posteriori probability (APP) decoding, with an interleaver and a disinterleaver between them (Berrou et al. 1993). This process cycles with the "confidence" level of the system increasing with each circuit. This is known as belief propagation. This circular motion resembles the feedback system of a turbine engine, hence the name "turbo code".

It is pertinent to consider the advantages and disadvantages of the turbo code design. The encoder is simple and performs much better than a convolutional code with a similar decoding complexity (Costello Jr. and Forney 2007), and as stated previously, the code closely approaches the Shannon limit. However, one of the major disadvantages of turbo codes is the presence of an "error floor", where the performance of the code levels off for bit error rates (BER) below 10^{-5}. This is because turbo codes have relatively small minimum distances between code words, so the likelihood of confusing one codeword for another creates a limiting factor in the correction process (Costello Jr. and Forney 2007). This error floor can be lowered by improving the interleaver design (Vucetic et al. 2007), but not eliminated.

3 Research Methods

The aim of this research project, more than simply bringing a theoretical discussion of the advantages and disadvantages in turbo coding, was to construct a practical framework under which turbo codes operate, in order to examine real world results. Therefore data was generated that could validate or repudiate the arguments set forth previously. A test environment was designed which mimicked the channel structure set forth in Shannon's paper. It consisted of: an encoder; a channel which the encoded signals pass through; and a decoder to process received signals. The testing area had the capacity to function both as a convolutional encoder and a turbo encoder. It was one of the aims of this project to compare the functionality of these two types of encoding technique. The effects of a change in interleaver design on the coding efficiency of a turbo code was also studied, in addition to the effects of changing the memory length of its constituent codes.

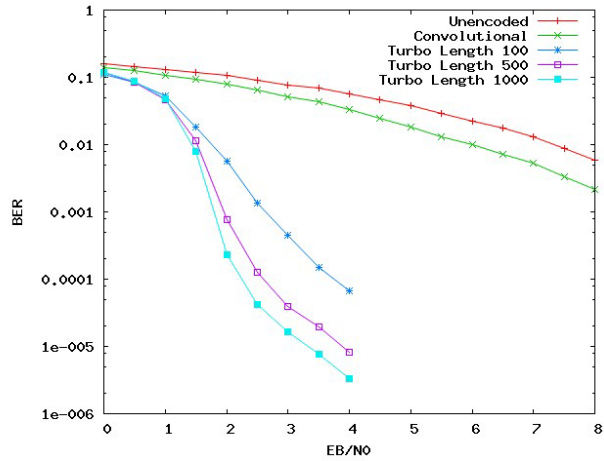

Figure 1: Bit Error Rates for Turbo Codes of Various Lengths Compared to Convolutional Codes and Unencoded Data

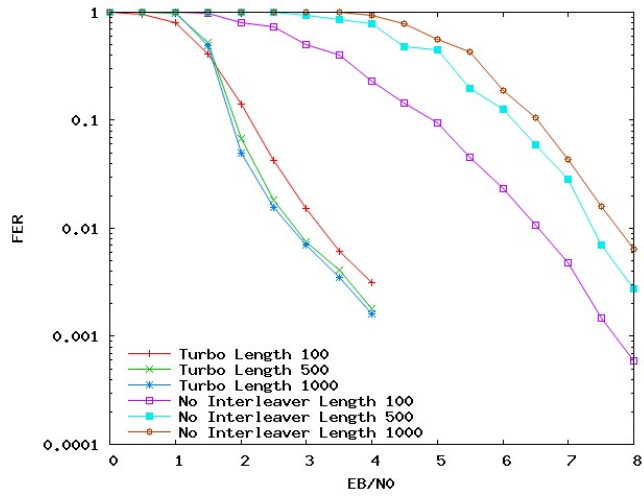

Figure 2: Frame Error Rates for Turbo Codes of Various Block Lengths and Code with No interleaving

4 Results and Analysis

Coding in the real world cannot yet hope to achieve Shannon limit levels of error correction. Each code behaves differently depending on circumstance, and as a result, there is a need to track many parameters to decide the best code to use for specific situations (Lin and Costello 2004).

The in graph in Figure 1 demonstrates the bit error rates for a number of different possible coding situations. The graph illustrates the relationship between turbo codes

and their constituent convolutional code. It also includes a baseline measurement. This measurement was collected by tracking the received values on the decoder side, and then evaluating their bit likelihood based on a simple positive or negative check. In this test, The turbo code significantly outperformed the convolutional code at low signal to noise ratios. Both codes outperform a simple positive or negative likelihood measurement, so it is clear that encoding the signals has an effect on signal reception.

However, the comparision between Convolutional codes and Turbo codes is uneven because Turbo codes transmit data at a rate of 1/3. This means an extra parity bit is sent in comparison to the convolutional codes. However, this can be corrected for by tracking data from a turbo code with no interleaver and no iterative process. This generates a 1/3 rate convolutional code. The previous data was generated to analyse bits in error. The frames (or blocks) in error were also evaluated. A frame is considered to be in error if a single bit in the processed block is in error. The comparison in Figure 2 shows turbo codes in comparison to their equivalent 1/3 rate convolutional codes (codes with no interleaver).

In addition to generating fewer bit errors, the turbo codes were more likely to transmit frames without error than equivalent rate codes. The coding gain here was very large, with a gain of 3-5 dBs even for very low amounts of frames in error. In addition the turbo code frame error rates improved with block length. The block length 500 code had an improvement of almost one full dB at rates below 10^{-2} in comparison to lengths of 100. In contrast, the non interleaved code had improved rates for smaller block lengths.

Changes in the turbo code affect code performance characteristics. Research suggests that modification of the interleaver can improve results in the “error floor” (Costello and Forney 2007; Vucetic et al. 2007). Dolinar and Divsalar (1995) set forth parameters for the creation of a pseudo-random interleaver which eliminates some of the problems of low weight encodings being transferred between the two component codes. Within this test environment a pseudo-random interleaver was created and programmed to function in the same place as the random interleaver. The resulting data is shown in Figure 3.

Both codes operated in a similar fashion for low signal to noise ratios. However, in the higher ranges of E_b/N_0, the pseudo-random code outperformed its random counterpart. However, in some cases, as in the block length 1000, it only slightly improved performance. The reason the interleaver design was specifically affecting the error floor area was because this floor represents the relatively small Hamming distances (the amount that two near codewords differ) between component codes being transferred to one another. With a random interleaver, at least some bit matchings were likely to be within a close range of one another. The S-random interleaver was supposed to correct for this, but in some cases the random interleaver did not have these performance problems, and therefore behaved as well as an S-random interleaver would.

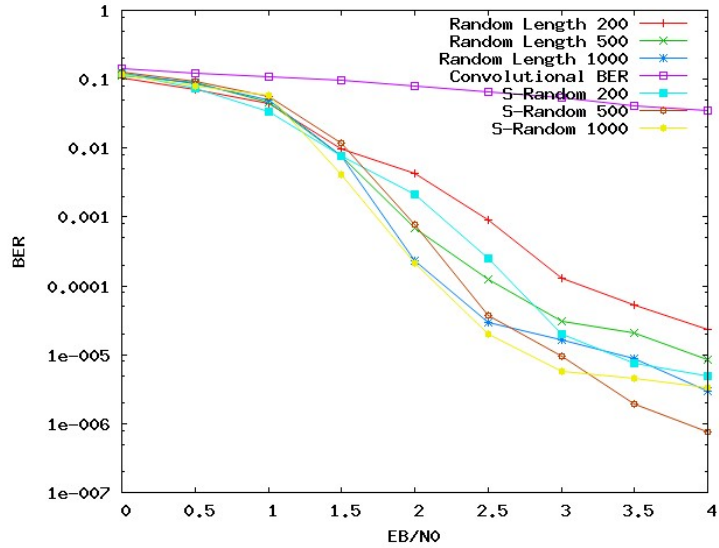

Figure 3: Bit Error Rates for Turbo Codes of Different Block Lengths and Varied Interleaving

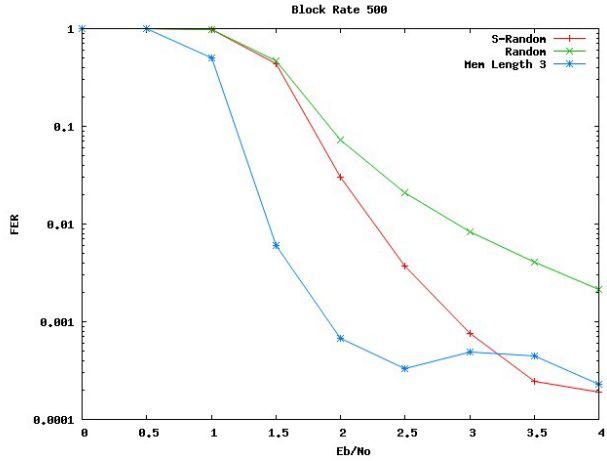

Figure 4: Frame Error Rate for Block Length 500 Turbo Codes with Random

A further significant element of turbo code construction is the design of the constituent code (Lin and Costello 2004). Initially, the experiment had employed a memory length 2 systematic recursive code. In the second stage this code was changed to a memory length 3 code, to evaluate if this produced an improvement. The code was also systematic and recursive, and had the octal notation of (15,13). It was one of the aims of this project to evaluate this code with the presence of both random and S-random interleavers as discussed above. The graph in Figure 4

evaluates the memory length 3 code with random interleaving, in comparison to the memory length 2 code with both the random interleaver and S-random interleaver.

From this chart it can be seen that the increased memory length did tend to improve the code performance. Even without the presence of a spread interleaver such as the S-random interleaver, it outperformed memory length 2 codes with the more advanced interleaver. The coding gain for these longer memory codes ranged from 0.5 dB to almost 1 full dB up to an E_b/N_0 of 2.5 dBs. After this point, the memory length 3 encoder performs poorly. This might be explained by the encoder encountering its error floor. The "waterfall" area for the memory length 3 encoder is much steeper than its counterpart memory length 2 codes. This means it reached error rates of less than 10^{-3} at 2 dBs. As can be seen by the S-random encoder, the memory length 2 encoder does not appear to slow its progress until it approaches the range of 3.5 dBs and greater. Another explanation for the performance of the memory length 3 code after 2.5 dBs may relate to programming difficulties. It is possible that somewhere in coded environment an overflow error occurred. As the memory length 3 code which was run in this experiment required such a large number of iterations, it is possible that there were too many iterations and a bit flip occurred to reset the frame numbers back to 0.

When the component code design change was combined with the interleaving design changes, the combination resulted in slightly better code performance, as well as a slightly lowered error floor. The longer length codes continued to outperform the shorter ones, giving extremely good bit and frame error rates at low SNRs. These codes had BERs of below 10^{-5} at signal rates of 1.5 dBs which is similar to results generated by other researchers (Dolinar and Divsalar 1995; Andrews et al. 2007). However, after 1.5 dBs the slope of the curve changes, as the turbo code entered its error floor area.

The performance of the S-random interleaver for low block lengths gave very marginal improvement. This is due to multiple factors. First, the choice of termination for the S-random interleaver created additional bit errors. Secondly, at lower block rates, the S-distance was smaller and thus could not guarantee improved Hamming distances.

4.1 Discussion of Results

The data suggest that at low memory lengths turbo codes significantly outperform their component convolutional codes, especially at low signal to noise ratios. At high memory lengths, convolutional code performance can be brought closer in line with that of the results here, but the decoding complexity is similarly increased (Divsalar and Pollara 1995). If the available signal to noise ratio is high, and the code required is short, it is possible that shorter convolutional codes may match turbo output. If a hard input-output algorithm was chosen, such as Viterbi decoding, their decoding complexity would be less than turbo codes.

The S-random interleaver created was terminated through a simple process, but one which unfortunately did have limitations. After running through all choices, if no possibilities existed outside the S range, the interleaver simply selected an unused bit

which was within the S-range and continued on. This could be responsible for the same type of bit errors that can be found with random interleavers, and also made the last bits in the stream (those most likely to run into seeding problems) less protected than other bits. This meant that the spread interleaver was not operating optimally and thus the results were closer to random interleaver performance.

The memory length 2 code was unfortunately too short for it to convey an effective relationship between its memory and its encodings. If more interleavers were added, and three memory length 2 codes or more were employed, these short codes might prove to be more effective, as demonstrated in Divsalar and Pollara (1995) and Andrews et al. (2007). However, the memory length 3 code performed much better in the presence of a single interleaver. In terms of practical coding in real world systems, the DVB-RCS standard specifies a code of memory length 3, (Douillard et al. 2000) which has a similar representation as the encoder employed in this project. In addition, while it appears that the modification of the code was a more significant change than the introduction of a new interleaving technique, it is more likely that this relates to the poor performance of a memory length 2 code and an already good performance of random interleaving. It is likely that further modification of the internal coding structure would produce less significant gains; however, further study is needed into the subject.

5 Conclusions

The initial hypothesis was that codes with designed interleavers and a longer memory length would outperform the original code with a length 2 memory and random interleaving. This proved to be the case, but not to the expected extent. As mentioned before, the improvement of the interleaver seemed to play less of a role in coding gain than other factors, such as increased block lengths and component code design. However, when taken against the same type of code with no interleaving, the performance gain was significant. Therefore, the interleaver must play an integral role in improving code performance. It must be concluded, then, that the lack of improved performance from designed interleavers can be attributed to design flaws, and thus these codes were not able to show further coding gain.

The aim of this research was to study the design and structure of turbo codes. In terms of the literature a lack of clarity remains as to how these codes manage to perform so well, or how they can be optimized (Andrews et al. 2007). The error floor can be lowered, but its existence still suggests a flaw in the code design. This experiment was carried out to analyse what components of turbo code design contribute to code improvement. It was found that a combination of increased component code memory length, as well as a pseudo-random interleaver, led to much better code performance. In addition, the longer block lengths performed much better than shorter ones, due to an increased length interleaver.

The field of turbo coding is wide-ranging and there are so many pieces of the codes' design that can be modified. This means that the scope for future research is extensive and can be fruitfully explored until the perfect code is found for all situations. Other types of spreading algorithms should be pursued and evaluated for performance. Of the interleaving types currently available, many perform very well

in lowering the error floor, but none have eliminated it completely (Vucetic et al. 2007). Surveying more interleaving techniques should lead to identifying the design flaws in the interleaver. The possibility of designing a “perfect” interleaver is still debatable, but analysis of each separate interleaving technique may result in more patterns for “good” interleaving. For example, analysis of row/column interleaving led to conclusions on random interleaving performance and introduced the concept of “spreading” (Divsalar and Pollara 1995). Other factors are yet to be discovered in relation to interleaver behaviour and thus may lead to further improvements in design.

These codes continue to be effective technologies to ensure that optimal communication is possible on noisy channels. Satellite technology would not be possible without these forms of error correction to overcome the interference levels which deep space environments impose on communications. Until the perfect code is found, these codes will continue to be improved on to ensure error free communication. The MESSENGER spacecraft, Mars Reconnaissance Orbiter, and New Horizons are all currently using turbo codes for the return of their data to earth. Improvement of these codes will mean that images transmitted from space will be able to be sharper and more detailed, and even opens the possibility of sending movies.

6 References

Andrews, K.S., Divsalar, D., Dolinar, S., Hamkins, J., Jones, C.R., and Pollara, F. (2007) “The Development of Turbo and LDPC Codes for Deep-Space Applications.” *Proceedings of the IEEE.* Vol. 95(11), 2142-2156 [Online] Available at: http://ieeexplore.ieee.org/.

Berrou, C.; Glavieux, A.; and Thitimajshima, P., (1993) "Near Shannon limit error-correcting coding and decoding: Turbo-codes. 1," *Communications, 1993. ICC 93. Geneva. Technical Program, Conference Record, IEEE International Conference on* , Vol.2, no., pp.1064-1070 [Online].

Costello Jr., D. J., and Forney Jr., G.D. (2007) “Channel Coding: The Road to Channel Capacity” *Proceeding of the IEEE.* Vol. 95(6), 1150-1177 [Online] Available at: http://ieeexplore.ieee.org/.

Divsalar, D. and Pollara, E. (1995) “Turbo Codes for Deep-Space Communications”, *JPL TDA Progress Report* 42-120.

Dolinar, S. and Divsalar, D. (1995) “Weight Distributions for Turbo Codes Using Random and Nonrandom Permutations”, *JPL TDA Progress Report* 42-122.

Douillard, C., Jezequel, M., Berrou, C., Brengarth, N., Tousch, J., and Pham, N., (2000) "The Turbo Code Standard for DVB-RCS", 2nd International Symposium on Turbo Codes and Related Topics", Brest, France.

Lin, S., and Costello, Jr., D.J. (2004) *Error Control Coding: Fundamentals and Applications 2nd Ed.* Upper Saddle River: Pearson Prentice Hall.

Shannon, C.E. (1948) “A Mathematical Theory of Communication” *Bell Syst. Tech. J.,* Vol 27 pp. 379-423 and 623-656.

Vucetic, B., Li, Y., Perez, L.C., and Jiang, F. (2007) “Recent Advances in Turbo Code Design and Theory” *Proceedings of the IEEE.* Vol. 95(6), 1323-1344 [Online] Available at: http://ieeexplore.ieee.org/.