# Optimization of the Interleaver Structure for Turbo-codes

**Amir K. Khandani***
Dept. of Elect. and Comp. Eng., University of Waterloo, Waterloo, ON

**Abstract — This work discusses the effect of the interleaver optimization on the performance of Turbo codes for large block lengths, $N \to \infty$. It is shown that for $N \to \infty$, the weight of the systematic and parity check sequences, denoted by $w_1$, $w_2$, $w_3$, respectively, has a Gaussian distribution. To optimize the code performance, it is desirable that the corresponding correlation coefficients, denoted as $\rho_{ij}$, $i,j = 1,2,3$, to be as small as possible. It is however shown that: $\rho_{ij} > 0$, $i,j = 1,2,3$, (ii) $\rho_{12}, \rho_{13} \to 0$ as $N \to \infty$, and (iii) $\rho_{23} \to 0$ as $N \to \infty$ for almost any random interleaver. This means that for $N \to \infty$, the optimization of the interleaver has no effect on the Turbo-code performance and in this case, the weight of the three coded streams will be independent of each other.**

**For shorter block lengths, a method is presented to optimize the structure of a Turbo-code interleaver via a deterministic algorithm. The resulting interleaver offers a substantial improvement in the error performance over a random interleaver and a noticeable improvement over an S-random interleaver (which is believed to offer the best performance among the previously known interleavers). In addition, the proposed method allows for an efficient representation of the interleaver structure and can be used to dynamically reconstruct the interleaver array with a low computational complexity. This is a useful property in multi-rate systems where several interleavers of different block lengths may be needed. The proposed method also provides: (i) a systematic method to search for good interleavers, and (ii) a common language to describe the results of such findings.**

## 1 Introduction

The advent of Turbo codes [1] is perhaps the most important development in coding theory in many years. These codes can achieve near Shannon-limit error correcting performance with a relatively simple decoding method. The basic idea of Turbo-codes is to make use of some Recursive Convolutional Codes (RCC) which are connected in parallel through pseudo-random interleavers. Note that as the RCC's and also the interleaver have linearity property[1], the resulting code is linear[2], and consequently, group property and distance invariance property hold.

Figure 1 shows the block diagram of the encoder of a rate 1/3 Turbo-code composed of two RCC's, where $b_1(m)$ is the systematic bit, and $b_2(m)$, $b_3(m)$ are the parity check bits. The weight of the code in Fig. 1 is equal to the sum of the weights of the $b_1(m)$, $b_2(m)$ and $b_3(m)$ sequences over a block, which are denoted by $w_1$, $w_2$, and $w_3$, respectively.

We present results for the performance of Turbo codes for large block lengths, $N \to \infty$. It is shown that for $N \to \infty$, the weight of the systematic and parity check sequences, denoted by $w_1$, $w_2$, $w_3$, respectively, has a Gaussian distribution. To optimize the code performance, it is desirable that the corresponding correlation coefficients, denoted as $\rho_{ij}$, $i,j = 1,2,3$, to be as small as possible. It is however shown that: (i) $\rho_{ij} > 0$ for $i,j = 1,2,3$, (ii) $\rho_{12}, \rho_{13} \to 0$ as $N \to \infty$, and (iii) $\rho_{23} \to 0$ for $N \to \infty$ with probability one (for almost any random interleaver). This means that for $N \to \infty$, the optimization of the interleaver has no effect on the Turbo-code performance. In this case, the weight of the three coded streams will be independent of each other.

We also present a method to optimize the structure of the Turbo-code interleaver for smaller block lengths. The basic idea in our approach is to use a linear recursion to describe the permutation group of a base interleaver which is subsequently adjusted to minimize the frequency of the low weight error events in a Turbo-code structure. The proposed method has the following desirable features:

- It offers a substantial improvement over a random interleaver and a noticeable improvement over an S-random interleaver (an S-random interleave is generally believed to offer the best performance among the previously known interleaver structures).

- It is generated using a deterministic algorithm with a low complexity which enables us to store the whole interleaver structure using a few parameters and regenerate the same structure whenever needed. The ability to efficiently store the interleaver structure is useful in multi-rate communication systems where several interleaver structures are needed. Such a deterministic description using a small set of parameters also has the advantage of providing a unified framework for different researchers to describe the interleaver structures used in their studies.

---

*This work has been supported by Communications and Information Technology Ontario (CITO).

[1] The effect of interleaving is equivalent to multiplying the input sequence by a permutation matrix which corresponds to a linear operation.

[2] This is based on neglecting the effect of the possible non-linearity caused by the method used to terminate the trellis.

## 2 Interleaver optimization for $N \to \infty$

We assume that the RCCs are generated by the transfer function $G(d) = N(d)/D(d)$. Using the results of [3], we know that the impulse response of $G(d)$ is periodic with period $p \leq 2^r - 1$ where $r$ is the memory length of the code. We are mainly interested in the group structure and also the periodicity property of the impulse response of $G(d)$. In this respect, we limit our attention to the structure of $D(d)$. This does not result in any loss of generality because the group structure and also the periodicity property of the impulse response of $G(d)$ is not affected by the choice of $N(d)$.

In general, we would like the period of the impulse repose of $G(d)$ to be as large as possible. If the period is equal to $2^r - 1$, the resulting impulse response is called a maximum length sequence (MLS). The rules to determine all the possible configurations of $D(d)$ to obtain a maximum length sequence of period $2^r - 1$ (for given $r$) are given in [3]. It can be shown that any MLS-sequence satisfy the three randomness postulates [3]. One consequence of this property is that in any period of an MLS-sequence, the number of ones is equal to $2^{r-1}$ and the number of zeros is $2^{r-1} - 1$.

If we look at the impulse response of $D(d)$ as a periodic sequence (started infinite in the past), we obtain $K = 2^r - 1$ non-zero sequences which are time shifts of each other. Each sequence corresponds to a specific positioning of the impulse within the period. We refer to these sequences as different phases of the periodic signal. We assume that different phases are labeled by integer numbers, say $1, \ldots, K$, where the label of a phase corresponds to the relative position of the corresponding impulse within the period. It can be shown that the set of phases of an MLS-sequence (plus the all-zero sequence) constitute a group under binary addition [3]. The order of each element in this group is equal to two, meaning that the sum of each phase with itself results in the all-zero sequence (denoted as the zero phase).

For bit position $k = 1, \ldots, N$ within the $j$th output stream, we refer to the set of bit positions $i \leq k$ for which an impulse at position $i$ results in a 1 at position $k$ as $\mathcal{R}_j(k)$, $j = 1, 2, 3$. Obviously, $\mathcal{R}_1(k) = \{k\}$. If the bit position $k$ is located in the $M$th period, i.e., $M = \lceil k/p \rceil$, then the number of positions within each of the periods $1, \ldots, M-1$ which belong to $\mathcal{R}_j(k)$, $j = 2, 3$, is equal to $2^{r-1}$. The number of positions within the $M$th period (the period containing $k$ itself) depends on the relative position of $k$ within the $M$th period and also on the numerator of $G(d)$. We are mainly interested in large values of $M$ for which the effect of the elements within the $M$th period itself is negligible. This means $|\mathcal{R}_2(k)|, |\mathcal{R}_3(k)| \simeq \lceil k/p \rceil 2^{r-1}$ where $|.|$ denotes the cardinality of the corresponding set.

We use the notation $b_i(m)$, $i = 1, 2, 3$, $m = 1, \ldots, N$, to refer to the $m$th bit within the $i$th output stream. We have $\overline{b_i(m)} = \overline{b_i^2(m)} = 1/2$.

We use the notation $P_w$ to refer to the probability of error between two code words which are different in $w$ input bits. We limit our discussions to the cases that the overall probability of bit error has a non-zero value. Under this assumption and for $N \to \infty$, the effect of the $P_w$ for finite values of $w$ on the overall bit error probability tends to zero. We disregard such error events from consideration. In other words, we limit our attention to the input sequences for which $w \to \infty$ as $N \to \infty$.

In the following, we first show that $w_1$, $w_2$ and $w_3$ have a Gaussian distribution. This is easily verified noting that all the $2^N$ possible combinations within the three steams are allowed, and consequently, the positions within the three output streams are iid binary random variables (where 0 and 1 are equally probable). Using the Central Limit Theorem, we conclude that $w_1$, $w_2$ and $w_3$ which are the sum of $N$ iid random variables have a Gaussian distribution with mean $N/2$ and Variance $N/4$. In this case, to show that $w_1$, $w_2$ and $w_3$ are independent, we need to show that the corresponding correlation coefficients $\rho_{ij}$, $i, j = 1, 2, 3$, tend to zero as $N \to \infty$, where $\rho_{ij} = (4/N)(\overline{w_i w_j} - \overline{w_i}\ \overline{w_j}) = (4/N)[\overline{w_i w_j} - (N/2)^2]$. We have $\overline{w_i w_j} = \sum_m \sum_n \overline{b_i(m) b_j(n)}$ where the expectation is over all possible $2^N$ combinations of the input. The overall weight of the output sequence is equal to $w_t = w_1 + w_2 + w_3$ which has a Gaussian distribution of mean $3N/2$ and variance $N(3 + 2\rho_{12} + 2\rho_{13} + 2\rho_{23})/4$. Noting that sequences with a weight smaller than the mean value result in larger probability of error as compared to sequences with a weight larger than the mean, we conclude that the main objective in the code design is to sharpen the peak of the PDF of $w_t$ which is equivalent to minimizing its variance. This is in turn equivalent to minimizing the $\rho_{ij}$ coefficients. In the following, we first show that the $\rho_{ij} > 0$, so the minimum possible value for each of them is zero. Noting the properties of the Gaussian distribution, this is equivalent to having independence between the coded streams.

**Theorem:** We have $\rho_{ij} > 0$ for $i, j = 1, 2, 3$.

Any of the pairs $b_i(m), b_j(n)$, $i, j = 1, 2, 3$, $m, n = 1, \ldots, N$, can take four different values, namely $\{00, 01, 10, 11\}$. The set of the input sequences resulting in the value of 00 form a sub-group of the group of all possible $2^N$ input combinations. This is a direct consequence of the linearity, and consequently, the group property of the code. Noting the group property of the set of corresponding coset leaders, two cases may happen. There is either only one coset with the coset leader 11, or there are three cosets with the cosets leaders 01, 10 and 11. The important point is that in both of these cases, using the basic results of the group theory, we conclude that the 00 sub-group and its cosets contain the same number of input sequences. This means that for the probability of the pair $b_i(m), b_j(n)$, we have the following two cases:

*Case I*: $b_i(m), b_j(n)$ take the values $00, 11$ each with probability $1/2$, resulting in $\overline{b_i(m)b_j(n)} = 1/2$.

*Case II*: $b_i(m), b_j(n)$ take the values $00, 01, 10, 11$ each with probability $1/4$, resulting in $\overline{b_i(m)b_j(n)} = 1/4$.

The important point is that in both cases, we have $\overline{b_i(m)b_j(n)} - \overline{b_i(m)}\ \overline{b_j(n)} > 0$. This means that the correlation coefficients $\rho_{ij}$, $i, j = 1, 2, 3$ are always positive.

**Theorem:** We have $\rho_{12}, \rho_{13} \to 0$ as $N \to \infty$.

For $\rho_{12}$ and $\rho_{13}$ (interaction of the systematic stream with each of the parity checks), Case II of the above two cases is valid resulting in $\rho_{12}, \rho_{13} \to 0$ for $N \to \infty$. Note that $b_1(m)$ and $b_2(n)$ will be independent of each other if $b_1(m)$ is not mapped (through interleaving) to a bit position within $\mathcal{R}_2(n)$, or otherwise, if $\mathcal{R}_2(n)$ contains at least two elements. This will be always valid unless for some trivial cases which will have a vanishing effect on the overall result.

**Theorem:** We have $\rho_{23} \to 0$ for $N \to \infty$ with probability one (for almost any random interleaver).

If $\mathcal{R}_2(m)$, $\mathcal{R}_3(n)$ contain at least one bit which are different from each other. then $b_2(m)$ and $b_3(n)$ will be independent of each other. This results in $\overline{b_2(m)b_3(n)} = \overline{b_2(m)}\ \overline{b_3(n)} = 1/4$. This will be the case unless $m = n$ and the elements of $\mathcal{R}_2(m)$ and $\mathcal{R}_3(m)$ contain the same input bits (before and after interleaving). This means that the corresponding interleaver has restriction on the mapping of infinitely many bit positions. Obviously, the faction of such interleavers tend to zero as $N \to \infty$. This means that for almost any random interleaver, we have $\rho_{23} \to 0$ as $N \to \infty$.

## 3 Dynamic generation of interleavers

The best interleaver structure currently known are the so-called S-random interleaver [2] which offer a large improvement in the bit error performance over a random interleaver. These are based on performing the following admissibility test in the process of *random* assignment of the entries in the interleaver array (S-rule): *either the difference between two indices, or the difference between the values assigned to those indices should be greater than a given threshold, say $S$.* One can also impose other admissibility conditions, including the elimination of low weight error events caused by the weight two input sequences.

In spite of their good performance, the S-random interleavers have the shortcomings that they can not be efficiently represented, nor easily reproduced. In the following, we present a method to optimize the structure of the Turbo-code interleaver via a deterministic algorithm of a low complexity which overcomes these shortcomings, and at the same time results in a noticeable improvement in the overall bit error performance. The proposed method allows for a compact representation and can be used to dynamically reconstruct the interleaver with a low computational complexity. The possibility of having an efficient storage and an easy dynamic reproduction for the interleaver is useful in multi-rate systems where several interleavers of different block lengths may be needed. In addition, such representation provides a unified framework for different researchers to search for good interleavers and/or to describe the interleaver structure used in their studies.

The basic idea in our approach is to use a linear recursion to produce an initial interleaver which is subsequently optimized by pair-wise exchange of its elements. The procedure starts from the first element of the initial interleaver and proceeds forward through the array. At step $c$, $c = 1, \ldots, N-1$, the suitability of the element indexed by $c$ is verified by checking its preceding elements bounded with windows of certain lengths. In case that the suitability conditions are not satisfied, the current element (indexed by $c$) is switched with the first element among its subsequent positions (indexed by $i = c+1, \ldots, N-1$) which satisfies the suitability conditions, and the whole procedure repeats from the next element in the array.

We make use of two suitability tests performed within two windows of lengths $S_1$, $S_2$. The first one is similar to the test performed in the design of the conventional S-random interleavers and makes sure that the element mapped to the current position has a distance of at least $S_1$ to all the elements mapped to its preceding positions within a window of size $S_1$. The second test is designed to break the weight two error events of a small span. This test makes sure that there is no weight two error event within a window of size $S_2$ prior to the current position. Numerical results are presented showing that the resulting interleaver offers a substantial improvement in the error performance over a random interleaver and a noticeable improvement over an S-random interleaver.

## 4 Algorithm

Motivated by the most common method of generating random numbers, namely the linear congruential method [4], we take advantage of the following recursive equation to generate a unique permutation of the set of integers $\{0, \ldots, N-1\}$, where $N$ is the code-block length,

$$X_{m+1} = (aX_m + b) \bmod N. \tag{1}$$

To guarantee that the above recursion results in a permutation of the set of numbers $\{0, \ldots, N-1\}$, we require that [4],

- $b$ is relatively prime to $N$
- $c \equiv a - 1$ is a multiple of every prime $p$ dividing $N$
- $c$ is a multiple of 4, if $N$ is a multiple of 4

Note that the above recursion is completely described in terms of $a, c$ and the starting value of $X_m$, namely $X_0$ (known as the seed of the random number generator). We assume that the permutation corresponding to the values of $X_m$, $m = 0, \ldots, N-1$, are stored in an array $I(.)$. To select the element $c$ of the interleaver, $c = 1, \ldots, N$, when the first $c-1$ elements are already fixed, we perform the tests:

- **Test I:** $|I(c) - I(i)|$ should be larger than $S_1$ for $i = c - S_1, \ldots, c-1$.
- **Test II:** $|I(c) - I(i)|$ and $|c - i|$, $i = c - S_2, \ldots, c-1$, should not simultaneously divide the period $P$.

If one of these two conditions is not satisfied, then starting from $i = c+1$ and incrementing $i$, we find the first element $I(i)$, $c+1 \leq i \leq N-1$ in the interleaver array which satisfies the above conditions and exchange the contents of the positions $c$ and $i$, i.e., $I(c) \leftrightarrow I(i)$, where $\leftrightarrow$ means pair-wise exchange.

The design algorithm explained earlier is implemented and used it in conjunction with a rate 1/3 Turbo-code where the underlying RCCs are of $m = 3$ memory elements and have the generator polynomial $(g_{feedback}, g_{forward}) = (13, 15)$. The code block length is equal to 576 and the number of iteration (using BCJR algorithm [5]) is 5. The sizes of the windows $(S_1, S_2)$ are selected as $(16, 27)$ and the base interleaver is initialized with the value of 13 (seed), i.e., $I(0) = 13$. The values of $a, b, X_0$ in (1) are selected as $13, 13, 13$, respectively. Figure (2) shows the error probability of a random, S-random ($S = 12$) and optimized interleaver.

## 5 Summary

It is shown that for large block lengths, $N \to \infty$, the optimization of the interleaver has no effect on the Turbo-code performance (for an error probability bounded away from zero). For shorter block lengths, a method is presented to optimize the structure of a Turbo-code interleaver which results in a substantial improvement over a random interleaver and a noticeable improvement over an S-random interleaver. In addition, the proposed method allows for an efficient representation and can be used to dynamically reconstruct the interleaver with a low computational complexity.

## References


[1] C. Berrou, A. Glavieux, and P. Thitimajshima, *Near Shannon Limit Error-Correcting Coding and Decoding: Turbo-codes, IEEE Int. Conf. on Comm. 1993 (ICC'93)* (Geneva, Switzerland), pp. 1064—1070, May 1993.

[2] D. Divsalar and F. Pollara, "Turbocodes for PCS Applications," *Proc. IEEE Int. Conf. on Comm.*, June 1995, pp. 54—59.

[3] S. W. Golomb, "Shift register sequences," San Francisco, Holden-Day, 1967.

[4] D. E. Knuth, "The Art of Computer Programming, Vol.2: Semi-numerical Algorithms", MA: Addison-Wesely, 1981.

[5] L. Bahl, J. Cocke, F. Jelinek, and J. Raviv, "Optimal decoding of linear codes for minimizing symbol error rate," *IEEE Trans. on Inform. Theory,* vol. 20, pp. 284-287, March 1974.


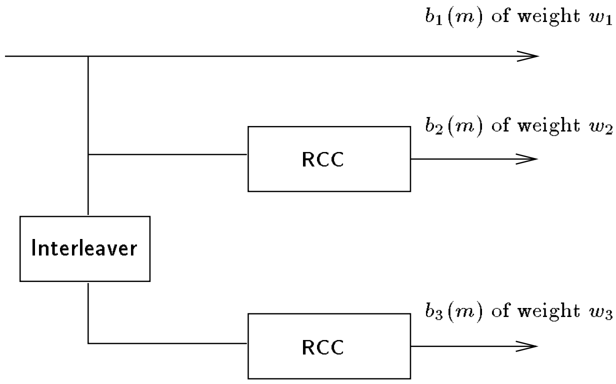


Figure 1: Basic structure of the Turbo encoder.

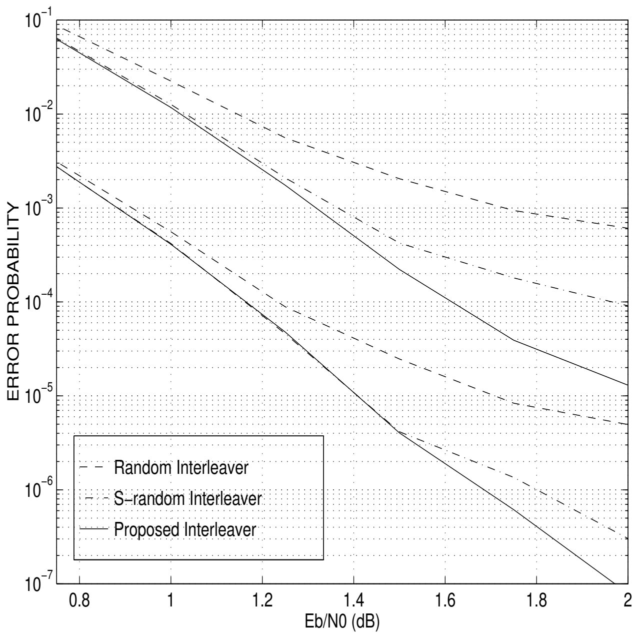


Figure 2: Bit error performance of a random vs. optimized interleaver.