

# Iterative Demodulation and Turbo Decoding for Distributed Radio Receivers <sup>\*</sup>

Preston A. Jackson, Joel I. Goodman, and Hector Chan

MIT Lincoln Laboratory  
244 Wood ST, Lexington, MA 02420  
{preston, jgoodman, chanh}@ll.mit.edu

## Abstract

This paper presents a novel iterative algorithm for demodulation and turbo decoding. Using this algorithm, short coded messages, which can be independently processed on a distributed computer, can achieve an improved bit-error rate for a small increase in computational load. We also present a trade study of the effects of message length, memory order, signal to noise ratio, and iterations in our iterative demodulator-decoder.

Keywords: turbo code, modulation, demodulation, iterative, parallel, distributed, high-performance.

## 1. Introduction

Turbo decoders dominate the computational requirements of radio receivers that utilize them. Unfortunately, turbo decoding has an inherently sequential structure which makes distribution of long messages to a multiprocessor difficult. One method to overcome this problem, is to send a series of short messages, or packets, to independent sequential processors. However, since message length affects the performance of the decoder, this method reduces performance. In this paper, we present a novel iterative approach to demodulation and decoding which decreases the performance penalty of a shortened message without a significant increase in computational requirements. This iterative approach improves bit-error rate (BER) in systems that distribute messages to multiple processors in a computationally efficient manner. We also perform a parametric trade-study of receiver performance by varying message length and memory order for various signal-to-noise ratios.

The outline of this paper is as follows. Section 2 presents the iterative demodulation-decode algorithm. Section 3 states the assumptions and parameters in our trade space. Section 4 describes some of the results of our trade study, and Section 5 draws conclusions.

## 2. Iterative Decoding Algorithm

Our baseline radio receiver consists of a demodulator, an interleaver, and a turbo decoder, as seen in Figure 1. The demodulator transforms a series of received symbols,  $r$ , into a series of soft decision bits using maximum likelihood (ML) ratios. The interleaver permutes the output of the demodulator to distribute the affect of the noise to nonadjacent bits. The turbo decoder, using the soft decisions from the demodulator, performs maximum *a posteriori* (MAP) decoding to produce a result  $y$ .



Figure 1. Receiver block diagram.

The information flow in the iterative radio receiver is shown in in Figure 2. The received information flows along the “Bootstrap Path” its first time through the receiver. When  $y$  is produced, a cyclic redundancy check (CRC) is tested for errors. When no errors are detected, the receiver continues processing the next received message. However, if errors are detected, the receiver follows the “Iterative Path,” in which the soft decisions from the MAP turbo decoder are fed into the demodulator and used as *a priori* information, turning the ML demodulator into a MAP demodulator. This process is iterated until no errors are detected or a maximum number of iterations is reached, after which the receiver proceeds to the next message.

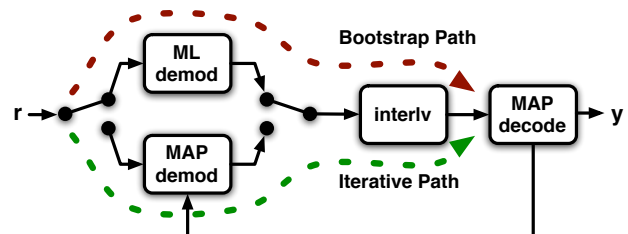


Figure 2. Iterative receiver block diagram.

<sup>\*</sup>This work was sponsored by the Navy under Air Force Contract FA8721-05-C-0002. Opinions, interpretations, conclusions and recommendations are those of the authors and are not necessarily endorsed by the Department of Defense.

### 3. Parametric Trade-space

In order to determine the performance of our iterative algorithm, we studied the affect of various parameters on the bit-error rate (BER) and the latency of the system. These two metrics are affected by six key factors:

- The signal-to-noise ratio, varied from 3.5 dB to 5 dB,
- S-random interleaver with  $S = \arg \max_s \{s < \sqrt{N/2}\}$ ,
- 32 processors,
- The message length,  $N$ , varied from 512 to 16384 bits,
- The states in the decoder, varied from 2 to 16, and
- The number of iterations in the iterative demodulation-decoding process, varied from 1 to 4.

### 4. Performance

The BER of the demodulator-decoder are presented in Figure 3 for various parameters. From this plot we see iterations can buy a factor of 3 improvement in BER. However, no number of iterations will approach the performance of a system with a twice the message length. We also see that the first iteration buys the most improvement, with iterations three and four fighting for marginally better performance.

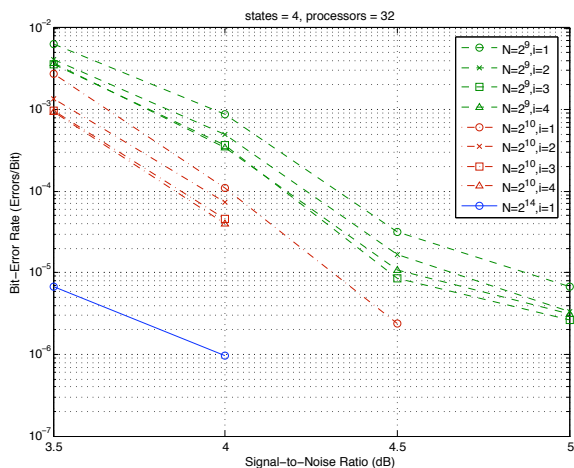


Figure 3. Bit-error rate vs. signal-to-noise ratio for a 4-state turbo decoder, running on a 32-processor parallel computer.

The normalized latency of the iterative demodulator-decoder can be seen in Figure 4. This plot shows us the tremendous improvement in latency that a parallel implementation can provide. In this plot, we have chosen the message length 16384 as our baseline, since it cannot utilize the multiple processors of our system. In contrast, the 512-bit packet size can effectively utilize all 32 processors. Another important note is that the multiple iterations do not significantly increase the latency because the iterations only run on those few packets which have errors.

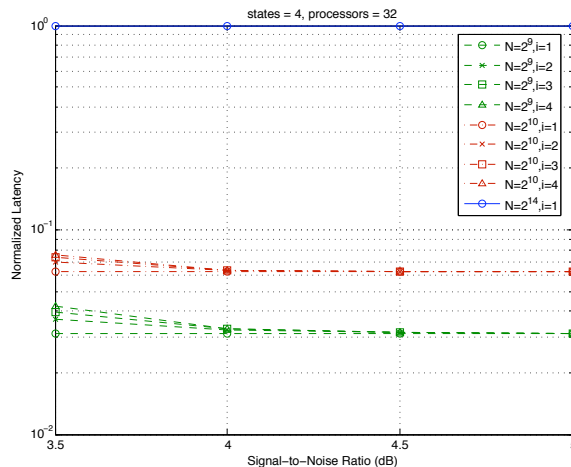


Figure 4. Normalized latency vs. signal-to-noise ratio for a 4-state turbo decoder, running on a 32-processor parallel computer.

### 5. Conclusion

In conclusion, we have presented a novel algorithm for iterative demodulation and turbo decoding. We have also performed a parametric trade-study of the turbo decoding algorithm targeted for parallel computing. From this study we have determined that doubling the message length significantly improves the BER of the system (factor of 10), but at a cost of increased the latency (factor of 2). The iterative demodulation-decoding algorithm provides a factor of 3 improvement in the BER over a system without iterations. Also, since the iterations are targeted on the few packets with errors, the additional latency for the algorithm is small.