Performance comparison of convolutional and block turbo codes

K. Ramasamy¹a), Mohammad Umar Siddiqi², Mohamad Yusoff Alias¹, and A. Arunagiri¹

¹ Faculty of Engineering, Multimedia University, 63100, Cyberjaya, Malaysia
² Faculty of Engineering, International Islamic University, Kuala Lumpur, Malaysia

¹a) k.ramasamy@mmu.edu.my

Abstract: Turbo code is recommended as a channel coding scheme, which has been shown to be capable of performing close to the Shannon Limit. In this paper, we compare the performance of both convolutional and block turbo codes over AWGN and Rayleigh fading channels. It is observed that the performance of convolutional turbo code is slightly better in the waterfall region and the coding gain ranges from 0.5 to 0.7 dB at BER of $10^{-4}$ depending on the channel conditions. But below BER of $10^{-4}$ or $10^{-5}$ an error floor occurs in the case of convolutional turbo code. Block turbo codes tend to outperform convolutional turbo codes for low BER.

Keywords: convolutional turbo code, block turbo code, waterfall region, error-floor region

Classification: Science and engineering for electronics

References

1 Introduction

Powerful error correcting code is essential to offer reliable data and multimedia service in the next generation mobile communication systems. Turbo code is known to have the most powerful error correcting capability up to now [1, 3]. Generally, turbo codes can be classified into two categories according to the type of constituent encoder such as convolutional turbo code and block turbo code or turbo product code. Convolutional turbo code consists of a parallel concatenation of Recursive Systematic Convolutional (RSC) encoders separated by a pseudo-random interleaver and turbo product codes are built from product of two systematic block codes, separated by a uniform interleaver [3]. Turbo decoders make use of very powerful soft-input/soft-output iterative decoding algorithms that are numerically intensive. It is shown that the performance of both is close to Shannon’s capacity bound. The performance of a turbo code may be affected more by different parameters of the component codes, block size, interleaver design and weight spectrum [1, 3]. The bit error rate (BER) curve of a turbo code is divided into two regions. The first region, called “waterfall region” in which the BER decreases rapidly at low signal-to-noise ratio (SNR) and second region, called “error-floor region” where the BER decreases at low rate at high SNR. In waterfall region, the performance depends on existence of low weight code words. Low weight code words will reduce the decoding convergence, thus the BER will decrease rapidly and the number of iteration required in decoding process is also reduced. The error-floor region occurs due to the presence of a few low weight code words. At low SNR, these code words are insignificant, but as SNR increases they begin to dominate the performance of the code [4, 5].

The paper is organized as follows: In Section 2, the principle of both convolutional and block turbo coding is described. The simulation results of convolutional and block turbo codes over AWGN and Rayleigh fading channels are compared in Section 3. In Section 4, the advantages of both codes are discussed. Conclusions are given in Section 5.

2 Turbo Codes

In this chapter, the principle of convolutional and block turbo codes are presented.

2.1 Convolutional Turbo Code

The convolutional turbo coder consists of a parallel concatenation of recursive systematic convolutional encoders separated by a pseudo-random interleaver [1]. The stream of input bits is fed to the first encoder without any modifications and is randomly interleaved for the second encoder. A natural rate for such a code is 1/3 (one systematic bit and two parity bits for one data bit). The rate can be relatively easy increased by puncturing the parity bits but reducing the rate under 1/3 is more difficult and may involve repetition of some bits, adding another RSC encoder or decreasing the rate of the component encoders.
The MAP criterion is used to provide a soft-output. For each decoded bit $u_k$, given the received symbol sequence $\vec{y}$, a-posteriori LLR over a Gaussian channel is given by

$$L(u_k|\vec{y}) = \log \left[ \frac{p(u_k = +1|\vec{y})}{p(u_k = -1|\vec{y})} \right]$$

$$= L(u_k) + L_{cy_k} + L_e(u_k)$$

(1)

where $L(u_k)$ is the a-priori LLR, provided by a component decoder, $L_c$ is the channel reliability measure, $2/\sigma^2$, $y_k$ is the received version of the transmitted systematic bit and $L_e(u_k)$ is the extrinsic LLR for the bit $u_k$. The optimal trellis-based iterative MAP decoding method presented in [2] is applied to decode. In iterative decoding, there are two SISO decoders where the extrinsic information obtained from the decoding process is exchanged between them.

The received signal from the wireless channel can be written as $r(t) = \alpha(t)S_i(t) + n(t)$, where $\alpha(t)$ is a Rayleigh process that satisfies $E(\alpha^2) = 1$. On the Gaussian channel $\alpha(t) = 1$. $S_i(t)$ is the modulated waveform for symbol $S_i$ and $n(t)$ is a Gaussian noise process with two-sided power spectral density $N_0/2$. The modulation method used is binary phase shift keying (BPSK).

### 2.2 Block Turbo Code

Block turbo code or turbo product code is a serial concatenation of linear block codes such as Hamming or BCH codes, and it is decoded iteratively by simple component decoders. In a 2-dimensional turbo product code, the information symbols are first encoded horizontally by a row-wise encoder. Then, they are encoded vertically by a column-wise encoder. This can be thought of as the interleaved version of horizontal one resulted from block interleaving [3]. The structure of the generated codeword in the encoder of block turbo code is shown in Fig. 1. Consider two systematic linear block codes $C^1$ and $C^2$ with parameters $(n_1, k_1, \delta_1)$ and $(n_2, k_2, \delta_2)$ respectively, where ‘n’ is the code length, ‘k’ is the number of information symbols and ‘\delta’ is the minimum Hamming distance. The product code, $P = C^1 \times C^2$ is obtained. We first place information bits in a $k_1 \times k_2$ matrix. Then we encode

![Fig. 1. Structure of codeword in a 2-dimensional block turbo code.](image-url)
the $k_1$ rows using $C^2$ and finally we encode the $n_2$ columns using $C^1$. The parameters of the resulting product code are given by $n = n_1n_2$; $k = k_1k_2$; $\delta = \delta_1\delta_2$ and code rate $R = R_1R_2$ [3]. The same trellis-based iterative MAP decoding method presented in [2] is applied to decode the block turbo code with extended hamming component codes.

3 Simulation results of convolutional and block turbo codes

The performance comparison between the convolutional turbo codes and block turbo codes over AWGN channel for smaller frame sizes of 90 bits and 56 bits with two code rates is shown in the Fig. 2(a). Constraint length, $K = 4$ and generator polynomial, $G = [13/11]$ in decimal are used for convolutional turbo code. Code shortening is used to obtain the desired frame size and code rate in turbo product code with extended hamming codes. From Fig. 2(a) it is observed that the performance of convolutional turbo code is slightly better in the water fall region. For these small frame sizes, the coding gain is 0.6 dB at BER of $10^{-4}$ in the water fall region. But below BER of $10^{-4}$ or $10^{-5}$ an error floor occurs in the case of convolutional turbo code. At low BER turbo product codes tend to outperform convolutional turbo code. From Fig. 2(b), it is observed that for larger frame sizes of 494 bits and 960 bits the convolutional turbo code continues to perform better than the turbo product code with both code rates and the coding gain is 0.5 to 0.7 dB.

![Fig. 2. Performance comparison of convolutional turbo code and turbo product code on AWGN channel with (a) smaller frame size, (b) larger frame size.](image)

The performance of convolutional turbo code and turbo product code is also compared over Rayleigh fading channel. Figure 3(a) shows that for smaller frame sizes of 56 bits and 90 bits convolutional turbo codes perform better than the turbo product code in the water fall region and the coding gain ranges from 0.5 to 0.6 dB. From Fig. 3(b), it is observed that even for larger frame sizes of 494 bits and 960 bits the convolutional turbo code continues to perform better than the turbo product code with both code
Fig. 3. Performance comparison of convolutional turbo code and turbo product code on Rayleigh fading channel with (a) smaller frame size, (b) larger frame size.

rates and the coding gain ranges from 0.5 to 0.6 dB. But the error floor occurs much deeper for turbo product code compared to convolutional turbo code.

4 Advantages of turbo codes

In this chapter, the advantages of both convolutional and block turbo codes are discussed.

4.1 Advantages of convolutional turbo code

(i) Simple implementation

In addition to better performance in the waterfall region, convolutional turbo codes have distinct implementation advantages over block turbo codes in a multi-rate variable frame size system. This is because convolutional turbo codes use a single constituent code for all frame sizes and all code rates. Simply changing the interleaver parameters supports different frame sizes. Changing the puncturing matrix changes the code rate. Turbo product codes require multiple constituent codes to provide a reasonable variety of code rates and frame sizes and no puncturing is required.

(ii) Compatibility with IEEE 802.11 rates

Convolutional turbo codes can operate at virtually any code rates, but the most natural are rates such as 1/2, 2/3, 3/4 which are compatible with IEEE 802.11. But the code rates and frame sizes of turbo product code are not compatible with IEEE 802.11.

(iii) Reduced memory

The required memory size for convolutional turbo code is less than that of turbo product code because 3-dimensional turbo product code requires three sub iterations per iteration, and is therefore slower and requires twice as much extrinsic information as 2-dimensional codes. Memory makes up the largest portion of a turbo decoder circuit. Convolutional turbo codes have been
adopted by the most widely used standards in the world including WCDMA and CDMA2000.

4.2 Advantages of block turbo code
Turbo product codes are very flexible in terms of performance complexity and code rate. Constituent codes can be mixed and matched to achieve desired code characteristics. They can support any block size, a very wide range of code rates from below rate 1/3 to as high as rate 0.98. Code shortening enhances this flexibility. Turbo product codes provide excellent performance at high code rates and can offer a wide range of block sizes and code rates with change in coding strategy. No puncturing is required. Since turbo product codes are multi dimensional codes, there are no low weight code words and no error floor while convolutional turbo codes are found to have an error floor of approximately $10^{-5}$. So turbo product codes are suited to applications where latency (and hence interleaver size) is limited, but error floor would be unacceptable because of BER requirement. Further, block codes have decoders that can operate at very high speeds, which has lead to ASIC decoders. Turbo product codes are available as standard products and as licensable cores.

5 Conclusions
The simulation results show that both coding schemes have good performance and flexibility to be applied for WLAN applications. Turbo product codes tend to outperform convolutional turbo codes for low BER, while convolutional turbo codes outperform turbo product code in the water fall region. The coding gain in the water fall region ranges from 0.5 to 0.7 dB at BER of $10^{-4}$ depending on the channel conditions. So it is important to well define the BER requirement in order to make a proper code selection. The significant coding gains are only applicable over a small region on the performance curves. Other concerns, such as latency and low cost parts availability, may be of greater concern than the small performances differences at a particular $E_b/N_0$ operating point. The turbo product code technology is currently in production from multiple vendors. Since the block codes are based upon standard constituents with no randomized components (interleavers for example), interoperability already exists. The convolutional turbo code community is not yet as stabilized and settled upon a common interleaver or even constituent codes.